Getting Started with Sun™ ONE

By Stacy Thurston

Publisher: Prentice Hall PTR
Pub Date: March 17, 2003
ISBN: 0-13-089390-0
Pages: 384

Direct from Sun, this is your step-by-step guide to delivering high-value solutions based on Sun ONE technologies. Sun ONE technical specialist Stacy Thurston brings together Sun's best practices for planning, designing, configuring, integrating, and implementing Sun ONE Internet infrastructures. Drawing on his immense experience supporting Sun ONE customers, Thurston offers new insight into the Sun ONE Application Server, Directory Server, and Web Server, and expert guidance on building business applications with Sun ONE Studio 4. Coverage includes:

- Installing and configuring the Sun ONE Web Server to serve the specific requirements of your environment
- Delivering dynamic content with CGI, Java(tm) servlets, and JSPs
- Administering Sun ONE Directory Server databases
- Configuring and deploying applications on the Sun ONE Application Server
- Using Sun ONE Studio 4 to develop high-performance Java applications and Web applications
- Establishing efficient data flows, application designs, and business information architectures

Getting Started with Sun ONE is an indispensable resource for everyone considering or utilizing Sun ONE products: analysts and decision-makers planning new systems; project managers deploying Internet infrastructure; and developers creating prototypes and delivering production-quality systems.

• Table of Contents

Getting Started with Sun™ ONE

By Stacy Thurston

Publisher: Prentice Hall PTR
Pub Date: March 17, 2003
ISBN: 0-13-089390-0
Pages: 384

PREVIOUS NEXT

Copyright

For information regarding corporate and government bulk discounts please contact: Corporate and Government Sales (800)382-3419 or corpsales@pearsontechgroup.com. Or write: Prentice Hall PTR. Corporate Sales Dept., One Lake Street, Upper Saddle River, NJ 07458.

Sun Microsystems Press Publisher: Myrna Rivera

Executive Editor: Greg Doench

Marketing Manager: Debby vanDijk

Manufacturing Buyer: Alexis Heydt-Long

Cover Design director: Jerry Votta

Cover Design: Talar A. Boorujy

Compositor: HuraDesign, Inc.

Editorial/Production Supervision: BooksCraft, Inc.

Full-Service Production Manager: Anne Garcia

10 9 8 7 6 5 4 3 2 1

Sun Microsystems Press

A Prentice Hall Title

Dedication

Dedicated to May, Branden, and Riley.

Also dedicated to my friends in the computer industry: Don Campbell, Jerry White, Curtis Materi, Mike Flynn, Shawn Price, Brian Kelly, Paul Charles, Tim Tai, Bernie Chisholm, Mary O'Connor,

and to Richard Murray for giving me my first computer job.

About Prentice Hall Professional Technical Reference

With origins reaching back to the industry's first computer science publishing program in the 1960s, and formally launched as its own imprint in 1986, Prentice Hall Professional Technical Reference (PH PTR) has developed into the leading provider of technical books in the world today. Our editors now publish over 200 books annually, authored by leaders in the fields of computing, engineering, and business.

Our roots are firmly planted in the soil that gave rise to the technical revolution. Our bookshelf contains many of the industry's computing and engineering classics: Kernighan and Ritchie's *C Programming Language*, Nemeth's *UNIX System Adminstration Handbook*, Horstmann's *Core Java*, and Johnson's *High-Speed Digital Design*.

PRENTICE HALL PTR

PH PTR acknowledges its auspicious beginnings while it looks to the future for inspiration. We continue to evolve and break new ground in publishing by providing today's professionals with tomorrow's solutions.

Foreword

Network computing is the heart and soul of Sun Microsystems. Always has been; always will be. I've been saying it for years, but I'll say it again: Our aim is to connect anyone, anywhere, anytime—using virtually anything—to the resources they need.

While some companies change directions more often than I change the oil in my car, you can count on Sun to keep delivering a clear, consistent vision. Sun ONE is Sun's software strategy and it includes products that are open standards–based software that provide information, data, and applications to anyone, anytime, anywhere, on anything. These products are taking Sun in the direction to become one of the largest software organizations in the industry.

I believe the opportunities ahead are enormous. For years people have been amazed at the increases in microprocessor speed, which has doubled every 24 months since about 1975. Well, bandwidth has been going up at an even faster rate—doubling, on average, every 16 months in the same time frame. The globe is being wrapped in fiber-optic cable, dotted with relay antennas, circled by satellites. And our Net-based products and technologies have been catching on in a big way.

Sun, with a proven ability to build reliable, scalable solutions, will play a big part in delivering the goods. Everything Sun does, everything we've always done, is about open network computing. Sun ONE is another part of the engine driving us forward.

Sun has the technology. It's affordable. First, last, and always, network computing is what we do.

Scott McNealy
Chief Executive of Sun Microsystems

Preface

The core Internet servers that make up most of the World Wide Web www) are the web server, the directory server, and the application server. This book describes these servers and explains how to start using them.

This book shows you how to set up and create web sites and web site, applications as well as how to administer a directory server (a specialized database). These servers are the basic infrastructure for corporate web applications and Internet Service Providers (ISPs). This infrastructure works for both personal web sites (at home or in the office) and company web sites or company web applications.

To make the learning experience faster, easier, and more complete, I conclude descriptions of Internet web site designs, with many illustrations and example configurations. I include information about installing, configuring, testing, and **using**

- The Sun Open Net Environment (ONE) Web Server
- The Sun ONE Directory Server
- The Sun ONE Application Server
- The Sun ONE Studio

The web server and application server are for hosting web sites and applications; the directory server manages data; Sun ONE Studio is for developing web site applications. Because these are core Internet products, you should gain core Internet skills!

You should find the server configurations in this book instructive and useful. Practical exercises will help you learn by doing.

This book is written for computer administrators and developers who need to set up and configure prototypes, proof of concept, and production systems. Business and systems analysts can read about sample designs and example applications that work over the Internet while project managers can get a sense of which tools and skill sets are required to set up a web site infrastructure.

A "How to" Book

This book's success will be measured by the skills you gain from reading it. In preparing the manuscript, I had a person test the instructions for using the Sun ONE Studio to create a web application who had never used the Sun ONE Studio before. The results were very positive—he commented that he is now comfortable creating web applications and wants to try doing more things with the Sun ONE Studio product.

One of the book's highlights is the section on the Sun ONE Directory Server. It has chapters on working with the directory data and the directory schema (data structure). This is an important topic, and yet I have not found another source that covers it as completely.

For administrators and developers who work with the servers, I have included instructions on installing and configuring servers. In the section on the Sun ONE Application Server, you will see how, once the server is installed, you can create web applications in a matter of minutes. I have made these chapters as complete as possible—you start with a computer, install software, configure it, test it, and then use it. Once you start to use it, you will continue to gain more skills with the major services of the Internet—skills such as setting up a web site.

I have tested the material in the book using Microsoft Windows 2000 and the Solaris 8 Operating Environment, UNIX. On MS Windows NT, I have tested the Sun ONE Web Server and Sun ONE Directory Server. On Linux, I have tested the Sun ONE Web Server and Sun ONE Directory Server.

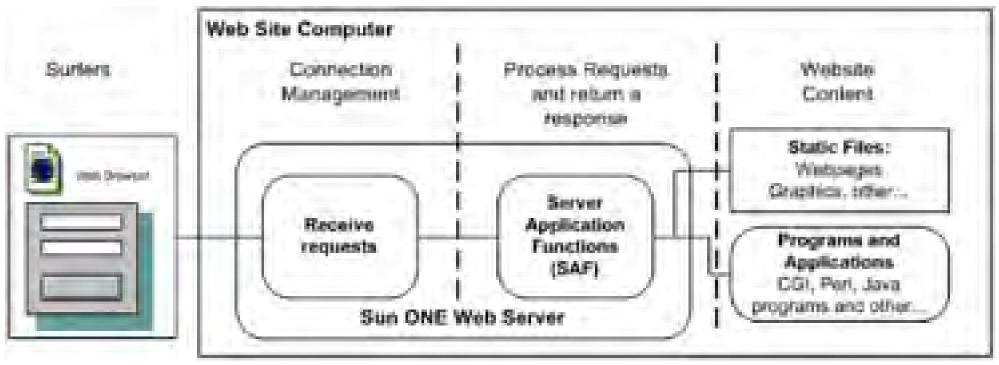
For a list of operating systems that the servers in this book will run on, go to the Sun web site, www.sun.com.

Building a Web Site

Everything is included with this book—software and instructions—for building a web site. Setting up a web site, includes the following steps (see Figure P.1):

1. Select the software to create a web site. The software is included on the DVD and/or you can download the software from the Sun Microsystems web site www.sun.com.
2. Set up and configure a web server or an application server.
3. Create the web site using HyperText Markup Language (HTML) web pages and graphics.
4. Create an application using Java servlets, Java Server Pages (JSPs), and Common Gateway Interface (cgi) programs.

Figure P.1. An Internet web site setup.


Web Site Computer
Surfers
Connection Management
Process Requests and return a response
Website Content
Web Browser
Static Files: Webpages Graphics, other...
Receive requests
Server Application Functions (SAF)
Programs and Applications CGI, Perl, Java programs and other...
Sun ONE Web Server

Project Using a Directory Database

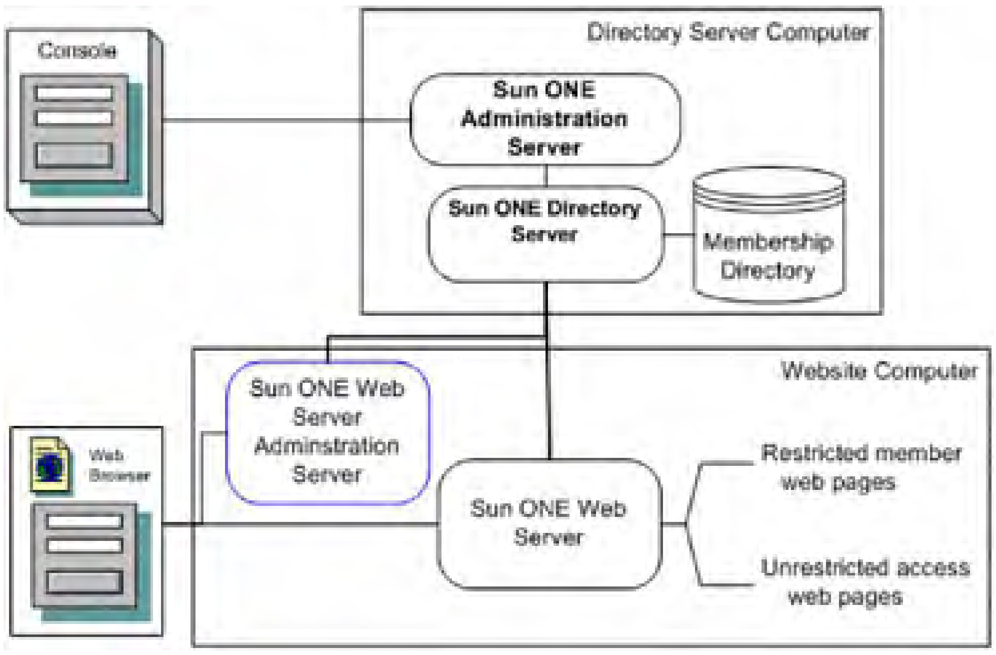
Some of the projects in this book extend through a number of the book's chapters. The setup in Figure P.1 is first established in the web server section. In the directory server section, another piece is added. Once each server is set up and tested, the servers are linked together. Web site members are maintained in the directory, and restrictions are set up to restrict access to the member areas of the web site.

About the Included DVD

The DVD is the Sun ONE Developer kit; it contains many developer tools, including the software you need for this book. Among these tools are the Sun ONE

- Web Server 6.0
- Directory Server 5
- Application Server 7
- Studio 4

Figure P.2. Connecting the Sun ONE Directory Server to the Sun ONE Web Server.


Console
Directory Server Computer
Sun ONE Administration Server
Sun ONE Directory Server
Membership Directory
Website Computer
Sun ONE Web Server Adminstration Server
Web Browser
Sun ONE Web Server
Restricted member web pages
Unrestricted access web pages

Acknowledgments

Thanks to Peter Fernandez and Alan Bernard for hiring me at Sun to work as a Sun ONE course developer; Scott Jolly for mentoring me as a product manager; Judith Fleenor for helping me become a quality instructor and communicator; Annelies Habermacher for hiring me at Netscape and bringing me to Silicon Valley.

Also thanks to those people who have helped make this book what it is!

Technical testing: Chris Guzman, Eile Dragher, Marty Banting, Robin Bravenboer, and Tim Atluru

Content proofreaders: Andrey Ostashko, Christen Lee, Christopher Vervais, Chun-Jui Yang, CP Thompson, Etienne Remillon, Francois Lepretre, Inyoung Cho, Rick Evans, Robert Holt, Rohit Valia, and Sreeram Duvvuru

Technical support: Alex Bugar, Axel Gouze, Blaine Williams, Bill Hemlock, Binu Kurup, Jasson Nassi, Jay Plesset, Ken Johnson, Laura Pan, Hans Lachman, Homer Yau, Mark Reynolds, Samit Rao, Sebastian Kamyshenko, Stan Santiago, and Yunpeng Zang

About the Author

At the University of Manitoba, Canada, Stacy Thurston was an entrepreneurial student, writing software documentation that he later sold to the university. After graduation, he worked for Nestle of Canada as a programmer and system analyst. His main project was a multicorporate computer communication system, which lead to a job with Canada's largest telecom company, Bell Canada. It was in 1994 at Bell that a fellow engineer showed him the Netscape 0.9 browser. After five minutes he was hooked on the World Wide Web.

When businesses began entering the Internet, the scale of his projects became nationwide, industrywide, and international. For example, he designed an Internet and intranet solution for the insurance companies of Canada. Along with some associates, he opened an office to create a business ISP whose biggest project was to set up a secure Internet gateway system for the credit unions of Canada.

In 1998 he began working for Netscape Communications in California's Silicon Valley. At Netscape, he supported companies installing, configuring, and setting up Internet and business services.

When America Online (AOL) took over Netscape, Stacy's group got involved in the Sun-Netscape alliance. He moved into training and began traveling the world to train people on alliance products. He travelled to Asia, Australia, Europe, and throughout North America to spread the excitement and fun of working with Sun ONE. Then, as a product manager, he wrote presentation material and gave presentations to company executives, business people, and developers. In 2001 he joined Sun Microsystems and became a course developer, writing the Sun ONE training material.

He has drawn from all of these experiences to write this relevant, practical, and timely book.

Stacy can be reached at author@internetflow.com.

Chapter 1. Sun Open Net Environment (Sun ONE)

What Is Sun ONE?

Sun ONE is Sun's software strategy that encompasses both a network-oriented software architecture and a set of products to instantiate that architecture. The ultimate aim of Sun ONE is to enable standards-based, network-oriented applications and systems by building upon and working with existing systems.

Sun ONE Stands for Sun Open Net Environment

Open: The key word here is *Open*. Sun supports open standards. An excellent example is the support Sun gives to the open-source Java specifications. Another example is the Apache Jakarta project (*www.apache.org*) which uses Tomcat. Tomcat is from the Sun Java reference implementation. The Apache projects are run by a community of dedicated people working on open-source software such as the Apache web server.

Net: To quote Scott McNealy, the CEO of Sun Microsystems, "Network computing is the heart and soul of this company." Sun designs applications to run on networks.

> *Sun ONE is about applications that work on the net.*

Environment: Sun supports the Java language; Java programs run on a number of operating environments. Sun ONE products run on the major operating systems of the Internet: *Solaris* Operating Environment, *Linux*, other UNIX systems, and *Windows*. Sun has computers that run *Solaris* and *Linux*. Sun ONE creates an environment to set up and run applications on the net.

Middleware Orientation

Middleware is the software between the operating system and user client programs. Sun ONE middleware is integration technology that pulls together applications and services from multiple systems on the net. Sun ONE middleware is between applications and clients on the net.

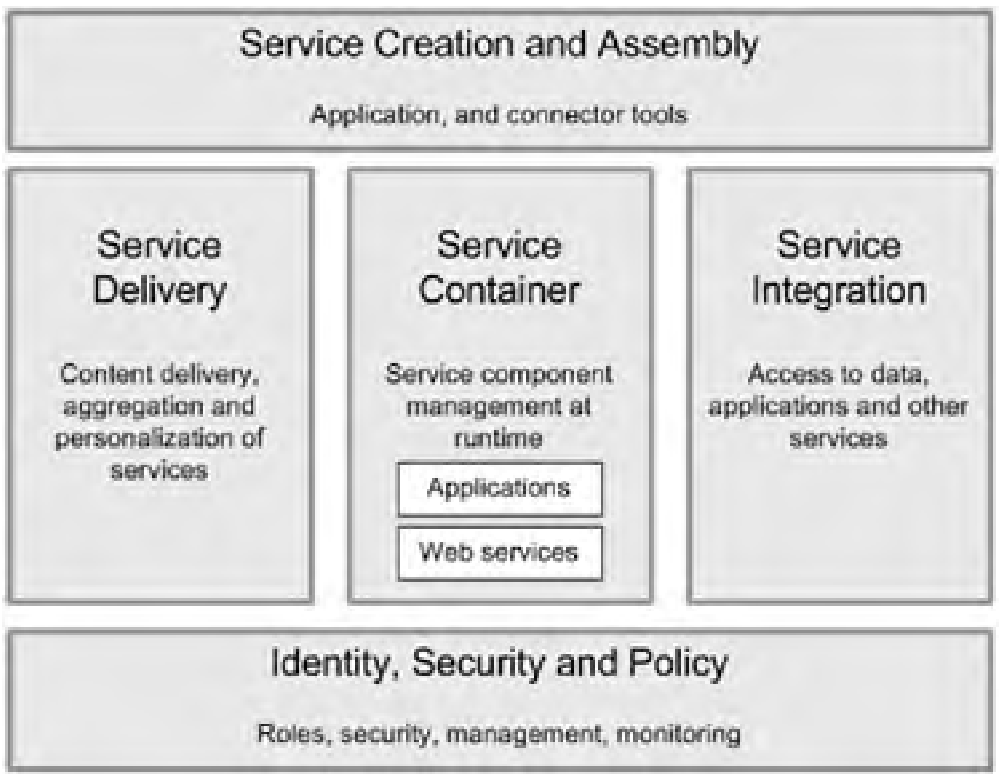
When Sun began building computers, they selected UNIX as the operating system. There are of course a number of operating systems that are UNIX based and some that are not UNIX based. Sun ONE middleware is designed to run on the most popular operating systems used on the Internet. This allows developers to build applications to run over the Internet and to integrate a number of applications from a number of operating systems.

Sun ONE software refers to a breadth of products spanning all the key elements of a complete software infrastructure. Sun ONE software product offerings include operating environments, developer tools, and middleware. This orientation is designed to articulate what is meant by *middleware* in the networked economy. This orientation categorizes the various types of middleware in simple terms and helps you conceptualize the often complex architecture in an elegant form.

The Sun ONE Architecture

Figure 1.1 captures the basic parts of a platform needed to support enterprise computing and establishes a framework from which system architects start to design and organize information technology (IT) projects.

Figure 1.1. Sun ONE architecture.


Service Creation and Assembly
Application, and connector tools
Service Delivery
Content delivery, aggregation and personalization of services
Service Container
Service component management at runtime
Applications
Web services
Service Integration
Access to data, applications and other services
Identity, Security and Policy
Roles, security, management, monitoring

To be able to rely on computers and the network to automate certain business transactions and to offer meaningful *services* through the Internet, a sound software infrastructure must be established. The fundamental building blocks of such an infrastructure include software categorized as

- **Service Creation, Assembly, and Deployment.** Simply put, software products that fit into this category are applications used to write applications. They are themselves applications that are used by developers to write new online applications more quickly and with fewer mistakes. The Service Creation software products can be thought of as offering a *development environment*. These software applications are referred to as developer tools and are used to create, assemble, and deploy services to the Service Container.
- **Service Container.** Often called the *runtime environment*, this box is appropriately at the heart of Figure 1.1

and indicates the main engine of the middleware stack of software. At the most basic level, middleware in this category can be thought of as server-side software that manages multiple online applications being requested by an unpredictable number of users.

- **Service Integration.** This refers to software used to connect to information in another server or system. Service Integration middleware acts as a bridge between different, perhaps older, computing systems and the Internet applications running in the Server Containers.
- **Identity, Security, and Policy.** Server software that is used to quickly retrieve and store user information such as user IDs, passwords, and permissions falls into this category. Security relates to a Public Key Infrastructure system to manage certificates for users.

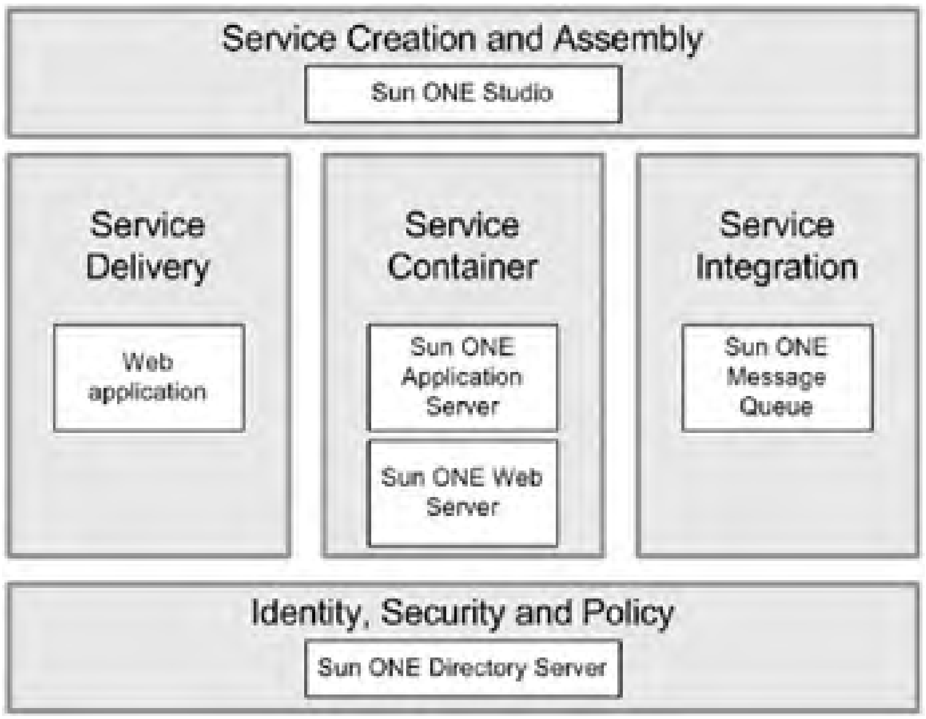
  Service Delivery. Service Delivery software is used to deliver information from various applications in one coherent, consistent interface. It is also software used to cater the information for different devices.

A number of vendors have middleware products based on the Sun ONE architecture that interoperate with Sun technology. Sun itself has its own Sun ONE middleware products.

Sun Microsystems' Sun ONE Middleware Software

Figure 1.2 places the Sun ONE middleware products discussed in this book into the Sun ONE software functionality categories.

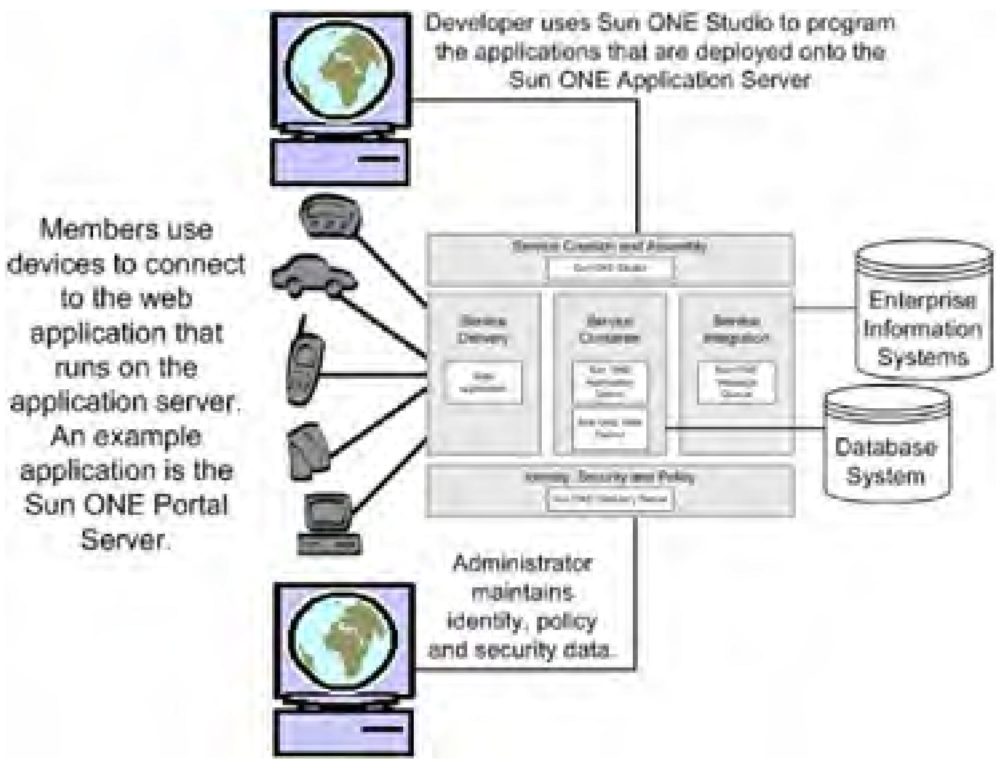
Figure 1.2. Sun ONE server product architecture.



- Sun ONE Studio is used to create, assemble, and deploy applications. In this book there are instructions to create, assemble, and deploy a web application.
- Sun ONE Application Server and the Sun ONE Web Server both have a web container to run Java programs. The application server has an extra container to run Enterprise Java Beans (EJBs).
- The Sun ONE Directory Server manages member data information.

Figure 1.3 shows a sample of Figure 1.2's Sun ONE product architecture diagram in an application environment.

Figure 1.3. Example diagram using Sun Microsystems' Sun ONE products.


Developer uses Sun ONE Studio to program the applications that are deployed onto the Sun ONE Application Server
Members use devices to connect to the web application that runs on the application server. An example application is the Sun ONE Portal Server.
Service Creation and Assembly
Service Delivery
Service Container
Service Integration
Identity, Security and Policy
Enterprise Information Systems
Database System
Administrator maintains identity, policy and security data.

This book is about getting started setting up services based on the Sun ONE design architecture using Sun Microsystems' Sun ONE products. To start using this architecture, this book has instructions to set up the Sun ONE Directory Server, the Sun ONE Web Server, and the Sun ONE Application Server. Once these are set up, the Sun ONE Studio tool is used to create Java applications that are assembled, deployed, and tested.

◄ PREVIOUS NEXT ►

Sun ONE Reference Material

Here are some information links I have used to gather Sun ONE information:

- High-level Sun ONE overview: *wwws.sun.com/software/sunone/*
- Sun ONE Architecture Guide: *wwws.sun.com/software/sunone/docs/arch/index.htm*
- Supply Chain Management Presentation, a Sun ONE solution: *wwws.sun.com/software/sunone/tour/supplychain/overview.html*
- Sun ONE Infrastructure Software Training Courses: *http://suned.sun.com/US/catalog/infrastructure.html*. (I have taken as well as created material for a number of courses on Sun ONE products.)

Sun ONE Open Standards Analogy

To close this discussion of Sun ONE, here is a story that a fellow courseware developer tells his friends.

If you need a computer for your home to get onto the Internet, you would go to a store and buy a box, go home and plug it in, and it would just work. The box would already have the core software: an Internet browser, an email package, a program to set up an Internet account, and so on. The box would also have the core hardware: a modem to connect to the Internet, a monitor, speakers, and the like. If you needed something special, like Macromedia's Dreamweaver to create web sites or a better sound system, you would buy it, install it, and, again, it would just work.

If you and I were going to start a business tomorrow, you would go to a store and buy a box, go the office and plug it in, and it would just work. You would have your core office software: a word processor, a spreadsheet, an Internet browser, etc. If you needed something special, like Adobe FrameMaker to write documentation, you would buy it, install it, and, again, it would just work.

Nowadays, that is what you reasonably expect because a lot of work has gone into standardizing software packages and hardware parts. Sun ONE is the same concept, but on a larger scale—on the enterprise scale. You buy a box from Sun, that box includes the core enterprise services—just like the personal computer box you buy from your local computer vendor—it is ready to go or, at the very least, it is a straightforward process to get it up and running.

If you and I were going to start a business enterprise tomorrow and I asked you to get a company computer system up and running, you would go to the Sun store and buy a box, go to the office and plug it in, and it would just work. The box would already have the core Internet software: a web server, a directory server, etc. If you needed something special, like a database server, you would go to Oracle, buy it, install it, and it would just work. If you wanted to buy some hardware from other vendors such as Cisco, you would buy it and install it and it would just work.

Sun has worked hard with other companies in the industry to make sure Sun systems adhere to open standards so that you can expect those add-ons to just work.

Sun ONE is about applications that work on the network.

Sun ONE makes the net works.

Chapter 2. Transition from Netscape to iPlanet to Sun ONE

The story behind the Sun ONE products discussed in this book begins with Netscape and moves forward to the point at which Sun has its own Sun ONE products.

Corporate Transition

From a product manager's point of view, history is invaluable because knowing where we have been helps us know where we are going. The history begins with Netscape creating its first products in 1994—a web browser and two web servers, one with security and one without.

Netscape then bought other companies to increase the Netscape product line and to increase the number of people in the company.

In 1999 AOL acquired Netscape and with Sun established the Sun-Netscape Alliance. The alliance was formed of people from Netscape and people from Sun. The products produced by the alliance were branded the iPlanet products (iPlanet is a brand name owned by Sun). For the next three years, Sun increased the number of people in the alliance and AOL did not replace anyone who left the alliance. When the alliance ended, AOL laid off most of the people left in the alliance and moved the rest to another AOL project. Many of the people who were laid off were hired by Sun to continue its work with the iPlanet products—I was one of those people.

When the alliance ended in 2002 the product brand names changed from iPlanet to Sun ONE. This chapter will elaborate on these changes in more detail.

Brand-name transition: Products started out as Netscape products, then became iPlanet products, and are now Sun ONE products.

The Netscape Browser and Servers

Netscape started with a simple business plan:

Make it easy for everybody to use the Internet.

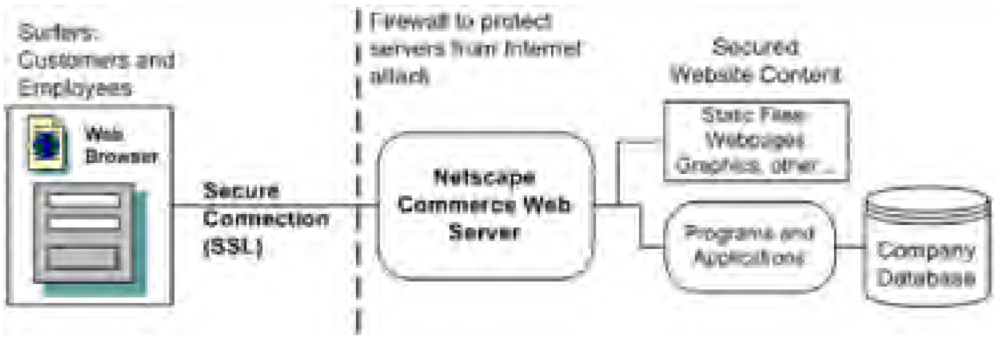
Make it possible to do business over the Internet.

A Browser for Everybody

The browser is the software that made it possible for everybody to use the Internet. The networks that were joined together to create the Internet may have been around since the 1960s, but it was the browser that helped cause a quantum leap in Internet traffic.

The target market for the Netscape browser was business. The browser managed the link between customers, employees, general surfers, and web sites—this was Netscape's area of expertise, its big business innovation. Netscape secured the link between people and web sites and created the Secure Socket Layer (SSL) which is a secure connection between surfers (people with browsers) and the company web sites (illustrated in Figure 2.1).

Figure 2.1. Beauty in simplicity, browser-to-web-server security.


Surfers: Customers and Employees
Web Browser
Secure Connection (SSL)
Firewall to protect servers from Internet attack
Netscape Commerce Web Server
Secured Website Content
Static Files Webpages Graphics, other...
Programs and Applications
Company Database

For business to be done over the Internet, the connection between the browser and web server needed to be secure. It was the combination of the browser, security, and the web server that would make business over the Internet possible. This was the original product plan of Netscape.

Netscape personnel wrote the specifications for SSL using public/ private key pair security. The Netscape browser was capable of using SSL, and a version of its web server was capable of creating SSL connections. They created security between the customer's browser and the business web sites. Business transactions such as buying and selling could be carried out securely over the Internet. Also, employees could communicate securely with the company web site over the Internet.

Trivia: The pre-1.0 version of the original Netscape browser was called *Mozilla*. Legend has it that *Mo* comes from the name of a browser called Mosaic and *zilla* is from Godzilla. Mozilla was the application to defeat Mosaic. Jamie Zawinski had a web site called *Save Mozilla* that was internal to Netscape people, from which I downloaded Mozilla 0.6 and Mozilla 0.9 for testing.

Netscape Stage I: Jim Barksdale Hired

After the release of Netscape 1 in 1995, Jim Barksdale became the CEO of Netscape. Jim's strategy was straightforward.

Get Known, Get In...

The browser, which could be downloaded easily, brought attention to Netscape. And the combination of browser-security-web server got Netscape into business. Security was something that business would pay money to have—business people could understand it and wanted to use it. The products and features were easy to understand:

1. **Client:** The Netscape browser is a user-friendly browser that anyone can use. It is capable of making SSL connections.

2. **Server:** Netscape Commerce Web Server with SSL is capable of making secure connection between the browser and commercial websites.
3. **Tools:** Web site tools are used to create commercial web sites and tools to make connections to back-end databases.

Netscape's plan was simple, concise, and, most of all, original.

> Question: Who did you think created SSL security? Who do you think had the first Internet store?
>
> Answer: Netscape.
>
> Question: Can you name two products Netscape sold in its Internet store?
>
> Answer: The Netscape browser and the Netscape Commerce Web Server.

There is a saying in Silicon Valley: "eat your own dog food." Netscape created products that it used to sell its own products. It created a browser and web server that helped to sell its own browser and web server. Ingenious!

A test of a good product is *whether the creator uses the product*. Netscape certainly did "eat its own dog food."

Figure 2.2 is a screen shot of the Netscape 1.0 browser. Prominently displayed on the About page is the RSA logo. RSA is a company that sells public key data security products. Public key data is used to create SSL connections.

Figure 2.2. Netscape Browser 1.0, RSA security, and Netscape Commerce Web Server.

Dr. Whitfield Diffie was one of the designers of public/private key cryptography. RSA uses public/private key technology to create its products. RSA public/private key technology is used in the commerce version of the Netscape web server. Sun has taken over from Netscape to create the next generation of web servers from Netscape. And Dr. Whitfield Diffie works for Sun Microsystems.

Netscape Stage II: Broader Server Base

The browser/security/web server products allowed Netscape to grow very quickly from the start. During this stage, the company widened its product base to include a more complete Internet infrastructure. The new products added to the product portfolio included an email server, a newsgroup server, a certificate server, and a proxy server. On the browser side, the Netscape browser version 2, Netscape Navigator, was upgraded with the capabilities to do email and browse newsgroups.

Now when companies came knocking on Netscape's electronic doors, they could buy a more complete Internet solution. Netscape had moved from a browser company to become more of a server company.

Netscape Stage III: The JAVA Language

Sun Microsystem's motto, "The computer is the network," worked in their favor as the Internet grew and grew. In 1995 Sun released the Java programming language, a language that has become the programming language of the Internet. At the time Java language was introduced, programmers really needed a programming language that would make writing Internet client-server software easier. Another capability of Java allowed programmers to write programs once and have them work on multiple hardware platforms. It was because of its love of the network that Sun developed the Java programming language.

While Sun created the Java language, Netscape created JavaScript. Netscape was motivated to make the browser even more business capable. JavaScript allowed the programmer to validate form data before the data was sent to the business application, thereby increasing the quality of the data.

JavaScript improved the browsing experience. Once again, the Netscape team gave the Netscape browser features that kept it at number one.

Netscape Stage IV: Internet Applications

Netscape was moving up the application food chain. Business had always been its focus, so Netscape started to focus on business applications. For example, the company bought out General Electric Information Services' (GEIS's) share of the Actra venture (Actra was a joint venture between Netscape and GE) in order to completely own ECXpert. ECXpert has become an integral part of the Sun ONE Integration Server business-to-business (B2B) solution.

The Application Server

To better enhance a web site developer's ability to create web applications with Netscape products, Netscape bought the company Kiva to obtain their application server. This product has evolved into the Sun ONE Application Server 6.5.

Lightweight Directory Application Protocol (LDAP) Directory Server

During the early growth of the Internet, at the University of Michigan Tim Howes helped design the LDAP Internet database protocol. In 1992 LDAP was becoming an Internet standard protocol. For more information, read a transcript of Marc Andreessen's interviewing Tim at *http://wp.netscape.com/columns/techvision/innovators_th.html*

After the university, Tim joined the most progressive software company on the Internet—Netscape. Netscape was continually increasing its talent pool with the *best of the net*. A team was put together, and the Netscape Directory Server was built. This server has grown and been enhanced to become the Sun ONE Directory Server.

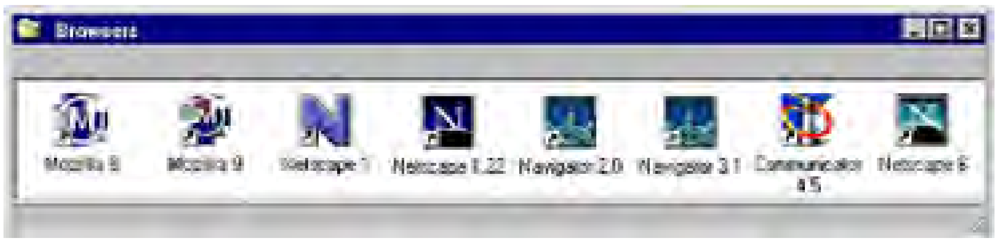
Netscape Stage V: The Sun-Netscape Alliance

It happened—Netscape had purchased a number of companies, and in 1999 it was its turn to be bought. The buyer was AOL. Executives from AOL, Netscape, and Sun got together and worked out some deals, and these are the cards Netscape was dealt:

1. AOL would be the owner of Netscape. Netscape would no longer be a separate company.
2. Netscape would remain as a product brand name; for example, the Netscape browser and the Netscape home page (*http://home.netscape.com*).
3. AOL and Netscape executives would manage the Netscape home page and the Netscape browser.
4. Employees from Sun and Netscape would manage the Netscape server products. This group would be called the Sun-Netscape Alliance.

Some time after the buyout, AOL released its first version of the Netscape browser, calling it Netscape 6.0. The big Netscape *N* had returned as a browser icon! Figure 2.3 shows the changing browser icons.

Figure 2.3. Netscape browser icons from 1994 to 2000.



The Netscape Handover, All-Hands Meeting

This handover meeting was the most amazing business meeting I have ever attended. Imagine having all of these Internet characters on the stage at once:

Jim Barksdale, Netscape CEO

Steve Case, AOL CEO

Bob Pitman, AOL president (formerly from Much Music)

Mark Andreessen, from the original Netscape browser team

Scott McNealy, Sun Microsystems CEO

It was interesting to see the contrast between Jim and Scott's West Coast business humor and Steve's East Coast formal business manner. At times it was like a comedy play—Jim and Scott having fun and Steve the straight man. For example, during the question and answer period someone asked, "Will AOL ever stop sending out all those AOL disks?" While Steve was saying, "Umm..., well..., it is...," Scott speaks up and answers with a big smile: "Never!" and the crowd goes wild with laughter.

Working for Netscape was fun.

The Netscape fun was done.

The Transition Period from iPlanet to Sun ONE

The Sun-Netscape Alliance (1999–2002) was a growth stage for many of the Netscape servers such as the web server, the directory server, and the application server. An example of a growth area was product internationalization because Sun Microsystems is such a large international company. Some products were localized to the local languages of the countries in which the servers were being used.

iPlanet Product Branding

It was decided that the product names would be based on the generic software name prefixed by the iPlanet brand name; i.e., iPlanet Application Server, iPlanet Web Server, and iPlanet Directory Server. As a side note, the iPlanet Directory Server was iPlanet's most successful product (see Figure 2.4 for the iPlanet logo).

Figure 2.4. Figure 2.4 iPlanet logo.

iPlanet inherited products from Netscape/AOL, including

- Netscape Web Server
- Netscape Directory Server
- Netscape Application Server: a server for serving up Java and C++ applications onto the Internet and intranets
- Netscape Messaging Server: email server
- Netscape ECXpert: B2B document communications over the Internet and intranets
- Netscape Certificate Management Service

From Sun, iPlanet has received **other** products, including

- A directory server
- An email server
- The Netdynamics application server

During the time of the alliance:

- Sun bought Forte for Java, now called Sun ONE Studio.
- Sun bought Forte Fusion, now called Sun ONE Integration Server, enterprise application integration (EAI) version.
- Sun developed their Java message queue product, which is now called the Sun ONE Message Queue.

As a group, the Sun-Netscape Alliance

- Developed the next version of the web server called the iPlanet Web Server 6.0, now called the Sun ONE Web Server.

- Updated the Netscape Application Server 4.0 to be the iPlanet Application Server 6.5, which is Java 2 Enterprise Edition (J2EE) compliant based on Sun specifications.
- Created the next version of the Netscape Directory Server, calling it the iPlanet Directory Server. Now this server is called the Sun ONE Directory Server.
- Developed the new email server called the iPlanet Messaging Server from the Message Transfer Agent (MTA) of the Sun email software package. Other parts—POP3, Internet Messaging Access Protocol (IMAP), and webmail—are from the Netscape messaging server. The messaging server is now called the Sun ONE Messaging Server.
- Developed the iPlanet Portal Server 3.0. The Sun ONE Portal Server 6.0 is based on some of the features from the iPlanet Portal Server 3.0. However the Portal Server 6.0 is a rewrite that runs on the Sun ONE Web Server and is soon to run on the Sun ONE Application Server 7 (this information is as of August 2002).
- Developed iPlanet Wireless Server, a bridge between Internet service, cellular phones, and other devices. Accessing the Sun ONE Messaging Server and Sun ONE Calendar Server information on a cellular phone would make use of this server. It is now called the Sun ONE Wireless Server.

iPlanet Services That Are Now Sun ONE Services

In addition to software products, the alliance had service products:

- iPlanet Learning Solutions create, maintain, and deliver courses on the products. The team has moved into Sun Education Services.
- iPlanet Technical Support is now Sun ONE Technical Support. It helps customers with installations, configurations, setup, testing, and anything else they do with the products.
- Professional Services is made up of iPlanet people, now Sun personnel, and helps companies implement and integrate iPlanet products.

Product Internationalization

Because of Sun's size and its contacts with companies in other countries, it was necessary to train people around the world on iPlanet products. In other countries, the dominant business language is not English (see Table 2-1). When I was training people in Bejing, China, the Solaris Operating System (Solaris OS) on the Sun computers used in the training course used Chinese characters. While I know a few Chinese characters, I learned how to toggle between an English keyboard and a Chinese keyboard.

Table 2-1. Sample World Languages

Location	Dominant Languages
United States	English. The second language is Spanish.
Canada	English and French are the national languages. Software companies selling to large corporations in Canada must have English and French versions of their software.
South America	Portuguese and Spanish. English is not as popular here as in North America.
Europe	English, German, French, and Spanish. Other prominent languages are Portuguese, Italian, and Dutch.
Japan	Various Japanese script languages.
Korea	Korean script language.
China	Simplified Chinese characters.
Hong Kong, Malaysia	Traditional Chinese characters. English is very popular as well.
Taiwan	Traditional Chinese characters.
Thailand	Thai. The written Thai language has 42 characters in its alphabet. International companies use English in Thailand.
Singapore	English is the official national language. Chinese and Malay are very popular. They have an interesting spoken dialect they call "Singlish" that is based on English.

Australia, New Zealand	Business language is English.
India	Dominant business language is English. There are a number of local dialectic written languages; however, these are not as popular as English.

Product Localization

When Sun developed products, the business language was English. Localization changes the products to the local dominant business language, taking a product and making it work in other languages. The effort is broken up into three parts:

1. The front-end interactions with the users of the product—such as web page forms, prompts, and menus—are translated into the *local* language. For example, in the J2EE model, Java Server Pages (JSPs) are used for the presentation layer. In multilanguage applications, the business logic is separated from the business language, thereby giving the developer the ability to manage more than one written language easily.
2. Back-end and administration programs are the part of the product that is not used by end users. This is a separate effort for multiple-language localization. Examples are management tools, command line and administration GUI interfaces, and system messages such as tracking and error messages.
3. Translation of product documentation is a major task. This includes translating text, instructions, and screen prints.

Product localization is carried out first on the front end. In some countries, it is acceptable if only the product's front end is localized—this portion is used by the end user. The second and third parts of localization are done for the administrators of the product. Since there are many users of a system and only few administrators of a system, parts two and three are considered secondary to the first part when making the decision to invest resources in the localization effort.

From the Sun-Netscape Alliance to Sun ONE

The alliance era was a real transition period where the focus of product development changed dramatically from the individualized products of Netscape to the integrated products of the alliance.

To give you a taste of the spirit of the Sun-Netscape Alliance that has carried over into Sun ONE, here are some words by Mark Tolliver, the president and general manager of iPlanet e-commerce solutions at the time and now the chief strategy manager of Sun ONE.

Something New

> *I started to manage the Sun-Netscape Alliance in the spring of 1999. I knew it was all about the challenge of going from the Netscape's mission and the America Online's mission and Sun's mission and merging that into iPlanet's mission. This is different than Sun and different than AOL and Netscape, it's* ***something new***.
>
> *Our mission:*
>
> 1. *Allow businesses to make valuable information, applications and resources in the corporate Intranet available to remote employees, field offices, customers, partners, suppliers and consultants from anywhere, through any device.*
> 2. *Build software platforms to create and manage business applications with massive scalability, deliver an individualized experience for each and every person and do so with an uncompromising commitment to open architecture.*
>
> *The spirit of iPlanet software is* ***something new****!*
>
> *Thank you, Mark Tolliver*

Yes, the alliance was something new—it was a transition from a Netscape culture to a Sun culture that is still changing and developing into something new every day.

Netscape was about the development of key Internet products, and iPlanet was about product alignment and integration. Sun ONE is about a software middleware strategy. Sun ONE is a strategy that encompasses both a network-oriented software architecture and a set of products to instantiate that architecture. The ultimate aim of Sun ONE is to enable standards-based, network-oriented applications and systems by building upon and working with existing systems.

Sun ONE

As of 2002, the products have been branded Sun ONE products. All of the employees working on the products are Sun employees. Products are being updated, revised, and upgraded. These updates revolve around the Sun ONE strategy and architecture.

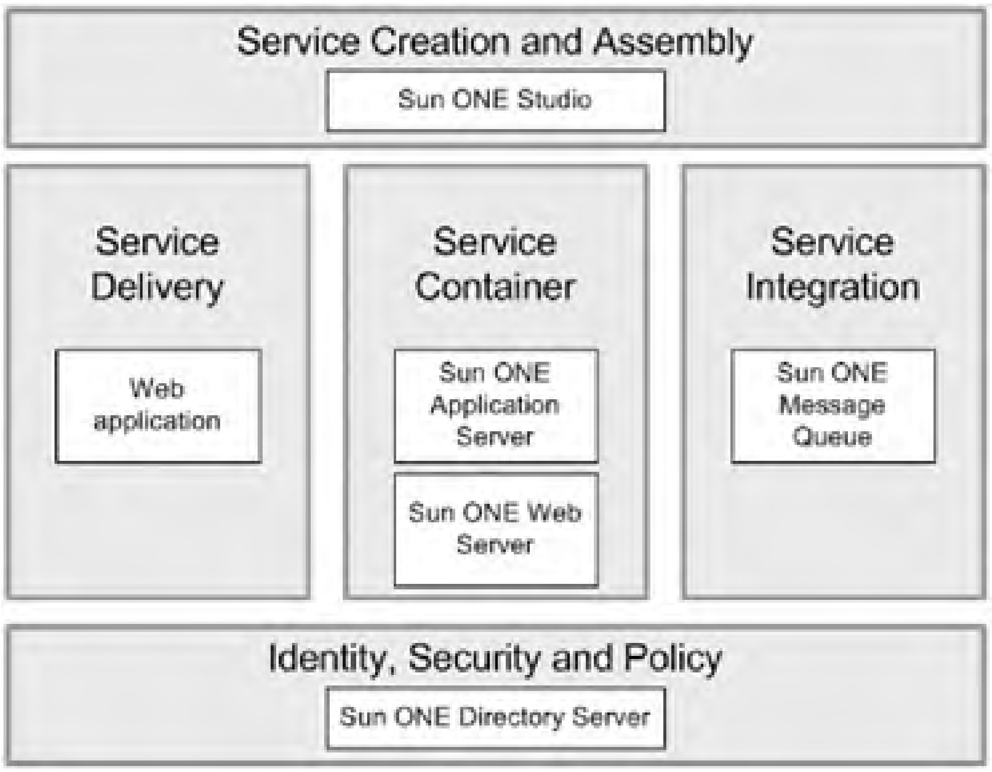
Also in 2002, a new key Sun ONE product was released—the new Sun ONE Application Server 7. This is one cool product with a big future. It has web server functions based on the Sun One Web Server (previously called the iPlanet Web Server derived from the Netscape web server). The Java containers are based on the Java reference implementation that is used in the Apache Jakarta project. Apache is an open-source project. Sun Microsystems is a big supporter of open-source projects. The Apache web server is the most popular web server on the Internet, whereas the Sun ONE Web Server is the most popular commercial web server on the Internet. The Sun ONE Application Server 7 is an awesome combination of solid technologies.

The sister product to the Sun ONE Application Server is the Sun ONE Studio. Sun ONE Studio is used to develop applications to run on application servers. This product, previously called Forte for Java, is now enhanced.

Sun is being extremely aggressive with these product releases. There is the free Community Edition of the Sun ONE Studio and the free version of the Sun ONE Application Server 7, the Platform Edition. This is an incredible combination of free software. All the software on a computer can be totally free if you develop applications using the Community Edition of the Sun ONE Studio, then run the applications on the Platform Edition of the Sun ONE Application Server 7, which is run on Linux or the Sun Solaris OS. These operating systems are both available for free as well! Your only cost is hardware and, when you choose, technical support and training.

Figure 2.5 shows Sun Microsystems Sun ONE products plugged into the Sun ONE architecture diagram. The architecture diagram has products that have grown out of Netscape products into iPlanet products and then into Sun ONE products. The architecture diagram also has products that have grown out of products from Sun in the days before Sun ONE. Sun's Sun ONE products have history from Netscape, iPlanet, Sun, and other companies bought by Sun and Netscape. Sun ONE products are **something new**!

Figure 2.5. Sun ONE server product architecture.



It is quite fun and exciting living and working in an Internet world that is constantly growing and changing. Where the Netscape employees in the early days had the saying, "We are doomed," I often hear at the Sun office the Chinese curse, "We are living in interesting times." The transition has been more than a merging of products; it has been a merging of people and corporate cultures.

Famous People of the Internet

To close this chapter, I want to mention some of the key people in the development of the Internet. Actually, the products they created are better known than they are. Everybody has heard of a browser, but not many people know the names of the individuals who created the technology that made the browser possible. So I have listed the people of Sun and Netscape who have created key technologies and products that make the Internet what it is today.

From Sun Microsystems

- Scott McNealy

  CEO and Cofounder of Sun Microsystems

- Bill Joy

  Cofounder of Sun Microsystems

  Implementer of Transmission Control Protocol/Internet Protocol (TCP/IP) on the Berkeley version of the UNIX operating system

- James Gosling (a Canadian)

  Original design and implementation of Java

  See *http://java.sun.com/people/jag/green/index.html* for a history of Java written by James.

- Dr. Whitfield Diffie

  In 1975 he discovered the concept of public key cryptography.

From Netscape

- Jim Clark

  Founder of Silicon Graphics

  Cofounder of Netscape Communications

  Jim wrote *Netscape Time*. It is from his book among others, Internet web pages, and conversations at work that I came up with Netscape material for this chapter. I met Jim at a book signing at the Palo Alto bookstore *Stacie's*.

- Marc Andreessen

  Original development team of National Center for Supercomputing Applications (NCSA) Mosaic

  Cofounder of Netscape Communications

  One of the founders of Loud Cloud, now called Opsware

- Rob McCool

  Original writer of the NCSA web server, which later developed into the Apache web server

- Eric Bina

  Original development team of NCSA Mosaic

  Created the first Netscape web server

- Kipp Hickman

  Key developer of SSL; note: Taher Elgamal and other team members did a lot for security.

- Lou Montulli

  Developed Lynx, the text-based web browser found on UNIX Original development team for Netscape

- Jim Barksdale

  CEO of Netscape

  Jim was the true company leader. After the AOL buyout, there was a meeting/celebration to say goodbye to the company (as it was). To start the meeting, a few people talked, and then Jim Barksdale was introduced and he came on stage; he got a standing ovation from everyone present (over 1,000 people). This was the first time I had ever seen a group of employees that truly liked and respected their CEO.

- John Myers

  Coauthor of the POP3 protocol standard

- Tim Howes

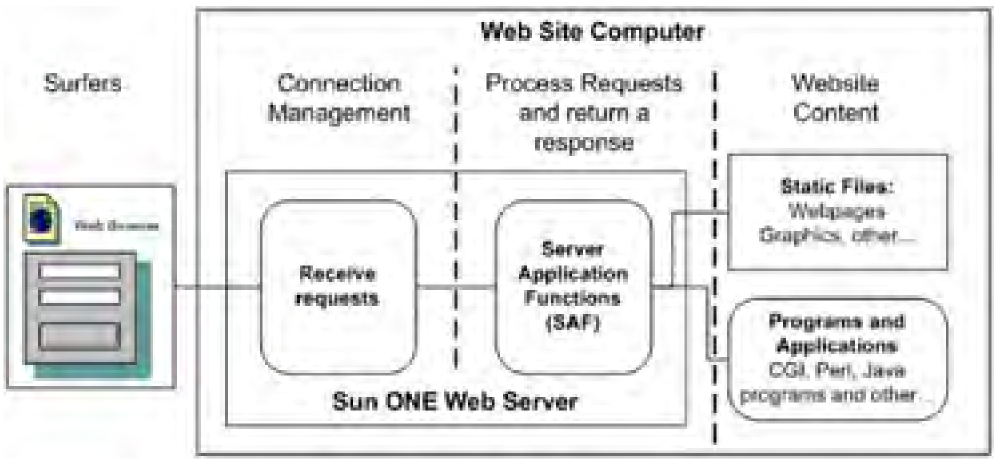
  One of the original designers of LDAP

  Left Netscape to work at Loud Cloud with Marc Andreessen

Chapter 3. The Sun ONE Web Server

You cannot describe the Sun ONE Web Server without mentioning connections—it manages connections between web browsers and the content of a web site. The Sun ONE Web Server is really the link—the gateway—between the users of the Internet, web site content, and web site applications. As illustrated in Figure 3.1, connection management involves orchestrating these three ingredients (surfers, servers, and content) in such a way that, when a person views a web page, the content—whether static or from an application—is displayed in the manner in which it was originally intended.

Figure 3.1. The web server connects surfers to content.


Web Site Computer
Surfers
Connection Management
Process Requests and return a response
Website Content
Web Browser
Static Files: Webpages Graphics, other...
Receive requests
Server Application Functions (SAF)
Programs and Applications CGI, Perl, Java programs and other...
Sun ONE Web Server

Surfers are people using web browsers to look around the World Wide Web (WWW). The browser is a program used by surfers to make requests, receive data, and format the data based on Internet standards.

When a browser connects to a web site, this may be one of many browsers making connections to this web site. To handle a high volume of traffic (user requests), the Sun ONE Web Server has a connection management system. It listens for requests, receives a request, and puts the request into a buffer area to be processed; then it goes back to listening for the next request. Another process picks up a request from the buffer and passes the request on to a server application function (SAF).

The SAF receives the request and creates an appropriate response to the request. The response is created from the web site content—from the files and programs accessible by the web server.

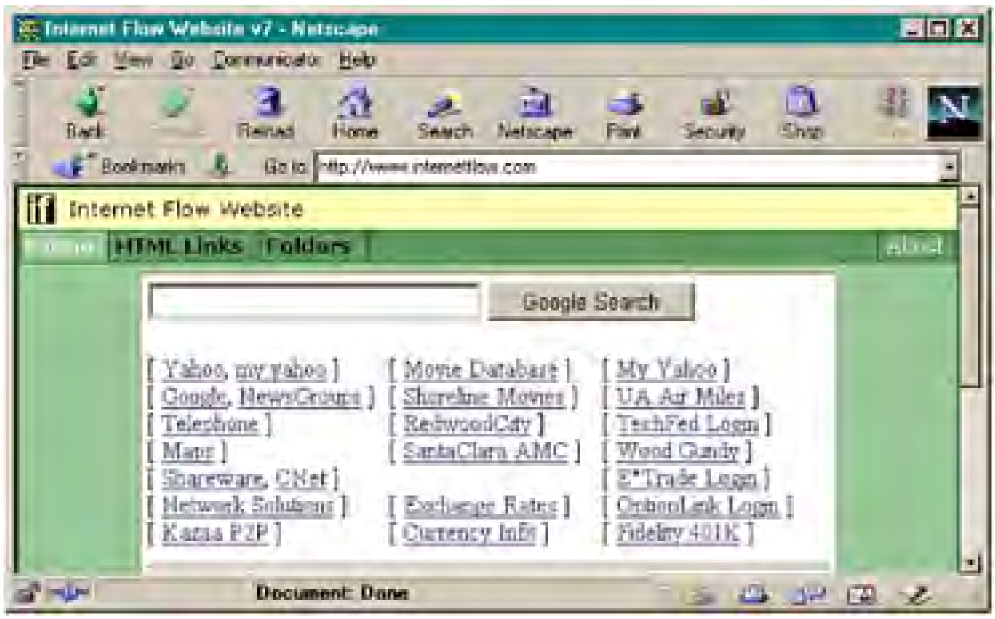
Connection Management

To connect to a web site, a surfer enters a web site address, also known as a Uniform Resource Locator (URL), into a browser and hits the Enter key. The browser then attempts to connect to the web server that is named in the URL. The browser is able to connect to the web server because the web server is waiting for the connection; in other words, the web server is *listening* for web browsers to call it to make a connection.

The Web Browser and the Web Server Listener

When I enter a web site address such as *www.internetflow.com* into my browser, my browser makes a connection to the web server that is associated with the hostname *www.internetflow.com*. Once the connection is made, the browser sends a request to the web server for a file. The web server responds by returning a file to the browser, and the browser displays the file as in Figure 3.2.

Figure 3.2. My browser displays files from the web server that is at *www.internetflow.com*.


Internet Flow Website v7 - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security Shop
Bookmarks Go to: http://www.internetflow.com
Internet Flow Website
HTML Links Folders
Google Search
[Yahoo, my yahoo] [Movie Database] [My Yahoo]
[Google, NewsGroups] [Shoreline Movies] [UA Air Miles]
[Telephone] [RedwoodCity] [TechFed Login]
[Maps] [SantaClara AMC] [Wood Candy]
[Shareware, CNet] [E*Trade Login]
[Network Solutions] [Exchange Rates] [OptionLink Login]
[Kazaa P2P] [Currency Info] [Fidelity 401K]
Document: Done

Listeners listen to ports and browsers make calls on ports. By default, the browser will use port number 80; therefore these two URLs are the same:

www.internetflow.com

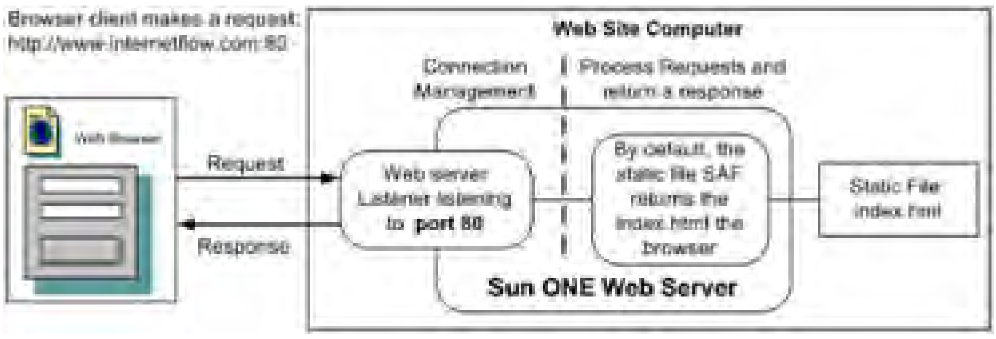
www.internetflow.com:80

The port number is important because the web server listens for browser connections on a port. If the web server at address *www.internetflow.com* was listening on port 85, you need to tell the browser to use port 85:

www.internetflow.com:85

Figure 3.3 illustrates the point of a web server listening on port 80. Note that, by default, the web server will return the web file index.html. This is because the web browser did not ask for a specific file.

Figure 3.3. Browser makes a request; Sun ONE Web Server listening on port 80 responds.


Browser client makes a request:
http://www.internetflow.com:80
Web Site Computer
Connection Management
Process Requests and return a response
Web Browser
Request
Response
Web server Listener listening to port 80
By default, the static file SAF returns the index.html the browser
Static File index.html
Sun ONE Web Server

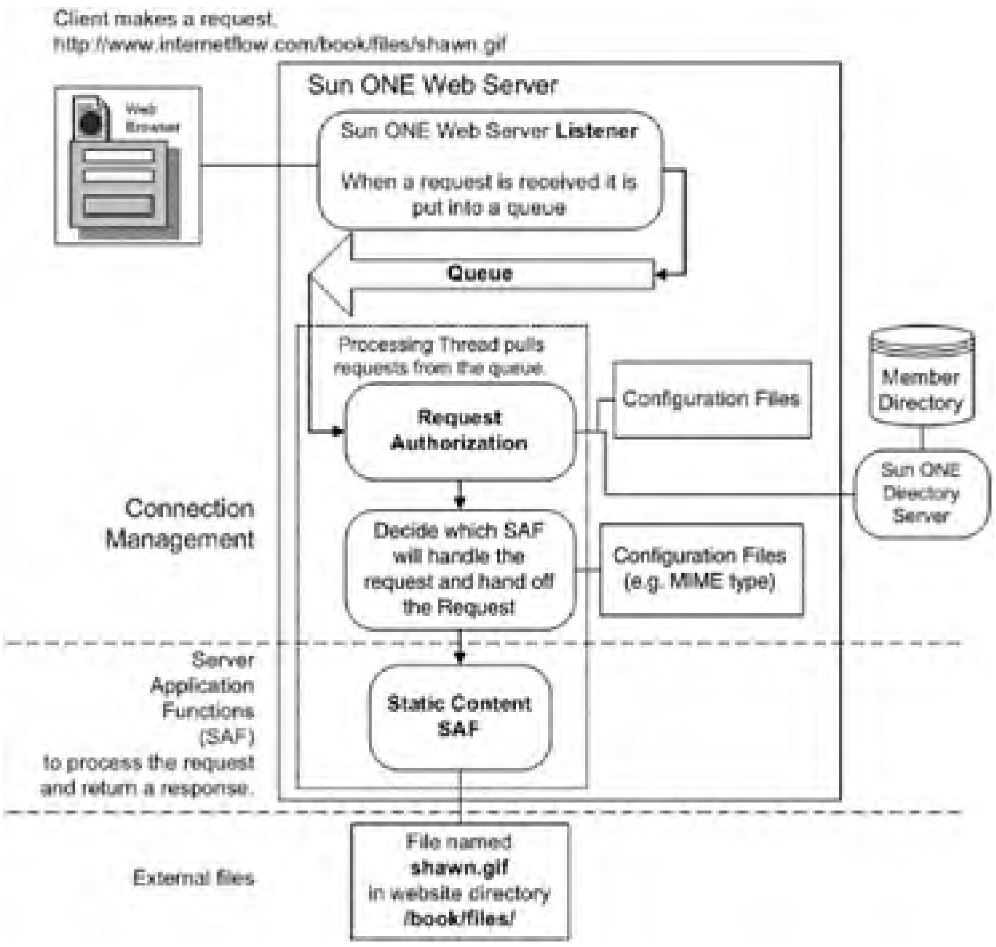
Once the web server listener receives a request, the request is put into a queue. Then a process will pick up the request from the queue and make some decisions about what to do with the request. With the request in Figure 3.3, the decision is easy because the request is based on default configuration settings. The web server will read the default home page file index.html and return the file to the web browser.

The Complete Sun ONE Web Server Process

Let's look at another user request—*www.internetflow.com/book/files/shawn.gif*—and follow it through the entire process from browser request to web server response.

Figure 3.4 is a diagram of the components that manage browser requests. Here is the sequence of events:

Figure 3.4. Sun ONE Web Server managing a browser request for a file.


Client makes a request.
http://www.internetflow.com/book/files/shawn.gif
Web Browser
Sun ONE Web Server
Sun ONE Web Server Listener
When a request is received it is put into a queue
Queue
Processing Thread pulls requests from the queue.
Request Authorization
Configuration Files
Member Directory
Sun ONE Directory Server
Connection Management
Decide which SAF will handle the request and hand off the Request
Configuration Files (e.g. MIME type)
Server Application Functions (SAF) to process the request and return a response.
Static Content SAF
External files
File named shawn.gif in website directory /book/files/

1. Browser connects to the web server listener on port 80, the default port.
2. The listener receives the request and puts the request into a holding queue. This queue creates a buffer between receiving requests and processing requests. This way, the listener is not tied up trying to process requests and can focus on receiving and queuing requests so it can handle receiving a high volume of requests.

3. A process thread picks up the request from the queue and makes a decision based on the following questions:
   - Is the user making the request authorized? The web server will check its configuration files for authorization settings. Also, if the web server is connected to a directory server, it will check the security data in the directory. In this case, the user is authorized.
   - What directory is the requested file in? The web server will check its configuration files for the directory /book/files/. In this case, there are no special configurations for /book/files/.
   - What is the Multipurpose Internet Mail Extension (MIME) type of the requested file? The web server will check its MIME configuration file for the filename extension gif. The MIME type for gif files says that the request is to be handled by the static content SAF.
4. The request is now handed off to the static content SAF.
5. The SAF will read the file shawn.gif and send the file back to the browser.

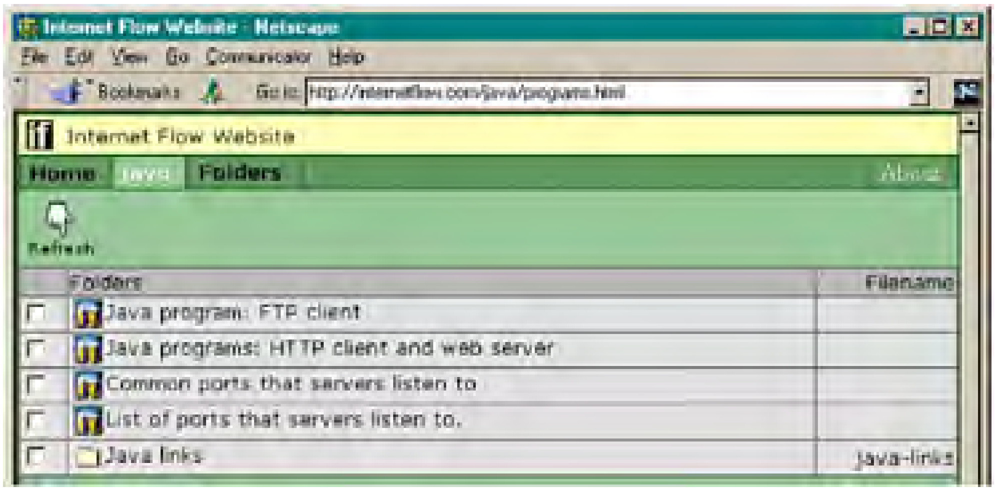
The web server has returned a response to the browser's request.

The listener is the first step in the process to handle a request from a browser. Once the listener has received a request the next process is to authorize the request.

Request Authorization Using the Sun ONE Directory Server

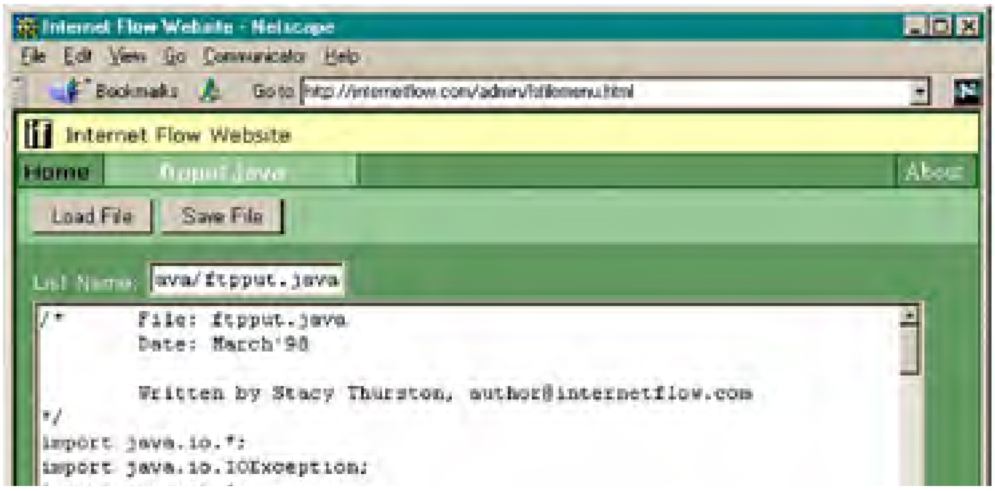
There are times when you need to restrict access to directories for a specific user or group. For example, a member-oriented web site has general members accessing the member web pages (see Figure 3.5) and administrators accessing the application administration web pages (see Figure 3.6).

Figure 3.5. General members access Java program source files for download.


Internet Flow Website - Netscape
File Edit View Go Communicator Help
Bookmarks
Go to: http://internetflow.com/java/programs.html
Internet Flow Website
Home
Java
Folders
About
Refresh

Folders	Filename
Java program: FTP client	
Java programs: HTTP client and web server	
Common ports that servers listen to	
List of ports that servers listen to.	
Java links	java-links

Figure 3.6. Administrator web page to edit Java program source files.


Internet Flow Website - Netscape
File Edit View Go Communicator Help
Bookmarks
Go to: http://internetflow.com/admin/fsfilemenu.html
Internet Flow Website
Home
ftpput Java
About
Load File
Save File

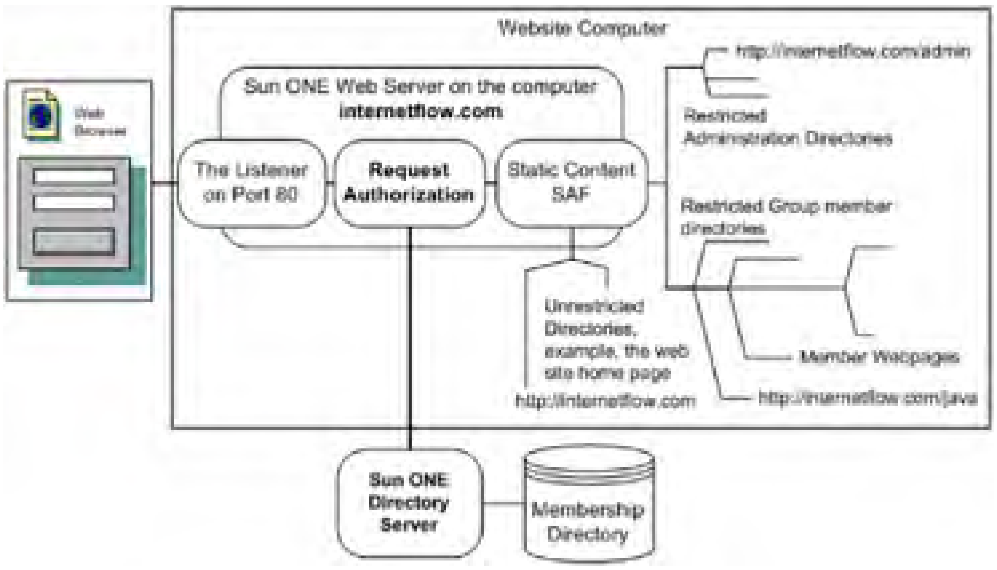
List Name: java/ftpput.java

```
/*      File: ftpput.java
        Date: March'98

        Written by Stacy Thurston, author@internetflow.com
*/
import java.io.*;
import java.io.IOException;
```

```
import sun.net.*;
import sun.net.ftp.*;
import java.util.Enumeration;
import java.util.Vector;

public class ftpput extends FtpClient
```

An elegant method to restrict access to the files and directories is to use the member, group, and security features available in the Sun ONE Directory Server. The directory server is a specialized database system that will maintain membership information and the security information regarding the web site directories. Figure 3.7 lays out the relationship between the Sun ONE Web Server, the Sun ONE Directory Server, and the web site directories. (Note: The instructions for setting up the components in Figure 3.7 are discussed in later chapters.)

Figure 3.7. Directory-controlled web site access.


Website Computer
Web Browser
Sun ONE Web Server on the computer internetflow.com
The Listener on Port 80
Request Authorization
Static Content SAF
http://internetflow.com/admin
Restricted Administration Directories
Restricted Group member directories
Unrestricted Directories, example, the web site home page
http://internetflow.com
Member Webpages
http://internetflow.com/java
Sun ONE Directory Server
Membership Directory

The first step in controlling access to the Sun ONE Web Server web site directories is to connect the web server to the Sun ONE Directory Server. The connection simply requires entering a few parameters into the Sun ONE Web Server administration system—very easy. Now that the directory server is connected to the web server, members are added into the directory using the Sun ONE Web Server Administration Server. This is a web-enabled application to maintain members and groups. Now that members have been added, they are put into the appropriate group. The groups separate the regular members from the administrators.

The next step is to add Access Control Lists (ACLs). These ACLs provide the appropriate access privileges for member groups and for the administrators. In Figure 3.7, the administrators are given access to the web site administration directories and the regular members are given access to the member web pages.

So far this chapter has covered the web server listener receiving requests from browsers and the web server using a directory server to manage request authorization. Now a decision needs to be made as to how to respond to the request; the web server needs to answer the question: What web server process will handle the request?

Selecting a Function to Respond to a Request

Once the web server has made the decision to process the request, it must decide what program function to use to process the request. To make the decision, the web server will check its configuration files:

- obj.conf file— this file contains a mapping between file directories and the SAF used to process requests for files in those directories. For example, any file in the cgi-bin directory will be processed by the Common Gateway Interface (CGI) SAF.
- mime.type file— this file contains a mapping between file extensions and the SAF used to process requests for files having those specific extensions. For example, any file with extension pl (such as login.pl) will be processed by the CGI SAF.

Once the SAF has been chosen, the request is passed on to that SAF to be processed. Whereas connection management covers receiving requests and deciding what to do with those requests, the SAF responds to the requests. The SAF manages and creates the web site content. Next we will look into the SAF process.

SAF Process

The connection management part of the web server receives the requests. However, it is the SAF that will respond with content. The SAF is the work engine of the Sun ONE Web Server. This is where content is selected or created and returned to the browser as a response to the browser's request.

With a web site, content is king! Therefore, to make the most of a web site, you will need to understand the SAFs. If you are the administrator of the web server, you need to set up the SAFs required for the web site developers. If you develop content for the web site, you need to utilize the full potential of the Sun ONE Web Server SAFs to create the best content possible.

Every Sun ONE Web Server SAF has its own features and its own strengths and weaknesses. Each SAF is an executable program that is linked to the web server through an Application Programming Interface (API) called the Netscape Server API (NSAPI).

The Sun ONE Web Server comes with a number of SAFs. The *static* content SAF returns files in the web site directory to the requesting browser. The term *static* is used because the content files are static; this SAF does not modify files, nor does it create content.

The rest of the SAFs create content based on programs that are run. This type of content is called *dynamic* content because the content changes based on the factors of the programs that are called to create a response to the browser request. Dynamic content SAFs include the following:

- The CGI SAF runs programs to create web site content. This SAF will spawn a new process to run the program. The most popular languages for writing CGI programs include Perl, operating system shells, or the C language.

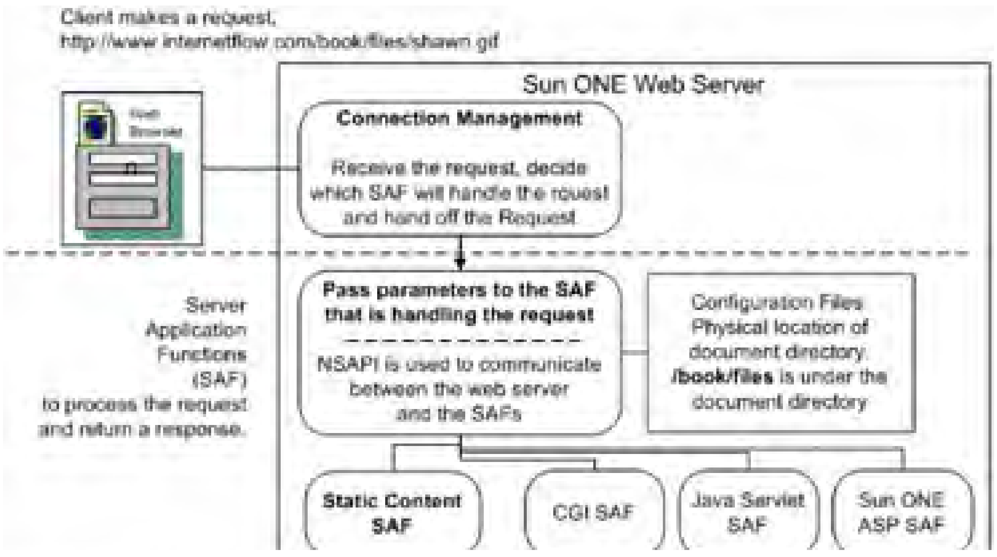
- The Java servlet SAF loads and runs Java servlets using the Java Virtual Machine (JVM). The JVM uses a Java thread to execute the servlet program. Threads are more efficient than spawning a process (CGI SAFs). Another point of efficiency of servlets over CGI is this: When running a servlet, the servlet SAF loads the servlet program into memory and runs it. Afterwards, the servlet stays in memory; therefore, the next time the servlet is called, it does not have to be reloaded—it runs right away.

- The Java Web Application SAF is another type of dynamic SAF. The Sun ONE Web Server comes with the tools to deploy and run web applications. Web applications are made up of a combination of static web site files, Java servlets, and Java beans (except EJBs). The improvement here is that a solution can be packaged into a single web archive file (war file) and deployed to a web server.

- Another SAF worth mentioning is the Sun ONE Active Server Page (ASP) SAF. This SAF gives the Sun ONE Web Server the power to serve up ASP web applications. The Sun ONE ASP system can be downloaded from the website *wwws.sun.com/software/chilisoft/*. This is a Sun ONE ASP system for Sun Solaris, Linux, Microsoft Windows, and other operating systems.

Figure 3.4 outlined the details of the Sun ONE Web Server connection management. Figure 3.8 details the SAF components of the Sun ONE Web Server.

Figure 3.8. Static content SAF selected to respond to the browser request.


Client makes a request,
http://www.internetflow.com/book/files/shawn.gif
Sun ONE Web Server
Connection Management
Receive the request, decide which SAF will handle the rquest and hand off the Request
Server Application Functions (SAF) to process the request and return a response.
Pass parameters to the SAF that is handling the request
NSAPI is used to communicate between the web server and the SAFs
Configuration Files Physical location of document directory. /book/files is under the document directory
Static Content SAF
CGI SAF
Java Servlet SAF
Sun ONE ASP SAF

External files and programs
File named shawn.gif in the website directory /book/files/
CGI program
Java program
Sun ONE ASP server
ASP application

Jackson Thompson, Netscape/iPlanet/Sun ONE Web Server technical expert, says, "To get a deep understanding of the Sun ONE Web Server, read the first three chapters of the *iPlanet Web Server 6.0, Enterprise Edition NSAPI Programmer's Guide*" (*docs.sun.com/db/coll/S1_ipwebsrvree60_en*). I have used this guide and I highly recommend it if you want to understand the technical insides of this server. The guide is available from the Sun documentation web site: http://docs.sun.com. In the *Search* box, type *nsapi web*.

Now that we have had an overview of the SAFs, let's look at three of the SAFs in detail. The following sections contain configuration information and sample setups for the Sun ONE Web Server SAFs mentioned in Figure 3.8.

Static Content SAF

A web browser sends request to web servers. The SUN One Web Server uses its static content SAF to respond to requests by reading and sending files back to the browser. This is the basic method to serve web site content. During an installation of the Sun ONE Web Server, this method of serving files is set up by default.

To follow the steps through the process of replying to a static file request, we consider this URL:

http://www.internetflow.com/book/files/shawn.jpg

The syntax of the URL is

<protocol>://<computer>/<offset directory>/<filename>

- The <protocol> is HyperText Transport Protocol (HTTP), which is a straightforward request/response protocol. This means simply that a client makes a request and the server returns a response. (Note: It is no longer necessary to actually type *http://* before *www* when typing a web address. Hence, you will not see it used in this book except for explanatory purposes as it is used here.)
- <computer> is the name of the computer running the web server that is hosting the web site.
- <offset directory> is the directory portion of a URL. On the web server computer, this directory is accessible by static content SAF. In a default setup, the offset directory is under the *document root directory*.

In our example URL

- The protocol is http.
- The computer is *www.internetflow.com*.
- The offset directory is /book/files/.
- The file is shawn.jpg.

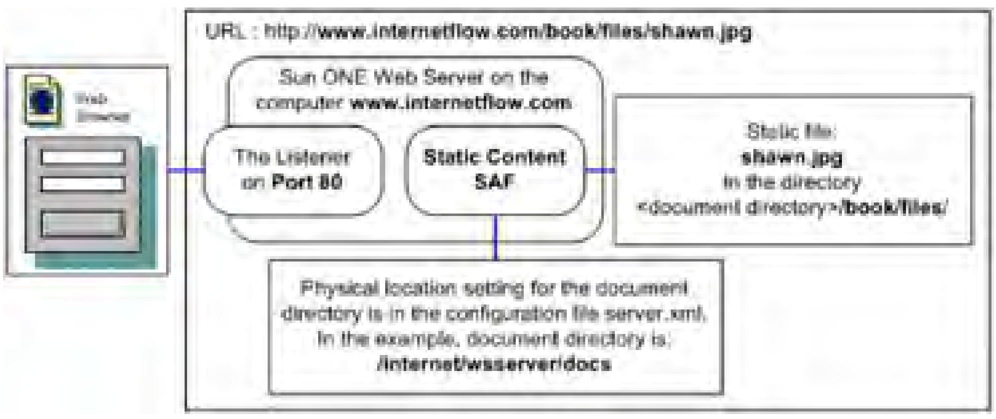
To demonstrate the relationship between a URL and the physical file on a web server computer, here are the mappings from the sample Sun ONE Web Server set up on the computer *www.internetflow.com*. The physical *document directory* is /internet/wsserver/docs/. This means that

- The URL *www.internetflow.com/* is mapped to the physical directory /internet/wsserver/docs/.
- The URL *www.internetflow.com/book/files/* is mapped to the directory /internet/wsserver/docs/book/files/.
- The complete URL *www.internetflow.com/book/files/shawn.jpg* is mapped to the file /internet/wsserver/docs/book/files/shawn.jpg.

When this URL is requested, the file shawn.jpg is read by the static content SAF and sent back to the web browser.

Figure 3.9 illustrates the relationship (or mapping) between a URL and the physical file on a computer.

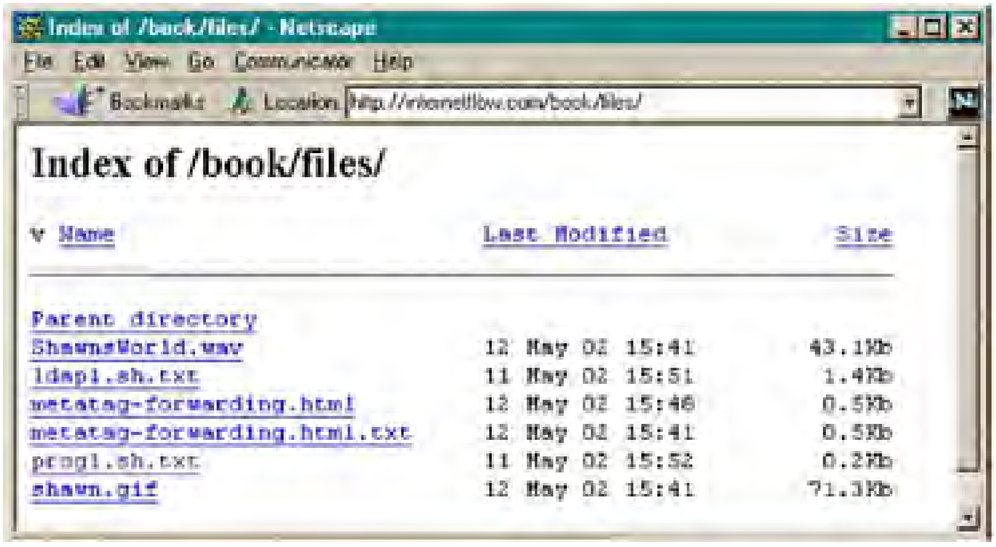
Figure 3.9. Mapping of a URL to a physical file.


URL : http://www.internetflow.com/book/files/shawn.jpg
Web Browser
Sun ONE Web Server on the computer www.internetflow.com
The Listener on Port 80
Static Content SAF
Static file: shawn.jpg in the directory <document directory>/book/files/
Physical location setting for the document directory is in the configuration file server.xml. In the example, document directory is: /internet/wsserver/docs

File Directory Listing

A file directory listing is another function of the static content SAF. The web server at *www.internetflow.com* is set up to allow file directory listings. From a browser, the URL *www.internetflow.com/book/files/* will give a listing of the files in the directory /internet/wsserver/docs/book/files/. The results are shown in Figure 3.10.

Figure 3.10. Static content SAF directory listing.


Index of /book/files/ - Netscape
File Edit View Go Communicator Help
Bookmarks
Location: http://internetflow.com/book/files/

```
Index of /book/files/

Name                              Last Modified          Size

Parent directory
ShawnsWorld.wav                   12 May 02 15:41      43.1Kb
ldap1.sh.txt                      11 May 02 15:51       1.4Kb
metatag-forwarding.html           12 May 02 15:46       0.5Kb
metatag-forwarding.html.txt       12 May 02 15:41       0.5Kb
prog1.sh.txt                      11 May 02 15:52       0.2Kb
shawn.gif                         12 May 02 15:41      71.3Kb
```

When the web page in Figure 3.10 is on the screen, select *View*, then *Source* and see what the web page looks like that the static SAF created and sent to your web browser. From the browser, clicking on any of the files listed in this web page will cause the browser to send a request to the web server for that file. The web server's static SAF will respond by sending the file back to the browser. Then, based on the configuration of the browser, the browser will process the received file. For examples:

- Click on *shawn.jpg* and the browser receives the file and displays it because the browser is configured to handle files with the extension jpg.
- Click on *ShawnsWorld.wav* and the sound file will be downloaded to the browser and the browser will run a program to play the wav file. My Microsoft Internet Explorer downloads the file, then asks me if I want to open the file or save the file. When I choose to open the file, Winamp will start up and play the sound file. Winamp plays the file because Winamp is set up to play files with the extension wav on my computer. How your browser handles a wav file depends on the browser configuration. An easy way to find out is to try it.
- Click on *prog1.sh.txt* and the file will be sent by the web server to the browser and the browser will display the file.
- With Mozilla or Netscape browsers, to override browser configurations, hold the Shift key down and then click on the link. A popup window prompts you for a location to save the file to. This is a convenient option for saving files from web sites.

Dynamic Content SAF

While the static content SAF deals with reading and sending files without making any changes to the files (content is static; it is not changed by the web server), dynamic content SAFs are dynamic content engines that run programs to create content. Dynamic content SAFs create content based on the output of programs.

Dynamic content will make a web site come alive! The web site will react to user input and will become more useful. Let's say you want to know how many British pounds it will cost to buy 100 Euro dollars. There are web sites on the Internet that will do the calculation for you. The calculation is based on user input to generate dynamic content. At one web site, I entered 100 Euros and asked that it be converted into British pounds. It calculated the answer to be 63.88 British pounds to buy 100 Euros. I then asked for the cost in Hong Kong dollars and received the answer that it costs 743.61 Hong Kong dollars to buy 100 Euros. For a world traveler, such sites are very useful.

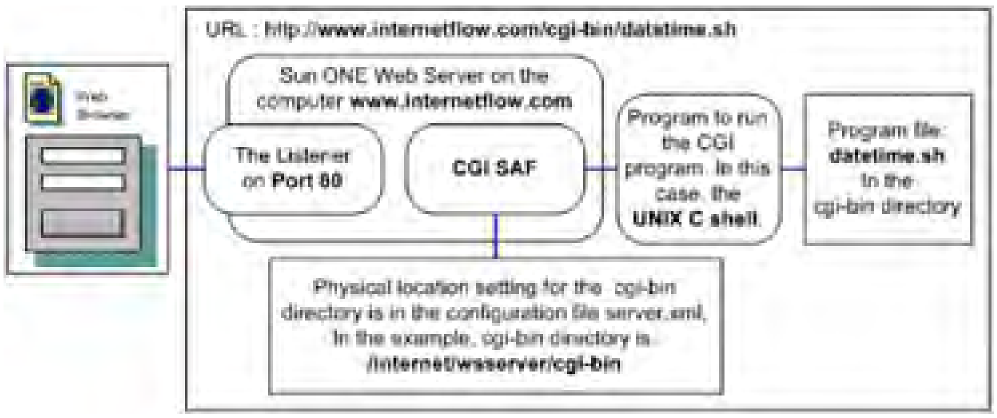
While calculations are an example of dynamic content, it is possible to create complete web-enabled applications using the dynamic content SAFs of the Sun ONE Web Server. Note: By web-enabled applications, I mean applications that are accessible by a web browser. Web mail (email from a web site), likely the most popular application on the Internet, is an example of a web-enabled application.

Dynamic Content from the CGI SAF

Now let's look at how to create dynamic content using the Sun ONE Web Server CGI SAF. This description is based on a CGI sample program I created and posted on the Internet. The URL to my CGI sample program is *www.internetflow.com/cgi-bin/datetime.sh*. The program displays the system date and time of the computer on which the CGI program is run.

When generating *dynamic* content, the same URL is given to the same web server, and the response changes based on the data the program uses to generate the web page. With datetime.sh, every time the page is reloaded or refreshed (same URL), the CGI program is run again and the time that is displayed on the web page will move ahead (see Figure 3.11). The web page changes because the data used to generate the web page changes; that data is the computer's system date and time.

Figure 3.11. Sun ONE Web Server configured to run programs.


URL : http://www.internetflow.com/cgi-bin/datetime.sh
Web Browser
Sun ONE Web Server on the computer www.internetflow.com
The Listener on Port 80
CGI SAF
Program to run the CGI program. In this case, the UNIX C shell
Program file: datetime.sh In the cgi-bin directory
Physical location setting for the cgi-bin directory is in the configuration file server.xml. In the example, cgi-bin directory is /internet/wsserver/cgi-bin

The setup of the Sun ONE Web Server CGI SAF has some similarities to the static content SAF, for example, the configuration of an offset directory. In this CGI URL, the offset /cgi-bin/ directory is mapped to the physical directory /internet/wsserver/cgi-bin/. This means that all requests for files in /cgi-bin/ will cause the web server to spawn a process that will run the program file. When the program sends/prints to the standard output, the print characters are sent back to the requesting browser.

To demonstrate the relationship between a CGI URL and the physical program file on a web server computer, here are the mappings from my sample Sun ONE Web Server setup. The physical /cgi-bin/ directory is /internet/wsserver/cgi-bin/.

This means that

- The URL *www.internetflow.com/cgi-bin/* is mapped to the physical directory /internet/wsserver/cgi-bin/.
- The complete URL *www.internetflow.com/cgi-bin/datetime.sh* is mapped to the program file /internet/wsserver/cgi-bin/datetime.sh.

To add another CGI program to the Sun ONE Web Server CGI SAF environment, simply create the program and copy it into the /cgi-bin/ directory. As soon as the program is in the directory, it can be tested. Here is a sample shell CGI program, but Perl programs are very popular CGI programs too. There are many freeware Perl programs on the Internet to be downloaded and used to enhance a web site. Furthermore, many other programming languages (such as C) are used to create CGI programs.

To show how a program creates content, here is the listing of the source code for datetime.sh. This will create an HTML web page.

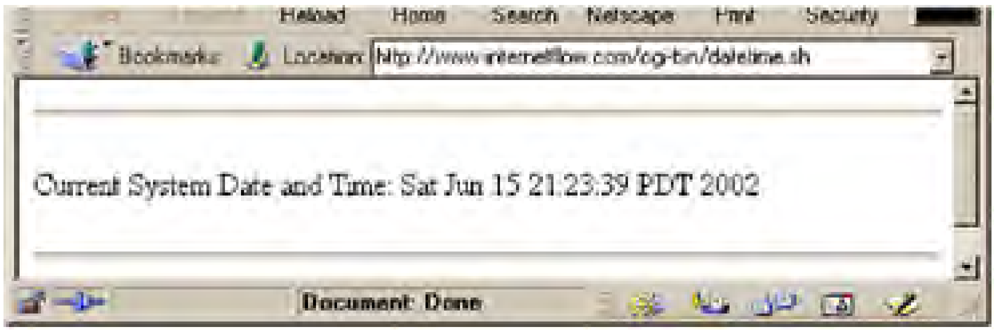
```
01. #!/usr/bin/csh
02. echo "Content-type: text/html"
03. echo ""
04. echo "<html>"
05. echo "<HEAD>"
06. echo "<TITLE>Date and Time</TITLE>"
07. echo "</head>"
08. echo "<body>"
09. echo "<p><hr><p>"
10. echo "Current System Date and Time:"
11. date
12. echo "<p><hr><p>"
13. echo "</body>"
14. echo "</html>"
15. exit
```

When the program datetime.sh is called, the CGI SAF spawns a process:

- The first line, line 01, tells the process to run the program using the csh (pronounced C-shell) environment.
- The command echo writes the string—the string that follows the echo command—back to the browser.
- The lines 02-03 tell the browser an HTML web page is coming.
- Lines 04-14 create the web page.
- Line 11 is a system call that writes the system date and time. This is also sent back to the browser.
- Line 15 is the last line of the program.

Figure 3.12 is the browser showing a sample result from datetime.sh.

Figure 3.12. A browser displaying the generated web page (from datatime.sh).



http://www.internetflow.com/cgi-bin/datetime.sh

Current System Date and Time: Sat Jun 15 21:23:39 PDT 2002

When I do *View Source* from the browser in Figure 3.12, the listing of the HTML web page is displayed:

```
<html>
<HEAD>
<TITLE>Date and Time</TITLE>
</head>
<body>
<p><hr><p>
Current System Date and Time:
Sat Jun 15 21:23:39 PDT 2002
<p><hr><p>
</body>
</html>
```

Looking closely, you will see that each line in the HTML web page matches the echo commands in lines 04-14 in the program datatime.sh. In other words, whatever is written out by the CGI program is received by the browser.

CGI is the oldest method used to run programs from a web server. The Java servlet SAF has some improvements over CGI. Let's have a look.

Java Servlet SAF

Some SAFs can be more efficient than the CGI SAF. The Java servlet SAF is one of those. For example, every time a CGI program is called, the CGI SAF spawns a process to run the program; in the servlet SAF, the JVM uses a Java thread to execute the servlet program. Threads are more efficient than spawning a process. Furthermore, with the Java servlet SAF, the servlet program is loaded into memory and run. Afterwards, the servlet stays in memory; therefore, the next time the servlet is called, it does not have to be reloaded—it runs right away.

The loading and management of servlets is handled by what is called the web container. When a servlet is being called, the request is passed to the web container and the container checks to see if the servlet is or is not already in memory. If it is not, the servlet is loaded and run; if in memory, the servlet is run right away because it does not have to be loaded again. Once in the container, the servlets are run using the JVM. The JVM is part of the Java Runtime Environment (JRE).

The Java programming language is a good choice in which to write SAF programs because servlets are easy to write and, during a typical installation of the Sun ONE Web Server, the Java servlet SAF is automatically installed and configured. Also, included with the installation of the web server are sample servlet source code and class files.

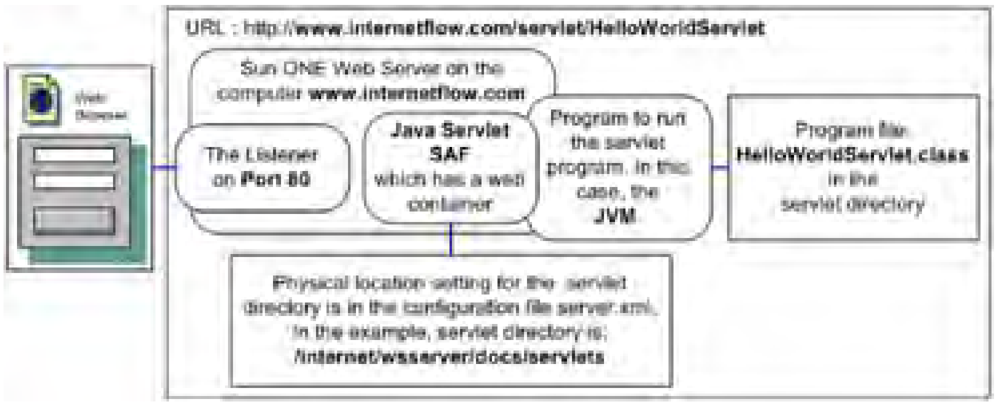
The steps to set up and test a Java servlet are easy.

1. Install the Sun ONE Web Server. Instructions are included in a later chapter. This is a very straightforward installation.
2. Make a servlet directory.
3. Copy a sample into the servlet directory.
4. Call the servlet program, and the sample servlet will run.

An example of a servlet URL is *www.internetflow.com/servlet/HelloWorldServlet.*

Just like the other SAFs, the Java servlet SAF has an offset directory configured as shown in Figure 3.13. This offset directory is configured for the servlet class files (programs). In this servlet URL, the offset /servlet/ directory is mapped to the physical directory /internet/wsserver/docs/servlets/. This means all requests for files in the /servlet/ directory cause the web server to run the Java servlet files. And, when the servlet sends/prints to the standard output, the print characters are sent back to the requesting browser.

Figure 3.13. Java servlet SAF components.


URL : http://www.internetflow.com/servlet/HelloWorldServlet
Web Browser
Sun ONE Web Server on the computer www.internetflow.com
The Listener on Port 80
Java Servlet SAF which has a web container
Program to run the servlet program, in this case, the JVM
Program file: HelloWorldServlet.class in the servlet directory
Physical location setting for the servlet directory is in the configuration file server.xml. In the example, servlet directory is: /internet/wsserver/docs/servlets

To further demonstrate the relationship between a servlet URL and the physical Java servlet class file on a web server computer, here are the mappings from my Sun ONE Web Server setup on computer *www.internetflow.com*. The physical *servlet directory* is /internet/wsserver/docs/servlets/.

This means that

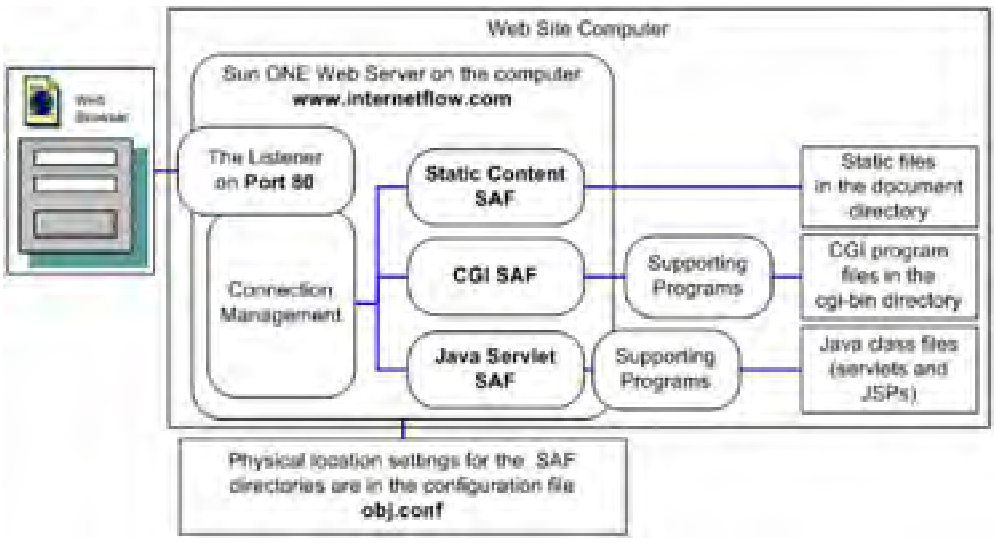
- The URL *www.internetflow.com/servlet/* is mapped to the physical directory /internet/wsserver/docs/servlets/.
- The complete URL *www.internetflow.com/servlet/HelloWorldServlet* is mapped to the servlet class file /internet/wsserver/docs/servlets/HelloWorldServlet.class.

Another example of a servlet is CounterServlet, which shows how the servlet's memory can be accessed by a number of browsers. An example of this type of URL is *www.internetflow.com/servlet/CounterServlet*. The first time this servlet is called, the servlet program is loaded into memory by the servlet container and the servlet is run by the JVM. The counter value is initialized to 1 by the servlet. The next time the servlet is called, the servlet is already in memory, so it is not reloaded. The servlet program will increment the memory value to 2. Then, each time the servlet is requested, the same memory variable is incremented, and the result is returned to the calling web browser. A number of different browsers can access the same memory variable.

Summary

The Sun ONE Web Server is a flexible tool for serving web sites. The serving of a web site is broken into two main parts: connection management, which receives and manages browser requests, and SAFs, which send web site content back to browsers. Figure 3.14 diagrams the SAFs that have been discussed in this chapter.

Figure 3.14. Sun ONE Web Server SAFs and related components.


Web Site Computer
Web Browser
Sun ONE Web Server on the computer www.internetflow.com
The Listener on Port 80
Connection Management
Static Content SAF
CGI SAF
Java Servlet SAF
Supporting Programs
Supporting Programs
Static files in the document directory
CGI program files in the cgi-bin directory
Java class files (servlets and JSPs)
Physical location settings for the SAF directories are in the configuration file obj.conf

The Sun ONE Web Server can be connected to a Sun ONE Directory Server to make use of the directory server features of user and group data management and security. This feature gives the web site developer the ability to control access to selected directories.

Web site content is either static or dynamic. Content is either loaded into the web server environment and served to surfers as is or it is dynamically created by programs that are run by the web server. Either way, SAFs send the content back to the browsers.

URLs can be mapped to files on the web server computer. This is the key to configuring the Sun ONE Web Server and setting up static files and programs on the web server computer as well as the key to making content available on the Internet.

The number of web servers on the Internet has grown from tens of thousands to tens of millions to hundreds of millions because the web server is a great creative tool. The Sun ONE Web Server handles small-scale, single-computer web servers that are easy to set up, or it can be part of a large, complicated server farm system serving millions of users. Whether small or large, web servers are diverse in terms of function. They can serve text files or multimedia files, run programs, and be gateways to other systems.

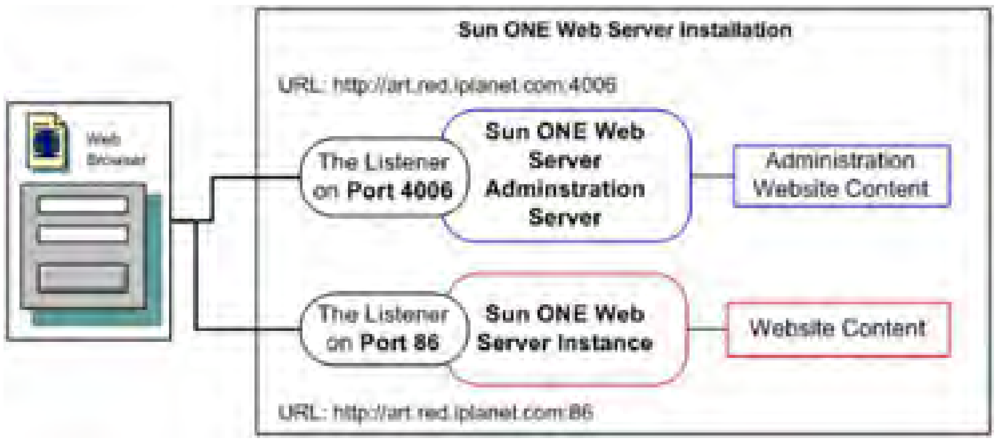
The Sun ONE Web Server is an integral part of an Internet or intranet application design. Whether an application needs to be accessed by a small number of people over a local area network (LAN) or by millions of people worldwide over the Internet, the Sun ONE Web Server is a great tool!

Chapter 4. Installing the Sun ONE Web Server 6.0

The installation of a Sun ONE Web Server 6.0 provides an administration server for managing web server instances as well as a regular web instance for serving a web site. The administration server is called the Sun ONE Web Server Administration Server. It is specialized web server instance with a web-enabled application to manage and configure web site web server instances.

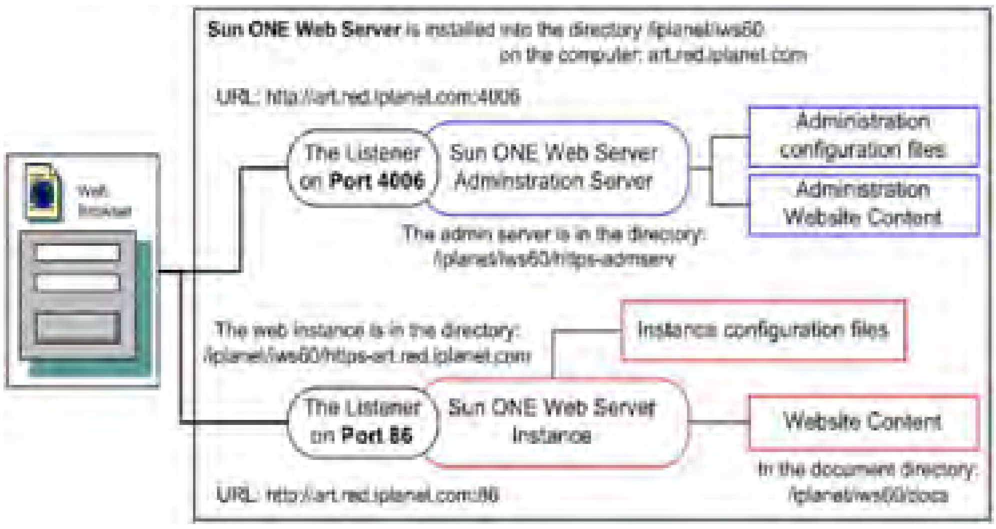
Figure 4.1 shows a web browser being used to communicate with each of the web server instances. The web browser uses one URL to connect to the Sun ONE Administration Server and another URL to connect to the web server instance.

Figure 4.1. Two web server instances are installed during an installation.


Sun ONE Web Server Installation
URL: http://art.red.iplanet.com:4006
Web Browser
The Listener on Port 4006
Sun ONE Web Server Adminstration Server
Administration Website Content
The Listener on Port 86
Sun ONE Web Server Instance
Website Content
URL: http://art.red.iplanet.com:86

During the installation, the installer gives the installation program the values to set up the complete Sun ONE Web Server. Figure 4.2 is a diagram with the values that I used to set up my sample web server; note that all the directories for the web servers are under the root installation directory /iplanet/iws60.

Figure 4.2. Sun ONE Web Server sample installation configuration.


Sun ONE Web Server is installed into the directory /iplanet/iws60
on the computer: art.red.iplanet.com
URL: http://art.red.iplanet.com:4006
Web Browser
The Listener on Port 4006
Sun ONE Web Server Adminstration Server
Administration configuration files
Administration Website Content
The admin server is in the directory: /iplanet/iws60/https-admserv
The web instance is in the directory: /iplanet/iws60/https-art.red.iplanet.com
Instance configuration files
The Listener on Port 86
Sun ONE Web Server Instance
Website Content
In the document directory: /iplanet/iws60/docs
URL: http://art.red.iplanet.com:86

/iplanet/iws60/https-admserv is for the administration server.

/iplanet/iws60/https-art.red.iplanet.com is for the web site.

/iplanet/iws60/docs is the document directory.

By keeping everything under one root directory, it is easy to track the location of all the Sun ONE Web Server files.

The value for the administration user is also required during an installation. For the book's sample installations, I used the administration user ID of *admin* and the password *admin*. When a person connects to the Sun ONE Web Server Administration Server, he or she is prompted for this user ID and password. The administration user gets you logged in to the administration server.

For UNIX installations, two UNIX IDs are required. One UNIX user ID is to run the administration processes, and the other is to run the web server web site instance process. If you are using listener ports below 1024, say 80, *root* must be used to run the administration processes because only *root* can run processes below 1024. For running the web site processes, a common user ID is *nobody*, which is in group *nobody*. The reason for using *nobody* is because it does not have privileges on the network.

Now that you know what to expect when a Sun ONE Web Server is installed, install one on your computer. Following are step-by-step instructions for installing a Sun ONE Web Server on a UNIX Solaris operating system. Following that are the steps needed to install the web server on a computer running Windows NT or Windows 2000.

[Team LiB] PREVIOUS NEXT

SUN Solaris UNIX Installation

For my testing, I used a computer running Solaris 8 with a static IP address, hostname, and domain name.

The Sun ONE Web Server for Solaris is on the DVD that is included on the book, or you can download it, along with the latest service packs, from *www.sun.com*. Follow these directions:

1. Once connected to *www.sun.com*, select *Downloads.*
2. Click on *View All.*
3. Click on S–Z to find *Sun ONE Web Server*—I use *Service Pack 2* in this book, but a higher service pack should be fine.
4. Then select your operating system (for us, *Solaris 2.6, 7, 8*). Click *Download*.
5. Then follow the steps to download the software onto your computer.
6. Put the file into a working directory that you can retain and use later.
7. Decompress the file. Use these commands to decompress a tar.gz file:

```
gzip -d *.gz
tar xvf *.tar
```

Here is a listing after my gzip and tar:

```
root[sh]@art# ls
LICENSE.txt   WebServer   iws60.tar   setup.inf
README.txt  installWrapper  setup   template.inf
```

iws60.tar is the file I downloaded.

Running the Installation Program ./setup

```
root[sh]@art# ./setup
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
Welcome to the iPlanet Web Server installation program
This program will install iPlanet Server Products and
  the iPlanet Console on your computer.
It is recommended that you have "root" privilege to
  install the software.
Tips for using the installation program:
  - Press "Enter" to choose the default and go to the
  next screen
  - Type "Control-B" to go back to the previous screen
  - Type "Control-C" to cancel the installation program
  - You can enter multiple items using commas to sepa-
  rate them.
For example: 1, 2, 3
Would you like to continue with installation? [Yes]:
```

Hit the Enter key; i.e., take the default [Yes].

Also note that pressing Ctrl+B to go back can be useful; however, there are times when you cannot go back because the previous selection has been committed.

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
BY INSTALLING THIS SOFTWARE YOU ARE CONSENTING TO BE
  BOUND BY AND ARE BECOMING A PARTY TO THE AGREEMENT
  FOUND IN THE LICENSE.TXT FILE. IF YOU DO NOT AGREE TO
  ALL OF THE TERMS OF THIS AGREEMENT, PLEASE DO NOT
  INSTALL OR USE THIS SOFTWARE.
Do you agree to the license terms? [No]: y
```

As long as you agree with the terms, enter y.

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
Choose an installation type:
1. Express installation
Allows you to quickly install the servers using the most
  common options and pre-defined defaults. Useful for
  quick evaluation of the products.
2. Typical installation
Allows you to specify common defaults and options.
3. Custom installation
Allows you to specify more advanced options. This is
  recommended for experienced server administrators
  only.
To accept the default shown in brackets, press the Enter
  key.
Choose an installation type [2]:
```

Hit the Enter key; i.e., take the default [2]

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
This program will extract the server files and install
  them into a directory you specify. That directory is
  called the server root in the product documentation
  and will contain the server programs, the Administra-
  tion Server, and the server configuration files.
To accept the default shown in brackets, press the Enter
  key.
Install location [/usr/iplanet/servers]: /iplanet/iws60
```

You can put the server wherever you have disk space; I'm going to choose a common top level of /iplanet.

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
iPlanet Web Server components:
Components with a number in () contain additional sub-
  components which you can select using subsequent
  screens.
1. iPlanet Web Server, Enterprise Edition (5)
Specify the components you wish to install [All]:
```

Hit the Enter key; i.e., take the default [All].

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
iPlanet Web Server, Enterprise Edition components:
Components with a number in () contain additional sub-
  components which you can select using subsequent
  screens.
1. Server Core
2. Java Runtime Environment
3. Java Support
4. SNMP Support
Specify the components you wish to install [1, 2, 3, 4]:
1,2,3
```

Enter [1, 2, 3, 4]. Note, this book looks at only 1, 2, and 3; 4 is optional.

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
Enter the fully qualified domain name of the computer on
  which you're installing server software. Using the
  form <hostname>.<domainname>
Example: eros.airius.com.
To accept the default shown in brackets, press the Enter
  key.
Computer name [art.red.iplanet.com]:
```

Hit the Enter key; i.e., take the default [hostname].

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
Choose a UNIX user and group to represent the iPlanet
```

```
  server in the user directory. The iPlanet server will
  run as this user.
It is recommended that this user should have no privi-
  leges in the computer network system. The Administra-
  tion Server will give this group some permissions in
  the server root to perform server-specific operations.
If you have not yet created a user and group for the
  iPlanet server, create this user and group using your
  native UNIX system utilities.
To accept the default shown in brackets, press the
  Return key.
System User [nobody]:
System Group [nobody]:
```

Hit the Enter key; i.e., take the default [nobody].

Note that the web instances can be run as users other than [nobody].

```
Sun-Netscape Alliance
iPlanet Web Server Installation/Uninstallation
-------------------------------------------------------
The iWS Administration Server is separate from the other
  web servers on the system and should not be confused
  with the Mission Control Admin Server. It is recom-
  mended that the iWS Administration Server run with a
  different user id than those used by the other web
  servers on the machine.
The Administration Server user is the only user able to
  write web server configuration files. If the iWS
  Administration Server is run as "root", the adminis-
  tration GUI can be used to start and stop web servers.
Run iWS Administration Server as [root]:
```

Hit the Enter key; i.e., take the default [root].

```
Sun-Netscape Alliance
iPlanet Web Server Installation/Uninstallation
-------------------------------------------------------
The iWS Administration Server requires its own adminis-
  trative user name and password for GUI access. When
  you access the iWS Administration Server GUI, it will
  prompt you for the administrative user name and pass-
  word.
Please select a user name and password now.
iWS Admin Server User Name [admin]:
iWS Admin Server Password:
iWS Admin Server Password (again):
```

Hit the Enter key; i.e., take the default [admin]. Use *admin* for the Password.

Yes, *admin/admin* is easy to remember; however, do use a more creative password when you are in production.

```
Sun-Netscape Alliance
iPlanet Web Server Installation/Uninstallation
-------------------------------------------------------
The iWS Administration Server also listens to a differ-
  ent port (with restricted access). Pick a port number
  between 1024 and 65535 on which to run your iWS Admin-
  istration Server. It must be different than the Mis-
  sion Control Admin Port and your web server port.
iWS Admin Server Port [8888]: 4006
```

Enter 4006 or some other available port.

I try to use the same port for all my web server administration servers; this way I can remember what the port is at a later date.

```
Sun-Netscape Alliance
iPlanet Web Server Installation/Uninstallation
-------------------------------------------------------
Pick a port number between 1024 and 65535 on which to
  run your Web Server.
You should NOT use a port number on which you plan to
  run other servers.
Web Server Port [80]: 86
```

Enter 86 or some other available port.

This is a default web server instance. Once the installation is completed, we will use the Sun ONE Web Server Administration Server to remove this web server instance and then create (add) a new web server instance.

```
Sun-Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
  An LDAP Directory Server can be used to administer users
    and groups.
  Do you want to register this with an existing Directory
Server [No]:
```

If this question comes up, hit the Enter key; i.e., take the default [No].

The instructions to add a connection to a Sun ONE Directory Server from the Sun ONE Web Server administration system are in the next chapter.

```
Sun-Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
Enter content root for the Web Server.
Web Server Content Root [/iplanet/iws60/docs]:
```

Hit the Enter key; i.e., take the default [<install directory>/docs].

```
Sun-Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
Java support in the Web Server requires either a Java
  Runtime Environment (JRE) or a Java Development Kit
  (JDK) of version 1.2 or greater.
While a default JRE is provided, you may use any JDK
  with the Web Server.
Do you want to use your own JDK [No]:
```

Hit the Enter key; i.e., take the default [No]. This can always be changed later.

```
Sun Netscape Alliance
iPlanet Web Server Installation/Uninstallation
--------------------------------------------------------
Extracting Server Core...
Extracting Java Runtime Environment...
Extracting Java Support...
Extracting Search and Indexing Support...
Extracting SNMP Support...
Extracting Upgrade Files...
Server Core installed successfully.
Java Runtime Environment installed successfully.
Java Support installed successfully.
Search and Indexing Support installed successfully.
SNMP Support installed successfully.
Press Return to continue...
```

Hit the Enter key.

```
Go to /iplanet/iws60 and type startconsole to begin man-
  aging your servers.
```

Hit the <install directory>/startconsole to start the iPlanet Web Server administration system.

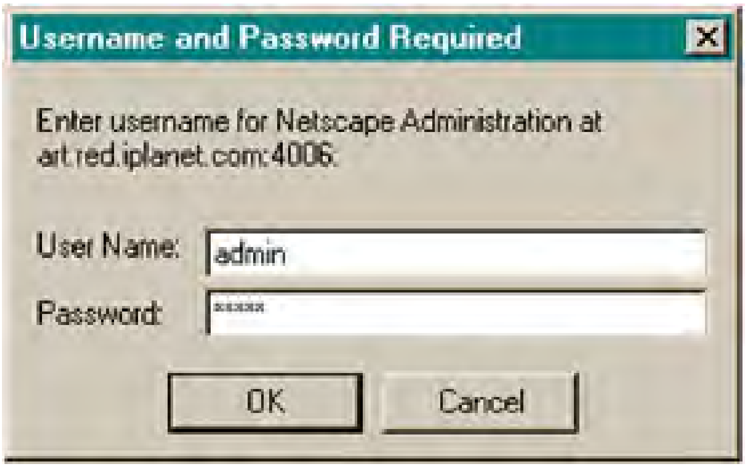
Testing the Sun ONE Web Server Administration Server

To start the administration server use the command /iplanet/iws60/https-admserv/start or use startconsole. This will run /iplanet/iws60/https-admserv/start and launch a web browser to connect to the Sun ONE Web Server Administration Server:

```
root[sh]@art#  /iplanet/iws60/startconsole
iPlanet-WebServer-Enterprise/4.1SP6 BB1-02/12/2001 11:27
startup: listening to http://art.red.iplanet.com, port
  4006 as root
Launching browser....
```

If you choose not to use startconsole, start up a web browser and connect to the administration server using the URL *http://<hostname>:4006*. For example, *http://art.red.iplanet.com:4006*. The Sun ONE Web Administration Server will return a prompt for the administrator's user ID and password, as in Figure 4.3. Enter *admin* into the box for *User Name* and *admin* into the box for the *Password*.

Figure 4.3. Administration prompt.


Username and Password Required

Enter username for Netscape Administration at art.red.iplanet.com:4006.

User Name: admin

Password: *****

If you forget the password, remove the password from the password file; for example:

```
# cd /sunone/ws6/https-admserv/config/admpw
# cat org.admpw
admin:{SHA}0DPiKuNIrrVmD8IUCuw1hQxNqZc=
# vi admpw
```

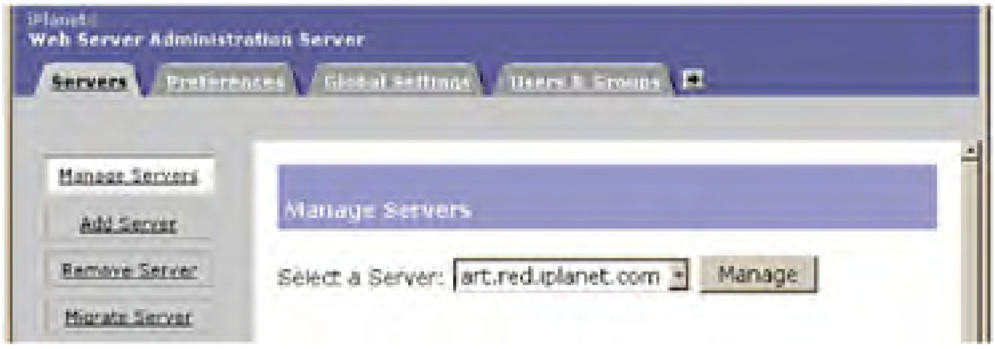
Change the file to look like

```
# cat admpw
admin:
```

Then, when logging in, use *admin* for the user ID and leave the password blank. Once logged in, reset the password (see next chapter).

The main Sun ONE Web Server Administration Server web page appears (Figure 4.4).

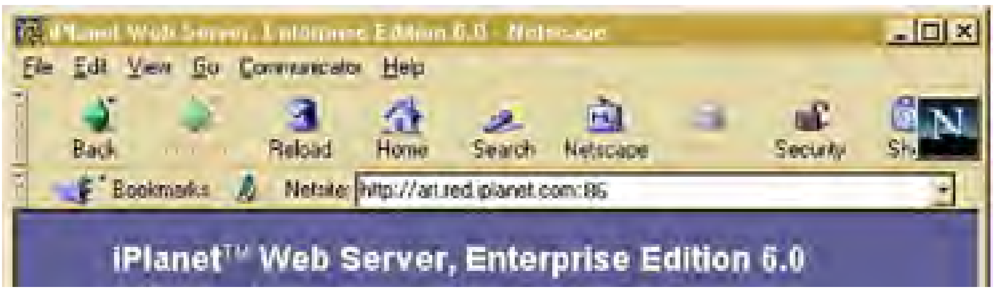
Figure 4.4. The Sun ONE Web Server Administration Server main web page.



Testing the Sun ONE Web Server Default Web Server

Figure 4.5 shows the installed default web site. From a browser, go to the URL *http://<hostname>:86*.

Figure 4.5. Default web site that was installed.


iPlanet Web Server, Enterprise Edition 6.0 - Netscape
File Edit View Go Communicator Help
Back
Reload
Home
Search
Netscape
Security
Bookmarks
Netsite: http://art.red.iplanet.com:86
iPlanet™ Web Server, Enterprise Edition 6.0

© 2001 Sun Microsystems, Inc. Some preexisting portions
© 2001 Netscape Communications Corp. All Rights Reserved

Success! Now that you have successfully installed a Sun ONE Web Server, go to the next chapter to configure and use this web server.

Uninstalling the Server

It is very easy to uninstall the server—stop the processes and remove the install directory and all subdirectories; for example, enter the command `rm -r /iplanet/iws60`. After the command completes, the Sun ONE Web Server will be completely removed/ uninstalled.

Reference Documentation

The Sun Microsystems web site at *http://docs.sun.com* has a considerable amount of documentation. The link for the Sun ONE Web Server is *http://docs.sun.com/db/prod/sunone*.

[Team LiB]   PREVIOUS NEXT

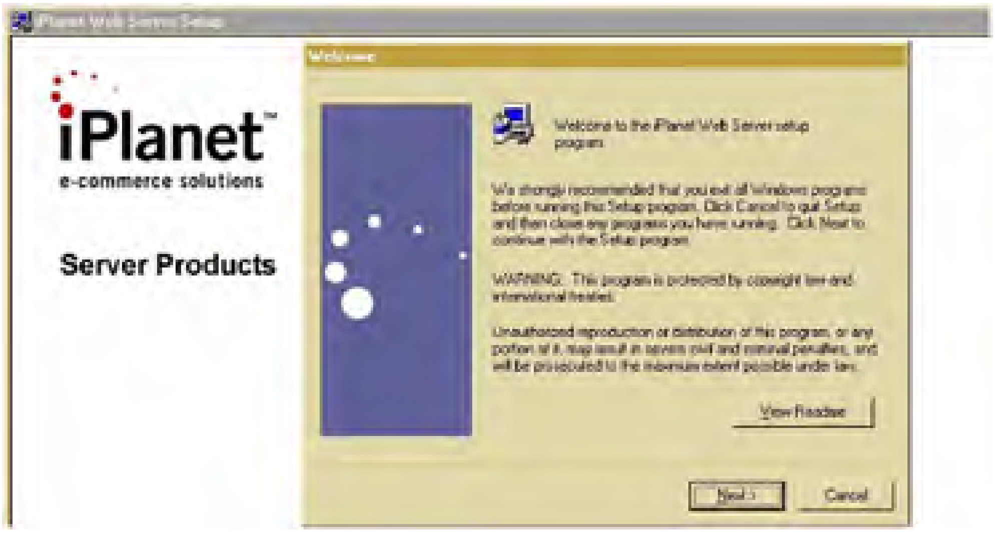
Windows Installation

The opening to this chapter has a description of the components that will be installed and gives an overview of a sample installation. This section contains the steps for installing the Sun ONE Web Server on a computer running Windows NT or Windows 2000.

The Sun ONE Web Server for Windows is included on the DVD with this book, or you can download the Sun ONE Web Server, with the latest service packs, from *www.sun.com*.

Run the setup executable file to start the installation. Zipped files will be unpacked, and then the welcome window will appear (Figure 4.6).

Figure 4.6. Sun ONE Web Server installation welcome page.



Welcome to the iPlanet Web Server setup program.

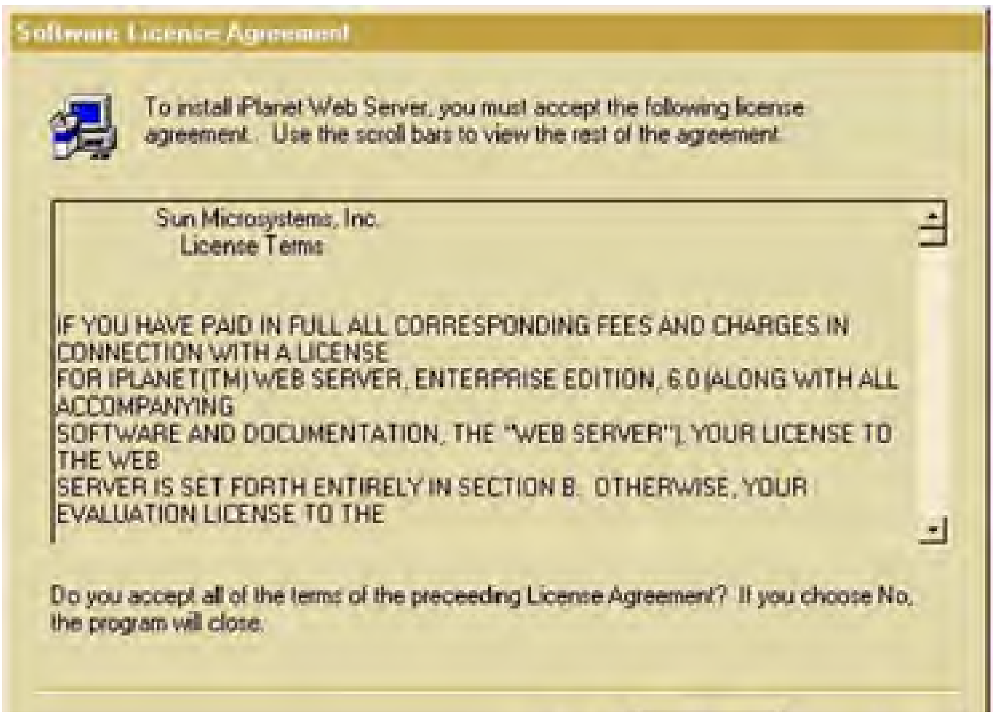
We strongly recommended that you exit all Windows programs before running this Setup program. Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

Click the *Next* button, and the *Software License Agreement* opens (Figure 4.7).

Figure 4.7. License agreement.



To install iPlanet Web Server, you must accept the following license agreement. Use the scroll bars to view the rest of the agreement.

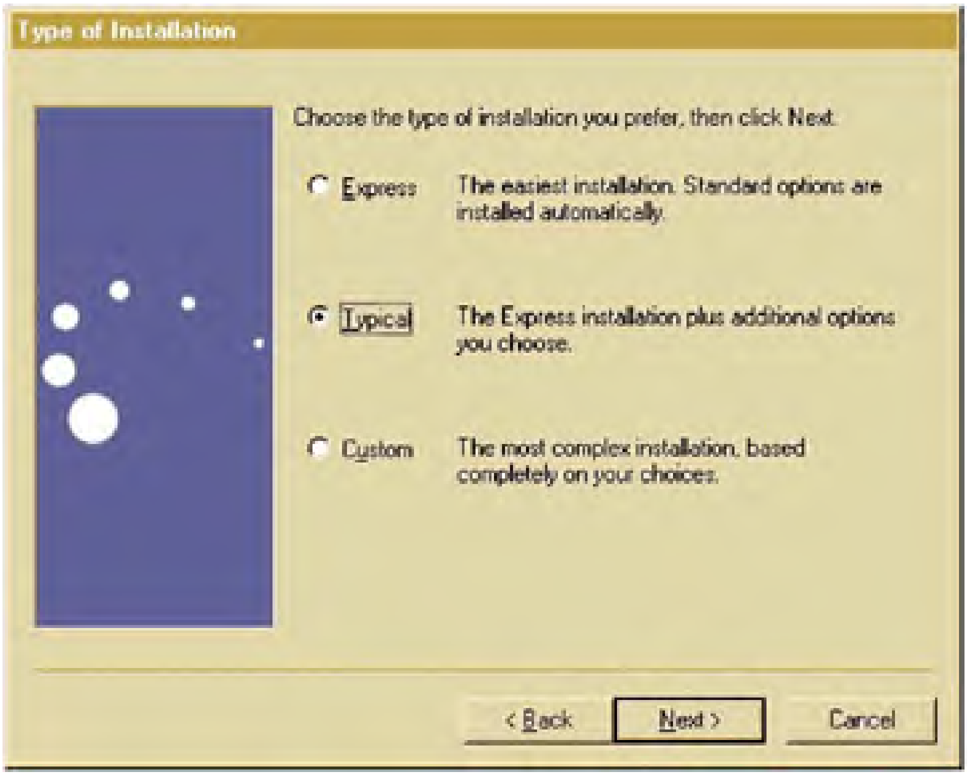
Sun Microsystems, Inc.
License Terms

IF YOU HAVE PAID IN FULL ALL CORRESPONDING FEES AND CHARGES IN CONNECTION WITH A LICENSE
FOR IPLANET(TM) WEB SERVER, ENTERPRISE EDITION, 6.0 (ALONG WITH ALL ACCOMPANYING
SOFTWARE AND DOCUMENTATION, THE "WEB SERVER"), YOUR LICENSE TO THE WEB
SERVER IS SET FORTH ENTIRELY IN SECTION B. OTHERWISE, YOUR EVALUATION LICENSE TO THE

Do you accept all of the terms of the preceeding License Agreement? If you choose No, the program will close.

If you accept the terms of the agreement, click the *Yes* button. The next window—*Type of Installation*—is shown in Figure 4.8.

Figure 4.8. Installation type.



Type of Installation

Choose the type of installation you prefer, then click Next.

( ) Express — The easiest installation. Standard options are installed automatically.

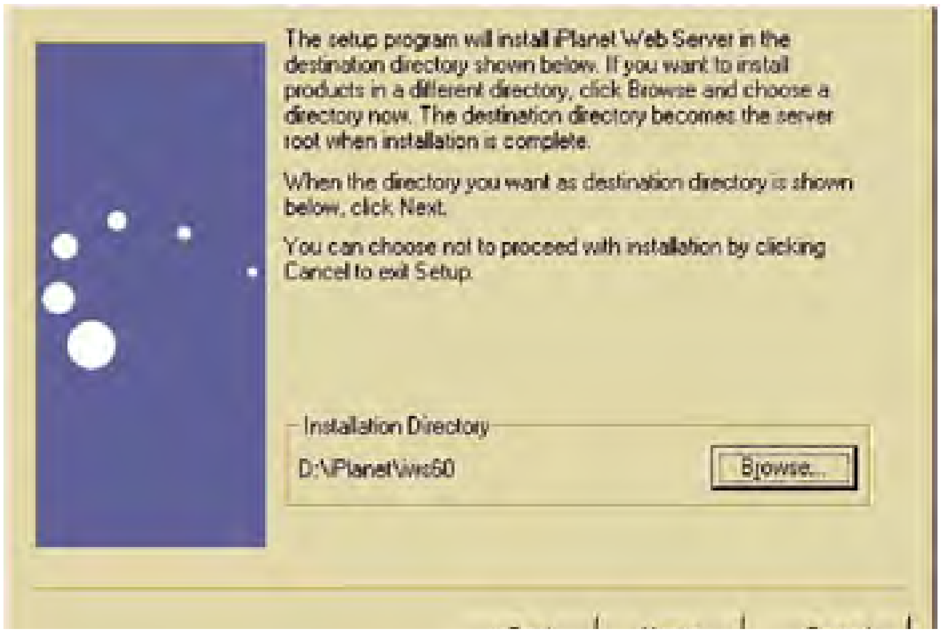
(•) Typical — The Express installation plus additional options you choose.

( ) Custom — The most complex installation, based completely on your choices.

A Sun professional service person told me to "do a typical install. Get it working. Then later you can make modifications." Wise words.

Use the default *Typical* and click *Next* to open the window in Figure 4.9.

Figure 4.9. Destination directory.

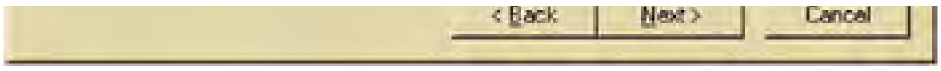


The setup program will install iPlanet Web Server in the destination directory shown below. If you want to install products in a different directory, click Browse and choose a directory now. The destination directory becomes the server root when installation is complete.

When the directory you want as destination directory is shown below, click Next.

You can choose not to proceed with installation by clicking Cancel to exit Setup.

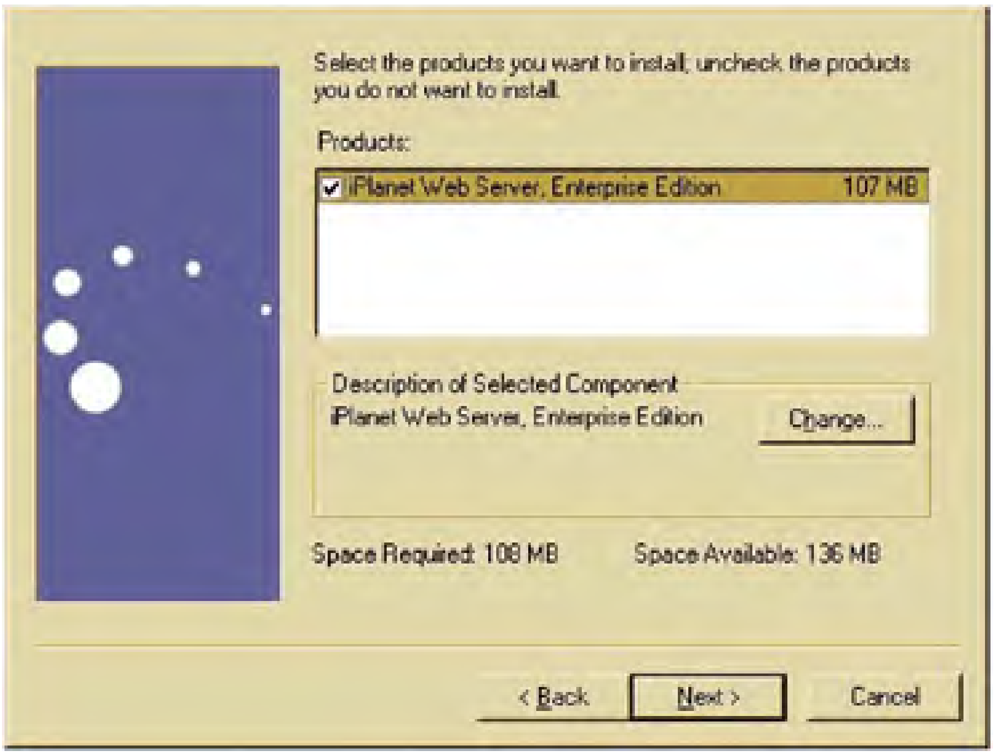
Installation Directory
D:\iPlanet\ws60



Normally the operating system and other programs are on the C drive. The D drive typically has more disk space and is usually where the work applications are. If you wish to install the application in the directory shown, click *Next*. Otherwise, click *Browse* and select your directory. Then click *Next*. Note that I'm using the directory name iplanet.

The next window (Figure 4.10) allows you to choose the components you wish to install. Also check to be sure you have enough disk space. Use the default and click *Next*.

Figure 4.10. Component selection and disk space availability.



Select the products you want to install; uncheck the products you do not want to install.

Products:

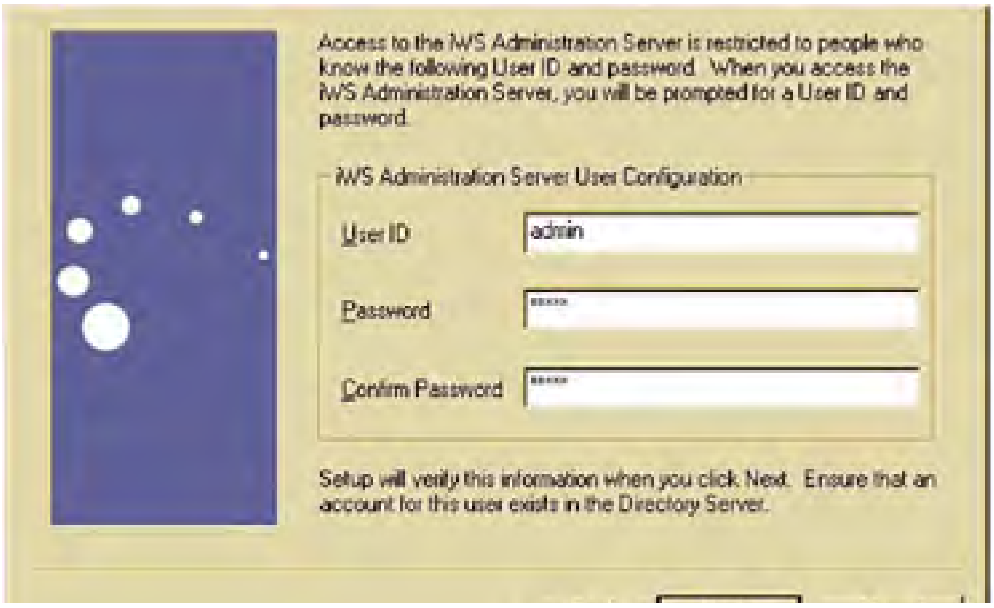
☑ iPlanet Web Server, Enterprise Edition 107 MB

Description of Selected Component
iPlanet Web Server, Enterprise Edition

Space Required: 108 MB   Space Available: 136 MB

The Sun ONE Web Server Administration Server needs a special administration account. For now use user ID *admin* and password *admin* (Figure 4.11). Later, when the system is put into production, the password can be changed. Then click *Next*.

Figure 4.11. User ID and password selection.



Access to the iWS Administration Server is restricted to people who know the following User ID and password. When you access the iWS Administration Server, you will be prompted for a User ID and password.

iWS Administration Server User Configuration

User ID: admin

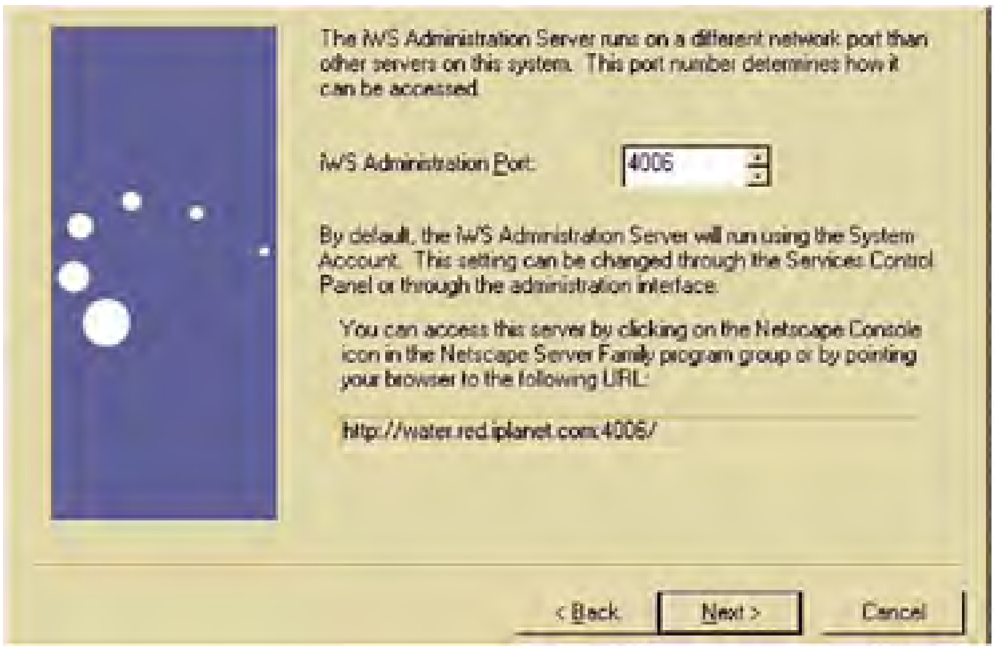
Password: *****

Confirm Password: *****

Setup will verify this information when you click Next. Ensure that an account for this user exists in the Directory Server.

When running, the administration server listens on a port (see Figure 4.12). Pick a port number you will remember; the port can be changed later. I suggest you use the same number every time you install a system. This will make it easier on your memory. The port I'm using here is *4006*; choose the port using the scroll arrows. Then click *Next*.

Figure 4.12. Port selection.



The iWS Administration Server runs on a different network port than other servers on this system. This port number determines how it can be accessed.

iWS Administration Port: 4006

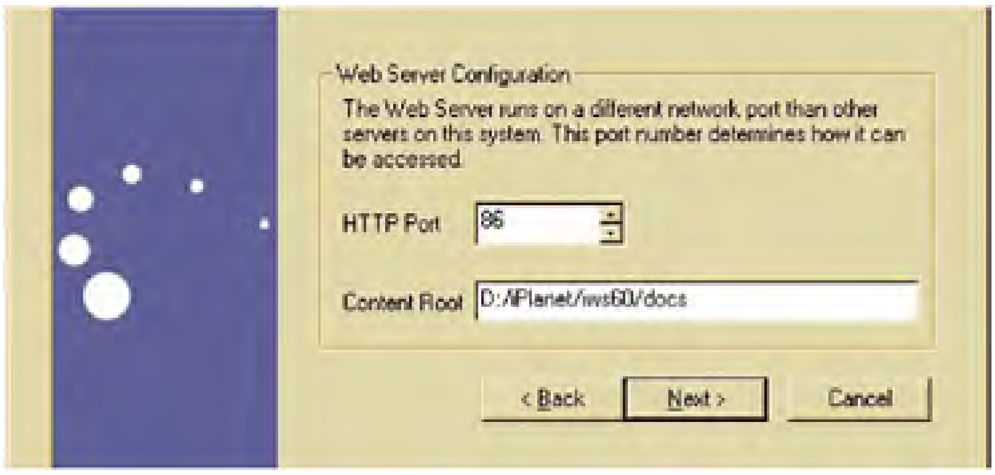
By default, the iWS Administration Server will run using the System Account. This setting can be changed through the Services Control Panel or through the administration interface.

You can access this server by clicking on the Netscape Console icon in the Netscape Server Family program group or by pointing your browser to the following URL:

http://water.red.iplanet.com:4006/

Next set up a default web instance that will listen to port *86* and that will get web page files from a directory; for example, *D:/iPlanet/ iws60/docs* (Figure 4.13). Set *HTTP Port* to *86*; leave *Content Root* (document directory) as is. Then click *Next*. (I have chosen port 86 because I have another web server instance already on port 80.)

Figure 4.13. Default web instance.



Web Server Configuration

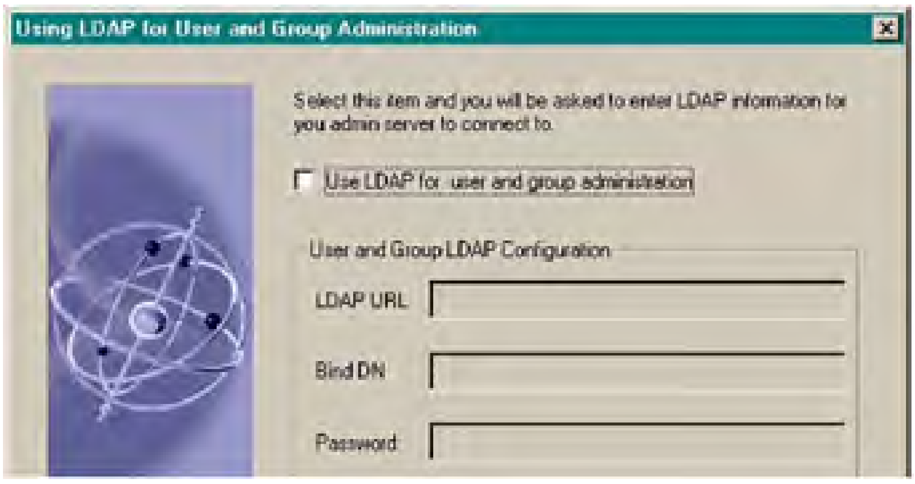
The Web Server runs on a different network port than other servers on this system. This port number determines how it can be accessed.

HTTP Port: 86

Content Root: D:/iPlanet/iws60/docs

My installation of iPlanet Web Server 6.0 did not ask me for LDAP directory information (Figure 4.14). If you are prompted for this information, do not select an LDAP directory server at this time. In Chapter 12 there are instructions to connect the web server to a Sun ONE Directory Server. For now, let's just get the server installed and running by clicking *Next*.

Figure 4.14. LDAP directory information (do not select directory now).



Using LDAP for User and Group Administration

Select this item and you will be asked to enter LDAP information for you admin server to connect to.

☐ Use LDAP for user and group administration

User and Group LDAP Configuration

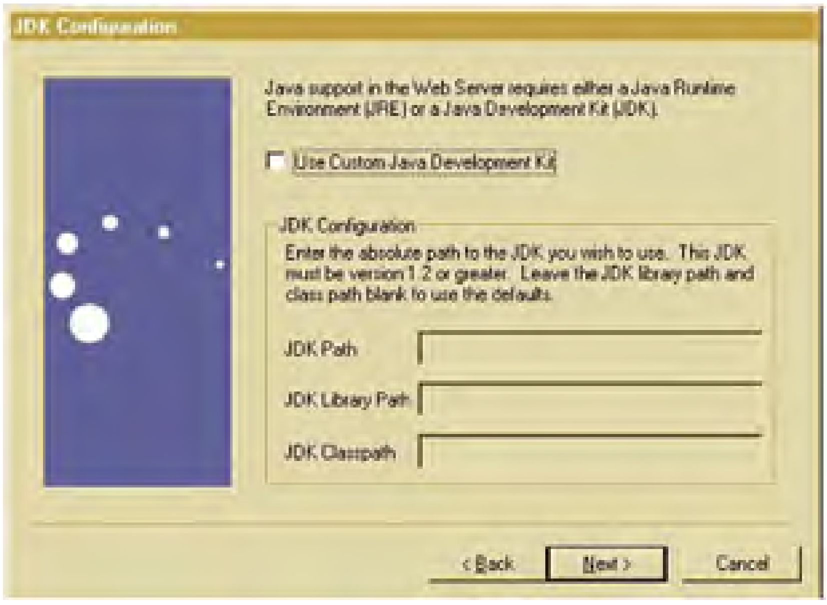
LDAP URL

Bind DN

Password

Now you are ready for Java Development Kit (JDK) configuration (Figure 4.15). By default, JRE is installed; this is all you need to run servlets. If you are going to be doing JSPs, you will need to install Java 1.2.2 (or higher). If you already have Java 1.2.2 installed, you can select *Use Custom Java Development Kit* and set the *JDK Path* or change to the configuration settings after the installation. For now, just click *Next*.

Figure 4.15. JDK configuration.



JDK Configuration

Java support in the Web Server requires either a Java Runtime Environment (JRE) or a Java Development Kit (JDK).

☐ Use Custom Java Development Kit

JDK Configuration

Enter the absolute path to the JDK you wish to use. This JDK must be version 1.2 or greater. Leave the JDK library path and class path blank to use the defaults.

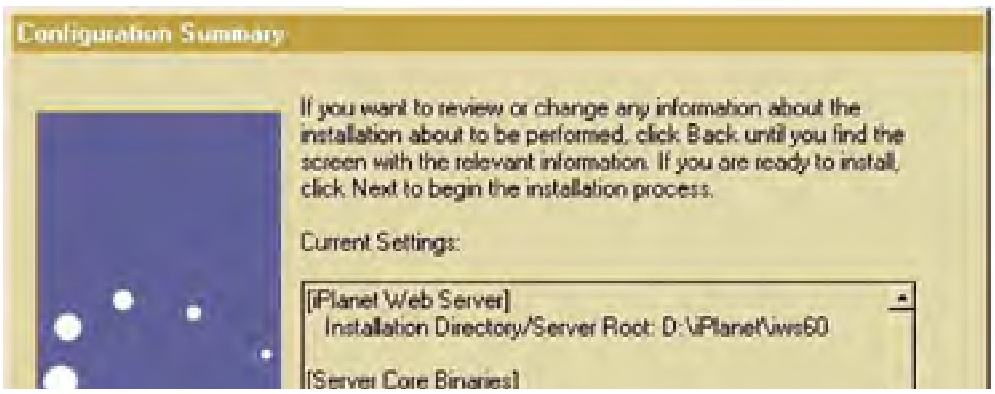
JDK Path

JDK Library Path

JDK Classpath

All right. To install the software, click *Install* in the window shown in Figure 4.16.

Figure 4.16. Time to finally install.

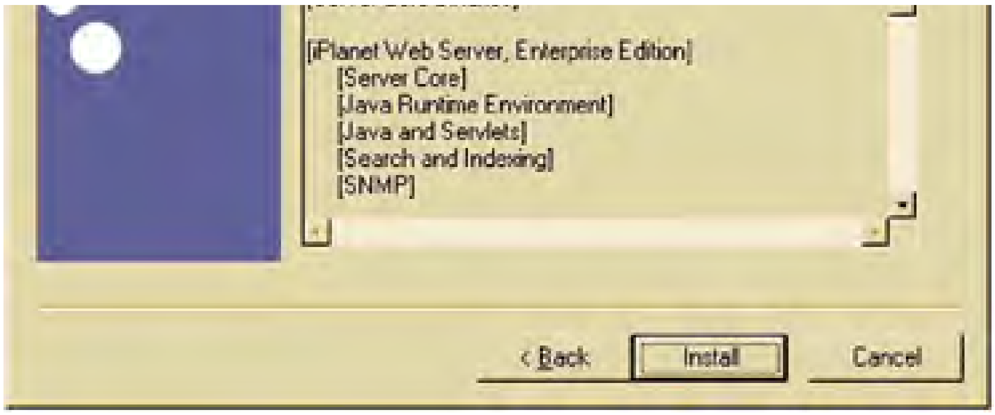


Configuration Summary

If you want to review or change any information about the installation about to be performed, click Back until you find the screen with the relevant information. If you are ready to install, click Next to begin the installation process.

Current Settings:

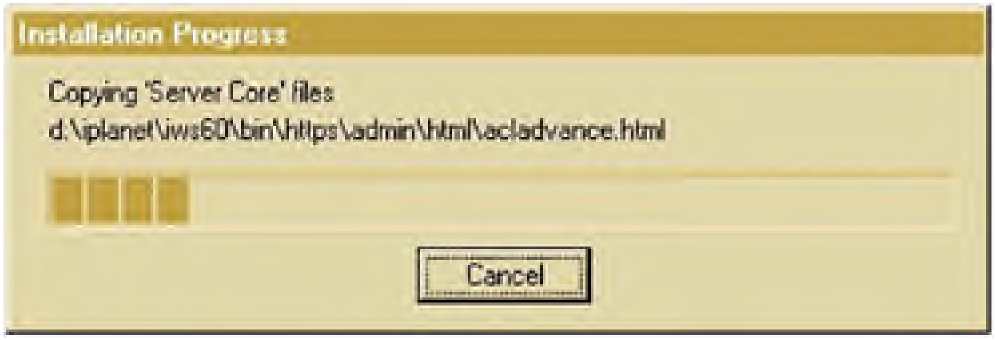
```
[iPlanet Web Server]
   Installation Directory/Server Root: D:\iPlanet\iws60

[Server Core Binaries]
```



[iPlanet Web Server, Enterprise Edition]
    [Server Core]
    [Java Runtime Environment]
    [Java and Servlets]
    [Search and Indexing]
    [SNMP]

Files will be installed—their installation process will be shown in the window in Figure 4.17.

Figure 4.17. Installation process.

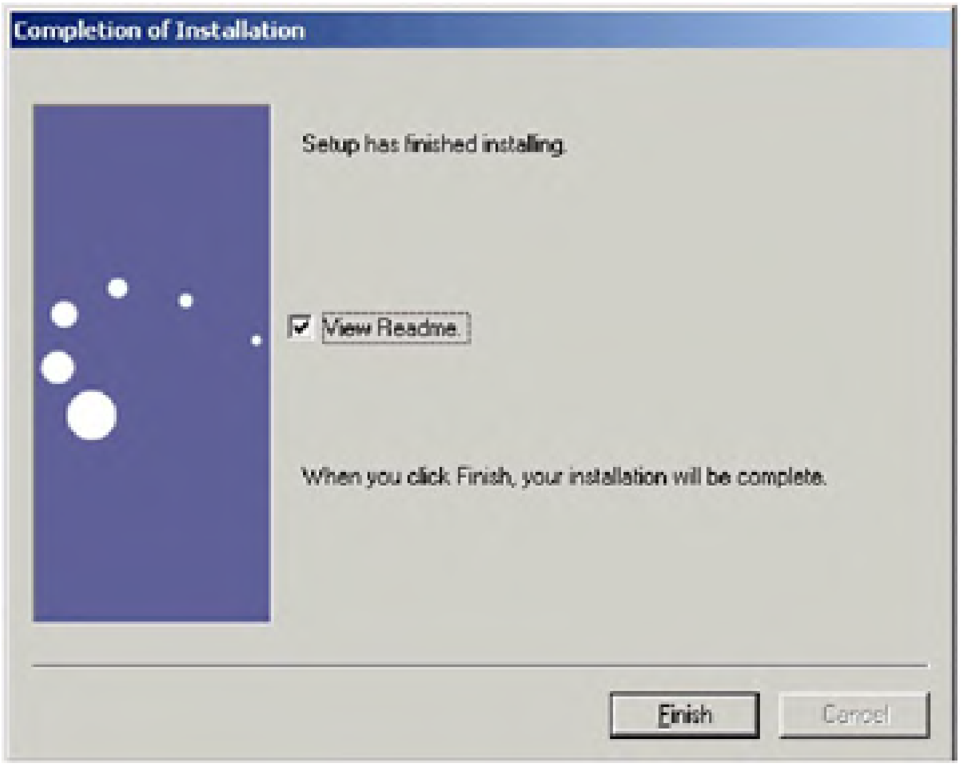


Installation Progress

Copying 'Server Core' files
d:\iplanet\iws60\bin\https\admin\html\acladvance.html

The last screen for a Windows 2000 installation is the completion screen (Figure 4.18). Your final move is to click *Finish*.

Figure 4.18. Final screen—Windows 2000.



Completion of Installation

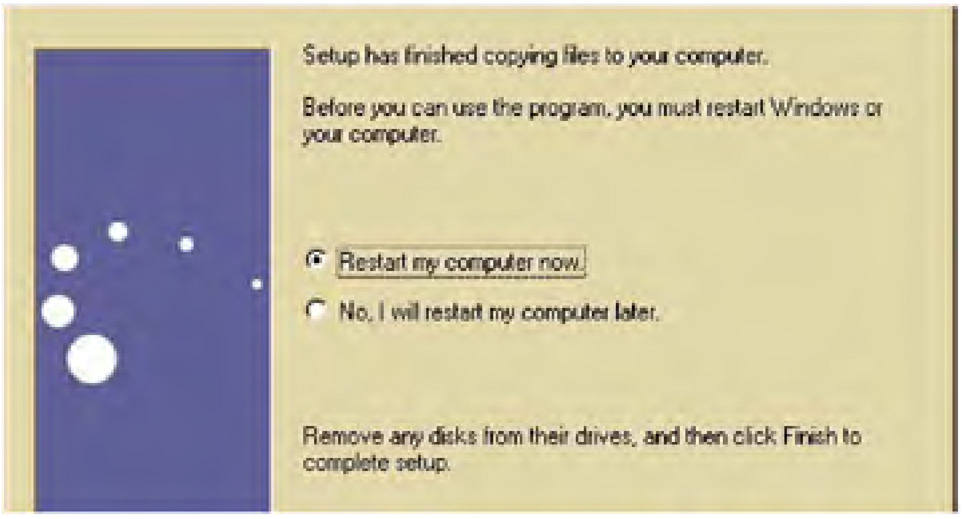
Setup has finished installing.

☑ View Readme.

When you click Finish, your installation will be complete.

The final screen for a Windows NT installation is shown in Figure 4.19. Windows NT requires a restart. Once you've restarted your computer, you're finished—the installation is now complete! QED (quite easily done).

Figure 4.19. Final screen—Windows NT installation.



Setup has finished copying files to your computer.

Before you can use the program, you must restart Windows or your computer.

(●) Restart my computer now.

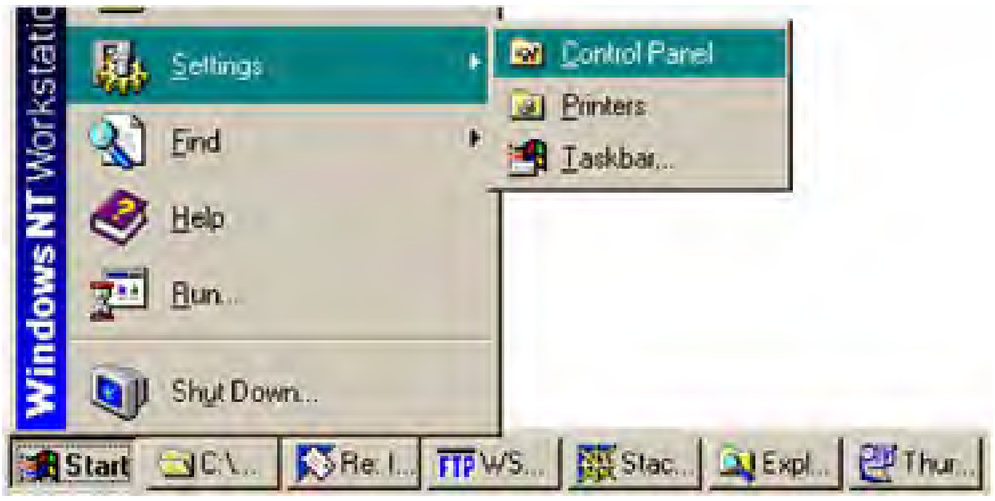
( ) No, I will restart my computer later.

Remove any disks from their drives, and then click Finish to complete setup.

Now restart the computer; then test it and see it work.

First, check for new services. Figure 4.20 shows the sequence to use to get to the Windows *Control Panel*: *Start Button / Settings / Control Panel.* When the *Control Panel* comes up, click on *Services*. In Windows 2000, when the *Control Panel* comes up, click on *Administrative Tools*; then click on *Services*. The *Services* window is now displayed.

Figure 4.20. Opening the Windows Control Panel.



In the *Services* window, you will see the Sun ONE Web Server processes:

- *iWS (water.red.iplanet.com)* is the Sun ONE Web Server instance.
- *iWS Administration Server (6.0)* is the Sun ONE Web Server Administration Server 6.0.

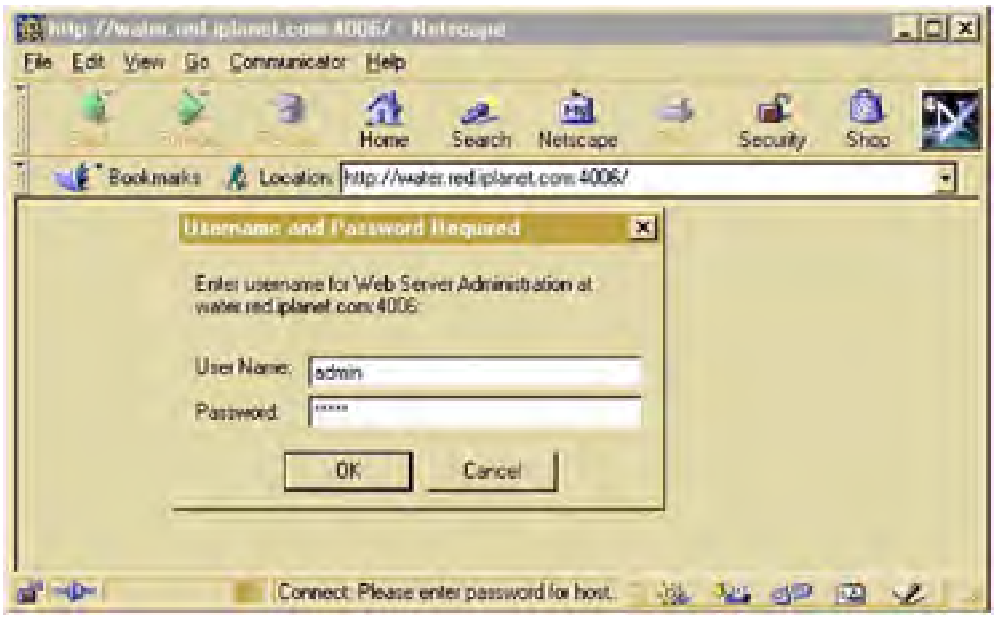
Testing the Sun ONE Web Server Administration Server

The default setting is *Automatic*; i.e., the Sun ONE Web Server processes are set to start up automatically. You may change the setting to suit your own preference. I set mine to *Manual* because I want the processes running only when I am testing. You can also use the Services window to start and stop the web server processes. If the administration server is not running, start it.

Using a browser, connect to the administration server *http://water.red.iplanet.com:4006*.

As you can see in Figure 4.21, you will be prompted for the administrator's user ID and password. Use *admin* for both the *User Name* and the *Password*, and click *OK*.

Figure 4.21. Connecting to the administration server.



http://water.red.iplanet.com:4006/ - Netscape

http://water.red.iplanet.com:4006/

Username and Password Required

Enter username for Web Server Administration at water.red.iplanet.com:4006:

User Name: admin

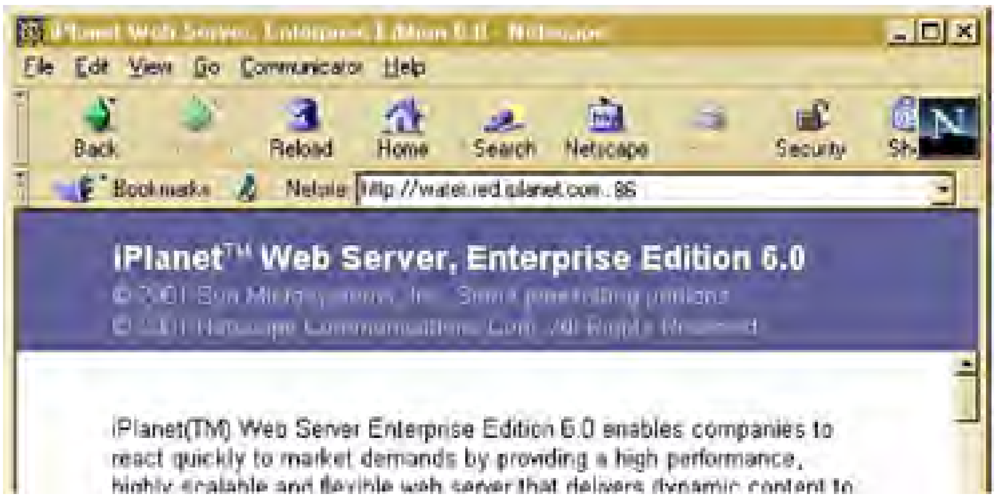
Password: *****

Connect: Please enter password for host.

Testing the Sun ONE Web Server Default Web Server

Go to the URL *http://water.red.iplanet.com:86* (Figure 4.22).

Figure 4.22. Installed default web site.



iPlanet Web Server, Enterprise Edition 6.0 - Netscape

http://water.red.iplanet.com:86

iPlanet™ Web Server, Enterprise Edition 6.0

© 2001 Sun Microsystems, Inc. Some preexisting portions

© 2001 Netscape Communications Corp. All Rights Reserved.

iPlanet(TM) Web Server Enterprise Edition 6.0 enables companies to react quickly to market demands by providing a high performance, highly scalable and flexible web server that delivers dynamic content to

Success! Now that you have sucessfully installed a Sun ONE Web Server, go to the next chapter to configure and use this web server.

Uninstalling the Server

Uninstalling is very easy—run the uninstall option from the Windows *Control Panel Add/Remove* option. After the uninstall has been run, remove the installation directory. At this point, the Sun ONE Web Server has been completely removed/uninstalled.

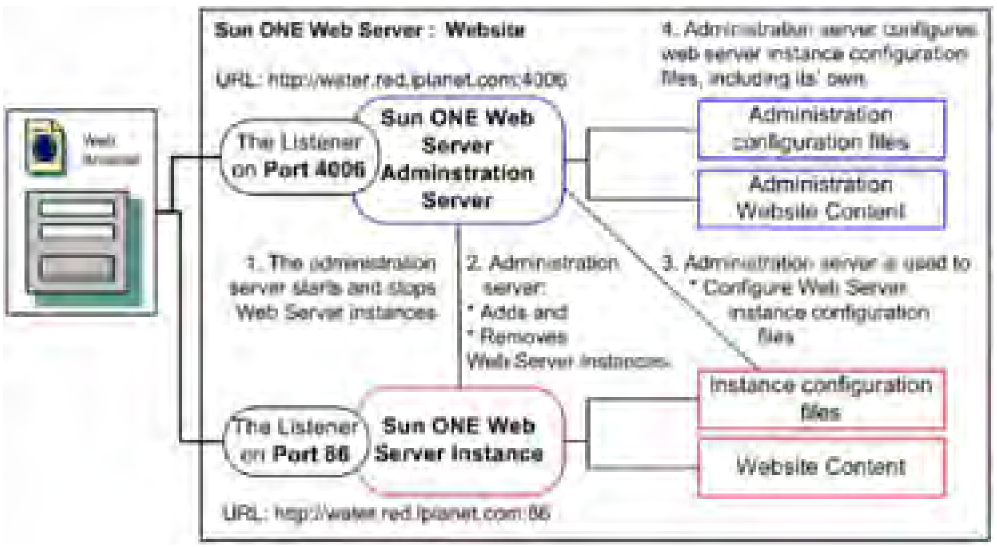
Reference Documentation

The Sun Microsystems web site *http://docs.sun.com* has a considerable amount of documentation. The link for the Sun ONE Web Server is *http://docs.sun.com/db/prod/sunone*.

Chapter 5. Starting, Stopping, and Testing the Sun ONE Web Server

After installing new software, you want to be able to start it, stop it, and monitor the processes. During the installation of the Sun ONE Web Server, two web servers are installed (Figure 5.1), but neither server has been started. In this chapter, the server processes will be started, tested, monitored, and stopped. Learning the steps discussed in this chapter is the beginning of administering a web site. Also, the four functions of the Sun ONE Web Server Administration Server mentioned in Figure 5.1 are covered in this chapter.

Figure 5.1. Installed web servers.


Sun ONE Web Server : Website
URL: http://water.red.iplanet.com:4006
4. Administration server configures web server instance configuration files, including its' own.
Web Browser
The Listener on Port 4006
Sun ONE Web Server Adminstration Server
Administration configuration files
Administration Website Content
1. The administration server starts and stops Web Server instances
2. Administration server:
* Adds and
* Removes
Web Server instances
3. Administration server is used to
* Configure Web Server instance configuration files
Instance configuration files
The Listener on Port 86
Sun ONE Web Server Instance
Website Content
URL: http://water.red.iplanet.com:86

Starting and Stopping the Web Servers On UNIX

Start the Sun ONE Web Server administration server.

```
# /iplanet/iws60/https-admserv/start
iPlanet-WebServer-Enterprise/6.0SP2 B11/13/2001 00:49
warning: daemon is running as super-user
[LS] http://art.red.iplanet.com, port 8888 ready to
  accept requests
startup: server started successfully
#
```

An alternative start command is

```
cd /iplanet/iws60
./startconsole
```

Note: The startconsole command runs the start script and then tries to launch a web browser on the same computer running the web server. Therefore, use the start command unless you are on the same computer as the web server computer.

Monitor the web server processes.

```
# ps -ef | grep iws60
ns-httpd -d /iplanet/iws60/https-admserv/config
root 935 872 0 19:08:18 pts/7 grep webserver1
root 930 1 19:04:30 ? 0:00
      ./uxwdog -d /iplanet/iws60/https-admserv/config
root 931 930 0 19:04:30 ? 0:00
      ns-httpd -d /iplanet/iws60/https-admserv/config
root 932 931 0 19:04:31 ? 0:02
      ns-httpd -d /iplanet/iws60/https-admserv/config
```

Notes about the listing:

- Process 932 ns-httpd is the Sun ONE Web Server Administration Server web instance process. This process is a child of 931.
- Process 931 ns-httpd is a parent process to the iPlanet Web Server Administration server process. This process has two functions: to start the actual web instance; to watch/monitor the child process and, if the child process stops running, to restart the process.
- Process uxwdog is the watchdog process to restart the parent web server instance process (process 931) if it stops.

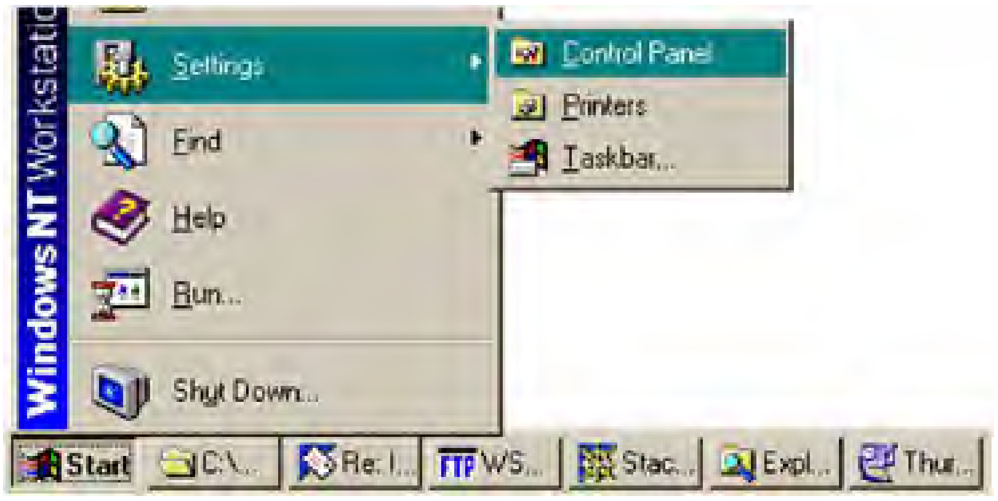
The shutdown/stop command is in the same directory as the start command:

```
#/iplanet/iws60/https-admserv/stop
shutdown: server shut down
```

On Windows

First, go to the *Services* window from the Windows *Control Panel*. Figure 5.2 shows the sequence to get to the Windows *Control Panel*: *Start* button / *Settings* / *Control Panel.* When the *Control Panel* comes up, click on *Services*.

Figure 5.2. Opening the Windows Control Panel.



On Windows 2000, when the *Control Panel* comes up, click on *Administrative Tools*; then click on *Services*.

In the *Services* window, you will see the Sun ONE Web Server processes:

- *iWS (water.red.iplanet.com)* is the Sun ONE Web Server instance.
- *iWS Administration Server (6.0)* is the Sun ONE Web Server Administration Web Server 6.0.

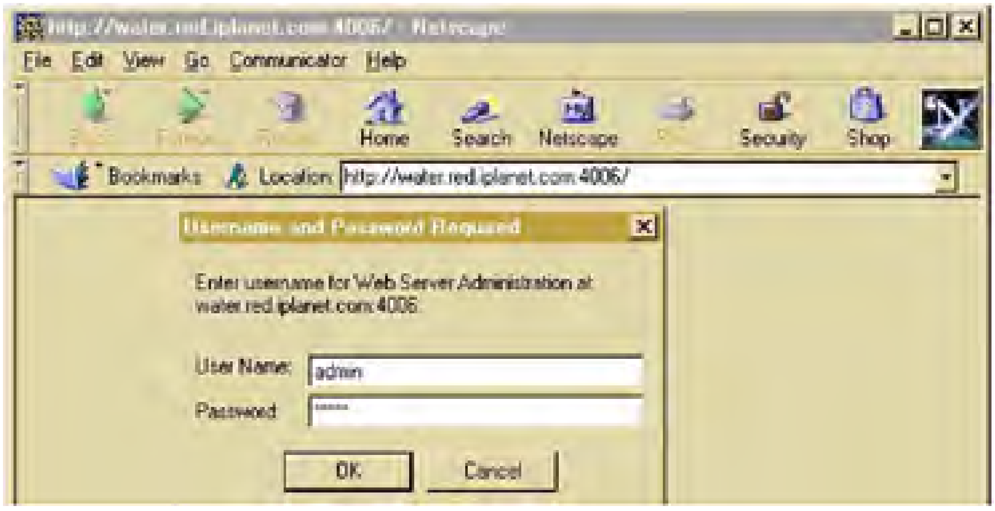
From the *Services* window, you can start and stop the Sun ONE Web Server processes by selecting the process and clicking *Start* to start and *Stop* to stop the process.

During the installation, the processes are installed and set up to start up automatically during a reboot. For development installations of software, from the *Services* window I set the process to *Manual*, which means they do *not* start up automatically when I reboot the computer. I prefer this because I do not want the extra processes running unless I need them. Even for production systems, I set the Sun ONE Administration Web Server to *Manual* because I want this process running only when I need it. This becomes an extra level of security.

Sun ONE Web Server Administration Server

If the administration server is not running, start it now. Then, using a web browser, connect to the administration server. Note: The port values are based on the installation. If you followed the previous installation instructions, the user ID is *admin* and the password is *admin* (Figure 5.3).

Figure 5.3. Prompt for administration user ID and password.



http://water.red.iplanet.com:4006/ - Netscape

Location: http://water.red.iplanet.com:4006/

Username and Password Required

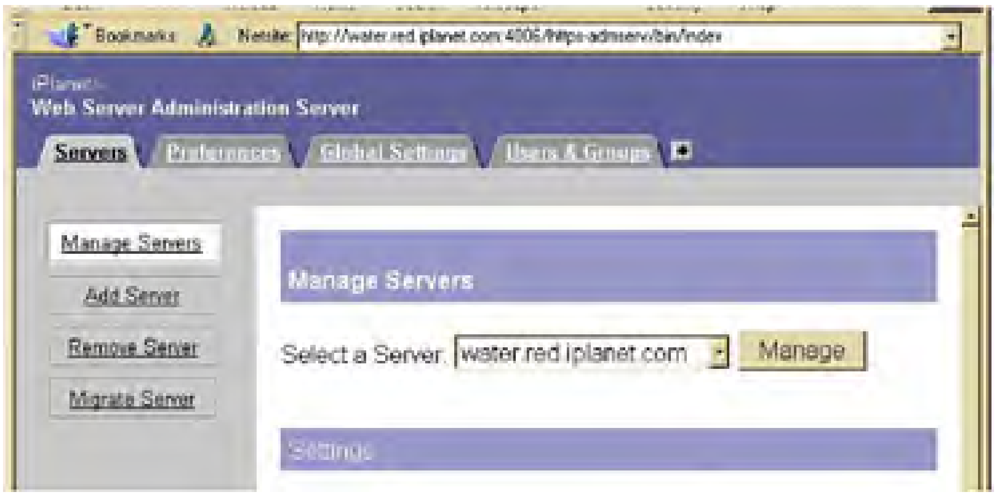
Enter username for Web Server Administration at water.red.iplanet.com:4006.

User Name: admin

Password: *****

Now the main/top web administration web page appears, from which web server instances are managed and configured (Figure 5.4).

Figure 5.4. Main web administration web page.



Netsite: http://water.red.iplanet.com:4006/https-admserv/bin/index

iPlanet Web Server Administration Server

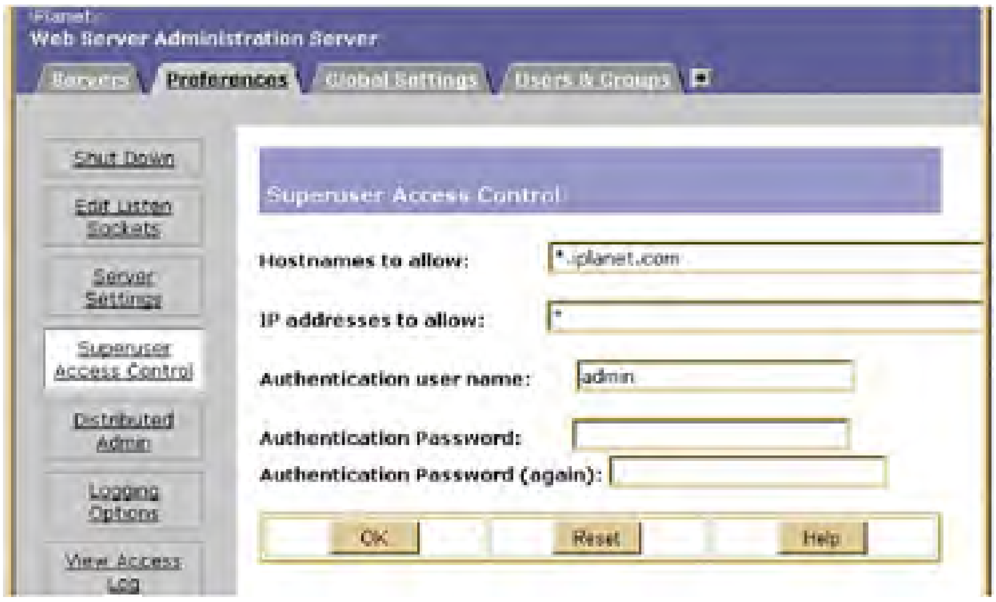
Manage Servers

Select a Server: water.red.iplanet.com

Administering the Administrator Account

The administrator account is a special account that is maintained by the Sun ONE Web Server administration system. From the main administration web page, click on the *Preferences* tab to change the *admin* password. The administrator's password is maintained from this web page (Figure 5.5).

Figure 5.5. Superuser administration account.


iPlanet
Web Server Administration Server

Servers | Preferences | Global Settings | Users & Groups

Shut Down
Edit Listen Sockets
Server Settings
Superuser Access Control
Distributed Admin
Logging Options
View Access Log

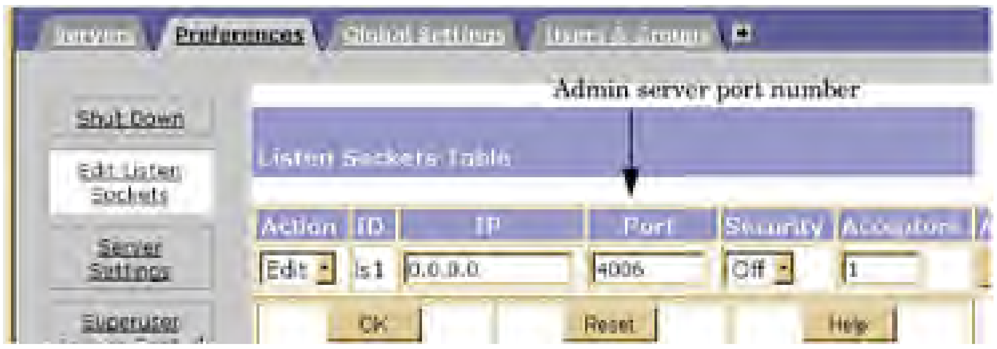
Superuser Access Control

Hostnames to allow:	*.iplanet.com
IP addresses to allow:	*
Authentication user name:	admin
Authentication Password:	
Authentication Password (again):	

OK | Reset | Help

Click the *Edit Listen Sockets* menu option (on the left) to view or change the web server administration server listener port (Figure 5.6). For now, do not change this value.

Figure 5.6. Administration server listener port.



Servers | Preferences | Global Settings | Users & Groups

Shut Down
Edit Listen Sockets
Server Settings
Superuser

Listen Sockets Table

Action	ID	IP	Port	Security	Acceptors
Edit	ls1	0.0.0.0	4006	Off	1

OK | Reset | Help

Managing Web Server Instances

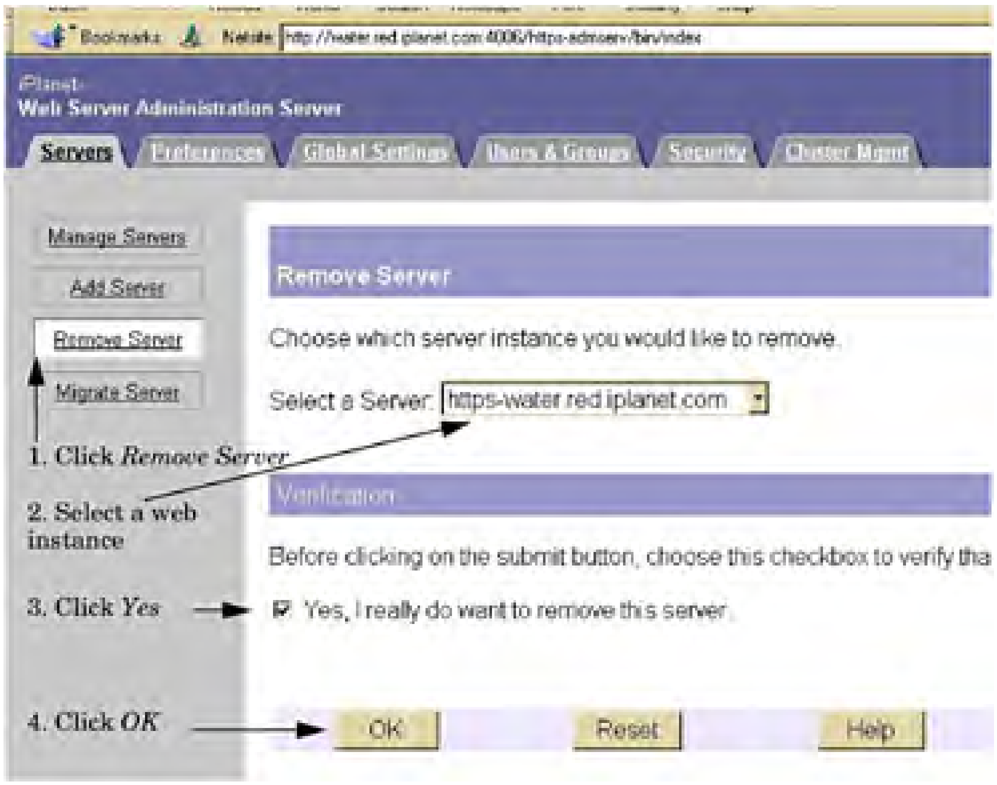
When in development, I often modify the configuration settings of a web server instance. To reset the configurations back to the originally installed settings, I remove the instance and add a new instance. With the new instance, I start testing all over again. This is easier than reinstalling the complete web server.

Another situation that calls for managing web server instances is in the case where there are multiple web applications. To keep the applications separated, I have one web server instance per application. Then, when needed, I can reset one web instance without affecting any of the other applications.

Removing a Web Instance

From the main web page, click the *Servers* tab; then click *Remove Server*. This displays the web page used to remove a web server instance (Figure 5.7).

Figure 5.7. Steps to remove a web server instance.



Choose which server instance you would like to remove.

Select a Server: https-water.red.iplanet.com

Before clicking on the submit button, choose this checkbox to verify tha

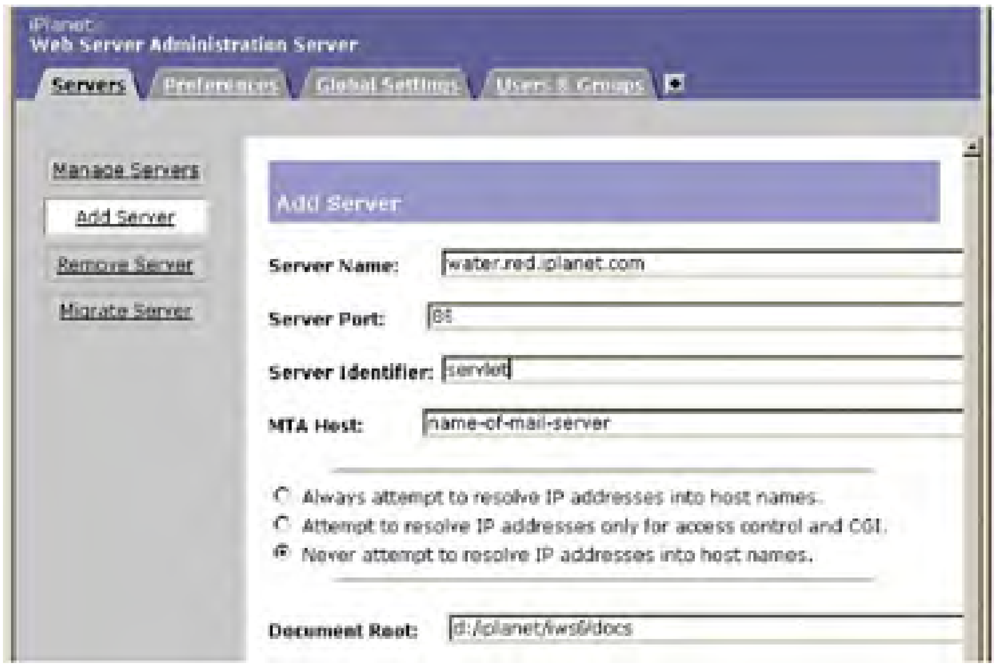
☑ Yes, I really do want to remove this server.

Adding a New Web Server Instance

To add a new web server instance, follow these steps:

1. Click on left-hand menu option *Add Server*. The web page in Figure 5.8 appears.

Figure 5.8. Administration web page to add server instances.



Server Name: water.red.iplanet.com

Server Port: 86

Server Identifier: servlet

MTA Host: name-of-mail-server

○ Always attempt to resolve IP addresses into host names.
○ Attempt to resolve IP addresses only for access control and CGI.
● Never attempt to resolve IP addresses into host names.

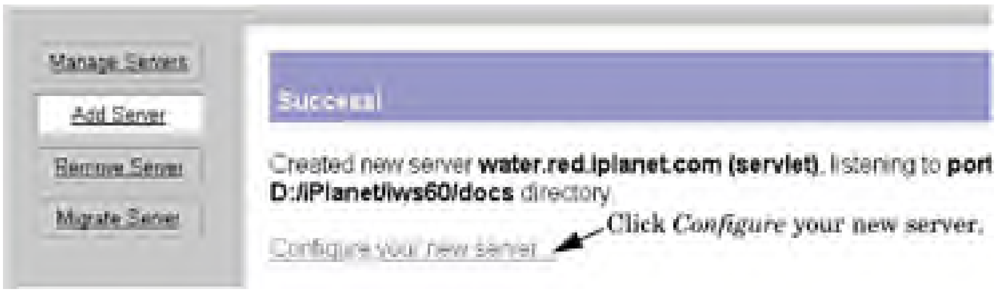
Document Root: d:/iplanet/iws6/docs

2. Your computer's hostname appears in *Server Name*.
3. Enter the port that the new web server instance listens to: port *86*.

4. Enter the *Server Identifier* name of *servlet*. This is to identify the web server instance for administration purposes. I choose names that represent the function for which the web server instance is used. In this case I have chosen *servlet* because this web server instance is for testing servlets.
5. On UNIX, you will see the username under which the web server instance process will be run. I normally create a group called *igroup* (for *Internet* group) and a user in the group named *iuser*. Then *iuser* is used here.
6. Document Root is the top file directory location from whichthe web server instance gets files.
7. Click *Ok*.

After clicking *OK*, the instance is created (Figure 5.9).

Figure 5.9. Web server instance has been created.

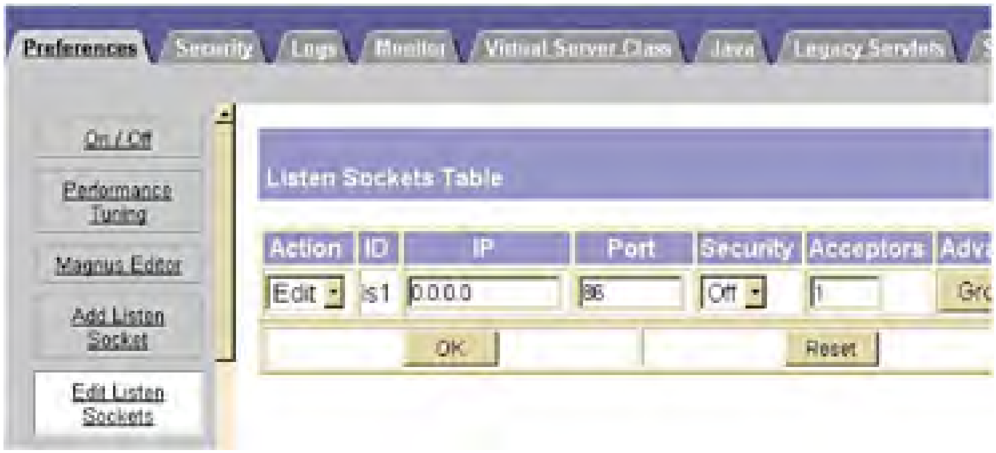


Created new server **water.red.iplanet.com (servlet)**, listening to port
D:/iPlanet/iws60/docs directory

Then click *Configure your new server*.

In Figure 5.10, the left-hand menu is used to maintain the web instance settings. Click *Edit Listen Sockets* to view the port that the web server instance listens to. In this case, the instance is listening on port *86*. The IP address of *0.0.0.0* means that the listener is listening on all the IP addresses of the computer. If the web server computer has more than one IP address, this web server instance answers to all browser requests on port 86 for all the IP addresses of the computer.

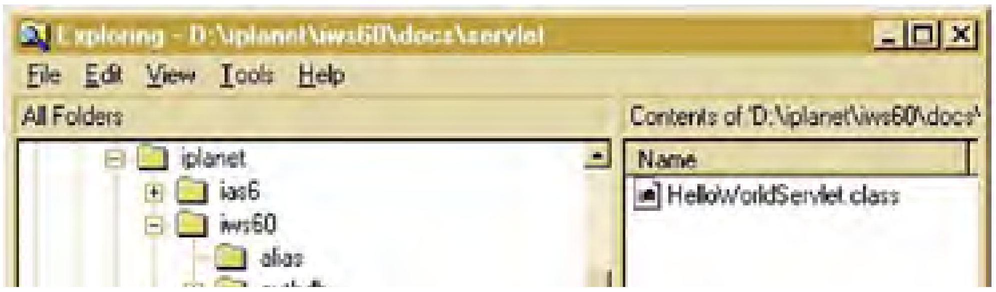
Figure 5.10. Listener settings.

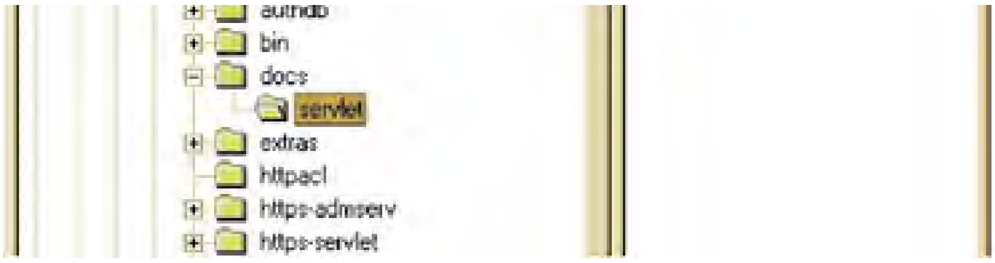


Action	ID	IP	Port	Security	Acceptors	Adv
Edit	ls1	0.0.0.0	86	Off	1	Gro

Figure 5.11 shows the directory listing of the installation: *iplanet/ iws60* is the main installation directory; *https-servlet* is the directory for the web instance that we are administering; *https-admserv* is the administration server directory.

Figure 5.11. https-servlet/config is where the configuration files are located.

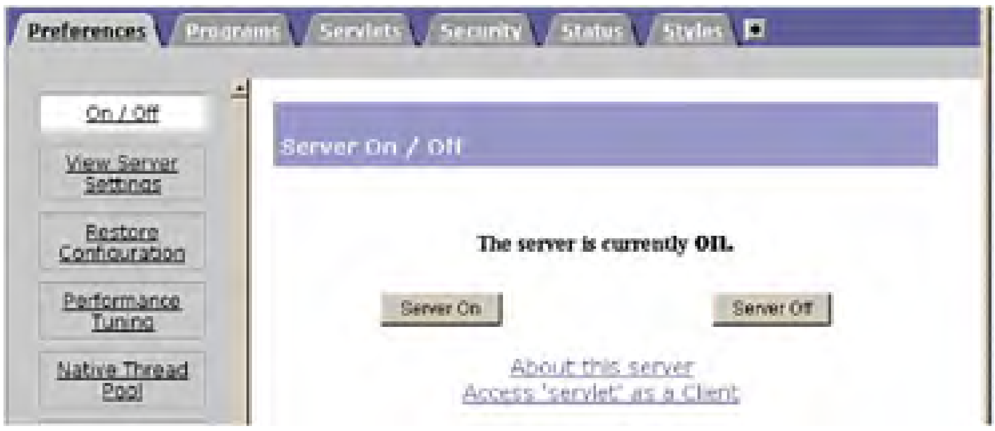




Web Browser Test

Now that the web instance has been added, it is time to start it and test it. Since it is a web server instance, the basic test is to turn the server on and surf to it with a browser. To turn the web server on from the administration server, click the *Preferences* tab; then click the *On/Off* menu option; then click the *Server On* button (Figure 5.12).

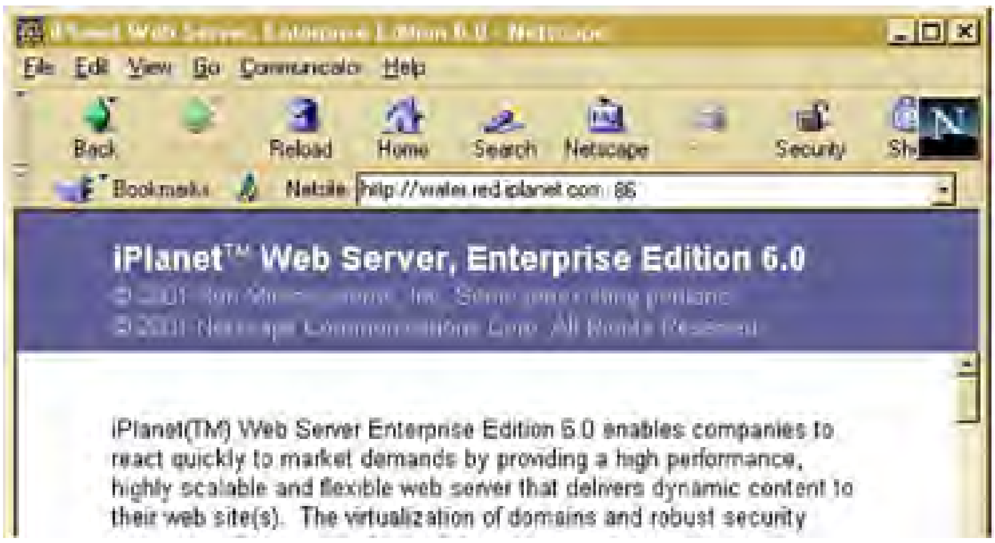
Figure 5.12. Starting and stopping web server instances from the administration server.



The server is currently Off.

Click *Access 'servlet' as a Client* to start another browser window to display the default home web page for this web instance (see Figure 5.13).

Figure 5.13. The default home web page that is created when a web server is installed.


File Edit View Go Communicator Help
Back
Reload
Home
Search
Netscape
Security
Bookmarks
Netsite: http://waiter.red.iplanet.com:86

iPlanet™ Web Server, Enterprise Edition 6.0

iPlanet(TM) Web Server Enterprise Edition 6.0 enables companies to react quickly to market demands by providing a high performance, highly scalable and flexible web server that delivers dynamic content to their web site(s). The virtualization of domains and robust security

Summary

Each of the four functions of the Sun ONE Web Server Administration Server mentioned in Figure 5.1 have been touched upon in this chapter.

The Sun ONE Web Server Administration Server was started from a command on UNIX or from the *Services* window on Windows. After starting the web server, you logged into it using the URL *http://water.red.iplanet.com:4006*.

From the administration server web site, administration web server configurations such as the listener settings and the administrator password are maintained. In a production system, the password is important. In development, I use user ID *admin* and password *admin*.

Like most computer consultants I know, the first thing I do after installing a Sun ONE Web Server is this: I remove the default web server web instance and create a new one. From the administration web server, this is a matter of removing the default web server instance from an administration web page and adding a web server instance from another administration web page.

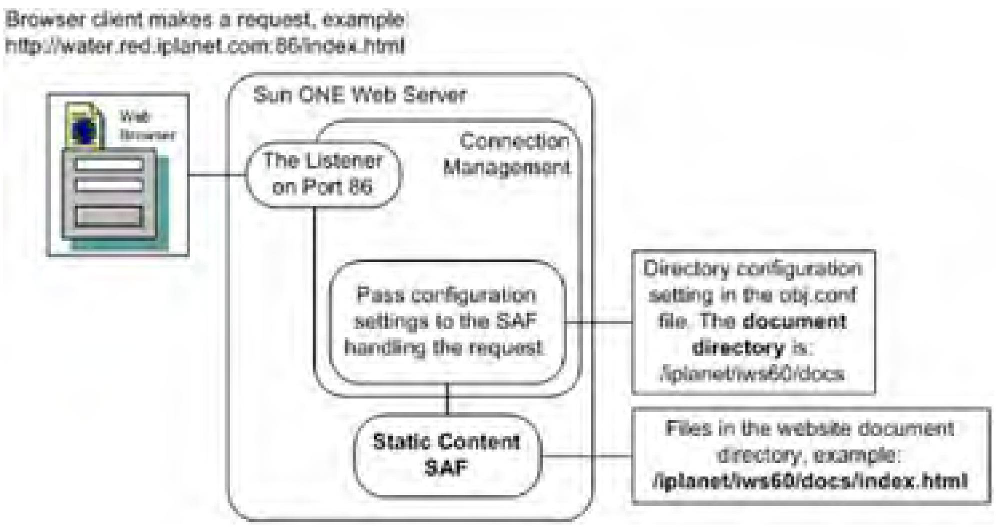
Web site content web server instances are maintained from the administration web server. A web server instance is selected from the main administration web page. This topic of maintaining web site content instances is covered in the next chapter.

Chapter 6. Configuring the Sun ONE Web Server for Static Content

The web server has been installed and tested. Now it is time to use it. The most fundamental thing a web server does is serve files. The serving of files is handled by the static content server application function (SAF). The SAF reads files and sends the files to the requester. The files are selected from the document directory on the web server computer.

In Figure 6.1, the browser is requesting the file index.html. The web server receives this request, and the static content SAF reads the index.html file and returns it to the browser.

Figure 6.1. Static content SAF document directory configuration setting.


Browser client makes a request, example:
http://water.red.iplanet.com:86/index.html
Web Browser
Sun ONE Web Server
The Listener on Port 86
Connection Management
Pass configuration settings to the SAF handling the request
Directory configuration setting in the obj.conf file. The document directory is: /iplanet/iws60/docs
Static Content SAF
Files in the website document directory, example: /iplanet/iws60/docs/index.html

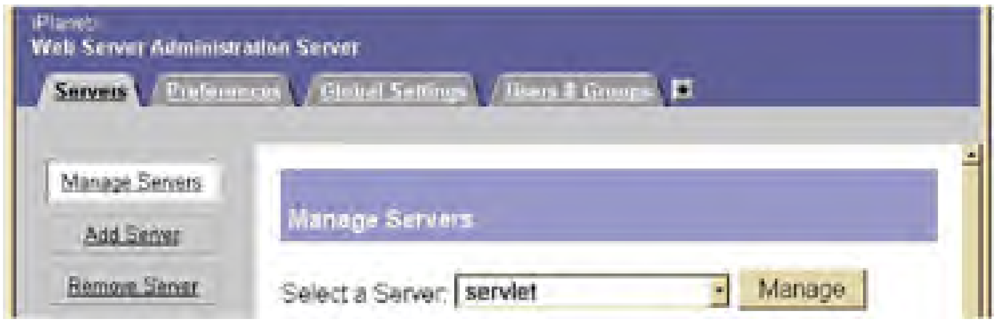
The Static Content SAF

The static content SAF is the basic web site tool for serving files to web browsers.

Exercise to View the Document Directory Configuration

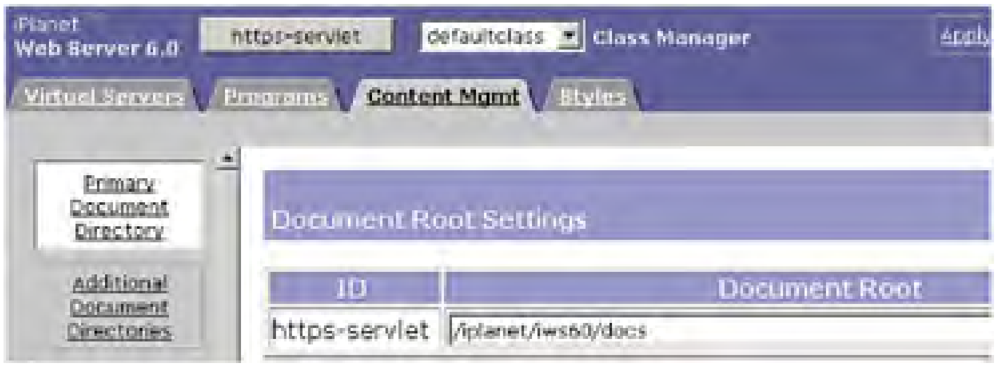
Before starting, ensure that the Sun ONE Web Server Administration Server is running. Log into the administration server. This brings you to the main/top web administration web page (shown in Figure 6.2). Web server instances are selected for management from this page.

Figure 6.2. Main web administration web page.



On the web page shown in Figure 6.2, click the *Manage* button to manage the *servlet* web server instance. On the top of the server management web page, click *Class Manager*. On the *Class Management* web page, click the *Content Mgmt* tab, and your browser should look like Figure 6.3. This shows that the document directory is file directory */iplanet/iws60/docs*.

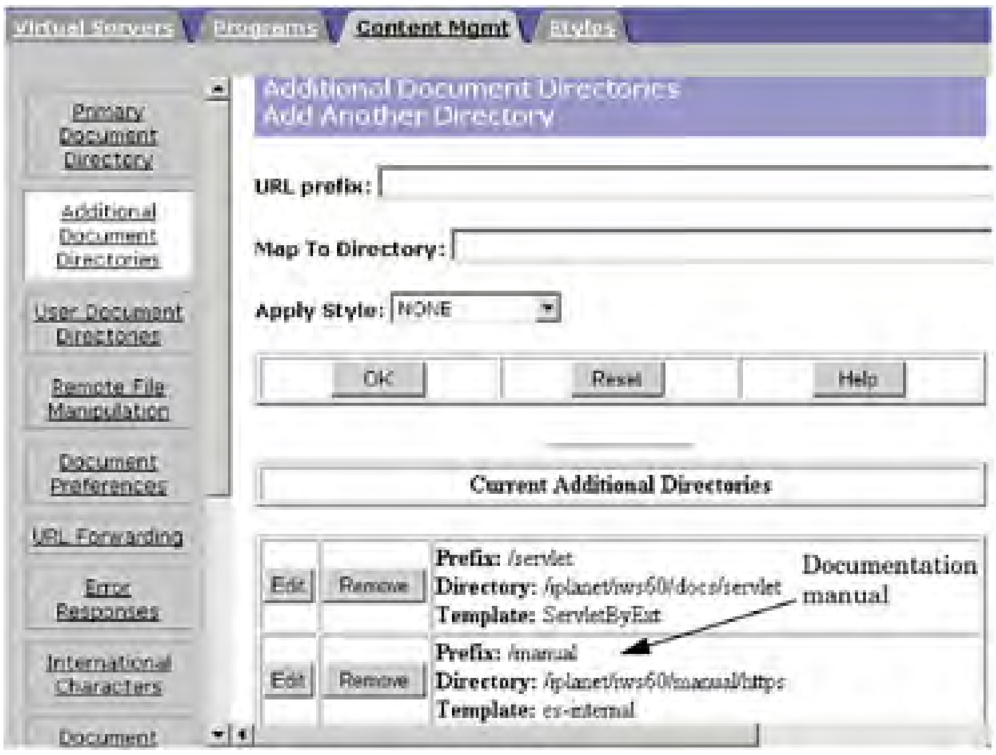
Figure 6.3. Class management web page to view or modify the document directory.



ID	Document Root
https-servlet	/iplanet/iws60/docs

On the web page pictured in Figure 6.3, click *Additional Document Directories*. This brings up the web page in Figure 6.4, which is used to configure more directory URL mappings. For example, there is a mapping setup from the offset directory /manual to the file directory /iplanet/iws60/manual/https. This maps the URL *http://water.iplanet.red.com:86/manual* to the file directory /iplanet/iws60/manual/https.

Figure 6.4. Other document directory mappings.



URL prefix:

Map To Directory:

Apply Style: NONE

Current Additional Directories

Prefix: /servlet
Directory: /iplanet/iws60/docs/servlet
Template: ServletByExt

Prefix: /manual
Directory: /iplanet/iws60/manual/https
Template: es-internal

Ensure that your web server instance is running and go to the URL *http://<hostname>:86/manual/index.htm*.

/manual/index.htm is a web page for Sun ONE Web Server documentation.

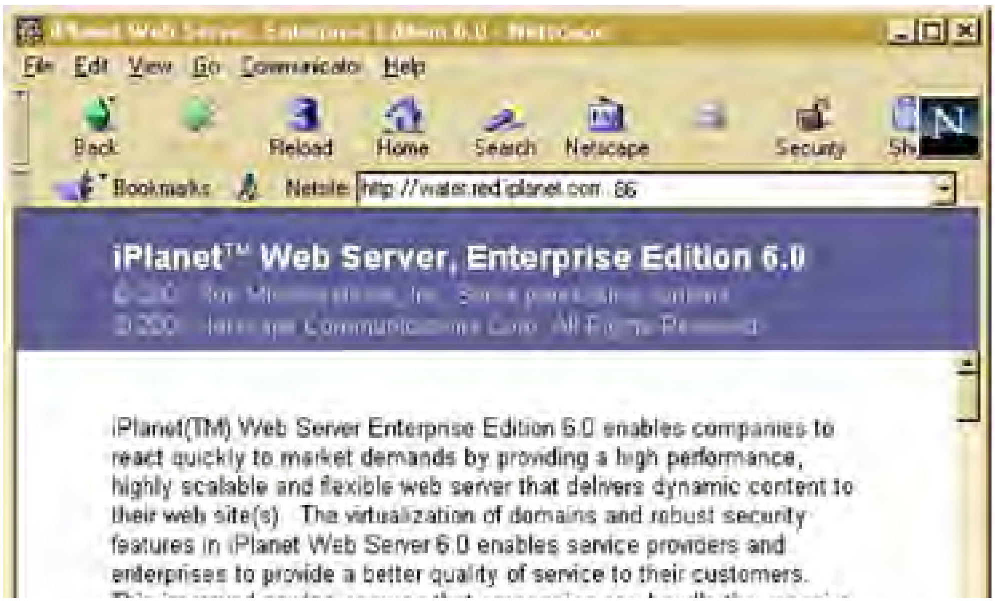
Now that we know where the document directory /iplanet/iws60/docs is, the next exercise is to modify the default home web page in the document directory.

Exercise to Modify the Static Web Page Called the Home Page

Before starting, ensure that the Sun ONE Web Server is running. From a web browser bring up the home page; for example, URL *http://water.read.iplanet.com:86*.

When a web server is installed, a default home web page is installed. Figure 6.5 is a screen print of the home web page.

Figure 6.5. Default Sun ONE Web Server home web page.



http://water.red.iplanet.com:86

iPlanet™ Web Server, Enterprise Edition 6.0

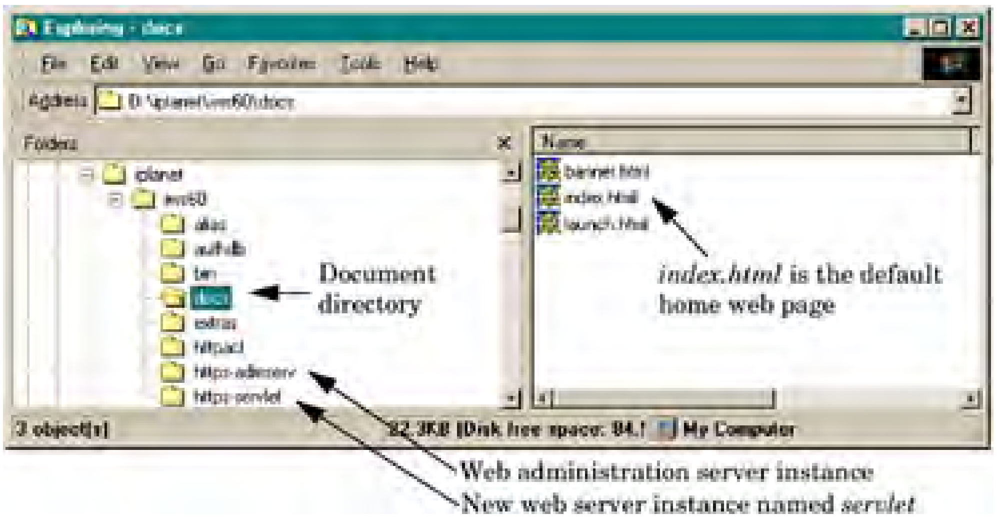
iPlanet(TM) Web Server Enterprise Edition 6.0 enables companies to react quickly to market demands by providing a high performance, highly scalable and flexible web server that delivers dynamic content to their web site(s). The virtualization of domains and robust security features in iPlanet Web Server 6.0 enables service providers and enterprises to provide a better quality of service to their customers.

This improved service ensures that companies can handle the massive volume demanded by successful web sites in the Net Economy and improve the management of their operations by lowering the costs of their infrastructure

The web server maps URLs to directories. In our sample installation, *http://water.read.iplanet.com:86* is mapped to the file directory iplanet/iws60/docs.

iplanet/iws60/docs is called the document directory. Figure 6.6 is the directory listing for the web server's document directory. The directory for the web instance that is being used here is *https-servlet*. The configuration files are located at *https-servlet/config.* The top web site directory, the document directory, is *docs*.

Figure 6.6. Document directory listing.



Exploring - docs
File Edit View Go Favorites Tools Help
Address D:\iplanet\iws60\docs
Folders
iplanet
iws60
alias
authdb
bin
docs
extras
httpacl
https-admserv
https-servlet
Name
banner.html
index.html
launch.html
Document directory
index.html is the default home web page
3 object(s)
My Computer
Web administration server instance
New web server instance named *servlet*

Use your favorite text file editor and load the file /iplanet/iws60/index.html for editing.

Here is a listing of my index.html:

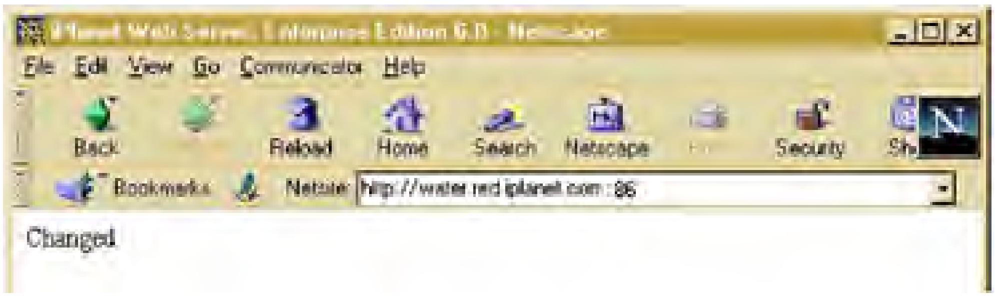
```
<HTML>
<HEAD>
<TITLE>Sun ONE Web Server, Enterprise Edition 6.0</
 TITLE>
</HEAD>

<FRAMESET BORDER=0 ROWS="80, *">

<FRAME NAME="banner" SRC="banner.html" SCROLLING="no">
<FRAME NAME="content" SRC="launch.html" SCROLL-
 ING="auto">

</FRAMESET><noframes></noframes>

</HTML>
```

Change the index.html file to look like this:

```
<HTML>
<HEAD>
<TITLE>Sun ONE Web Server, Enterprise Edition 6.0
 </TITLE>
</HEAD>

Changed.

</HTML>
```

After the changes are saved, reload the browser. The web changes show up as pictured in Figure 6.7. Note: When using a Netscape browser, to ensure that the page is reloaded, hold the Shift key down and hit the Reload button. This causes a forced reload; the page is reloaded from the web server, not from the cache.

Figure 6.7. Changed home web page.


iPlanet Web Server, Enterprise Edition 6.0 - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Security
Bookmarks Netsite: http://water.red.iplanet.com:86

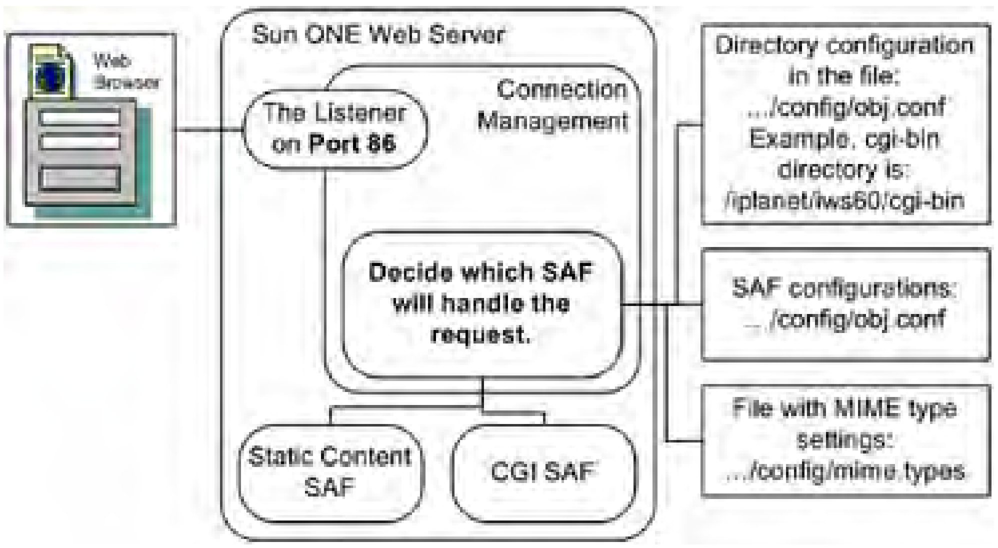
Changed

Now that you know where the static web pages are located, create your own web site. Easy.

The SAF Decision Process

Once a request comes in to the listener, the request goes through the connection management process. The last part of the process is to get configuration information from the configuration files and decide which SAF will handle the request. Once the decision is made, the request and the related SAF configuration information is passed to the SAF. This is illustrated in Figure 6.8.

Figure 6.8. Web server request process.


Web Browser
Sun ONE Web Server
The Listener on Port 86
Connection Management
Decide which SAF will handle the request.
Static Content SAF
CGI SAF
Directory configuration in the file: .../config/obj.conf Example, cgi-bin directory is: /iplanet/iws60/cgi-bin
SAF configurations: .../config/obj.conf
File with MIME type settings: .../config/mime.types

There are two major factors to consider when determining how a request is to be handled by the web server. One is the offset directory and the other is the MIME type or the extension of the filename.

First let's look at an example of a URL where the decision is based on the offset directory:

http://internetflow.com/cgi-bin/prog.sh

The offset directory /cgi-bin/ is in the obj.conf configuration file. The configuration setting tells us that all files in the directory /cgi-bin/ will be handled by the CGI SAF. In this instance, MIME types are not used to make the decision.

Now let's look at an example where the decision is based on the MIME type:

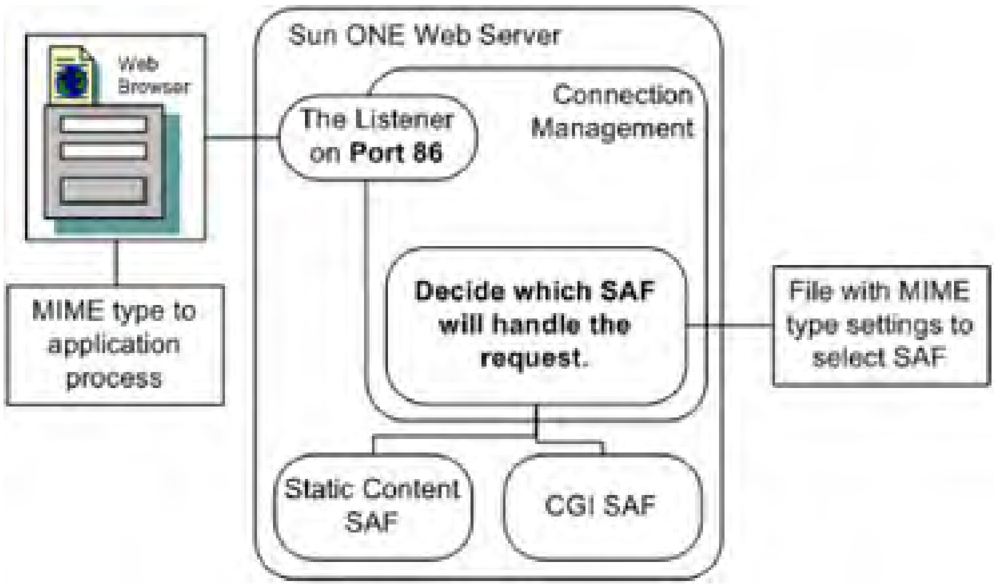
http://internetflow.com/files/prog.cgi

Files with the filename extension cgi are processed the same way CGI programs are processed in the /cgi-bin/ directory. With MIME types, it is the filename extension, not the directory, that decides how the web server processes the file. In our sample URL above, there are no settings regarding the /files/ directory in the obj.conf configuration file. However, there is a setting in the MIME type configuration file; this setting specifies that files with the extension cgi are handled by the CGI SAF. This request is then handed off to the CGI SAF.

MIME Types

The concept of handling files based on filename extensions has been extended to both the web browser and the web server areas. In the Sun ONE Web Server, MIME types are used to specify which SAF will handle a request and, in some instances, how the request is to be handled by the SAF. On a web server, a MIME type setting is used to control the request process flowing from one SAF to another (illustrated in Figure 6.9). On the web browser side, when a file is received by the browser, the browser makes a decision on how to process the file based on the MIME type sent from the web server. For example, if the browser receives a file with the MIME type image/gif, the browser displays the image file. Or, if the browser receives an audio/x-wav sound file, the browser starts an external program or a plug-in-to-play sound file.

Figure 6.9. MIME type settings and the SAF decision process.


Sun ONE Web Server
Web Browser
Connection Management
The Listener on Port 86
MIME type to application process
Decide which SAF will handle the request.
File with MIME type settings to select SAF
Static Content SAF
CGI SAF

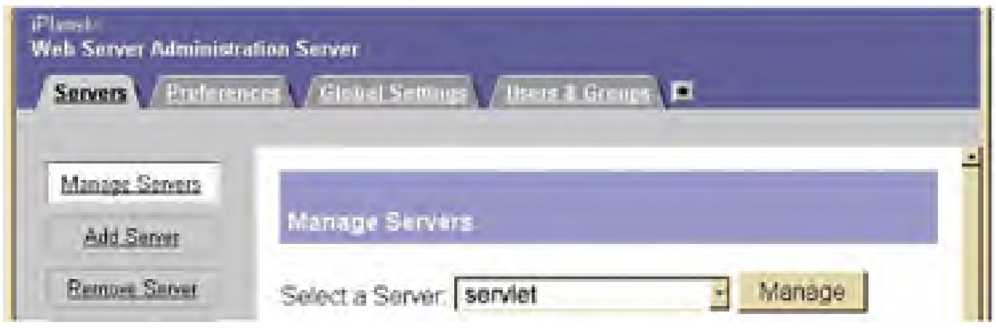
First let's look at the web server side.

Viewing and Modifying Sun ONE Web Server MIME Settings

This is an interactive section, so follow the screen prints that are in the figures or, better yet, follow along with your own Sun ONE web server installation. This section covers the steps to view and make changes to the MIME type settings.

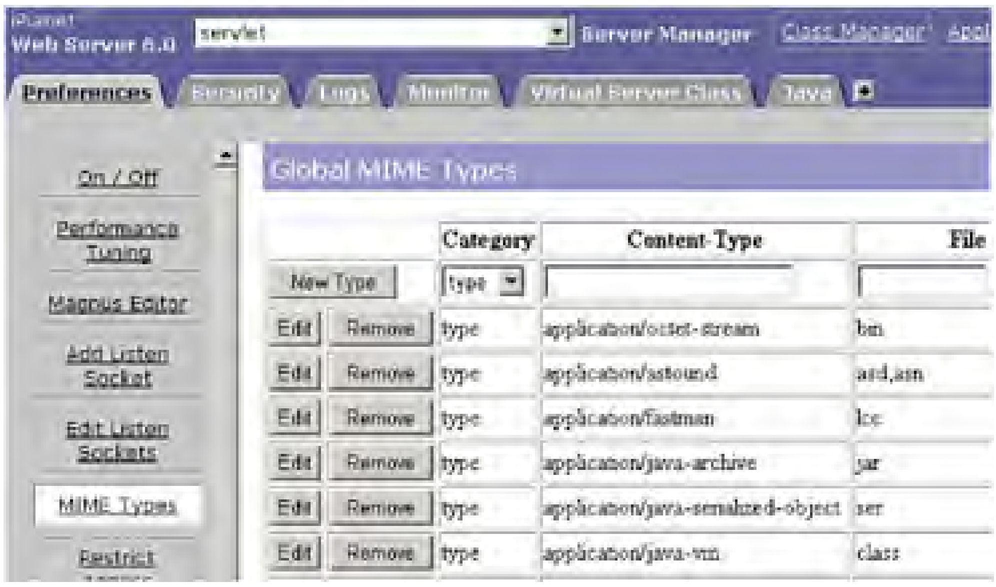
Before starting, ensure that the Sun ONE Web Server Administration Server is running. Log in to the administration server. This brings you to the main/top web administration web page (Figure 6.10). From here, web server instances are selected for management.

Figure 6.10. Main administration web page.



On the web page in Figure 6.10, click the *Manage* button to manage the *servlet* web server instance. Note: On the web instance management page (Figure 6.10), the *Preferences* tab is highlighted. On the menu under the *Preferences* tab, click *MIME Types* to display the *Edit MIME File* web page. Click the OK button to view and edit the list of default MIME types (Figure 6.11).

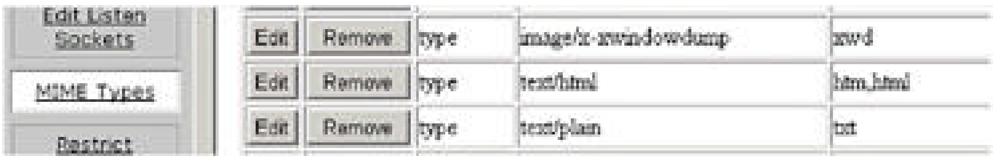
Figure 6.11. Web server MIME type listing.



	Category	Content-Type	File
New Type	type		
Edit Remove	type	application/octet-stream	bin
Edit Remove	type	application/astound	asd,asn
Edit Remove	type	application/fastman	lcc
Edit Remove	type	application/java-archive	jar
Edit Remove	type	application/java-serialized-object	ser
Edit Remove	type	application/java-vm	class

Scroll down until you see the filename extension *html,* which has the MIME type *text/html*, as in Figure 6.12. When a request is made for an html file, the web server's static content SAF reads the file; it then sends the MIME type value text/html and sends the html file to the requesting browser.

Figure 6.12. MIME type text/html.



Edit Remove	type	image/x-xwindowdump	xwd
Edit Remove	type	text/html	htm,html
Edit Remove	type	text/plain	txt

For example, requesting the URL *http://water.red.iplanet.com/index.html* causes the web server to read the file index.html, send the MIME type value text/html, and send or stream the index.html file to the browser. The browser displays the html file.

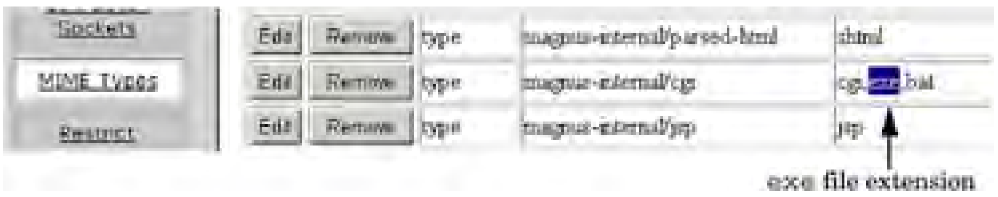
Another way to find a MIME type in the list is to use the browser Find function. From the browser, use Ctrl+F to display the Find window. Enter the MIME type and click *Find Next*.

Modifying How the Web Server Handles Files with the Extension EXE

This section has the instructions for configuring the Sun ONE Web Server to select which SAF will process files that have the extension exe. From the web server's point of view, a MIME type is the filename extension. The web server is configured to use a specific SAF to process files based on the MIME type. Therefore, changing the MIME type settings can change the SAF that will process a file based on the filename extension.

Let's look at an example for the file named lfnl.exe. In the MIME type listing in Figure 6.13, find the file type exe. The filename extension exe relates the filename of lfnl.exe to the MIME type or *Context-type* of magnus-internal/cgi. This means that the CGI SAF processes files with the extension exe.

Figure 6.13. Sun ONE Web Server MIME type listing.



type	magnus-internal/parsed-html	shtml
type	magnus-internal/cgi	cgi,exe,bat
type	magnus-internal/jsp	jsp

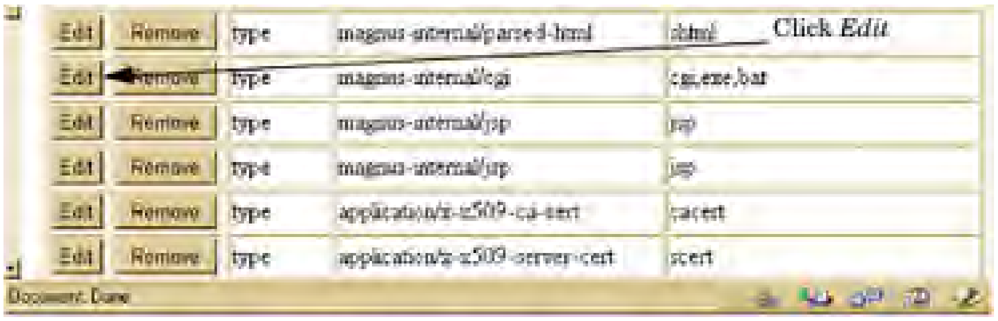
For an exercise, rather than having the CGI SAF process the exe files, let's configure the web server to use the static content SAF. Whereas the CGI SAF will attempt to run the exe file, the static content SAF will send the exe file to the browser. This means changing the MIME type setting that controls which SAF handles the browser request, based on the filename extension of exe.

Before making the changes, test how the web server currently handles a request for an exe file. Put a small exe program into your docs directory by copying an exe file into the web server document directory /iplanet/iws60/docs. Name the file lfnl.exe. Now make a request for the file using a browser and sample URL *http://water.red.iplanet.com:86/lfnl.exe*.

Your browser may give you an error message, or (like my browser) it may ask where to save the file. Then select a directory, and the browser sets up a directory to receive the file. However, this step fails because the web server does not send the file to the browser. The web server instance tries to run lfnl.exe as a CGI program because the web server is processing the file using the CGI SAF. This is an example of a case where the web browser MIME type is mismatched with the web server MIME settings—the browser expects one thing and the server does something different.

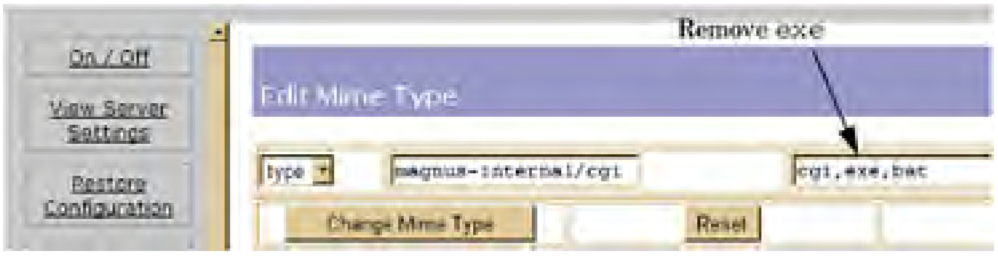
To fix this problem, from the *MIME Settings* web page, click the *Edit* button on the row/line for MIME type *exe* in Figure 6.14. This will bring up the Edit screen (Figure 6.15).

Figure 6.14. Edit MIME settings.



type	magnus-internal/parsed-html	shtml
type	magnus-internal/cgi	cgi,exe,bat
type	magnus-internal/jsp	jsp
type	magnus-internal/jsp	jsp
type	application/x-x509-ca-cert	cacert
type	application/x-x509-server-cert	scert

Figure 6.15. Edit screen.

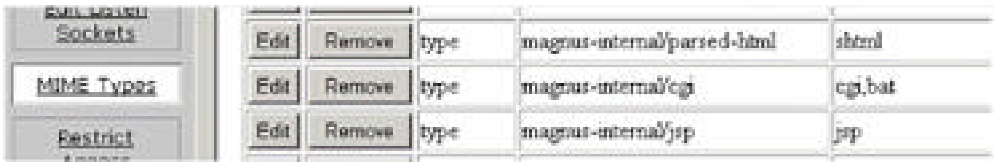


type	magnus-internal/cgi	cgi,exe,bat

Remove *exe* and click *Change Mime Type*.

The list is now refreshed. Find the file extension *cgi*. Note that exe has been removed (see Figure 6.16).

Figure 6.16. exe files are no longer processed by the CGI SAF.



type	magnus-internal/parsed-html	shtml
type	magnus-internal/cgi	cgi,bat
type	magnus-internal/jsp	jsp

To apply the changes to the web server, click Apply at the top of the web page (refer back to Figure 6.10 to see the top of the page with Apply). Then click the *Load Configuration Files* button to apply the changes.

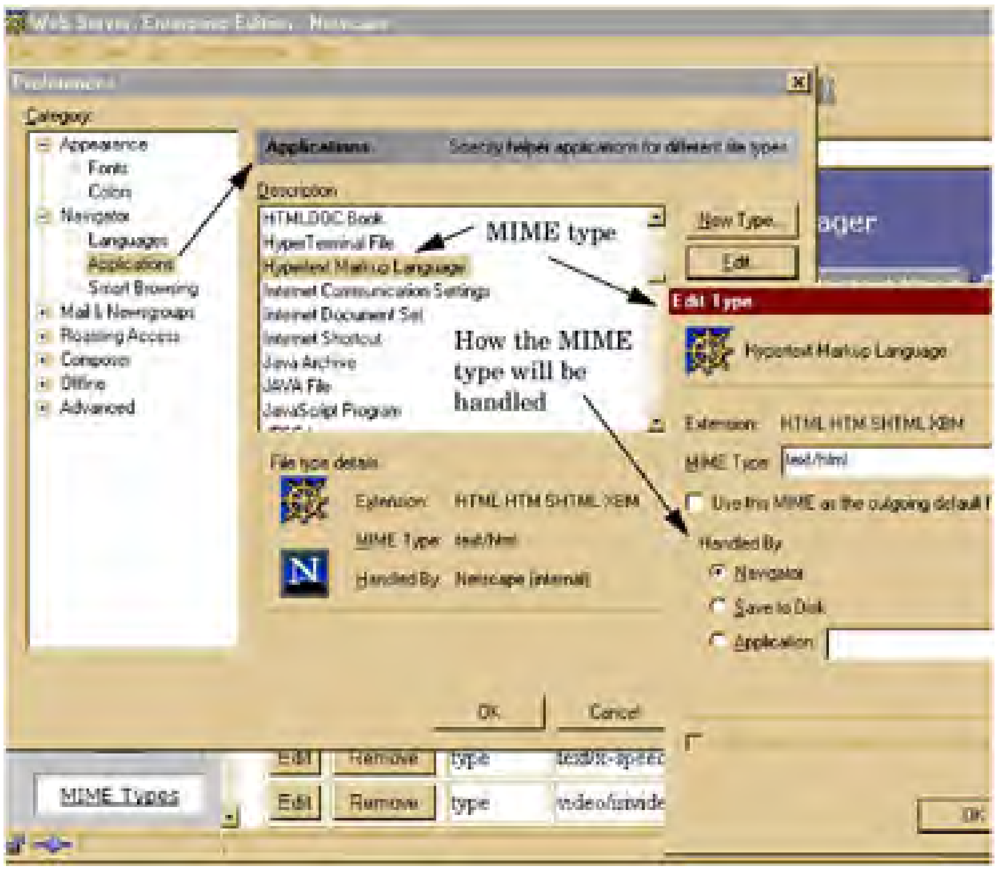
To test the results of the modification, try to download the file again using a browser and the example URL *http://water.red.iplanet.com:86/lfnl.exe*.

Now the web server uses the static content SAF to upload the file to the browser's computer. The web server does not try to run the file as a CGI program; the web server sends (or streams) the file to the browser. The web server is now configured to use the static content SAF for exe rather than the CGI SAF.

The following is what happens when a web browser makes a request for an exe file:

1. Browser calls the web server instance; for example: *http://water.red.iplanet.com:86/lfnl.exe*.
2. Web server checks the MIME settings list to see what is to be done with files that have a file extension of exe.
   - If the MIME type is magnus-internal/cgi, the web server runs the program using the CGI SAF. This is the default setting.
   - If there is no MIME setting, the web server uses the default static content SAF to return the file to the browser. Before returning the file, the web server first sends the MIME type value to the web browser. If the file extension is html, the web server will tell the browser the MIME type is text/html and then send the file. In this example, exe is not in the MIME type list, so the web server sends the default MIME type text/plain and then sends the file to the browser.
3. The browser receives the file MIME type heading and the file. The browser looks up the MIME type in its MIME type list (Figure 6.17) and then acts appropriately. For example, if the MIME type is text/html, the browser displays the file using *Navigator* (the browser itself). If the MIME type is not in the list, *Navigator*, by default, displays the file.

Figure 6.17. Netscape browser MIME types.



The Netscape Web Browser and MIME Types

To view or change how your Netscape browser handles files based on the MIME type, click *Edit* on the top menu option, then *Preferences* and then *Applications* (Figure 6.17).

Based on the *Description*, find the MIME type you want; for example *Hypertext Markup Language*. Click *Edit* and the options to handle the files that have MIME type *text/html* are displayed.

The Netscape Browser has three options to handle files received by the browser:

1. Navigator: The browser will handle the file and will get and display the web page (text/html) in the browser window.
2. Save to Disk: The browser will prompt the user for a file directory location to save the file and then will save the file to disk, as in our exe example.
3. Application: Save the file to a temporary file directory and then call the selected application to run the file.

Summary

The static content SAF serves files from the document directory. The location of the document directory is configured in the obj.conf file. This file contains URL offset directory mappings to physical file directories such as the document directory.

When a request is made for a file in the document directory, the web server checks the MIME type settings for the file extension. Some file extensions are mapped to other SAFs—SAFs other than the static content SAF. The configuration file mime.types contains the SAF filename extension mapping. These configurations are managed from the Sun ONE Web Server Administration Server web site. If the request is for a file that is handled by a static content SAF, the static content SAF reads the file and returns it to the requester. If the MIME type is mapped to the CGI SAF, the file is run as a CGI program.

On the web server, a MIME type describes a file type based on the filename extension. The MIME type settings on a web server are used to control the request process flowing from one SAF to another as illustrated in Figure 6.9.

When a browser receives a file from a web server, the browser will process the file based on the MIME type as described in the heading information that the web server sends to the browser.

One last MIME type point: The browser and web server MIME types need to match. If they do not match, the web server may misinterpret a browser request and the request will fail. However, with a little testing and a little configuration setup, requests can be tuned to work perfectly.

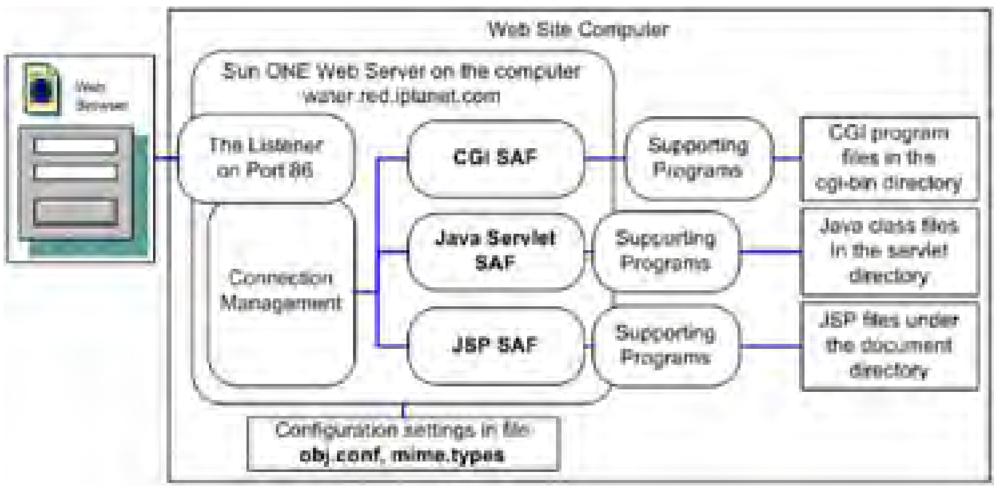
Chapter 7. Configuring the Sun ONE Web Server for Dynamic Content

Dynamic content results from the Sun ONE Web Server running programs to make web pages for a web site. This chapter discusses configuring the web server instance to run CGI programs, Java servlets, and JSPs. Once configured, programs can then be added into the web server environment. Next, the programs are tested—this is a matter of putting the correct URL into a web browser to call the web server. The web server runs the program, which creates a web page; the web page then goes back to the browser to be displayed.

The first step is to configure the Sun ONE Web Server to run programs. Configuration settings are managed through the Sun ONE Web Administration Server. The configurations are twofold: one type is used to decide which SAF is to be selected to process the request, and the other type involves the configuration settings used by the web server to run the supporting programs. The supporting programs are programs such as JVM that run servlet programs.

Referring to Figure 7.1, once a request comes in to the listener, the request goes through the connection management process. This process gets the configuration settings from the configuration files and decides which SAF is to be handed the request. There are two configuration options to decide which SAF will run the program: One setting is to use the offset directory in the URL, and the other setting is to use the MIME type (filename extension). Once the decision is made, the request and the related SAF configuration settings are passed to the SAF. The SAF runs the program, and the program returns dynamic content.

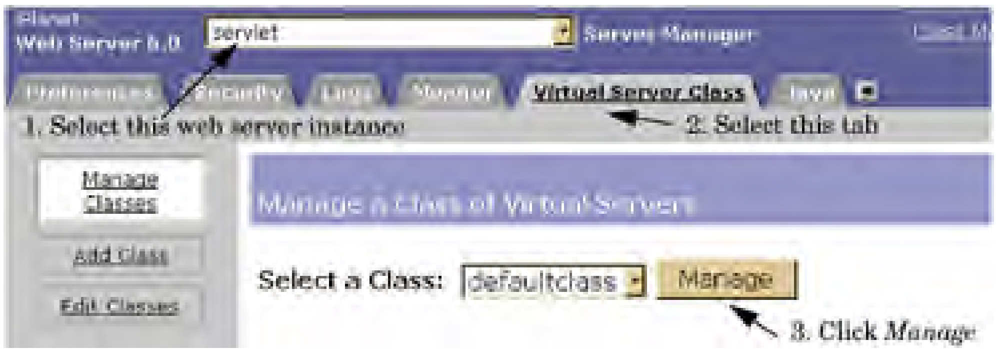
Figure 7.1. Dynamic content is as easy as 1, 2, 3 (1: configure; 2: program; 3: test).


Web Site Computer
Web Browser
Sun ONE Web Server on the computer
waner.red.iplanet.com
The Listener on Port 86
Connection Management
CGI SAF
Java Servlet SAF
JSP SAF
Supporting Programs
Supporting Programs
Supporting Programs
CGI program files in the cgi-bin directory
Java class files in the servlet directory
JSP files under the document directory
Configuration settings in file: obj.conf, mime.types

Setting Up a CGI-BIN Directory

The first step is to configure the web server. From the main Sun ONE Web Server Administration Server admin page, select to manage the web server instance named *servlet*; then click the tab *Virtual Server Class*, as shown in Figure 7.2.

Figure 7.2. Configuring the web server instance.



From the *Virtual Server Class* web page, click the *Programs* tab to bring up the screen for setting up a new CGI directory (Figure 7.3a). In the box next to *URL prefix*, enter */cgi-bin.* Next to CGI directory, enter a physical directory name of */iplanet/iws60/cgi-bin*. Then click *OK*. In Figure 7.3b, you will see that the cgi-bin directory has been added. To return to the server management web page, click the *https-servlet* button. Now click the *Server on* button to restart the web server instance. The web server is now ready to run CGI programs.

Figure 7.3a. Configure cgi-bin directory settings.

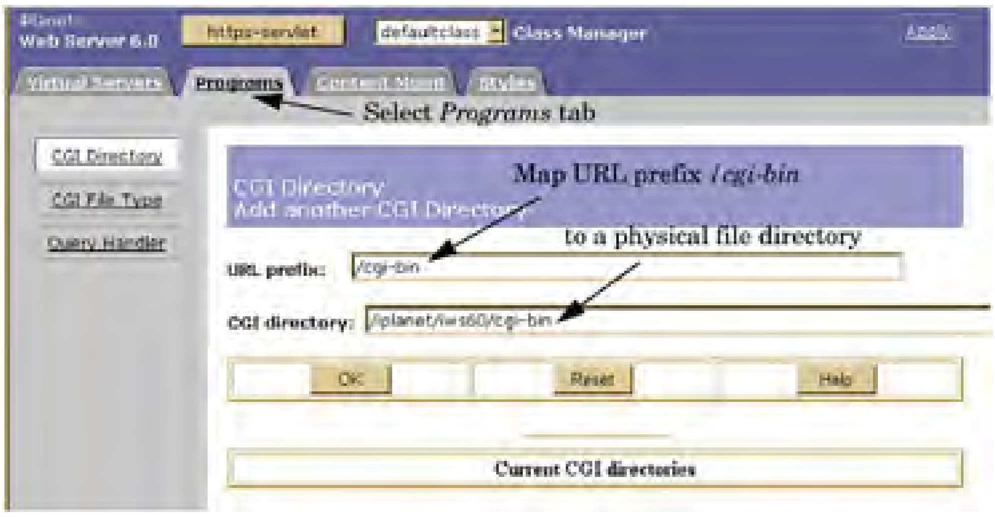
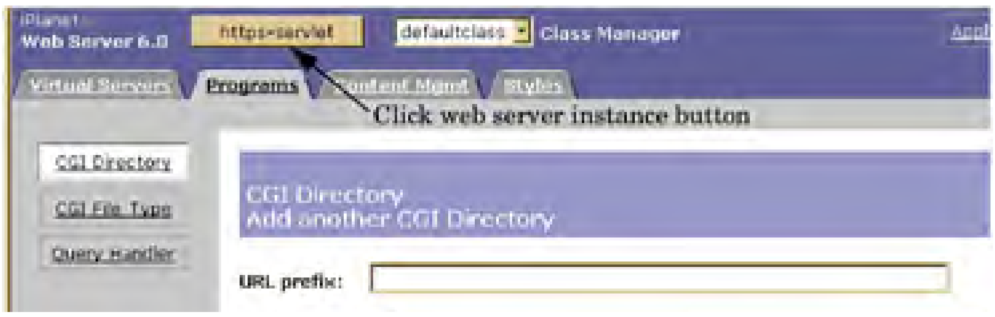
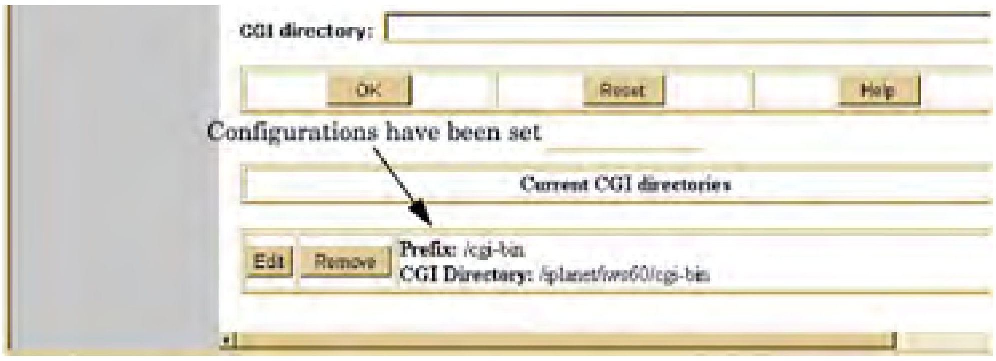


Figure 7.3b. Configuring cgi-bin directory settings.




CGI directory:

Configurations have been set

Current CGI directories

Prefix: /cgi-bin
CGI Directory: /iplanet/iws60/cgi-bin

On your web server computer, make a directory that matches the *CGI Directory*; our example directory is /iplanet/iws60/cgi-bin.

On UNIX, the command is `mkdir/iplanet/iws60/cgi-bin`.

Program Testing

UNIX CGI Program Test

To run a UNIX test, create a shell program named prog1.sh in the cgi-bin directory. Here is a sample UNIX shell program. Note: In the first line, /usr/bin/csh needs to match the location of csh on your computer.

```
#! /usr/bin/csh
echo "Content-type: text/html"
echo ""

echo "<html>"

echo "<HEAD>"
echo "<TITLE>Super Simple</TITLE>"
echo "</head>"
echo "<body>"
echo "<H1>Super Simple Test C-Shell Program</H1>"
echo "</body>"
echo "</html>"
exit
# eof
```

On UNIX, you must make the program file executable:

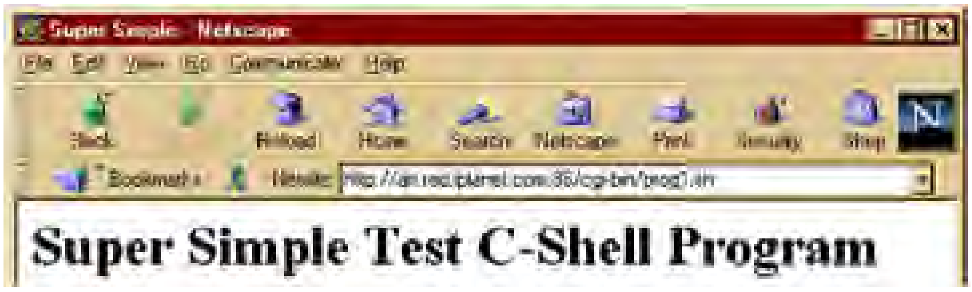
```
root[sh]@art# cd /iplanet/iws60/cgi-bin
root[sh]@art# chmod o+x prog1.sh
```

Now use your browser to test the program. The syntax to call the program is:

http://art.red.iplanet.com:86/cgi-bin/prog1.sh

Figure 7.4 shows that my test is successful.

Figure 7.4. Dynamic web page from the CGI program prog1.sh.



http://art.red.iplanet.com:86/cgi-bin/prog1.sh

Super Simple Test C-Shell Program

Windows CGI Program Test

To run a Windows test, create a batch program named prog1.sh in the cgi-bin directory. Here is a sample Windows batch program.

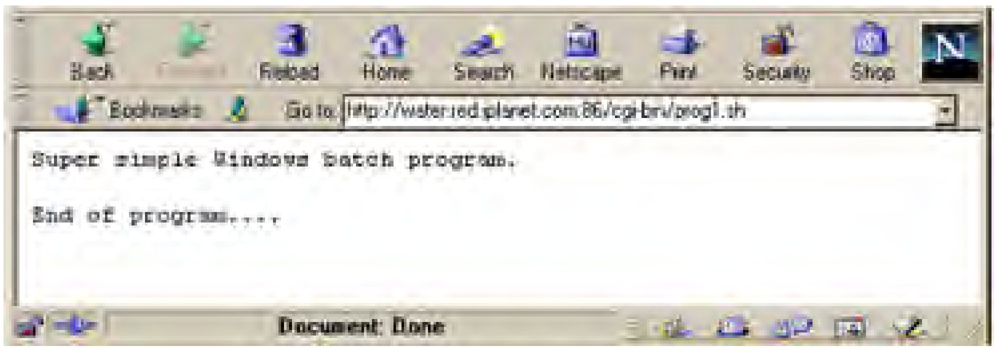
```
@echo off
echo Content-type: text/plain
echo.
echo Super simple Windows batch program.
echo.
echo End of program....
```

Now use your browser to test the program. The syntax to call the program is

http://water.red.iplanet.com:86/cgi-bin/prog1.sh

Figure 7.5 shows my test is successful.

Figure 7.5. Dynamic web page from the CGI program prog1.sh.



```
Super simple Windows batch program.

End of program....
```

CGI Configuration Summary

Configuring the Sun ONE Web Server to run CGI programs is very straightforward:

1. Configure the cgi-bin directory setting using the web administration system.
2. Set up the cgi-bin directory on the web server computer.
3. Write a program in the cgi-bin directory or write the program somewhere else and load or copy the program into the cgi-bin directory.
4. On UNIX, make the program file executable.
5. Test the program by using a web browser.

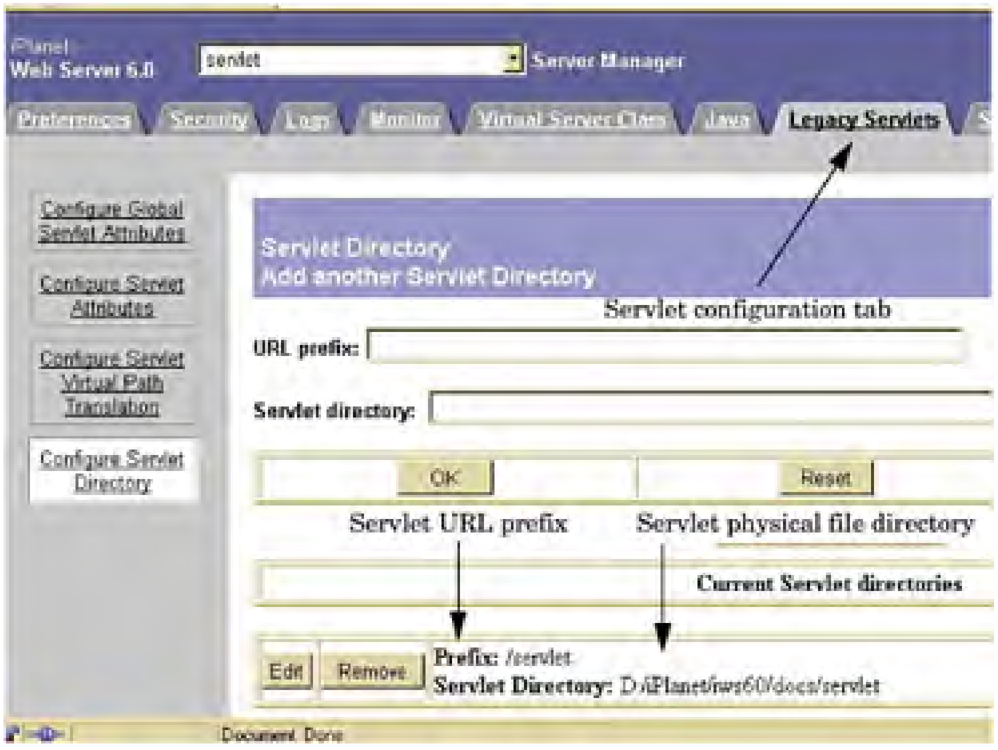
QED, which is likely why CGI is so popular.

Running the Hello World Servlet

By default, the Sun ONE Web Server is configured to run Java servlets, so let's start by viewing the configuration setting. This setting is the file directory location in which this web server instance expects to find the Java servlet class files. Any file in this directory is assumed to be a servlet and therefore is processed by the Java servlet SAF.

From the main Sun ONE Web Server Administration Server admin page, select to manage the web server instance named *servlet*. The web server instance main web page appears. On this web page, click on the *Legacy Servlets* tab, and then click on the left menu item *Configure Servlet Directory* as shown in Figure 7.6.

Figure 7.6. Java servlet directory setting.


iPlanet Web Server 6.0
servlet
Server Manager
Preferences
Security
Logs
Monitor
Virtual Server Class
Java
Legacy Servlets
Configure Global Servlet Attributes
Configure Servlet Attributes
Configure Servlet Virtual Path Translation
Configure Servlet Directory
Servlet configuration tab
Servlet URL prefix
Servlet physical file directory
OK
Reset
Edit
Remove
Document: Done

Servlet Directory
Add another Servlet Directory

URL prefix:

Servlet directory:

Current Servlet directories

Prefix: /servlet
Servlet Directory: D:/iPlanet/iws60/docs/servlet

In Figure 7.6, the servlet file directory is

- For Windows NT or Windows 2000: D:/iplanet/iws60/docs/servlet
- For UNIX: /iplanet/iws60/docs/servlet

You will need to make these directories.

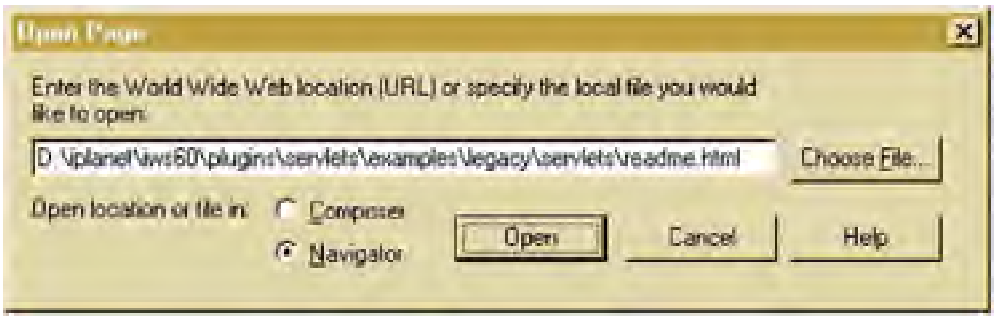
In Windows, either use the File Manager to create the directory or use the command mkdir d:\iplanet\iws60\docs\servlet.

The UNIX command to set up the servlet directory is mkdir /iplanet/iws60/docs/servlet.

To have the web server run the servlets, use the web browser to make a URL request using the offset directory (Prefix) /servlet. For example, if the Java class filename is HelloWorldServlet.class, then the URL to run the servlet program is *http://water.red.iplanet.com:86/servlet/HelloWorldServlet*.

A number of sample Java servlet programs come with the Sun ONE Web Server. The samples are found under the plugins directory, which is under the web server installation directory. To read information about the sample servlets, use a web browser. From the browser, select *File/Open Page*; then choose the servlet *readme* web page (Figure 7.7).

Figure 7.7. How to access the servlet readme files.


Open Page
Enter the World Wide Web location (URL) or specify the local file you would like to open:
D:\iplanet\iws60\plugins\servlets\examples\legacy\servlets\readme.html
Choose File...
Open location or file in:
Composer
Navigator
Open
Cancel
Help

To install the sample Hello World Java servlet into the web server environment, copy the class file named HelloWorldServlet.class from the samples directory into the servlet program directory /iplanet/iws60/docs/servlet. (Also, copy the Java program into the same directory because it will be modified, recompiled, and run later in this chapter.)

Note: For UNIX, you must first make the servlet directory mkdir/iplanet/iws60/docs/servlet and then copy the servlet files /iplanet/iws60/plugins/samples/servlets/servlets/HelloWorld/*.class. For Windows, the file explorer works fine, as in Figures 7.8 and 7.9.

Figure 7.8. Hello World class file in the servlet directory.

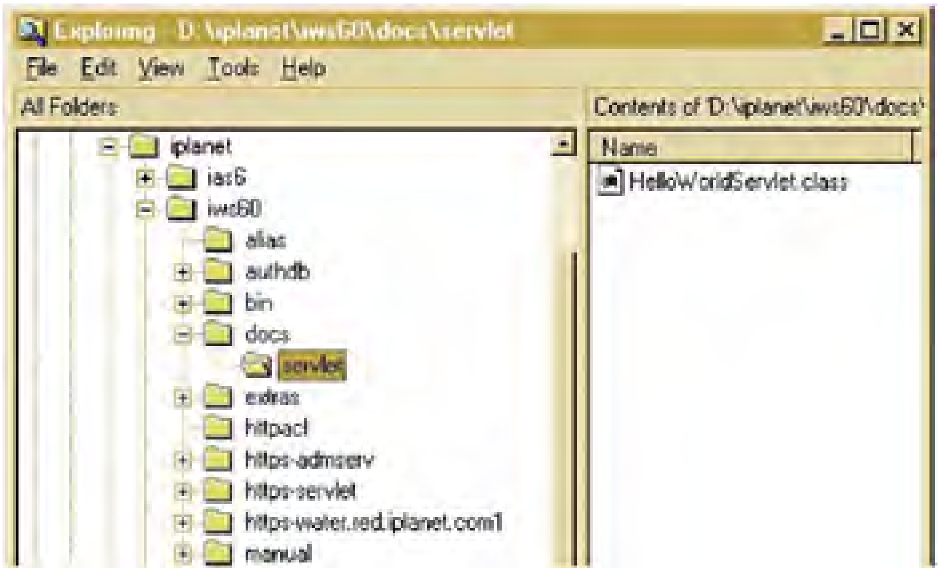
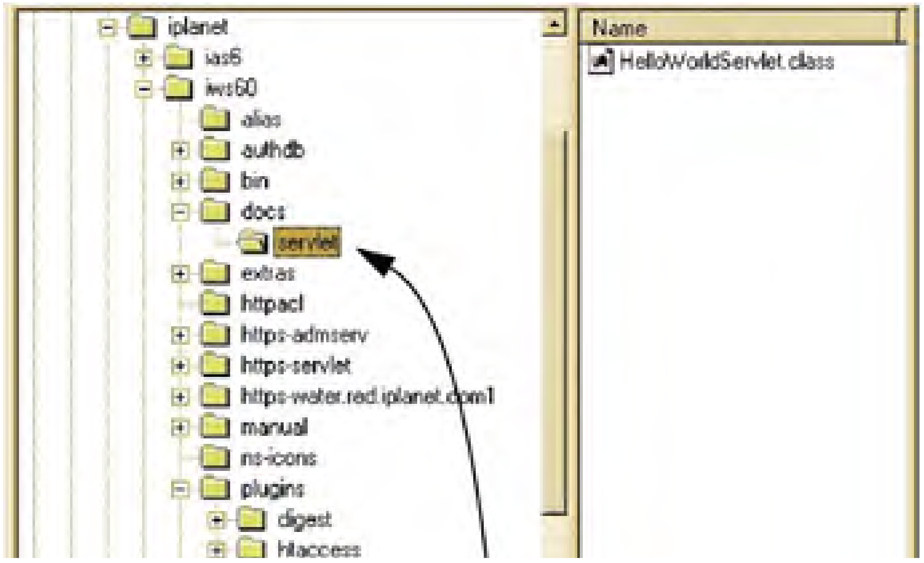
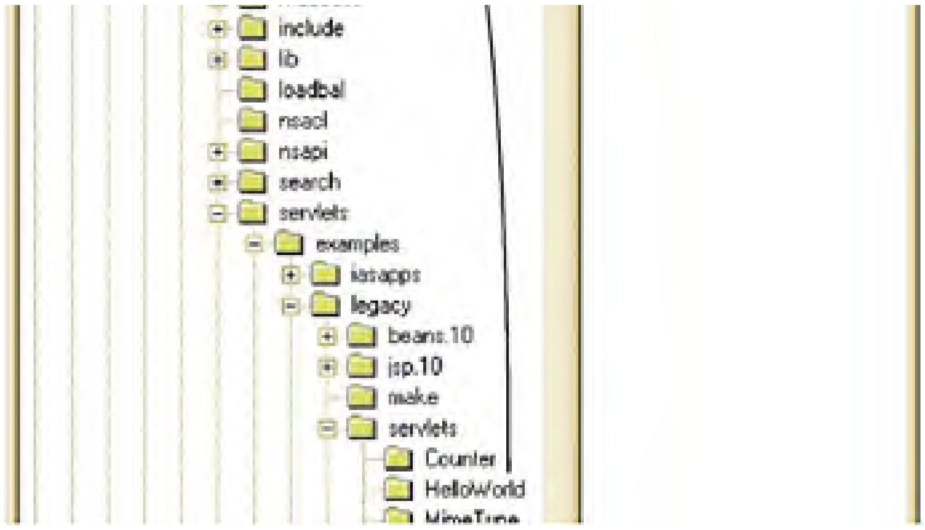

Exploring - D:\iplanet\iws60\docs\servlet
File Edit View Tools Help
All Folders
Contents of 'D:\iplanet\iws60\docs\'
iplanet
ias6
iws60
alias
authdb
bin
docs
servlet
extras
httpacl
https-admserv
https-servlet
https-water.red.iplanet.com1
manual
Name
HelloWorldServlet.class

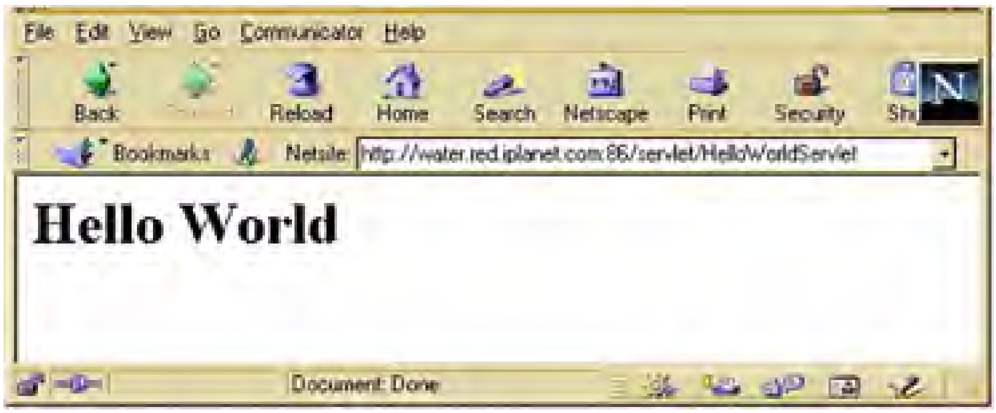
Figure 7.9. Copying the Hello World class file into the servlet directory.





Once a Java class file is put into the servlet directory, it can be called from a web browser. From a web browser, go to *http://water.red.iplanet.com:86/servlet/HelloWorldServlet* (Figure 7.10).

Figure 7.10. Output from the Hello World servlet program.


File Edit View Go Communicator Help
Back
Reload
Home
Search
Netscape
Print
Security
Bookmarks
Netsite: http://water.red.iplanet.com:86/servlet/HelloWorldServlet
Document: Done

Hello World

Making Changes to the Hello World Servlet

Try this:

1. To make testing easier, use the servlet directory as the work directory `cd /iplanet/iws60/docs/servlet`.
2. Copy the Java program to the work directory:

   ```
   cp /iplanet/iws60/plugins/samples/servlets/serv-
   lets/HelloWorld/*.java.
   ```
3. Make a copy of the program and use a new name: `cp HelloWorldServlet.java hello.java`
4. Modify `hello.java`. There are two ways to make the modification:
   - Change

     ```
     public class HelloWorldServlet extends HttpS-
     ervlet {
     ```

     to

     ```
     public class hello extends HttpServlet {
     ```
   - Change

     ```
     out.print("<h1>Hello World</h1>");
     ```

     to

     ```
     out.print("<h1>Hello there...</h1>");
     ```

5. Now compile the code. When compiling servlets, you need to include the Sun ONE Web Server jar file servlet.jar. The other file in the compile command is classes.zip, which is installed with the JDK.

   With UNIX, Java automatically installs under the /usr directory. The command

   ```
   javac   -g   -classpath   .:/usr/java1.1/lib/
   classes.zip:/iplanet/iws60/bin/https/jar/serv-
   let.jar hello.java
   ```

   should work.

   With Windows, if you have not already installed the JDK, go to *http://java.sun.com/downloads.html* and download the JDK, using the option *Java 2 Platform, Standard Edition (J2SE)*. Then install it.

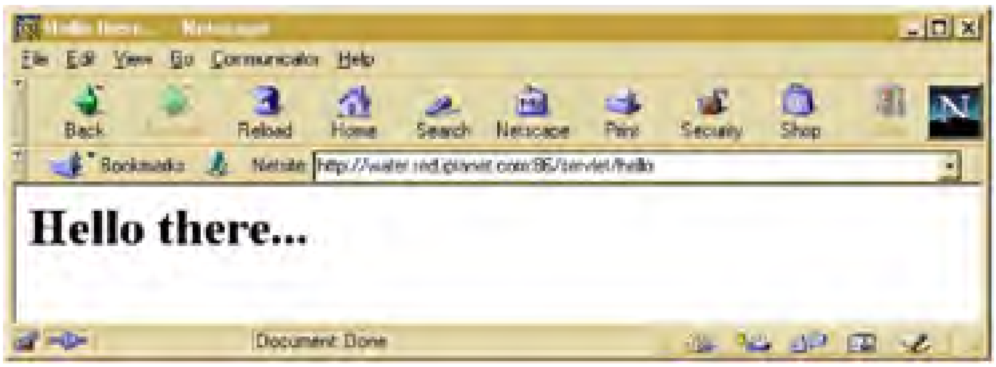
   Here is a sample compile command in Windows:

   ```
   javac -g -classpath .;c:\internet\java\jdk\lib
   \classes.zip;d:\iplanet\wsserver4\bin\https\jar\
   servlet.jar hello.java.
   ```

   When using Windows, I put the Java compile command into a bat file; for example, hello-comp.bat. Then, from the file explorer, I double-click on the bat file to run the compile command. This saves time.

6. To test, from a web browser, go to the URL *http://water.red.iplanet.com:86/servlet/HelloWorldServlet* (Figure 7.11).

Figure 7.11. Output from the Hello There servlet.

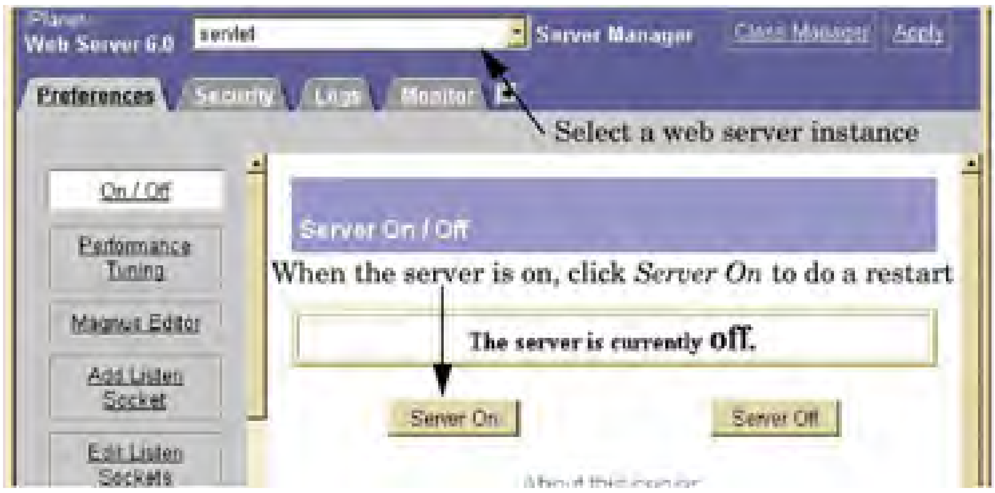

File Edit View Go Communicator Help
Back
Reload
Home
Search
Netscape
Print
Security
Shop
Bookmarks
Netsite: http://water.red.iplanet.com:86/servlet/hello
Document: Done

Hello there...

Note: After changing the servlet class that has already been run (loaded into memory), restart the web instance to remove the old servlet from memory. Once restarted, no servlets will be in the web instance's memory. Now call the servlet again. The new servlet will be loaded and run.

To restart the web instance, just click the *Server On* button as in Figure 7.12.

Figure 7.12. Web server instance main web page.



The server is currently off.

Running JSPs

The only thing you need to run servlets on a web server is a JRE. The Sun ONE Web Server comes with the JRE to run servlets. However, to run JSPs, the JDK is needed. Here are the steps the web server goes through to run a JSP.

1. The JSP is transformed into a Java servlet program.
2. The Java servlet is compiled by the compiler javac.exe into a class file. The javac.exe compiler comes with the JDK.
3. Once compiled, the class file is loaded into memory (cached) and is run by the web server using the JRE.

A JSP is basically an HTML file with embedded Java code. Here is an example:

```
<head>
<title>JSP Super Simple Sample</title>
</head>
<BODY  bgcolor="#FFFFFF" text="#000000">

<h2>JSP Super Simple Sample</h2>

<% String s1 = "cool"; %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

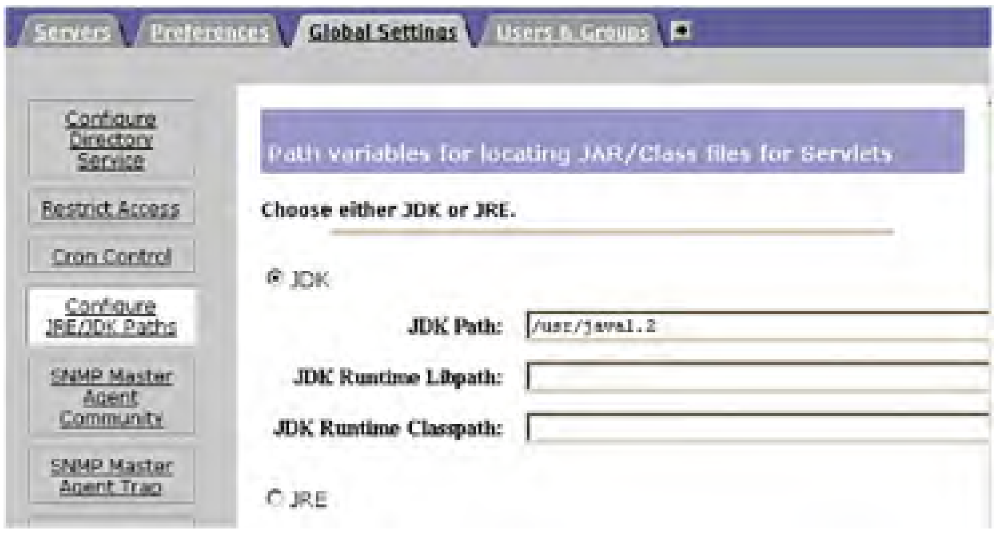
String s1 = "cool"; is the Java command to declare a variable of type *String* and to set the variable to the value *cool*. <% and %> are the tokens used to embed Java code.

<%=s1%> is the JSP command to display/print a Java program variable.

The rest of the code is HTML.

For a web server to run a JSP, the JSP first needs to be compiled. From the Sun ONE Web Server Administration Server main page, select tab *Global Settings* and click *Configure JRE/JDK Paths* on the left-hand menu. The displayed web page looks like Figure 7.13.

Figure 7.13. Configuration setting to run JSPs.



Path variables for locating JAR/Class files for Servlets

Choose either JDK or JRE.

[✓ JDK]

JDK Path: /usr/java1.2

JDK Runtime Libpath:

JDK Runtime Classpath:

JRE

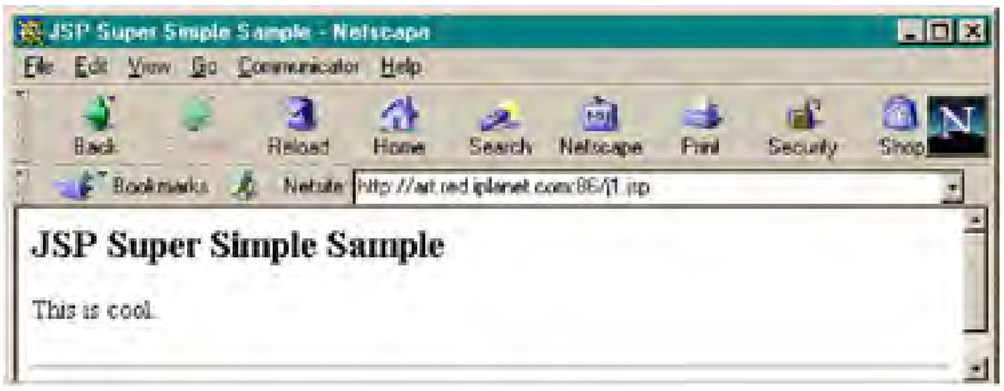
During the installation of your Sun ONE Web Server, you used the default JRE which has the JVM but not the Java compiler. To switch to the JDK, which also has the compiler, click the *JDK* button (see Figure 7.13) and enter the path of the installed JDK. After saving the changes, restart the web server instance.

In the document directory, create the JSP named /iplanet/iws60/docs/j1.jsp

```
<head>
<title>JSP Super Simple Sample</title>
</head>
<BODY  bgcolor="#FFFFFF" text="#000000">

<h2>JSP Super Simple Sample</h2>
<% String s1 = "cool"; %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

To test, from a web browser, go to the URL *http://water.red.iplanet.com:86/j1.jsp* (Figure 7.14).

Figure 7.14. Output from the JSP j1.jsp.


JSP Super Simple Sample - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security Shop
Bookmarks Netsite: http://air.red.iplanet.com:86/j1.jsp

JSP Super Simple Sample

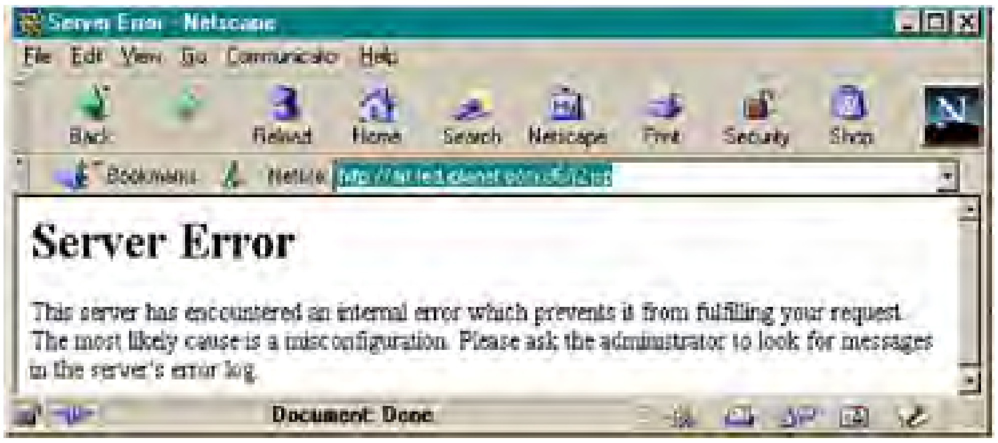
This is cool.

Again, setting up dynamic content is a straightforward process!

Error Log Checking

Of course, if there is an error in the JSP program code—say, omission of the semicolon in the first Java command (<% String s1 ="cool" %>)—an error will happen. Create j2.jsp based on j1.jsp but omitting the semicolon. The result will be the error output message shown in Figure 7.15.

Figure 7.15. Error output message from j2.jsp.


Server Error - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security Shop
Bookmarks Netsite: http://air.red.iplanet.com:86/j2.jsp
Document: Done

Server Error

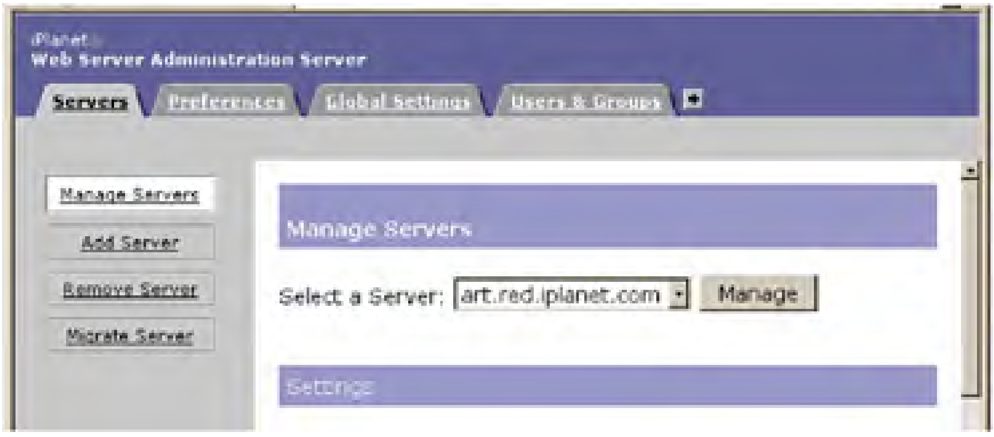
This server has encountered an internal error which prevents it from fulfilling your request. The most likely cause is a misconfiguration. Please ask the administrator to look for messages in the server's error log.

```
<head>
<title>JSP Super Simple Sample</title>
</head>
<BODY bgcolor="#FFFFFF" text="#000000">

<h2>JSP Super Simple Sample</h2>
```

```
<% String s1 = "cool" %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

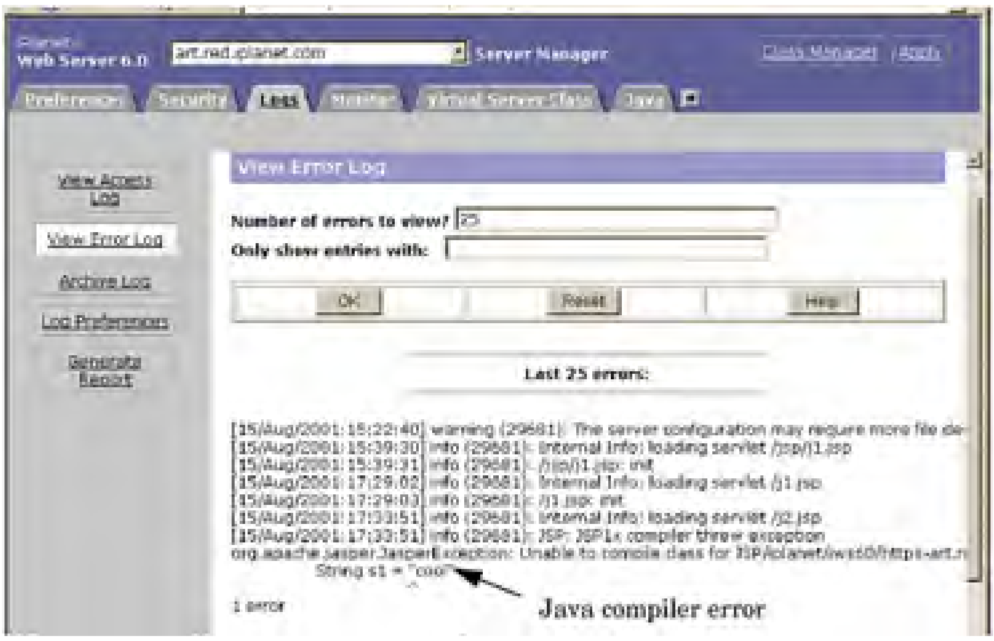
To debug the error, log in to the Sun ONE Web Server Administration Server main web page, select to manage your *servlet* server (Figure 7.16). Select the *Logs* tab on the top and the *View Error Log* from the menu on the left. Your error listing will look like Figure 7.17. Fix the error in j2.jsp and the JSP will work again.

Figure 7.16. Managing the servlet server from Sun ONE Web Server Administration Server main web page.



Select a Server: art.red.iplanet.com

Figure 7.17. Viewing error messages.



Number of errors to view? 25
Only show entries with:

Last 25 errors:

```
[15/Aug/2001:15:22:40] warning (29681): The server configuration may require more file de
[15/Aug/2001:15:39:30] info (29681): Internal Info: loading servlet /jsp/j1.jsp
[15/Aug/2001:15:39:31] info (29681): /jsp/j1.jsp: init
[15/Aug/2001:17:29:02] info (29681): Internal Info: loading servlet /j1.jsp
[15/Aug/2001:17:29:03] info (29681): /j1.jsp: init
[15/Aug/2001:17:33:51] info (29681): Internal Info: loading servlet /j2.jsp
[15/Aug/2001:17:33:51] info (29681): JSP: JSP1x compiler threw exception
org.apache.jasper.JasperException: Unable to compile class for JSP/iplanet/ws60/https-art.r
        String s1 = "cool"
                  ^
1 error
```

JSPs allow a person to create flexible, powerful, dynamic web pages. They are similar in concept to embedding Structured Query Language (SQL) database calls into C programs. However, what JSPs are in fact doing is embedding Java program calls into HTML.

So go for it, use the power...

Summary

Dynamic content is essential on any web site—makes a web site come alive. Configuring the Sun ONE Web Server for dynamic content is a very straightforward process:

1. Configure a program directory using the web administration system.
2. Make the program directory on the web server computer.
3. Write a program in the program directory or write the program somewhere else and load or copy the program into the program directory.
4. If it is a CGI program on UNIX, make the program file executable.
5. Test the program by using a web browser.

Although CGI programming is offered by most Internet Server Providers (ISPs), I wish more of them would also offer a Java option. The Java program environment is so easy for the ISP to set up, and the Java program language is such an excellent language for programming Internet programs.

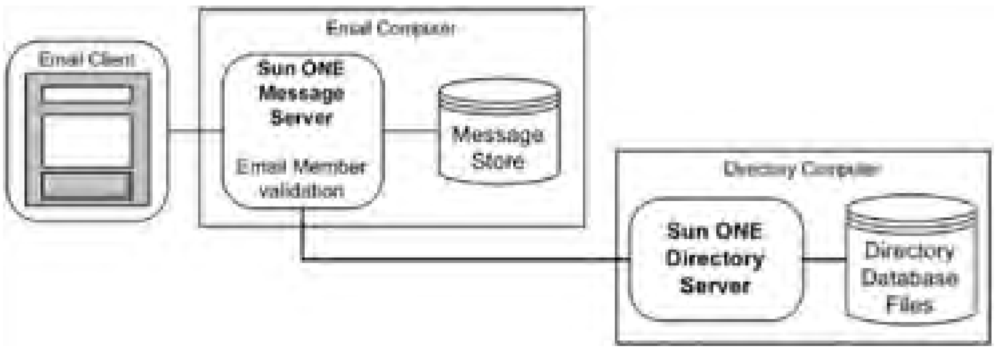
Chapter 8. The Sun ONE Directory Server

The Sun ONE Directory Server is a specialized hierarchical database system that is reliable and scalable. Its main specialties are very fast, efficient data retrieval; communication over a network; data stored in a hierarchical structure; and data replication across a number of directory servers. It is an extremely flexible system in a multiple server setup and includes the following features:

- The base installation contains a *base* schema to store information.
- The schema is modified by using command-line tools. Since these tools can be run from a script, application developers can write scripts to automatically update the directory schema.
- The schema can be modified using a Graphical User Interface (GUI) which a Sun ONE Console has (included with the Sun ONE Directory Server).
- Indexes are created to speed up searches.
- The Sun ONE Directory Server contains data security access systems such as ACLs and Access Control Information (ACI).
- The Sun ONE Directory Server is massively scalable.
- The Sun ONE Directory Server contains multiple backup and restore systems.

A typical example of information stored in a directory includes name and address information. Email is a common application that uses a directory (see Figure 8.1).

Figure 8.1. Email application using the Sun ONE Directory Server to validate member information.



When an email client logs in to the Sun ONE Message Server, the directory is used to validate user preferences. The directory server stores the users contact information (names and addresses) and email information.

Application Membership

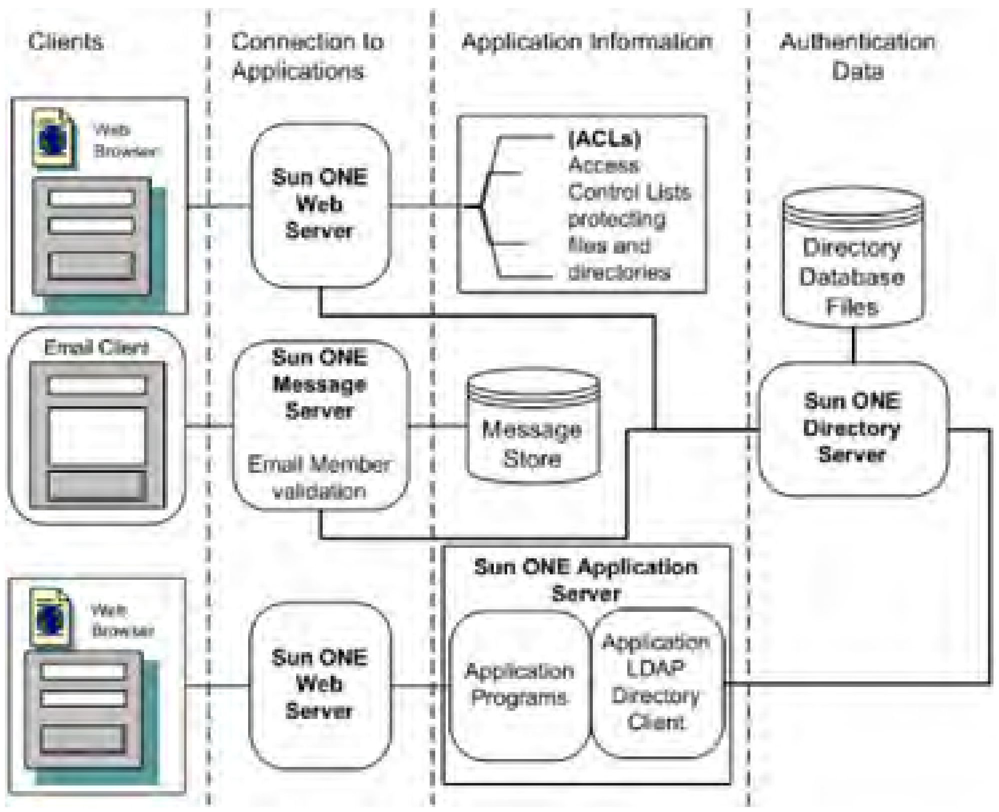
The Sun ONE Directory Server is designed to communicate with multiple applications over a network. An application must be configured to connect to the directory. Once connected, the application issues commands to the directory to do things such as maintain directory data (add, modify, and delete) or to retrieve and use the data in the directory. A common task directories can perform for applications is to authenticate users before they can use the application.

When you first connect to an application, the first thing it asks you is to identify yourself and to log in to the application. Once you are logged in and permissions have been verified, you can use the application.

Sun ONE Servers Using the Sun ONE Directory Server for Membership Authentication

Figure 8.2 shows three separate applications connecting over a network to the Sun ONE Directory Server. The Sun ONE Web Server, the Sun ONE Message Server, and the Sun ONE Application Server are all designed for easy connection to the Sun ONE Directory Server.

Figure 8.2. Login authentication through the Sun ONE Directory Server.



Consider the following examples of membership authentication. Don comes to work in the morning and checks his email—his email client connects to the email server. Later in the morning prior to a meeting, Don logs in to the human resources application to find out who the meeting participants work for—his web browser connects to the human resources application server. In the afternoon, an administrator makes a new program available to people in her group—her web browser connects to the Sun ONE Web Server group maintenance application.

In each of these instances, the members' data is stored in the Sun ONE Directory Server in a single location, and multiple applications access this single directory data store.

Each time members log in to an application, they must be authenticated before they are allowed to access the application. Each application goes through the following steps:

1. A member connects to the server application using a client program such as a browser or an email client.
2. The member is prompted for his user ID and password. He enters his user ID and password into a form and clicks a button to initiate the login sequence.

3. The sequence starts with the user ID and password information going from the client to the server application.
4. The server application passes the data over the network to the directory server.
5. The directory server checks its directory (database files) and returns a valid or invalid response.
6. The server application acts appropriately and sends back a notice that the member is authorized or not authorized on this application.

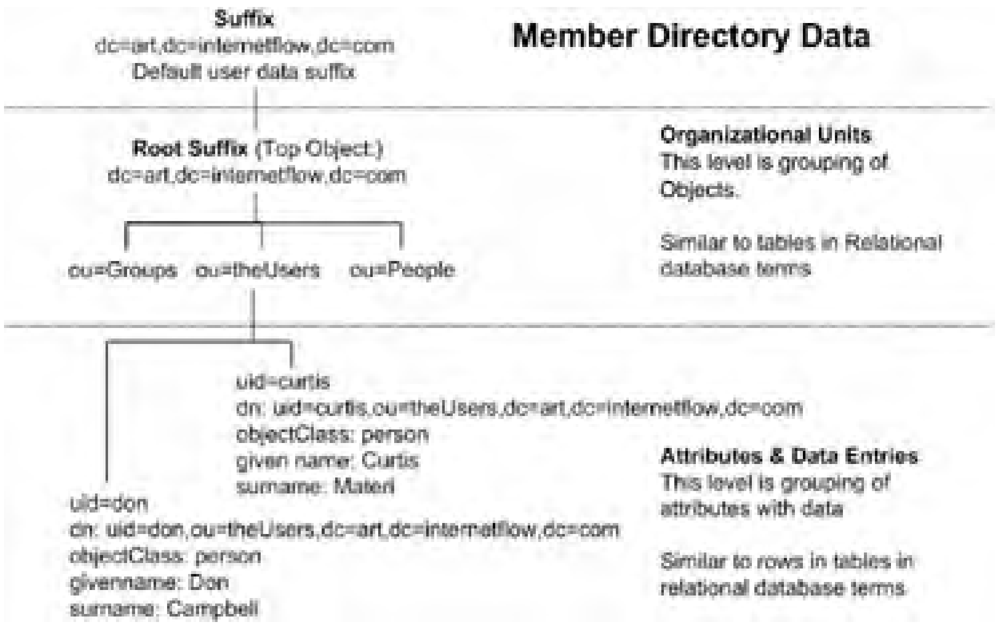
Once authorized, the member has access to the appropriate resources of the application, depending on the membership information.

Of course, before there can be authentication, there must be data in the directory. The Sun ONE Web Server and the Sun ONE Messaging Server come with applications to maintain member data and member authorization information. The Sun ONE Directory Server also comes with a console program to do simple data maintenance. Applications that are run on the Sun ONE Application Server do not come with programs to maintain member data on the directory server. Member data maintenance will need to be programmed into these applications.

Application Data in a Directory

Because the Sun ONE Directory Server comes with a base schema to store information, it is a straightforward process to set up a directory of members. The base schema has been developed over a number of years and is quite robust and functional. It includes definitions for storing group definitions, as well as member names and addresses. For example, to set up the directory described in Figure 8.3, I did not have to manually add data definitions. In this figure, the top level is the root suffix:

Figure 8.3. User directory data hierarchy.


Suffix
dc=art,dc=internetflow,dc=com
Default user data suffix
Member Directory Data
Root Suffix (Top Object)
dc=art,dc=internetflow,dc=com
Organizational Units
This level is grouping of Objects.
Similar to tables in Relational database terms
ou=Groups
ou=theUsers
ou=People
uid=curtis
dn: uid=curtis,ou=theUsers,dc=art,dc=internetflow,dc=com
objectClass: person
given name: Curtis
surname: Materi
Attributes & Data Entries
This level is grouping of attributes with data
Similar to rows in tables in relational database terms
uid=don
dn: uid=don,ou=theUsers,dc=art,dc=internetflow,dc=com
objectClass: person
givenname: Don
surname: Campbell

dc=art,dc=internetflow,dc=com

The levels under the root suffix are organizational units. The organizational units have objects attached to them. The members uid=don and uid=curtis are attached to the organizational unit ou=theUsers. The member data is stored on the third level.

uid=don is called an attribute pair; uid is the attribute, and don is the unit of data attached to the attribute. In the schema, uid has a definition that describes what type of data can be stored in a uid.

uid=don, givenname=Don, and surname=Campbell are attribute pairs that make up an object identified by what is called the Distinguished Name (DN). In the case of uid=don, the DN is

uid=don,ou=theUsers,dc=art,dc=internetflow,dc=com

The DN must be unique.

uid=don is called a *special attribute pair* in that it is the left-most attribute pair in a DN. The special attribute pair is called the Relative Distinguished Name (RDN).

Since it is extremely important to be able to describe directory data and schemas, you should understand the following points describing the directory pictured in Figure 8.3.

- The data, Don, is stored in the attribute givenname. The data, Campbell, is stored in the attribute surname. The data, don, is stored in the attribute uid.
- The attributes of givenname, surname, and uid make up the object called uid=don, which is uniquely identified by the DN uid=don,ou=theUsers,dc=art,dc=internet-flow,dc=com.
- uid=don is the RDN, the left-most attribute pair.
- The uid=don object is described by an objectClass called person, which is defined as having (at least) the attributes of givenname, surname, and uid.
- The uid=don object is attached to the ou=theUser object. In other words, this object has an attribute that is also an object, and in this case the object is the uid=don object.
- uid=don and uid=curtis are under the organizational unit with the DN of ou=theUsers,dc=art,dc=internet-flow,dc=com.
- The object, theUsers, is attached to the root suffix dc=art,dc=internetflow,dc=com.

Given a schema, data is organized and loaded into the directory. Groups are set up, members are added, and members are assigned to groups. Each of the groups is given authorization to the appropriate resources of the application.

Applications Interpret Data into Authorization

To illustrate data interpretation into authorization, consider this example. A user logs in to a purchasing system and gains access to the application areas she is authorized to enter. Which areas are authorized depends on how the application was set up and which group a member is in. Authorization is based on the data stored in the directory.

In the member directory described in Figure 8.3, Curtis Materi (uid=curtis) is assigned to the group administrator. Administrators are authorized to add new members. Curtis uses the purchasing application to add Don Campbell (uid=don) into the directory. Curtis puts Don into the purchasing group (a buying member). The application authorizes anybody in the group purchasing to create purchase orders. Since Don is in the purchasing group, he can create purchase orders. The application interprets the directory data to allow Curtis to add members and Don to create purchase orders.

[Team LiB]

Spreading Directory Data Around

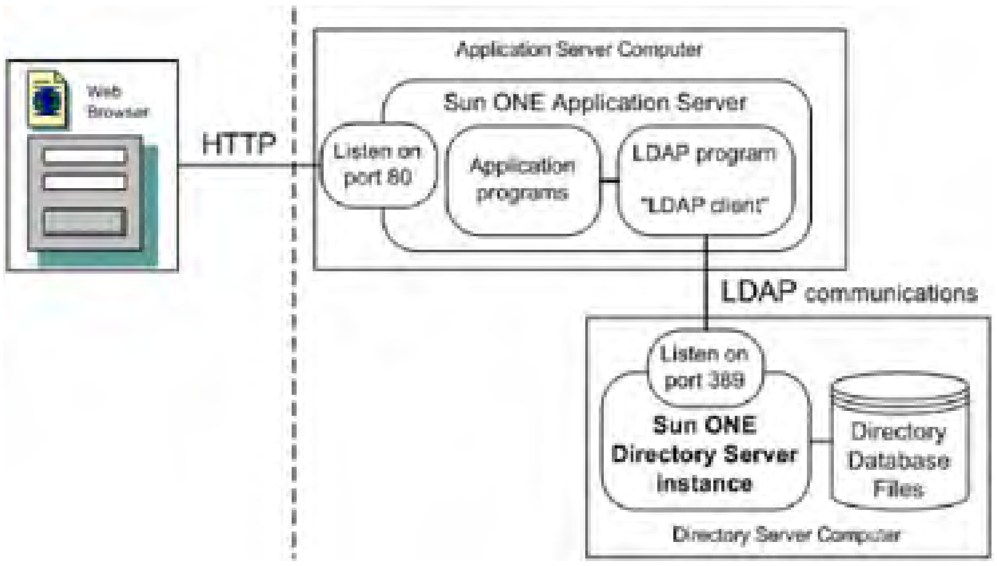
So far, this chapter has focused on applications using directory data and the way in which data is configured in a directory. Now let's look at how applications communicate with a directory.

A huge strength of the Sun ONE Directory Server is its ability to spread data across a single computer, across an intranet within a company, or across the Internet. There are two key features that make this possible: First, the Sun ONE Directory Server communicates over networks using Lightweight Directory Access Protocol (LDAP). The other feature is the capability of replication, i.e., duplicating data across a number of Sun ONE Directory Servers.

The Directory Server and the LDAP Client

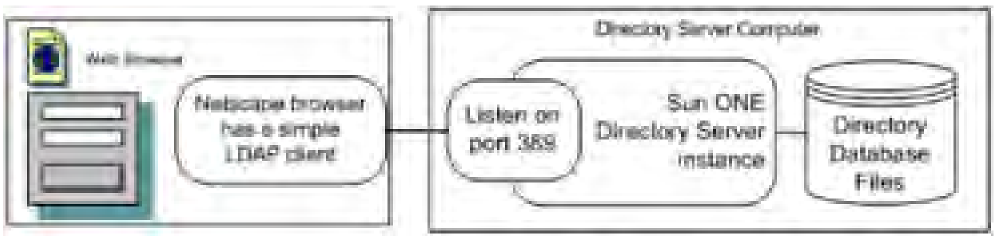
The Sun ONE Directory Server is designed to communicate over a network. This allows multiple applications on a single computer to communicate with the directory server; it also enables applications spread across a number of computers to use the same directory server. The tool used to connect an application to a directory server is an LDAP client. Applications that use the Sun ONE Directory Server database must be able to "talk" LDAP. Figure 8.4 shows a simple example of an LDAP client talking to a directory server.

Figure 8.4. LDAP client communicating with the Sun ONE Directory Server.



Applications running on the Sun ONE Application Server also use LDAP to communicate with the directory server. Figure 8.5 shows the LDAP client as part of the application. Through an LDAP client, the application can do the normal directory/database functions: search, add, edit/change, and delete.

Figure 8.5. The Netscape browser communicating with the directory server using LDAP.


Web Browser
Netscape browser has a simple LDAP client
Directory Server Computer
Listen on port 389
Sun ONE Directory Server instance
Directory Database Files

Both the web server and the directory server are designed to work over a network. There are similarities between the two in that clients communicating with a directory server and clients communicating with a web server both have default (commonly used) ports. The default port that the directory server listens to is port *389,* and for web servers the default port number is *80*. In both cases, the client connects to the server and makes requests. The server then returns a response to the client.

An example of a browser URL might be

ldap://art.internetflow.com/ou=people,dc=art,dc=internetflow, dc=com?x?sub

And a sample browser request to a web server:

```
www.internetflow.com/index.html
```

The browser requests the web server to return the index.html file.

Here is a sample browser request to a directory server:

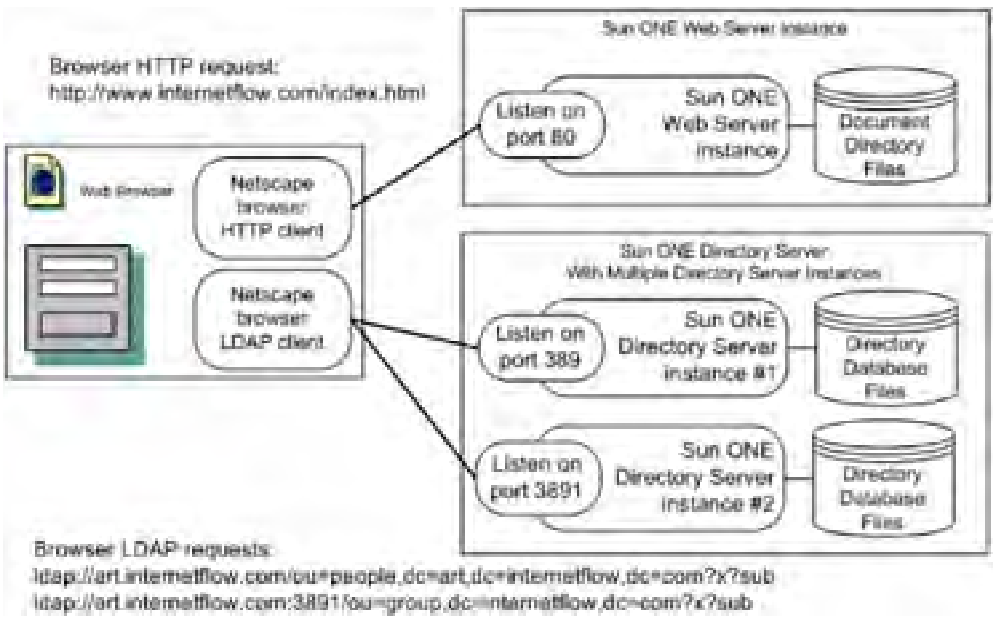
```
ldap://art.internetflow.com:389/ou=people,dc=art,
dc=internetflow,dc=com?x?sub
```

The browser requests the directory server to return a list of people. There are details and examples of using the browser to communicate with a directory server in later chapters.

The main point is that a client communicates over a network to make a request of a server, and the server responds to the request by sending something back. Between a client and a web server, the communication is HTTP. Between a client and a directory server, the communication is LDAP.

Just like the Sun ONE Web Server, the Sun ONE Directory Server has server instances. And, like web server instances, each of the directory server instances listens on a port. Figure 8.6 shows the dexterity of the Netscape browser as it talks to both a web server and a directory server on the appropriate ports.

Figure 8.6. Using the browser to make HTTP and LDAP requests from multiple instances.

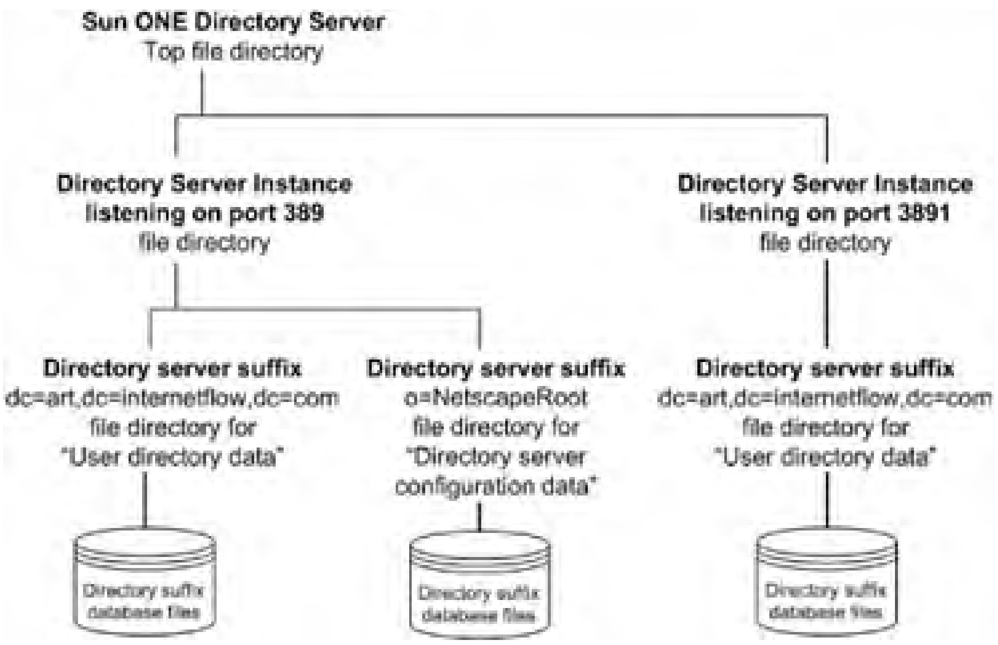

Browser HTTP request:
http://www.internetflow.com/index.html
Web Browser
Netscape browser HTTP client
Netscape browser LDAP client
Sun ONE Web Server instance
Listen on port 80
Sun ONE Web Server instance
Document Directory Files
Sun ONE Directory Server With Multiple Directory Server Instances
Listen on port 389
Sun ONE Directory Server instance #1
Directory Database Files
Listen on port 3891
Sun ONE Directory Server instance #2
Directory Database Files
Browser LDAP requests:
ldap://art.internetflow.com/ou=people,dc=art,dc=internetflow,dc=com?x?sub
ldap://art.internetflow.com:3891/ou=group,dc=internetflow,dc=com?x?sub

Physical File Hierarchy on a Single Computer

Figure 8.3 showed a diagram of directory data under a root suffix. A directory instance, however, can have multiple root suffixes. A suffix node is a root node for a directory tree of data. Or, in other words, each suffix node is a directory database. A directory database is made up of a set of *directory database files*.

The directory server installation (install) directory is the top-level file directory. Each directory instance has its own file directory, and each suffix root or database has its own subdirectory. This is illustrated in Figure 8.7.

Figure 8.7. Sun ONE Directory Server physical file hierarchy.


Sun ONE Directory Server
Top file directory
Directory Server Instance listening on port 389 file directory
Directory Server Instance listening on port 3891 file directory
Directory server suffix dc=art,dc=internetflow,dc=com file directory for "User directory data"
Directory server suffix o=NetscapeRoot file directory for "Directory server configuration data"
Directory server suffix dc=art,dc=internetflow,dc=com file directory for "User directory data"
Directory suffix database files
Directory suffix database files
Directory suffix database files

In a Sun ONE Directory Server installation, data is separated into:

- User directory data— applications will search, add, edit, and delete data in this directory tree database (dc=art,dc=internetflow,dc=com).
- Sun ONE Directory Server configuration data— when a Sun ONE Directory Server is installed, configuration data is stored in a directory under a suffix node called o=NetscapeRoot.

Directory Replication

Replication is the duplication of data from one directory to another directory. The concept of replication is simple. You have a master directory server and consumer directory servers. Only the master directory is updated, and then the master directory replicates/duplicates/sends the data to other directory servers called *consumer directories*. The consumer directory *consumes* data from the master directory. Once the data is consumed, the consumer directory serves the data to applications.

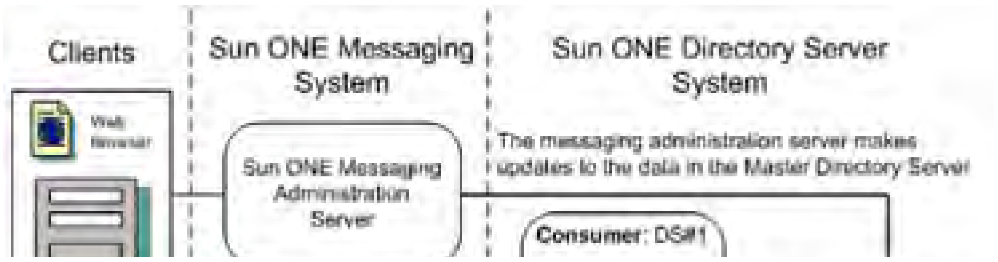
Replication improves data reliability, system scalability, and performance. Replication keeps applications performing at high levels. As application usage grows, more directory servers are added to maintain performance. This is called *scalability*. Here are the steps: When the current number of directory servers cannot handle the load, another directory server is added and data is replicated to the new directory server. Some applications are configured to use the new directory server, increasing the number of directory servers and maintaining performance.

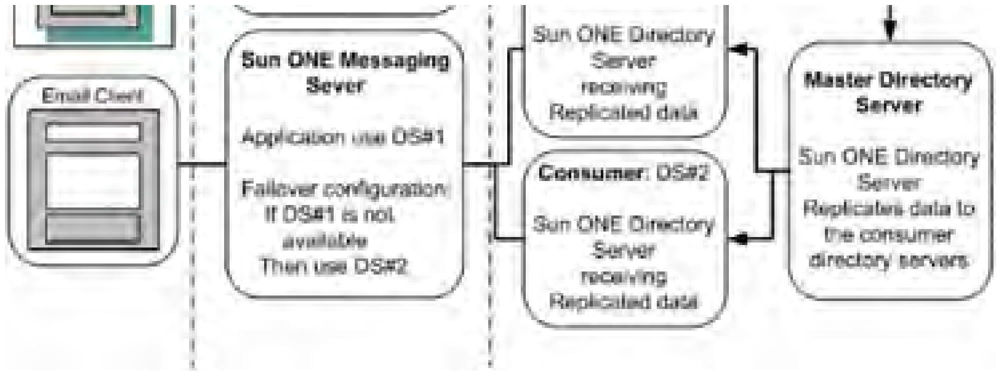
Replication for Failover

If one directory server is down, the application gets the data from another directory server. This is called *failover* or, to be more specific, application data failover.

Figure 8.8 shows two kinds of replication architecture in a directory application failover configuration. They are:

Figure 8.8. Sample replication architecture in a directory application failover configuration.


Clients
Sun ONE Messaging System
Sun ONE Directory Server System
Web Browser
Sun ONE Messaging Administration Server
The messaging administration server makes updates to the data in the Master Directory Server
Consumer: DS#1


Email Client
Sun ONE Messaging Sever
Application use DS#1
Failover configuration: If DS#1 is not available Then use DS#2
Sun ONE Directory Server receiving Replicated data
Consumer: DS#2
Sun ONE Directory Server receiving Replicated data
Master Directory Server
Sun ONE Directory Server Replicates data to the consumer directory servers

- Complete replication— when a new directory server is added, the master directory server replicates the complete database directory to DS#2.
- Update/change replication— after the complete database has been replicated, only the changes are replicated.

Large-Scale Replication

Figure 8.9 is a diagram of a large-scale setup with multiple applications and multiple directory servers used to create a hierarchical directory server system. The master directory server replicates data to the first line of directory servers (DS#1.1 and DS#1.2). Those directory servers feed the next line of consumer directory servers.

Figure 8.9. Large-scale replication to multiple directories used by multiple applications.

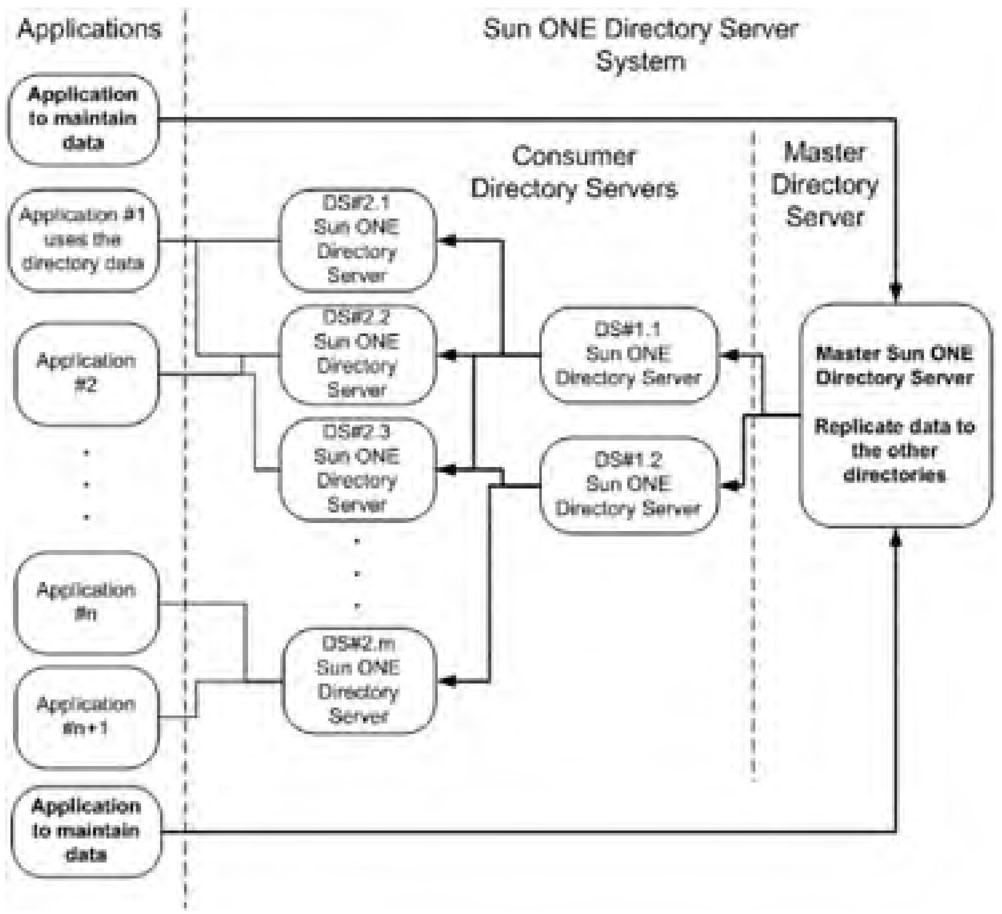

Applications
Sun ONE Directory Server System
Application to maintain data
Consumer Directory Servers
Master Directory Server
Application #1 uses the directory data
DS#2.1 Sun ONE Directory Server
DS#2.2 Sun ONE Directory Server
Application #2
DS#1.1 Sun ONE Directory Server
Master Sun ONE Directory Server
Replicate data to the other directories
DS#2.3 Sun ONE Directory Server
DS#1.2 Sun ONE Directory Server
Application #n
DS#2.m Sun ONE Directory Server
Application #n+1
Application to maintain data

Figure 8.9 illustrates two types of applications: applications that *maintain* directory data and applications that *use* the directory data. Only the master directory is updated. The master in turn updates the consumer directories.

Sun ONE Directory Server 5.1 and up have what is called *chaining*—multiple master directories that are updated. The master directories replicate to each other and then replicate to the consumer directories.

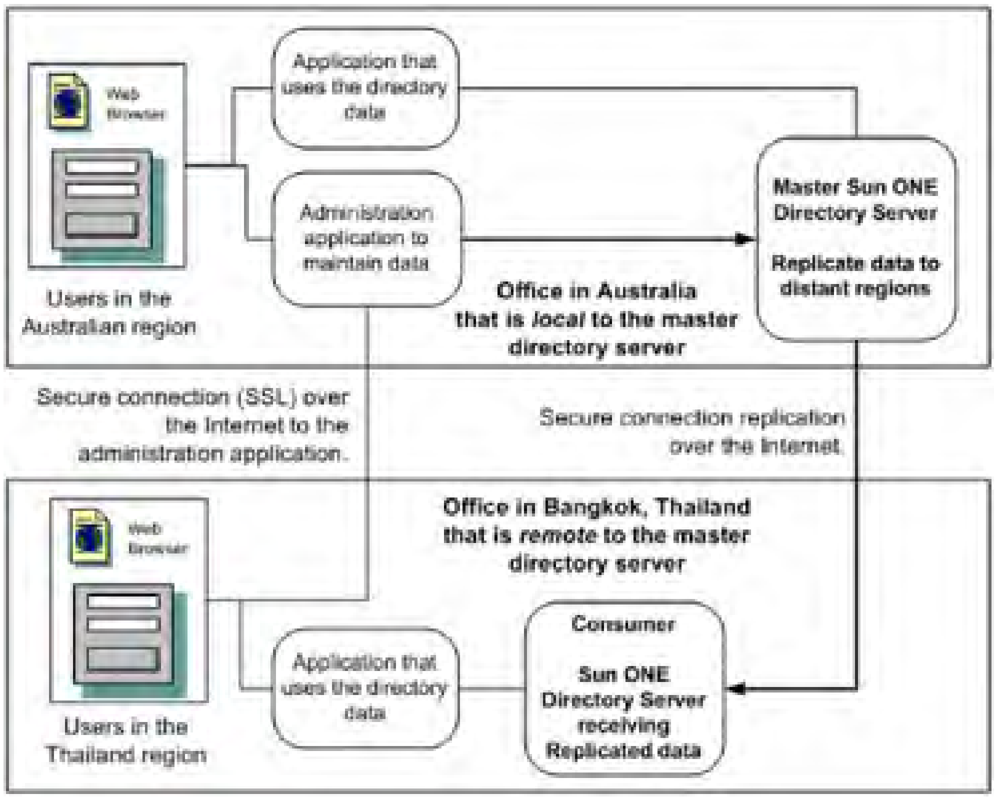
This type of hierarchical directory server system allows unlimited scalability. The number of servers in each level and the depth or number of levels can be increased.

Geographic Replication

In addition to failover and scalability, another reason for doing replication is to increase performance in remote offices. Replication improves performance for users in remote locations because local applications do not have to go a long distance to get data. Whether replication is done within the same physical computer room or over the Internet to other offices, consistent data is maintained and response times are excellent in both the local office and the remote office.

In the configuration described in Figure 8.10, all of the directory data can be replicated from Australia to Thailand. However, it is not necessary to replicate *all* the data. To be efficient, you could replicate the Thai data from the master directory to the Thailand consumer directory. This saves bandwidth and reduces the number of directory updates in the Thailand office. There is no need for the Thai directory to be updated every time someone changes information in the Australia office. Replicate only Thai data to the Thailand directory.

Figure 8.10. Replication to maintain performance in a remote office.


Web Browser
Users in the Australian region
Application that uses the directory data
Administration application to maintain data
Master Sun ONE Directory Server
Replicate data to distant regions
Office in Australia that is *local* to the master directory server
Secure connection (SSL) over the Internet to the administration application.
Secure connection replication over the Internet
Office in Bangkok, Thailand that is *remote* to the master directory server
Web Browser
Users in the Thailand region
Application that uses the directory data
Consumer
Sun ONE Directory Server receiving Replicated data

Note: In this configuration it is not necessary to back up the Thailand directory data. If the directory server in Thailand is replaced with a new directory server, simply replicate the data from the Australia directory server to the Thailand directory server.

Summary

The Sun ONE Directory Server is a specialized database system. The most common use of a directory server is to store user contact information. The information is used to validate users and give them access to application resources such as email servers and web site directories.

The data in a directory is stored in a hierarchical format. The top level is the root suffix. Under the suffix are the organizational units (ou). Under the organizational units are the actual data entries. Data entries are a group of attribute pairs. An attribute pair is an attribute paired with a unit of data, e.g., first name = Don.

Applications use the Internet protocol LDAP to communicate with the Sun ONE Directory Server to validate users and get data from the directory. LDAP is a standard protocol to allow LDAP clients to talk to the directories over a network. Applications that use directory data have an LDAP client built in that manages the communications with the directory servers.

When directories contain a large amount of data, the data is distributed to a number of directory servers organized into a hierarchy. At the top of the hierarchy is the server (or servers) that receives the updates to the directory data. The data updates are then distributed down the tree to each of the lower level directory servers at the base of the hierarchy. When applications retrieve data, they retrieve the data from the base-level directory servers. Because there are multiple base directory servers, the application has the option of getting data from more than one directory server. So, if one directory server is down, the application gets the data from another directory server.

Another way to distribute data is to replicate the data from one physical location to another physical location. Since the data may be distributed across the Internet, the directory servers can be located great distances apart, even on separate continents. This allows offices in remote locations to have their own directory server for fast retrievals of data from their local directory server.

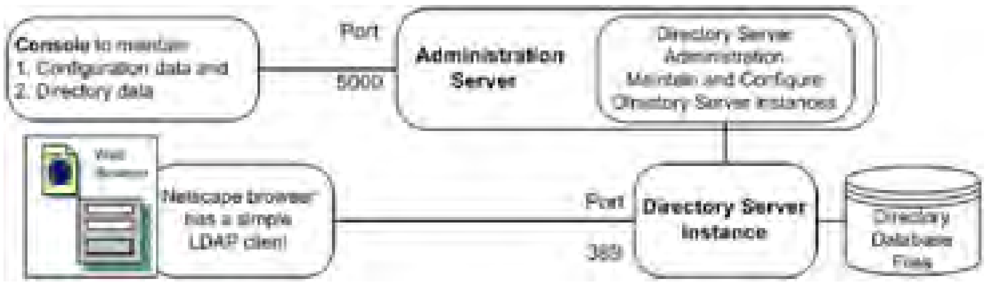
The Sun ONE Directory Server is used for small-scale storage of data on a single computer or it is used to build large-scale directory systems for millions of users spanning the globe. It is a very flexible, expandable, fast database system.

Chapter 9. Installing the Sun ONE Directory Server 5

We are now ready to install the Sun ONE Directory Server 5.0. [*] An installation of the directory server includes three major components (as shown in Figure 9.1):

[*] There has been a 5.1 release that requires Solaris patches. However, the 5.0 and 5.1 releases are the same in terms of the material covered in this chapter.

Figure 9.1. Sun ONE Directory Server components.

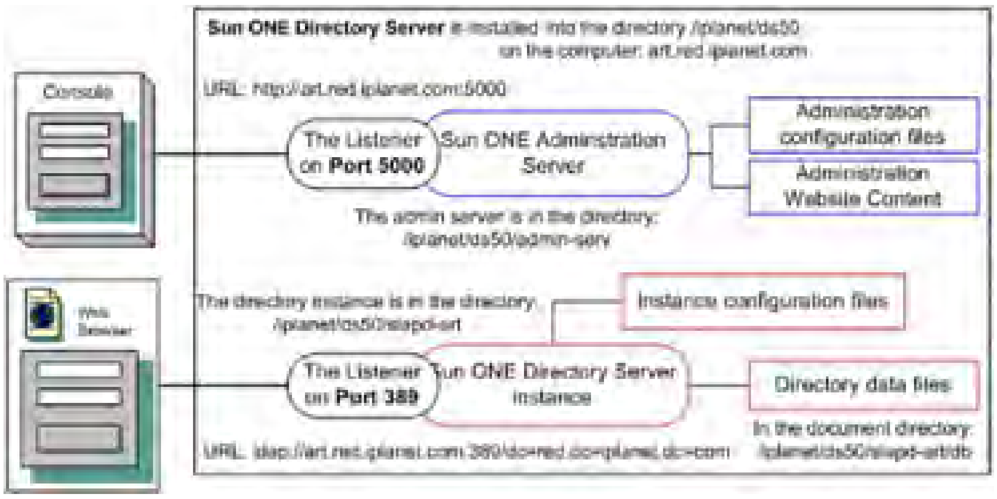

Console to maintain
1. Configuration data and
2. Directory data
Port
5000
Administration Server
Directory Server Administration Maintain and Configure Directory Server instances
Web Browser
Netscape browser has a simple LDAP client
Port
389
Directory Server instance
Directory Database Files

1. Directory server engine—to maintain and serve the directory data
2. Sun ONE Administration Server—to administer the directory instances
3. Administration tools:
    - Console—a GUI client tool to administer the directory server instances, maintain directory data, and modify the directory schema
    - Directory server command-line tools to do similar tasks

Installation Setup Values

During the installation, the installer gives the installation program the values to set up the complete Sun ONE Directory Server. Figure 9.2 shows the values that I used to set up my sample directory server. Note that all the directories for the directory server are under the root installation directory /iplanet/ds50.

Figure 9.2. Sun ONE Directory Server sample installation configuration.


Sun ONE Directory Server is installed into the directory /iplanet/ds50
on the computer: art.red.iplanet.com
URL: http://art.red.iplanet.com:5000
Console
The Listener on Port 5000
Sun ONE Administration Server
Administration configuration files
Administration Website Content
The admin server is in the directory: /iplanet/ds50/admin-serv
Web Browser
The directory instance is in the directory /iplanet/ds50/slapd-art
Instance configuration files
The Listener on Port 389
Sun ONE Directory Server instance
Directory data files
In the document directory: /iplanet/ds50/slapd-art/db
URL: ldap://art.red.iplanet.com:389/dc=red,dc=iplanet,dc=com

/iplanet/ds50/admin-serv is for the administration server.

/iplanet/ds50/slapd-art is for the directory instance.

/iplanet/ds50/docs is the document directory.

By keeping everything under one root directory, it is easy to track the location of all the Sun ONE Directory Server files.

Other values required during an installation are the *directory administration user* and the *administration user*. For the directory administration user, I used the directory administration user ID of *cn=Directory Manager* and the password *netscape*. As in other installations, I used the administration user ID *admin* and the password *admin*. When you start the Console to connect to the Sun ONE Administration Server, you are prompted for a user ID and a password. Both administration user IDs get you logged in to the administration server. However, only the *directory administration user ID* has authorization to modify the directory schema and data.

For UNIX installations, two UNIX IDs are required. One UNIX user ID is to run the administration processes, and the other is to run the directory server data instance process. The defaults of *nobody* to run the directory instances and *root* for the administration instances work fine.

Sun Solaris UNIX Installation

For my testing, I used a computer running Solaris 8 with a static IP address, hostname, and domain name.

The Sun ONE Directory Server for Solaris is on the DVD that is included with this book, or you can download it from *www.sun.com*. Follow these directions:

1. Once connected to *www.sun.com*, select *Downloads.*
2. Click on *View All.*
3. Find *Sun ONE Directory Server 5.0*. The products are listed in alphabetical order.
4. Select your operating system (in this case, *Solaris, 8 English*). Click *Download*.
5. Follow the instructions onscreen to download the software onto your computer.
6. Put the file into a working directory that you can retain and use later.
7. Decompress the file. Use these commands to decompress a tar.gz file:

```
gzip -d *.gz
tar xvf *.tar
```

The following listing shows the commands for a typical installation (note: ids50.tar.gz is the example file downloaded from *www.sun.com*):

```
# ls
ids50.tar.gz
# gzip -d *
#  tar -xvf *
x setup.inf, 1472 bytes, 3 tape blocks
x slapd, 0 bytes, 0 tape blocks
...

# ls
LICENSE.txt admin ids50.tar perldap setup.inf slapd
README.txt base nsperl setup silent.inf svrcorer
```

Running the Installation

After downloading and saving the software, you run the installation program, called setup.

```
# ./setup

Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
----------------------------------------------------
Welcome to the iPlanet Server Products installation pro-
  gram. This program will install iPlanet Server Prod-
  ucts and the iPlanet Console on your computer.

It is recommended that you have "root" privilege to
  install the software.

Tips for using the installation program:
  - Press "Enter" to choose the default and go to the
  next screen
  - Type "Control-B" to go back to the previous screen
  - Type "Control-C" to cancel the installation program
  - You can enter multiple items using commas to sepa-
  rate them.
 For example: 1, 2, 3

Would you like to continue with installation? [Yes]:
```

Press the Enter key to take the default [Yes].

(Note: You can use Ctrl+B to go back; however, this does not work if a previous selection has been committed.)

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
BY INSTALLING THIS SOFTWARE YOU ARE CONSENTING TO BE
  BOUND BY AND ARE BECOMING A PARTY TO THE AGREEMENT
  FOUND IN THE LICENSE.TXT FILE. IF YOU DO NOT AGREE TO
  ALL OF THE TERMS OF THIS AGREEMENT, PLEASE DO NOT
  INSTALL OR USE THIS SOFTWARE.

Do you agree to the license terms? [No]: Yes
```

If you agree with the terms, enter Yes.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
Select the items you would like to install:
1. iPlanet Servers
 Installs iPlanet Servers with the integrated iPlanet
  Console onto your computer.
2. iPlanet Console
 Installs iPlanet Console as a stand-alone Java applica-
  tion on your computer.

To accept the default, press the Enter key.

Select the component you want to install [1]:
```

Press the Enter key to take the default [1].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
Choose an installation type:

   1. Express installation
 Allows you to quickly install the servers using the
  most common options and predefined defaults. Useful
  for quick evaluation of the products.

   2. Typical installation
 Allows you to specify common defaults and options.

   3. Custom installation
 Allows you to specify more advanced options. This is
  recommended for experienced server administrators
  only.

To accept the default, press the Enter key.

Choose an installation type [2]:
```

Press the Enter key to take the default [2], typical install.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
This program will extract the server files and install
  them into a directory you specify. That directory is
  called the server root in the product documentation
  and will contain the server programs, the Administra-
  tion Server, and the server configuration files.

To accept the default, press the Enter key.

Install location [/usr/iplanet/servers]: /iplanet/ds50
```

You can put the server wherever you have enough disk space.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
iPlanet Server Products components:

Components with a number in () contain additional sub-
  components which you can select using subsequent
  screens.
 1. Server Core Components (3)
 2. iPlanet Directory Suite (2)
 3. Administration Services (2)

Specify the components you wish to install [All]:
```

Press the Enter key to take the default [All].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
----------------------------------------------------
Server Core Components components:

Components with a number in () contain additional sub-
  components which you can select using subsequent
  screens.

 1. Server Core Components
 2. Core Java classes
 3. Java Runtime Environment

Specify the components you wish to install [1, 2, 3]:
```

Hit the Enter key, to take the default [1, 2, 3].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
-------------------------------------------------------
iPlanet Directory Suite components:

Components with a number in () contain additional sub-
  components which you can select using subsequent
  screens.

 1. iPlanet Directory Server
 2. iPlanet Directory Server Console

Specify the components you wish to install [1, 2]:
```

Press the Enter key to take the default [1, 2].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------
Administration Services components:

Components with a number in () contain additional sub-
  components which you can select using subsequent
  screens.

 1. iPlanet Administration Server
 2. Administration Server Console

Specify the components you wish to install [1, 2]:
```

Press the Enter key to take the default [1, 2].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
----------------------------------------------------
Enter the fully qualified domain name of the computer
on which you're installing server software. Using the
  form
<hostname>.<domainname>
Example: eros.airius.com.

To accept the default, press the Enter key.

Computer name [art.red.iplanet.com]:
```

Press the Enter key to take the default <hostname>.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
----------------------------------------------------
Choose a UNIX user and group to represent the iPlanet
  server in the user directory.  The iPlanet server will
  run as this user. It is recommended that this user
  should have no privileges in the computer network sys-
  tem.  The Administration Server will give this group
  some permissions in the server root to perform server-
  specific operations.

If you have not yet created a user and group for the
  iPlanet server, create this user and group using your
  native UNIX system utilities.

To accept the default shown in brackets, press the
  Return key.
System User [nobody]: nobody
System Group [nobody]: nobody
```

For server installation and testing, I use *nobody*.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
Netscape server information is stored in the Netscape
  configuration directory server, which you may have
  already set up.  If so, you should configure this
  server to be managed by the configuration server.  To
  do so, the following information about the configura-
  tion server is required: the fully qualified host name
  of the form
<hostname>.<domainname>(e.g. hostname.domain.com), the
  port number, the suffix, and the DN and password of a
  user having permission to write the configuration
  information, usually the Netscape configuration direc-
  tory administrator. If you want to install this soft-
  ware as a standalone server, or if you want this
  instance to serve as your Netscape configuration
  directory server, press Enter.

Do you want to register this software with an existing
Netscape configuration directory server? [No]:
```

Press the Enter key to take the default [No].

Remember there are two types of data:

1. Configuration data that is stored in a Sun ONE Directory Server
2. User data (the directory database of user information)

You can store the configuration and user data separately or in the same directory server. For this example, server configuration data and user data are stored in the same directory server.

```
  Sun-Netscape Alliance
   iPlanet Server Products Installation/Uninstallation
  ---------------------------------------------------
  If you already have a directory server you want to use
    to store your data, such as user and group informa-
    tion, answer Yes to the following question.  You will
    be prompted for the host, port, suffix, and bind DN to
    use for that directory server.

  If you want this directory server to store your data,
    answer No.
  Do you want to use another directory to store your data?
[No]:
```

Hit the Enter key to take the default [No].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
The standard directory server network port number is
  389.  However, if you are not logged as the super-
  user, or port 389 is in use, the default value will be
  a random unused port number greater than 1024.
If you want to use port 389, make sure that you are
  logged in as the superuser, that port 389 is not in
  use, and that you run the admin server as the super-
  user.

Directory server network port [389]:
```

Press the Enter key to take the default [389].

Port 389 is the classic LDAP port. If there is another server on this port, choose another port number; for example, [3890].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
Each instance of a directory server requires a unique
  identifier.
Press Enter to accept the default, or type in another
  name and press Enter.

Directory server identifier [art]:
```

Press the Enter key to take the default [art].

A directory is created for this server instance; in this case, /iplanet/ds50/slapd-art.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
Please enter the administrator ID for the Netscape con-
  figuration directory server.  This is the ID typi-
  cally used to log in to the console.  You will also be
  prompted for the password.

Netscape configuration directory server
administrator ID [admin]:
Password:
Password (again):
```

Press the Enter key to take the default [admin]. For the password use *admin*.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
The suffix is the root of your directory tree.  You may
  have more than one suffix.
Suffix [dc=red, dc=iplanet, dc=com]:
```

Hit the Enter key to take the default [cd=<hostname>, dc=<your domain name part I>, dc=<root domain name>].

This is the top-level directory value. Under here, your member (e.g., email users), group, and other information will be stored.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
Certain directory server operations require an adminis-
  trative user. This user is referred to as the Direc-
  tory Manager and typically has a bind Distinguished
  Name (DN) of cn=Directory Manager. Press Enter to
  accept the default value, or enter another DN. In
  either case, you will be prompted for the password for
  this user. The password must be at least 8 characters
  long.

Directory Manager DN [cn=Directory Manager]:
Password:
Password (again):
```

Press the Enter key to take the default [cn=Directory Manager].

cn=Directory Directory is often used for a user ID when working with the directory from the Console Java application.

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------

The Administration Domain is a part of the configura-
  tion directory server used to store information about
  Netscape software. If you are managing multiple soft-
  ware releases at the same time, or managing informa-
  tion about multiple domains, you may use the
  Administration Domain to keep them separate.

If you are not using administrative domains, press Enter
  to select the default.  Otherwise, enter some descrip-
  tive, unique name for the administration domain, such
  as the name of the organization responsible for manag-
  ing the domain.

Administration Domain [red.iplanet.com]:
```

Press the Enter key to use the default [<your domain name>].

Installing the Directory Server Administration Server

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
The Administration Server is separate from any of your
  application servers since it listens to a different
  port and access to it is restricted.
Pick a port number between 1024 and 65535 to run your
  Administration Server on. You should NOT use a port
  number which you plan to run an application server on,
  rather, select a number which you will remember and
  which will not be used for anything else.

The default in brackets was randomly selected from the
  available ports on your system. To accept the default,
  press return.

Administration port [7475]: 5000
```

Enter something you can remember, e.g., 5000.

```
Netscape Communications Corporation
 Administration Installation/Uninstallation
------------------------------------------

The Administration Server program runs as a certain user
  on your system. This user should be different than the
  one which your application servers run as. Only the
  user you select will be able to write to your configu-
  ration files. If you run the Administration Server as
  "root", you will be able to use the Server Administra-
  tion screen to start and stop your application serv-
  ers.

Run Administration Server as [root]:
```

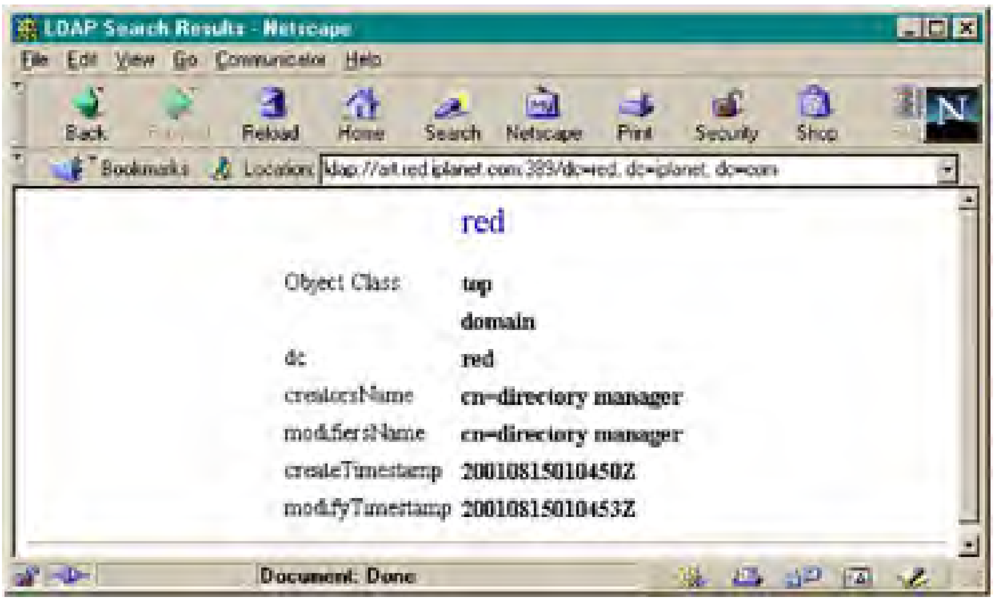
Press the Enter key to take the default [root].

```
Sun-Netscape Alliance
 iPlanet Server Products Installation/Uninstallation
---------------------------------------------------
Extracting Netscape core components...
Extracting Server Core Components...

Extracting Core Java classes...
Extracting Java Runtime Environment...
Extracting iPlanet Directory Server...
Extracting iPlanet Directory Server Console...
Extracting iPlanet Administration Server...
Extracting nsPerl 5.005_03...
Extracting PerLDAP 1.4.1...

[slapd-art]: starting up server...
[slapd-art]: [14/Aug/2001:18:04:44 -0700] - iPlanet-
  Directory/5.0 B2001.109.0903
 starting up
[slapd-art]: [14/Aug/2001:18:04:50 -0700] - slapd
  started.  Listening on all int
erfaces port 3891 for LDAP requests
Your new directory server has been started.
Created new Directory Server
Start Slapd  Starting Slapd server configuration.
Success Slapd Added Directory Server information to Con-
  figuration Server.
Configuring Administration Server...
Your parameters are now entered into the Administration
  Server database, and the Administration Server will be
  started.

Changing ownership to admin user root...
```

```
Setting up Administration Server Instance...
Configuring Administration Tasks in Directory Server...
Configuring Global Parameters in Directory Server...
iPlanet-WebServer-Enterprise/4.1SP3 BB1-09/17/2000 01:08

startup: listening to http://art.red.iplanet.com, port
  5000 as root

Press Return to continue...

Go to /iplanet/ds50 and type startconsole to begin man-
  aging your servers.
```

Success! Now you are ready to test the directory server. From a Netscape web browser, go to URL *ldap://art.red.iplanet.com:389/ dc=red,dc=iplanet,dc=com* (see Figure 9.3). (Note: The Netscape web browser uses the ldap:// protocol. This is convenient for testing.)

Figure 9.3. LDAP search.


LDAP Search Results - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop
Bookmarks Location: ldap://art.red.iplanet.com:389/dc=red, dc=iplanet, dc=com
Document: Done

red

```
Object Class      top
                  domain
dc                red
creatorsName      cn=directory manager
modifiersName     cn=directory manager
createTimestamp   20010815010450Z
modifyTimestamp   20010815010453Z
```

Uninstalling the Server

It is very easy to uninstall the server—stop the processes and remove the install directory and all subdirectories; for example, enter the command rm -r /iplanet/ds50. After the command completes, the Sun ONE Directory Server has been completely removed/uninstalled.

Reference Documentation

The Sun Microsystems web site at *http://docs.sun.com* has a considerable amount of documentation.

The link for the Sun ONE Web Server is *http://docs.sun.com/db/prod/sunone*.

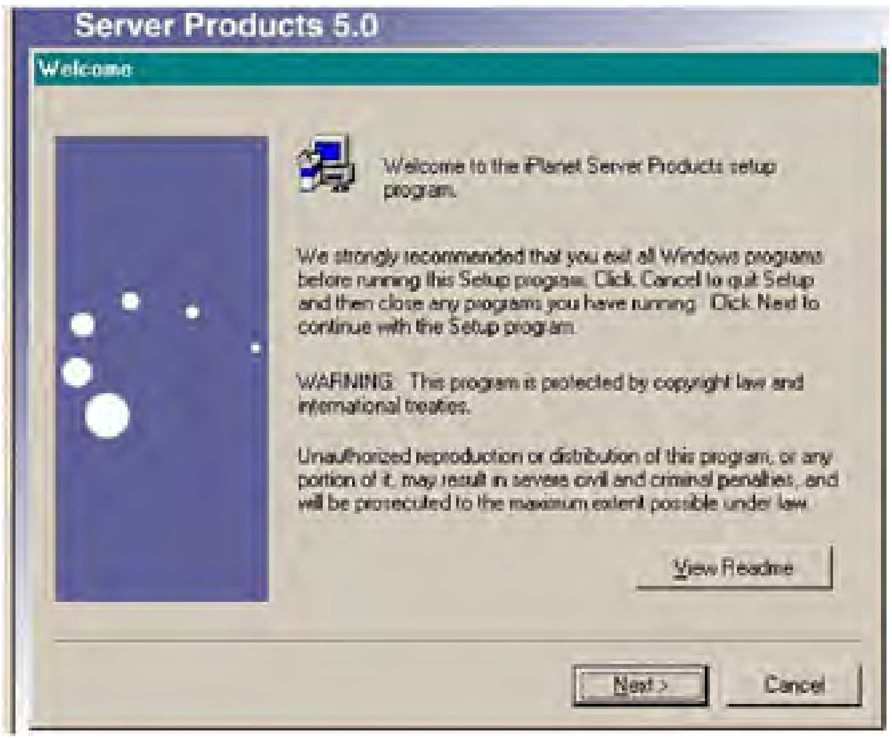
Windows Installation

The Sun ONE Directory Server for Windows is on the DVD included with this book, or you can download it from *www.sun.com*. Follow these instructions:

1. Once connected to *www.sun.com*, select *Downloads.*
2. Click on *View All.*
3. Find *Sun ONE Directory Server 5.0*; the products are listed in alphabetical order.
4. Then select your operating system (in this case, *Windows NT 4.0, 2000 Server English*). Click *Download*.
5. Then follow the onscreen instructions to download the software onto your computer.
6. Put the file into a working directory that you can retain and use later.

Once downloaded, run the executable file to start the installation. At that time, Figure 9.4 is displayed. Click *Next*.

Figure 9.4. Welcome screen.



Welcome to the iPlanet Server Products setup program.

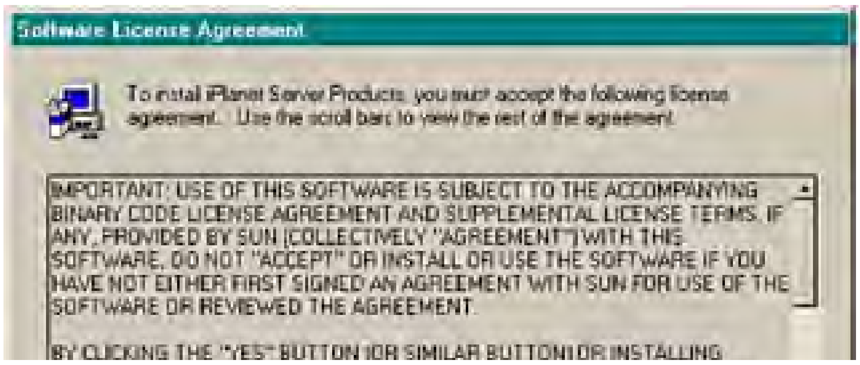
We strongly recommended that you exit all Windows programs before running this Setup program. Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

The license agreement appears next (Figure 9.5). Click *Yes* to accept the license.

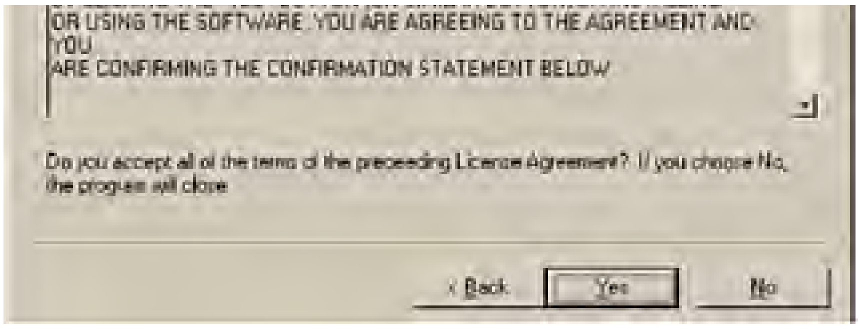
Figure 9.5. Software license agreement.



To install iPlanet Server Products, you must accept the following license agreement. Use the scroll bars to view the rest of the agreement.

IMPORTANT: USE OF THIS SOFTWARE IS SUBJECT TO THE ACCOMPANYING BINARY CODE LICENSE AGREEMENT AND SUPPLEMENTAL LICENSE TERMS, IF ANY, PROVIDED BY SUN (COLLECTIVELY "AGREEMENT") WITH THIS SOFTWARE. DO NOT "ACCEPT" OR INSTALL OR USE THE SOFTWARE IF YOU HAVE NOT EITHER FIRST SIGNED AN AGREEMENT WITH SUN FOR USE OF THE SOFTWARE OR REVIEWED THE AGREEMENT.

BY CLICKING THE "YES" BUTTON (OR SIMILAR BUTTON) OR INSTALLING

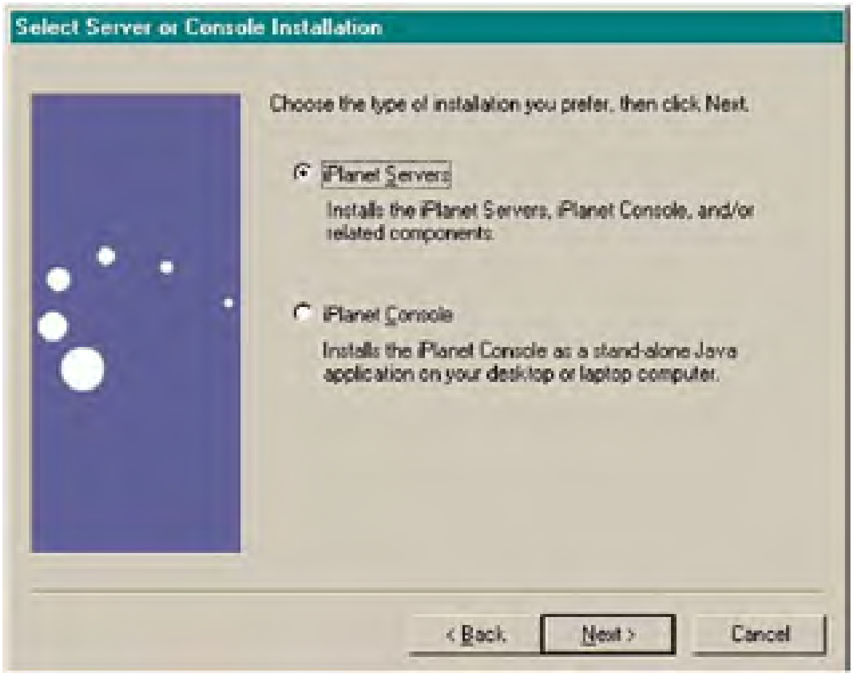


OR USING THE SOFTWARE, YOU ARE AGREEING TO THE AGREEMENT AND YOU
ARE CONFIRMING THE CONFIRMATION STATEMENT BELOW

Do you accept all of the terms of the preceding License Agreement? If you choose No, the program will close.

Select *iPlanet Servers,* and then click *Next* (Figure 9.6). (Note: Installing the Console as a standalone application is useful if managing other directory servers from another computer.)

Figure 9.6. Selecting servers.



Select Server or Console Installation

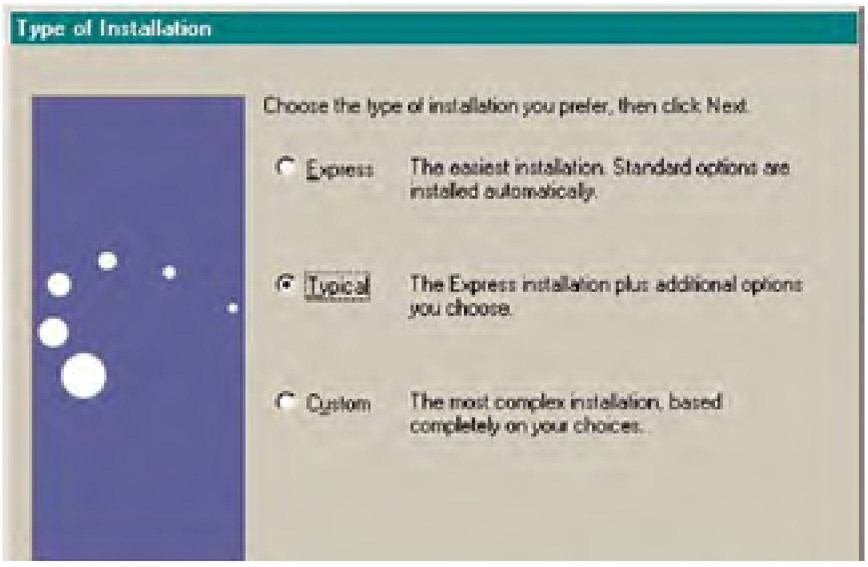
Choose the type of installation you prefer, then click Next.

(●) iPlanet Servers
Installs the iPlanet Servers, iPlanet Console, and/or related components.

( ) iPlanet Console
Installs the iPlanet Console as a stand-alone Java application on your desktop or laptop computer.

On the type of installation screen (Figure 9.7), select *Typical* and click *Next*.

Figure 9.7. Choosing type of installation.



Type of Installation

Choose the type of installation you prefer, then click Next.

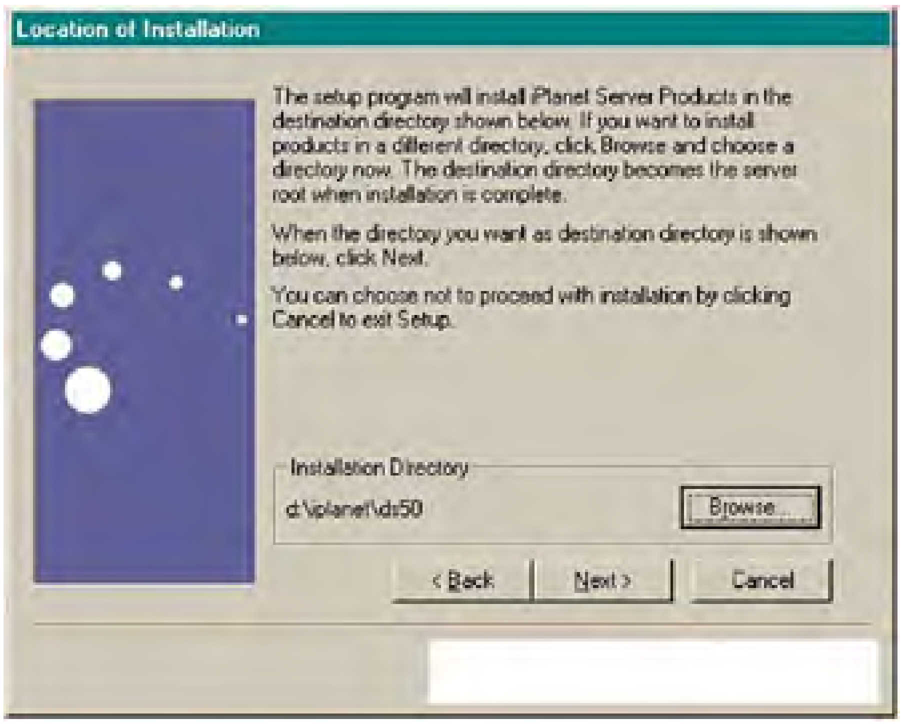
( ) Express — The easiest installation. Standard options are installed automatically.

(●) Typical — The Express installation plus additional options you choose.

( ) Custom — The most complex installation, based completely on your choices.

Select the directory in which you want the application installed and click *Next* (Figure 9.8). Normally, operating system and other programs are on the C drive. In this setup, the directory server is put in the d:\iplanet\ds50 directory.

Figure 9.8. Selecting the installation directory.



Location of Installation

The setup program will install iPlanet Server Products in the destination directory shown below. If you want to install products in a different directory, click Browse and choose a directory now. The destination directory becomes the server root when installation is complete.

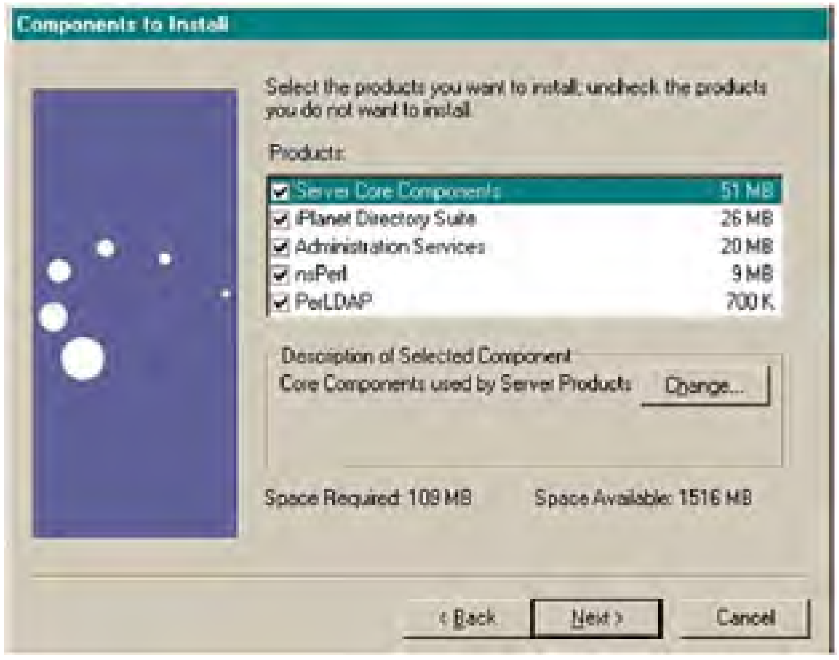
When the directory you want as destination directory is shown below, click Next.

You can choose not to proceed with installation by clicking Cancel to exit Setup.

Installation Directory
d:\iplanet\ds50

On the screen in Figure 9.9, select all the components, and then click *Next*.

Figure 9.9. Selecting installation components.



Components to Install

Select the products you want to install; uncheck the products you do not want to install.

Products:

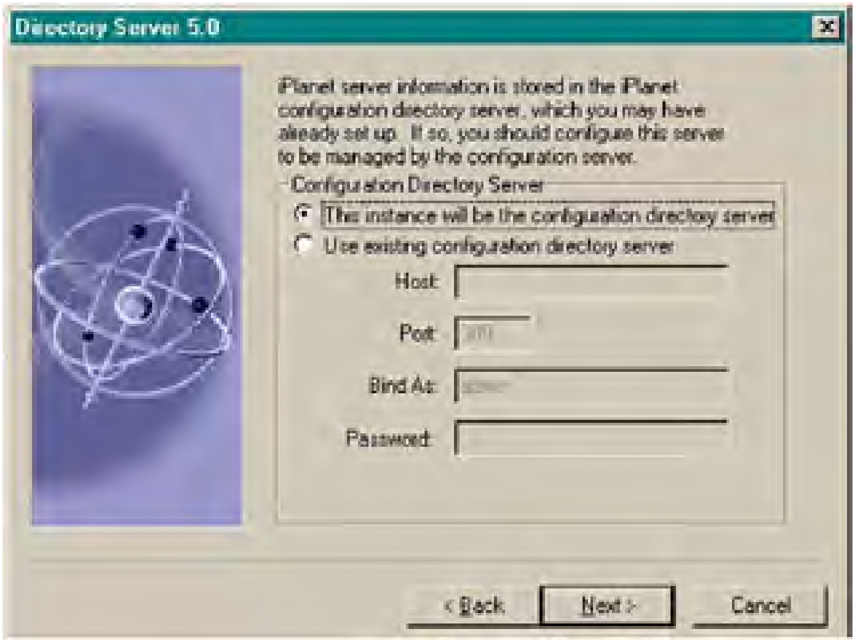
✓ Server Core Components	51 MB
✓ iPlanet Directory Suite	26 MB
✓ Administration Services	20 MB
✓ nsPerl	9 MB
✓ PerLDAP	700 K

Description of Selected Component
Core Components used by Server Products

Space Required: 109 MB   Space Available: 1516 MB

Figure 9.10 illustrates the options for selecting where to store configuration information about a directory server. Select the first option, *This new instance will be the configuration directory server*, and click *Next*. The directory server stores information about itself in a directory server instance.

Figure 9.10. Storing directory server information.



Directory Server 5.0

iPlanet server information is stored in the iPlanet configuration directory server, which you may have already set up. If so, you should configure this server to be managed by the configuration server.

Configuration Directory Server

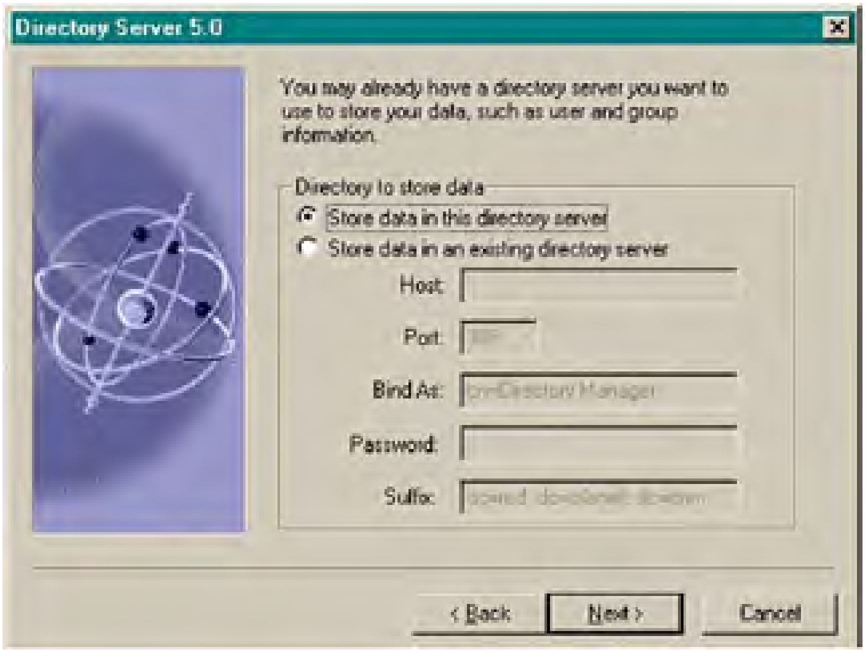
- [✓ This instance will be the configuration directory server]
- Use existing configuration directory server

Host:
Port: 389
Bind As: admin
Password:

< Back | Next > | Cancel

Figure 9.11 illustrates the option of selecting where to store directory data and information. In this example, the directory server will store its own data in this directory. Use the default *Store data in this directory server* and click *Next*.

Figure 9.11. Storing directory data.



Directory Server 5.0

You may already have a directory server you want to use to store your data, such as user and group information.

Directory to store data

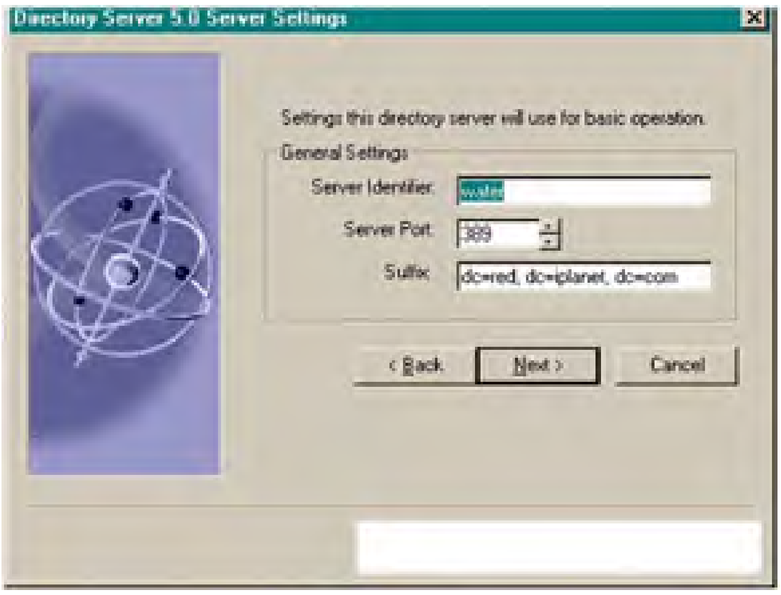
- [✓ Store data in this directory server]
- Store data in an existing directory server

Host:
Port: 389
Bind As: cn=Directory Manager
Password:
Suffix: [illegible]

< Back | Next > | Cancel

The *Server Identifier* field (Figure 9.12) defaults to your host name, which is used to create a directory server instance; for example, d:\iplanet\ds50\slapd-water. *Server Port 389* is the common LDAP server port. The *Suffix* field identifies the top of the directory. Accept the defaults and click *Next*.

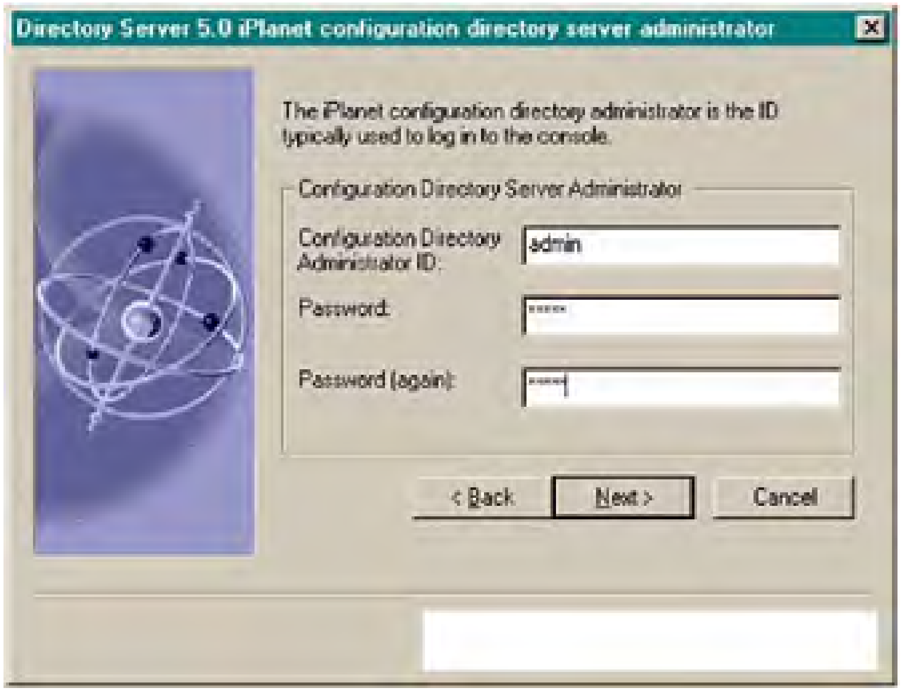
Figure 9.12. General settings.


Directory Server 5.0 Server Settings
Settings this directory server will use for basic operation.
General Settings
< Back
Next >
Cancel

Server Identifier: water
Server Port: 389
Suffix: dc=red, dc=iplanet, dc=com

Type *admin* for *Administrator ID* and *admin* for *Password* and click *Next* (Figure 9.13). When you put the system into production, you can change the password.

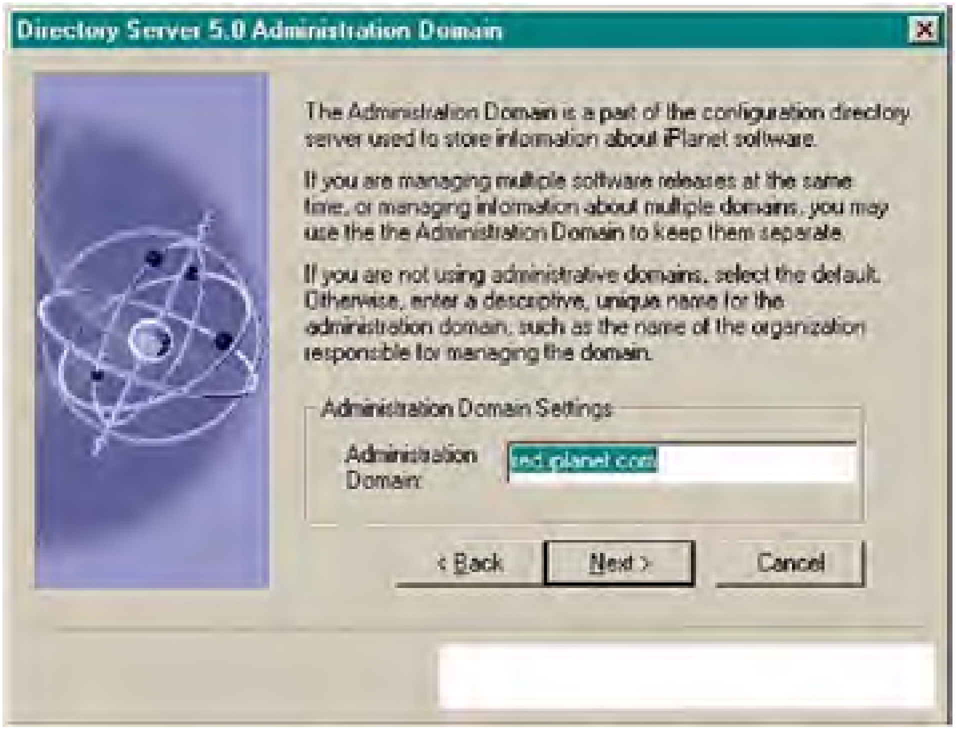
Figure 9.13. Administration user ID.


Directory Server 5.0 iPlanet configuration directory server administrator
The iPlanet configuration directory administrator is the ID typically used to log in to the console.
Configuration Directory Server Administrator
< Back
Next >
Cancel

Configuration Directory Administrator ID: admin
Password: *****
Password (again): *****

In the domain window (Figure 9.14), accept the default and click *Next*.

Figure 9.14. Administration domain settings.



The Administration Domain is a part of the configuration directory server used to store information about iPlanet software.

If you are managing multiple software releases at the same time, or managing information about multiple domains, you may use the the Administration Domain to keep them separate.

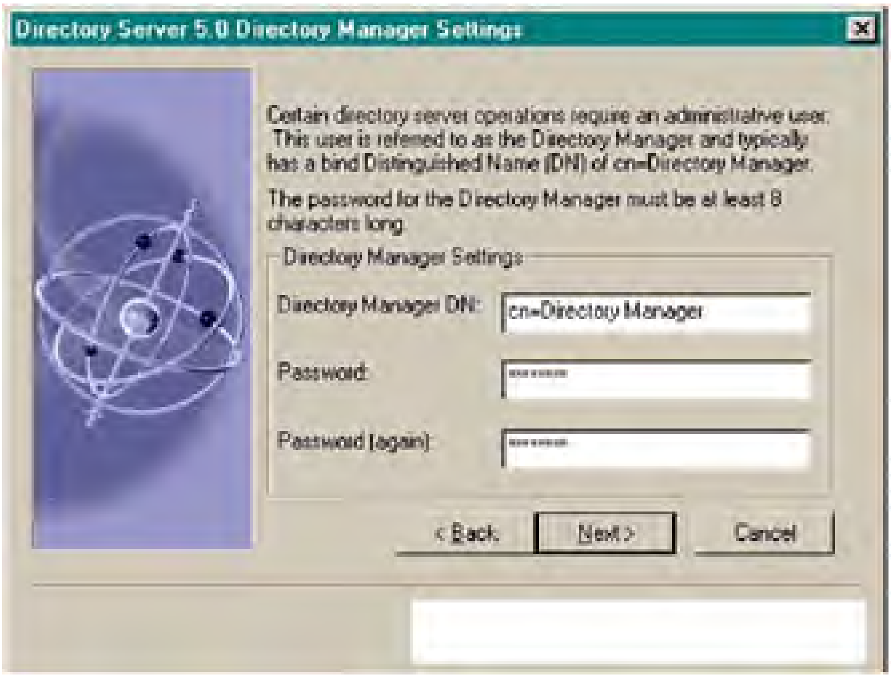
If you are not using administrative domains, select the default. Otherwise, enter a descriptive, unique name for the administration domain, such as the name of the organization responsible for managing the domain.

Administration Domain Settings

Administration Domain: sed.planet.com

Use the default directory administration user ID *cn=Directory Manager*, netscape as your password, and click *Next* (Figure 9.15).

Figure 9.15. Directory administration user ID settings.



Certain directory server operations require an administrative user. This user is referred to as the Directory Manager and typically has a bind Distinguished Name (DN) of cn=Directory Manager.

The password for the Directory Manager must be at least 8 characters long.

Directory Manager Settings

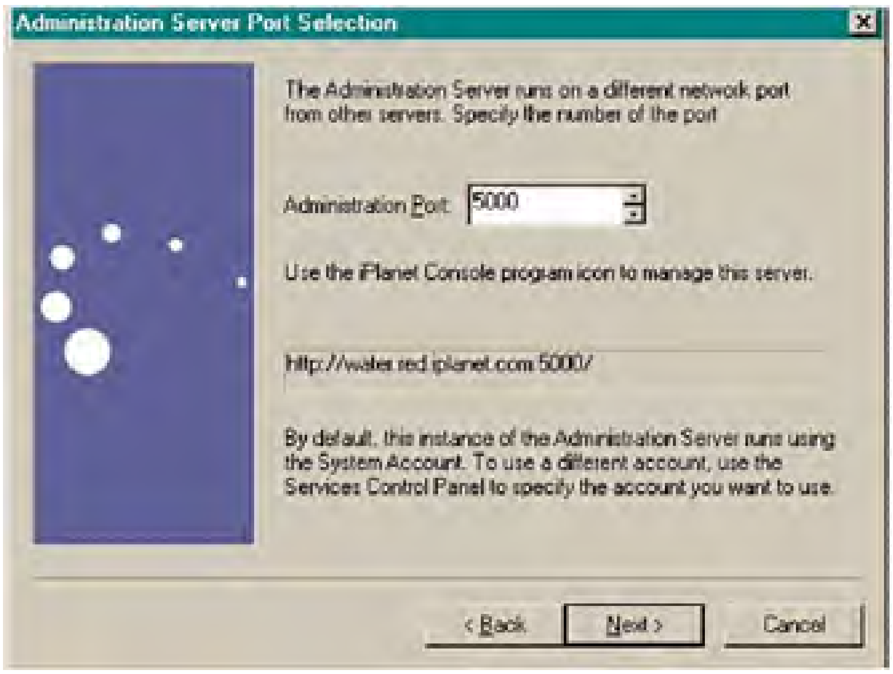
Directory Manager DN: cn=Directory Manager

Password: ********

Password (again): ********

Enter an *Administration Port of 5000* and click *Next* (Figure 9.16). The administration server listens on a port. Pick a number you can remember. You can change the port later. You should use the same number every time you install a system.

Figure 9.16. Administration server port selection.


Administration Server Port Selection

The Administration Server runs on a different network port from other servers. Specify the number of the port

Administration Port: 5000

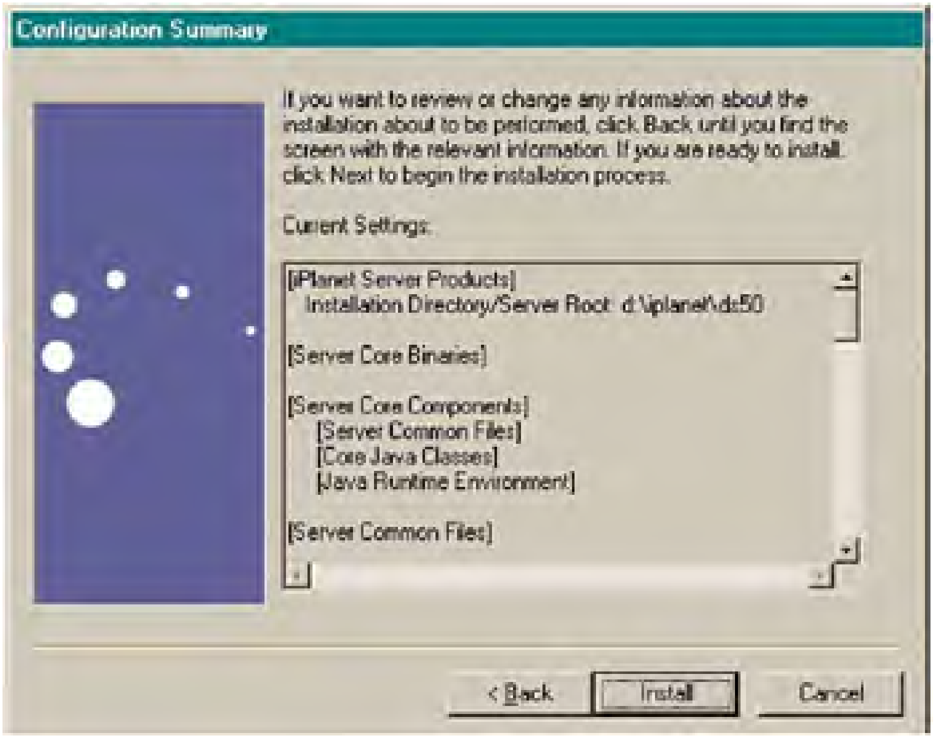
Use the iPlanet Console program icon to manage this server.

http://water.red.iplanet.com:5000/

By default, this instance of the Administration Server runs using the System Account. To use a different account, use the Services Control Panel to specify the account you want to use.

That is the end of input for the installation. Click *Install* in the summary window (Figure 9.17) to install the software.

Figure 9.17. Configuration summary.


Configuration Summary

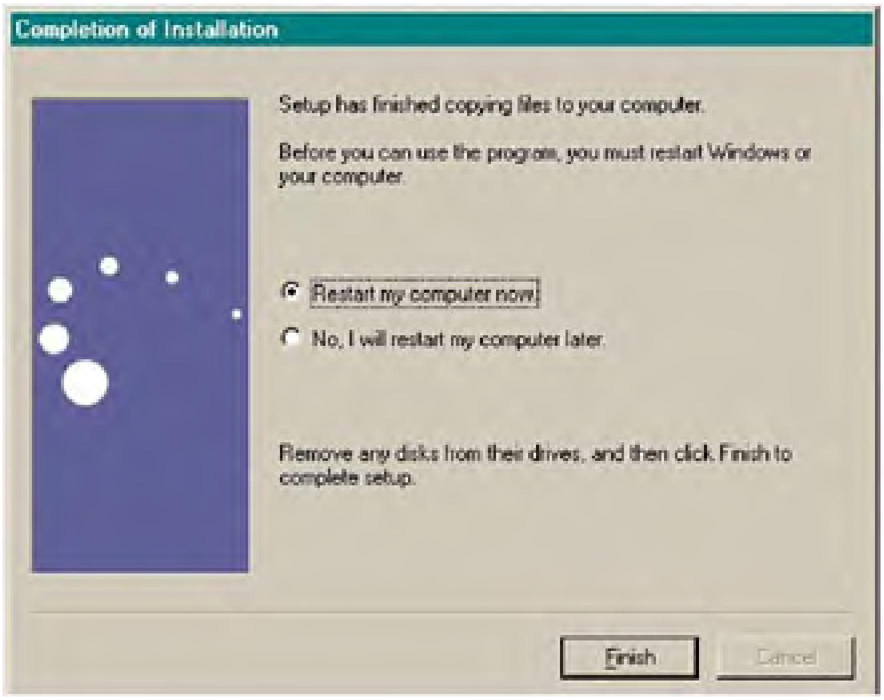
If you want to review or change any information about the installation about to be performed, click Back until you find the screen with the relevant information. If you are ready to install, click Next to begin the installation process.

Current Settings:

```
[iPlanet Server Products]
  Installation Directory/Server Root: d:\iplanet\ds50

[Server Core Binaries]

[Server Core Components]
    [Server Common Files]
    [Core Java Classes]
    [Java Runtime Environment]

[Server Common Files]
```

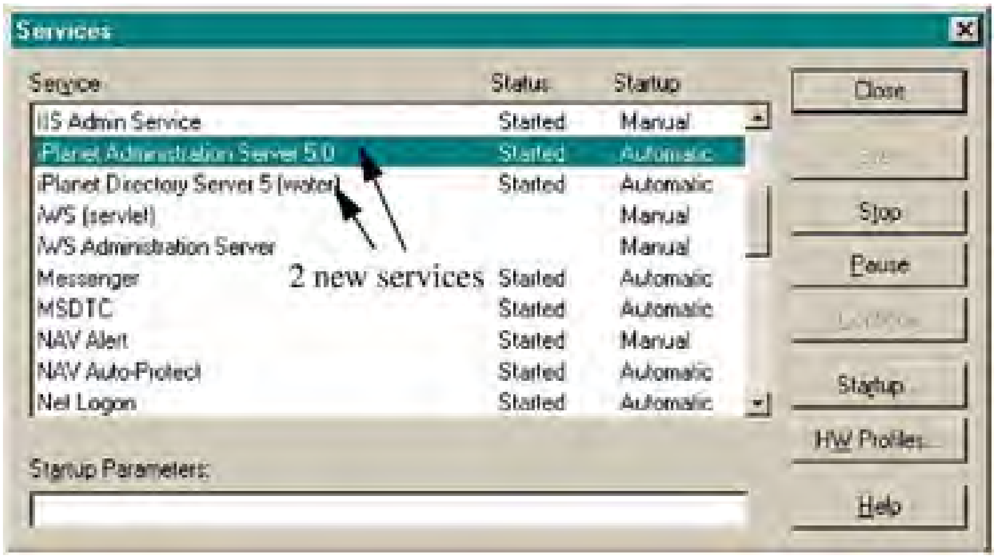
If you are using Windows NT, restart the computer (Figure 9.18); then test to ensure that the installation has been successful.

Figure 9.18. Installation complete.


Completion of Installation
Setup has finished copying files to your computer.
Before you can use the program, you must restart Windows or your computer.
Restart my computer now.
No, I will restart my computer later.
Remove any disks from their drives, and then click Finish to complete setup.
Finish
Cancel

After restarting the system, go to the Windows Services screen (Figure 9.19) to ensure the new services are listed and running. You should see two new services:

Figure 9.19. Services window.



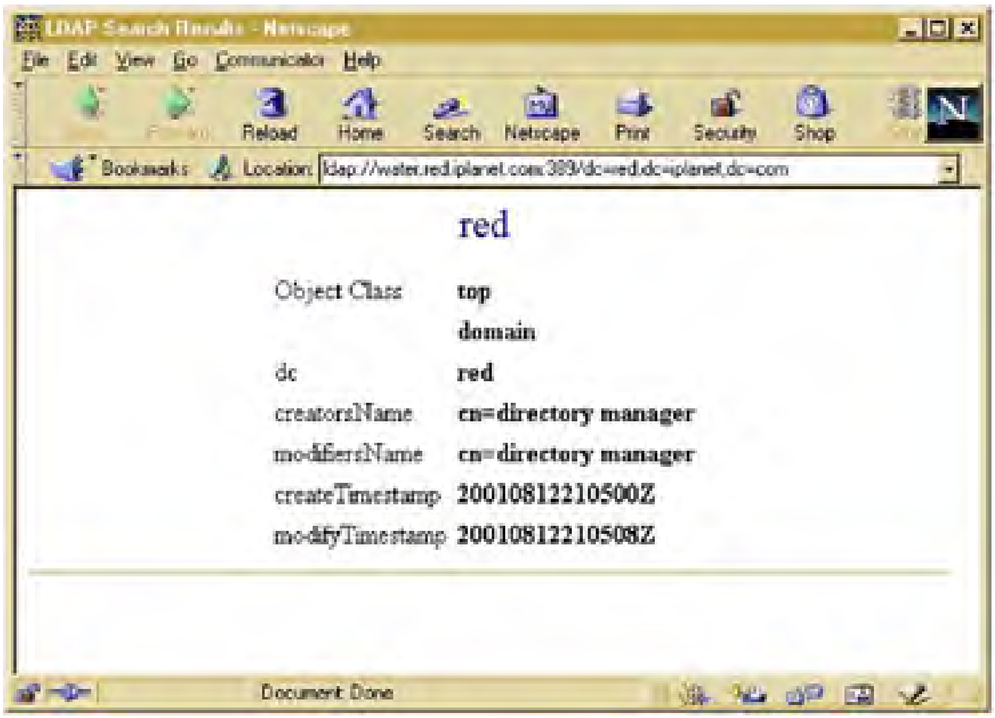
Service	Status	Startup
IIS Admin Service	Started	Manual
iPlanet Administration Server 5.0	Started	Automatic
iPlanet Directory Server 5 (water)	Started	Automatic
iWS (servlet)		Manual
iWS Administration Server		Manual
Messenger	Started	Automatic
MSDTC	Started	Automatic
NAV Alert	Started	Manual
NAV Auto-Protect	Started	Automatic
Net Logon	Started	Automatic

- iPlanet Administration Server 5.0
- The directory server instance iPlanet Directory Server 5 (water)

As an option, click twice on one of the services and set the startup type to *Manual*. This way, after rebooting the computer, the service does not automatically start up. *Manual* is most useful when testing servers.

Test your ability to connect to the directory server using the Netscape browser. The browser comes with a simplistic LDAP client. Go to URL *ldap://art.red.iplanet.com:389/dc=red,dc=iplanet,dc= com* (Figure 9.20).

Figure 9.20. LDAP search results.


LDAP Search Results - Netscape
File Edit View Go Communicator Help
Reload
Home
Search
Netscape
Print
Security
Shop
Bookmarks
Location: ldap://water.red.iplanet.com:389/dc=red,dc=iplanet,dc=com
Document: Done

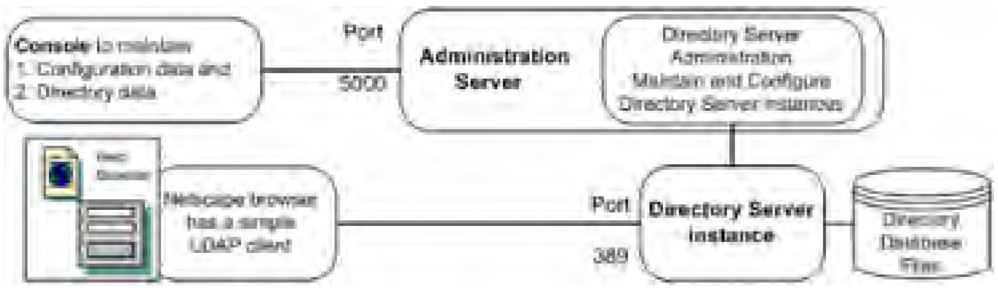
red

Object Class	top
	domain
dc	red
creatorsName	cn=directory manager
modifiersName	cn=directory manager
createTimestamp	20010812210500Z
modifyTimestamp	20010812210508Z

Starting and Stopping the Servers

The diagram in Figure 9.21 shows the two server processes that are stopped and started: the directory server administration server and the directory server instance. The administration server uses directory server instance data. Therefore, the directory server instance needs to be running before the administration server is started. Yes, order *does* matter.

Figure 9.21. Sun ONE Directory Server components.


Console to maintain
1. Configuration data and
2. Directory data
Port
5000
Administration Server
Directory Server Administration Maintain and Configure Directory Server instances
Netscape browser has a simple LDAP client
Port
389
Directory Server instance
Directory Database Files

On UNIX

To start the directory server instance, use the command

```
cd /iplanet/ds50/slapd-art
./start-slapd
```

or use the full path command

```
/iplanet/ds50/slapd-art/start-slapd
```

View the directory server instance process:

```
ps -ef | grep ds50
nobody 936 1 3 08:58:00 ? 0:01 ./ns-slapd -D /
  iplanet/ids/slapd-art
-i /iplanet/ids/slapd-art/logs/pid
```

To stop the directory server instance:

```
cd /iplanet/ds50/slapd-art./stop-slapd
```

or use the full path command:

```
/iplanet/ds50/slapd-art/stop-slapd
```

To start the directory administration server:

1. Start the directory server instance:

   ```
   /iplanet/ds50/slapd-art/start-slapd
   ```

2. Start the administration server:

   ```
   cd /iplanet/ds50
   ./start-admin
   ```

   or use the full path command:

   ```
   /iplanet/ds50/start-admin
   ```

View the directory server instance processes:

```
ps -ef | grep ds5
nobody 936 1 3 08:58:00 ? 0:01 ./ns-slapd -D /
  iplanet/ids/slapd-art
-i /iplanet/ids/slapd-art/logs/pid
root 944   943  2 09:01:51 ? 0:01 ns-httpd -d /
  iplanet/ids/admin-serv/config
root 943     1  0 09:01:51 ? 0:00 ./uxwdog -d /
  iplanet/ids/admin-serv/config
```

The processes are as follows:

- Process 936 is the directory server instance.
- Process 944 is the directory server administration server.
- Process 943 is the watchdog process that monitors process 944.

To stop the directory administration server:

```
cd /iplanet/ds50
./stop-admin
```

or use the full path command:

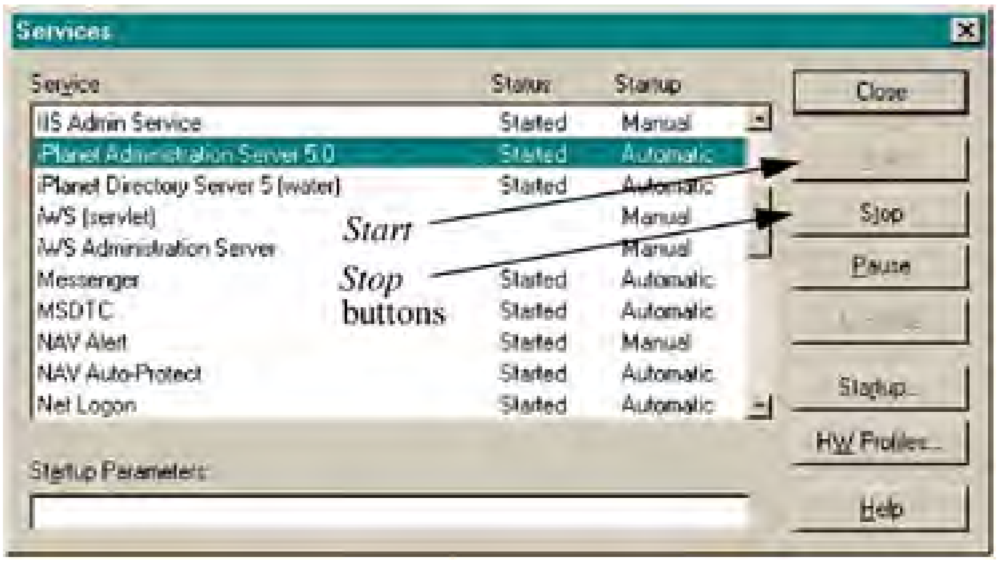
```
/iplanet/ds50/stop-admin
```

On Windows

To stop and start the Sun ONE Directory Server processes, use the *Services* window from the *Control Panel*: *Start* button / *Settings / Control Panel*. Then, when the *Control Panel* comes up,

- On Windows NT, click *Services* (Figure 9.22).

Figure 9.22. Services window.



Service	Status	Startup
IIS Admin Service	Started	Manual
iPlanet Administration Server 5.0	Started	Automatic
iPlanet Directory Server 5 (water)	Started	Automatic
iWS (servlet)		Manual
iWS Administration Server		Manual
Messenger	Started	Automatic
MSDTC	Started	Automatic
NAV Alert	Started	Manual
NAV Auto-Protect	Started	Automatic
Net Logon	Started	Automatic

- On Windows 2000, click *Administration Tools*; then click *Services*.

From the *Services* window, use the *Start* and *Stop* buttons to manage the processes.

[Team LiB]
PREVIOUS NEXT

Using the Console

The Console is used to administer the directory server instances, maintain directory data, and modify the directory schema.

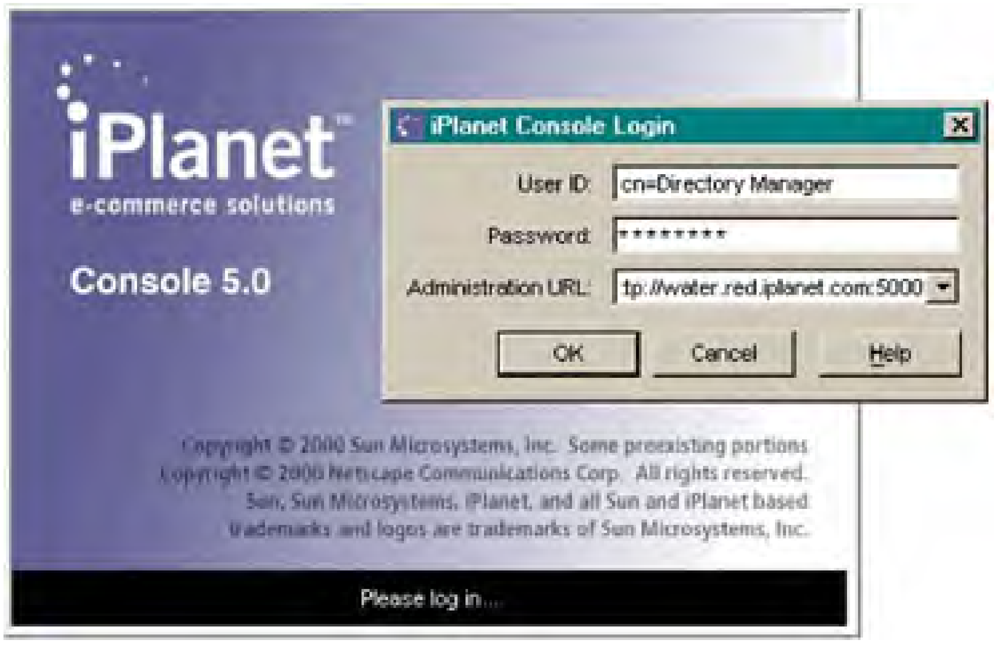
To use the Console:

1. Since the Console connects to the administration server (refer to Figure 9.21), start the directory instance before starting the administration server.
2. Start the Console program:
   - From Solaris use the command

     ```
     cd /iplanet/ds50
       ./startconsole &
     ```

   - From Windows, click the *Start* button /Programs/iPlanet Server Products/iPlanet Console 5.0.
3. Log in to the Administration Server, on port 5000, using the directory server manager account (Figure 9.23) using

Figure 9.23. iPlanet Console login.



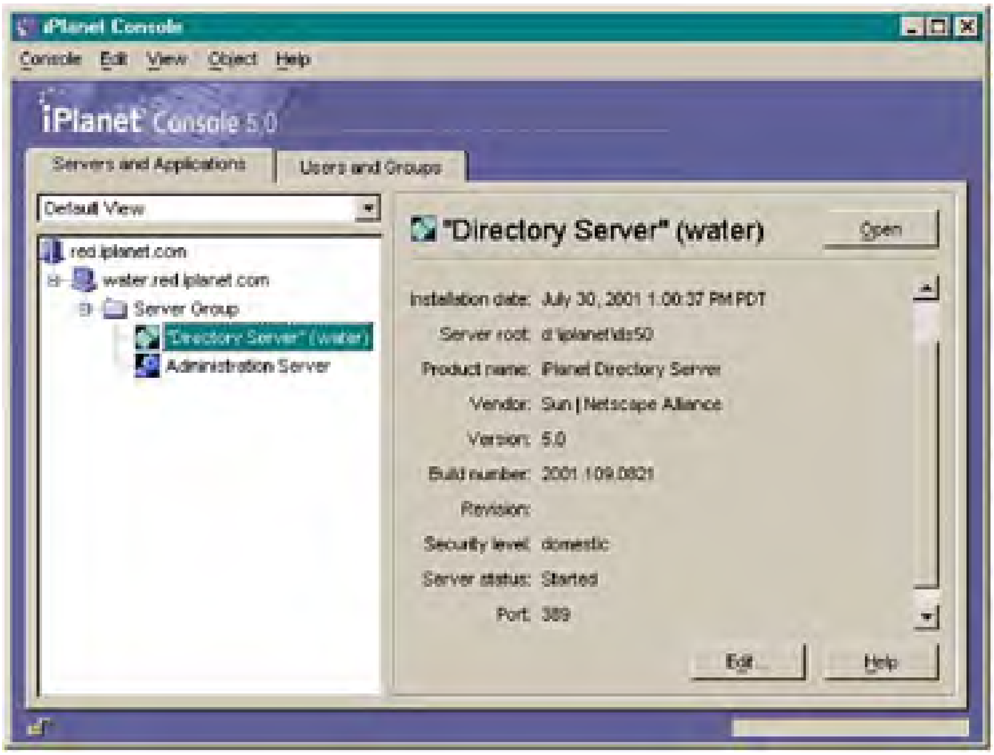
User ID: cn=Directory Manager
Password: ********
Administration URL: tp://water.red.iplanet.com:5000

Copyright © 2000 Sun Microsystems, Inc. Some preexisting portions Copyright © 2000 Netscape Communications Corp. All rights reserved. Sun, Sun Microsystems, iPlanet, and all Sun and iPlanet based trademarks and logos are trademarks of Sun Microsystems, Inc.

   - User ID: *cn=Directory Server*
   - Password: *netscape*
   - URL: *http://<hostname>:<port>*
   - Example URL: *http://art.red.iplanet.com:5000*

Once logged in to the Console, open the tree and you will see what you have installed (Figure 9.24).

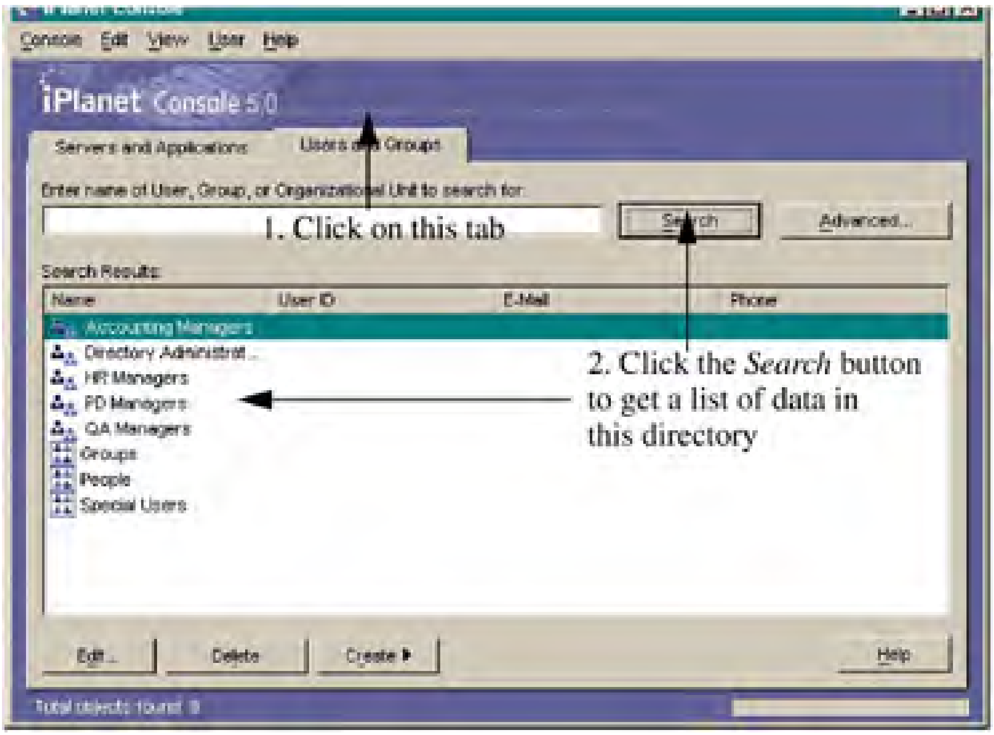
Figure 9.24. Servers and applications.



Installation date: July 30, 2001 1:00:37 PM PDT
Server root: d:\iplanet\ds50
Product name: iPlanet Directory Server
Vendor: Sun | Netscape Alliance
Version: 5.0
Build number: 2001.109.0821
Revision:
Security level: domestic
Server status: Started
Port: 389

To view the data that was added by default during the installation, click the *Users and Groups* tab and then click the *Search* button (Figure 9.25).

Figure 9.25. Users and groups.



You now have a fully functional Sun ONE Directory Server installed! You know how to start and stop the servers. And you know how to start the Console and log in to the administration server. The next chapter shows how to work with the directory data.

Uninstalling the Server

It is very easy to uninstall the server—run the uninstall option from the Windows *Control Panel Add/Remove* option. After the uninstall has been run, remove the installation directory. At this point, the Sun ONE Directory Server has been completely removed/uninstalled.

Reference Documentation

The Sun Microsystems web site at *http://docs.sun.com* has a considerable amount of documentation. The link for the Sun ONE Directory Server is *http://docs.sun.com/db/prod/sunone*.

Chapter 10. Using the Sun ONE Directory Server Database: The Console

This chapter focuses on using the Console to administer the Sun ONE Directory Server. The Console is used to

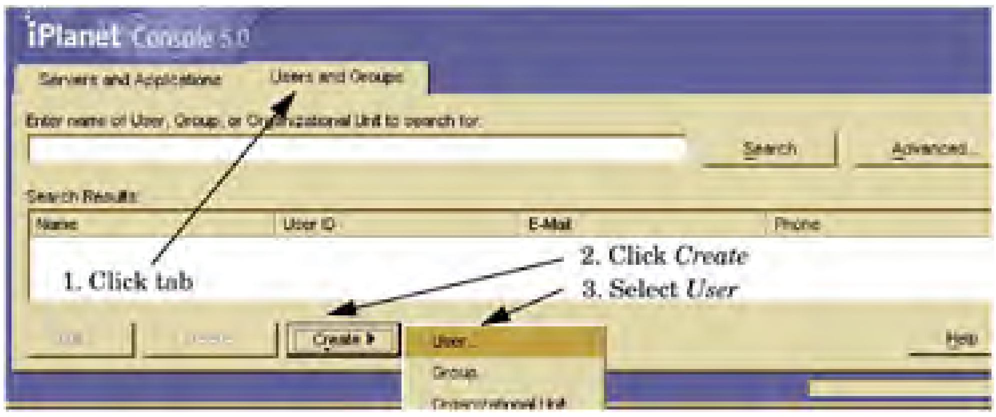
- Maintain passwords for administration accounts
- Modify data in the directory (add, change, copy, and delete)
- Add a branch of data into the directory (add an organization unit and data)
- Import and export data (save/export data into an LDAP Data Interchange Format [LDIF] file, delete the data from the directory, and restore/import the data using the LDIF file)

Adding User Data into the Directory

The Sun ONE Directory Server comes with a base schema. The base schema includes predefined database structures of objects and attributes. The objects and attributes are used to organize the data. The schema has defined objects and attributes to store data about users. The object called People stores a person's user ID, name, address, and other information.

To add users, start the Console program and log in using *cn=Directory Manager* (see previous chapter for instructions). From the main administration window, click the *Users and Groups* tab and the window should look like Figure 10.1.

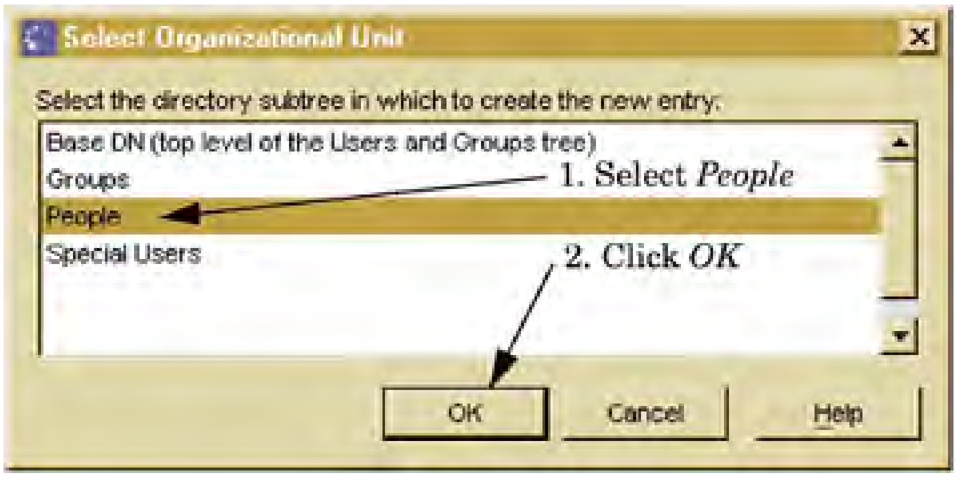
Figure 10.1. Search and maintain users and groups.



Add two users following the instructions in Figure 10.1.

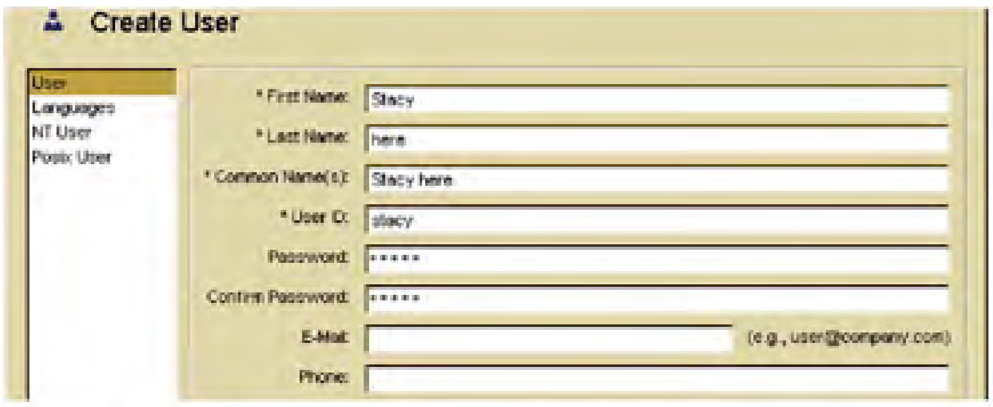
In Figure 10.2, from *Select Organizational Unit* window, select *People*. The user will be added under the node (organization unit [ou]) called *People*.

Figure 10.2. Create a user under the organization unit People.


Select Organizational Unit
Select the directory subtree in which to create the new entry:
Base DN (top level of the Users and Groups tree)
Groups
People
Special Users
1. Select *People*
2. Click *OK*
OK
Cancel
Help

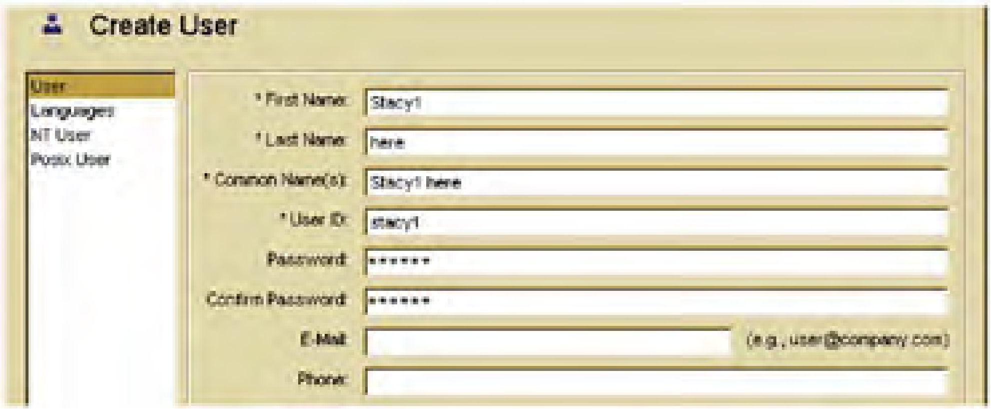
Add two users (Figure 10.3). Enter mandatory data into the fields with an asterisk (*) and click *OK*. Do this for both users. I'm using my first name to create the users *stacy* and *stacy1*.

Figure 10.3. Add mandatory data for two users.



Create User

* First Name:	Stacy
* Last Name:	here
* Common Name(s):	Stacy here
* User ID:	stacy
Password:	*****
Confirm Password:	*****
E-Mail:	(e.g., user@company.com)
Phone:	

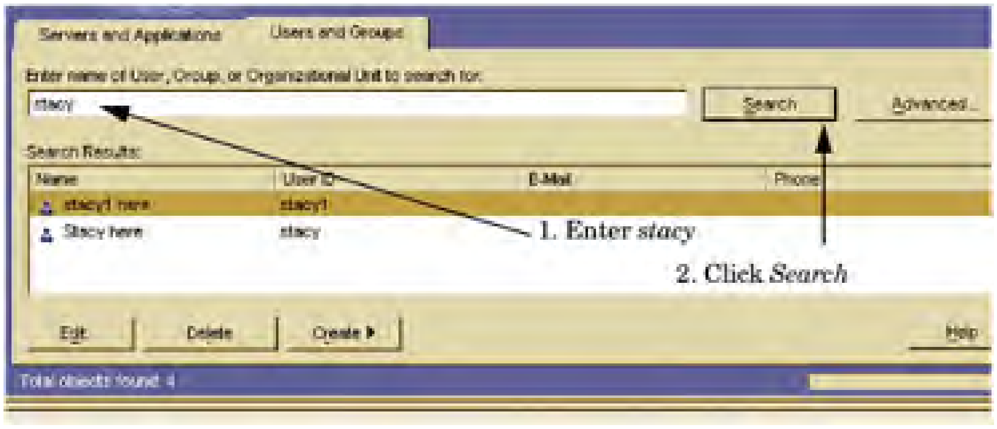


Create User

* First Name:	Stacy1
* Last Name:	here
* Common Name(s):	Stacy1 here
* User ID:	stacy1
Password:	******
Confirm Password:	******
E-Mail:	(e.g., user@company.com)
Phone:	

After the users are added, use the *Search* button to list the new users (Figure 10.4). You can also search using the Netscape web browser.

Figure 10.4. Search for new users.



Enter name of User, Group, or Organizational Unit to search for:

stacy

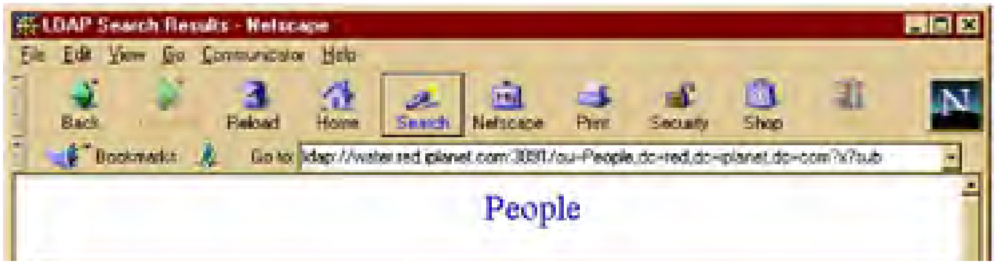
Search Results:

Name	User ID	E-Mail	Phone
stacy1 here	stacy1		
Stacy here	stacy		

Total objects found: 4

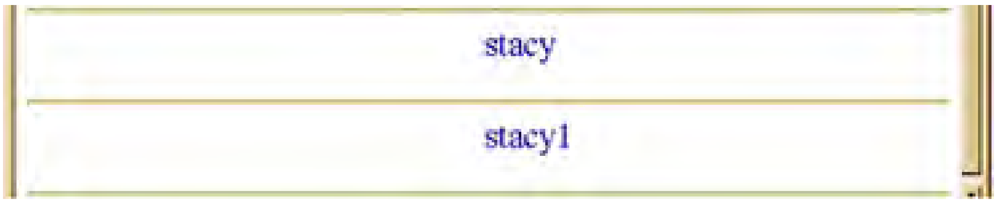
The Netscape web browser has the ability to make LDAP calls. Use this URL (Figure 10.5):

Figure 10.5. Browser directory search results.



Go to: ldap://water.red.iplanet.com:3881/ou=People,dc=red,dc=iplanet,dc=com??sub

People


stacy

stacy1

ldap://<hostname>:<port>/<ou values>,<root-suffix>?x?sub

- *<ou values>* is the organizational unit attribute pairs.
- *<root-suffix>* is based on the installation. Mine is (and yours should be) based on my domain name. For example:

  Domain red.iplanet.com

becomes suffix dc=red,dc=iplanet,dc=com

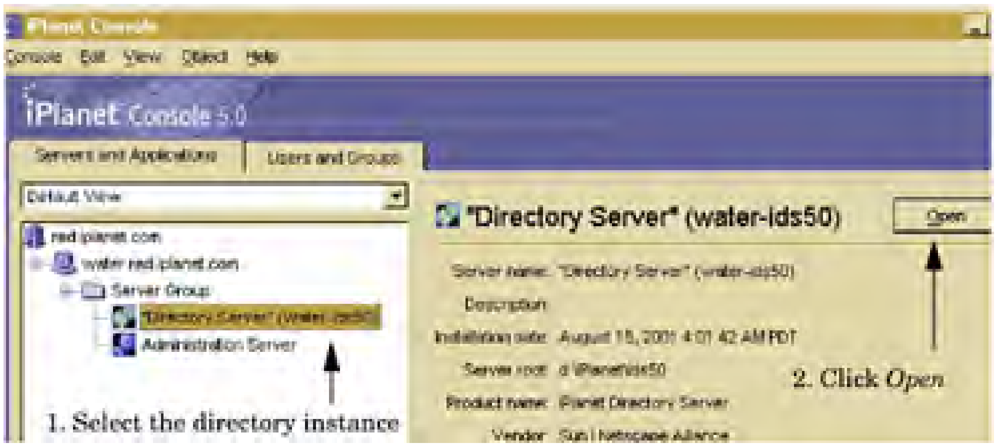
?x?sub is the search string. It means: Create a list using the main object attribute. In this case the attribute is uid.

Data has been added into the directory. Now open up another tool to work with the directory schema and administration data.

Administering the Directory Server Instance

This part of the Console is the main GUI tool for working on a directory instance schema and with the directory instance data. To start this tool, from the *Servers and Applications* tab, select the directory instance *"Directory Server"* and open the directory server administration application (Figure 10.6).

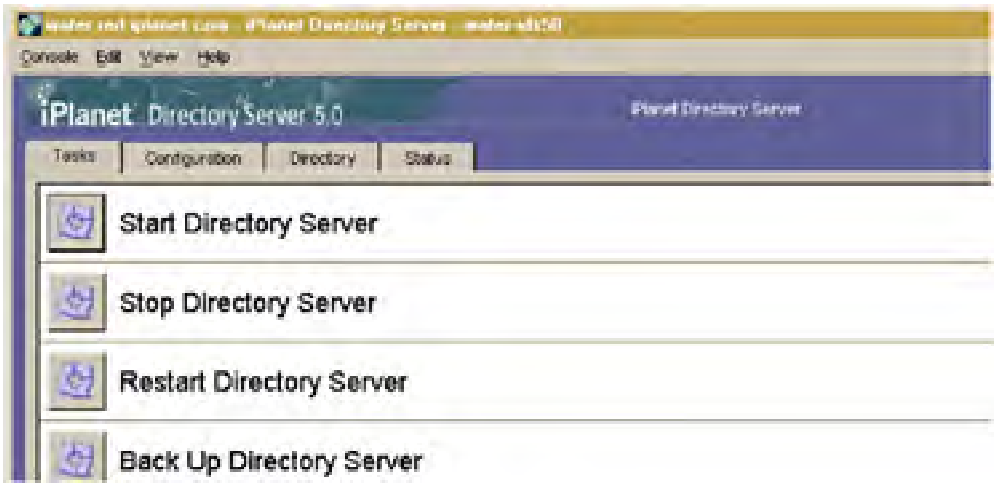
Figure 10.6. Open the "Directory Server."



Server Name: "Directory Server" (water-ids50)
Description:
Installation date: August 15, 2001 4:01:42 AM PDT
Server root: d:\iPlanet\ids50
Product name: iPlanet Directory Server
Vendor: Sun | Netscape Alliance

The first thing you are presented with is the task list as shown in Figure 10.7. For example, from here the directory instance you are working on may be started and stopped.

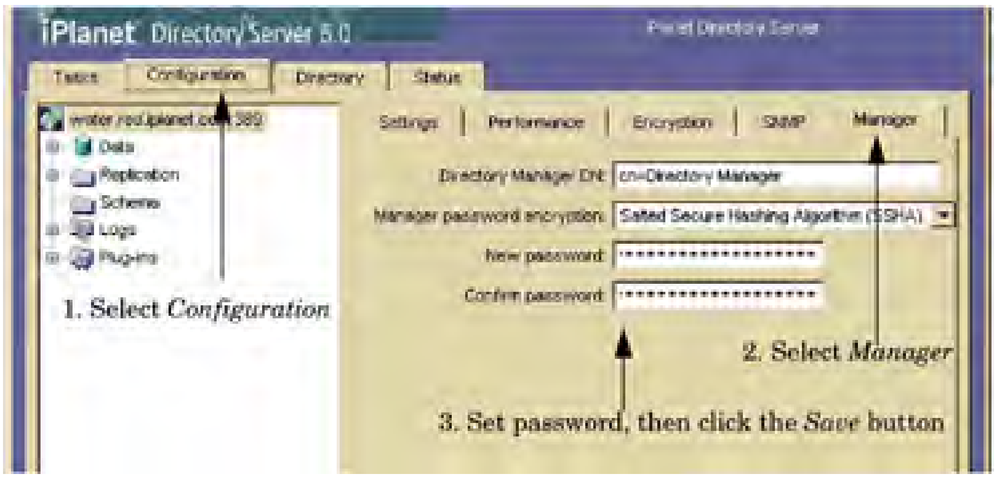
Figure 10.7. Directory instance task list.



Administration Accounts

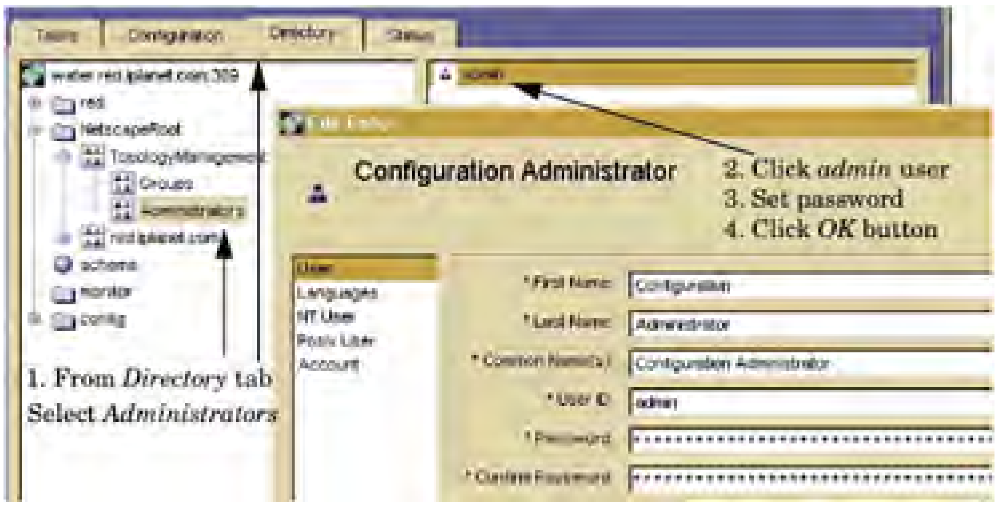
To maintain the password for the directory administrator account *cn=Directory manager*, follow the instructions in Figure 10.8.

Figure 10.8. Directory administrator account password.


iPlanet Directory Server 5.0
Tasks
Configuration
Directory
Status
water.red.iplanet.com:389
Data
Replication
Schema
Logs
Plug-ins
Settings
Performance
Encryption
SNMP
Manager
Directory Manager DN: cn=Directory Manager
Manager password encryption: Salted Secure Hashing Algorithm (SSHA)
New password:
Confirm password:
1. Select Configuration
2. Select Manager
3. Set password, then click the Save button

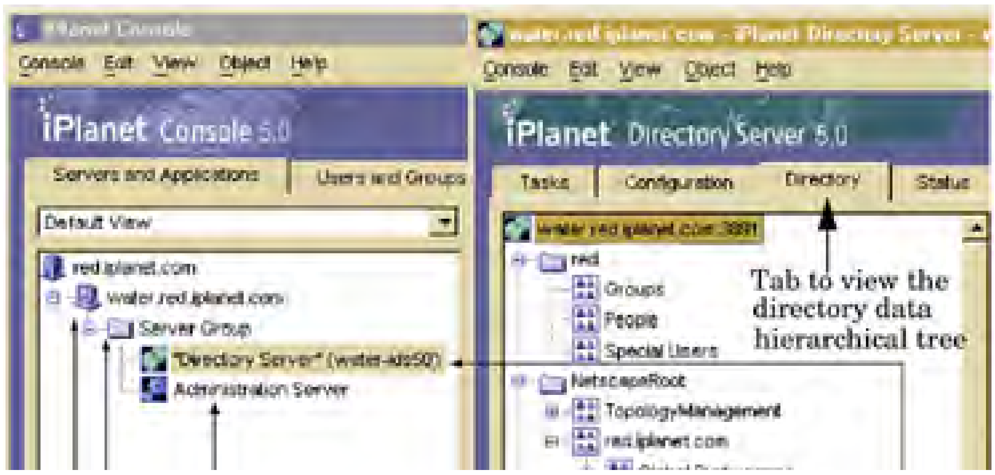
Select the *Directory* tab to maintain the administrator *admin* account password (Figure 10.9).

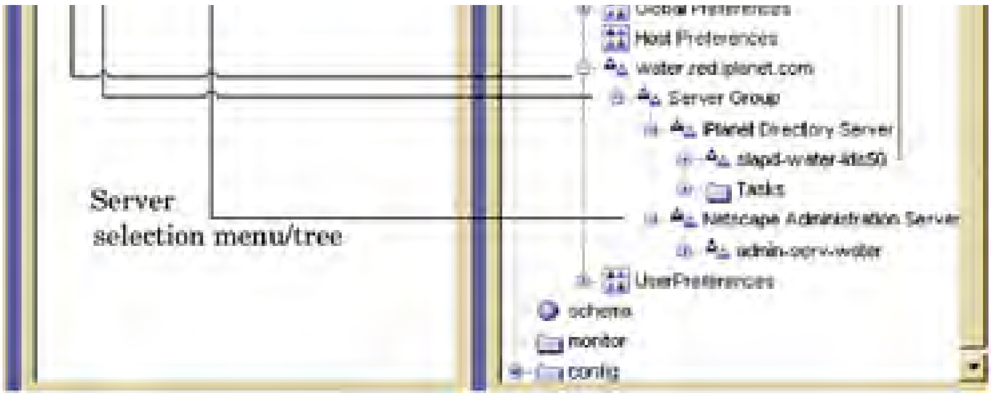
Figure 10.9. Administrator admin account password.



In Figure 10.9, note that, in the hierarchy, *NetscapeRoot* is below the node *water.red.iplanet.com:389*. *NetscapeRoot* is the data branch containing information about this directory server installation. Figure 10.10 shows the data mapping from the directory to the menu in the main Console window. The data that is used to create the menu is stored in the directory. In the case of Figure 10.10, the directory data mapping is between the iPlanet Admin Server screen on the left and the directory data under *NetscapeRoot* on the right.

Figure 10.10. Data mapping from directory to menu in main Console window.




Global Preferences
Host Preferences
water.red.iplanet.com
Server Group
iPlanet Directory Server
slapd-water-ldap
Tasks
Netscape Administration Server
admin-serv-water
UserPreferences
schema
monitor
config
Server selection menu/tree

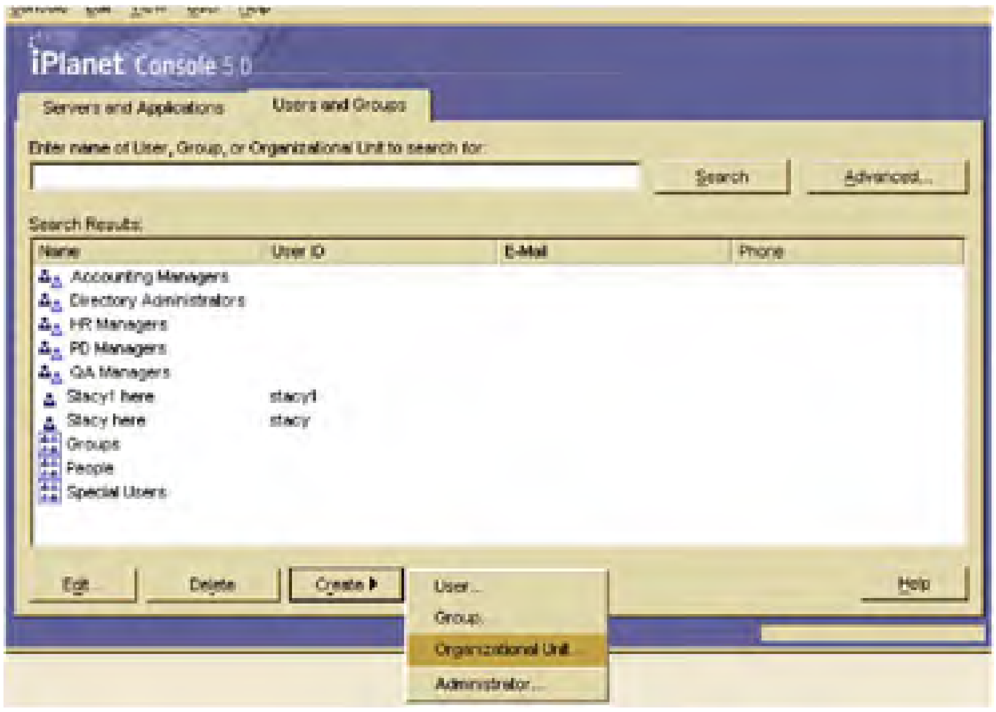
Adding an Organization Unit and Adding Data

An organization unit (*ou*) is a node in the directory. Objects are added under an *ou*. The *ou* organizes data. In this section, an *ou* named *InternetTree* is added, and data is put under it.

Creating an Organizational Unit

Go to the main Console window and click the *Users and Groups* tab. Click the *Create* button and select *Organizational Unit* (Figure 10.11).

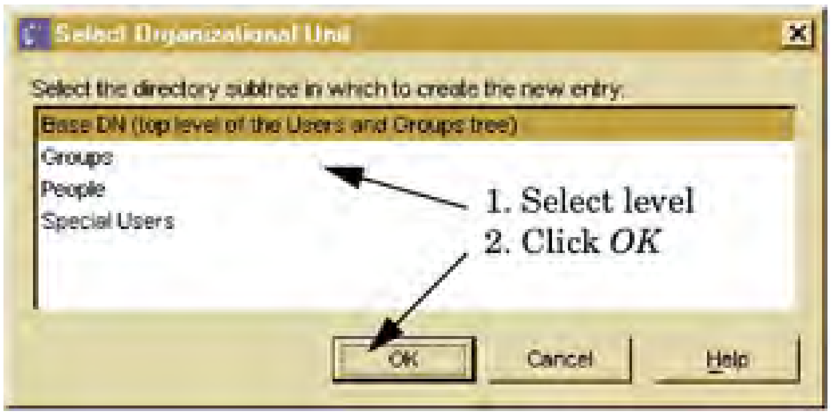
Figure 10.11. Create an Organizational Unit.



Name	User ID	E-Mail	Phone
Accounting Managers			
Directory Administrators			
HR Managers			
PD Managers			
QA Managers			
Stacy1 here	stacy1		
Stacy here	stacy		
Groups			
People			
Special Users			

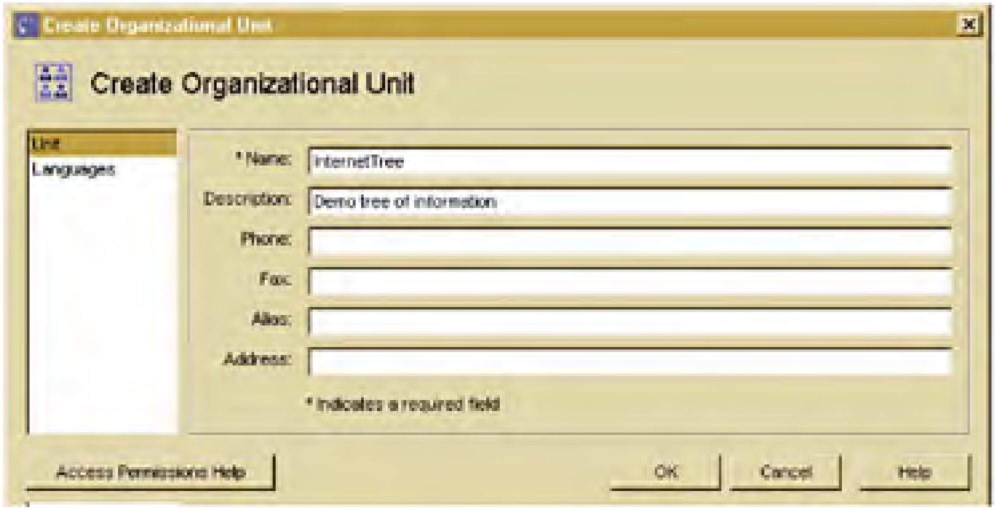
Create *InternetTree* at the top level by selecting the first item in the directory subtree list shown in Figure 10.12.

Figure 10.12. Select directory subtree for InternetTree.


Select Organizational Unit
Select the directory subtree in which to create the new entry:
Base DN (top level of the Users and Groups tree)
Groups
People
Special Users
1. Select level
2. Click OK
OK
Cancel
Help

Enter the *ou* name *InternetTree* in the *Name* box and a brief description in the *Description* box (Figure 10.13). Then click *OK*.

Figure 10.13. Name and describe InternetTree.



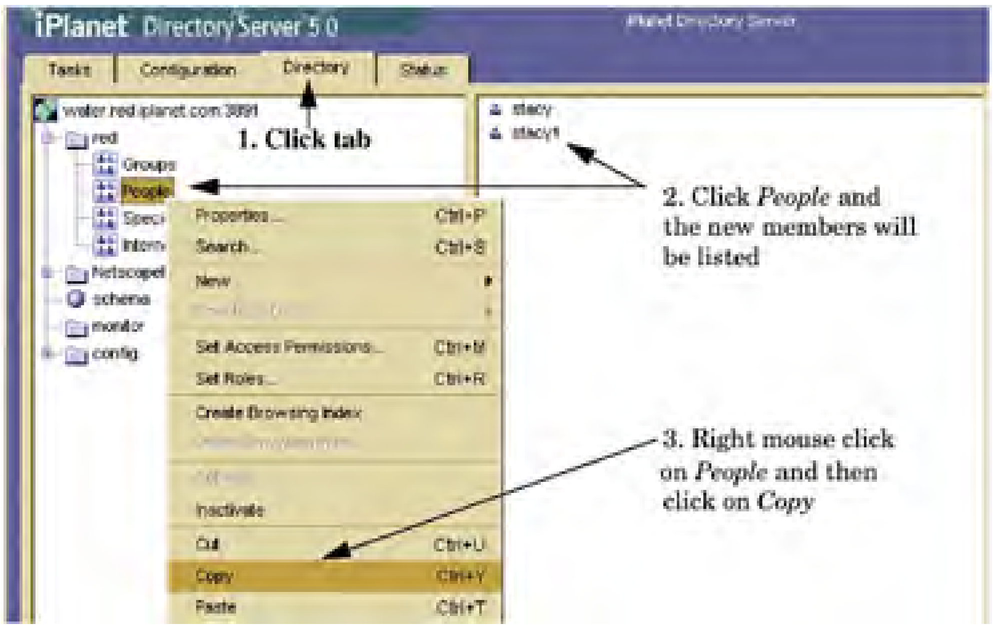
Create Organizational Unit

* Name:	InternetTree
Description:	Demo tree of information
Phone:	
Fax:	
Alias:	
Address:	

* Indicates a required field

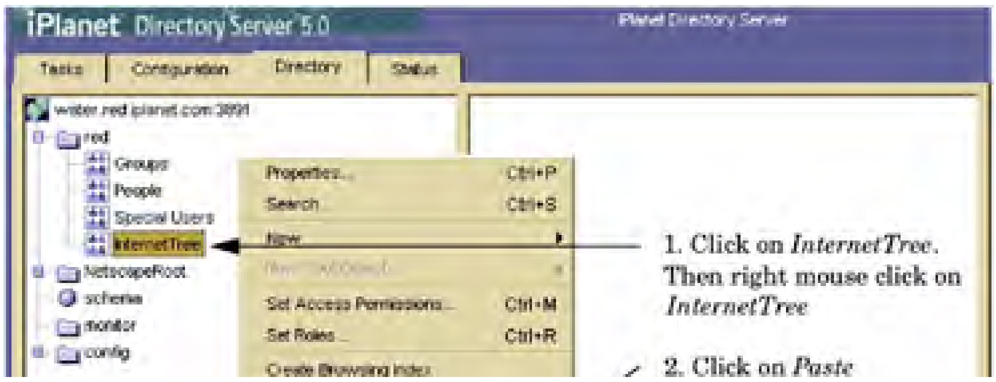
Go to the Administration application window to copy data under *InternetTree* as illustrated in Figure 10.14.

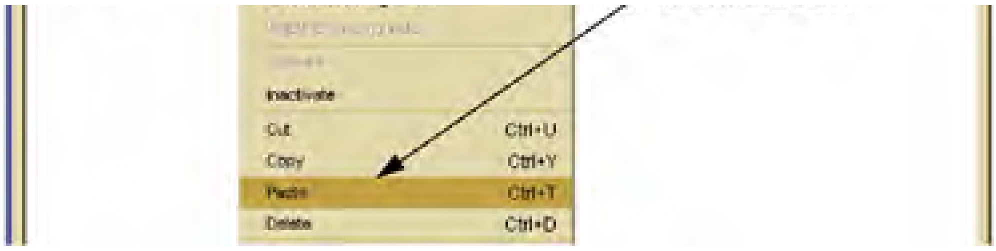
Figure 10.14. Copy data.



Now paste the data under the new organizational unit *InternetTree,* as illustrated in Figure 10.15.

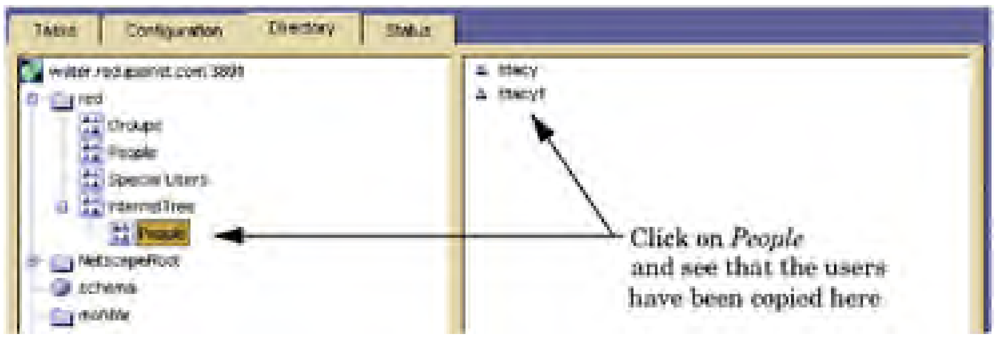
Figure 10.15. Paste data.





View the data under *InternetTree* (Figure 10.16).

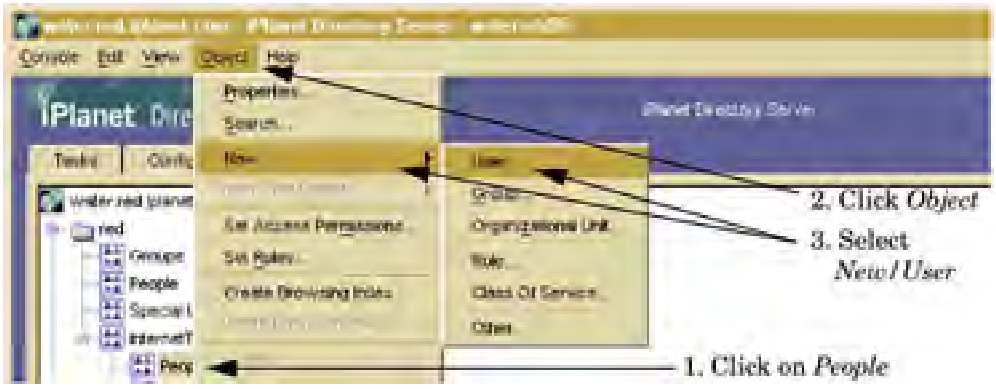
Figure 10.16. View data under InternetTree.



Using *Copy* and *Paste* is a convenient method when testing and working with the directory database.

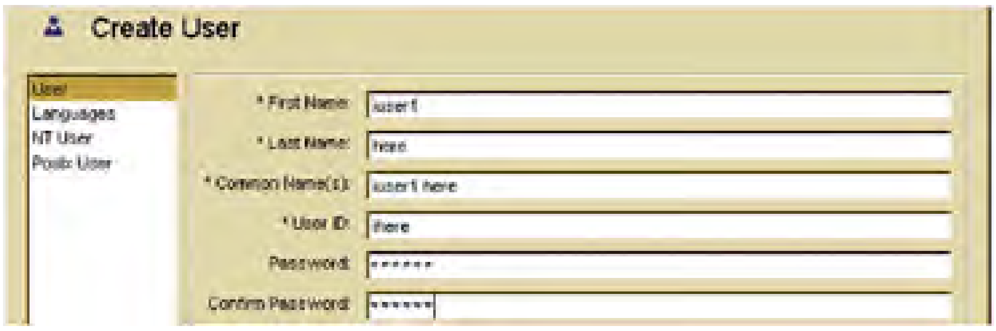
Insert or add users under the *People* node that is under the *InternetTree* node as in Figure 10.17.

Figure 10.17. Add users into the new People branch.



Add three new iusers: *iuser1*, *iuser2*, *iuser3* (Figure 10.18).

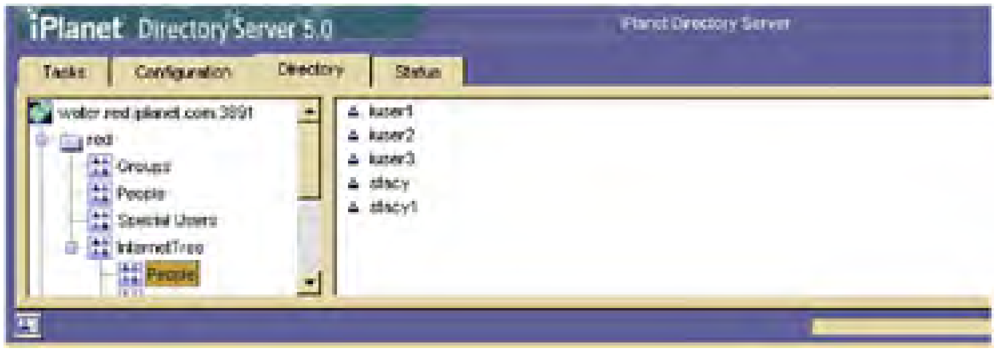
Figure 10.18. Enter new users.



* First Name:	iuser1
* Last Name:	here
* Common Name(s):	iuser1 here
* User ID:	ihere
Password:	******
Confirm Password:	******

Now there are five users under *People, InternetTree* (see Figure 10.19).

Figure 10.19. View five new users.

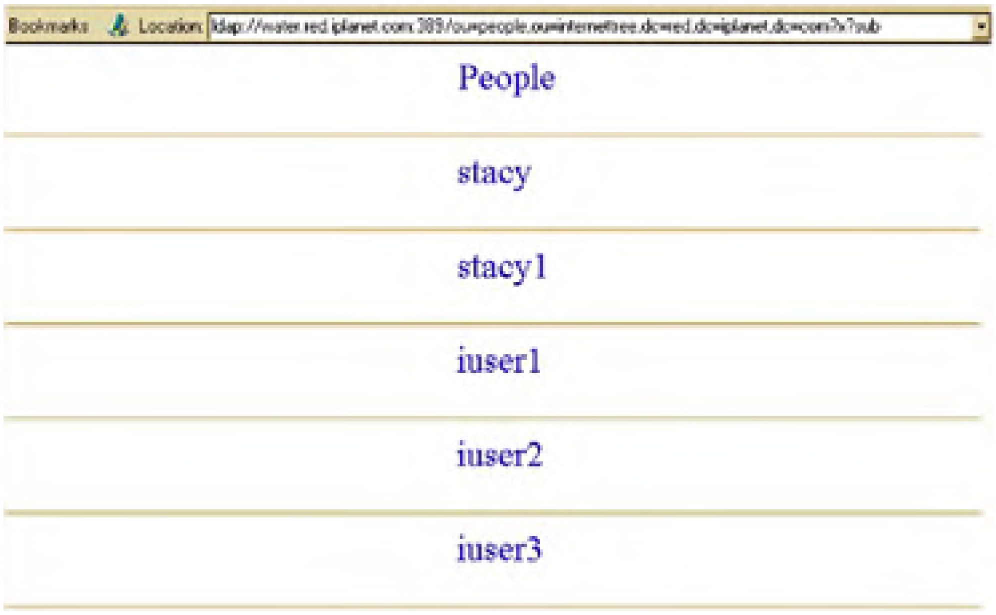


Use the Netscape browser and LDAP URL syntax

ldap://<host>:<port>/ou=people,ou=internettree,<domain suffix>?x?sub

to list the new user Go to the URL (Figure 10.20). In this example, use the following URL:

Figure 10.20. Listing of People users under InternetTree.


Bookmarks Location: ldap://water.red.iplanet.com:389/ou=people,ou=internettree,dc=red,dc=iplanet,dc=com?x?sub

People

stacy

stacy1

iuser1

iuser2

iuser3

ldap://water.red.iplanet.com:389/ou=people,ou=internettree,dc= red,dc=iplanet,dc=com?x?sub

The LDAP URL Syntax

The Netscape browser is a convenient tool for LDAP searching and the syntax that must be used is

ldap://<host>:<port>/<node>?<attributes>?<base|one|sub>?<filter>

The following explains the syntax parts.

- *<node>* is the branch from where to start searching
- *<attributes>* are the attributes to list
- *<base>* is the base node value, or
- *<one>* means search only to the depth of *one* level under the *<node>*, or
- *<sub>* means search all sublevels, from (including) current *<node>* down

Use your browser to query the directory database using the following examples. List only *iusers* under the *InternetTree/People* node.

ldap://water:3891/ou=People,ou=InternetTree,dc=red,dc=iplanet, dc=com?x?one?(uid=iuser)*

- *ou=People,ou=InternetTree,dc=red,dc=iplanet,dc=com* is the node or branch point from which the search is to start.
- *?x?one?(uid=iuser*)* is the search string.
- *?x* is a dummy value. This will cause only the user ID to be listed.
- *(uid=iuser*)* is the filter, where the user ID starts with *iuser*.

List the *givenname* and *cn* (Full/Common name) for users under the *InternetTree* node:

ldap://water:3891/ou=InternetTree,dc=red,dc=iplanet,dc=com?givenname,cn?sub?(uid=iuser)*

Note: This query lists the same users as the above list.

So far, the queries have used a simple single qualifier. There are qualifiers that can be used with queries that involve multiple filters. Multiple qualifiers are joined using "and" and "or." Here is the multiple filter qualifier syntax:

- (<logical>(<attribute filter>)(<attribute filter>)).
- Logicals are & (and) or | (or).
- An <attribute filter> is <attribute><comparison value><compare value>.
- Comparison values are = | > | < | >= | <=.

Comparison values can use strings or * for a wild card value.

Or multiple filter qualifier syntax for *not* is

(!(<attribute filter>))

Example filters include:

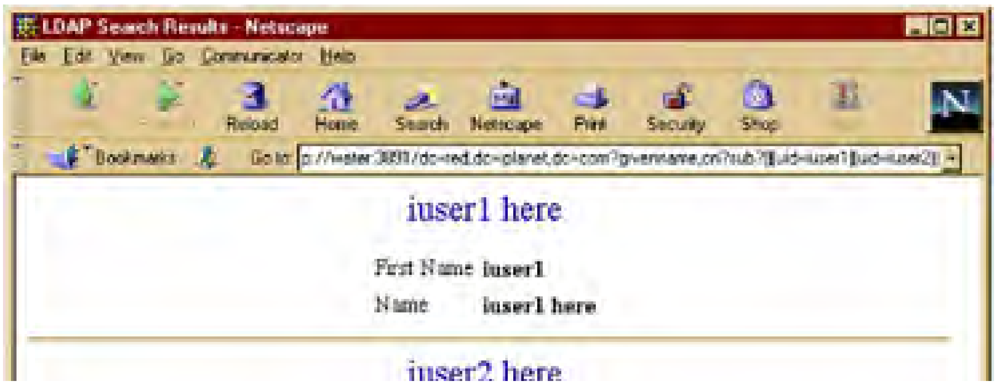
- (uid=iuser*) is a user ID starting with *iuser.*
- (!(uid=iuser*)) represents users whose names do not start with *iuser*.
- (|(uid=iuser1)(uid=iuser2)) are users that have a user ID of *iuser1* or *iuser2*.

An example query might be

ldap://water:3891/dc=red,dc=iplanet,dc=com?given-
name,cn?sub?(|(uid=iuser1)(uid=iuser2))

The result is in Figure 10.21.

Figure 10.21. LDAP query results.

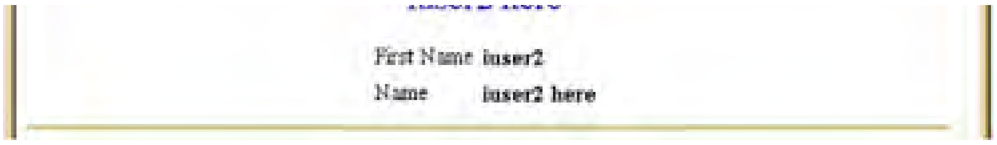

LDAP Search Results - Netscape
File Edit View Go Communicator Help
Reload
Home
Search
Netscape
Print
Security
Shop
Bookmarks
Go to:

p://water:3891/dc=red,dc=iplanet,dc=com?givenname,cn?sub?(|(uid=iuser1)(uid=iuser2))

iuser1 here

First Name iuser1
Name iuser1 here

iuser2 here



First Name	**iuser2**
Name	**iuser2 here**

The best way to learn how searching works is to add data into the directory and try a lot of searches. Try, try, try, and try some more.

Now that you are able to create directory information, you'll need to know about saving and restoring data from the directory.

LDIF Save and Restore

Sometimes the best way to learn is by doing. This section walks through a sample to save, delete, and restore users under the *InternetTree* node. There are three major steps:

1. Export the *InternetTree* data into an LDIF file.
2. Delete the data under *InternetTree.*
3. Restore the data by importing the LDIF file.

Step 1. Export the InternetTree Data into an LDIF File

Continuing with the Console, click the *Tasks* tab as in Figure 10.22. When the *Export Databases* screen displays, click the *Browse* button and the screen in Figure 10.23 appears. Select a directory to put the LDIF file into. As Figure 10.23 shows, the default directory for LDIF files is the directory instance file directory. Name the file, noting that the file extension must be.ldif. Then click *Open*.

Figure 10.22. Directory server instance tasks.

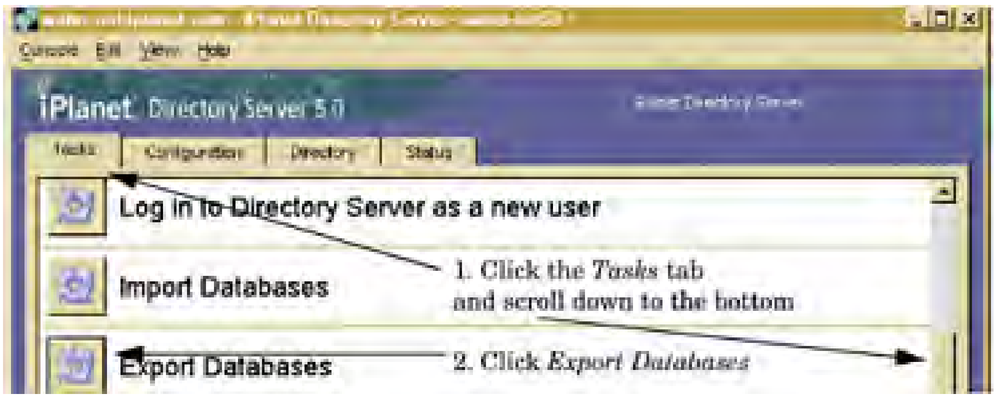
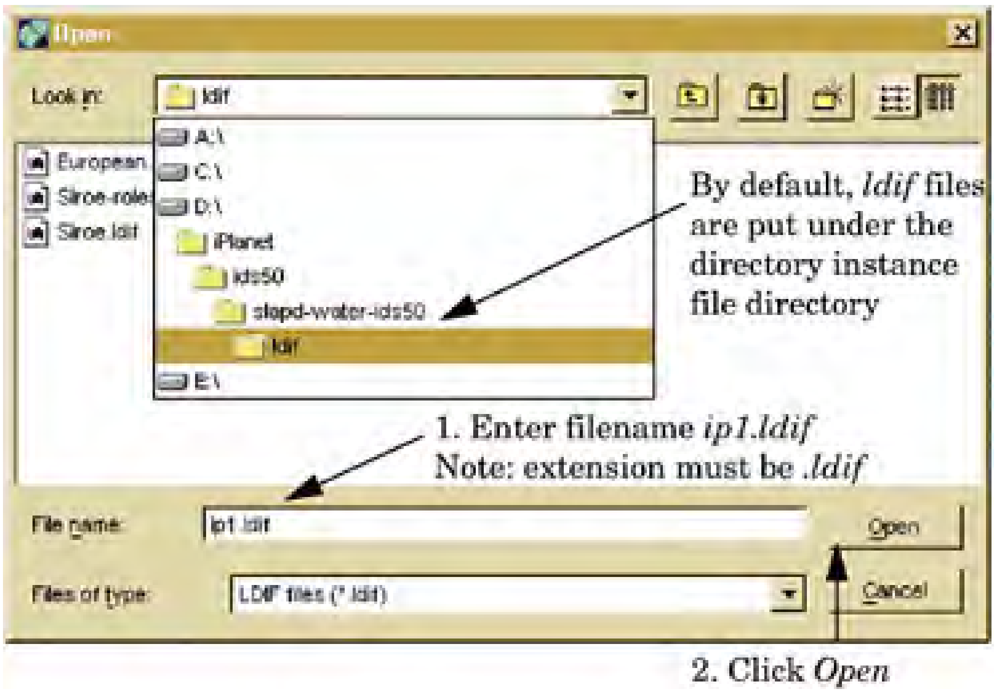
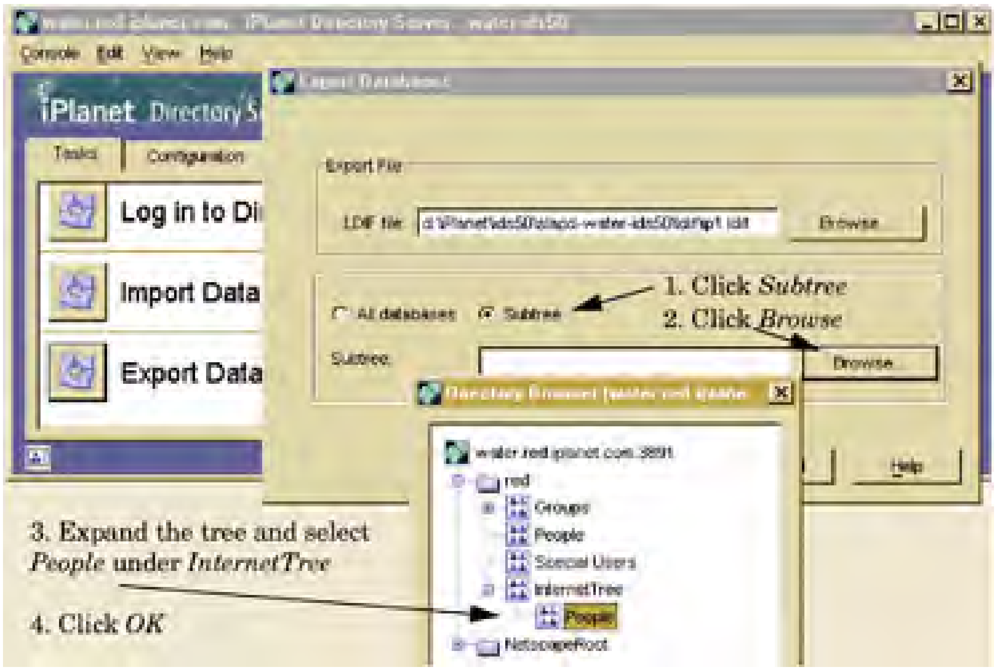

Console Edit View Help
iPlanet Directory Server 5.0
Tasks
Configuration
Directory
Status
Log in to Directory Server as a new user
Import Databases
Export Databases
1. Click the *Tasks* tab
and scroll down to the bottom
2. Click *Export Databases*

Figure 10.23. Select a directory and give the LDIF file a name.



Select the *InternetTree/People* branch of data to be backed up to an LDIF file (Figure 10.24).

Figure 10.24. Select branch of data to be backed up.



Export File

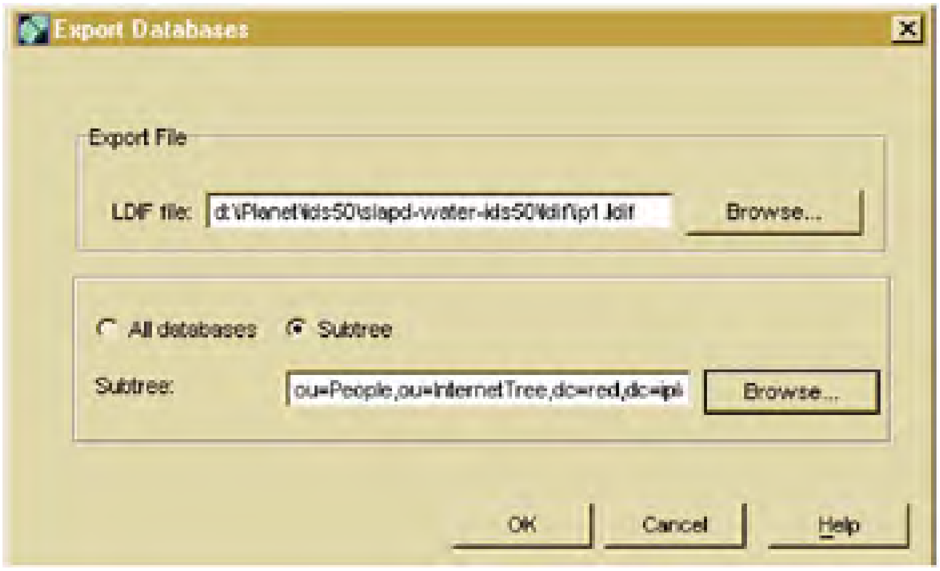
LDIF file: d:\iPlanet\ids50\slapd-water-ids50\ldif\ip1.ldif

( ) All databases (•) Subtree

1. Click *Subtree*
2. Click *Browse*
3. Expand the tree and select *People* under *InternetTree*
4. Click *OK*

Now the *Export Databases* option has a filename to export to and a directory branch to export (Figure 10.25). Click the *OK* button.

Figure 10.25. Export data into an LDIF file



Export Databases

Export File

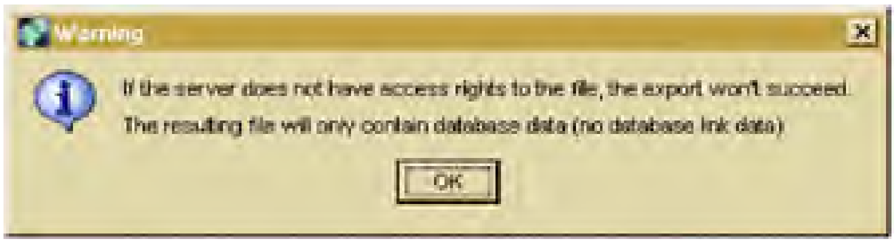
LDIF file: d:\iPlanet\ids50\slapd-water-ids50\ldif\ip1.ldif

( ) All databases (•) Subtree

Subtree: ou=People,ou=InternetTree,dc=red,dc=ipl

On a UNIX system, if the message in Figure 10.26 comes up, click *OK*.

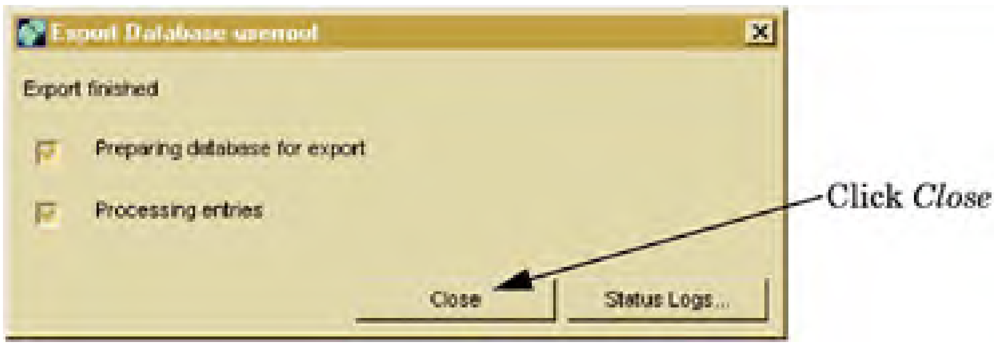
Figure 10.26. Warning on a UNIX system.



If the server does not have access rights to the file, the export won't succeed.
The resulting file will only contain database data (no database link data)

Data is exported; click the *Close* button (Figure 10.27).

Figure 10.27. Export completed.



Export finished

Preparing database for export

Processing entries

Here is a look at the first two entries in my LDIF file:

1. The *ou=People,ou=InternetTree* branch:

```
dn: ou=People,ou=InternetTree,dc=red,dc=iplanet,dc=com
objectClass: top
objectClass: organizationalunit
ou: People
creatorsName: cn=directory manager
modifiersName: cn=directory manager
createTimestamp: 20010816112810Z
modifyTimestamp: 20010816112810Z
nsUniqueId: a6692f01-1dd111b2-800cd250-749d06e5
```

2. The user *uid=stacy*

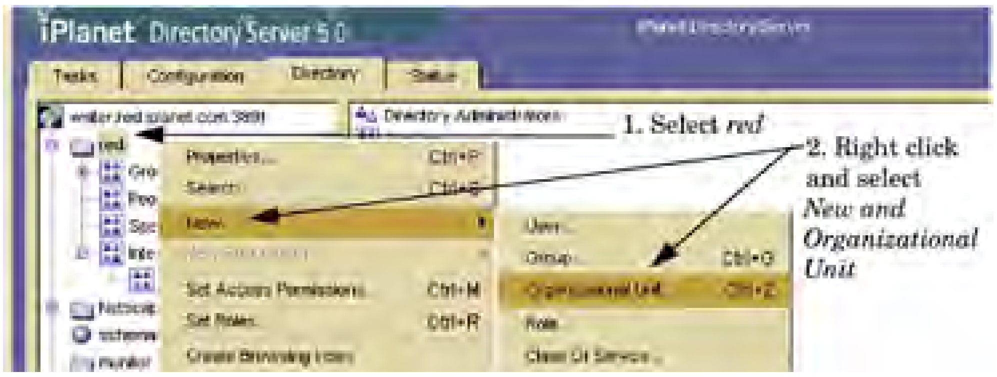
```
dn: uid=stacy,ou=People,ou=Inter-
  netTree,dc=red,dc=iplanet,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
givenName: Stacy
cn: Stacy here
uid: stacy
sn: here
userPassword: {SSHA}rLutXCI/REQaD5kOgAfzLLkpf/
  RJVRJt+E9Vfg==
creatorsName: cn=directory manager
modifiersName: cn=directory manager
createTimestamp: 20010816112811Z
modifyTimestamp: 20010816112811Z
nsUniqueId: a6692f02-1dd111b2-800cd250-749d06e5
```

Step 2. Back up and Remove the Data from InternetTree

Before restoring the data, back up the *People,InternetTree* data into another branch in the directory and then remove the data branch *People* under *InternetTree*.

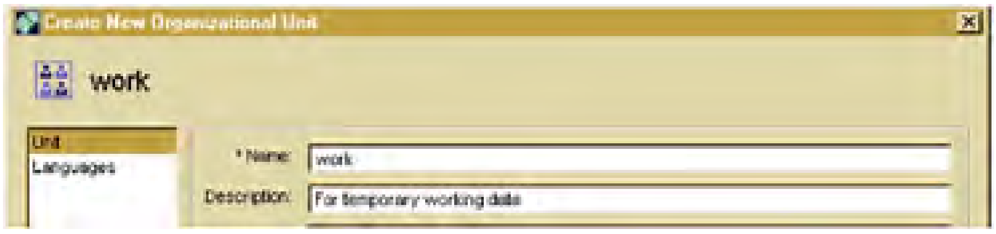
To do a simple data backup, create another organizational unit and put the *InternetTree* data underneath it (Figure 10.28). Create a new organizational unit called *work* (working data).

Figure 10.28. Create another organizational unit.



Enter the data and click *OK* (Figure 10.29).

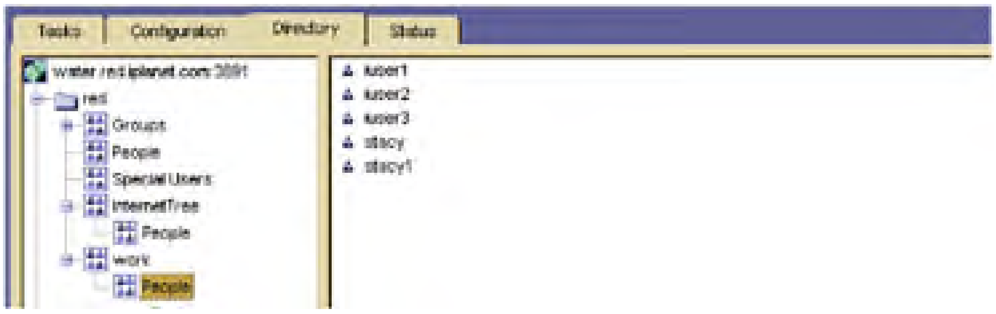
Figure 10.29. Name the new organizational unit.



* Name: work
Description: For temporary working data

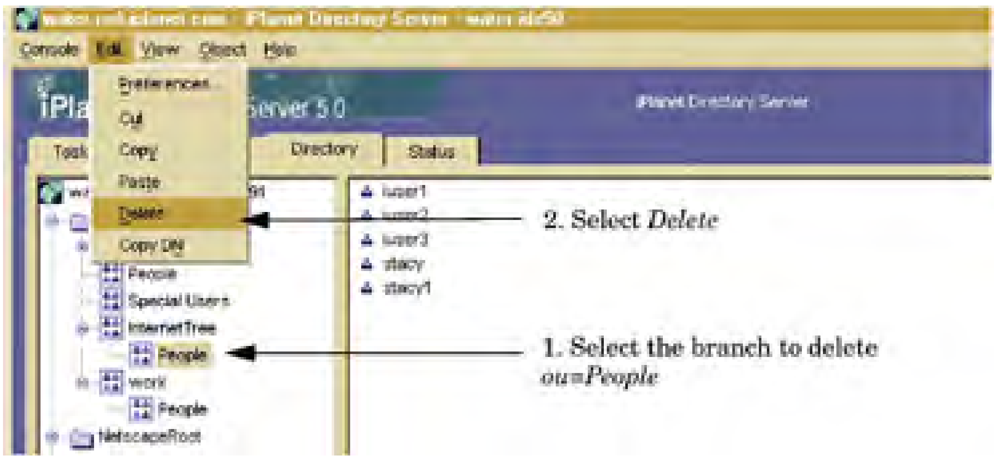
Then copy and paste the *People,InternetTree* branch under the *work* branch (Figure 10.30).

Figure 10.30. Copy and paste People under work.



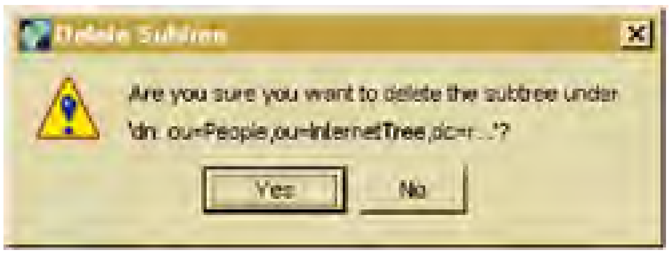
Delete the *People* branch under *InternetTree* (Figure 10.31).

Figure 10.31. Delete People under InternetTree.



To confirm the delete of the *People,InternetTree* branch, click *Yes* (Figure 10.32).

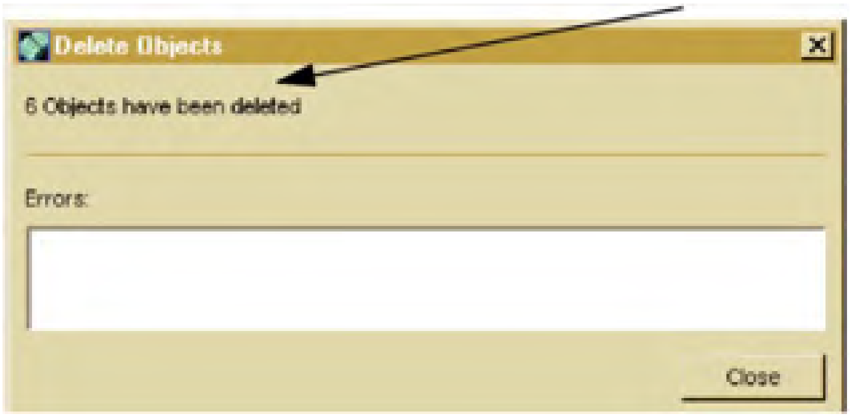
Figure 10.32. Confirm the delete.



Are you sure you want to delete the subtree under 'dn: ou=People,ou=InternetTree,dc=r...'?

The result is that five users and one ou (*People*), or six in total, have been deleted. Click *Close* (Figure 10.33).

Figure 10.33. Result of the delete.



6 Objects have been deleted

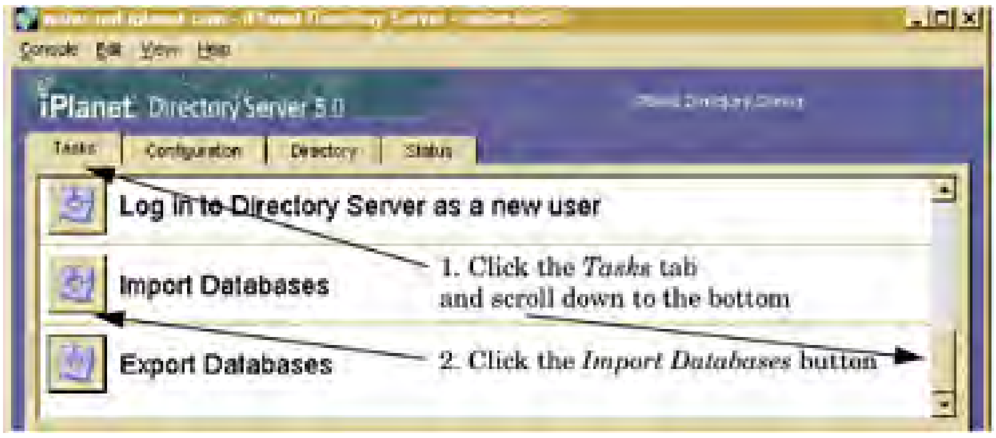
To summarize, the users from *People,InternetTree* branch have been

1. Backed up to ip1.ldif file
2. Copied to the *work* branch
3. Deleted from the *InternetTree* branch

Step 3. Restore the InternetTree/People Branch

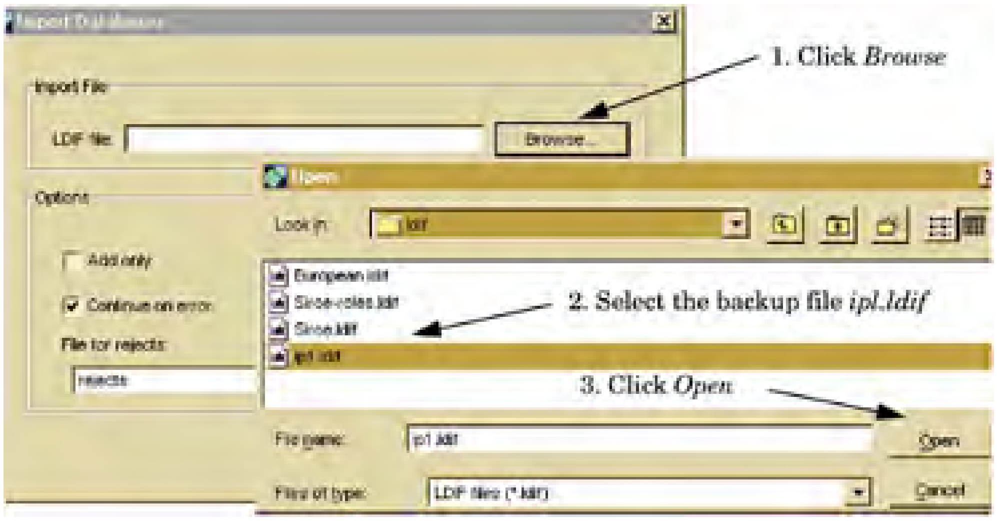
To restore the LDIF file that was exported in the previous example, use the directory server instance *Tasks* tab from the Console (Figure 10.34). Click the *Import Databases* button.

Figure 10.34. Import an LDIF file.


Console Edit View Help
iPlanet Directory Server 5.0
Tasks
Configuration
Directory
Status
Log in to Directory Server as a new user
Import Databases
Export Databases
1. Click the Tasks tab and scroll down to the bottom
2. Click the Import Databases button

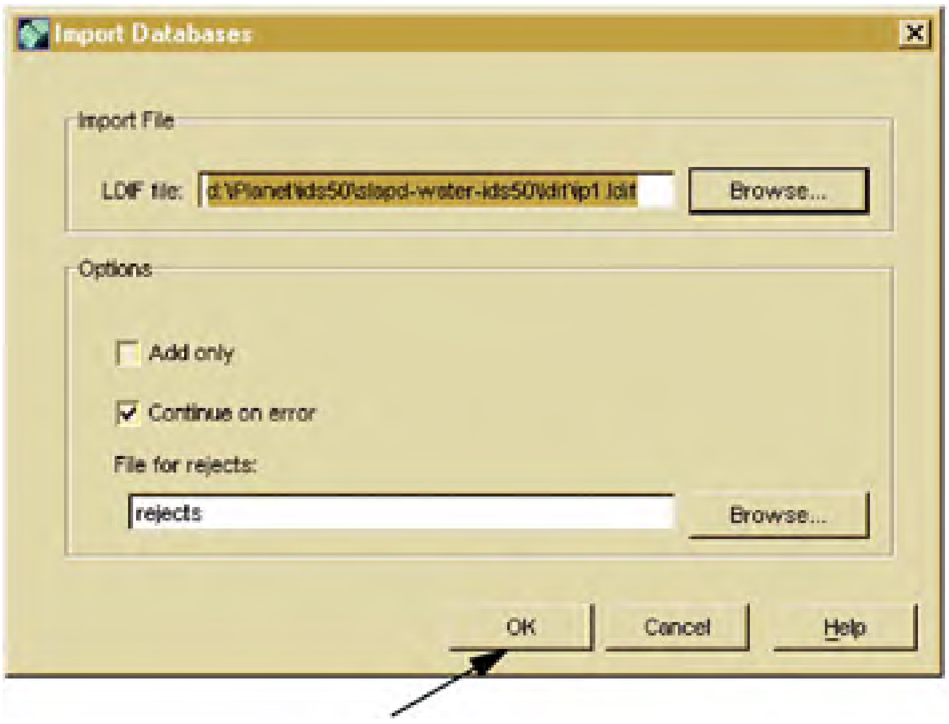
From the *Import Databases* window, select the exported LDIF file (Figure 10.35). To import the ip1.ldif LDIF file, click *Open*.

Figure 10.35. Select the exported LDIF file.



On the screen in Figure 10.36, click *OK* and the data from *ip1.ldif* will be loaded back into the directory database.

Figure 10.36. Load file into directory database.


Import Databases

Import File

LDIF file: d:\Planet\ids50\slapd-water-ids50\ldif\ip1.ldif

Options

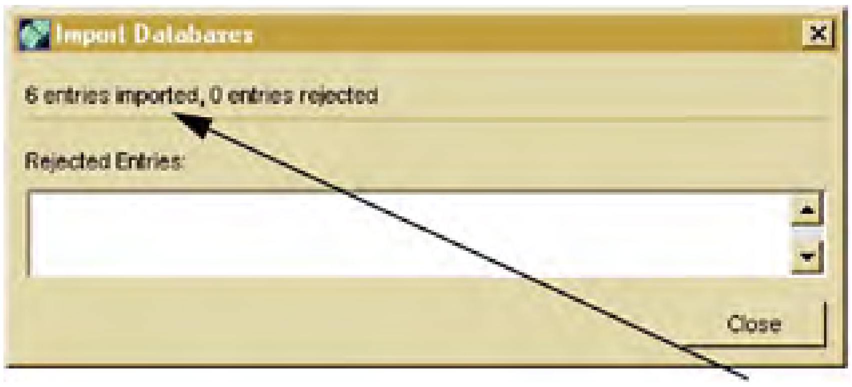
[ ] Add only

[✓] Continue on error

File for rejects: rejects

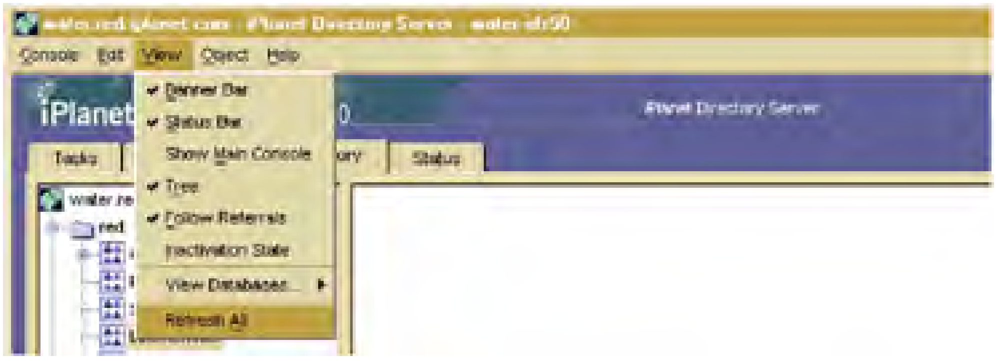
The *Import Databases* screen shows a total of six entries (five users and one *People,Internet* branch; Figure 10.37). To see the changes, return to the *Directory* tab and, from *View*, select *Refresh All* (Figure 10.38).

Figure 10.37. Screen showing six entries into the database.



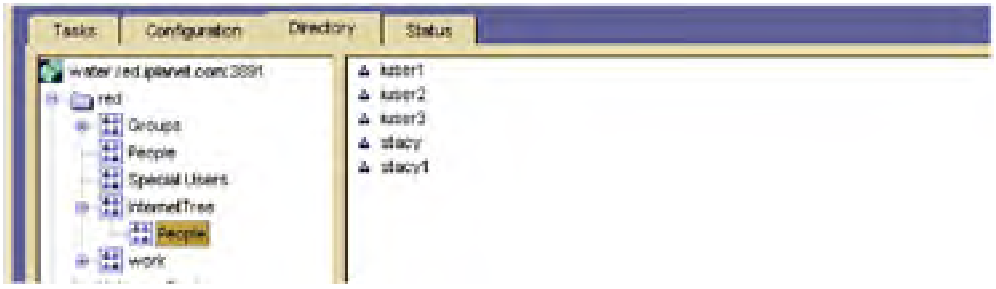
6 entries imported, 0 entries rejected

Figure 10.38. Reviewing the changes.



If you successfully followed these steps, the data branch is back (see Figure 10.39).

Figure 10.39. Return of the People data branch.



Saving and restoring LDIF files is very useful for

1. Backing up data in a text format so that the data can be altered manually using a simple text editor.
2. Restoring test data. Often, before I run a test, I save a copy of the original data. During testing, the data is modified. After the test, I restore the data to the original start test state. Then I start the test cycle all over again with fresh data. LDIF files are excellent for this.
3. Trading data with other directories. You can export data into an LDIF file, copy the data to another directory server, and import the LDIF file into the other directory server.

Summary

This chapter focused on using the Console and getting familiar with directory data. The Console is useful for searching for and adding data based on the default base schema that comes with the Sun ONE Directory Server. The Console includes tools to export and import data using standard LDIF files.

Chapter 11. Using the Sun ONE Directory Database: Directory Data Structures

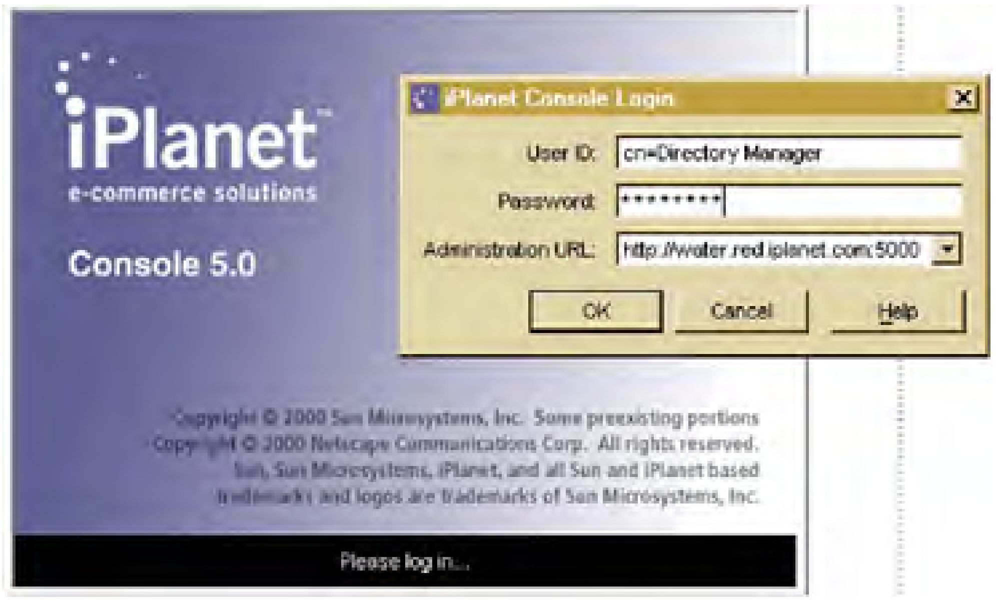
The Sun ONE Directory Server is a specialized hierarchical database system. From the very top, a Sun ONE Directory Server is ordered like this:

1. The directory server system has directory instances.
2. Within each directory instance are directory databases. Each of the databases has data files in a file directory.

Before starting this chapter, make sure that you are familiar with the material in the previous chapter and that you know how to use the Console and to search for members in the directory. You need that foundation to understand this chapter.

This chapter starts with definitions of directory data structures and continues with the steps to set up a complete data tree structure from top to bottom. To apply the definitions and work with the directory structure, log in to the directory server using the Console (as in the previous chapter) (see Figure 11.1).

Figure 11.1. Log in to directory server.

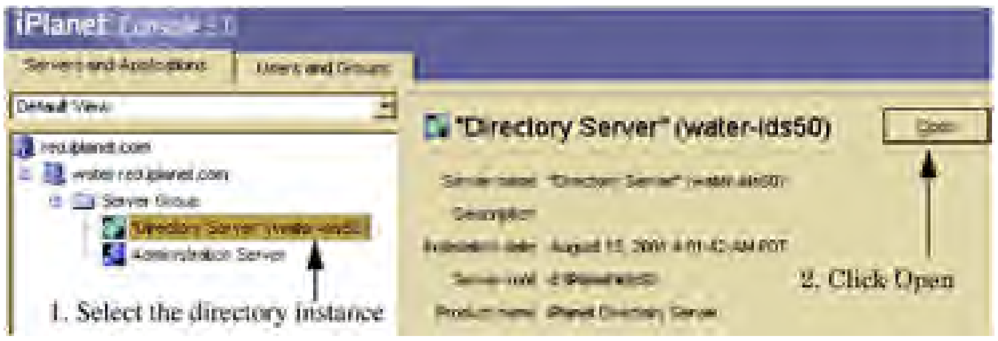


User ID: cn=Directory Manager
Password: ********
Administration URL: http://water.red.iplanet.com:5000

Copyright © 2000 Sun Microsystems, Inc. Some preexisting portions Copyright © 2000 Netscape Communications Corp. All rights reserved. Sun, Sun Microsystems, iPlanet, and all Sun and iPlanet based trademarks and logos are trademarks of Sun Microsystems, Inc.

After logging in, open the *"Directory Server" (water-ids50)* option (Figure 11.2) to manage the directory instance that listens to port 389.

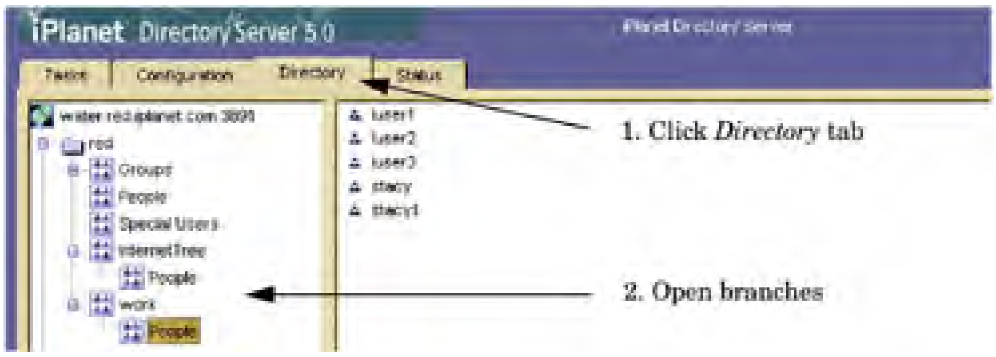
Figure 11.2. Open "Directory Server" (water-ids50).



Server name: "Directory Server" (water-ids50)
Description
Installation date: August 15, 2001 4:01:42 AM PDT
Server root: d:\iPlanet\ids50
Product name: iPlanet Directory Server

Figure 11.3 shows the directory data from the example in the previous chapter using the GUI view.

Figure 11.3. Directory administration GUI view of directory data



Directory Data Structure Language

Another useful tool for viewing the data is the command ldapsearch. The following examples start from the top suffix node and work down to the data at the bottom.

To use ldapsearch, go to the directory /iplanet/ids50/shared/bin.

Example 1: Suffixes

To get the list of suffixes in the directory, use the following command (Note: UNIX users use ./ldapsearch):

```
ldapsearch -1 -h "water.red.iplanet.com" -p 3891 -s base
  -b "" "objectclass=*" namingContexts
Search result:
dn:
namingContexts: dc=red,dc=iplanet,dc=com
namingContexts: o=NetscapeRoot
```

dc=red,dc=iplanet,dc=co
m
o=NetscapeRoot

Directory root suffix for user data

Directory root suffix for the Sun

ONE Directory Server configuration data

Definition: Suffix

A suffix is the root level of a directory.

Example: dc=red,dc=iplanet,dc=com

The directory search query list for Example 1 shows two suffixes.

Example 2. Command Syntax

The following represents ldapsearch command syntax (see Table 11-1):

```
ldapsearch [-1] -h <hostname> -p <port> -s <base|one|sub>
  -b "dn" "<filter>" [attribute list]
```

Here is a sample search query:

```
ldapsearch -1 -h "water.red.iplanet.com" -p 389 -s base -
  b "" "objectclass=*" namingContexts
```

Table 11-1. Sample ldapsearch Syntax

Tag	Description
-1 (numeric one)	Do not display the *Version 1* line in the output. This is optional.
-h "water.red.iplanet.com"	The hostname of the directory server.
-p 389	Port number the directory server listens on.
-s base	Type of listing; -s one means one level down; -s sub is for all levels down.
-b ""	This is the name of a branch node (a DN value); for example: "ou=InternetTree,dc=red,dc=iplanet,dc=com"
"objectclass=*	(the search filter). This example filter filters entries that have an *objectclass* (any value *).
namingContexts	Retrieve only the attribute values of the attribute namingContexts.

Example 3. Objects

This example is a search query to get a list of objects under a suffix. Note, I use x as an attribute; however, there is no x attribute. This causes ldapsearch to list only the DNs. Also, I am using -s one vs. -s base. This query lists only one level down from the search node "dc=red,dc=iplanet,dc=com".

```
l./ldapsearch -1 -h "water.red.iplanet.com" -p 389 -s one
  -b "dc=red,dc=iplanet,dc=com" "ou=*" x
Result:
dn: ou=Groups, dc=red,dc=iplanet,dc=com
dn: ou=People, dc=red,dc=iplanet,dc=com
dn: ou=Special Users,dc=red,dc=iplanet,dc=com
dn: ou=InternetTree,dc=red,dc=iplanet,dc=com
dn: ou=work,dc=red,dc=iplanet,dc=com
```

Definition: Data Tree

LDAP directories store data in a hierarchical structure that can easily be queried and updated. To allow applications to locate entries in the directory structure, LDAP uses a unique identifier called the distinguished name (DN). The DN consists of a list of key entry attributes from the suffix down to the unique attribute value pair that distinguishes the entry.

Example 4. People

To walk one more level down the data tree, take a DN from the above list and use it in another search query. Use the following query to get the list of *People*.

```
ldapsearch -1 -h "water.red.iplanet.com" -p 389 -s one -b
"ou=People,dc=red,dc=iplanet,dc=com" "objectclass
  =*" x
Result:
dn: uid=stacy,ou=People, dc=red,dc=iplanet,dc=com
dn: uid=stacy1,ou=People, dc=red,dc=iplanet,dc=com
```

Definition: DN (Distinguished Name)

Uniquely identifies an entry in the directory.

Example DN: uid=stacy,ou=People,dc=red,dc=iplanet, dc=com

As well, a DN uniquely identifies branch nodes in the directory.

Example DN: ou=People,dc=red,dc=iplanet,dc=com

Each DN is a branch node in the data tree. Each DN has a unique Relative Distinguished Name.

Definition: Relative Distinguished Name (RDN)

The left most attribute pair of a DN.

Example DN: uid=stacy,ou=People,dc=red,dc=iplanet, dc=com

The RDN is uid=stacy.

Furthermore, uid=stacy is a unique entry under ou=People, dc=red,dc=iplanet,dc=com.

And ou=People is a unique entry under dc=red,dc=iplanet, dc=com.

Example 5. Information about User Entry

Now take a DN from the above list to list information about a user entry:

```
ldapsearch -1 -h "water.red.iplanet.com" -p 389 -s one -b
"ou=People,dc=red,dc=iplanet,dc=com" "uid=stacy"
Result:
dn: uid=stacy,ou=People, dc=red,dc=iplanet,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
givenName: Stacy
cn: Stacy here
uid: stacy
sn: here
```

Definition: Entry, an entry in the directory

This is a collection of attribute value pairs located at a branch point. The above search query describes the entry for uid=stacy. An entry has a number of attribute value pairs such as givenName=Stacy, cn=Stacy here, uid=stacy, and sn=here.

Definition: Attribute value pair

An attribute value pair is an attribute value associated to an attribute.

Example DN: uid=stacy,ou=People, dc=red,dc=iplanet, dc=com

RDN: The attribute value pair uid=stacy

A branch point attribute value pair ou=People

The suffix has three attribute value pairs: dc=red,dc=iplanet, dc=com

Definition: Attribute

This is the left-hand side of an attribute value pair.

Example attribute value pair uid=stacy

The attribute is: uid

The value is: stacy

Each attribute is defined by an attribute type. The attribute type defines what data an attribute value can contain. An attribute type could, for example, be a string, a binary value, or even another attribute. The attribute value is constrained by the attribute type. Note, the attribute defines the type of data the value may have. The attribute value must match the attribute type.

Example 6. People under ou=InternetTree Branch

List of *People* under the *ou=InternetTree* branch whose user ID (*uid*) starts with iuser. Print the *cn* (common name or full name).

```
ldapsearch -1 -h "water.red.iplanet.com" -p 3891 -S uid
  -s one -b
"ou=People,ou=InternetTree,dc=red,dc=iplanet,dc=com"
  "uid=iuser*" cn
Result:
dn: uid=iuser1,ou=People,ou=Inter-
  netTree,dc=red,dc=iplanet,dc=com
cn: iuser1 here
dn: uid=iuser2,ou=People,ou=Inter-
  netTree,dc=red,dc=iplanet,dc=com
cn: iuser2 here
dn: uid=iuser3,ou=People,ou=Inter-
  netTree,dc=red,dc=iplanet,dc=com
cn: iuser3 here
```

In the previous search query, the filter used is "uid=iuser*". The syntax for using multiple filter qualifiers is

(<logical>(<attribute filter>)(<attribute filter>))

Logicals are & (and) or | (or)

An <attribute filter> is <attribute><comparison value><compare value>

Comparison values are = | > | < | >= | <=

Comparison values can use strings or * for a wild card value.

Or the multiple filter qualifier syntax for *not* is ! as in

(!(<attribute filter>))

Example filters include:

1. (uid=iuser*) is a uid starting with *iuser*
2. (!(uid=iuser*) represents users whose names do not start with *iuser*
3. (|(uid=iuser1)(uid=iuser2)) are users that have a uid of *iuser1* or uid of *iuser2*

Example 7. List dn Values for the Users, user1 and user2

```
ldapsearch -1 -h "water.red.iplanet.com" -p 389 -S uid
  -s one -b
"ou=People,ou=InternetTree,dc=red,dc=iplanet,dc=com"
  "(|(uid=user1)(uid=user2)" x
Result:
dn: uid=iuser1,ou=People,ou=InternetTree,dc=red,dc=
  iplanet,dc=com
dn: uid=iuser2,ou=People,ou=InternetTree,dc=red,dc=
  iplanet,dc=com
ldapsearch -1 -h "water.red.iplanet.com" -p 389 -s one -b
"ou=People,ou=InternetTree,dc=red,dc=iplanet,dc=com"
  "(|(uid=user1)(uid=user2)" x
```

The only way to get good at walking up and down the directory tree using the ldapsearch command is to try a lot of searches and add more data. Then do searches, and add data into the directory again, and try more searches, then add more data, and do more searches, and so on and so on.

Modifying the Directory Schema

In the previous section, the ldapsearch command was used to walk down the directory from the suffix root. This section describes how to set up a data tree structure starting with a new suffix and working on down.

Adding a Suffix and Database

When a *new root suffix* is added (Figure 11.4), a new database file directory is created. Create the new root suffix *o=BookWork*—enter it into the *New suffix* box (Figure 11.5). Figure 11.6 shows the new suffix *o=BookWork* has been added under the directory instance of water.red.iplanet.com:389 and under the *Data* branch. In addition, under the file directory for the water.red.iplanet.com:389 instance, a new file directory with database files has been created (Figure 11.7).

Figure 11.4. Create new root suffix.

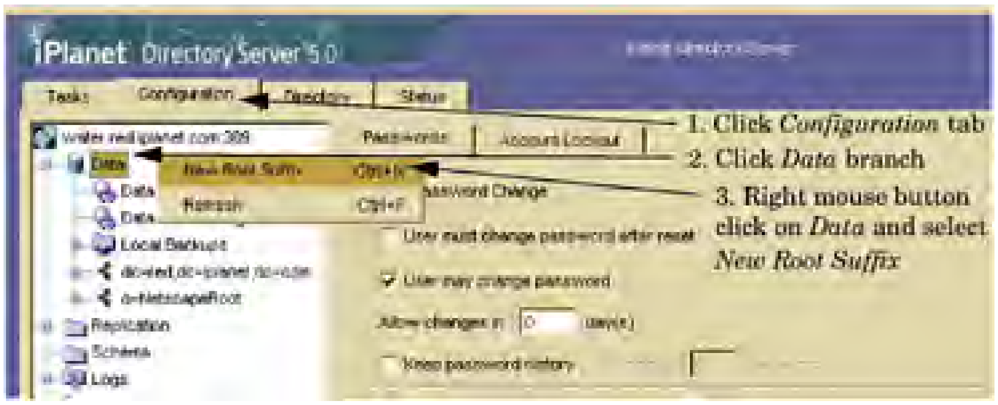
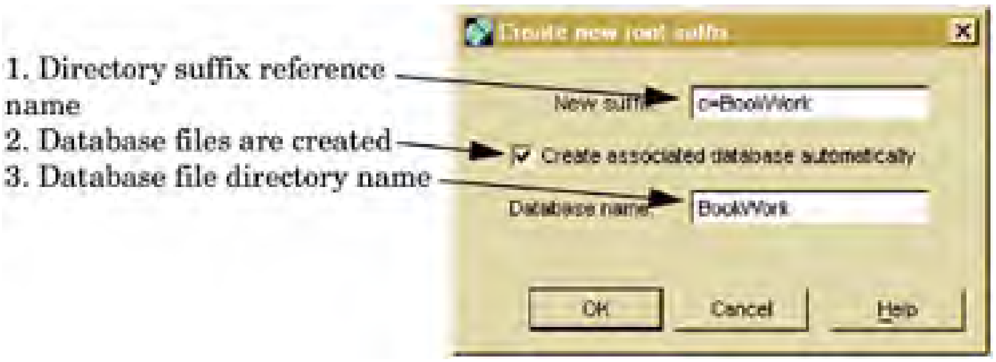
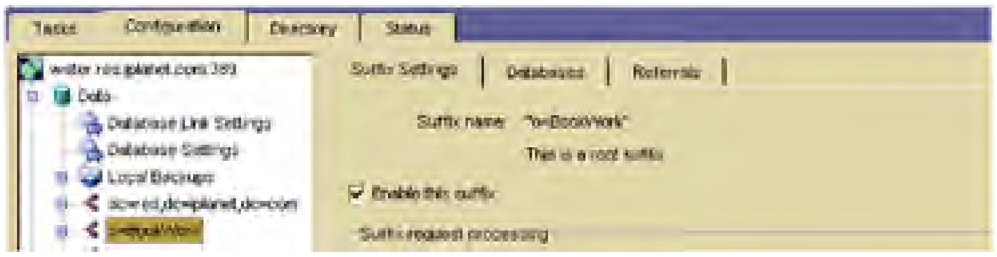


Figure 11.5. Enter new suffix and directory name.


Create new root suffix
1. Directory suffix reference name
2. Database files are created
3. Database file directory name
OK
Cancel
Help

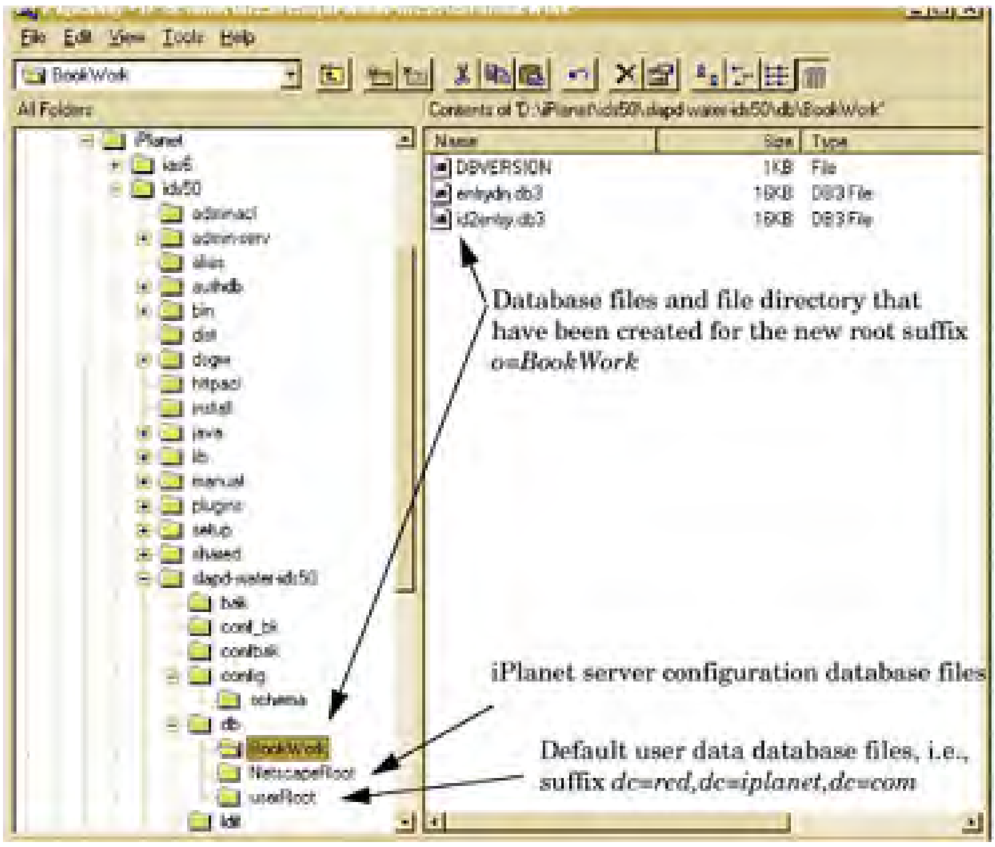
New suffix: o=BookWork
✓ Create associated database automatically
Database name: BookWork

Figure 11.6. New suffix under Data branch.



Suffix name: "o=BookWork"
This is a root suffix
✓ Enable this suffix

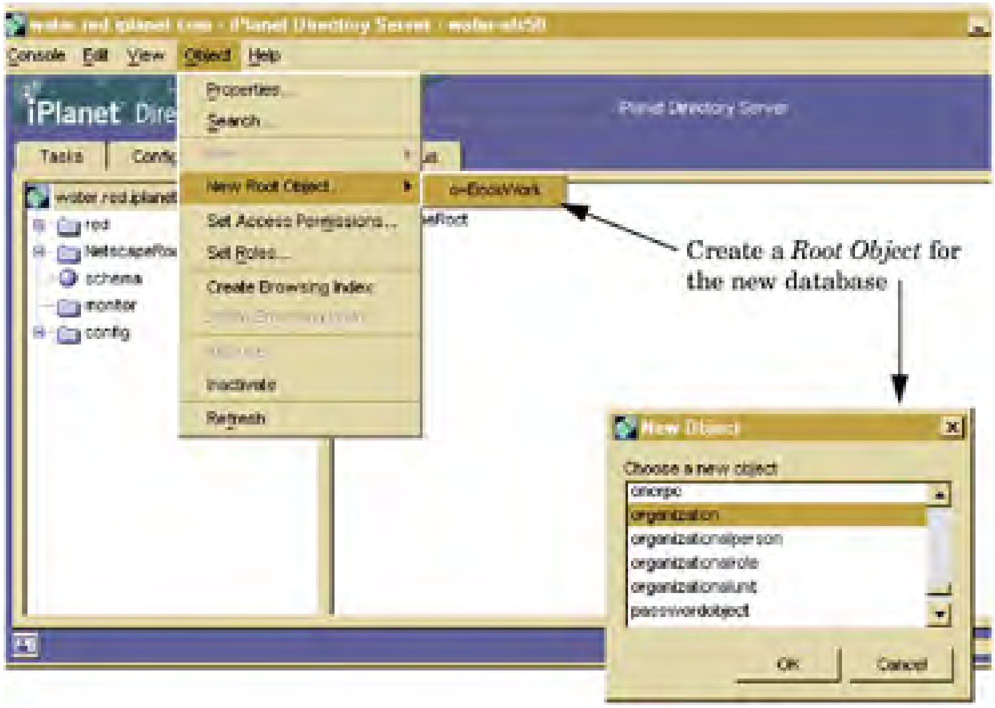
Figure 11.7. New file directory created.


Database files and file directory that have been created for the new root suffix o=BookWork
iPlanet server configuration database files
Default user data database files, i.e., suffix dc=red,dc=iplanet,dc=com
BookWork
NetscapeRoot
userRoot
DBVERSION
entrydn.db3
id2entry.db3

In a UNIX environment, the database files can be mounted on another hard drive. This separates database read/writes from the other database file read/writes, thereby improving read/write performance.

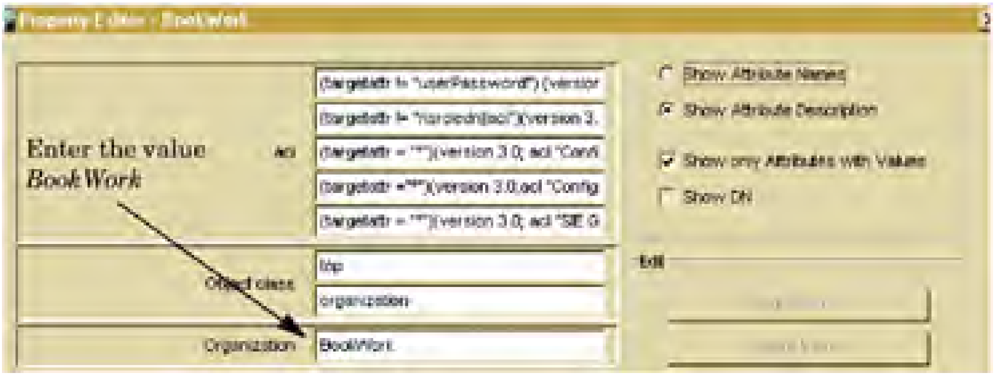
The next step is to create the suffix data branch as in Figure 11.8. Click the *Directory* tab to create a *New Root Object,* which is the top of the data branch.

Figure 11.8. Create the suffix data branch



In the *Property Editor* window, enter *BookWork* in the *Organization* field, and then click *OK* (Figure 11.9). To view the new object that has been added, refresh the Console directory data as in Figure 11.10. The result of the refresh is Figure 11.11.

Figure 11.9. Enter BookWork.


Property Editor - BookWork
Enter the value BookWork
aci
Object class
Organization
Show Attribute Names
Show Attribute Description
Show only Attributes with Values
Show DN
Edit

(targetattr != "userPassword") (versior
(targetattr != "nsroledn||aci")(version 3.
(targetattr = "*")(version 3.0; acl "Conf
(targetattr ="*")(version 3.0;acl "Config
(targetattr = "*")(version 3.0; acl "SIE G
top
organization
BookWork

Figure 11.10. Refresh the directory data.

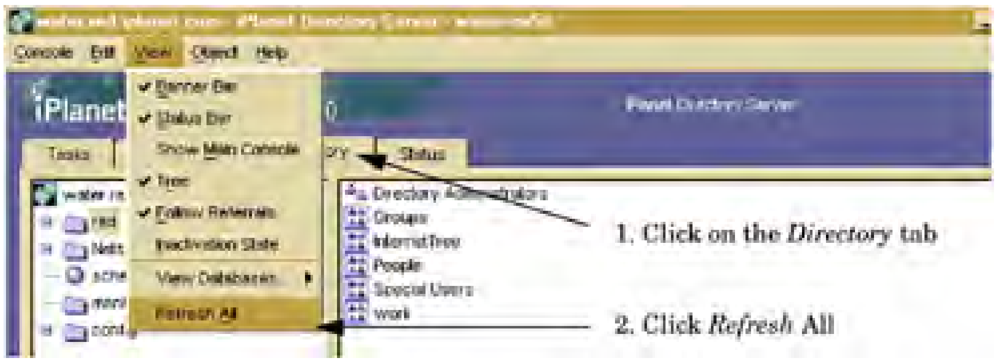
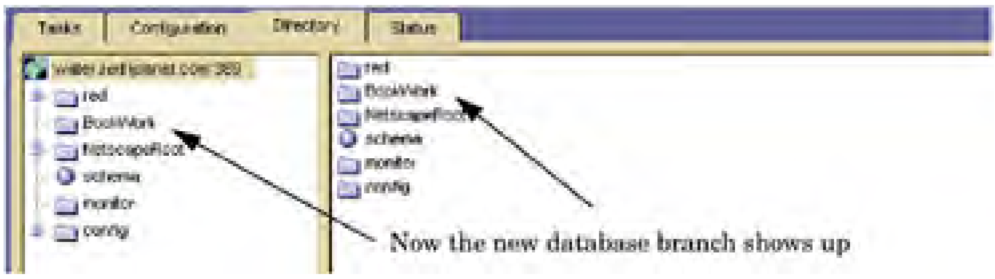
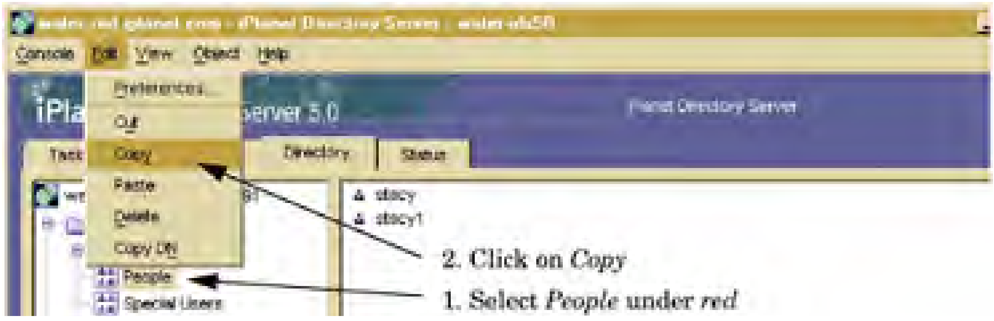


Figure 11.11. Result of refreshing Console directory data.



Add some *People* data under BookWork by copying People from red/People (Figure 11.12). Paste People data into the BookWork branch (Figure 11.13).

Figure 11.12. Copy People data under red.



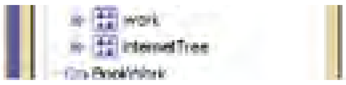
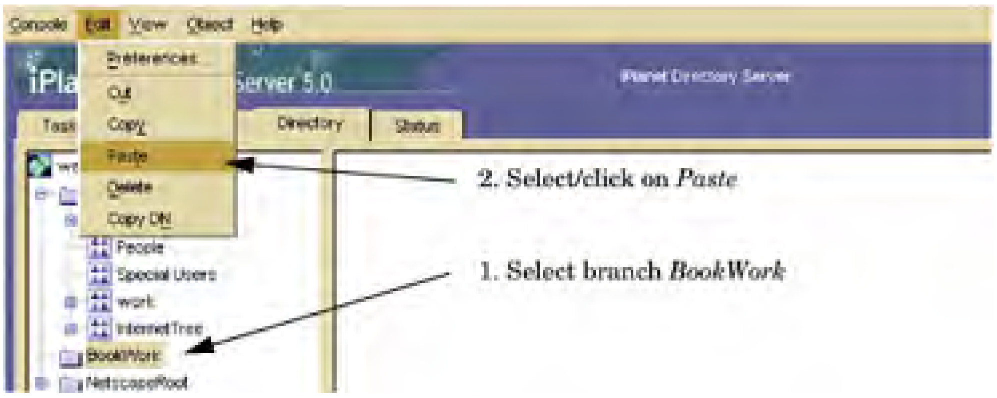
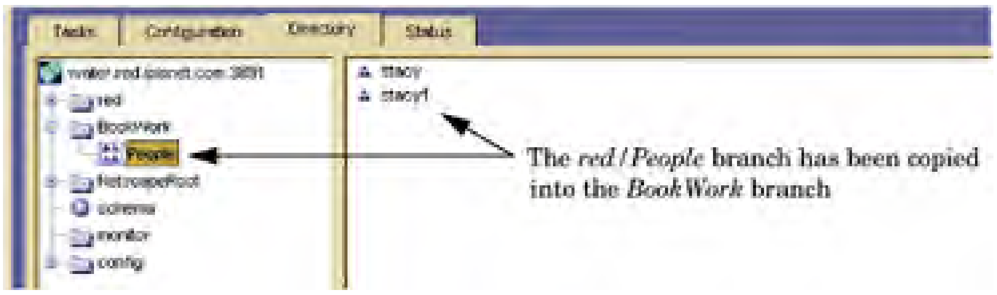


Figure 11.13. Add People into BookWork branch.



View the added data on the next screen that appears (Figure 11.14).

Figure 11.14. View added data.



Use ldapsearch to view the new data tree structures, using the following command to list the suffixes in the directory:

```
ldapsearch -1 -h "water.red.iplanet.com" -p 3891 -s base
  -b "" "objectclass=*" namingContexts
Result:
dn:
namingContexts: o=BookWork
namingContexts: dc=red,dc=iplanet,dc=com
namingContexts: o=NetscapeRoot
```

List objects under "o=BookWork" using the command

```
ldapsearch -1 -h "water.red.iplanet.com" -p 389 -s one -b
  "o=BookWork" "objectclass=*" x
Result:
dn: ou=People,o=BookWork
```

List *People* under "ou=People,o=BookWork" using the command

```
ldapsearch -1 -h "water.red.iplanet.com" -p 3891 -s one -
  b "ou=People,o=BookWork" "objectclass=*" x
Result:
dn: uid=stacy,ou=People,o=BookWork
dn: uid=stacy1,ou=People,o=BookWork
```

Adding an Object and Attributes into the Directory Schema

The previous section showed how to add a new suffix and data structures below the new suffix. Now let's look at how to create objects and attributes that are used to store data.

Since attributes are added into objects, the attributes are created before the objects. In this section the following will be created:

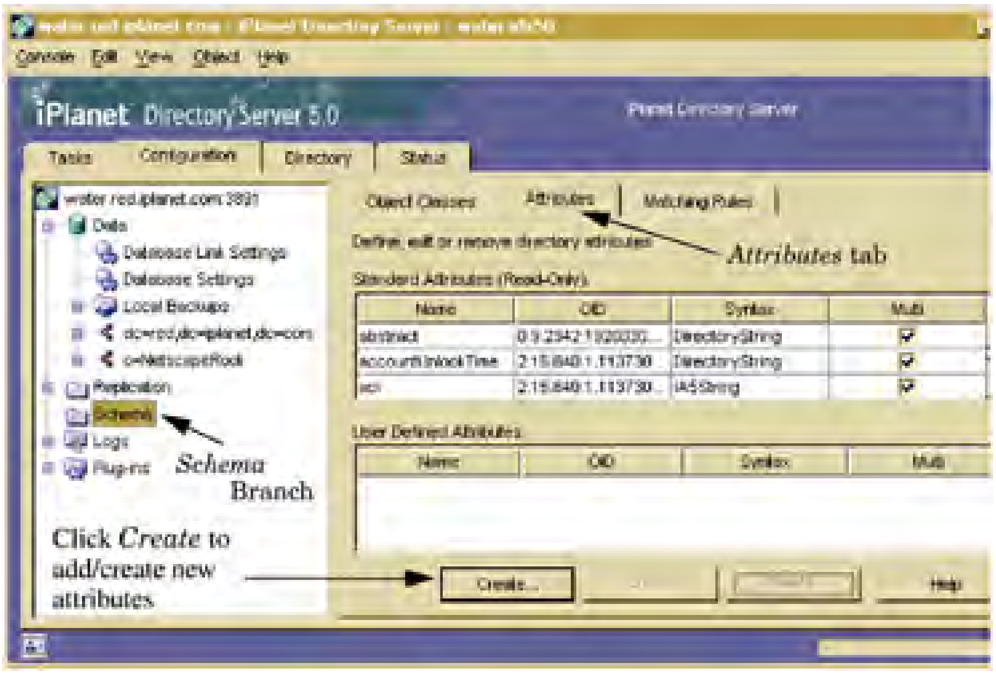
1. Create attributes: code and full-description. Attributes are like attributes in a relational database; they are the

low-level fieldnames for single pieces of data.

2. Create an object: task. Attributes are put together into objects. In relational terms, objects that have attributes can be compared conceptually to tables. The *tablename* is task.
3. Objects are organized into and under organization units. A task object is then added under o=BookWork.

Use the Console to modify the schema. From the directory instance administration window, click on the *Configuration* tab, then click the *Schema* branch, and then the *Attributes* tab (Figure 11.15). Now click *Create*, which will open the *Create Attributes* window as shown in Figure 11.16 and 11.17. Create two attributes: code and full-description.

Figure 11.15. Attributes in the schema.



Define, edit or remove directory attributes

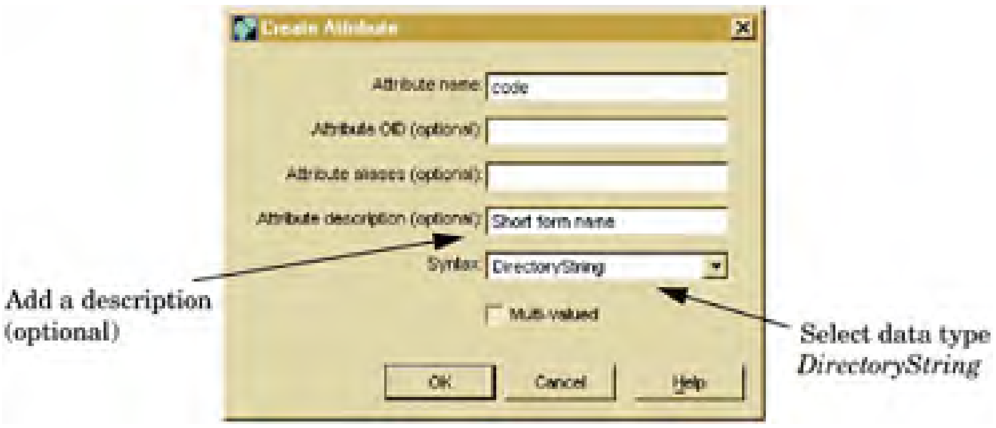
Standard Attributes (Read-Only)

Name	OID	Syntax	Multi
abstract	0.9.2342.1920030...	DirectoryString	✓
accountUnlockTime	2.16.840.1.113730...	DirectoryString	✓
aci	2.16.840.1.113730...	IA5String	✓

User Defined Attributes

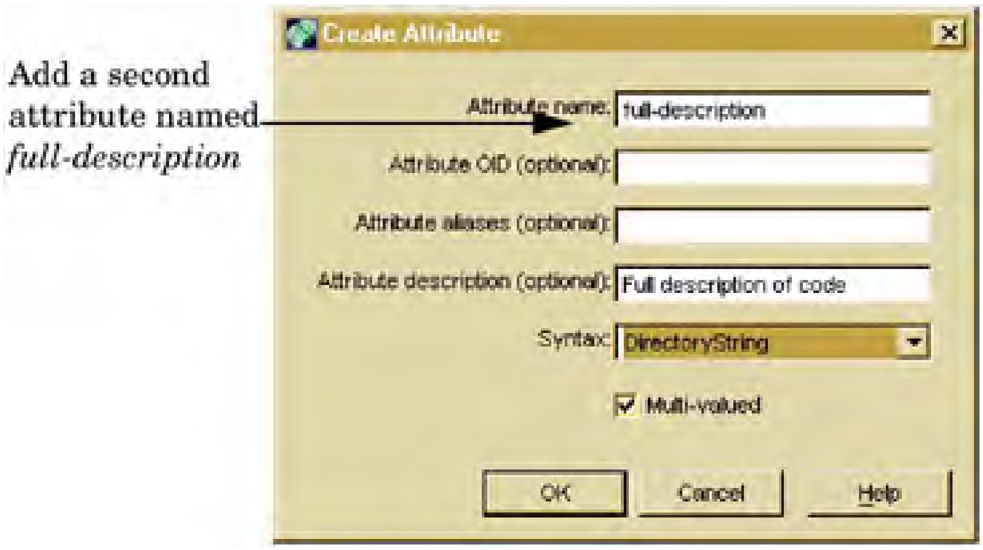
Name	OID	Syntax	Multi

Figure 11.16. Creating a new attribute: code.


Create Attribute
Add a description (optional)
Select data type DirectoryString
OK
Cancel
Help

Attribute name:	code
Attribute OID (optional):	
Attribute aliases (optional):	
Attribute description (optional):	Short form name
Syntax:	DirectoryString
Multi-valued	☐

Figure 11.17. Creating a new attribute: full-description.

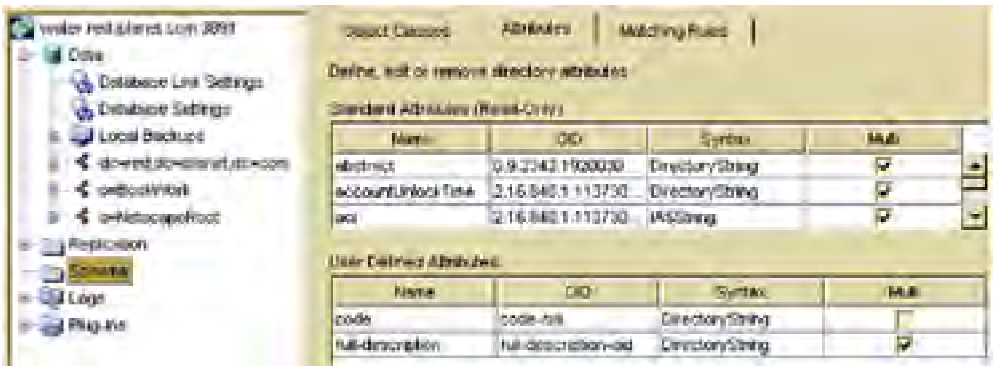


Attribute name: full-description
Attribute OID (optional):
Attribute aliases (optional):
Attribute description (optional): Full description of code
Syntax: DirectoryString
[✓] Multi-valued

First, the code attribute (Figure 11.16): Enter *code* as the *Attribute name* and select D*irectoryString* as the *Syntax*. You also have the option of describing the attribute. Then follow the same steps for the attribute full-description (Figure 11.17). Clicking *OK* will take you to the next step.

These two new attributes now appear under *User Defined Attributes* in Figure 11.18.

Figure 11.18. Viewing the new attributes.



Define, edit or remove directory attributes

Standard Attributes (Read-Only)

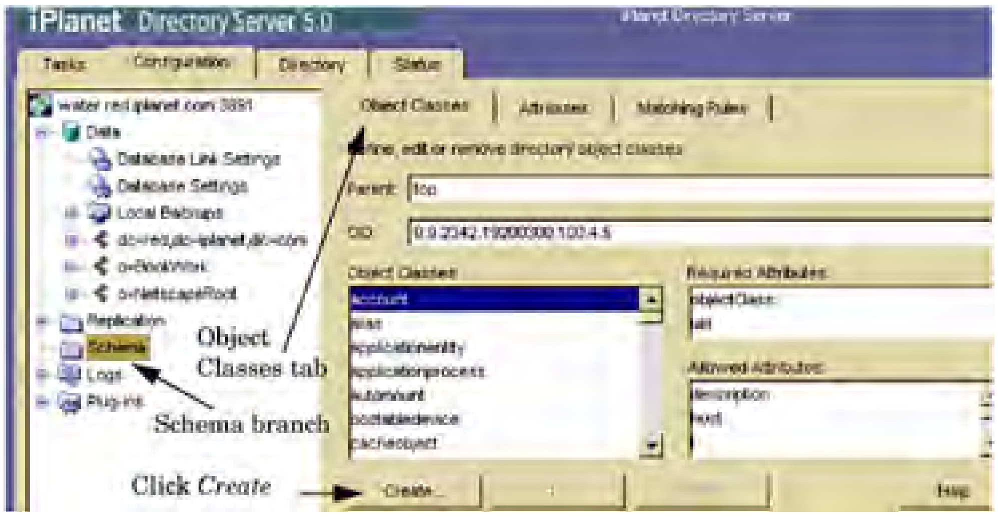
Name	OID	Syntax	Multi
abstract	0.9.2342.1920030...	DirectoryString	✓
accountUnlockTime	2.16.840.1.113730...	DirectoryString	✓
aci	2.16.840.1.113730...	IA5String	✓

User Defined Attributes

Name	OID	Syntax	Multi
code	code-oid	DirectoryString	
full-description	full-description-oid	DirectoryString	✓

Next, create an object named task that uses the new attributes code and full-description. To begin, click the *Object Classes* tab (Figure 11.19). Then click *Create*.

Figure 11.19. Creating the object task.



Define, edit or remove directory object classes
Parent: top
OID: 0.9.2342.19200300.100.4.5
Object Classes: account, alias, applicationentity, applicationprocess, automount, bootableDevice, cacheobject
Required Attributes: objectClass, uid
Allowed Attributes: description, host

In the Create Object Class window (Figure 11.20), leave the default *task* in the *Name* box. Then add the two attributes and click *OK*. A confirmation window will appear; click *OK*. A new object with new attributes has been created (Figure 11.21).

Figure 11.20. Adding the new attributes to the object.

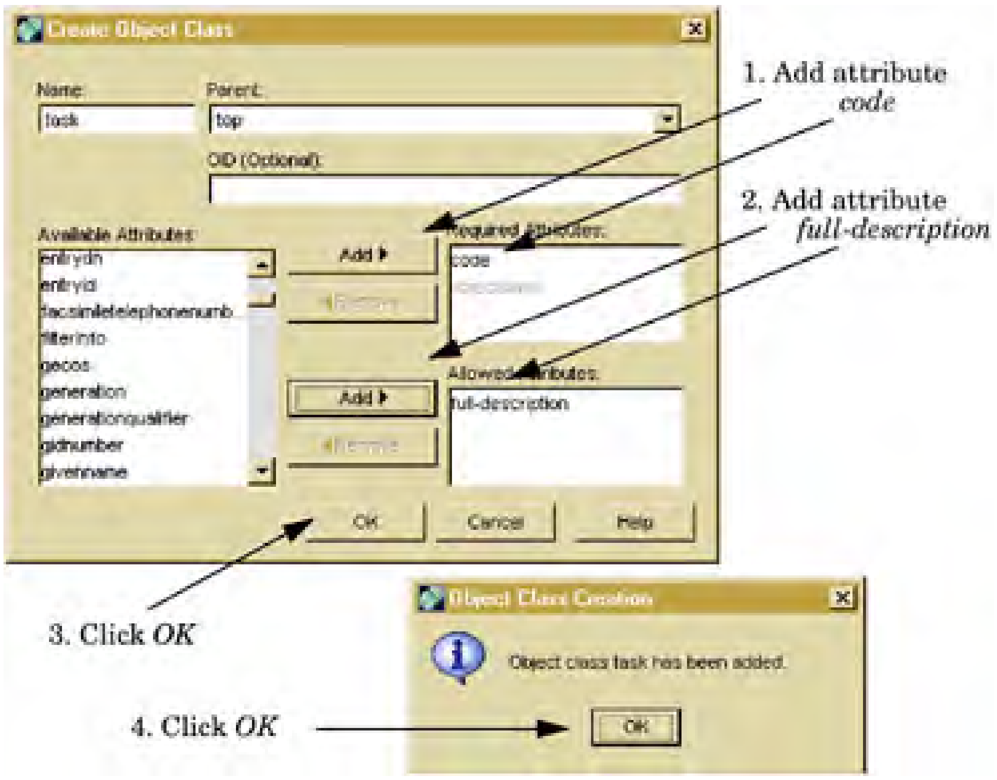
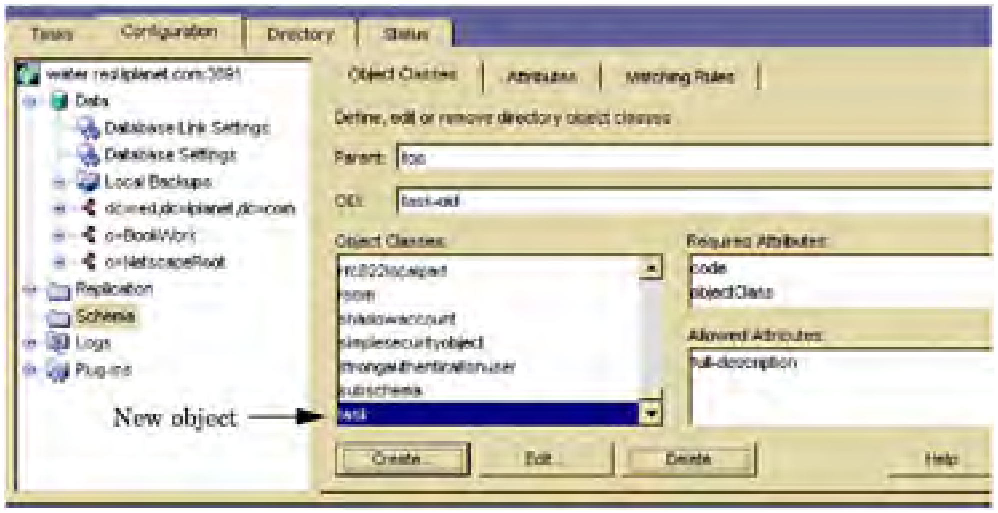


Figure 11.21. Attributes in the task object.



When objects and attributes are added into the schema, a line is added for each object or attribute in the directory server instance schema file 99-user.ldif. The file named /iplanet/ids50/slapd-water-ids50/config/schema/99-user.ldif has the lines in the box in the file.

The following lines were added in the directory server instance schema for the task object and the two new attributes:

```
objectClasses: ( task-oid NAME 'task' SUP top STRUCTURAL
  MUST code MAY full-
description X-ORIGIN 'user defined' )

attributeTypes: ( code-oid NAME 'code' DESC 'Short form
  name' SYNTAX
1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE X-ORIGIN
  'user defined' )

attributeTypes: ( full-description-oid NAME 'full-
  description' DESC 'Full description of code'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORIGIN 'user
  defined' )
```

(Note: When creating an LDIF file to move the *BookWork* database to another directory server, these schema lines need to be added into the 99-user.ldif file of the other directory server.)

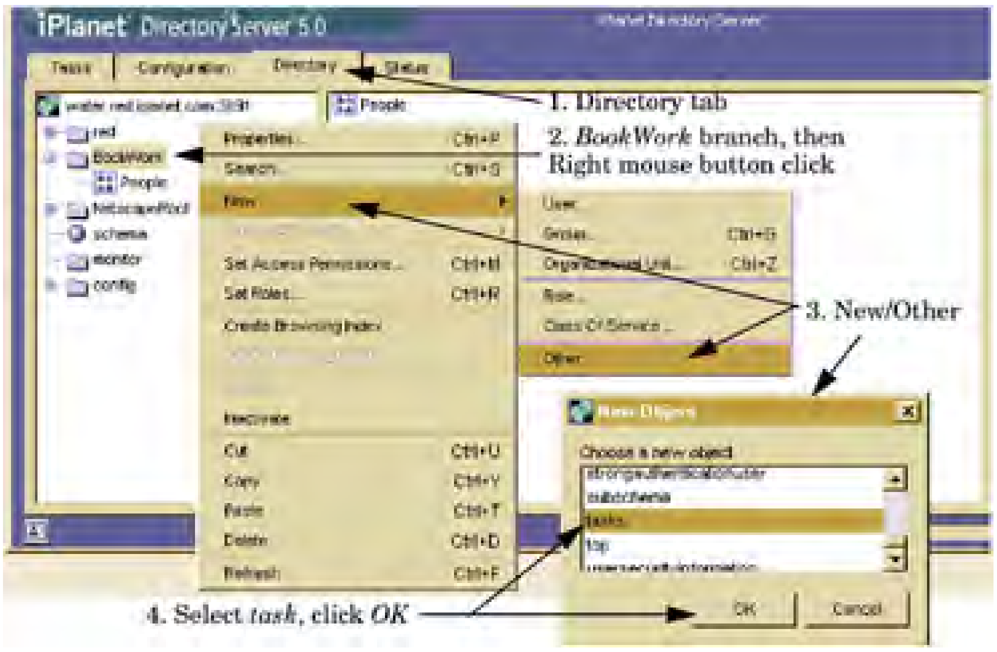
Adding the task Object under BookWork

The following example demonstrates how to

1. Add the task object under *BookWork*
2. Add data into the task object
3. Add task objects under the added task object
4. Add data into each of the task objects
5. Do some LDAP searches on the newly added objects and data.

To add the task object under BookWork, first click the *Directory* tab (shown in Figure 11.22). Click on *BookWork* in the left-hand column, and then right click. Select *New* and then *Other*. In the *New Object* window, select *task* and click *OK*.

Figure 11.22. Adding task object under BookWork.



The next window is the *Property Editor—New* (see Figure 11.23). At first, only the required attributes appear in the window. Click *Add Attribute*. When the *Add Attribute* window opens, select *full-description*, and click *OK*. In the next window (Figure 11.24), enter the values for *code* (*taskList*) and *full-description* (*List of things to do*). Select *full-description* as *Naming Attribute*. Then click *OK*.

Figure 11.23. Select the full-description attribute to be used.

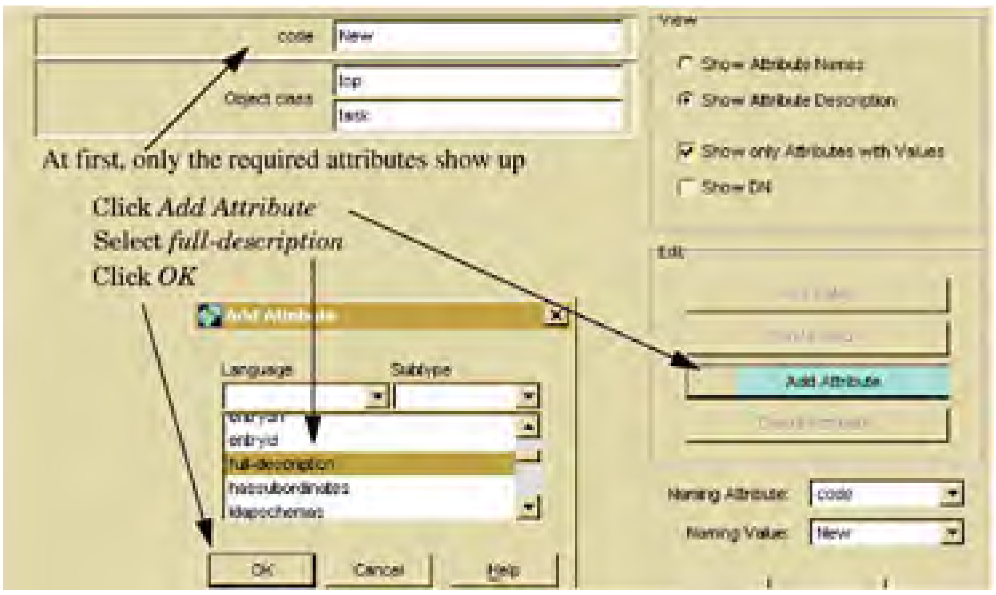
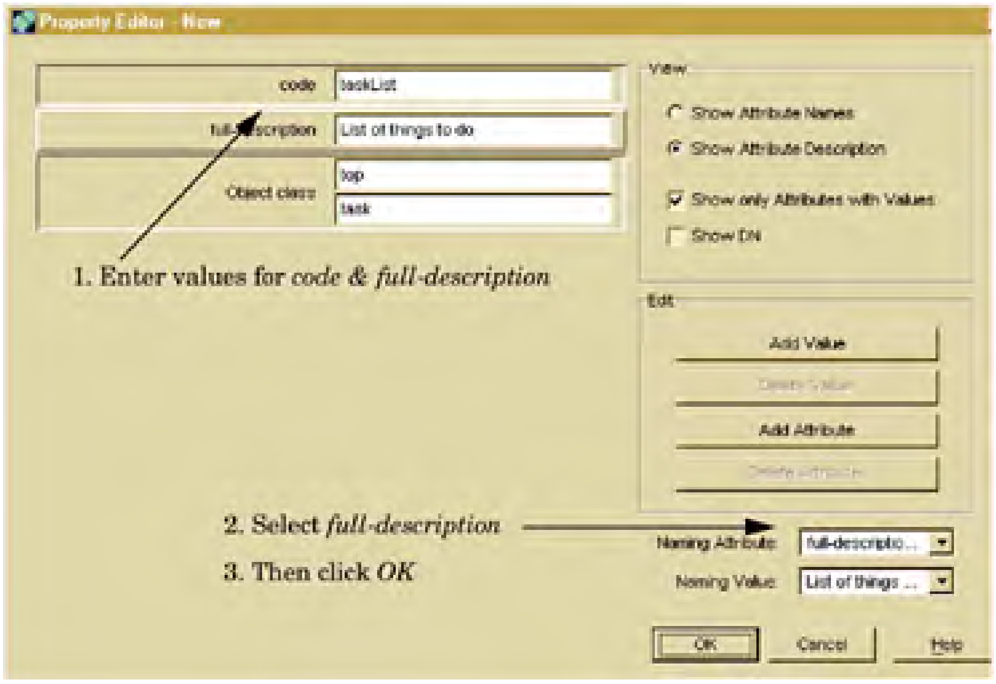


Figure 11.24. Entering values.



The new object name *List of things to do* shows up under *BookWork* (Figure 11.25). The next step is to add tasks under *List of things to do* (Figure 11.26). When you have added the first task, you can go through the process again and again for any number of added tasks (Figure 11.27). Then you can view all the tasks you have created for *List of things to do* under the *Directory* tab (Figure 11.28).

Figure 11.25. New object name—List of things to do.

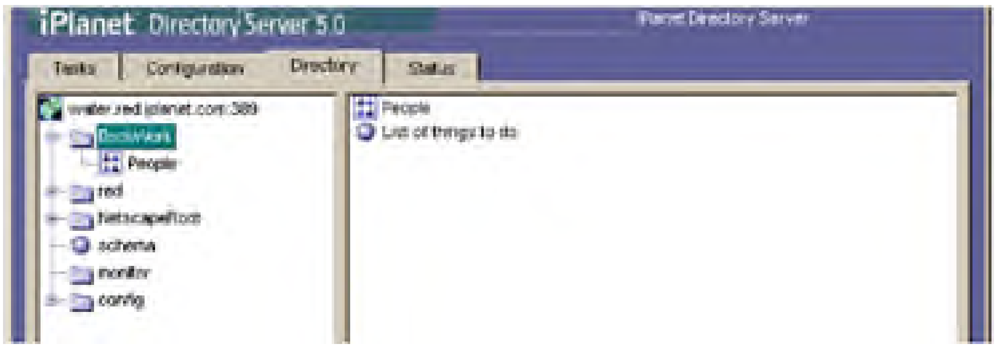
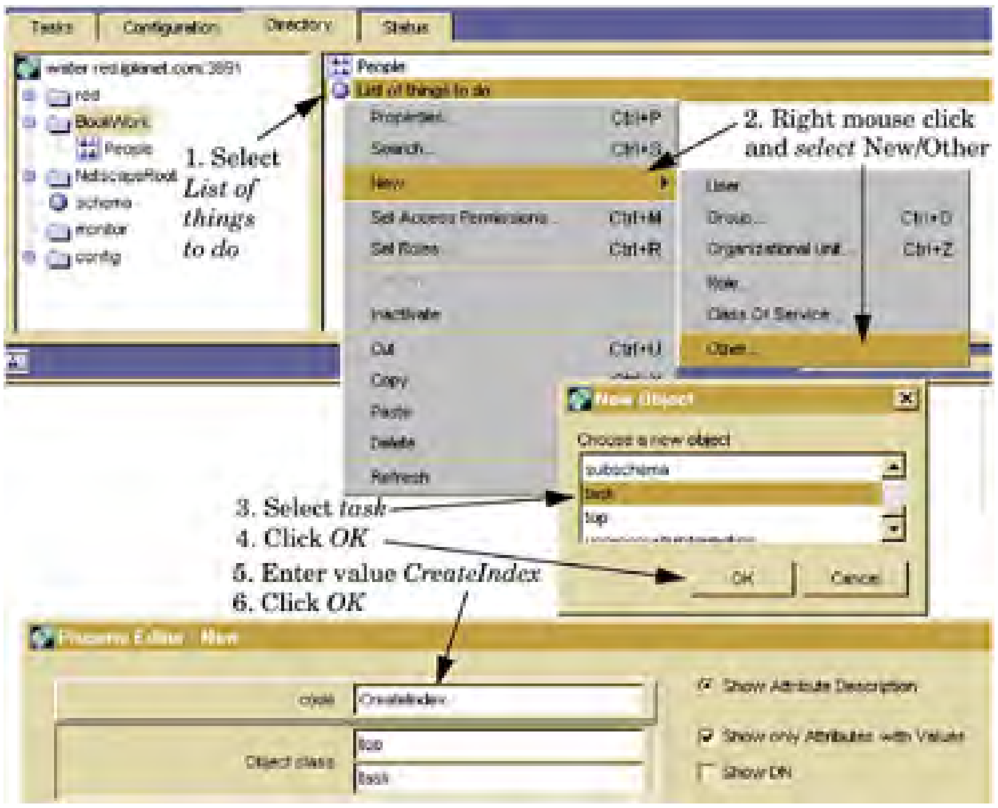


Figure 11.26. Add task objects under List of things to do.



cn: CreateIndex
Object class: top, task

Figure 11.27. Creating more tasks under List of things to do.

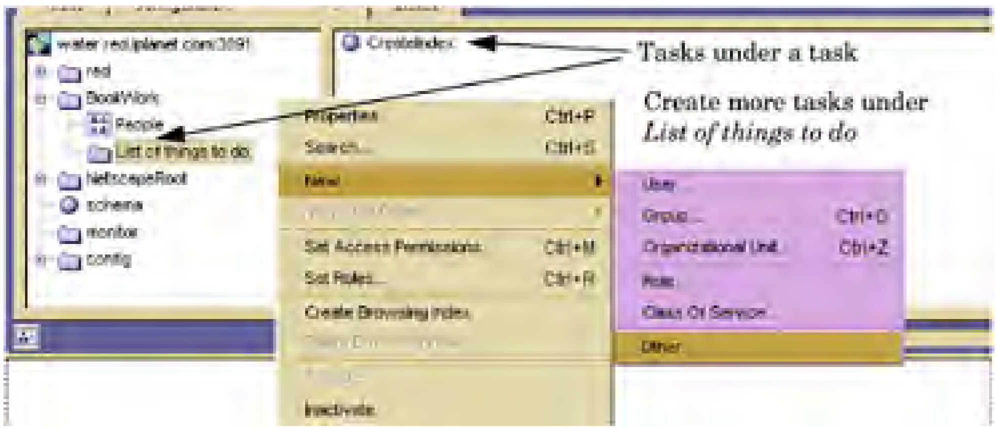
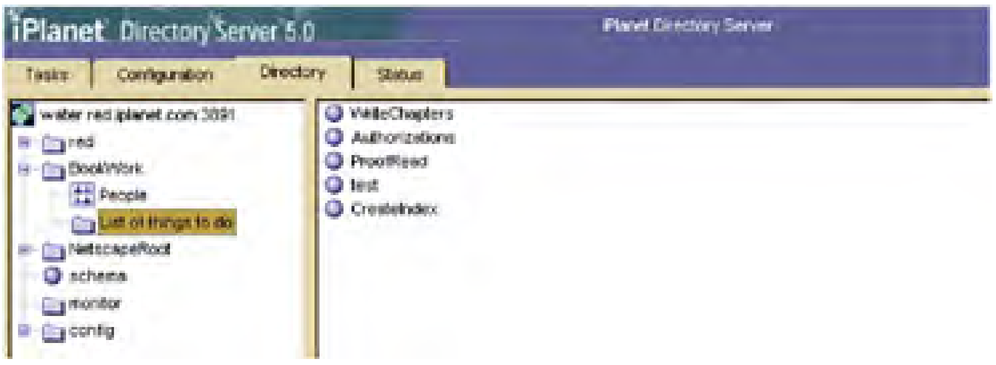


Figure 11.28. View the object named List of things to do.



LDAP Search Queries for the New Objects

The following list of object queries shows how to use LDAP search under the branch o=BookWork:

```
ldapsearch -1 -h "water.red.iplanet.com" -p 3891 -s one -
  b "o=BookWork" "objectclass=*" x
Search result:
dn: ou=People,o=BookWork
dn: full-description=List of things to do,o=BookWork
```

This example shows that it is best to keep names simple.

```
full-description=List of things to do
```

is an example of what not to do. It is difficult to type and remember. However, it is very descriptive.

Here is the list of tasks under *List of things to do*:

```
ldapsearch -1 -h "water.red.iplanet.com" -p 3891 -s one -
  b "full-description=List of things to do,o=BookWork"
  "objectclass=*" x
Search result:
dn: code=CreateIndex,full-description=List of things to
  do,o=BookWork
dn: code=WriteChapters,full-description=List of things
  to do,o=BookWork
dn: code=Authorizations,full-description=List of things
  to do,o=BookWork
dn: code=ProofRead,full-description=List of things to
  do,o=BookWork
dn: code=test,full-description=List of things to do,o=
  BookWork
```

The following query generates a sorted list. The sort parameter is -S code where code is an attribute.

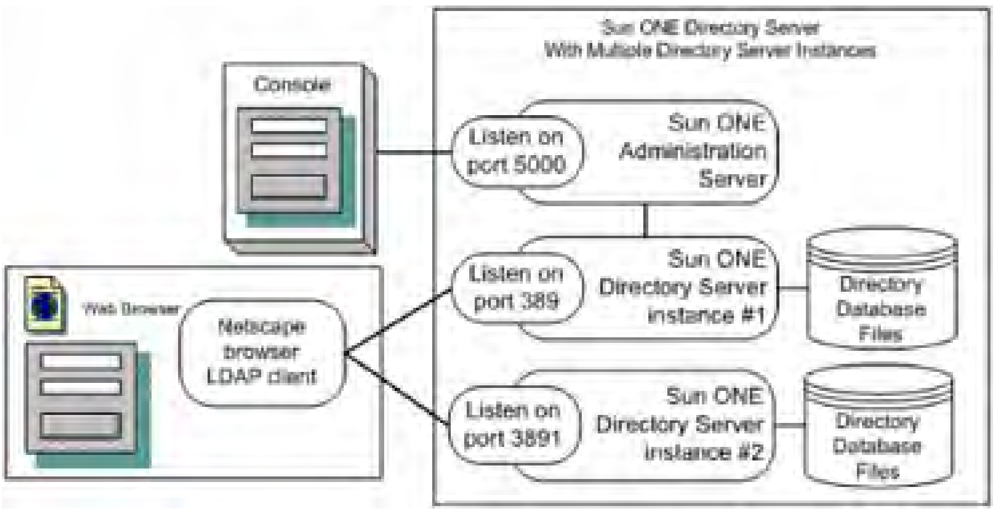
```
ldapsearch -1 -h "water.red.iplanet.com" -p 3891 -S code
  -s one -b "full-description=List of things to
  do,o=BookWork" "objectclass=*" x
Search result:
dn: code=Authorizations,full-description=List of things
  to do,o=BookWork
dn: code=CreateIndex,full-description=List of things to
  do,o=BookWork
dn: code=ProofRead,full-description=List of things to
  do,o=BookWork
dn: code=test,full-description=List of things to
  do,o=BookWork
dn: code=WriteChapters,full-description=List of things
  to do,o=BookWork
```

Having covered the directory schema, data structures, and directory structures within the directory, you should now be comfortable with the Console and using ldapsearch. The next section covers multiple directory instances.

Creating a New Directory Server Instance

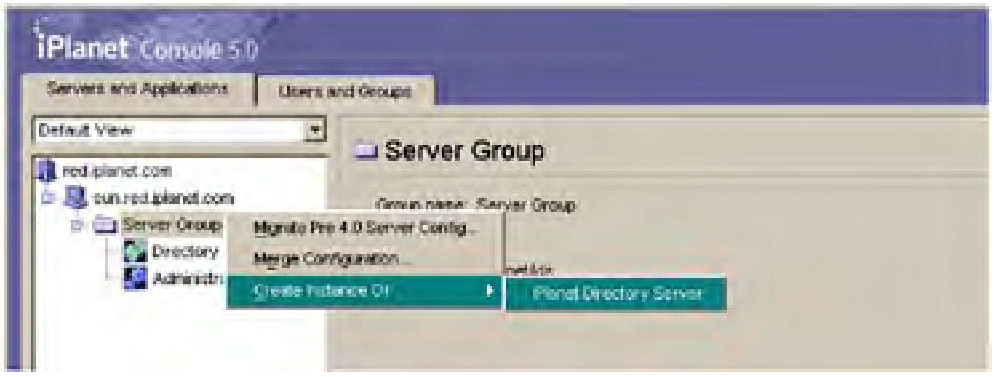
Like the Sun ONE Web Server, the Sun ONE Directory Server can be configured for multiple instances. Figure 11.29 shows a Netscape browser connected to each of the directory server instances.

Figure 11.29. Multiple directory server instances


Sun ONE Directory Server
With Multiple Directory Server Instances
Console
Listen on port 5000
Sun ONE Administration Server
Listen on port 389
Sun ONE Directory Server instance #1
Directory Database Files
Web Browser
Netscape browser LDAP client
Listen on port 3891
Sun ONE Directory Server instance #2
Directory Database Files

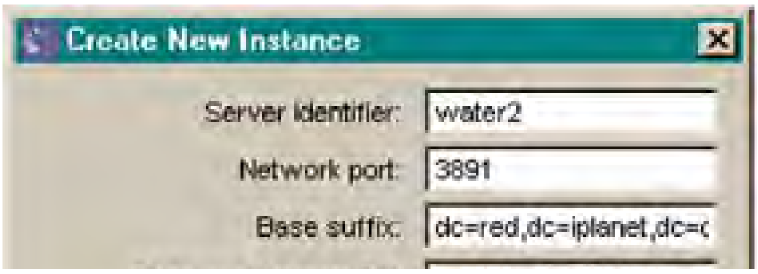
Using the Console, log in to the directory administration server using user ID *cn=Directory Manager*. Click *Server Group*. Then right mouse click to get the *Object* menu up. Select *Create Instance Of*, then *iPlanet Directory Server* as in Figure 11.30.

Figure 11.30. Menu option to create a new instance.



The *Create New Instance* window is displayed (Figure 11.31). Modify the port to be *3891,* enter the *Directory Manager Password* (we are using *netscape*), and then enter it again in the box labeled *Confirm password*. Then click *OK* to create the instance.

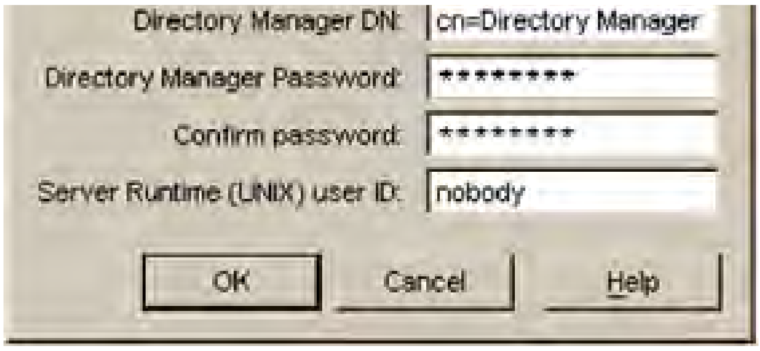
Figure 11.31. Creating a new instance.


Create New Instance

Server identifier: water2

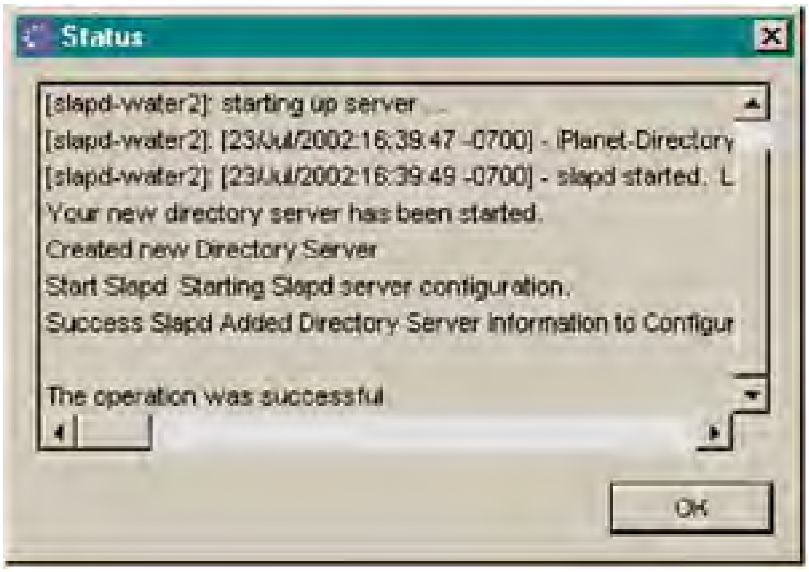
Network port: 3891

Base suffix: dc=red,dc=iplanet,dc=c


Directory Manager DN: cn=Directory Manager
Directory Manager Password: ********
Confirm password: ********
Server Runtime (UNIX) user ID: nobody

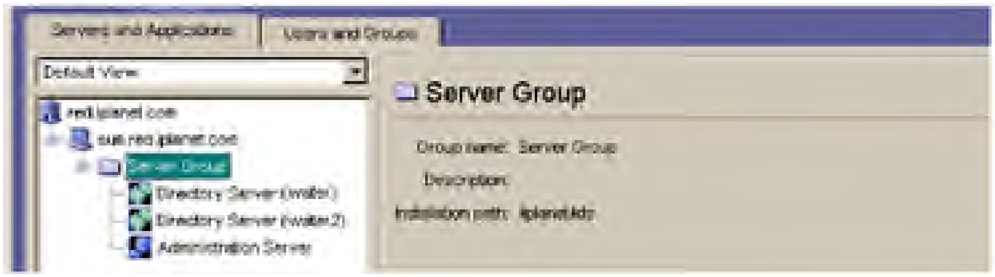
A *Status* window appears (Figure 11.32), and the instance is created and started up. Click *OK* and now you have two instances under the *Server Group* branch (Figure 11.33).

Figure 11.32. Status of the new instance.



```
[slapd-water2]: starting up server ...
[slapd-water2]: [23/Jul/2002:16:39:47 -0700] - iPlanet-Directory
[slapd-water2]: [23/Jul/2002:16:39:49 -0700] - slapd started.  L
Your new directory server has been started.
Created new Directory Server
Start Slapd  Starting Slapd server configuration.
Success Slapd Added Directory Server information to Configur

The operation was successful
```

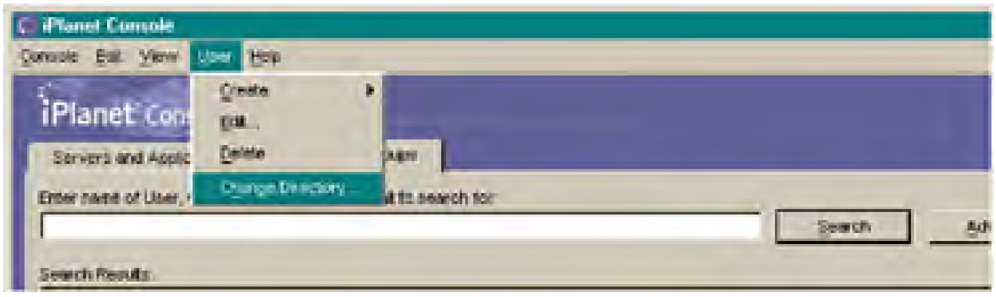
Figure 11.33. Instances under Server Group.



Group name: Server Group
Description:
Installation path: /iplanet/ds

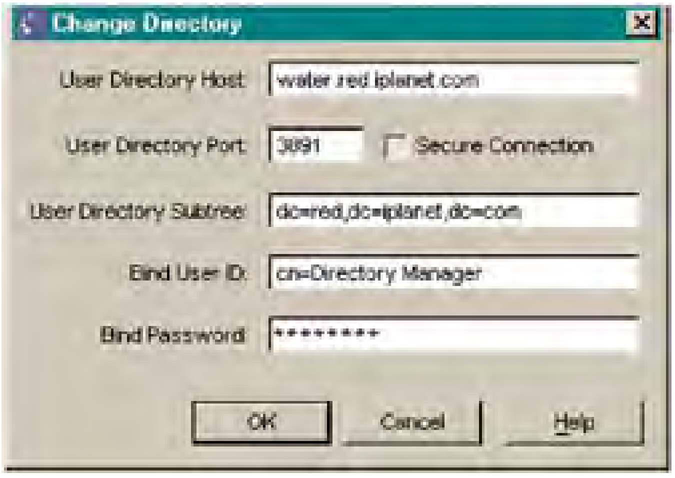
Click the *Users and Groups* tab, and then click the *Search* button (Figure 11.34). This option searches the default directory server, not the new server instance. Click the menu option *User* and select *Change Directory* (Figure 11.34).

Figure 11.34. Select the new directory instance.



In the *Change Directory* window (Figure 11.35) enter the new directory server port number *3891* and *cn=Directory Manager.* Then click *OK*. Note: This window allows you to select any directory server, regardless of whether the directory server is on this computer or on a remote computer.

Figure 11.35. Select a directory server.

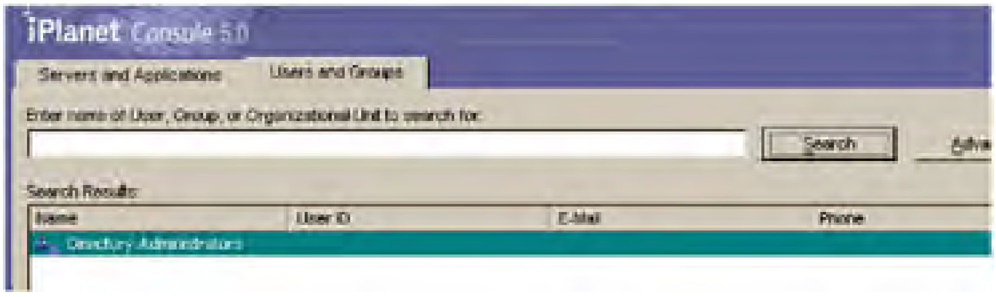


Change Directory

User Directory Host:	water.red.iplanet.com
User Directory Port:	3891 [ ] Secure Connection
User Directory Subtree:	dc=red,dc=iplanet,dc=com
Bind User ID:	cn=Directory Manager
Bind Password:	********

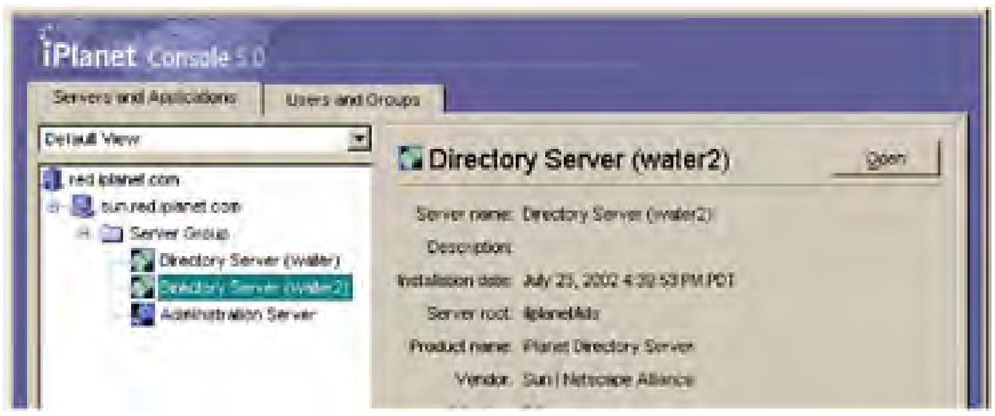
Back under the *Users and Groups* tab (Figure 11.36), click the *Search* button again. Now the results are from the new directory server. Click the *Servers and Applications* tab. When the window in Figure 11.37 opens, select the new directory instance named *Directory Server (water2)*. When you click on the *Open* button (on the right), the management window for the new directory server instance appears (Figure 11.38).

Figure 11.36. Searching the new directory server.



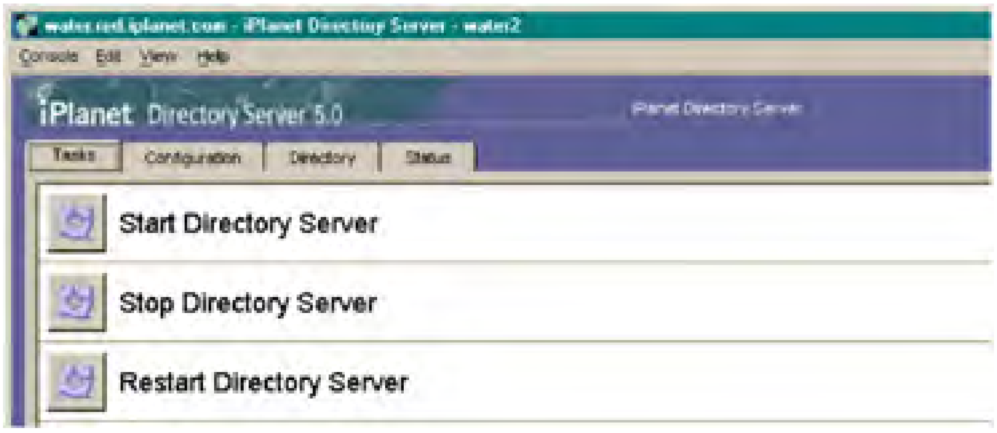
Name	User ID	E-Mail	Phone
Directory Administrators			

Figure 11.37. Select the new directory server instance for management.



Server name: Directory Server (water2)
Description:
Installation date: July 23, 2002 4:30:53 PM PDT
Server root: \iplanet\ds
Product name: iPlanet Directory Server
Vendor: Sun | Netscape Alliance

Figure 11.38. Management window for the new instance named water2.



Since you now have two directory server instances, I suggest that you create additional users in the new instance and look around the directory. This is your chance to practice what you have learned in this chapter.

Using your own hostname and directory suffix, from a Netscape web browser, try these queries:

```
ldap://water.red.iplanet.com/ou=people,dc=red,dc=
  iplanet,dc=com?x?sub

ldap://water.red.iplanet.com:3891/dc=red,dc=iplanet,dc=
  com?x?sub
```

The syntax is

```
ldap://<hostname>:<port>/<organization unit>,<suffix>
  ?x?sub
```

(<organization unit> is optional).

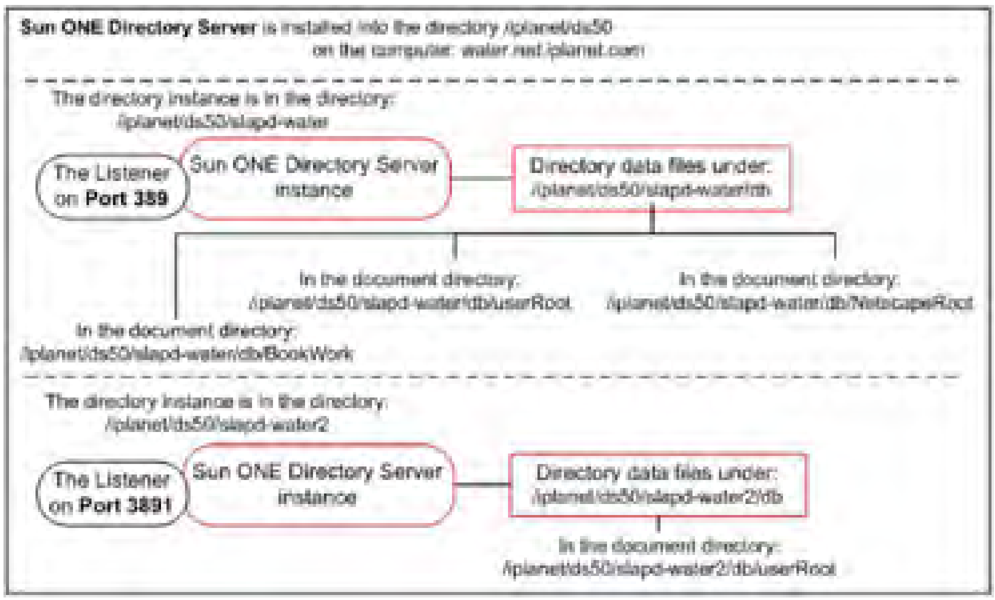
Summary

The Sun ONE Directory Server is a specialized hierarchical database system. From the very top, a Sun ONE Directory Server is ordered like this:

1. The directory server system has directory instances.
2. Within each directory instance are directory databases. Each of the databases has data files in a file directory.

Figure 11.39 illustrates the point that each instance has a data file directory. The instance listening on port 389 stores directory data under

Figure 11.39. Sun ONE Directory Server instance file directories.


Sun ONE Directory Server is installed into the directory /iplanet/ds50
on the computer: water.red.iplanet.com
The directory instance is in the directory:
/iplanet/ds50/slapd-water
The Listener on Port 389
Sun ONE Directory Server instance
Directory data files under:
/iplanet/ds50/slapd-water/db
In the document directory:
/iplanet/ds50/slapd-water/db/userRoot
In the document directory:
/iplanet/ds50/slapd-water/db/NetscapeRoot
In the document directory:
/iplanet/ds50/slapd-water/db/BookWork
The directory instance is in the directory:
/iplanet/ds50/slapd-water2
The Listener on Port 3891
Sun ONE Directory Server instance
Directory data files under:
/iplanet/ds50/slapd-water2/db
In the document directory:
/iplanet/ds50/slapd-water2/db/userRoot

/iplanet/ds50/slapd-water/db

The instance listening on port 3891 stores directory data under

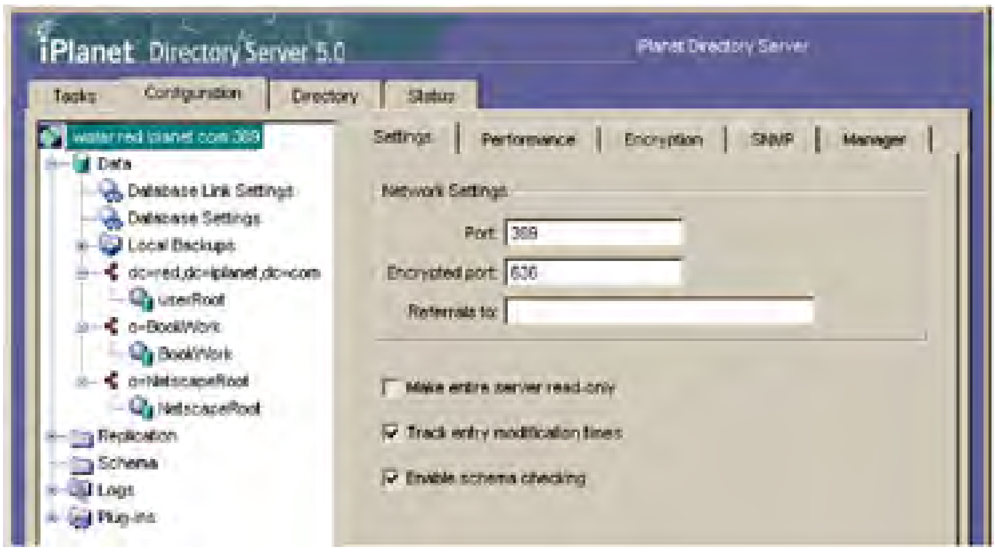
/iplanet/ds50/slapd-water2/db

Each directory instance has at least one root suffix. On the *Configuration* tab, directory instance *water.red.iplanet.com:389* has three root suffixes: *userRoot, BookWork,* and *NetscapeRoot*.

1. Within the suffix is a top objectClass, the root suffix.
2. Under the root suffix are organizational units (objectClasses).
3. Then, attributes are attached to an objectClass (an organization unit).
4. Each individual piece of data is attached to an attribute.

In Figure 11.40, *dc=red,dc=iplanet,dc=com*; *o=BookWork;* and *o=NetscapeRoot* are the three root suffixes for the instance *water.red.iplanet.com:389*.

Figure 11.40. The directory instance name water.red.iplanet.com and the root suffixes are listed under the Configuration tab.



Network Settings

Port: 389

Encrypted port: 636

Referrals to:

☐ Make entire server read-only

☑ Track entry modification times

☑ Enable schema checking

Under each directory suffix there is a root object that is the top of a tree of data. Figure 11.41 shows the root objects. Figure 11.42 shows the Sun ONE Directory Server system hierarchy.

Figure 11.41. Top-level root objects within each instance.

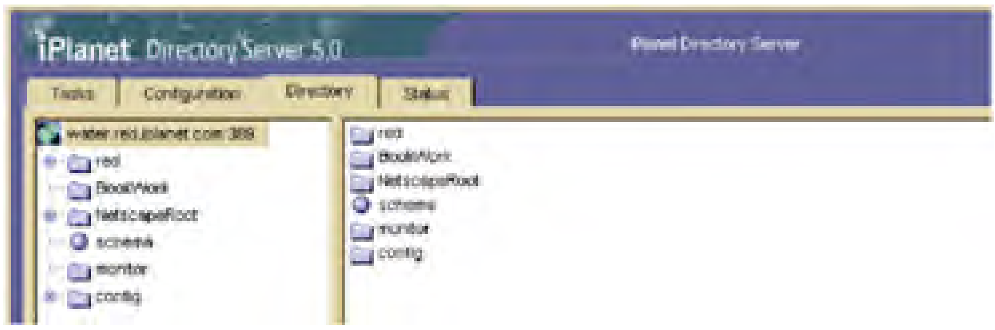
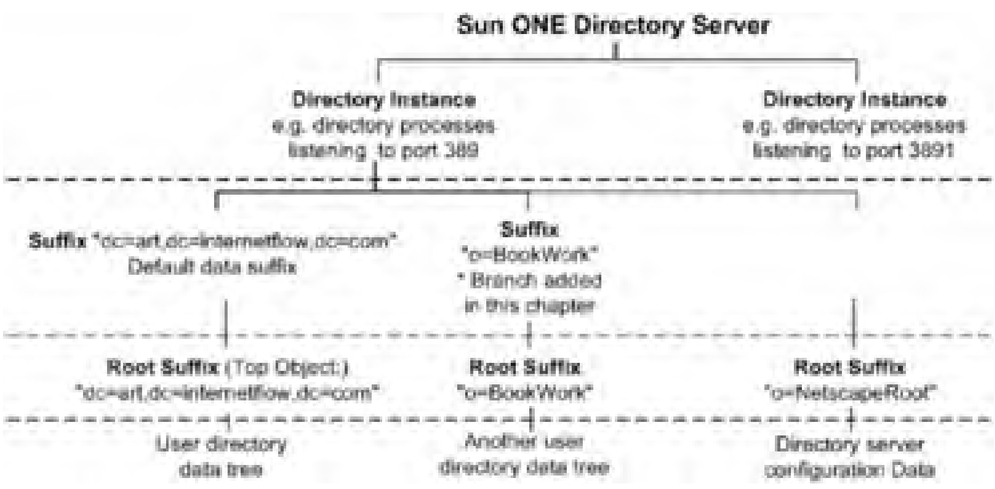


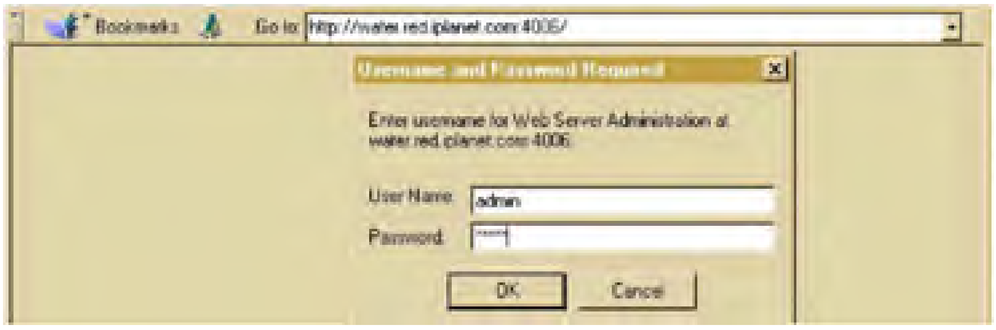
Figure 11.42. Top-level directory server hierarchy.



Chapter 12. Restricting Web Site Access to Privileged Directory Members

For the purposes of the chapter, the definition of a restricted access web site is a web site that prompts the surfer for a user ID and password. If the surfer fails to supply those items correctly, access to the website is denied. For example, when browsers connect to the Sun ONE Web Server Administration Server, the first thing they will see is a prompt for the administrator's user ID and password (Figure 12.1).

Figure 12.1. Prompt for User Name and Password.


Bookmarks
Go to: http://water.red.iplanet.com:4006/
Username and Password Required
OK
Cancel
Enter username for Web Server Administration at water.red.iplanet.com:4006:

User Name: admin
Password: *****

The steps for setting up restricted access to a web site are

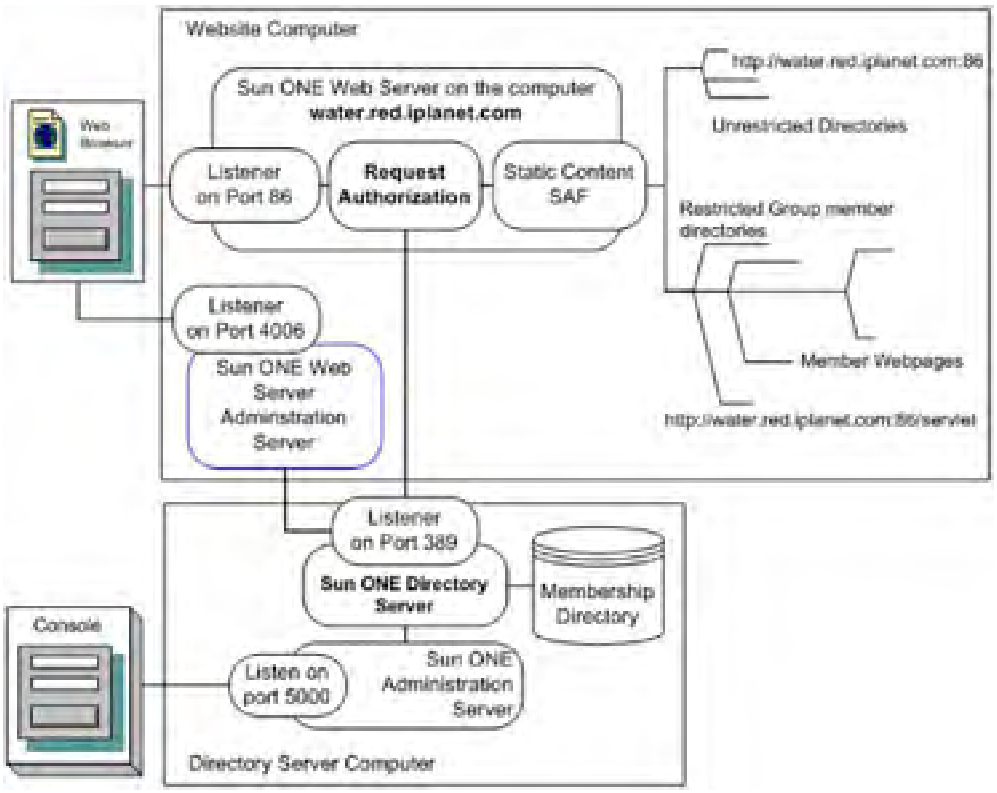
1. Link the Sun ONE Web Server to a Sun ONE Directory Server.
2. Add *new members* into the directory. Add a new group and put the *new members* into this *new group*.
3. Now restrict access to part of the web site by selecting a *file directory* under the web server's document directory. Access is restricted by defining an Access Control List (ACL). The ACL allows only the *new members* in the *new group* to access files in the restricted *file directory*.
4. Use a browser to test that access is restricted to the *file directory* described in the ACL. When prompted for a user ID and password, enter a *new member's* ID and password, and access is granted.

Connecting the Web Server to the Directory Server

Connecting the web server to the directory server includes linking the web administration server, along with all the web instances, to the directory server. Once the web server is configured to use the directory server, add new members and a group into the directory through the Sun ONE Web Server Administration Server or using the Console to communicate with the Sun ONE Administration Server. Once members have been added, through the web administration server, create ACLs to restrict access to the directories for directory members.

Figure 12.2 shows the server architecture. The web server instance validates user access using LDAP to communicate with the directory server. The web server administration server uses LDAP to maintain users in the directory.

Figure 12.2. Sun ONE Web Server and Sun ONE Directory Server components.


Website Computer
Web Browser
Sun ONE Web Server on the computer water.red.iplanet.com
Listener on Port 86
Request Authorization
Static Content SAF
http://water.red.iplanet.com:86
Unrestricted Directories
Restricted Group member directories
Member Webpages
http://water.red.iplanet.com:86/servlet
Listener on Port 4006
Sun ONE Web Server Administration Server
Listener on Port 389
Sun ONE Directory Server
Membership Directory
Console
Listen on port 5000
Sun ONE Administration Server
Directory Server Computer

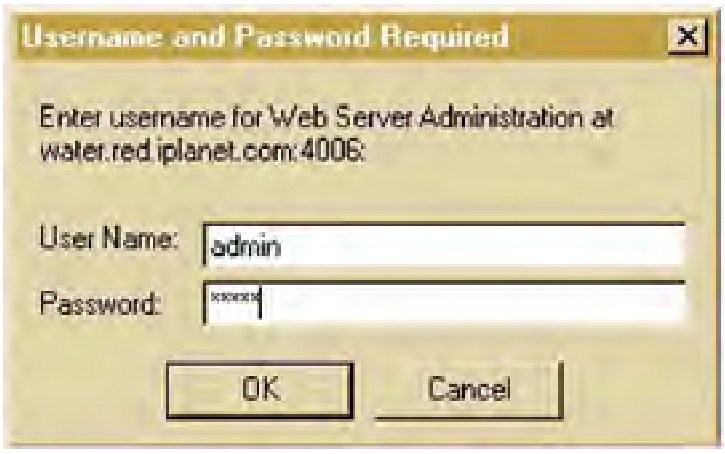
Before starting the lab work in this chapter, make sure the following servers are running:

- The Sun ONE Web Server Administration Server
- The Sun ONE Directory Server
- The Sun ONE Administration Server (installed with the directory server)

For instructions on starting and monitoring these servers, see previous chapters.

From a browser, log in to the Sun ONE Web Server Administration Server, using *admin* for both user ID and password and then clicking *OK* (Figure 12.3). Our sample URL is: *http://water.red.iplanet.com:4006*.

Figure 12.3. User Name and Password (admin/admin).


Username and Password Required

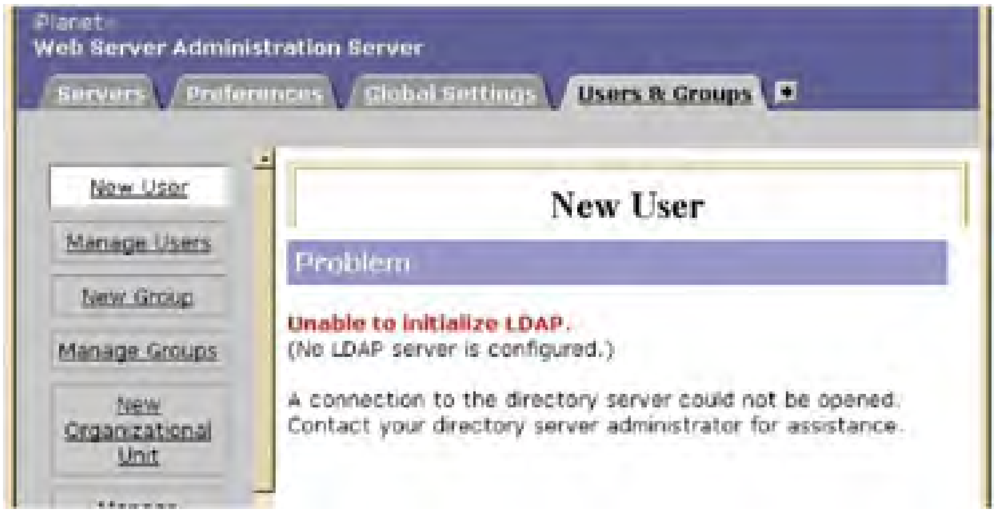
Enter username for Web Server Administration at water.red.iplanet.com:4006:

User Name: admin

Password: *****

Once logged in, click on the *Users & Groups* tab (Figure 12.4). You will see the warning message *Unable to initialize LDAP. (No LDAP server is configured.)* This is because a link to a directory server is not yet set up.

Figure 12.4. Warning message received because link not set up.



New User

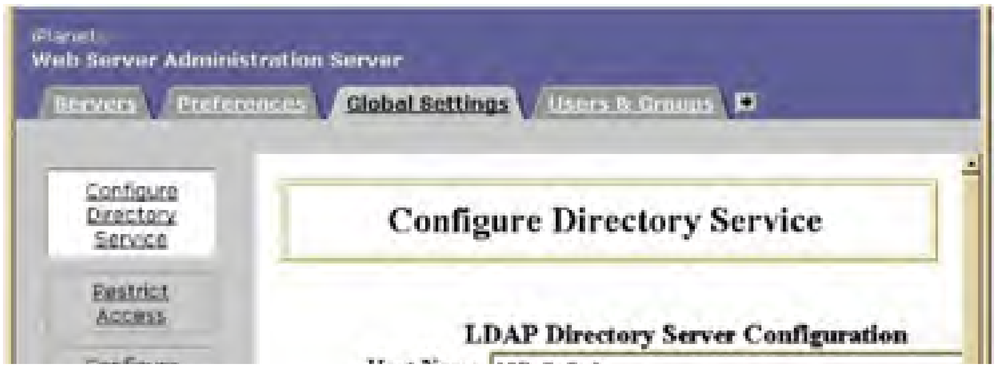
Problem

Unable to initialize LDAP.
(No LDAP server is configured.)

A connection to the directory server could not be opened. Contact your directory server administrator for assistance.

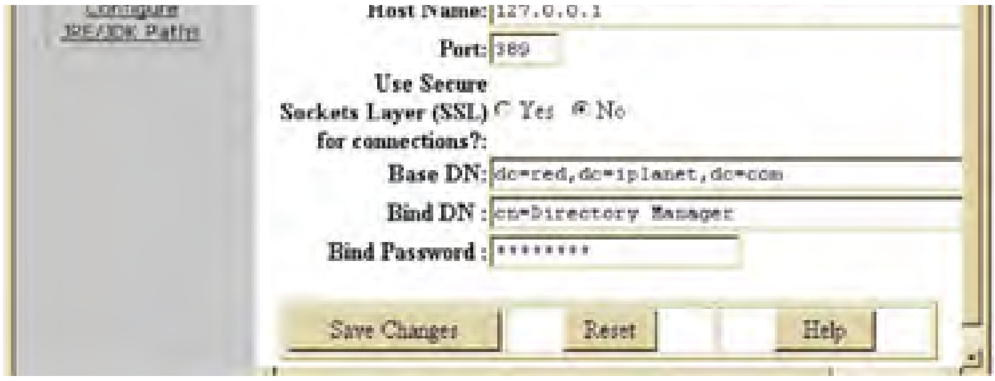
To enter the directory server information, click on *Global Settings* tab and the administration web page appears (Figure 12.5). Enter the information required to connect to the directory server. Use *127.0.0.1* for the *Host Name*, only if the directory server is on the same computer as the web server. If your directory server is on another computer, enter its IP address or hostname in the *Host Name* field. Enter the port number of the directory server instance; *389* is the default and is most likely correct. The *Bind DN* is a directory server user with access to the directory database. I am using *cn=Directory Manager* which is the administration user. The *Bind Password* is the password for *cn=Directory Manager*. In this book, I have used the password *netscape*. The *Base DN* is the starting branch node used to search for users in the directory. I am using the root suffix, or the top node. This way, the web server has access to all the users in the directory.

Figure 12.5. Parameters to link the web server to the directory server.



Configure Directory Service

LDAP Directory Server Configuration



Configure JRE/JDK Paths

Host Name: 127.0.0.1

Port: 389

Use Secure Sockets Layer (SSL) for connections?: Yes [✓ No]

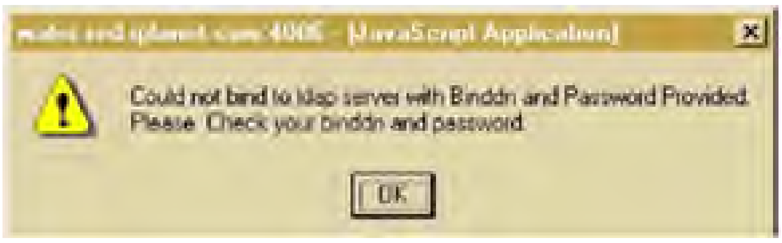
Base DN: dc=red,dc=iplanet,dc=com

Bind DN : cn=Directory Manager

Bind Password : *********

Once the values are entered on the administration web page (Figure 12.5), click *Save Changes*. If you enter the incorrect information, an error message that will tell you that the connection to the LDAP server failed (see Figure 12.6), or else, under *Configure Directory Service* (see Figure 12.5), the message *configuration has been updated* appears. To apply the changes, restart the web server administration server.

Figure 12.6. Error message: connection failure.



[illegible] - [JavaScript Application]

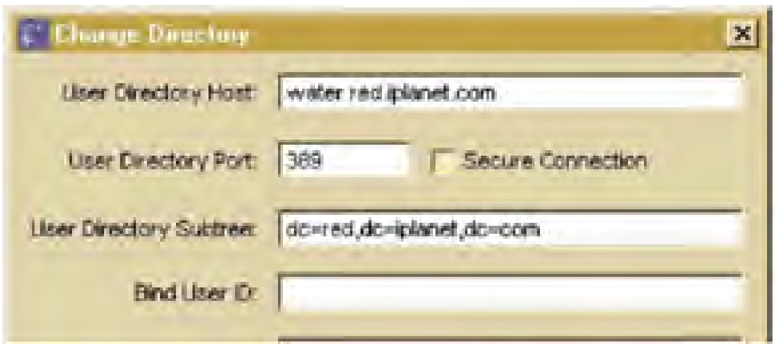
Could not bind to ldap server with Binddn and Password Provided.
Please Check your binddn and password

If you enter the incorrect information, the error message in Figure 12.6 appears. If the error message does appear, change the information (as in Figure 12.5) until no error message appears.

Note: If you have any questions regarding the values in Figure 12.5, use the Console to confirm those values.

1. From the Console, log in to the administration server using *cn=Directory Manager,* password *netscape*. This user is the *Bind DN* in Figure 12.5.
2. Click on the *Users and Groups* tab.
3. On the top menu, select *User/Change Directory*. This displays the rest of the required connection information as shown in Figure 12.7.

Figure 12.7. LDAP directory connection information.



Change Directory

User Directory Host: walter.red.iplanet.com

User Directory Port: 389 ☐ Secure Connection

User Directory Subtree: dc=red,dc=iplanet,dc=com

Bind User D:

The *User Directory Host* in Figure 12.7 is the *Host Name* value in Figure 12.5. The *User Directory Subtree* is the root suffix that was set up when the directory server was installed. This is the *Base DN* value in Figure 12.5.

Now your Sun ONE Web Server is connected to your Sun ONE Directory Server!

Managing Directory Members and Groups Using a Web Browser

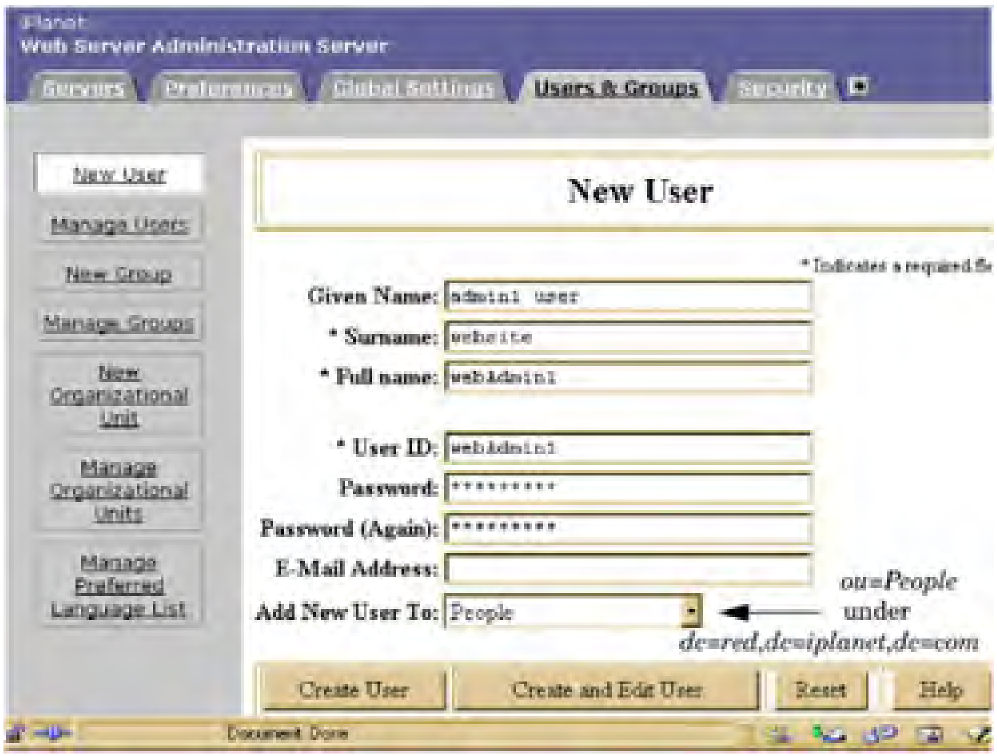
There are two tools for managing directory members: the Sun ONE Web Server Administration Server and the Console. Since member management with the Console was covered in a previous chapter, this chapter uses a browser to manage users through the Sun ONE Web Server Administration Server.

In this section, three members are added and are put into a group called webAdministrators. The three new members are named *webAdmin1*, *webAdmin2,* and *webAdmin3*.

Adding Directory Members

From a browser, connect to the web server administration server main web page (Figure 12.8). Click the *Users & Groups* tab. The left-hand menu shows all the options for managing members as well as managing groups and organizational units. This is a very flexible system for maintaining members and their related directory information.

Figure 12.8. Creating web site administrators.



New User

* Indicates a required fi

Given Name:	admin1 user
* Surname:	website
* Full name:	webAdmin1
* User ID:	webAdmin1
Password:	*********
Password (Again):	*********
E-Mail Address:	
Add New User To:	People

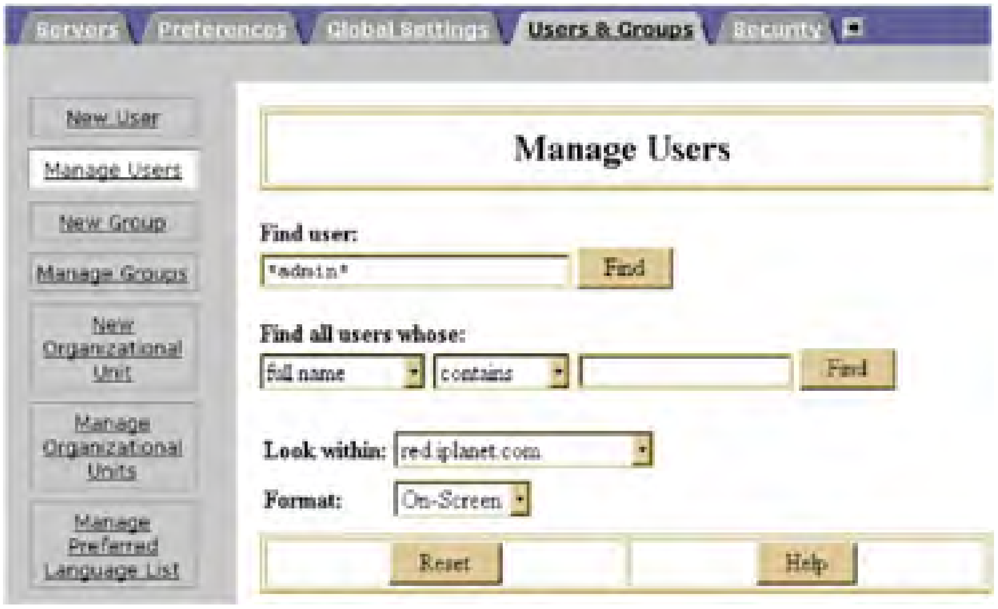
ou=People under dc=red,dc=iplanet,dc=com

Use this web page to create the three web site administrators, *webAdmin1*, *webAdmin2* and *webAdmin3*. Enter the information as in Figure 12.8 and click the *Create User* button. Note: Another name for full name is *common name* or *cn*, as in *cn=webAdmin1*.

Searching for Directory Members

In the window in Figure 12.9, click *Manage Users*, enter **admin** (part of a username), and click *Find*. A list of the new members that you just entered is displayed in the next window (Figure 12.10).

Figure 12.9. Finding user *admin*.



Manage Users

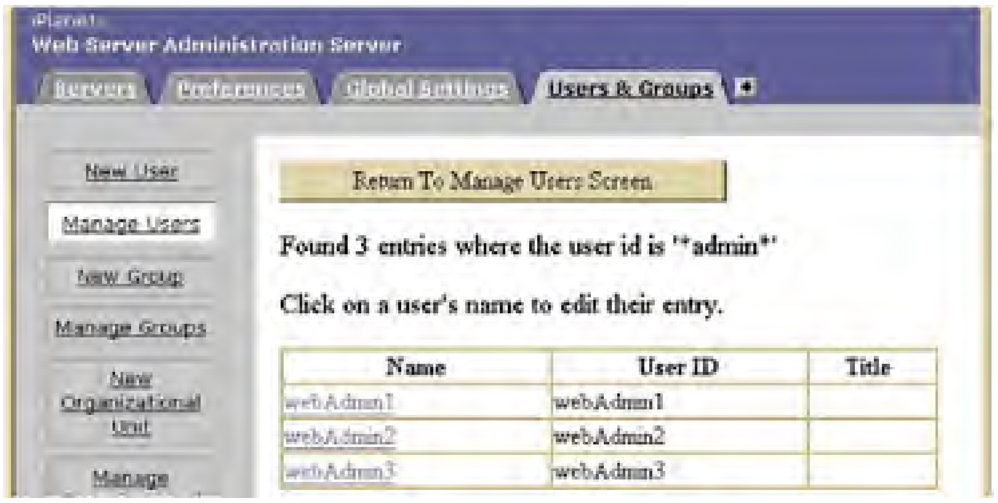
Find user: *admin*

Find all users whose: full name | contains

Look within: red.iplanet.com

Format: On-Screen

Figure 12.10. Results of search for *admin*.



Found 3 entries where the user id is "*admin*"

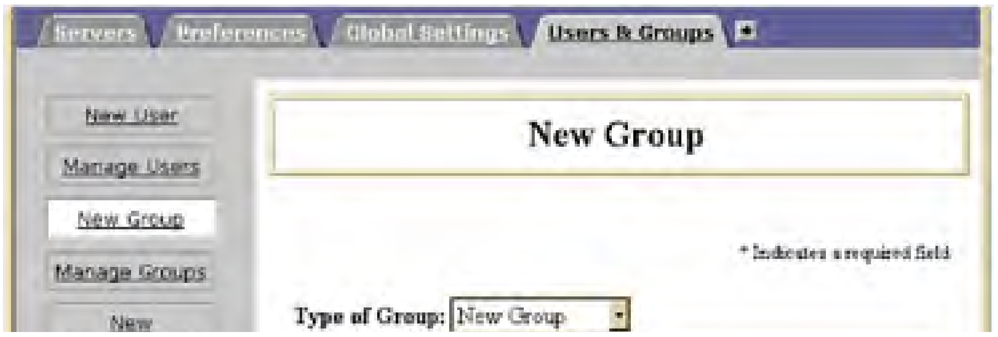
Click on a user's name to edit their entry.

Name	User ID	Title
webAdmin1	webAdmin1	
webAdmin2	webAdmin2	
webAdmin3	webAdmin3	

Adding a Group

A group is needed for the web site administrator members. Click the menu option *New Group* (see Figure 12.11) and create the group. Enter the information in the boxes and then click the *Create Group* button. The group is stored in the directory under ou=Groups, dc=red,dc=iplanet,dc=com.

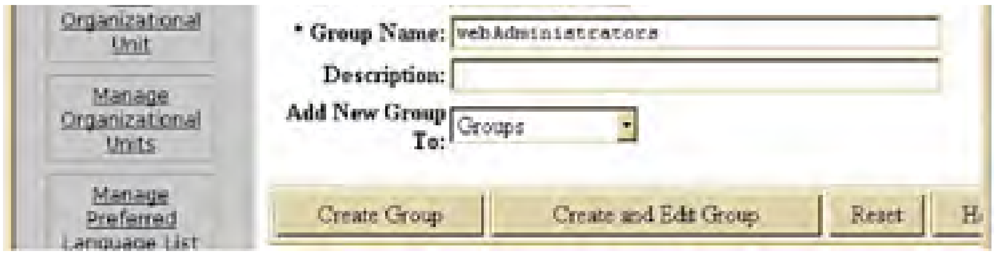
Figure 12.11. Create a web site administrator group.



New Group

* Indicates a required field

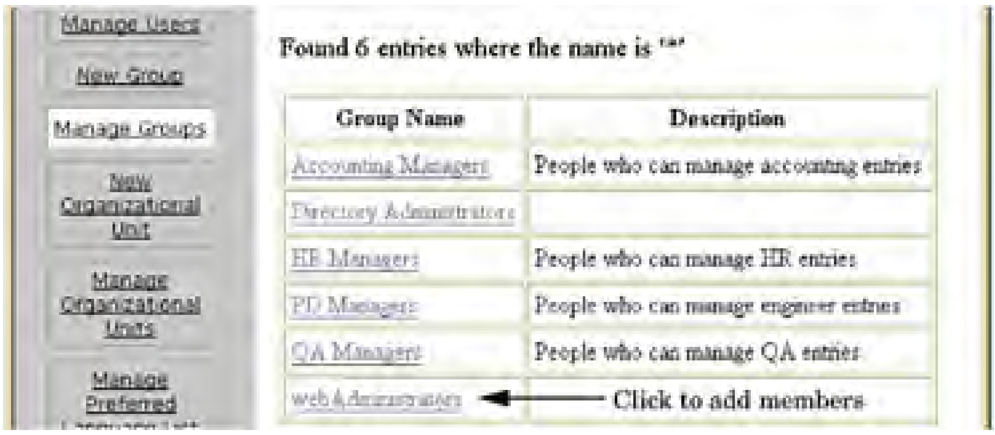
Type of Group: New Group



* Group Name: webAdministrators
Description:
Add New Group To: Groups

Now click *Manage Groups* directly beneath *New Group*; then click *Find*. When the list is displayed (see Figure 12.12), click *webAdministrators*, and the maintenance web page for the *webAdministrators* group opens (Figure 12.13).

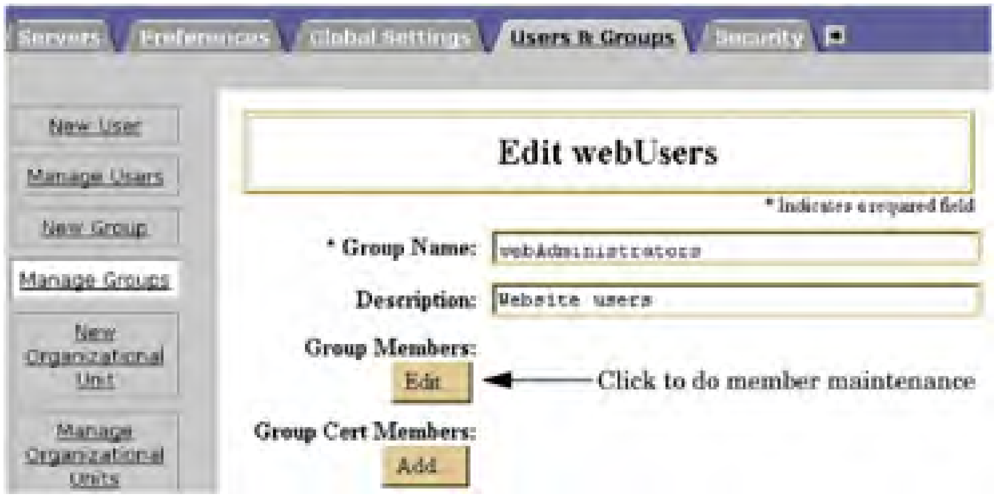
Figure 12.12. Groups in the directory.



Found 6 entries where the name is "*"

Group Name	Description
Accounting Managers	People who can manage accounting entries
Directory Administrators	
HR Managers	People who can manage HR entries
PD Managers	People who can manage engineer entries
QA Managers	People who can manage QA entries
webAdministrators ← Click to add members	

Figure 12.13. webAdministrators group maintenance web page.



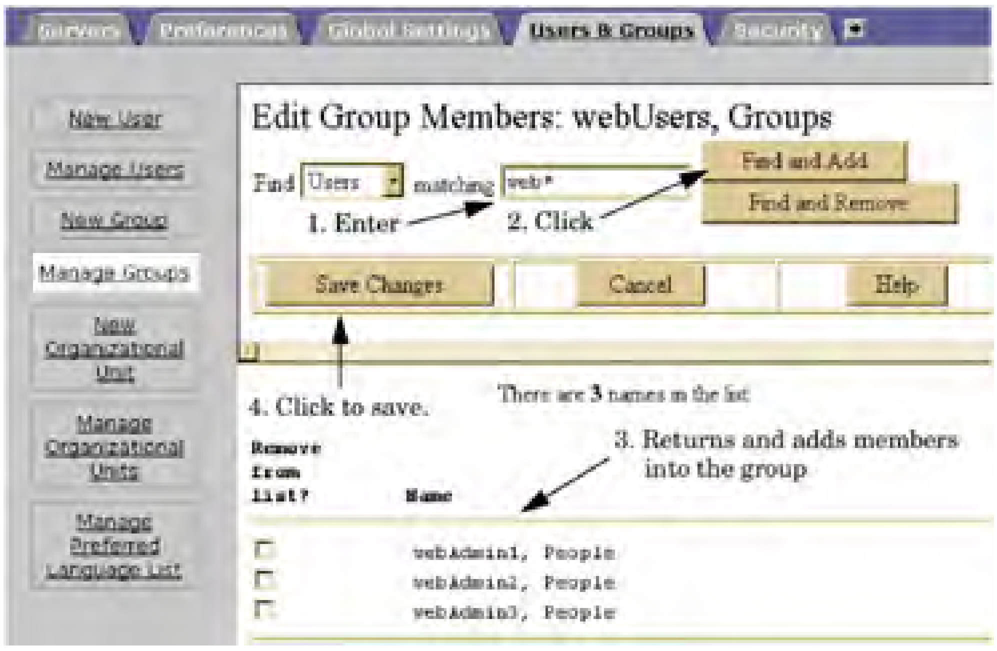
Edit webUsers

* Indicates a required field

* Group Name: webAdministrators
Description: Website users
Group Members: Edit ← Click to do member maintenance
Group Cert Members: Add

Click the *Group Members Edit* button to add members to the group. On the *Edit Group Members* page (Figure 12.14), enter *web** into the *matching* field and click the *Find and Add* button. View the list that comes up and click *Save Changes* to save this list into the group.

Figure 12.14. Saving the members into the group.



Edit Group Members: webUsers, Groups

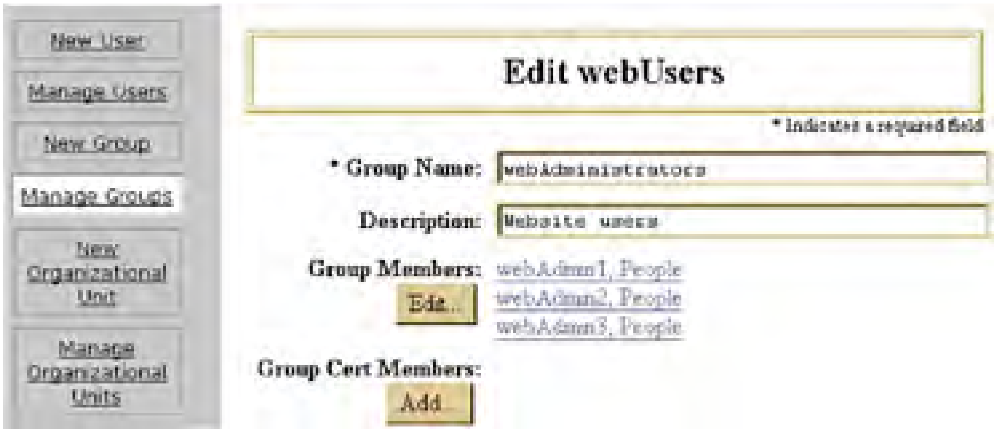
Find Users matching web*

There are 3 names in the list

Remove from list?	Name
☐	webAdmin1, People
☐	webAdmin2, People
☐	webAdmin3, People

Upon your return to the *webAdministrators* management web page, you see the list of members that are now in the *webAdministrators* group (Figure 12.15).

Figure 12.15. List of members in new group.



Edit webUsers

* Indicates a required field

* Group Name: webAdministrators

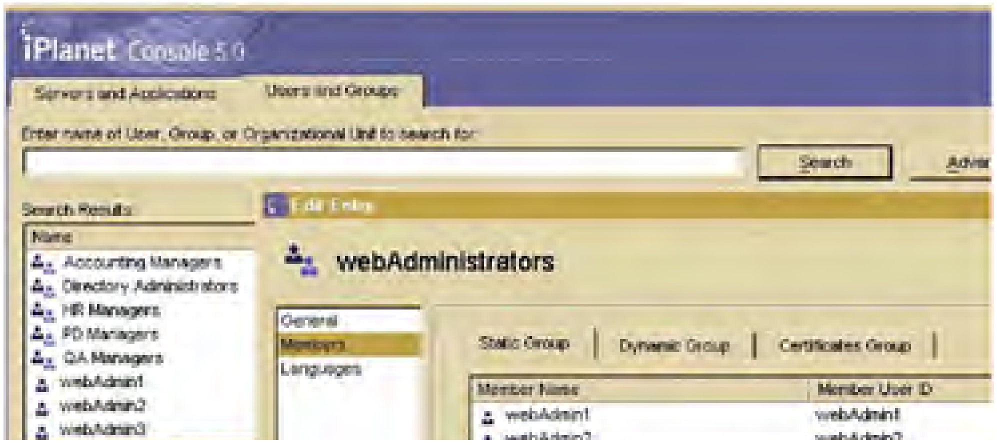
Description: Website users

Group Members: webAdmin1, People
webAdmin2, People
webAdmin3, People

Group Cert Members:

From the Console, to see the list of members in *webAdministrators* (refer to Figure 12.16)

Figure 12.16. Viewing list from the Console.

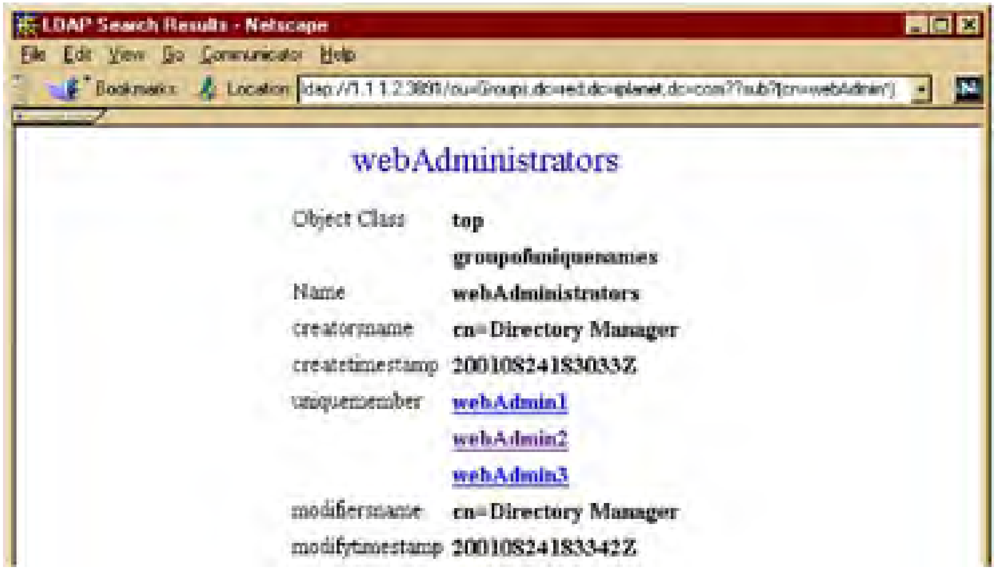


Member Name	Member User ID
webAdmin1	webAdmin1
webAdmin2	webAdmin2

1. Log in to the directory server from the Console.
2. Click the *Users and Groups* tab.
3. Click *Search.*
4. Double-click on *webAdministrators.*
5. Click *Members*.

A third method of viewing the members is to use a Netscape web browser (Figure 12.17):

Figure 12.17. Viewing members from Netscape browser.


LDAP Search Results - Netscape
File Edit View Go Communicator Help
Bookmarks Location

ldap://1.1.1.2:389/ou=Groups,dc=red,dc=iplanet,dc=com??sub?(cn=webAdmin*)

webAdministrators

Object Class	top
	groupofuniquenames
Name	webAdministrators
creatorsname	cn=Directory Manager
createtimestamp	20010824183033Z
uniquemember	webAdmin1
	webAdmin2
	webAdmin3
modifiersname	cn=Directory Manager
modifytimestamp	20010824183342Z

```
ldap://<host>:<port>/ou=Groups,dc=red,dc=iplanet,dc=com
  ??sub?(cn=webAd*)
```

For example:

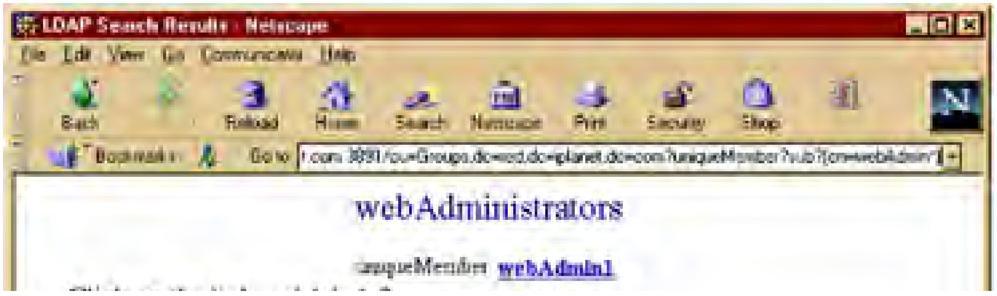
```
ldap://water.red.iplanet.com/ou=Groups,dc=red,dc=
  iplanet,dc=com??sub?(cn=webAd*)
```

Or, for a simpler list, use

```
ldap://<host>:<port>/ou=Groups,dc=red,dc=iplanet,dc=com?
  uniqueMember?sub?(cn=webAd*)
```

and see the result in Figure 12.18. Click on *webAdmin2* and then the information for *webAdmin2* comes up (Figure 12.19).

Figure 12.18. A simpler list of members (Netscape browser).

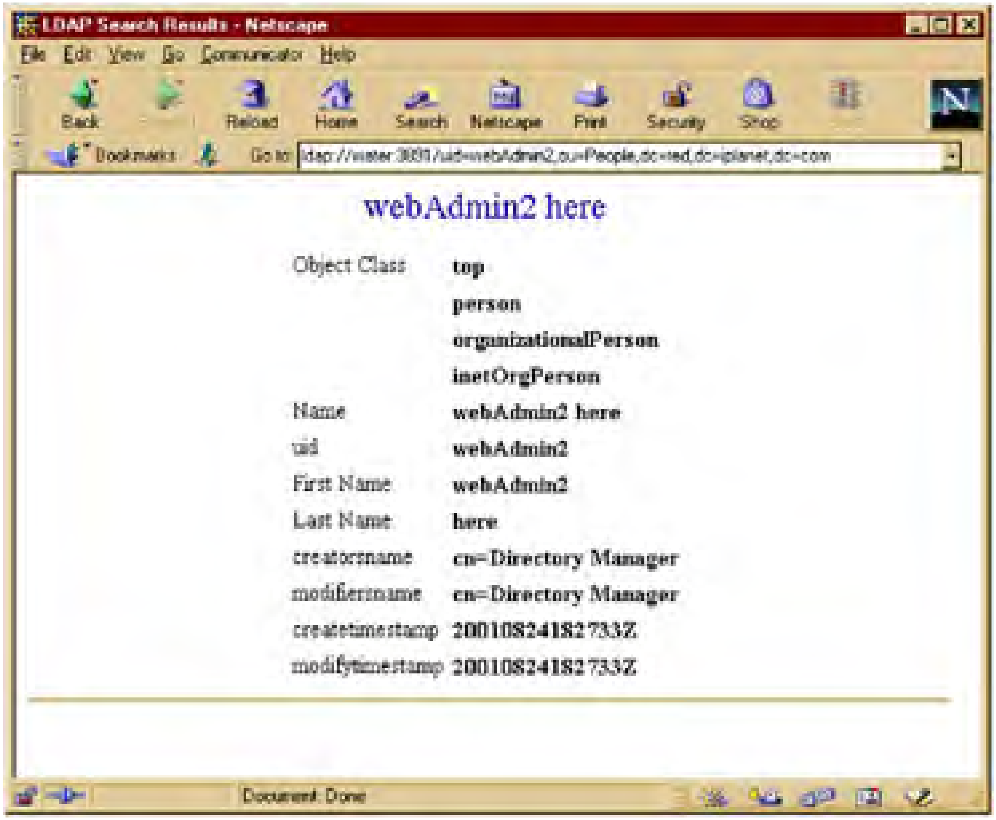

LDAP Search Results - Netscape
File Edit View Go Communicator Help
Back
Reload
Home
Search
Netscape
Print
Security
Shop
Bookmarks
Go to

.com:389/ou=Groups,dc=red,dc=iplanet,dc=com?uniqueMember?sub?(cn=webAdmin*)

webAdministrators

uniqueMember webAdmin1

Figure 12.19. Information for webAdmin2 member.


LDAP Search Results - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security Shop
Bookmarks
Go to: ldap://water:3891/uid=webAdmin2,ou=People,dc=red,dc=iplanet,dc=com
Document: Done

webAdmin2 here

Object Class	top
	person
	organizationalPerson
	inetOrgPerson
Name	webAdmin2 here
uid	webAdmin2
First Name	webAdmin2
Last Name	here
creatorsname	cn=Directory Manager
modifiersname	cn=Directory Manager
createtimestamp	20010824182733Z
modifytimestamp	20010824182733Z

Creating an Access Control List

In our example, we create an ACL to control access to the servlet directory. Since ACLs are for specific web server instances, select the servlet web instance from the main web administration web page.

If you have not done so in a previous chapter, create a servlet directory under the web server document directory and put the HelloWorldServlet program in the directory. For example, in UNIX:

```
mkdir /iplanet/iws60/docs/servlet
cd /iplanet/iws60/docs/servlet
cp /iplanet/iws60/plugins/samples/servlets/HelloWorld/
   *.class .
```

In WinNT, use the file explorer program (Figure 12.20). Then test the servlet using a web browser. Figure 12.21 shows the welcome page.

Figure 12.20. Using Windows NT file explorer.

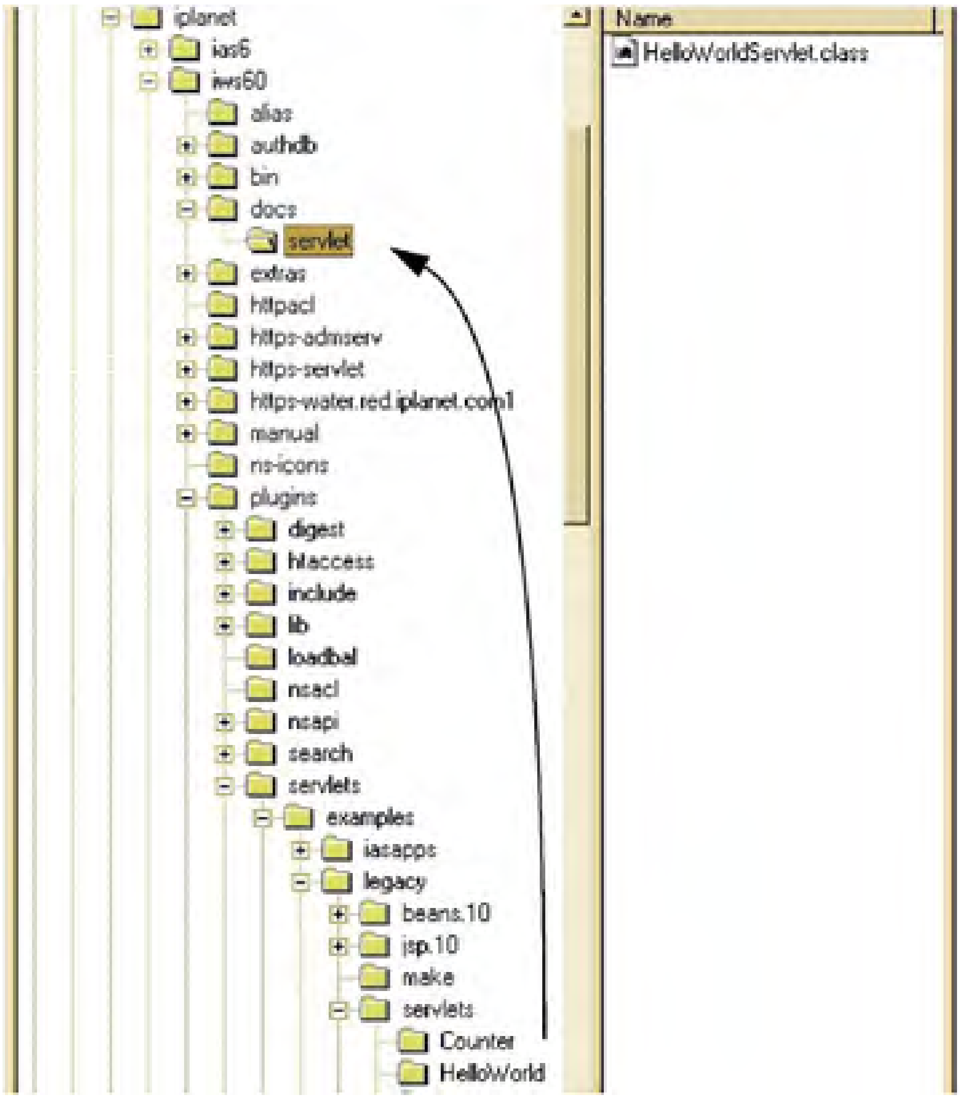
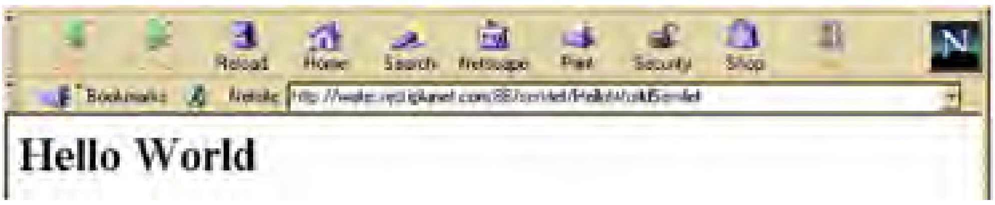


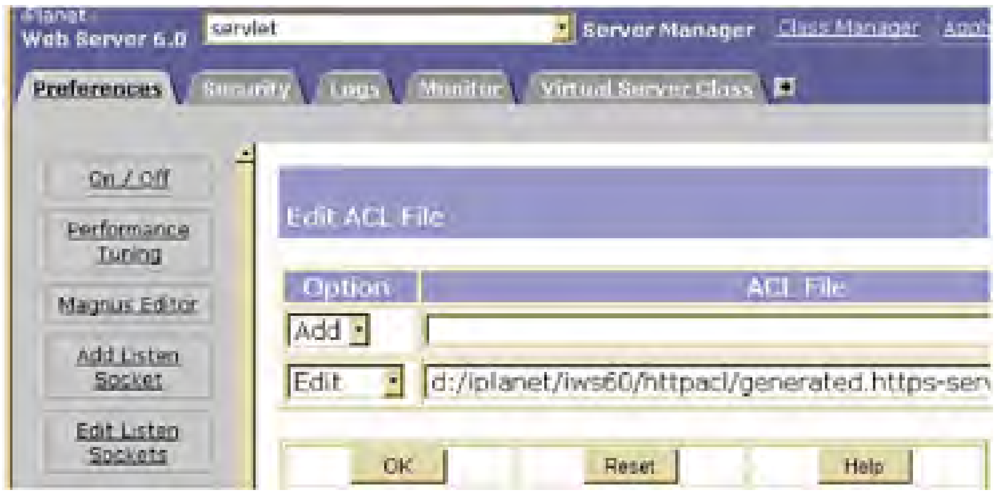
Figure 12.21. Welcome page for Hello World.



Hello World

From main web page of the web server administrator (Figure 12.22), under the *Preferences* tab, click *Restrict Access*. Then click *OK*. The message prompt in Figure 12.23 will appear. Click *OK*.

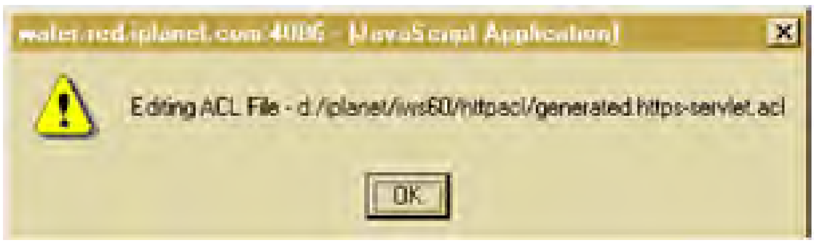
Figure 12.22. Choose to edit the ACL file.



Edit ACL File

Option	ACL File
Add	
Edit	d:/iplanet/iws60/httpacl/generated.https-serv

Figure 12.23. Message prompt for editing an ACL file.

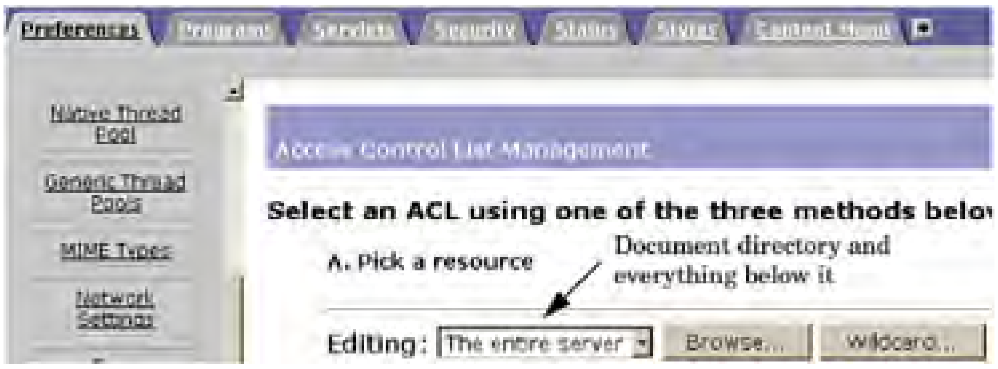


walet.red.iplanet.com:4086 - [JavaScript Application]

Editing ACL File - d:/iplanet/iws60/httpacl/generated.https-servlet.acl

In the *Access Control List Management* window (Figure 12.24), under *Pick a resource* you'll see the value *The entire server.* This means all directories under the web server document directory. Click on *Browse*.

Figure 12.24. Access Control List Management window.



Access Control List Management

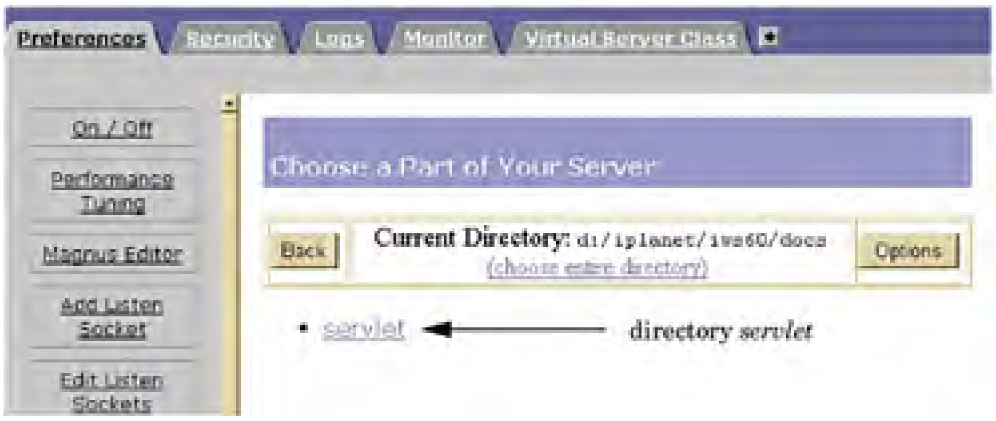
Select an ACL using one of the three methods belo

A. Pick a resource

Editing: The entire server

This shows the file directory, under the top web site docs directory (Figure 12.25). Click on *servlet* to select this file directory for access restriction.

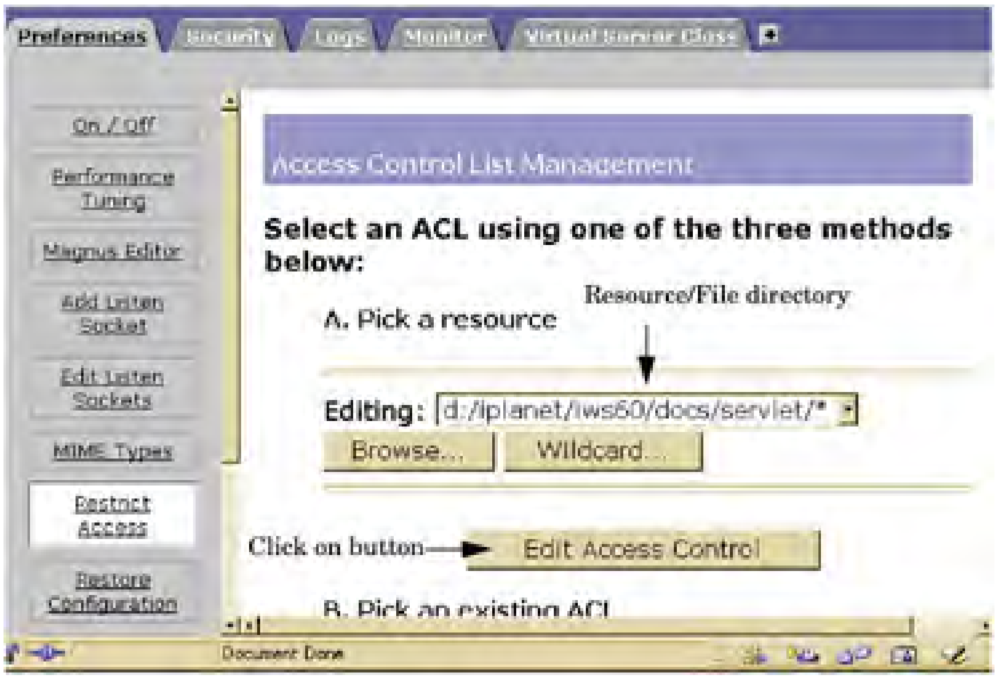
Figure 12.25. Selecting a file directory for restricted access.



Current Directory: d:/iplanet/iws60/docs
(choose entire directory)

To create an ACL, click on *Edit Access Control* (see Figure 12.26).

Figure 12.26. Create an ACL.



Select an ACL using one of the three methods below:

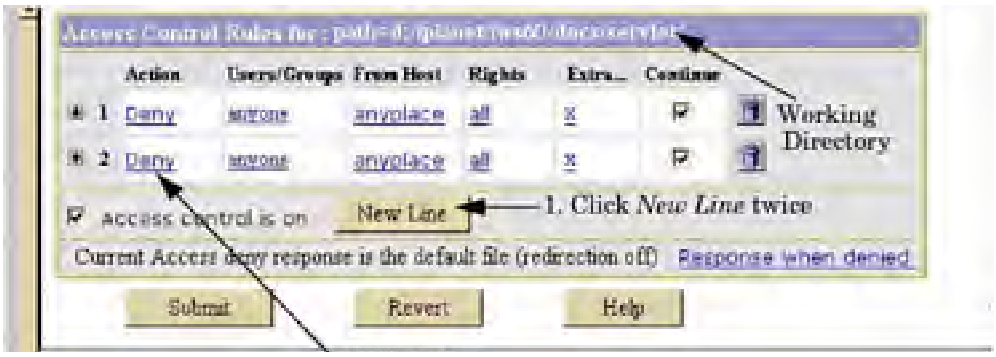
A. Pick a resource

Editing: d:/iplanet/iws60/docs/servlet/*

B. Pick an existing ACL

In Figure 12.27, there are two *Access Control Rules* on the top frame. The default (in Figure 12.27, both lines show the default) is *Deny anyone from anyplace all rights*.

Figure 12.27. Access Control Rules and set/update values.

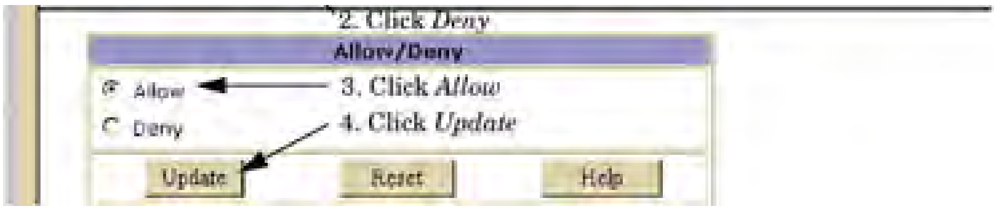


Access Control Rules for : path=d:/iplanet/iws60/docs/servlet

	Action	Users/Groups	From Host	Rights	Extra...	Continue
1	Deny	anyone	anyplace	all	x	☑
2	Deny	anyone	anyplace	all	x	☑

☑ Access control is on

Current Access deny response is the default file (redirection off)



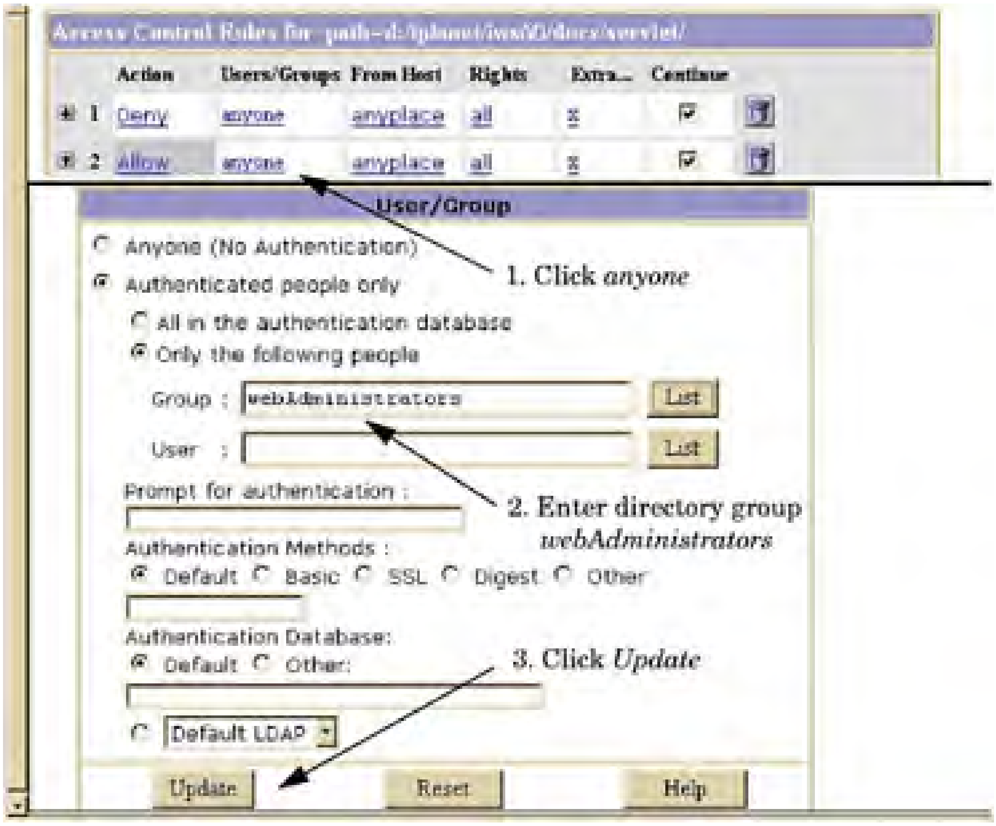
Allow/Deny

[✓ Allow] Deny

In the window shown in Figure 12.27, click *New Line* twice. Then click the *Deny* value on the second line. To allow access, click on *Allow*, and then click *Update*.

All of these clicks result in a new screen (Figure 12.28) where two ACL rule lines have been added:

Figure 12.28. Changed Access Control Rules.



Access Control Rules for: path=d:/iplanet/iws60/docs/servlet/

	Action	Users/Groups	From Host	Rights	Extra...	Continue
1	Deny	anyone	anyplace	all	x	✓
2	Allow	anyone	anyplace	all	x	✓

User/Group

Anyone (No Authentication) [✓ Authenticated people only]

All in the authentication database [✓ Only the following people]

Group : webAdministrators

User :

Prompt for authentication :

Authentication Methods : [✓ Default] Basic SSL Digest Other

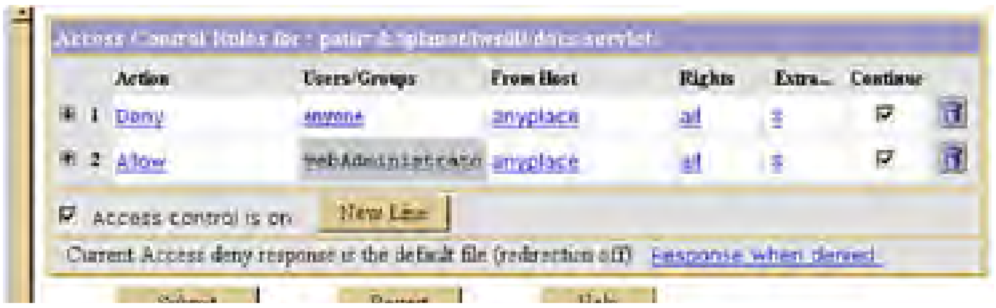
Authentication Database: [✓ Default] Other:

Default LDAP

1. Rule 1. Deny anyone from anyplace (any computer address) all rights to the directory *d*:/iplanet/iws60/docs/servlet/.
2. Rule 2. Allow anyone from anyplace (any computer address) all rights to the directory d:/iplanet/iws60/docs/servlet/.

Now, change the screen from allowing *anyone* access to allowing access only for members of the group *webAdministrators*. First click *anyone*, then enter *webAdministrators* in the *Group* field, and finally click *Update*. Now our rules are (as shown in Figure 12.29):

Figure 12.29. Access Control Rules allowing access to webAdministrators.



Access Control Rules for: path=d:/iplanet/iws60/docs/servlet/

	Action	Users/Groups	From Host	Rights	Extra...	Continue
1	Deny	anyone	anyplace	all	x	✓
2	Allow	webAdministrators	anyplace	all	x	✓

[✓ Access control is on]

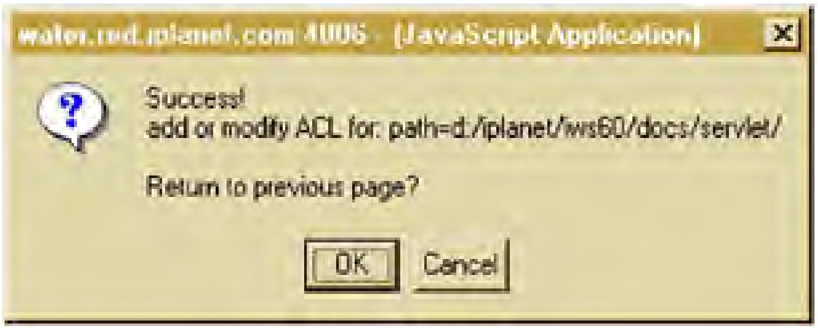
Current Access deny response is the default file (redirection off) Response when denied

1. Deny anyone from anyplace all rights to directory ../servlet.
2. Allow members of group *webAdministrators* from anyplace all rights to directory ../servlet.

Note: The order of the rules does matter (if the lines are reversed, the authorization will not work).

Click the *Submit* button to have the ACL added. A *Success* message prompt will be displayed (see Figure 12.30).

Figure 12.30. Successful ACL addition or modification.



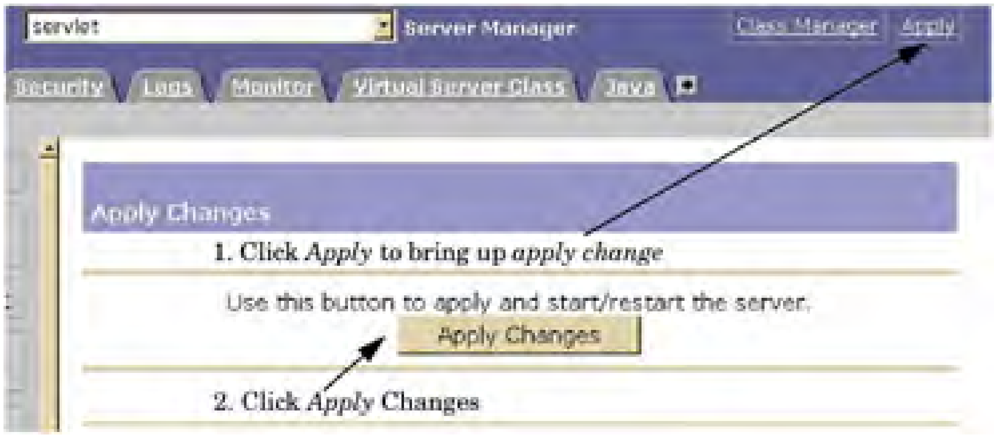
water.red.iplanet.com 4006 - [JavaScript Application]

Success!
add or modify ACL for: path=d:/iplanet/ws60/docs/servlet/

Return to previous page?

Click *OK* on the *Success* window, and you will be returned to the web server administrator main web page. Click *Apply* at the top right and the Apply Changes page will appear (Figure 12.31). Click *Apply Changes* to apply the new ACL and restart the web server instance.

Figure 12.31. Apply changes made to the configuration files.



Use this button to apply and start/restart the server.

The ACL is now in effect.

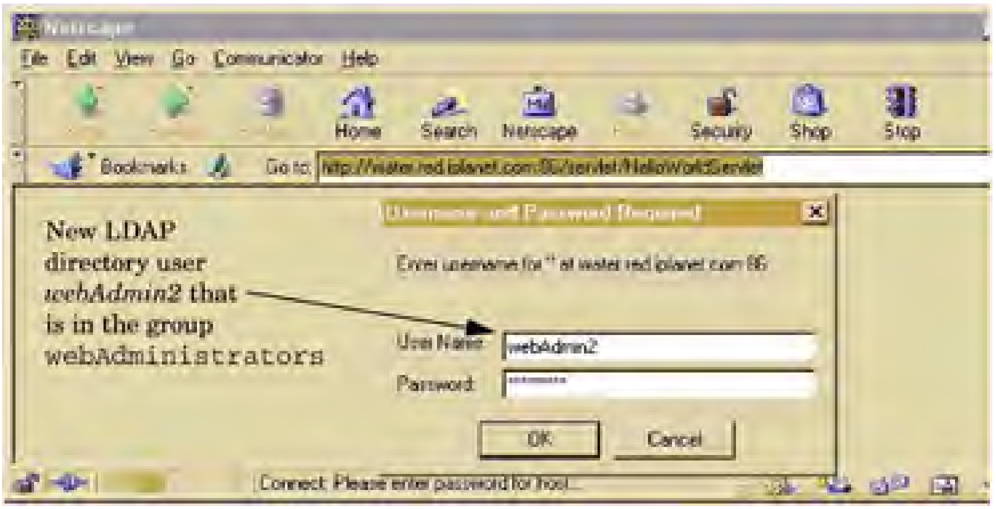
Testing the Restriction

When anyone tries to access a file in the servlet directory, the person is prompted for a user ID and password. For example:

http://water.red.iplanet.com:86/servlet/HellowWorldServlet

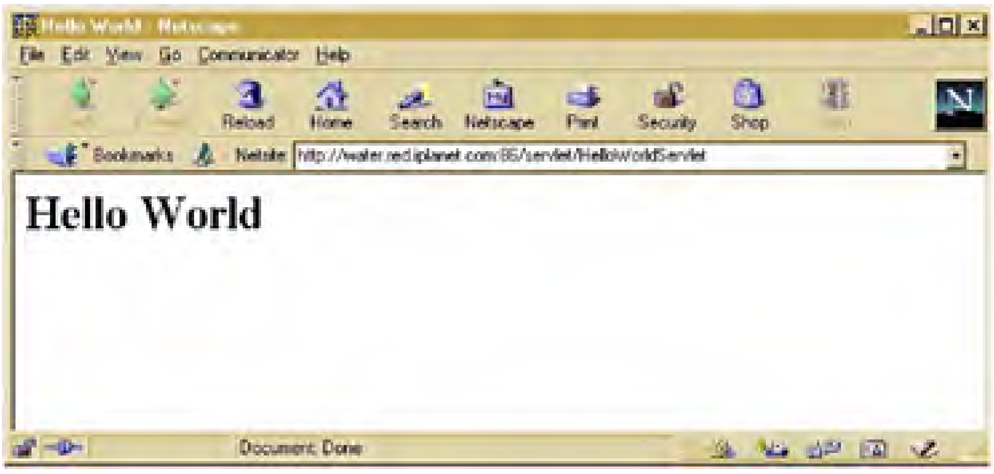
opens the window in Figure 12.32.

Figure 12.32. Login page.


Netscape
File Edit View Go Communicator Help
Home Search Netscape Security Shop Stop
Bookmarks Go to: http://water.red.iplanet.com:86/servlet/HelloWorldServlet
New LDAP directory user *webAdmin2* that is in the group webAdministrators
Username and Password Required
Enter username for " at water.red.iplanet.com:86
User Name: webAdmin2
Password:
OK
Cancel
Connect: Please enter password for host...

The servlet then executes and the *Hello World* message is returned (Figure 12.33).

Figure 12.33. Hello World welcome page.


Hello World - Netscape
File Edit View Go Communicator Help
Reload Home Search Netscape Print Security Shop
Bookmarks Netsite: http://water.red.iplanet.com:86/servlet/HelloWorldServlet
Document: Done

Hello World

You have successfully restricted access!

Summary

Restricting access to web site resources is a straightforward process:

1. Using the Sun ONE Web Server Administration Server, connect the web server to the Sun ONE Directory Server.
2. Still using the web server administration server, add users and put them under a group in the directory.
3. Select a web site directory resource to restrict access to.
4. Create an ACL to define the access restrictions.
5. Test.

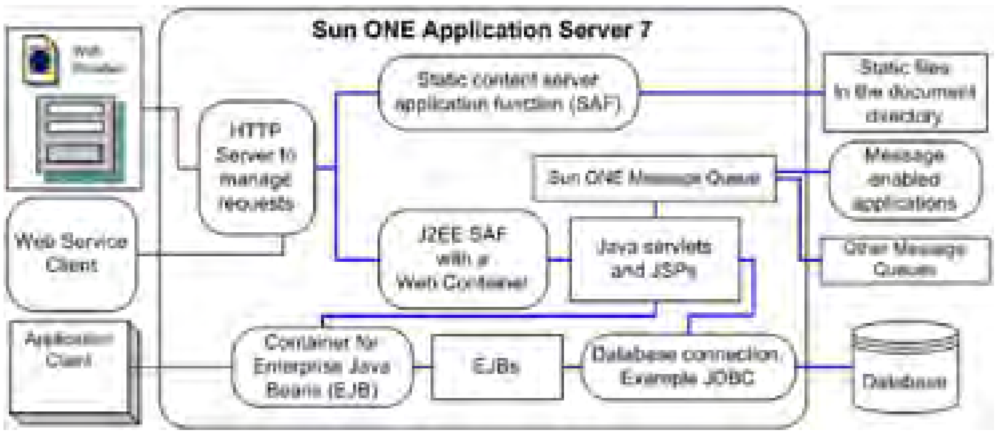
That's all there is to it.

Chapter 13. The Sun ONE Application Server 7

The Sun ONE Application Server components include some of Sun's best software technology together in one product. This application server is made from proven technology.

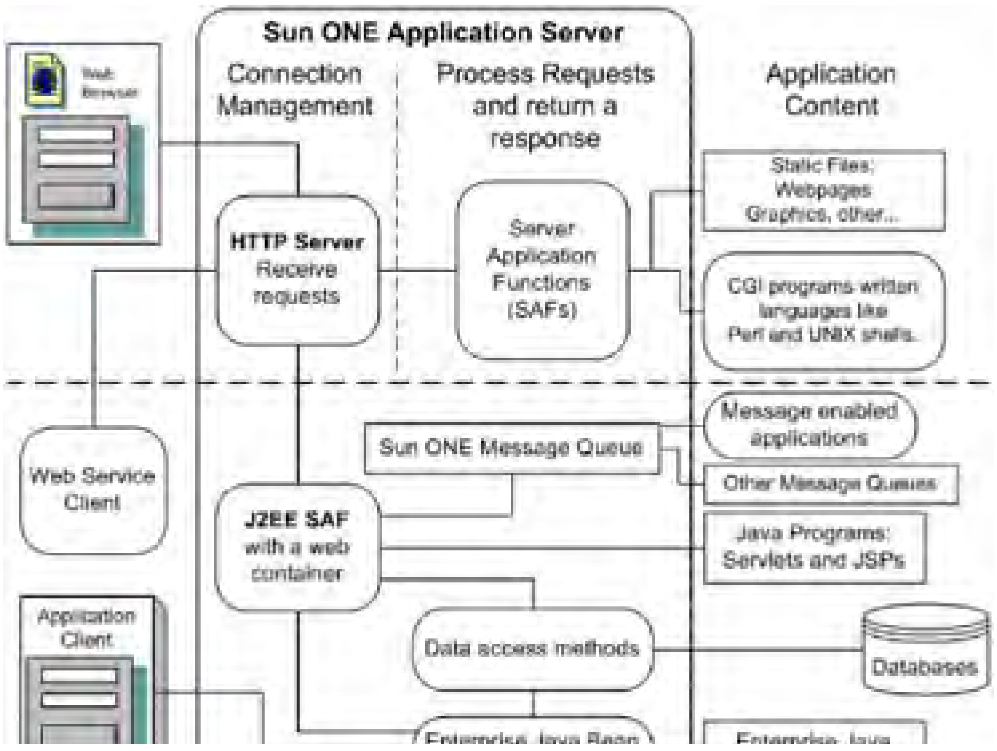
Figure 13.1 illustrates this point. For example, the Sun ONE Application Server includes an upgraded version of the Sun ONE Web Server for client request management and the Sun ONE Message Queue for Java Message Services (JMS). The web container is an enhanced version of the well-known Tomcat container. Tomcat is the web container for the Java 2 Enterprise Edition (J2EE) reference implementation. The message queue is the Sun ONE Message Queue product. All are proven technologies.

Figure 13.1. Some of the Sun ONE Application Server 7 components.


Sun ONE Application Server 7
Web Browser
HTTP Server to manage requests
Static content server application function (SAF)
Static files in the document directory
Sun ONE Message Queue
Message enabled applications
Other Message Queues
J2EE SAF with a Web Container
Java servlets and JSPs
Web Service Client
Application Client
Container for Enterprise Java Beans (EJB)
EJBs
Database connection Example JDBC
Database

Whereas the Sun ONE Web Server is designed to handle requests made by people using browsers, the application server goes over and above that. It is designed to handle client requests made by people using application clients and other applications using web service clients. Figure 13.2 shows each type of client communicating with a specific component of the application server.

Figure 13.2. The application server connects people and programs to content.


Sun ONE Application Server
Connection Management
Process Requests and return a response
Application Content
Web Browser
HTTP Server Receive requests
Server Application Functions (SAFs)
Static Files: Webpages Graphics, other...
CGI programs written languages like Perl and UNIX shells.
Message enabled applications
Sun ONE Message Queue
Other Message Queues
Web Service Client
J2EE SAF with a web container
Java Programs: Servlets and JSPs
Application Client
Data access methods
Databases

It is connectivity features that give the Sun ONE Application Server the ability to deliver services on demand today. On the front end, people and applications make requests to the application server. On the back end, there are a number of tools to connect to other applications and database systems.

[Team LiB]

Application Communications

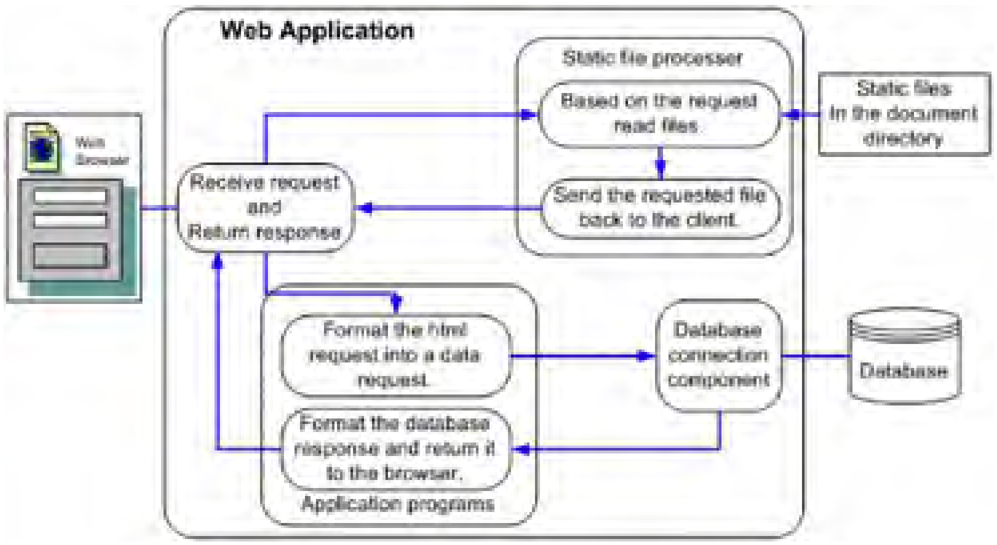
Remembering that the "ONE" in Sun ONE stands for Open Net Environment, it follows that the Sun ONE Application Server would support people and applications communicating over a network using open standards, and so it does.

This section covers both front-end and back-end communications. On the front end, people and applications make requests of the application server; on the back end, programs fulfill the requests by communicating with other applications and database systems.

Web Applications

Web applications run over the Internet and over intranets. Browser clients communicate with web applications using the HyperText Transfer Protocol (HTTP). The applications respond by returning HyperText Markup Language (HTML) files, pictures, and other types of media files as illustrated in Figure 13.3.

Figure 13.3. Structure of a web application.

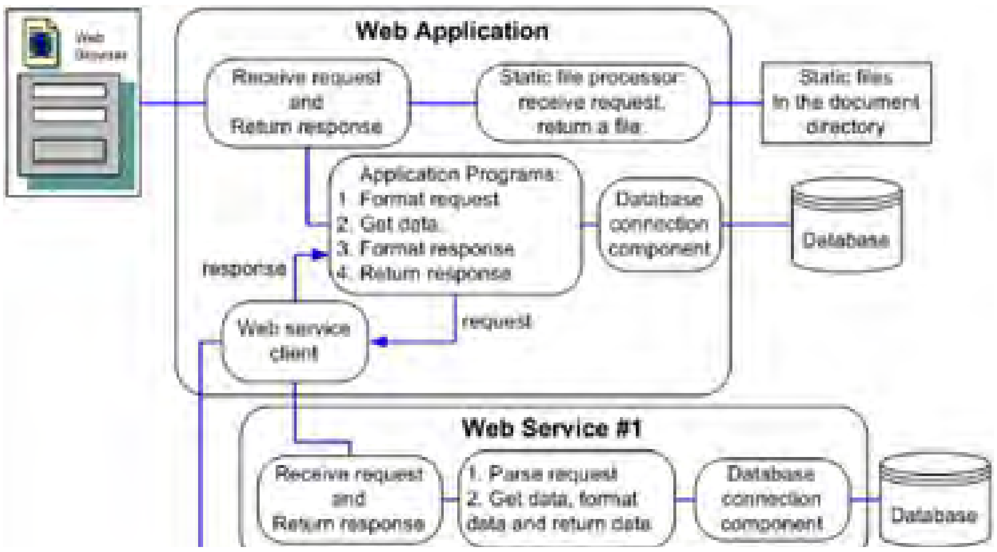

Web Application
Web Browser
Static file processor
Based on the request read files.
Send the requested file back to the client.
Static files In the document directory
Receive request and Return response
Format the html request into a data request.
Format the database response and return it to the browser.
Application programs
Database connection component
Database

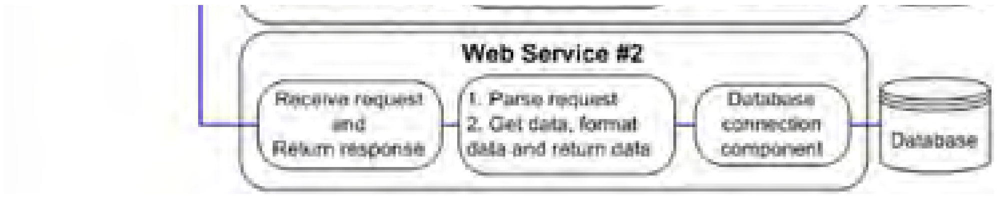
Web applications—search engines, shopping applications, web mail (doing email from a browser), and the like—have proliferated on the Internet, but they are designed to respond only to web browsers. As you'll see in the next section, web services overcome this limitation.

Web Services

Web services are applications that respond to requests from other applications. They run over the web to feed data to other services and applications. Figure 13.4 shows a web application responding to web browser requests. The web application selects data from various sources: the document directory (static files), local databases, or remote databases through web services. The web application and the web services might be in one location or they might be in separate geographic locations.

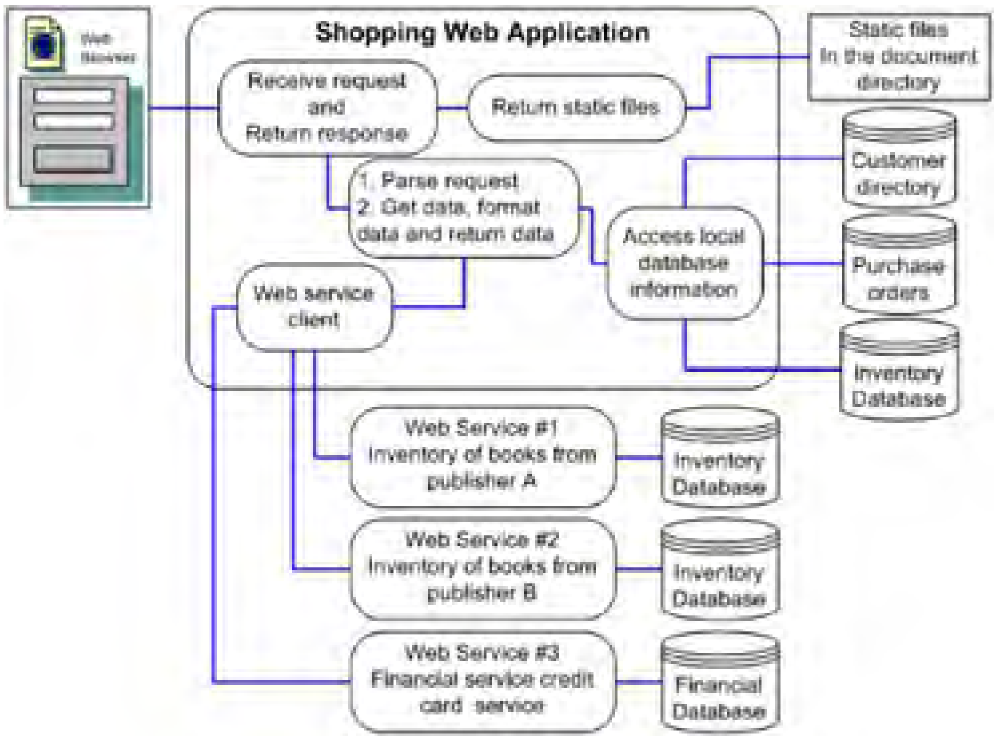
Figure 13.4. Web application getting information from a web service.


Web Application
Web Browser
Receive request and Return response
Static file processor: receive request, return a file
Static files In the document directory
Application Programs:
1. Format request
2. Get data.
3. Format response
4. Return response
Database connection component
Database
response
request
Web service client
Web Service #1
Receive request and Return response
1. Parse request
2. Get data, format data and return data
Database connection component
Database


Web Service #2
Receive request and Return response
1. Parse request
2. Get data, format data and return data
Database connection component
Database

Let's look at the steps the shopping application goes through to complete a shopper's request to buy products (Figure 13.5):

Figure 13.5. Web services and the shopping cycle.

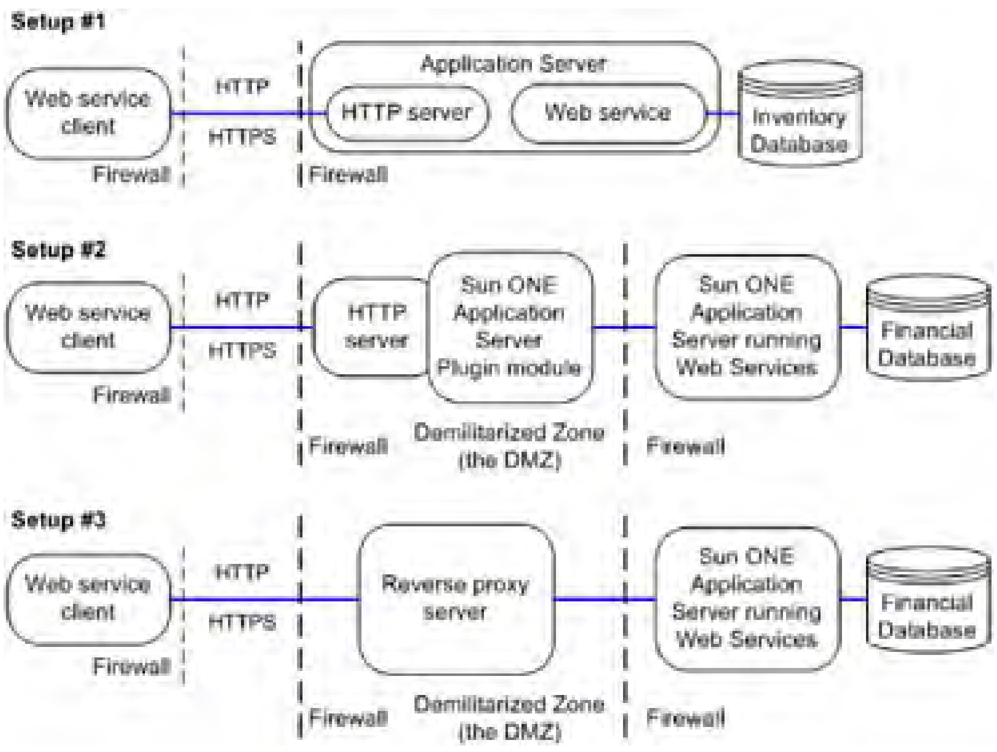

Web Browser
Shopping Web Application
Receive request and Return response
Return static files
Static files in the document directory
1. Parse request
2. Get data, format data and return data
Access local database information
Customer directory
Purchase orders
Inventory Database
Web service client
Web Service #1 Inventory of books from publisher A
Inventory Database
Web Service #2 Inventory of books from publisher B
Inventory Database
Web Service #3 Financial service credit card service
Financial Database

1. The shopper connects to the web shopping application using a browser.
2. The shopper selects books and goes to check out.
3. At the checkout, the web application checks its local inventory and, when necessary, checks with each publisher for available inventory. Publisher inventory checks are managed by the web service client communicating with each of the publishers.
4. The next step in the checkout process takes care of the financial end of the transaction. In this step, a process gets the shopper's credit card information from the local database and then communicates with the financial service's web service to debit the credit card.
5. The web application puts all this transaction information together and returns a response to the shopper, saying: The books will be shipped and your credit card has been debited. Thank you very much.

In Figure 13.5, the protocol used between web service clients and web service servers is HTTP or HTTPS (the secure version of HTTP). These protocols are extremely popular—so popular that most corporate firewalls already allow HTTP and HTTPS to get through.

Figure 13.6 shows three sample setups for web service security. The web service client is used to communicate with the web server that is behind a secure setup. Setup #1 is without a demilitarized zone (DMZ). In Setup #2, a plugin has been installed into the web server to act as a proxy running in the DMZ. The DMZ is another added level of security. The Sun ONE Web Server may be used for the HTTP server in this setup. Setup #3 illustrates the use of a reverse proxy server in the DMZ. The Sun ONE Application Server 7 is capable of acting as a reverse proxy server.

Figure 13.6. Sample security configurations for web services.

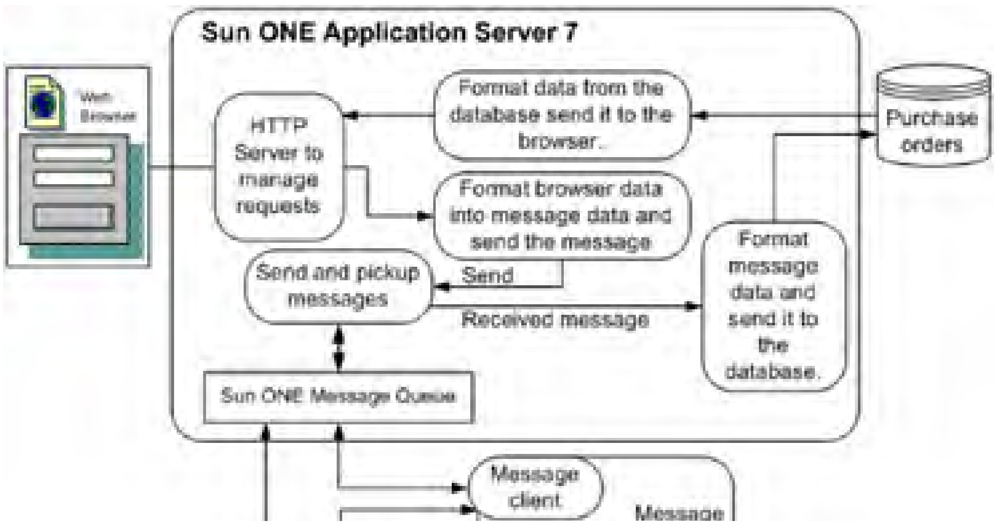


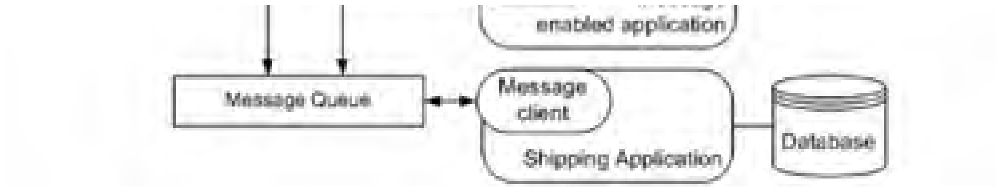
Application Communications with JMS

The previous shopping example showed information being retrieved from a number of separate applications via web services using HTTP. Another method of moving data between applications is the Java Message Service (JMS). JMS addresses the needs of applications to do reliable asynchronous messaging and JMS is a standard API for messaging that supports reliable point-to-point messaging as well as the publish-subscribe model. It is a common way for Java programs to create, send, receive, and read enterprise application messages.

Referring back to our shopping application (Figure 13.5), let's extend the functionality of the application. At first the application had access only to local data; then the web services extended the web shopping application to get data from external corporate sources. Now, the application is extended using JMS to bring in the shipping application (see Figure 13.7). In the example illustrated in Figure 13.5, the application stopped at taking the order and charging for the products. Now, using JMS, the shopping application sends a message to the shipping application to have the actual products shipped to the customer. Once the products have been shipped, the shipping application sends a message back to the shopping application. The shopping application updates the database. The next time the shopper logs on, she sees that the order has been fulfilled and that the shipment will be arriving on a specific date. Excellent customer service!

Figure 13.7. Application-to-application communications using JMS.


Sun ONE Application Server 7
Web Browser
HTTP Server to manage requests
Format data from the database send it to the browser.
Format browser data into message data and send the message
Purchase orders
Send and pickup messages
Send
Received message
Format message data and send it to the database.
Sun ONE Message Queue
Message client
Message

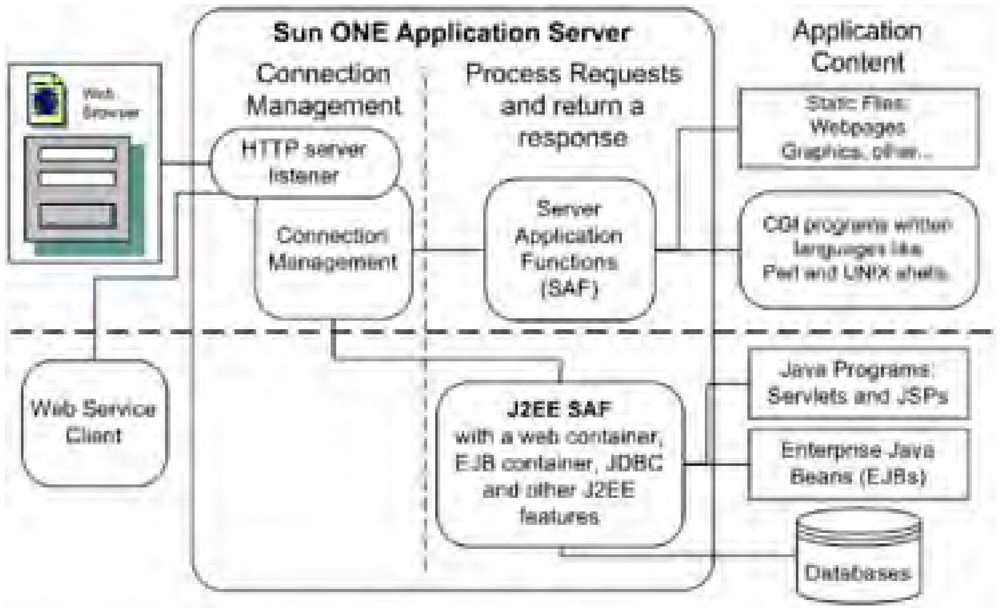


Web Server Features of the Sun ONE Application Server

So far, this chapter has focused on the communications between clients and servers, users, and applications using HTTP and JMS. Now we will talk about how the application server processes client HTTP requests.

The application server manages connections between web browsers and web site content. It makes sense, therefore, that the Sun ONE Application Server inherits many features from the Sun ONE Web Server. In Figure 13.8, the architecture above the dotted line is the same as the Sun ONE Web Server architecture. Below the line is the heart and soul of the Sun ONE Application Server: the J2EE engine for running application programs and creating web site content.

Figure 13.8. Web server features of the Sun ONE Application Server.

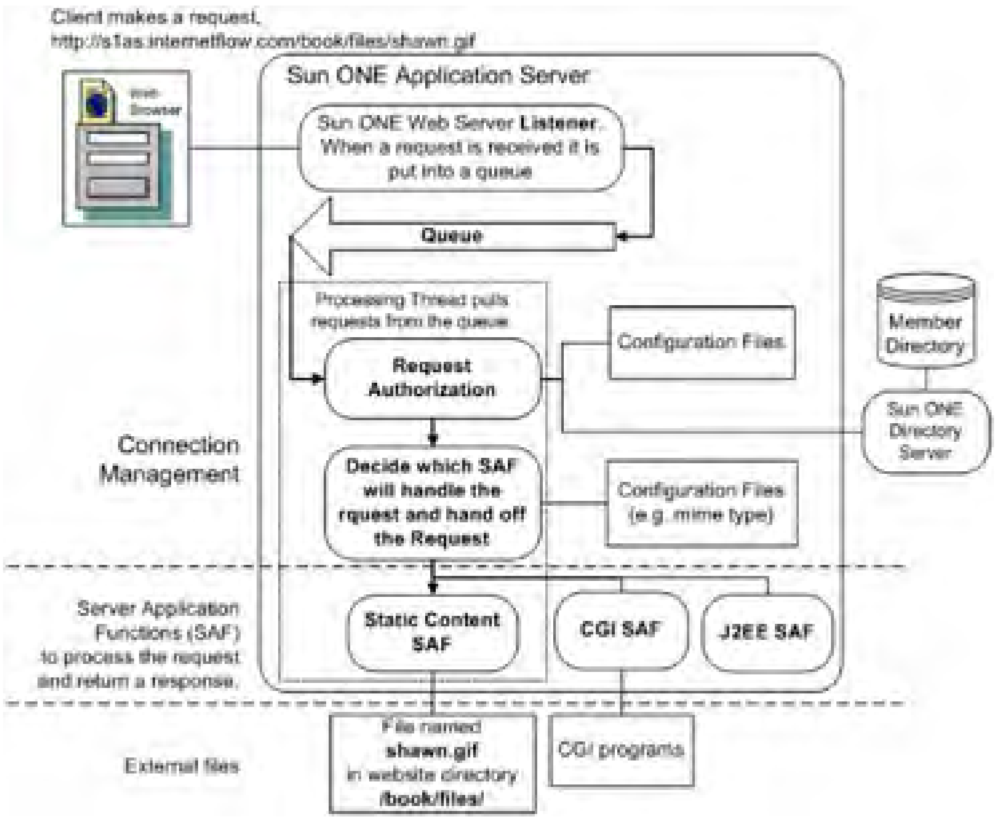

Sun ONE Application Server
Connection Management
Process Requests and return a response
Application Content
Web Browser
HTTP server listener
Connection Management
Server Application Functions (SAF)
Static Files: Webpages Graphics, other...
CGI programs written languages like Perl and UNIX shells
Web Service Client
J2EE SAF with a web container, EJB container, JDBC and other J2EE features
Java Programs: Servlets and JSPs
Enterprise Java Beans (EJBs)
Databases

To build the Sun ONE Application Server front end, the designers chose to use the code from the Sun ONE Web Server. They added enhancements, which give the application server an awesome connection management system. Because the Sun ONE Web Server was designed to have SAFs added to it, it was a straightforward process to add the new web container. The web container is an extremely modified version of the well-known Tomcat web container, which is the web container for the Sun J2EE reference implementation.

Processing Client Requests

Since the application server includes the features of the web server, it has an HTTP listener that listens for browser connections. In fact, the application server has all of the parts that the web server has (illustrated in Figure 13.9).

Figure 13.9. Sun ONE Application Server managing a browser request for a file.


Client makes a request,
http://s1as.internetflow.com/book/files/shawn.gif
Web Browser
Sun ONE Application Server
Sun ONE Web Server Listener. When a request is received it is put into a queue
Queue
Processing Thread pulls requests from the queue
Request Authorization
Configuration Files
Member Directory
Sun ONE Directory Server
Connection Management
Decide which SAF will handle the rquest and hand off the Request
Configuration Files (e.g. mime type)
Server Application Functions (SAF) to process the request and return a response.
Static Content SAF
CGI SAF
J2EE SAF
External files
File named shawn.gif in website directory /book/files/
CGI programs

In Figure 13.9, the application server handles a user request for a file:

http://s1as.internetflow.com/book/files/shawn.gif

Here are the steps that the application server goes through to respond to the request:

1. The browser connects to the web server listener on port 80, the default port.
2. The listener puts the request into the queue.
3. A process thread picks up the request from the queue and makes the following decisions:
    - Is the user making the request authorized to access the requested resource /book/files/shawn.gif? The web server checks its configuration files for authorization information, and, if connected to a directory server, it checks the security data in the directory as well. In this case, the user is authorized.
    - What is the MIME type of the resource requested? The web server checks its configuration files and makes the decision based on the directory of the resource /book/files/ and the filename extension gif. In this case, based on the information in the configuration files, it is decided that the static content SAF will handle this request.
4. The request is now handed off to the static content SAF.
5. The static content SAF reads the file shawn.gif and sends the file back to the browser. This is the final step. The web server has returned a response to the browser's request.

The first part of the serving process is the connection management part. The last part of the process is the SAF step. The SAF step is where the response to the request happens.

The static content SAF deals with reading and sending files without making any changes to the files. This type of content is static. It is not changed by the web server. By contrast, the J2EE SAF creates dynamic content-based web applications written in the Java programming language.

Static Content SAF

The static content SAF of the application server connects people to the files that are in directories on the server. This is the basic method to serve content. During an installation of the Sun ONE Application Server, this method is set up by default.

When an application server receives a request for a file, the static content SAF reads the file and sends the file to the requesting browser. To illustrate the mapping between a URL request and the physical file, here is an example:

URL: *http://s1as.internetflow.com/book/files/shawn.jpg*

In our example URL

- The protocol is *http*
- The computer is *s1as.internetflow.com*
- The offset directory is */book/files/*
- The file is *shawn.jpg*

The web server's physical document directory is configured as /internet/appserver/docs/. This means the URL *http://s1as.internetflow.com/* is mapped to the physical directory */internet/appserver/docs/* and the URL *http://s1as.internetflow.com/book/files/* is mapped to the directory /internet/appserver/docs/book/files/

The complete URL *http://s1as.internetflow.com/book/files/shawn.jpg* is mapped to the file /internet/appserver/docs/book/files/shawn.jpg. When this URL is requested, the file shawn.jpg is read by the static content SAF and sent back to the web browser.

The static content SAF handles the serving of files stored under the application server's document directory. This is a classic web server function. The classic application server function is handling client requests using application programs.

CGI SAF

The Common Gateway Interface SAF runs programs to create web site content. This SAF spawns a new process for running a program. CGI programs are generally written in Perl, in operating system shells, or in C language.

The setup of the Sun ONE Web Server CGI SAF has some similarities to the static content SAF. One of the similarities is the configuration of an offset directory. In this CGI URL:

http://s1as.internetflow.com/cgi-bin/prog1.sh

the offset directory /cgi-bin/ is mapped to the physical directory of /internet/wsserver/cgi-bin/. This means that all requests for files in /cgi-bin/ cause the web server to spawn a process to run the program file. And when the program sends/prints to the standard output, the print characters are sent back to the requesting browser.

J2EE SAF

The J2EE SAF handles client requests by running Java-based servlet programs. In turn, the servlet programs might do a number of things; for example:

1. Communicate with databases directly.
2. Communicate with EJBs. The EJBs may then perform database transactions.
3. Send and receive messages through the Java Message Queue.
4. Convert data and messages into new formats.
5. Create an HTML or an Extensible Markup Language (XML) response that is sent back to the requester.

The J2EE SAF responds to requests from browser clients and from application web service clients. The response is created by the programs run in the web container or the EJB container.

Application Programming

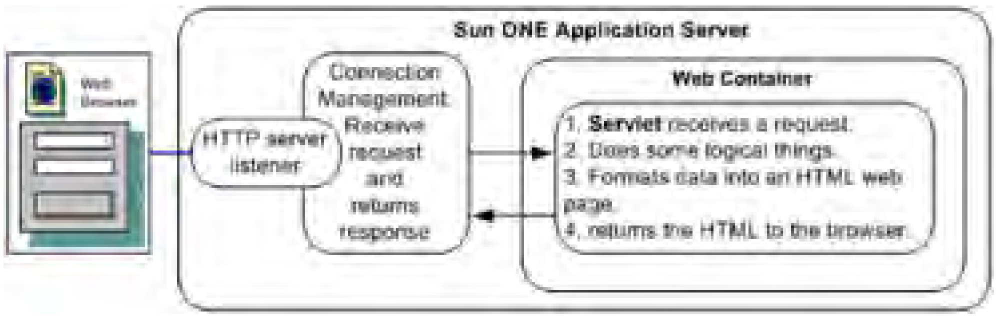
This section describes the process and architecture of creating application programs. The Sun ONE Application Server supports both types of programming models—J2EE programming and the classic CGI web server programming.

If you bought the application server, more than likely it is because you want to create J2EE applications. J2EE applications have three major components: servlets, Java Server Pages (JSPs), and Enterprise Java Beans (EJBs). Another important factor for choosing this application server is for data connectivity components that come with the application server—a major function of any major application is to access data.

J2EE Programming

J2EE programmers start with Java servlets. The servlet is the first program run in a web application. The browser request is passed to the web container from the connection management process (Figure 13.10). The web container checks its configuration files to determine what servlet is to be run. If the servlet is already in memory, the servlet is run right away. Otherwise the container loads and runs the servlet. The container manages the servlets in memory. To run the servlet, the container uses the Java Virtual Machine (JVM) to execute the Java servlet class file (the byte code).

Figure 13.10. Java servlet runs in the web container.

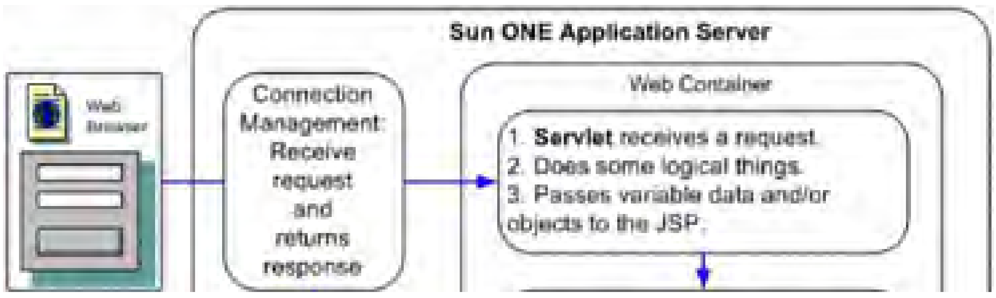

Sun ONE Application Server
Web Browser
HTTP server listener
Connection Management Receive request and returns response
Web Container
1. Servlet receives a request.
2. Does some logical things.
3. Formats data into an HTML web page.
4. returns the HTML to the browser.

Still referring to Figure 13.10, the servlet receives the request, runs some logic based on the request, formats the data into an HTML web page, and returns the HTML to the browser. For example (following the numbered steps in Figure 13.10):

1. A servlet receives a request from a user (using the browser) for the current time in Bangalore, India, i.e., the servlet receives the parameter Bangalore-India.
2. The servlet gets the system time of the computer and calculates the current date and time in Bangalore, India.
3. Results are formatted into an HTML web page.
4. The HTML is returned to the browser and the browser displays the date and time to the user.

The tedious part of doing everything with a servlet is writing the Java code to create the HTML web page. In Figure 13.11, a JSP is used to avoid this problem. JSPs are an ingenious programming method. A JSP is an HTML web page with Java program code embedded in it. The first time the servlet calls the JSP, the web container transforms the JSP into a servlet, compiles the servlet into a class file, and then uses the JVM to run the class file. The next time the JSP is called, it runs right away because it has already been translated, compiled, and loaded into memory. To compile the JSP servlet code, the application server uses version 1.4 of the Java Development Kit (JDK). (The JDK and the JVM are automatically installed during a regular Sun ONE Application Server 7 installation.)

Figure 13.11. Servlet passes control to JSP to format and return the data.


Sun ONE Application Server
Web Browser
Connection Management Receive request and returns response
Web Container
1. Servlet receives a request.
2. Does some logical things.
3. Passes variable data and/or objects to the JSP.

1. The JSP formats data into an HTML web page.
2. The HTML is sent to the browser

To create a JSP, a web page designer creates an HTML web page, and then a programmer adds the Java code or special JSP tags to make the web page dynamic. This saves programmer time, and the creative work is done by a web page designer, who is better equipped to handle the creative side. This is a lot easier than writing a servlet that formats HTML web pages. Here is a simple JSP example.

```
01. <%@page import="java.io.*,java.util.Date" %>
02. <html>
03. <head>
04. <title>System Time</title>
05. </head>
06. <body>
07. Time (hours:minutes)
08. <% Date d = new Date();
09. System.out.println(d.getHours()+":"+d.get-
  Minutes());
10. %>
11. </body>
12. </html>
```

Line 01 declares Java classes. Java code is enclosed by <% and %> as in lines 08 to 10. Lines 08 and 09 are lines of Java code. The rest is regular HTML.

Here is an example servlet that gives the same output as the JSP above:

```
import java.io.*;
import java.util.Date;
public class thedate {
public static void main( String args[] ) {
Date d = new Date();
System.out.println( "<html>" );
System.out.println( "<head>" );
System.out.println( "<title>System Time</title>"
  );
System.out.println( "</head>" );
System.out.println( "<body>" );
System.out.print( "Time (hours:minutes) " );
System.out.println( d.getHours()+":"+d.get-
  Minutes() );
System.out.println( "</body>" );
System.out.println( "</html>" );
}
}
```

The next type of Java program is EJB, a program used to retrieve data from a database. EJBs run in a different container—the EJB container. Figure 13.12 illustrates a classic example of the use of an EJB. The EJB gets the data from a database, applies some business logic to it, and then uses the data to respond back to the web browser.

Figure 13.12. Servlets, JSPs, and EJB programs working together.

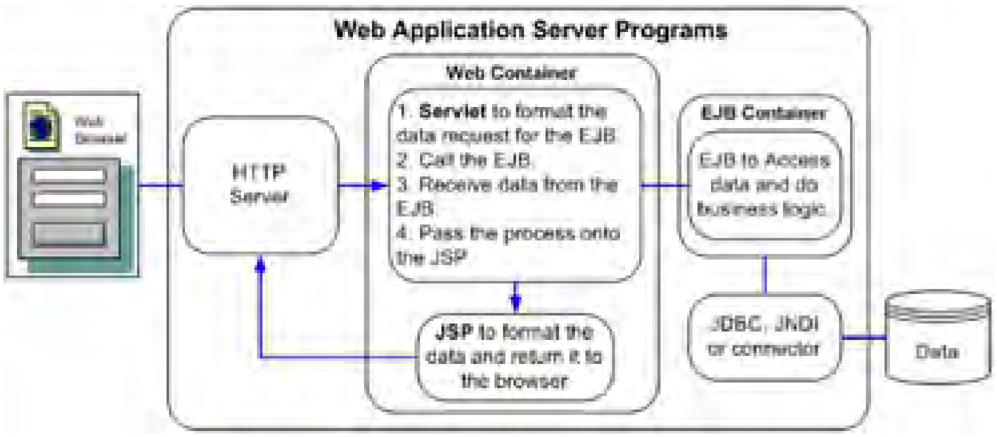
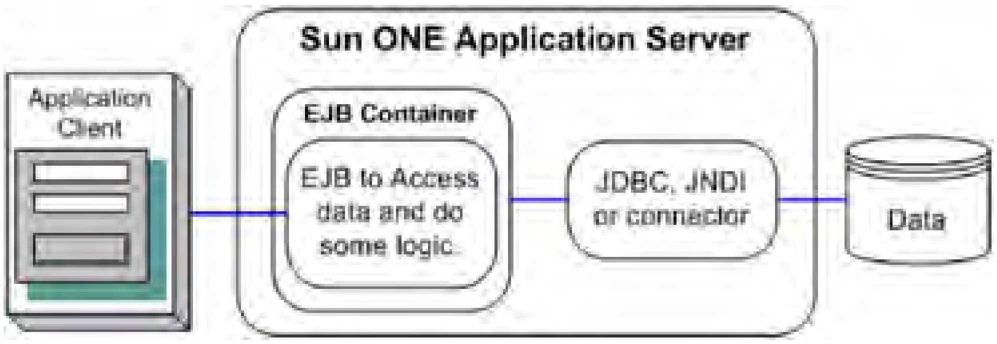

Web Application Server Programs
Web Browser
HTTP Server
Web Container
1. Servlet to format the data request for the EJB
2. Call the EJB
3. Receive data from the EJB
4. Pass the process onto the JSP
JSP to format the data and return it to the browser
EJB Container
EJB to Access data and do business logic
JDBC, JNDI or connector
Data

Figure 13.12 puts all the main Java program types together to do one job: respond to a browser request logically, and with database data. Once the EJB has been created and is being used in the web application, it might be used by other applications. And, since the EJB container has server properties, EJBs can be called by completely different applications on other computers as in Figure 13.13.

Figure 13.13. EJB container working as server to serve up data.


Sun ONE Application Server
Application Client
EJB Container
EJB to Access data and do some logic
JDBC, JNDI or connector
Data

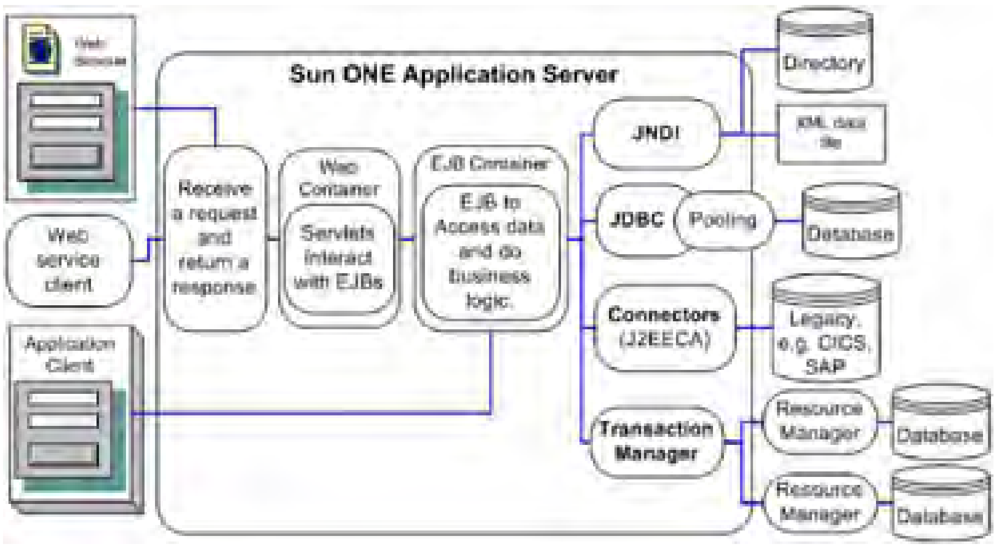
EJBs are very flexible program components. Once built, other programs call the EJB to get data that has already been pulled from the database and put into a Java program object. This makes database data very easy to use because, once programmed, other programs need to access only the Java program objects, not the database.

Data Connectivity

The Sun ONE Application Server provides a number of methods to connect to many types of data. Multiple methods are necessary because data is stored in many forms, including relational database structures, hierarchical structures in a directory, specialized formats of CICS and SAP, and the popular XML data structures in American Standard Code of Information Interchange (ASCII) files.

Figure 13.14 shows the J2EE components used to communicate with the Application Server's prospective data storage system. I use the term "J2EE components" because the J2EE standard defines how the components are to be created. For example, the standard defines the APIs to be used when programming EJBs to access the data.

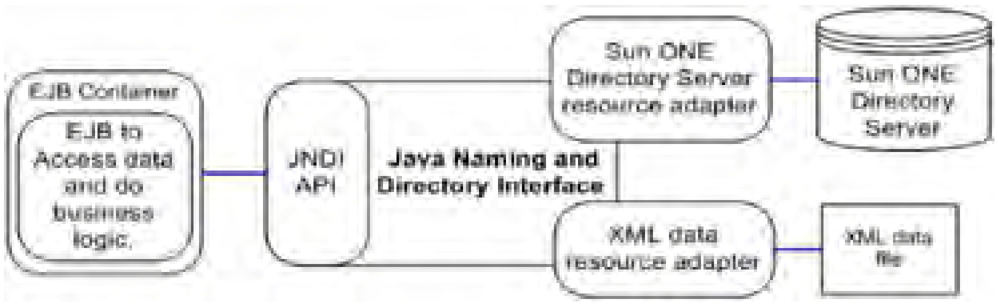
Figure 13.14. Many possibilities for data connectivity.



With the following data components, it is a straightforward process to set up data to be used with an application.

- The Java Naming and Directory Interface (JNDI) is used to access naming and directory data. Figure 13.15 illustrates the JNDI API on the EJB side and the vendor's adapter on the other side. An example of using JNDI is this: The Sun ONE Application Server stores database resource information in the XML data file format; when writing programs to access a database, a JNDI lookup is performed to open a database connection.

Figure 13.15. Java Naming and Directory Interface.


EJB Container
EJB to Access data and do business logic.
JNDI API
Java Naming and Directory Interface
Sun ONE Directory Server resource adapter
Sun ONE Directory Server
XML data resource adapter
XML data file

- Java DataBase Connectivity (JDBC) is a well-known API for accessing databases. JDBC has been around for a number of years and is a popular method for accessing databases such as Oracle and Sybase. Vendors implement JDBC API-compliant drivers to provide database access.
- The J2EE Connector Architecture (J2EECA) is an architecture for integrating J2EE programs to enterprise information systems (EISs) through an API. EIS vendors write a resource plugin into the connector. This gives us a common API on one side for the programmer to use and vendor-specific resource plugins on the other side as in Figure 13.16. Two examples are CICS and SAP, but there are many other vendor-specific connector resource adapters.

Figure 13.16. Connector architecture.

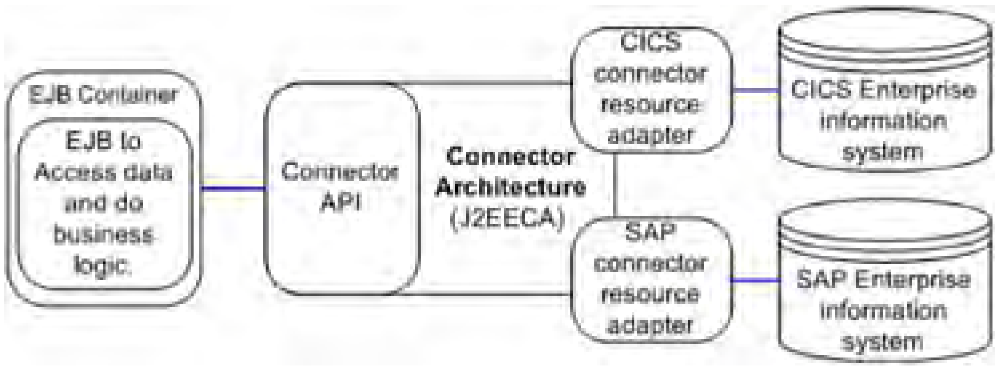
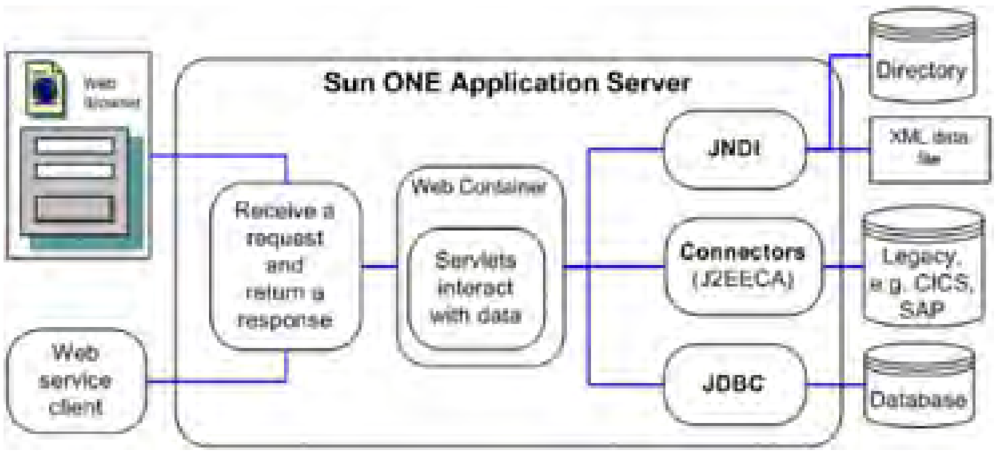

EJB Container
EJB to Access data and do business logic.
Connector API
Connector Architecture (J2EECA)
CICS connector resource adapter
CICS Enterprise information system
SAP connector resource adapter
SAP Enterprise information system

Figure 13.14 had EJBs accessing the data. In Figure 13.17 the data is accessed by servlets. The programmer has the option of choosing how to write the data connection part of the program. However, if the programmer chooses to use servlets, there will be a limited number of options. Not all of the data tools are usable by servlets. EJBs are more flexible and more powerful.

Figure 13.17. Data access from Java servlets.



Web Services

Web services can be simple or complex. When a web service becomes complex, the Sun ONE Application Server has the Java API for XML (JAX) pack to make writing J2EE web service programs very straightforward (even easy). The JAX pack is a collection of Java XML APIs that are the building blocks for developing Java web services. The JAX pack, among other things, translates data into XML format and translates XML data into Java objects based on the Document Object Model (DOM). The JAX pack helps programmers write web service client-side programs and web service server-side programs.

PREVIOUS NEXT

Summary

The Sun ONE Application Server is a powerful application server system. It is based on proven technology, including the Sun ONE Web Server, Tomcat, Sun ONE Message Queue, and much more.

The Sun ONE Application Server 7 is J2EE 1.3 compliant, which means it has a vast number of tools to create applications. It has a number of methods, such as XML files, to access a number of databases and other data systems.

Programming in a J2EE environment is often separated into four tiers or layers: client tier, presentation layer, business logic layer, and the database tier.

1. Client tier
    - Browser— uses HTTP and HTTPS to make requests for data. Data is returned to the browser in HTML format.
    - Web service client— uses HTTP and HTTPS to make requests for data. Data is returned to the browser in XML format.
    - Application client— makes method invocation requests of EJBs. Requests and responses are serialized Java objects.
2. Presentation layer
    - Connection management— receives HTTP or HTTPS requests and passes the requests onto servlets. Servlets do the preliminary work to prepare calls to EJBs. EJBs are typically for transactional business logic. When the EJB returns data to the servlet, the servlet passes control to the JSP. It is also possible to develop reasonably complex business applications using servlets and JSPs without EJBs.
3. Business logic layer: This is the domain of EJBs. They receive requests from servlets or from application clients. The EJB gets data from data sources and puts the data through some business logic. This prepared data is then given back to the caller. Servlets are sometimes used to manage business logic. For overall J2EE design considerations, servlets are secondary to EJBs when it comes to business logic.
4. Data tier: This is the source of data from simple ASCII files to complex database systems.

However you decide to do your application server program, be sure to DO IT. It is easy to get started and get going.

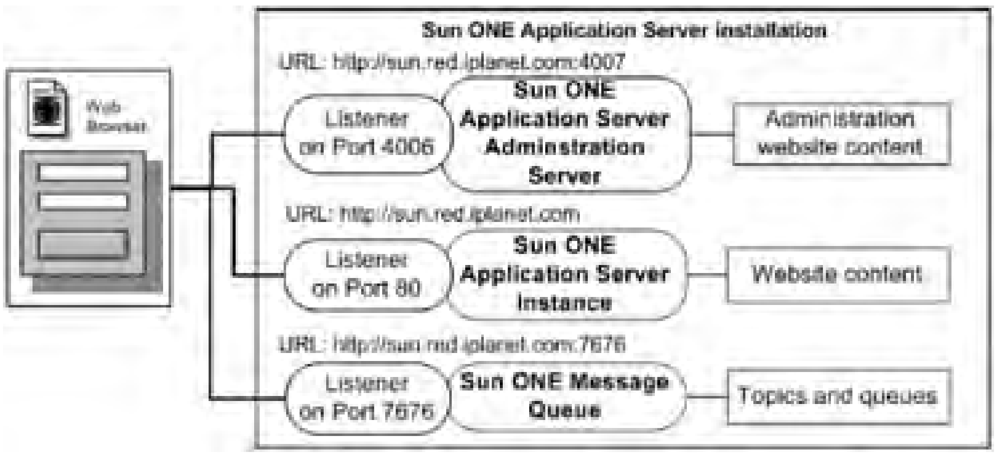
Chapter 14. Installing the Sun ONE Application Server 7

When you install a Sun ONE Application Server, you get

- **An administration server** for managing application server components. The Sun ONE Application Server Administration Server is a specialized HTTP (web) server instance with a web-enabled application to manage and configure the application site components.
- **An application server instance** for serving applications.
- **The Sun ONE Message Queue** for Java Message Services.

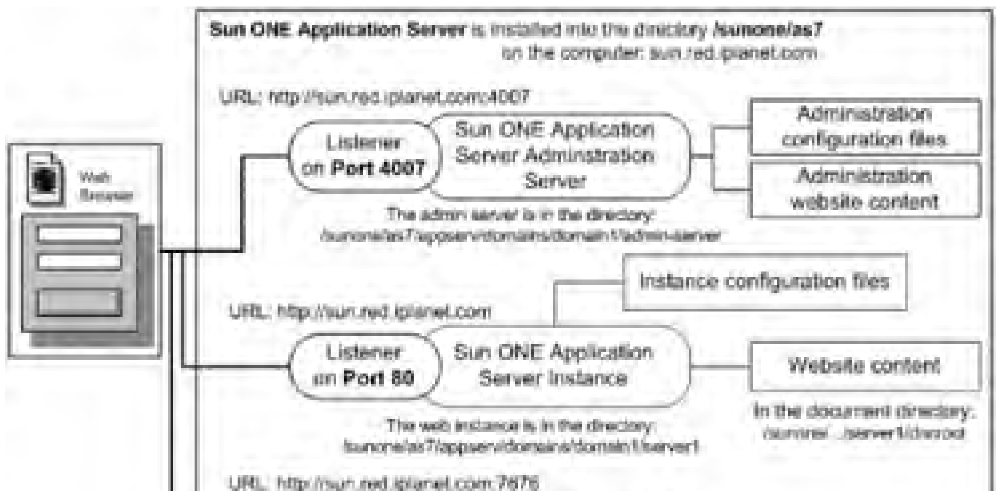
Figure 14.1 shows a web browser being used to communicate with each of these servers. The web browser uses one URL to connect to the administration server, a second URL to connect to the application server instance, and a third URL to test the message queue.

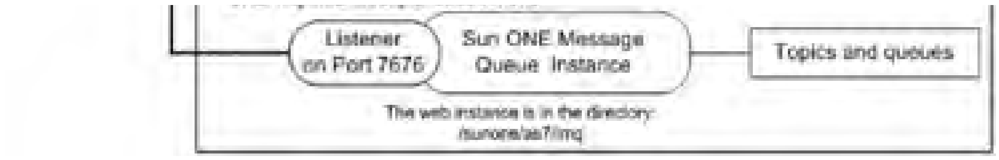
Figure 14.1. Web browser communicating with each server.


Sun ONE Application Server installation
Web Browser
URL: http://sun.red.iplanet.com:4007
Listener on Port 4006
Sun ONE Application Server Administration Server
Administration website content
URL: http://sun.red.iplanet.com
Listener on Port 80
Sun ONE Application Server Instance
Website content
URL: http://sun.red.iplanet.com:7676
Listener on Port 7676
Sun ONE Message Queue
Topics and queues

During the installation, the installer requires only a few values to set up the complete Sun ONE Application Server. Figure 14.2 has the values that I used to set up my sample web server. Note that all the directories for the application servers are under the root installation directory of /sunone/as7. Keeping everything under one root directory makes it easy to track the location of all the Sun ONE Application Server files.

Figure 14.2. Sun ONE Application Server sample installation configuration.


Sun ONE Application Server is installed into the directory /sunone/as7 on the computer: sun.red.iplanet.com
Web Browser
URL: http://sun.red.iplanet.com:4007
Listener on Port 4007
Sun ONE Application Server Administration Server
Administration configuration files
Administration website content
The admin server is in the directory: /sunone/as7/appserv/domains/domain1/admin-server
Instance configuration files
URL: http://sun.red.iplanet.com
Listener on Port 80
Sun ONE Application Server Instance
Website content
In the document directory: /sunone/.../server1/docroot
The web instance is in the directory: /sunone/as7/appserv/domains/domain1/server1
URL: http://sun.red.iplanet.com:7676


Listener on Port 7676
Sun ONE Message Queue Instance
Topics and queues
The web instance is in the directory: /sunone/as7/imq

Another value required during an installation is the administration user. In the sample installations, I have used the administration user ID of *admin* and the password *admin*. When a person connects to the Sun ONE Application Server Administration Server, he or she is prompted for this user ID and password. The administration user gets you logged in to the administration server.

There are two types of installation: a command-line install and a GUI install. In this chapter, the installation on the UNIX Solaris Operating System uses the command-line installation, and the install on a computer running Windows 2000 uses the GUI installation. A nice feature is that the GUI install is the same for both Solaris and Windows. So, whichever you choose—GUI or command-line—the setup prompts are all the same for Solaris and Windows. In the near future, maybe by the time you read this book, a Linux version will be available.

UNIX Command-Line Installation

I have used Solaris 8 for this installation. The Sun ONE Application Server 7 for Solaris is included on the book DVD or you can download it from www.sun.com.

1. Once connected to www.sun.com, select *Downloads.*
2. Select *View All.*
3. Find *Sun ONE Application Server 7*. Use either the *Standard Edition* or the *Platform Edition*—they both work well for our purposes.
4. Then select your operating system, *Solaris*. Click *Download*.
5. Then follow the steps to download the software onto your computer.
6. Put the file into a working directory that you can retain and use later.
7. Decompress the file. Use these commands to decompress a tar.gz file:

   ```
   gzip -d *.gz
   tar xvf *.tar
   ```

After gzip and tar, here is my file listing:

```
# ls
appserv.class     package          setup
sun-appserv7se-solaris.tar
```

sun-appserv7se-solaris.tar is the file I downloaded.

Using Command-Line Mode

Run the installation program setup in command-line mode. Installer responses are underscored.

```
# ./setup -console
... When you are ready, press Enter to continue.
    <Press ENTER to Continue>
Welcome to the Sun ONE Application Server Installation
  Program
    <Press ENTER to Continue>
Before you install this product, you must read and
  accept the entire Software License Agreement under
  which this product is licensed for your use.
    <Press ENTER to display the Software License Agree-
      ment>
...
  Have you read, and do you accept, all of the terms of
  the preceding Software License Agreement [no] {"<"
  goes back, "!" exits}? yes

The Sun ONE Application Server components will be
  installed in the following directory, which is
  referred to as the "Installation Directory".To use
  this directory, press only the Enter key. To use a
  different directory, type in the full path of the
  directory to use followed by pressing the Enter key.

  Sun ONE Application Server Installation Directory
  [/opt/SUNWappserver7] {"<" goes back, "!" exits}:
  /sunone/as7

The directory "/sunone/as7" does not exist.
Do you want to create it now or choose another direc-
  tory?
1. Create Directory
2. Choose New
Enter the number corresponding to your choice  [1] {"<"
  goes back, "!" exits} 1 <hit the enter key>

Select Components:
  Do you want to install Application Server [yes] {"<"
    goes back, "!" exits} <hit the enter key>
```

```
  Do you want to install Sample Applications [yes] {"<"
    goes back, "!" exits} <hit the enter key>
  Do you want to install PointBase Server 4.2 [yes] {"<"
    goes back, "!" exits} no
  Do you want to install Application Server Administra-
    tion Client [no] {"<" goes back, "!" exits} <hit the
    enter key>
  Do you want to install Support for Sun ONE Studio 4,
    Enterprise Edition for Java [no] {"<" goes back, "!"
    exits} <hit the enter key>
Java Configuration
1. Install Java 2 SDK (1.4.1)
2. Reuse existing Java 2 SDK
3. Exit Installation
What would you like to do  [1] {"<" goes back, "!"
  exits}? <hit the enter key>

The Sun ONE Application Server configuration files will
  be installed in the following directory, which is
  referred to as the "Product Configuration Direc-
  tory".To use this directory, press only the Enter key.
  To use a different directory, type in the full path of
  the directory to use followed by pressing the Enter
  key.

  Sun ONE Application Server Product Configuration
    Directory
  [/etc/opt/SUNWappserver7] {"<" goes back, "!" exits}:
    /sunone/as7/config

The directory "/sunone/as7/config" does not exist.
Do you want to create it now or choose another direc-
  tory?
1. Create Directory
2. Choose New
Enter the number corresponding to your choice  [1] {"<"
  goes back, "!" exits} 1 <hit the enter key>

The Sun ONE Application Server default domain files will
  be installed in the following directory, which is
  referred to as the "Default Server Configuration
  Directory".To use this directory, press only the Enter
  key. To use a different directory, type in the full
  path of the directory to use followed by pressing the
  Enter key.

  Sun ONE Application Server Default Server Configura-
    tion Directory
  [/var/opt/SUNWappserver7] {"<" goes back, "!" exits}:
    /sunone/as7/appserv

The directory "/sunone/as7/appserv" does not exist.
Do you want to create it now or choose another direc-
  tory?
1. Create Directory
2. Choose New
Enter the number corresponding to your cho" goes back,
  "!" exits} 1 <hit the enter key>

Supply the admin user's password and override any of the
  other initial configuration settings as necessary.
  Admin User [admin] {"<" goes back, "!" exits}:
  Admin User's Password (8 chars minimum) {"<" goes
    back, "!" exits}: password
  Re-enter Password {"<" goes back, "!" exits}: password
  Admin Server Port [4848] {"<" goes back, "!" exits}:
    4007
  HTTP Server Port [80] {"<" goes back, "!" exits}:
```

```
Checking disk space... (please wait)

The following items for the product Sun ONE Application
  Server will be installed:

Product: Sun ONE Application Server
Location: /sunone/as7
Space Required: 384.30 MB
-----------------------------------
    JDK 1.4.0_02
Sun ONE Studio 4, Enterprise Edition for Java
Application Server Core
Sun ONE Message Queue 3.0.1
Startup

Ready to Install

1. Install Now
2. Start Over
3. Exit Installation

  What would you like to do [1] {"<" goes back, "!"
    exits}?

Installing Sun ONE Application Server
|-1%---------25%---------50%---------75%---------100%|
Installation Successful.

Next Steps:

  1.Start the Application Server by executing:
    /sunone/as7/appserv/bin/asadmin start-appserv

  2.Start the Admin Console:
    http://localhost:4007

  3.Access the About Sun ONE Application server welcome
    page at:
    file://sunone/as7/appserv/docs/about.html

  4.See the Getting Started Guide on the About page for
    a hands on tour of the application server environ-
    ment.
#
```

The following are the steps I went through to upgrade my license. You should handle the steps based on your own requirements. Note: I received the License Key that I am using from the Sun Microsystems web site after I downloaded the application server.

```
Do you want … yes…

Do you want to  proceed to the license upgrade screen
  [no] {"!" exits}? yes

This installation of the Sun ONE Application Server 7 is
  bundled with a license. If you have a copy of another
  license handy and you want to upgrade now, please
  enter your license key. If you want to stay with the
  bundled license or if you want to upgrade at a later
  time, enter '!' to exit.
  License Key [] {"<" goes back, "!" exits}: 2284910653-
    4386095633
License has been successfully upgraded.
```

Testing the Application Server by Connecting to the Administration Server

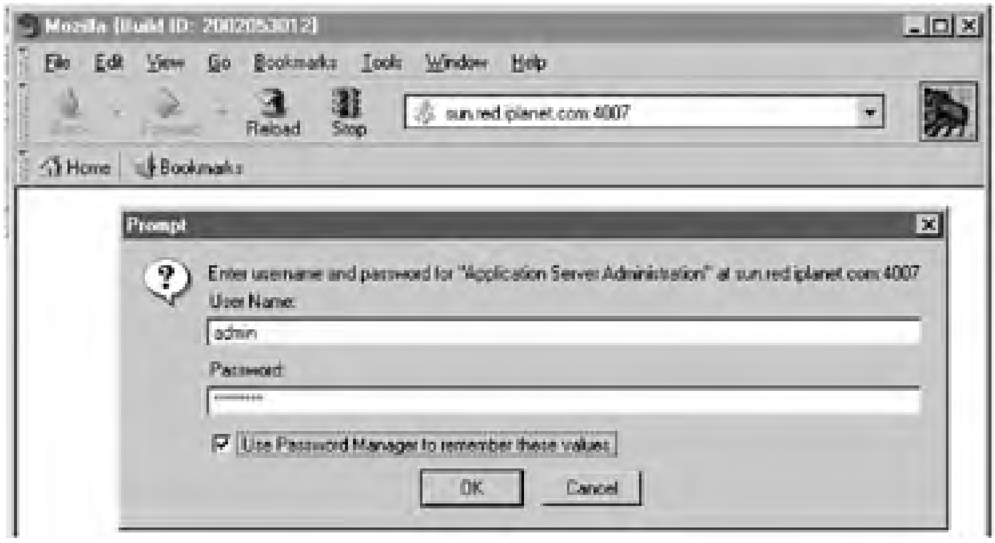
Use this command to start the server processes:

```
#  /sunone/as7/appserv/bin/asadmin start-appserv
Unable to read system environment. No system environ-
  ment will be used.
Instance domain1:admin-server started
Instance domain1:server1 started
Domain domain1 Started.
#
```

Start a web browser and connect to the administration server using URL *http://sun.red.iplanet.com:4007*.

The administration server returns a prompt for the administrator's user ID and password, as in Figure 14.3. Enter *admin* for the *User Name* and *password* for the *Password*.

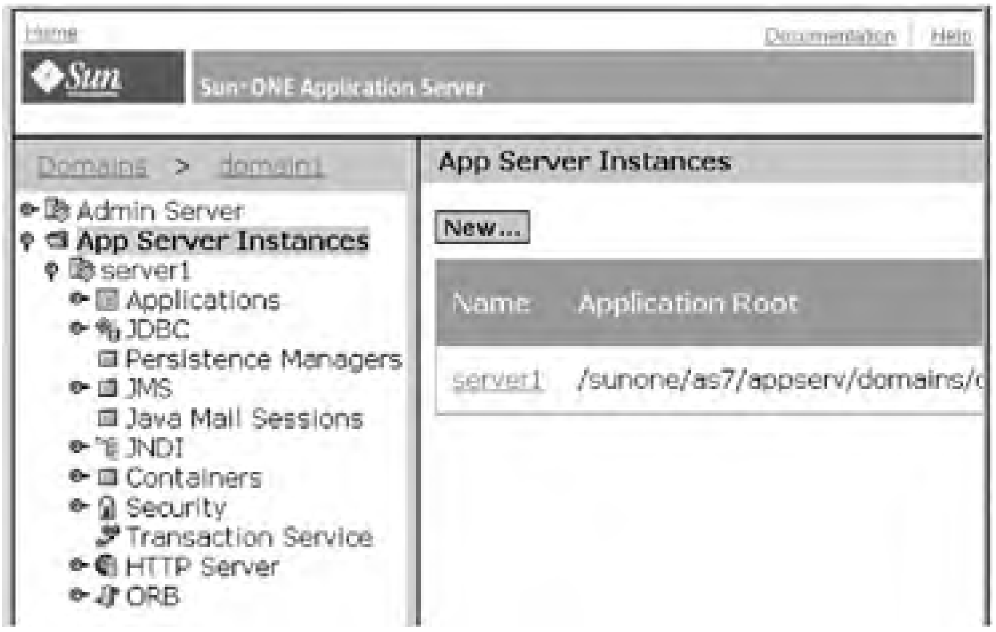
Figure 14.3. Administration prompt.


Mozilla {Build ID: 2002053012}
File Edit View Go Bookmarks Tools Window Help
Reload Stop
sun.red.iplanet.com:4007
Home Bookmarks
Prompt
OK Cancel

Enter username and password for "Application Server Administration" at sun.red.iplanet.com:4007
User Name:
admin
Password:
****
☑ Use Password Manager to remember these values.

The main Sun ONE Application Server Administration Server web page appears (Figure 14.4).

Figure 14.4. The Sun ONE Application Server Administration Server main web page.



Name	Application Root
server1	/sunone/as7/appserv/domains/

Testing the Sun ONE Application Server Default Web Site

Figure 14.5 shows the installed default web site. From a browser, go to URL: *http://sun.red.iplanet.com*.

Figure 14.5. Default web site that was installed.

Sun ONE Application Server 7 - Mozilla {Build ID: 2002053012}
File Edit View Go Bookmarks Tools Window Help
Reload
http://sunred.iplanet.com/

Sun ONE Application Server 7, Standard Edition, Beta

Success! You have sucessfully installed a Sun ONE Application Server. The next chapter is about configuring and using this server.

Uninstalling the Server

When you need to, this is a very easy process. Stop the processes and remove the install directory and all subdirectories, using the example command `rm -r /sunone/ws6`. After the command completes, the Sun ONE Application Server will be completely removed/uninstalled.

Reference Documentation

The Sun Microsystems web site at *http://docs.sun.com* has a considerable amount of documentation. The link for the Sun ONE Application Server is *http://docs.sun.com/db/prod/sunone*.

Windows Installation—GUI Installation

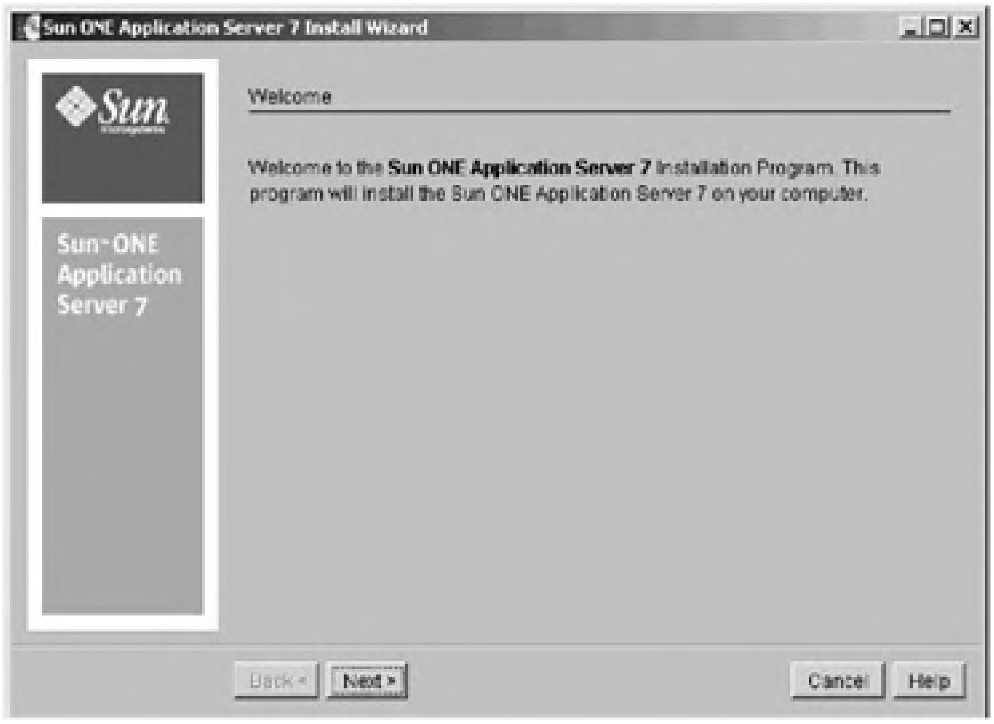
The opening of this chapter describes the components that will be installed and gives an overview of a sample installation setup. This section contains the steps for installing the Sun ONE Application Server on a computer running Windows 2000.

The Sun ONE Application Server for Windows is included on the DVD that comes with this book or you can download the Sun ONE

Application Server, with the latest service packs, from *www.sun.com*.

Run the setup executable file to start the installation. Zipped files are unpacked, and then the welcome window appears. Figure 14.6 shows the welcome page that opens when installing the Sun ONE Application Server.

Figure 14.6. Sun ONE Application Server installation welcome page.

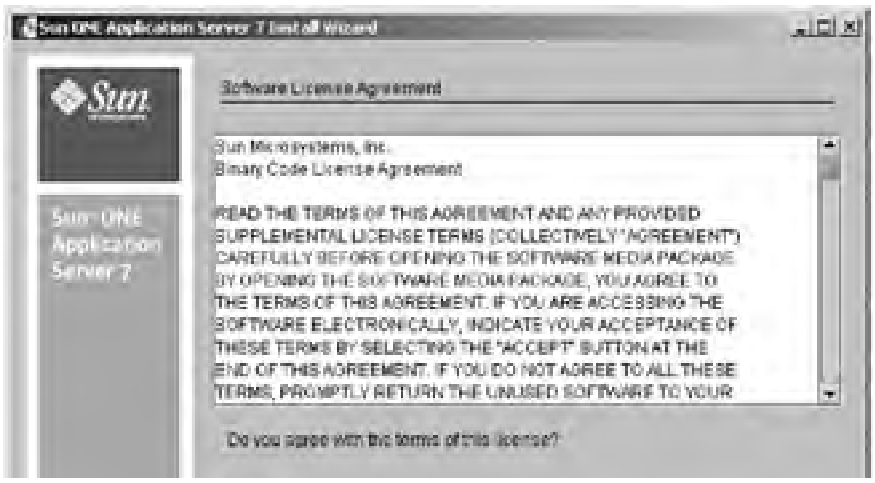

Sun ONE Application Server 7 Install Wizard
Sun
Sun ONE Application Server 7
Back <
Next >
Cancel
Help

Welcome

Welcome to the **Sun ONE Application Server 7** Installation Program. This program will install the Sun ONE Application Server 7 on your computer.

Click the *Next* button, and the Software License Agreement opens (Figure 14.7).

Figure 14.7. License agreement.


Sun ONE Application Server 7 Install Wizard
Sun
Sun ONE Application Server 7

Software License Agreement

Sun Microsystems, Inc.
Binary Code License Agreement

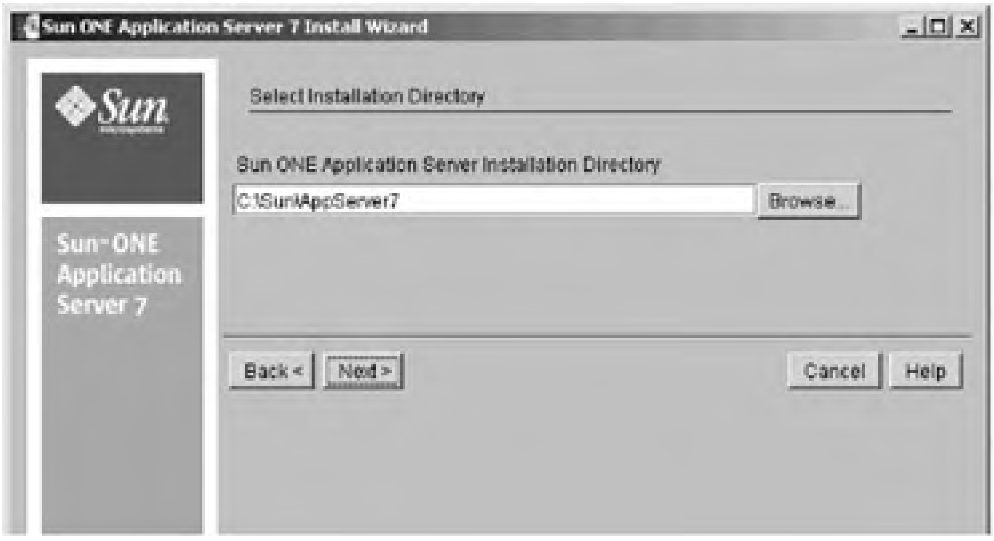
READ THE TERMS OF THIS AGREEMENT AND ANY PROVIDED SUPPLEMENTAL LICENSE TERMS (COLLECTIVELY "AGREEMENT") CAREFULLY BEFORE OPENING THE SOFTWARE MEDIA PACKAGE. BY OPENING THE SOFTWARE MEDIA PACKAGE, YOU AGREE TO THE TERMS OF THIS AGREEMENT. IF YOU ARE ACCESSING THE SOFTWARE ELECTRONICALLY, INDICATE YOUR ACCEPTANCE OF THESE TERMS BY SELECTING THE "ACCEPT" BUTTON AT THE END OF THIS AGREEMENT. IF YOU DO NOT AGREE TO ALL THESE TERMS, PROMPTLY RETURN THE UNUSED SOFTWARE TO YOUR

Do you agree with the terms of this license?

If you accept the terms of the agreement, choose *Yes*, and then click *Next*.

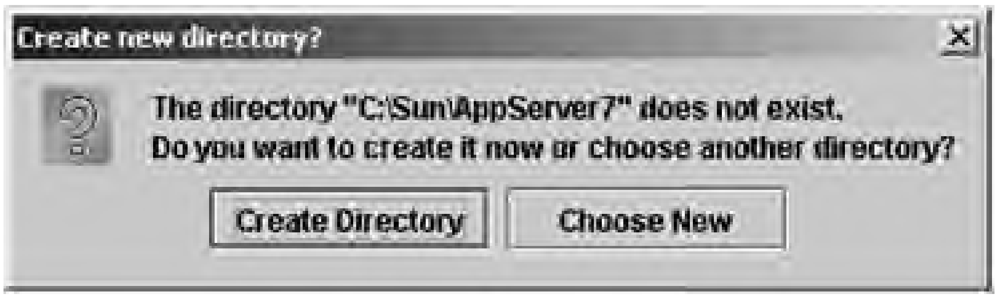
In the *Install Wizard* window that opens (Figure 14.8), select the installation directory (I am using the default directory). Then click *Next*.

Figure 14.8. Choosing the installation directory.


Sun ONE Application Server 7 Install Wizard
Sun-ONE Application Server 7
Select Installation Directory
Sun ONE Application Server Installation Directory
C:\Sun\AppServer7
Browse...
Back <
Next >
Cancel
Help

In the next window (see Figure 14.9), click *Create Directory*.

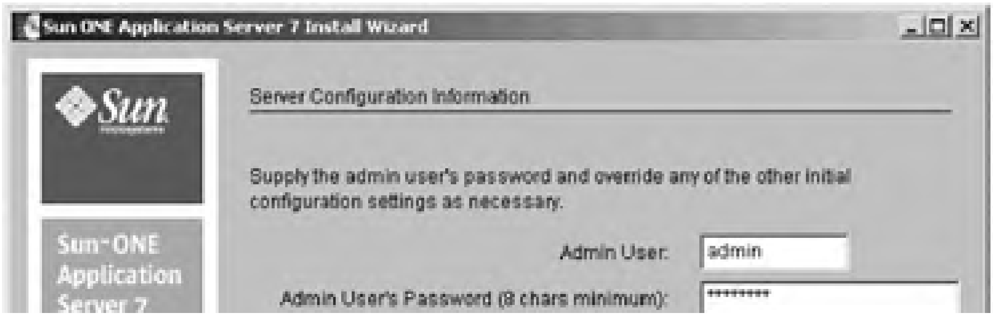
Figure 14.9. Creating the new directory.



The directory "C:\Sun\AppServer7" does not exist.
Do you want to create it now or choose another directory?

Back in the *Install Wizard* window enter installation information as shown in Figure 14.10. Enter the administration user ID and password. I'm using *admin* for the user and *password* for the password. Enter the administration server port of *4007* and application server instance HTTP port of *80*. Then click *Next*.

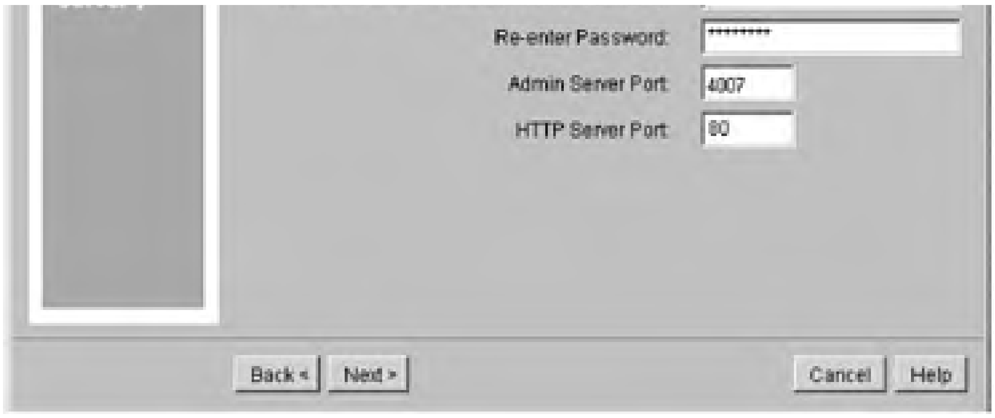
Figure 14.10. Server installation information.


Sun ONE Application Server 7 Install Wizard
Sun-ONE Application Server 7
Server Configuration Information

Supply the admin user's password and override any of the other initial configuration settings as necessary.

Admin User: admin

Admin User's Password (8 chars minimum): ********



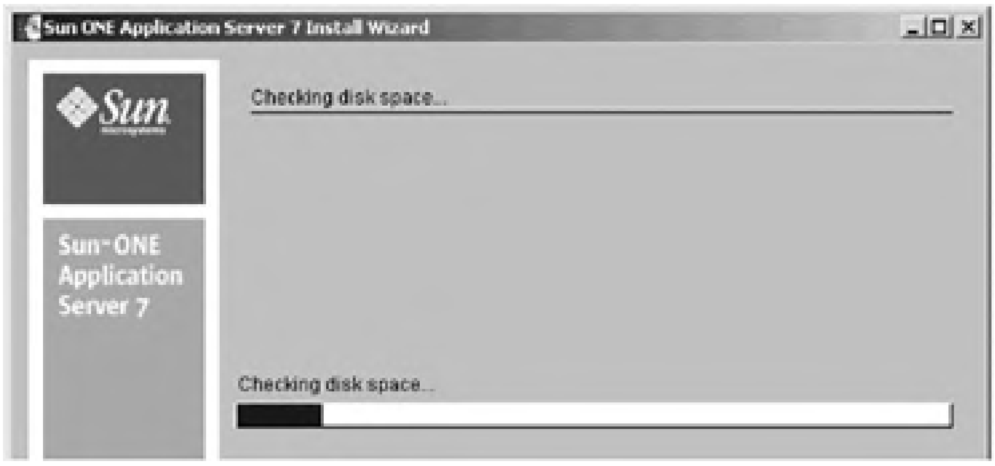
Re-enter Password: *********

Admin Server Port: 4007

HTTP Server Port: 80

The next window will indicate that the installation program is checking for available disk space (Figure 14.11).

Figure 14.11. Checking for available disk space.



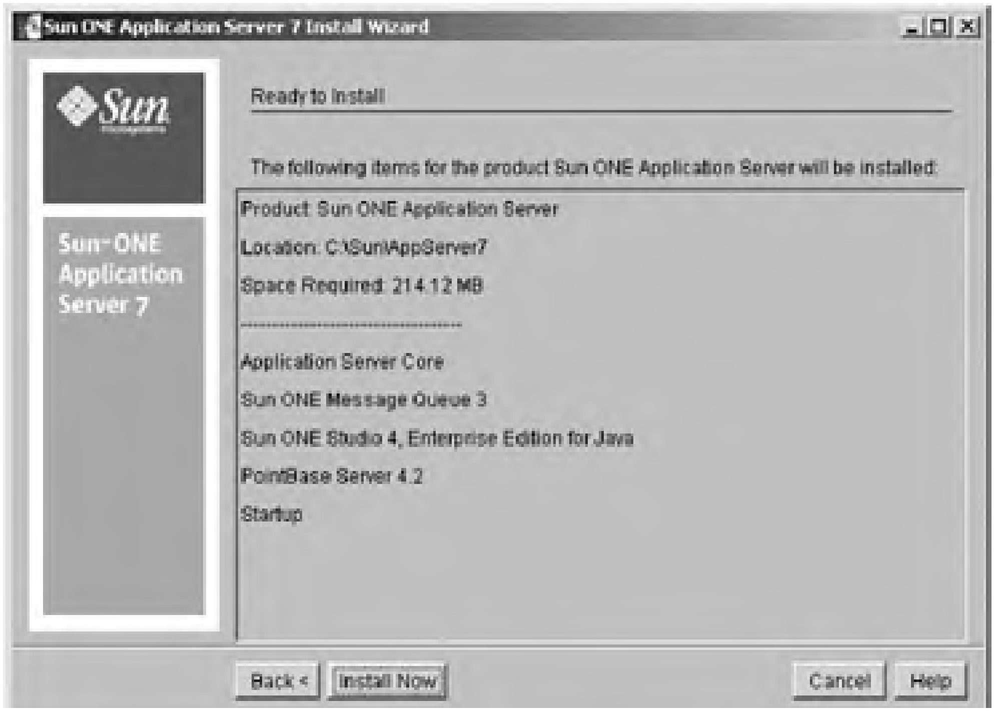
Sun ONE Application Server 7 Install Wizard
Sun
Sun ONE Application Server 7

Checking disk space...

Checking disk space...

When the *Ready to Install* window opens (Figure 14.12), click *Install Now*, and files will be installed (Figure 14.13).

Figure 14.12. Ready to Install window.



Sun ONE Application Server 7 Install Wizard
Sun
Sun ONE Application Server 7

Ready to Install

The following items for the product Sun ONE Application Server will be installed:

Product: Sun ONE Application Server

Location: C:\Sun\AppServer7

Space Required: 214.12 MB

Application Server Core

Sun ONE Message Queue 3

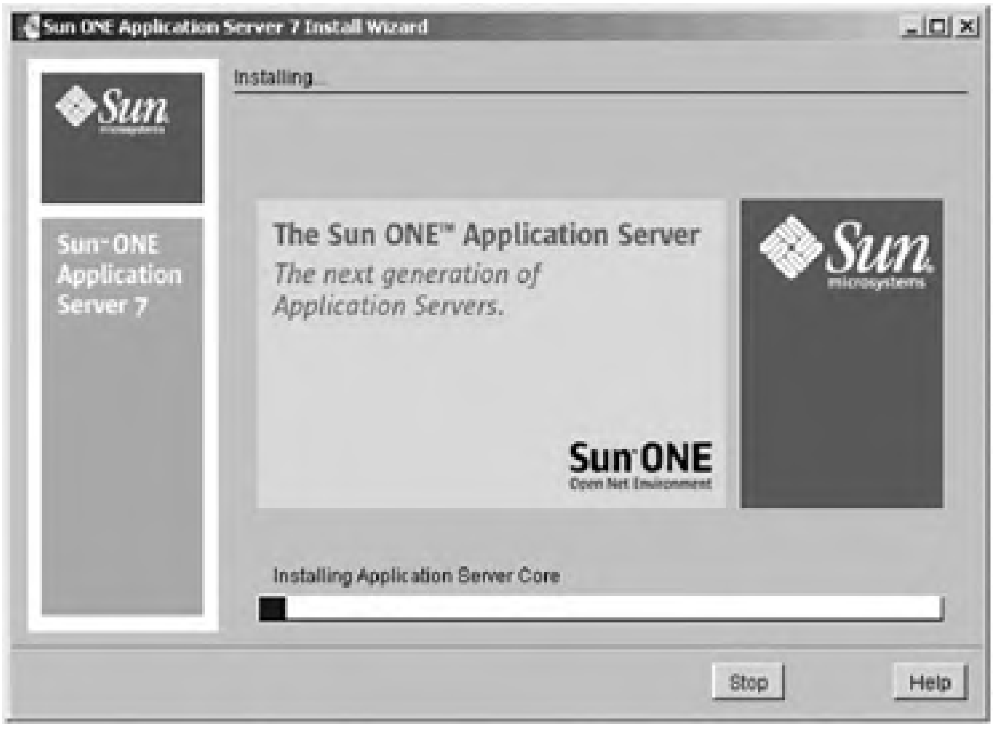
Sun ONE Studio 4, Enterprise Edition for Java

PointBase Server 4.2

Startup

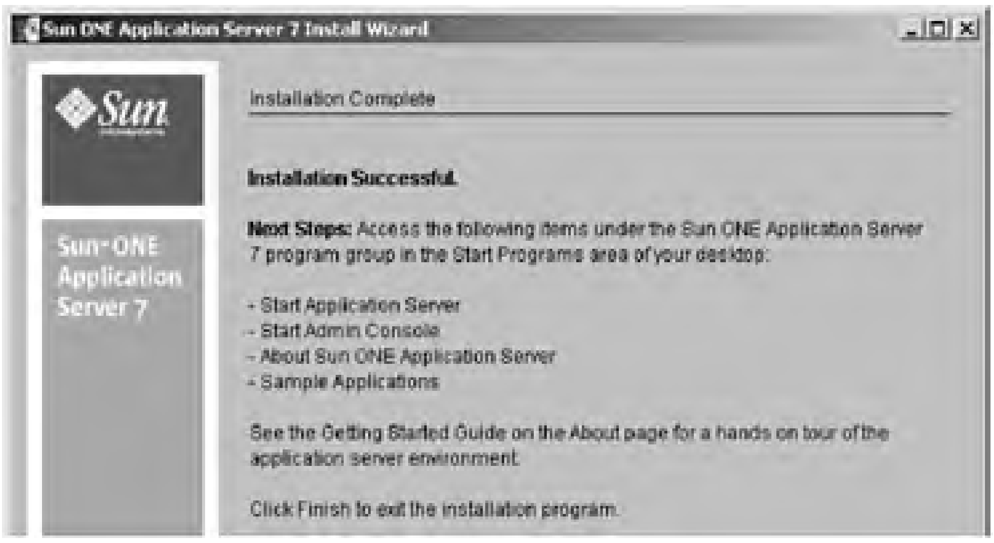
Back <
Install Now
Cancel
Help

Figure 14.13. Files are installing.


Sun ONE Application Server 7 Install Wizard
Installing...
Sun
Sun ONE Application Server 7
The Sun ONE™ Application Server
The next generation of Application Servers.
Sun ONE
Open Net Environment
Sun microsystems
Installing Application Server Core
Stop
Help

When the installation is complete (Figure 14.14), click *Finish*.

Figure 14.14. Successful installation.


Sun ONE Application Server 7 Install Wizard
Sun
Sun ONE Application Server 7
Installation Complete

Installation Successful.

Next Steps: Access the following items under the Sun ONE Application Server 7 program group in the Start Programs area of your desktop:

- Start Application Server
- Start Admin Console
- About Sun ONE Application Server
- Sample Applications

See the Getting Started Guide on the About page for a hands on tour of the application server environment.

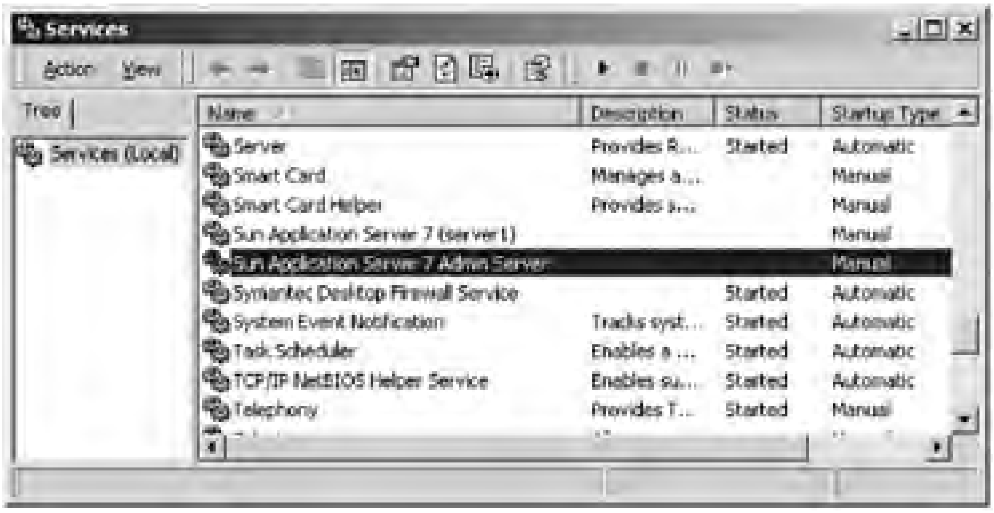
Click Finish to exit the installation program.

Starting the Services for Testing

First, check for new services. Go to the *Control Panel* (*Start / Settings / Control Panel*).

If you are using Windows 2000, when the *Control Panel* comes up, click on *Administrative Tools*; then click on *Services*. In the *Services* window (Figure 14.15), you will now find Sun ONE Application Server processes. I have set the *Startup Type* to *Manual* for our example because I do not want the application server starting every time I reboot my computer. I want the server to start only when I want to work with it.

Figure 14.15. Services window from Windows 2000.



Name	Description	Status	Startup Type
Server	Provides R...	Started	Automatic
Smart Card	Manages a...		Manual
Smart Card Helper	Provides s...		Manual
Sun Application Server 7 (server1)			Manual
Sun Application Server 7 Admin Server			Manual
Symantec Desktop Firewall Service		Started	Automatic
System Event Notification	Tracks syst...	Started	Automatic
Task Scheduler	Enables a ...	Started	Automatic
TCP/IP NetBIOS Helper Service	Enables su...	Started	Automatic
Telephony	Provides T...	Started	Manual

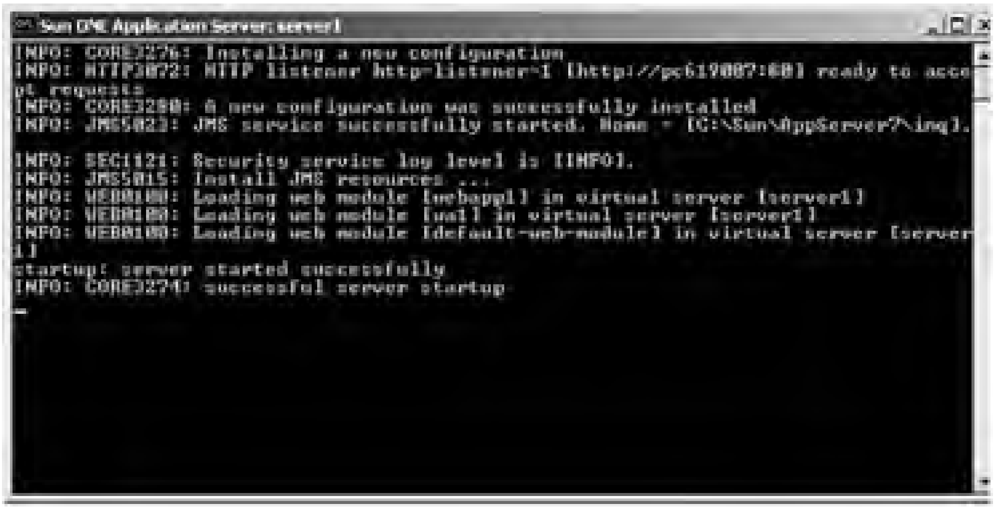
Testing the Application Server by Connecting to the Administration Server

Start the application server from the *Services* window (Figure 14.15) or from the command line. For example, from the file explorer, double-click on c:\sun\AppServer7\asadmin.bat. From the command window, type the following:

```
Use "exit" to exit and "help" for online help
asadmin>start-appserv
Instance admin-server started
Instance server1 started
asadmin>
```

A command window appears when starting the application server instance (an example is shown in Figure 14.16). If you close this window, the application server instance dies, and it is restarted by a watchdog process.

Figure 14.16. A command window.

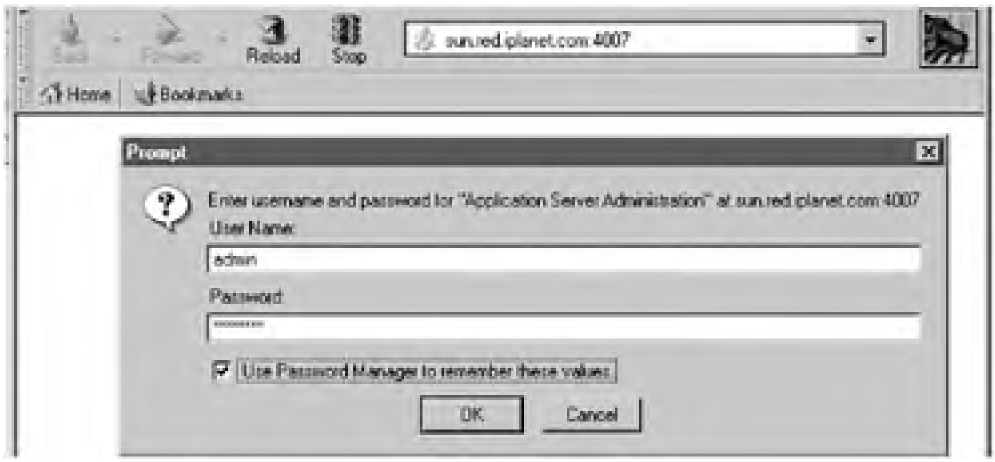


```
INFO: CORE3276: Installing a new configuration
INFO: HTTP3072: HTTP listener http-listener-1 [http://pc619007:80] ready to acce
pt requests
INFO: CORE3280: A new configuration was successfully installed
INFO: JMS5023: JMS service successfully started. Home = [C:\Sun\AppServer7\imq].

INFO: SEC1121: Security service log level is [INFO].
INFO: JMS5015: Install JMS resources ...
INFO: WEB0100: Loading web module [webapp1] in virtual server [server1]
INFO: WEB0100: Loading web module [ual] in virtual server [server1]
INFO: WEB0100: Loading web module [default-web-module] in virtual server [server
1]
startup: server started successfully
INFO: CORE3274: successful server startup
_
```

Start a web browser and connect to the administration server using URL *http://sun.red.iplanet.com:4007*.

The administration server returns a prompt for the administrator's user ID and password (Figure 14.17). Enter *admin* for the *User Name* and *password* for the *Password*.

Figure 14.17. Administration prompt.


Prompt

Enter username and password for "Application Server Administration" at sun.red.iplanet.com:4007

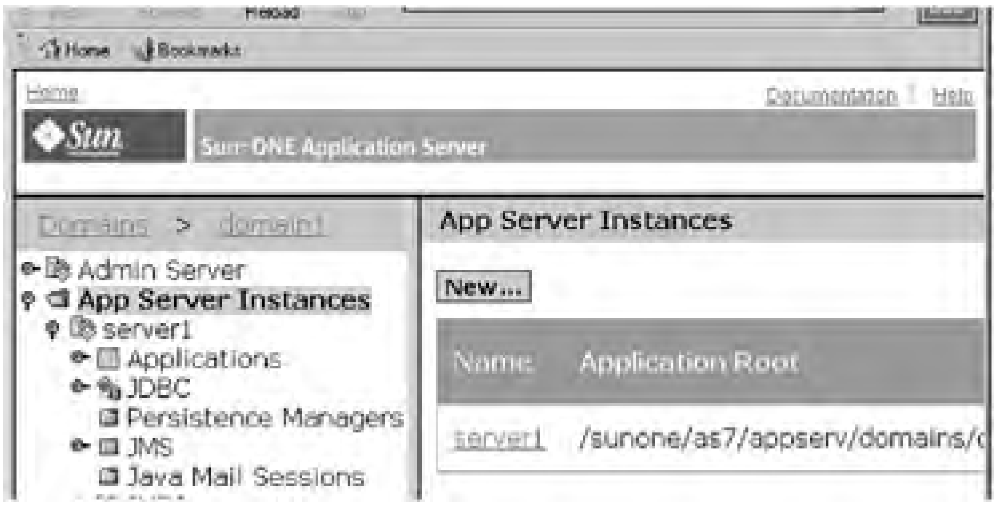
User Name: admin

Password: ********

[✓] Use Password Manager to remember these values

The main Sun ONE Application Server Administration Server web page appears (Figure 14.18).

Figure 14.18. Application server main web page.

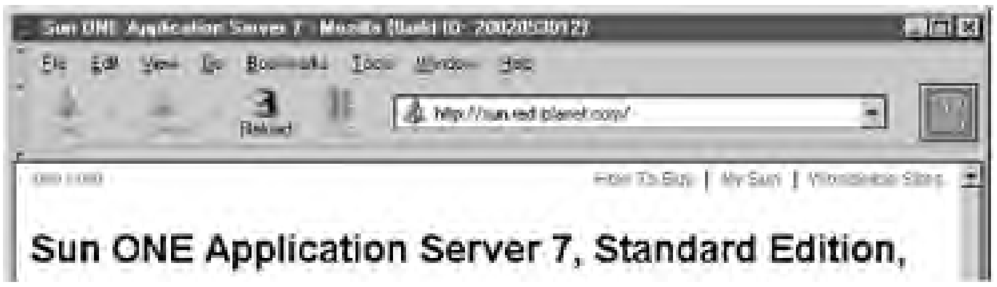


Name	Application Root
server1	/sunone/as7/appserv/domains/

Testing the Sun ONE Application Server Default Web Site

From a browser, go to URL: *http://sun.red.iplanet.com* (Figure 14.19).

Figure 14.19. Default web site.


Sun ONE Application Server 7 - Mozilla {Build ID: 2002053012}
File
Edit
View
Go
Bookmarks
Tools
Window
Help
Reload
http://sun.red.iplanet.com/
How To Buy
My Sun
Worldwide Sites
Sun ONE Application Server 7, Standard Edition,

Success! Now that you have sucessfully installed a Sun ONE Application Server, go to the next chapter to configure and use this server.

Uninstalling the Server

When you need to, this is a very easy process. Run the uninstall option from the Windows *Control Panel Add/Remove* option. After the uninstall has been run, remove the installation directory. At this point, the Sun ONE Application Server has been completely removed/uninstalled.

Reference Documentation

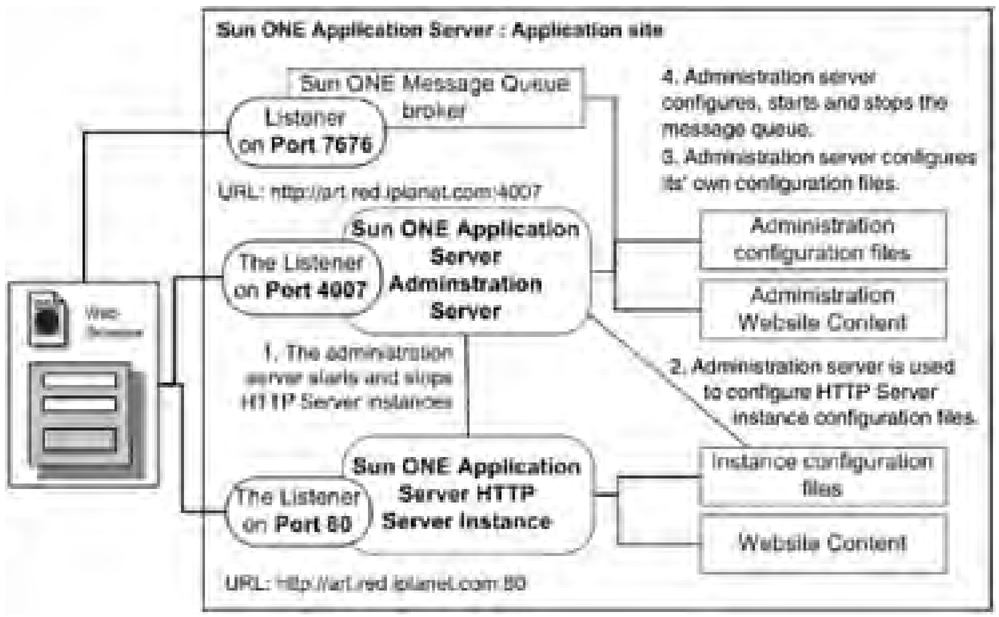
The Sun Microsystems web site at *http://docs.sun.com* has a considerable amount of documentation.

The link for the Sun ONE Directory Server is at *http://docs.sun.com/db/prod/sunone*

Chapter 15. Starting, Stopping, and Testing the Sun ONE Application Server

After installing new software, we want to be able to start it, view the processes, and stop the processes. During the installation of the Sun ONE Application Server, two HTTP (web) servers and one message queue are installed (Figure 15.1). In this chapter, the server processes are started, viewed, tested, and stopped.

Figure 15.1. Installed services.


Sun ONE Application Server : Application site
Sun ONE Message Queue broker
Listener on Port 7676
4. Administration server configures, starts and stops the message queue.
3. Administration server configures its' own configuration files.
URL: http://art.red.iplanet.com:4007
Sun ONE Application Server Administration Server
The Listener on Port 4007
Administration configuration files
Administration Website Content
Web Browser
1. The administration server starts and stops HTTP Server instances
2. Administration server is used to configure HTTP Server instance configuration files.
Sun ONE Application Server HTTP Server Instance
The Listener on Port 80
Instance configuration files
Website Content
URL: http://art.red.iplanet.com:80

Starting and Stopping Server Processes in UNIX

All the Application Server Processes

Start the Sun ONE Application Server processes using the administration command-line tool.

```
# /sunone/as7/appserv/bin/asadmin start-appserv
Instance domain1:admin-server started
Instance domain1:server1 started
Domain domain1 Started.
#
```

asadmin is an administration tool capable of doing everything that can be done with the Sun ONE Application Server Administration Server. Run the command tool and enter help at the asadmin prompt to see a list of asadmin commands.

```
# /sunone/as7/appserv/bin/asadmin
asadmin>help

User Commands                    help(1)
NAME
     help - displays a list of all the commands
     available in the Command-line interface.
... (commands listed)

asadmin>
```

Now get help on one of the commands:

```
jasadmin>help jms-ping
jms-ping          - Will ascertain if jms provider is
  up and running
Usage: jms-ping --user admin_user [--password
  admin_password] [--host localhost] [--port 4848] [--
  secure | -s] instancename
asadmin>
```

and exit the tool:

```
jasadmin> exit
```

Stop the Sun ONE Application Server processes using the administration command-line tool.

```
# /sunone/as7/appserv/bin/asadmin stop-appserv
Instance domain1:admin-server stopped
Instance domain1:server1 stopped
Domain domain1 Stopped.
#
```

Using these commands, all the processes are started and stopped. To start and stop each of the servers individually, specific server commands must be used.

The Administration Server

Start the Sun ONE Application Administration Server.

```
# /sunone/as7/appserv/domains/domain1/admin-
  server/bin/startserv
CORE1116: Sun ONE Application Server 7.0
INFO: CORE3016: daemon is running as super-user
INFO: j2eerunner.printstartinfo
INFO: ADM0002:System MBean initialized:[ias:type=
  controller]
INFO: ADM0002:System MBean initialized:[ias:type=
  configurator]
INFO: ADM0001:MBeanServer initialized successfully
INFO: ADM0005:Timestamp files for configuration created
  for:[admin-server]
INFO: ADM0005:Timestamp files for configuration created
  for:[server1]
INFO: CIS0056: Creating TCP ServerConnection at [End-
  Point [IIOP_CLEAR_TEXT:1.1.1.7:1071:false]]
```

```
INFO: CIS0057: Created TCP ServerConnection at [End-
  Point [IIOP_CLEAR_TEXT:1.1.1.7:1071:false]]
INFO: CIS0054: Creating TCP Connection from [-] to [End-
  Point [IIOP_CLEAR_TEXT:1.1.1.7:1071:false]]
INFO: CIS0054: Created TCP Connection from [EndPoint
  [IIOP_CLEAR_TEXT:1.1.1.7:33011:false]] to [EndPoint
  [IIOP_CLEAR_TEXT:1.1.1.7:1071:false]]
INFO: IOP5053: Received a locate request on a disabled
  connection. Locate requests are permitted.
INFO: RSR5060: Install JDBC Datasources ...
INFO: JMS5015: Install JMS resources ...
INFO: WEB0100: Loading web module [adminapp:admin-
  app.war] in virtual server [admin-server]
INFO: WEB0100: Loading web module [admingui:admin-
  GUI.war] in virtual server [admin-server]
INFO: HTTP3072: HTTP listener http-listener-1
  [http://art:4007] ready to accept requests startup:
  server started successfully
#
```

Monitor the Sun ONE Application Server Administration Server processes.

```
# ps -ef | grep sunone
root   847     1  0 17:10:21 ?        0:00 ./appservd-
  wdog -r /sunone/as7/appserv -d /sunone/as7/appserv
  /domains/domain1/a
root   848   847  0 17:10:21 ?        0:01 appservd -r
  /sunone/as7/appserv -d /sunone/as7/appserv/domains
  /domain1/admin-se
root   849   848  0 17:10:23 ?        1:28 appservd -r
  /sunone/as7/appserv -d /sunone/as7/appserv/domains
  /domain1/admin-se
root   851   849  0 17:10:27 ?        0:00
  /sunone/as7/appserv/lib/Cgistub -f /tmp/admin-server-
  7a4e6e17/.cgistub_849
root   852   851  0 17:10:27 ?        0:00
  /sunone/as7/appserv/lib/Cgistub -f /tmp/admin-server-
  7a4e6e17/.cgistub_849
root   853   851  0 17:10:27 ?        0:00
  /sunone/as7/appserv/lib/Cgistub -f /tmp/admin-server-
  7a4e6e17/.cgistub_849
```

Notes about the listing:

- Process 849 appservd is the Sun ONE Application Server Administration Server web instance process. This process is a child of 848.
- Process 848 appservd is the parent process of the Sun ONE Application Server Administration Server process. This process has two functions: to start the actual web instance; and to watch/monitor the child process and, if the child process stops running, to restart the process.
- Process 847 (wdog) is the watchdog process to restart the parent web server instance process (process 848) if it stops.

The shutdown/stop command is in the same directory as the start command:

```
# /sunone/as7/appserv/domains/domain1/admin-server/bin
  /stopserv
server has been shutdown
```

The HTTP (Web) Server Instance

Start the Sun ONE Application Server HTTP server instance.

```
# /sunone/as7/appserv/domains/domain1/server1/bin
  /startserv
```

```
CORE1116: Sun ONE Application Server 7.0
INFO: CORE3016: daemon is running as super-user
INFO: j2eerunner.printstartinfo
INFO: JMS5023: JMS service successfully started. Home =
  [/sunone/as7/imq/bin/..].
INFO: CIS0056: Creating TCP ServerConnection at [End-
  Point [IIOP_CLEAR_TEXT:1.1.1.7:3700:false]]
INFO: CIS0057: Created TCP ServerConnection at [End-
  Point [IIOP_CLEAR_TEXT:1.1.1.7:3700:false]]
INFO: CIS0054: Creating TCP Connection from [-] to [End-
  Point [IIOP_CLEAR_TEXT:1.1.1.7:3700:false]]
INFO: CIS0054: Created TCP Connection from [EndPoint
  [IIOP_CLEAR_TEXT:1.1.1.7:33065:false]] to [EndPoint
  [IIOP_CLEAR_TEXT:1.1.1.7:3700:false]]
INFO: RSR5060: Install JDBC Datasources ...
INFO: JMS5015: Install JMS resources ...
INFO: WEB0100: Loading web module [default-web-module]
  in virtual server [server1]
INFO: HTTP3072: HTTP listener http-listener-1
  [http://art:80] ready to accept requests
startup: server started successfully
```

Monitor the Sun ONE Application Server web server instance processes.

```
# ps -ef | grep sunone
root  1160     1  0 19:24:48 ? 0:00 ./appservd-wdog -r
  /sunone/as7/appserv -d /sunone/as7/apps-
  erv/domains/domain1/s
root  1161  1160  0 19:24:48 ? 0:01 appservd -r
  /sunone/as7/appserv -d /sunone/as7/apps-
  erv/domains/domain1/server1/
root  1162  1161 41 19:24:49 ? 0:53 appservd -r
  /sunone/as7/appserv -d /sunone/as7/apps-
  erv/domains/domain1/server1/
root  1164  1162  0 19:24:52 ? 0:00 /sunone/as7/apps-
  erv/lib/Cgistub -f /tmp/server1-7a4e6e17/.cgistub_1162
root  1165  1164  0 19:24:52 ? 0:00 /sunone/as7/apps-
  erv/lib/Cgistub -f /tmp/server1-7a4e6e17/.cgistub_1162
root  1166  1164  0 19:24:52 ? 0:00 /sunone/as7/apps-
  erv/lib/Cgistub -f /tmp/server1-7a4e6e17/.cgistub_1162
root  1168  1162  0 19:25:03 ? 0:00 /bin/sh
  /sunone/as7/imq/bin/../bin/imqbrokerd -javahome
  /sunone/as7/jdk -name d
root  1187  1168  3 19:25:05 ? 0:10
  /sunone/as7/jdk/jre/bin/java -server -cp
  /sunone/as7/imq/bin/../bin/../lib/imqb
```

Notes about the listing:

- Process 1162 appservd is the Sun ONE Application Server web server instance process. This process is a child of 1161 appservd.
- Process 1161 appservd is a parent process to the Sun ONE Application Server Administration Server process. This process has two functions: to start the actual web instance; to watch/monitor the child process and, if the child process stops running, to restart the process.
- Process 1160 (wdog) is the watchdog process to restart the parent web server instance process (process 1161) if it stops.
- Process 1168 (imqbrokerd) is the Sun ONE Message Queue broker process.
- Process 1187 (imqb) is another Sun ONE Message Queue process.

If you are not using the message queue, the message queue processes can be killed and the Sun ONE Application Server will keep working.

The shutdown/stop command is in the same directory as the start command:

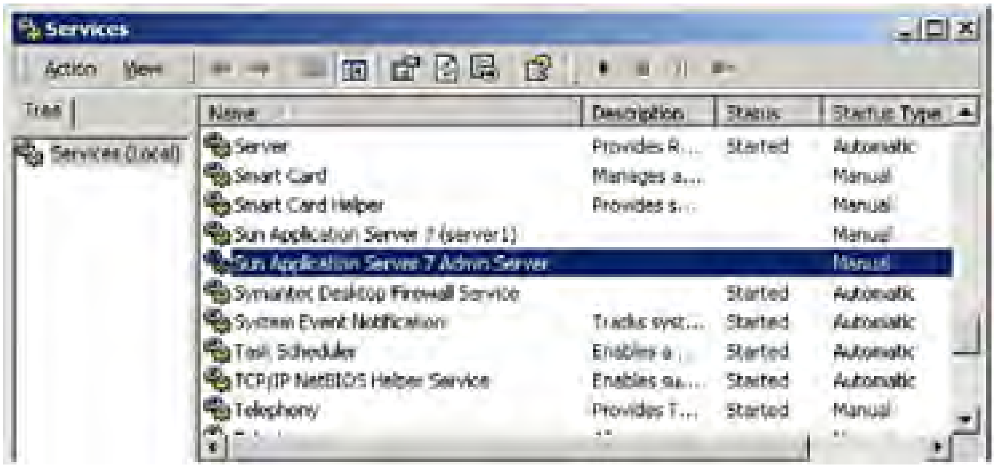
```
# /sunone/as7/appserv/domains/domain1/server1/bin
  /stopserv
server has been shutdown
```

Starting and Stopping Server Processes in Windows

Windows has two methods for starting and stopping Sun One Application Server processes. The first method is to use the *Services* window, and the second is to use the command-line tools that come with the Sun ONE Application Server.

To use the *Services* window, go to the *Control Panel* by clicking *Start / Settings / Control Panel / Services*. (On Windows 2000, when the *Control Panel* comes up, click on *Administrative Tools*; then click on *Services*.) In the *Services* window, you will see the Sun ONE Application Server processes as shown in Figure 15.2. The *Startup Type* is set to *Manual* for our example because I do not want the application server starting every time I reboot my computer. The server should start only when we are working with it.

Figure 15.2. Services window from Windows 2000.



Name	Description	Status	Startup Type
Server	Provides R...	Started	Automatic
Smart Card	Manages a...		Manual
Smart Card Helper	Provides s...		Manual
Sun Application Server 7 (server1)			Manual
Sun Application Server 7 Admin Server			Manual
Symantec Desktop Firewall Service		Started	Automatic
System Event Notification	Tracks syst...	Started	Automatic
Task Scheduler	Enables a...	Started	Automatic
TCP/IP NetBIOS Helper Service	Enables su...	Started	Automatic
Telephony	Provides T...	Started	Manual

The second method of starting and stopping uses the command-line tools that come with the application server. From the file explorer, double-click on

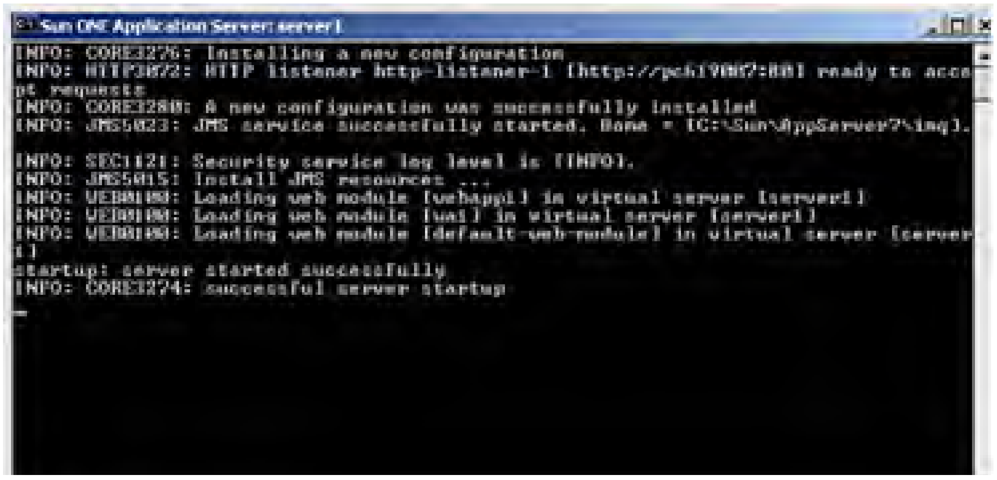
```
c:\sun\AppServer7\asadmin.bat
```

When the window comes up, type `start-appserv` as in

```
Use "exit" to exit and "help" for online help
asadmin>start-appserv
Instance admin-server started
Instance server1 started
asadmin>
```

Note: A command window appears when starting the application server instance (see example in Figure 15.3). If you close this window, the application server instance dies, and it is restarted by a watchdog process.

Figure 15.3. A command window.



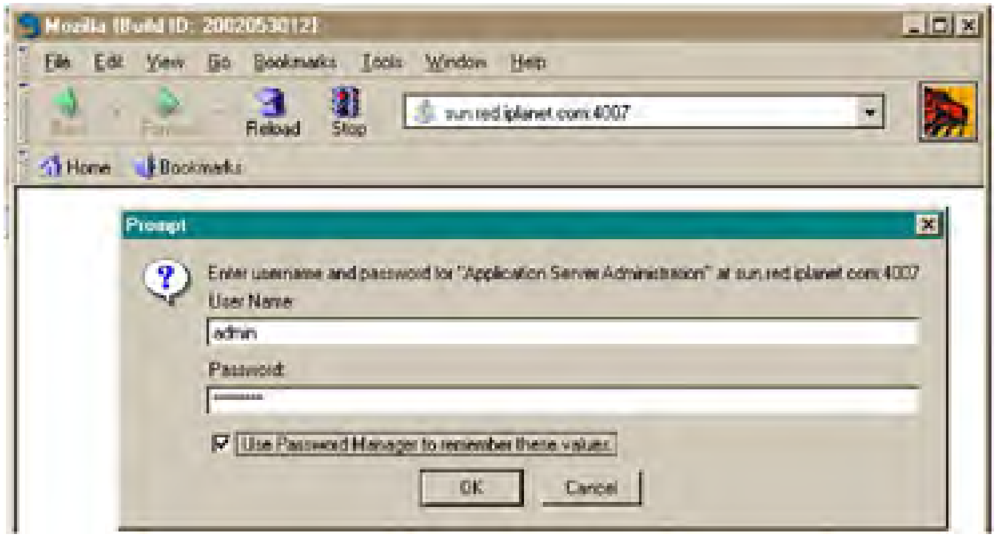
```
INFO: CORE3276: Installing a new configuration
INFO: HTTP3072: HTTP listener http-listener-1 [http://pc619007:80] ready to acce
pt requests
INFO: CORE3280: A new configuration was successfully installed
INFO: JMS5023: JMS service successfully started. Home = [C:\Sun\AppServer7\imq].

INFO: SEC1121: Security service log level is [INFO].
INFO: JMS5015: Install JMS resources ...
INFO: WEB0100: Loading web module [webapp1] in virtual server [server1]
INFO: WEB0100: Loading web module [uai] in virtual server [server1]
INFO: WEB0100: Loading web module [default-web-module] in virtual server [server
1]
startup: server started successfully
INFO: CORE3274: successful server startup
_
```

Now that the processes are running, start a web browser and connect to the administration server using the URL *http://sun.red.iplanet.com:4007*.

Sun ONE Application Server Administration Server

If the administration server is not running, start it now. Then, using a web browser, connect to the administration server. Note: The port values are based on the installation. If you followed the previous installation instructions, the user ID is *admin* and the password is *password* (Figure 15.4).

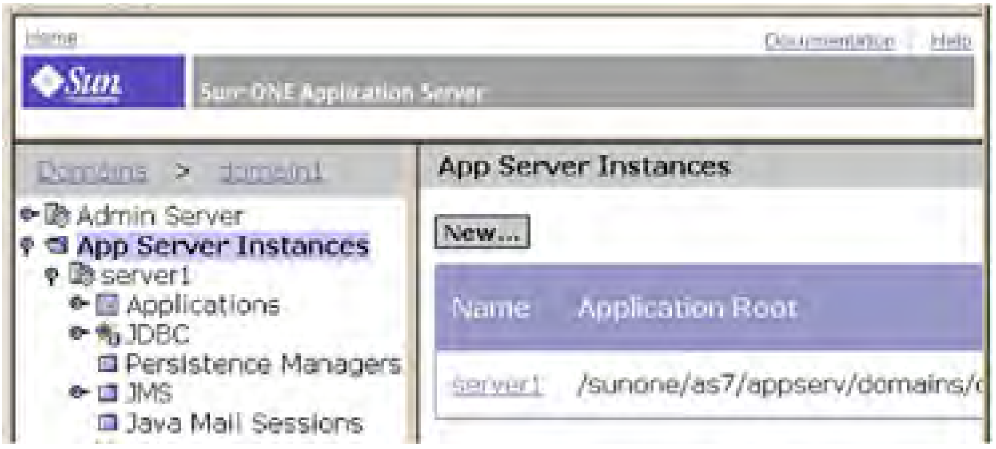
Figure 15.4. Prompt for administration user ID and password.



Enter username and password for "Application Server Administration" at sun.red.iplanet.com:4007
User Name
admin
Password
••••••••
[✓] Use Password Manager to remember these values.

Once logged in, the main/top web administration web page appears, from which web server instances are managed and configured (Figure 15.5).

Figure 15.5. Main web administration web page.

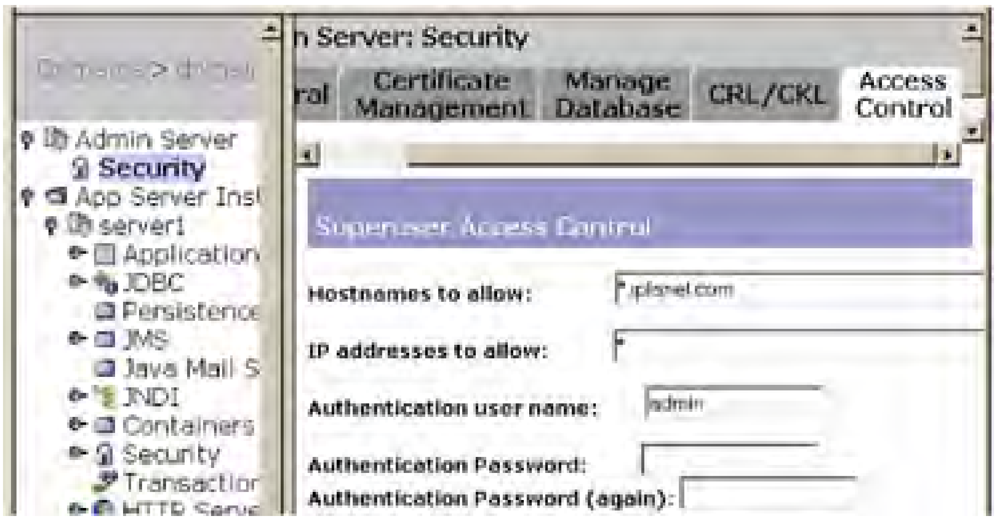


Name	Application Root
server1	/sunone/as7/appserv/domains/

Administering the Administrator Account

The administrator account is a special account that is maintained by the Sun ONE Application Server administration system. To maintain the *admin* password, from an administration web page, under the *Admin Server* branch (on the left side), click *Security*, and then click the *Access Control* tab (shown in Figure 15.6). The administrator's password is maintained from this web page.

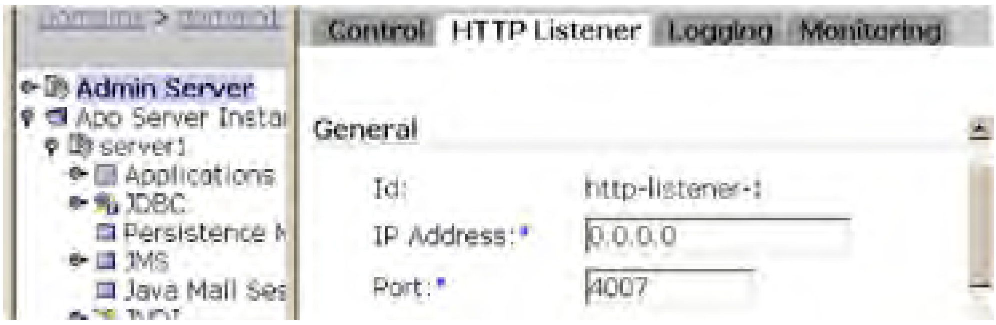
Figure 15.6. Superuser administration account.



Hostnames to allow: *.iplanet.com
IP addresses to allow: *
Authentication user name: admin
Authentication Password:
Authentication Password (again):

To view or change the application server administration server listener port, click the *Admin Server* branch, and then click the *HTTP Listener* tab (Figure 15.7). For now, do not change this value.

Figure 15.7. Administration server listener port.



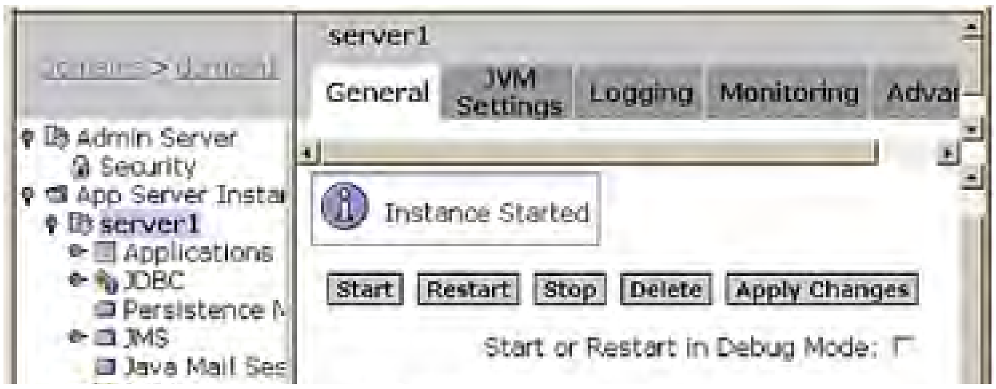
Id: http-listener-1
IP Address:* 0.0.0.0
Port:* 4007

An IP Address of *0.0.0.0* means that the listener listens on all the IP addresses of the computer. If the application server computer has only one IP address, then the listener listens to port 4007 on that address. If the computer has more than one IP address, the listener listens to port 4007 on all the IP addresses.

Web Server Tests

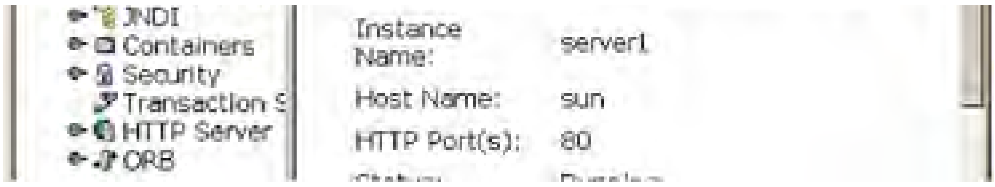
Since the application server has web (HTTP) server instances, the basic test is to turn the server on and surf to it with a browser. To turn the server on from the administration server, click the *server1* menu branch and click the *Start* button (Figure 15.8). Open another browser window (or browser tab) to display the default home page for this web server instance (Figure 15.9).

Figure 15.8. Starting and stopping the application server instance from the administration server.



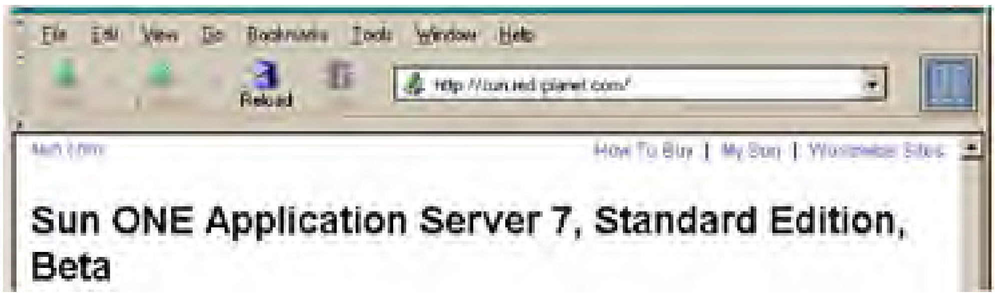
Instance Started

Start or Restart in Debug Mode: ☐



Instance Name: server1
Host Name: sun
HTTP Port(s): 80

Figure 15.9. The default home web page created when a server is installed.

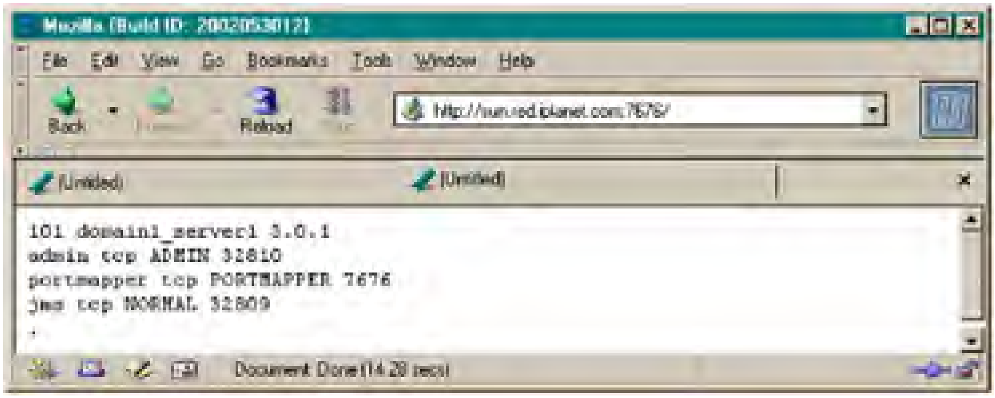

File Edit View Go Bookmarks Tools Window Help
Reload
http://sun.red.iplanet.com/
How To Buy | My Sun | Worldwide Sites

Sun ONE Application Server 7, Standard Edition, Beta

Java Message Queue Process Test

The Sun ONE Java Message Queue broker responds to a browser HTTP request. To test this process, call the broker from a browser using URL *http://sun.red.iplanet.com:7676*. The test results in the window are shown in Figure 15.10.

Figure 15.10. Testing the Java Message Queue process.


Mozilla (Build ID: 2002053012)
File Edit View Go Bookmarks Tools Window Help
Back
Reload
http://sun.red.iplanet.com:7676/
(Untitled)
(Untitled)
Document: Done (14.28 secs)

```
101 domain1_server1 3.0.1
admin tcp ADMIN 32810
portmapper tcp PORTMAPPER 7676
jms tcp NORMAL 32809
.
```

Summary

This chapter looked at each of the four functions of the Sun ONE Application Server Administration Server mentioned in Figure 15.10:

1. Starting and stopping HTTP server instances
2. Configuring HTTP server instance configuration files
3. Configuring its own configuration files
4. Configuring, starting, and stopping the message queue

You started the Sun ONE Application Server Administration Server both from a command on UNIX and from the *Service* window on Windows. You started and stopped

- All the application server processes
- The administration server processes
- The web instance processes

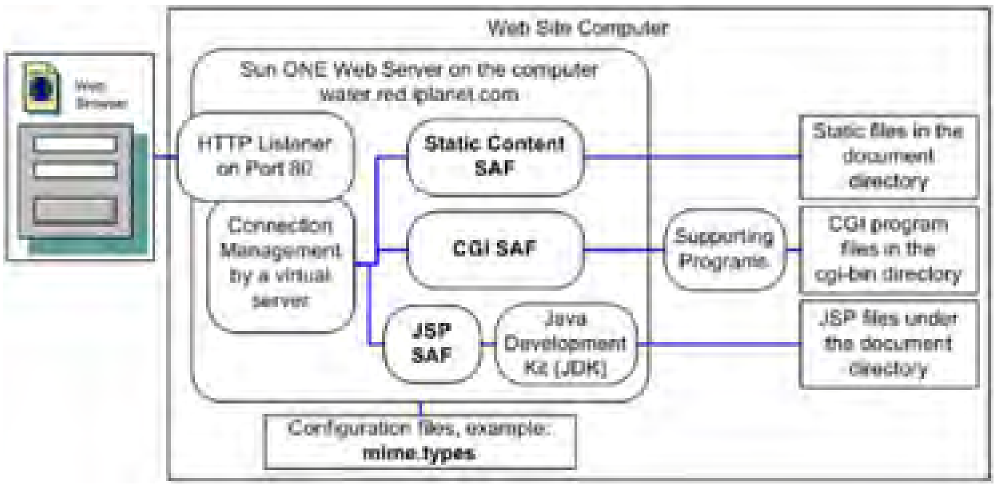
From the administration server web site, we can maintain administration web server configurations such as the listener settings and the administrator password. In a production system, the password is important—I use the user ID *admin* and the password *password*.

Chapter 16. Configuring the Sun ONE Application Server's Web Server Features

The Sun ONE Application Server 7 has a built-in web server that is based on the Sun ONE Web Server 6.0. In designing it this way, the developers retained the core features of the Sun One Web Server for the Sun ONE Application Server 7. Therefore, this server is not just an application server—it is also a full-featured, commercial-quality web server.

This chapter focuses on developing a web site with static content, CGI programs, and JSPs (Figure 16.1).

Figure 16.1. Sun ONE Application Server web SAFs.

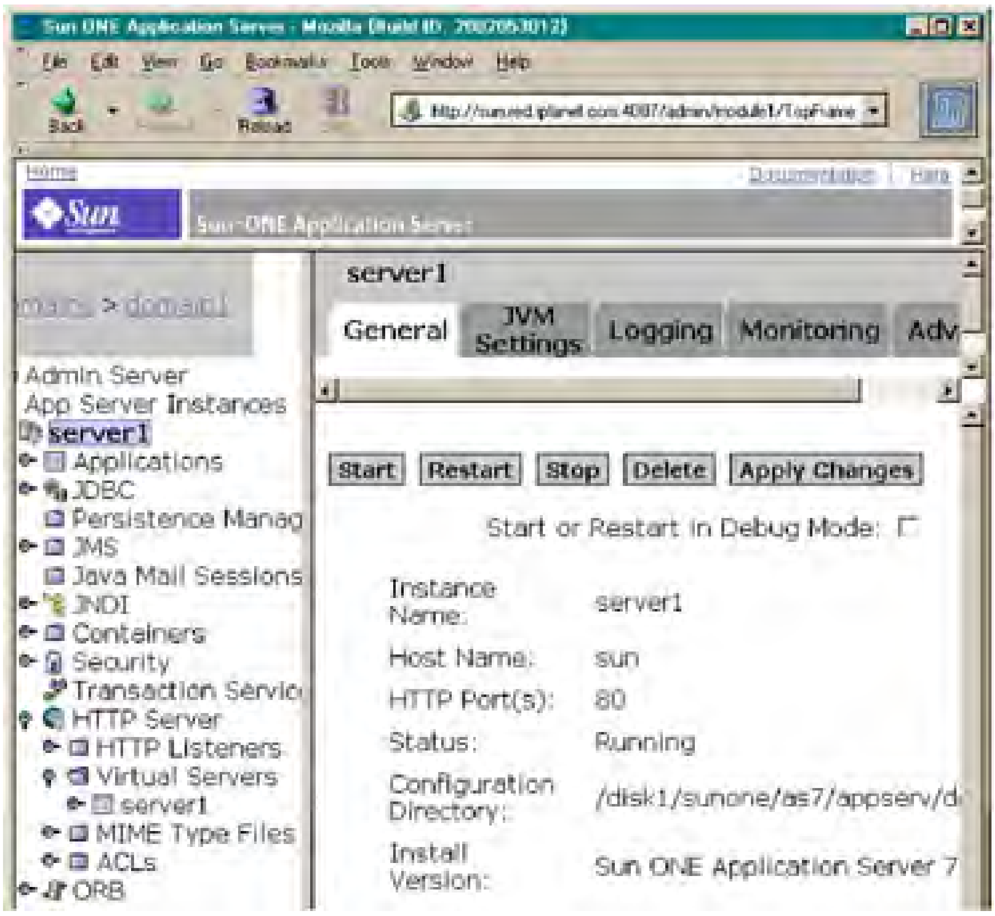

Web Site Computer
Web Browser
Sun ONE Web Server on the computer water.red.iplanet.com
HTTP Listener on Port 80
Connection Management by a virtual server
Static Content SAF
CGI SAF
JSP SAF
Java Development Kit (JDK)
Supporting Programs
Static files in the document directory
CGI program files in the cgi-bin directory
JSP files under the document directory
Configuration files, example: mime.types

The first step is to configure the Sun ONE Application Server. Configuration settings are managed through the Sun ONE Application Administration Server.

HTTP Server Configurations

If your administration server is not running, start it and log in to the server. Once logged in, click on the menu item *server1* on the left side of the screen (Figure 16.2). From this web page, the server instance is started and stopped; also, when changes are made to the configuration files, they are applied here. There will be times throughout this chapter that you will return to this page to apply configuration changes and restart the server. All the configuration settings in this chapter are under *HTTP Server* menu option. Under this option, click on *server1* as in Figure 16.3. The static content directories and CGI setting are configured from this web page.

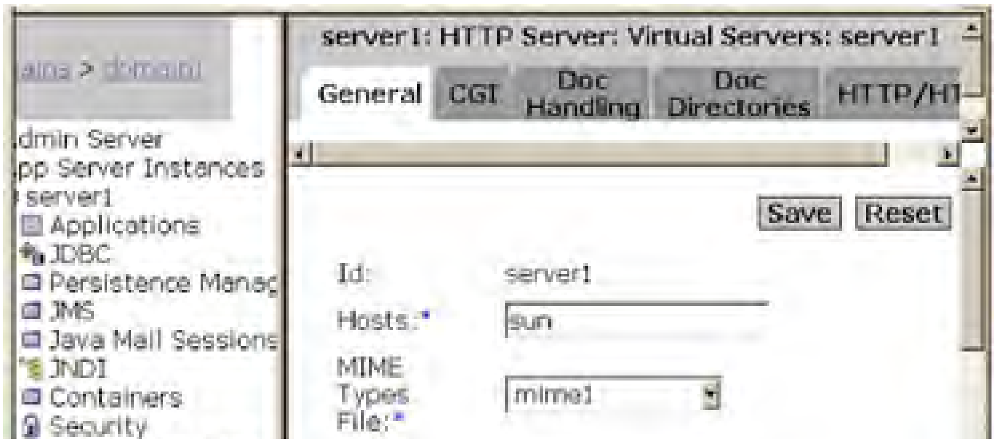
Figure 16.2. Application server instance configurations.



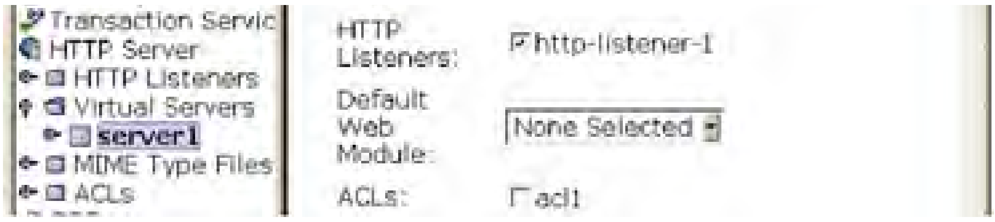
Start or Restart in Debug Mode: ☐

Instance Name:	server1
Host Name:	sun
HTTP Port(s):	80
Status:	Running
Configuration Directory:	/disk1/sunone/as7/appserv/d
Install Version:	Sun ONE Application Server 7

Figure 16.3. HTTP server configuration settings.



Id:	server1
Hosts:*	sun
MIME Types File:*	mime1



HTTP Listeners: ☑ http-listener-1

Default Web Module: None Selected

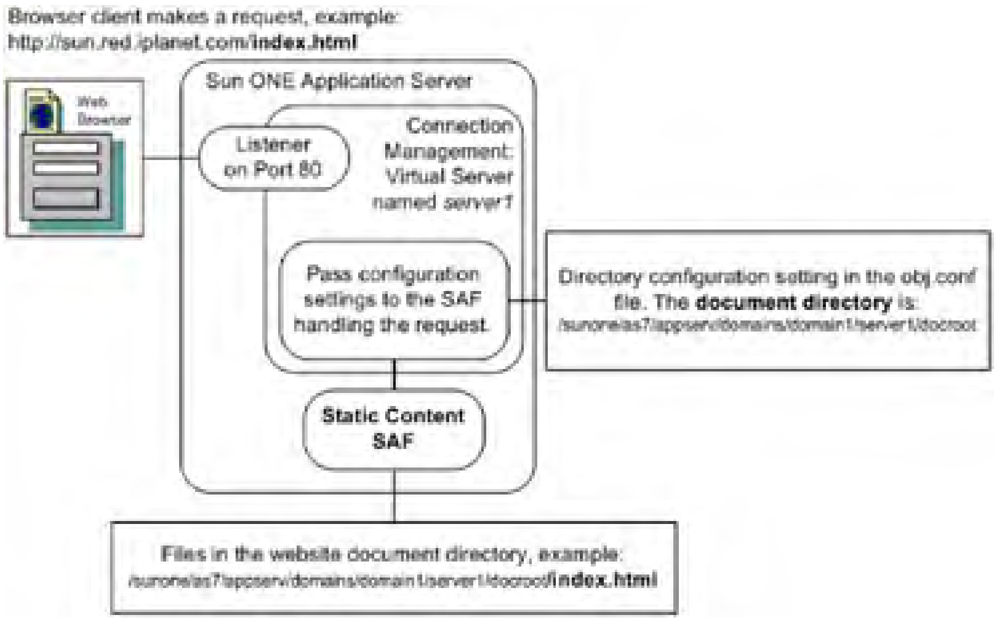
ACLs: ☐ acl1

The Static Content SAF

The static content SAF is the basic web site tool used to serve files to web browsers. The setup is simple: A directory is specified from the web page in Figure 16.3 (scroll down the page to see the value), and then files are put into the directory. That's it—files are ready to be served to web browsers!

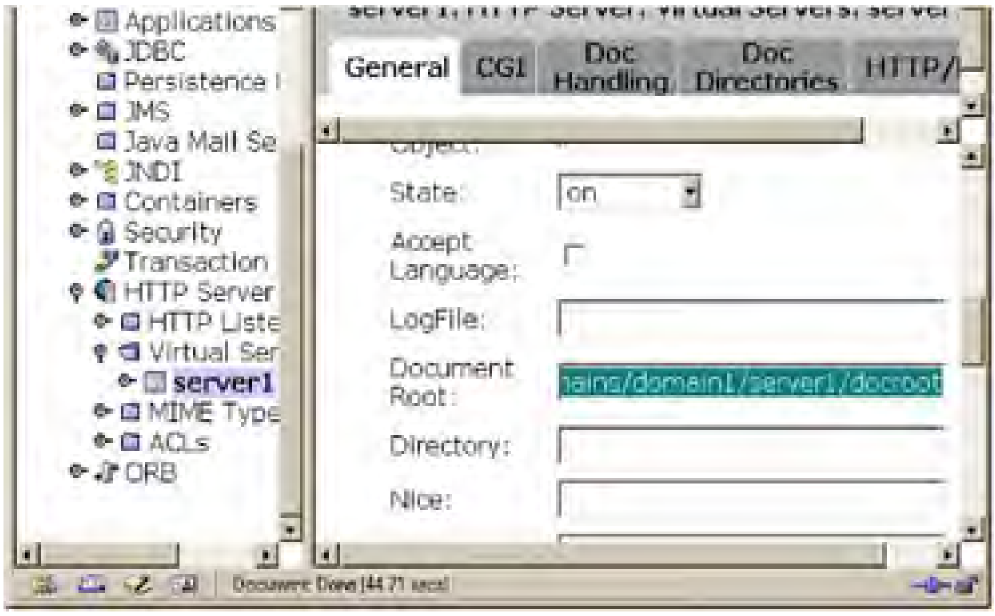
In Figure 16.4, the browser is requesting the file index.html. When the HTTP server receives this request, its connection management part gets static directory information from the configuration settings in the obj.conf file. Using these settings, the static content SAF finds and reads the index.html file and returns it to the browser.

Figure 16.4. Static content SAF document directory configuration setting.


Browser client makes a request, example:
http://sun.red.iplanet.com/**index.html**
Web Browser
Sun ONE Application Server
Listener on Port 80
Connection Management: Virtual Server named *server1*
Pass configuration settings to the SAF handling the request.
Directory configuration setting in the obj.conf file. The **document directory** is: /sunone/as7/appserv/domains/domain1/server1/docroot
Static Content SAF
Files in the website document directory, example: /sunone/as7/appserv/domains/domain1/server1/docroot/**index.html**

To view the document directory configuration, scroll down to *Document Root* (Figure 16.5). This shows that the document directory is file directory /sunone/as7/appserv/domains/domain1/server1/docroot.

Figure 16.5. General web site settings.



State: on
Accept Language:
LogFile:
Document Root: ains/domain1/server1/docroot
Directory:
Nice:

This setting maps the root document URL

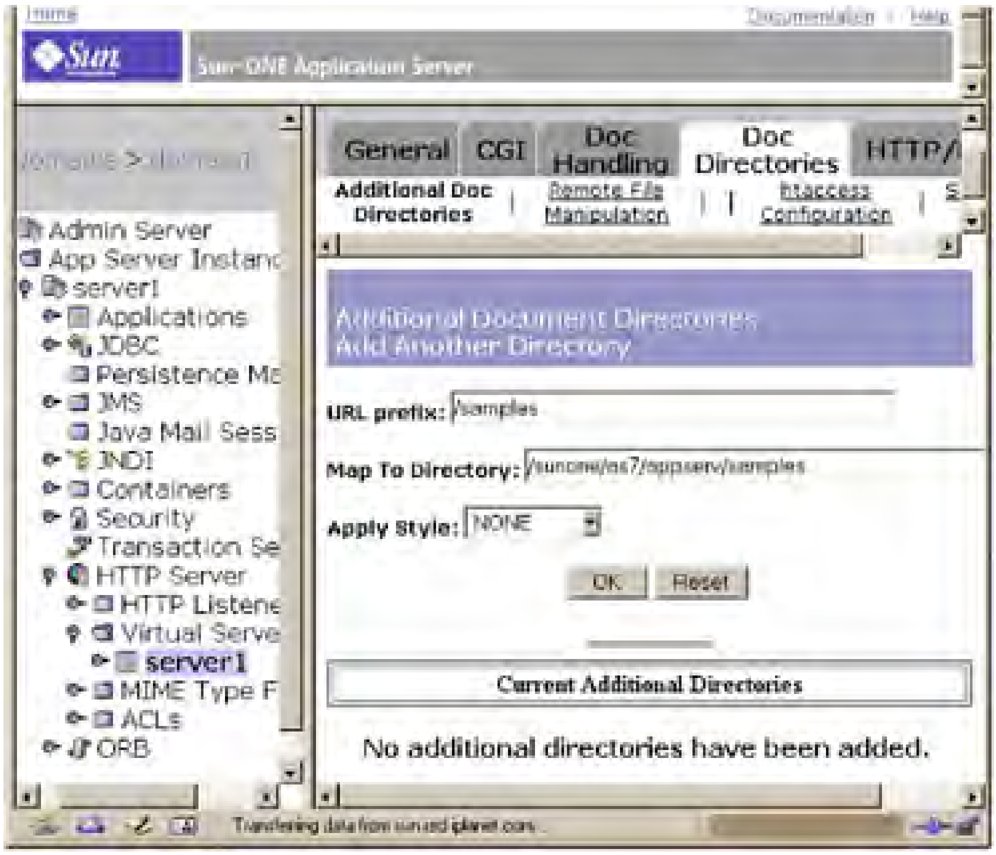
http://sun.red.iplanet.com/

to the root document file directory

/sunone/as7/apps-
  erv/domains/domain1/server1/docroot

It is possible to map other directories to other URLs. If you click the *Doc Directories* tab as shown in Figure 16.6, the web page appears that is used to manage other static document directories. Enter */samples* in the box after *URL prefix,* and enter */sunone/as7/appserv/samples* into the *Map To Directory* field; click *OK*. This maps the URL

Figure 16.6. Setting up another static document directory.



Additional Document Directories
Add Another Directory

URL prefix: /samples

Map To Directory: /sunone/as7/appserv/samples

Apply Style: NONE

Current Additional Directories

No additional directories have been added.

http://sun.red.iplanet.com/samples

to the file directory

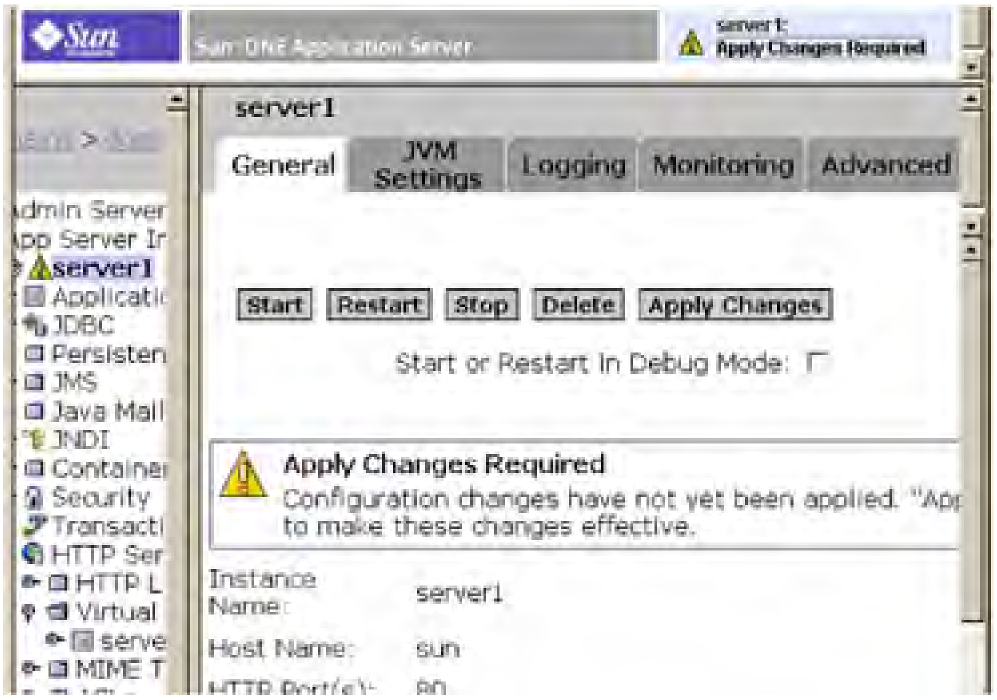
/sunone/as7/appserv/samples

After clicking *OK*, a prompt—*A directory mapping has been added*—is displayed. Click *OK*. This new mapping will appear under *Current Additional Directories*.

Applying Changes

Next we apply the configuration change. Under the menu item *App Server Instance*, click *server1*. The web page to apply changes is displayed as in Figure 16.7. Click *Apply Changes*. Now there are two messages displayed: *Changes applied to instance* and *Restart required*. Click the *Restart* button. The server instance is restarted and is now ready to serve files from the new document directory.

Figure 16.7. Apply configuration changes.

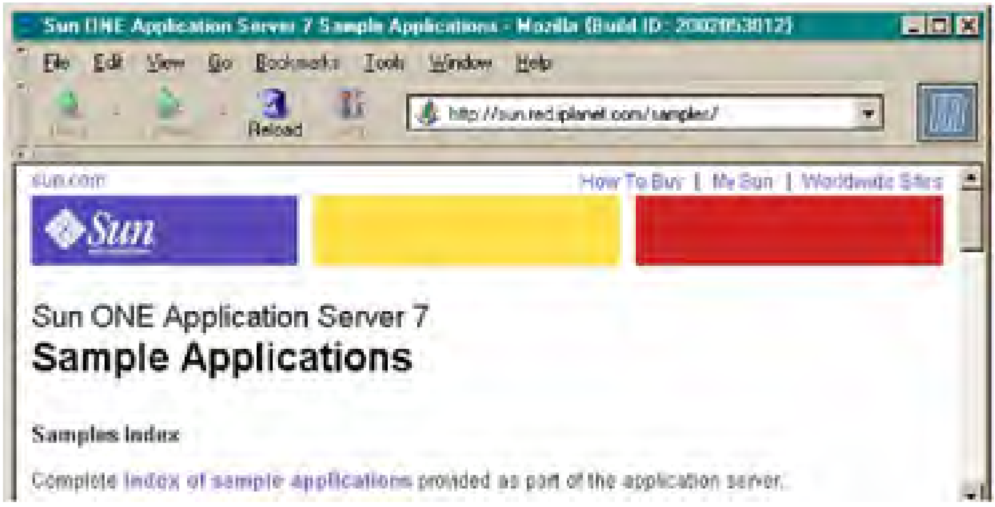


Apply Changes Required
Configuration changes have not yet been applied. "App
to make these changes effective.

Instance Name: server1
Host Name: sun
HTTP Port(s): 80

To test that the new document directory mapping is working, from a browser, go to the URL *http://sun.red.iplanet.com/samples* (Figure 16.8).

Figure 16.8. Sample applications home page.


Sun ONE Application Server 7 Sample Applications - Mozilla {Build ID: 2002053012}
File
Edit
View
Go
Bookmarks
Tools
Window
Help
Reload
http://sun.red.iplanet.com/samples/
sun.com
How To Buy
My Sun
Worldwide Sites
Sun

Sun ONE Application Server 7
Sample Applications

Samples Index

Complete index of sample applications provided as part of the application server.

Exercise to Modify the Default Home Page

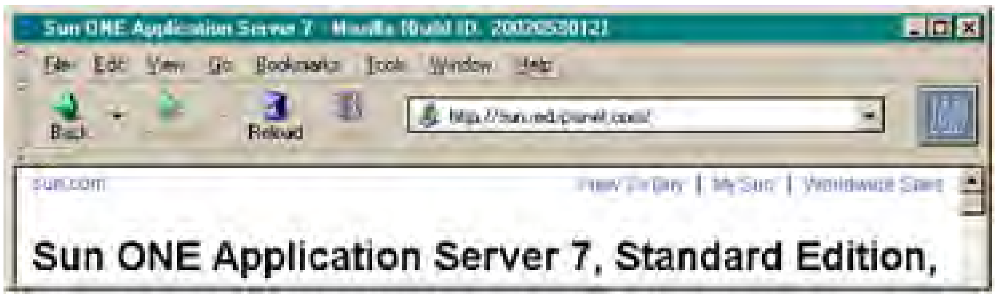
In Figure 16.5, we viewed the location of the root document directory which is:

```
/sunone/as7/appserv/domains/domain1/server1
  /docroot
```

In this exercise, the default home web page in the document directory is modified. Before starting, ensure that the Sun ONE Application Server instance is running. From a web browser bring up the home page *http://sun.red.iplanet.com*.

When the Sun ONE Application Server is installed, a default home web page is set up. Figure 16.9 is a screen print of the web page.

Figure 16.9. Default Sun ONE Application Server home web page.


Sun ONE Application Server 7 - Mozilla {Build ID: 2002053012}
File Edit View Go Bookmarks Tools Window Help
Back
Reload
http://sun.red.iplanet.com/
sun.com
How To Buy | My Sun | Worldwide Sites
Sun ONE Application Server 7, Standard Edition,

1. From a command-line window, go to the document root directory

   # cd /sunone/as7/apps-
   erv/domains/domain1/server1/docroot

2. Rename the current home page file; for example, # rename index.html index.txt.

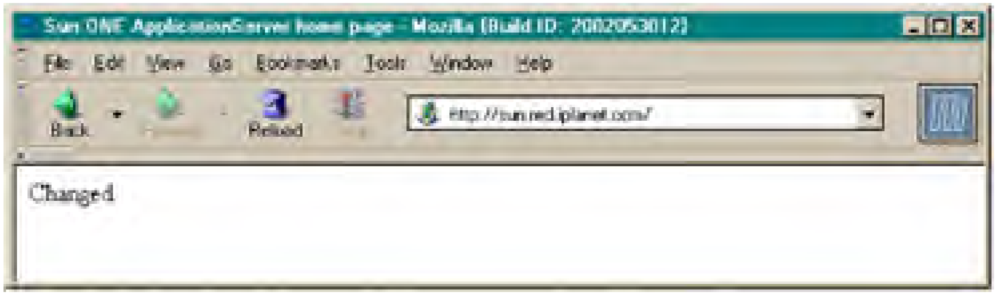
3. Use your favorite text file editor (such as vi or notepad) and create a new home page file: # vi index.html.

Enter the following text into your index.html file:

```
<html><head>
<TITLE>Sun ONE ApplicationServer home page</TITLE>
</head>
<body>
Changed.
</body>
</html>
```

Save the file and reload the browser. The web page change shows up as pictured in Figure 16.10. Note: When using a Netscape (or similar) browser, to ensure the page is reloaded, hold the Shift key down and hit the Reload button. This forces the web page to be reloaded. The page is reloaded from the server, not from the memory cache.

Figure 16.10. Changed home web page.


Sun ONE ApplicationServer home page - Mozilla {Build ID: 2002053012}
File Edit View Go Bookmarks Tools Window Help
Back
Reload
http://sun.red.iplanet.com/
Changed

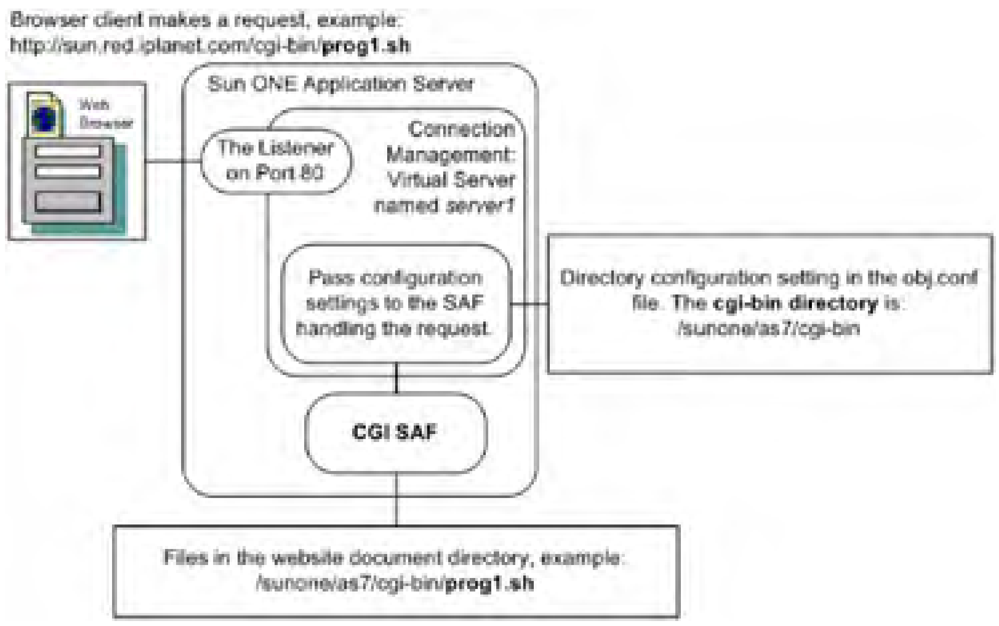
Now that you know where the static web pages are located, you can create your own web site. Static content setup is easy, isn't it?

Setting Up a CGI-BIN Directory

In this section, we will configure the application server to create dynamic content using programs that are run by the Common Gateway Interface SAF, or CGI SAF. A cgi-bin directory is a file directory from which programs are launched by the application server instance. The CGI SAF is the oldest and most basic web site tool for running programs. The setup is simple—a directory is specified; then program files are put into the directory. Now the application server is ready to run programs to create dynamic web site content.

As illustrated in Figure 16.11, the browser sends a request to the application server to run the program file prog1.sh. The listener receives the request. The connection management part of the HTTP server (the virtual server) gets the CGI directory information from the configuration settings in the obj.conf file. Using these settings, the CGI SAF finds the file prog1.sh. A process is spawned, the program is run, and everything that the program prints to screen (standard output) is returned to the browser.

Figure 16.11. CGI dynamic content SAF cgi-bin directory configuration setting.


Browser client makes a request, example:
http://sun.red.iplanet.com/cgi-bin/**prog1.sh**
Web Browser
Sun ONE Application Server
The Listener on Port 80
Connection Management: Virtual Server named server1
Pass configuration settings to the SAF handling the request.
Directory configuration setting in the obj.conf file. The **cgi-bin directory** is: /sunone/as7/cgi-bin
CGI SAF
Files in the website document directory, example: /sunone/as7/cgi-bin/**prog1.sh**

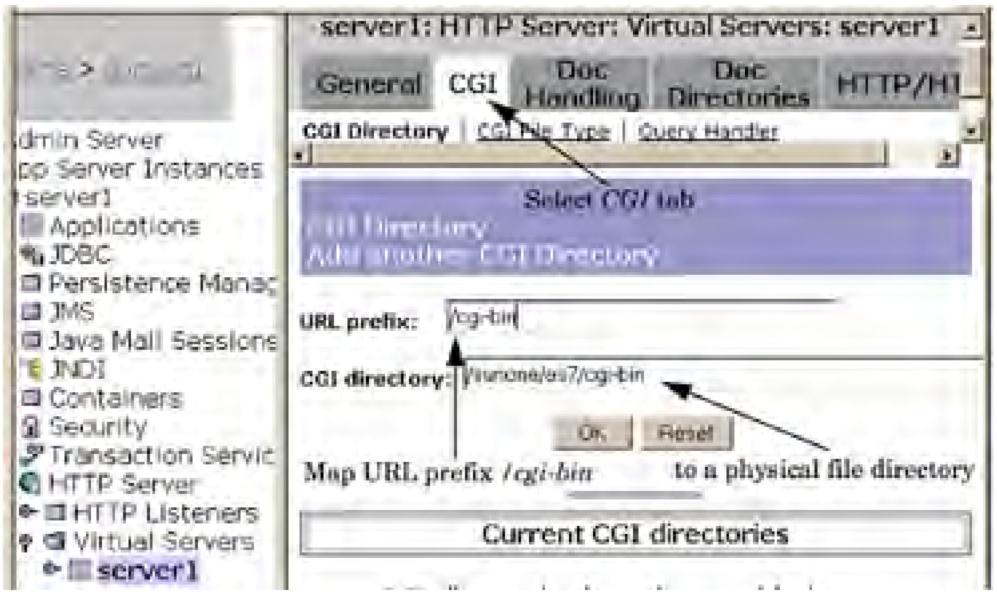
Configuring the Sun ONE Application Server to run CGI programs is very straightforward:

1. Configure the /cgi-bin/ directory setting using the application server administration server.
2. Make the /cgi-bin/ directory on the application server computer.
3. Write a program in the /cgi-bin/ directory, or write the program somewhere else and load or copy the program into the /cgi-bin/ directory.
4. On UNIX, make the program file executable. As a quick test, run the program in a command-line window (terminal window).
5. Test the program by using a web browser.

CGI Configuration

From a browser, log in to the Sun ONE Application Server Administration Server. Once logged in, under menu item *Virtual Servers* on the left, click on *server1*. Then click the CGI tab. Now enter values similar to what is in Figure 16.12. When you click OK, the application server is configured to run CGI programs.

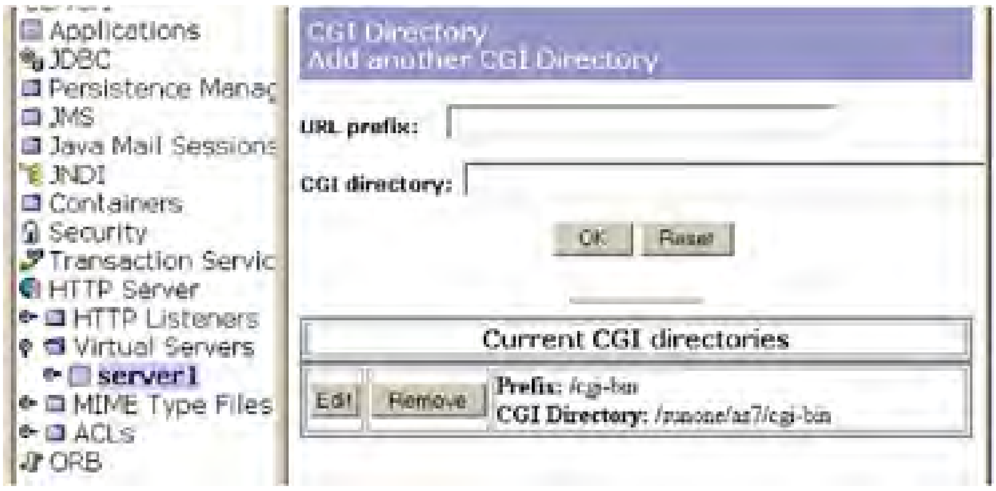
Figure 16.12. Configure the cgi-bin directory.



This setting maps the cgi-bin directory URL *http://sun.red.iplanet.com/cgi-bin/* to the cgi-bin file directory /sunone/as7/cgi-bin.

After clicking *OK*, the message *A CGI directory mapping has been added* is displayed. Click *OK* and the window refreshes showing the new mapping (Figure 16.13).

Figure 16.13. Refreshed window with new mapping.



Next, apply the CGI configuration changes. Under the menu item *App Server Instance*, click *server1*; then click *Apply Changes*. There are now two messages displayed: *Changes applied to instance* and *Restart required*. Click the *Restart* button, and the server instance is restarted and ready to serve CGI programs.

On the application server computer, make a directory that matches the *CGI Directory*; our example directory is /sunone/as7/cgi-bin. On UNIX, the command is

```
# mkdir /sunone/as7/cgi-bin
```

UNIX CGI Program Test

To run a test, create a shell program named prog1.sh in the cgi-bin directory. Below is an example shell program. Note: In the first line, /usr/bin/csh needs to match the location of csh on your computer.

```
#! /usr/bin/csh
echo "Content-type: text/html"
echo ""
echo "<html>"
echo "<HEAD>"
echo "<TITLE>Super Simple</TITLE>"
echo "</head>"
echo "<body>"
echo "<H1>Super Simple Test C-Shell Program</H1>"
echo "</body>"
echo "</html>"
exit
# eof
```

Make the program file executable:

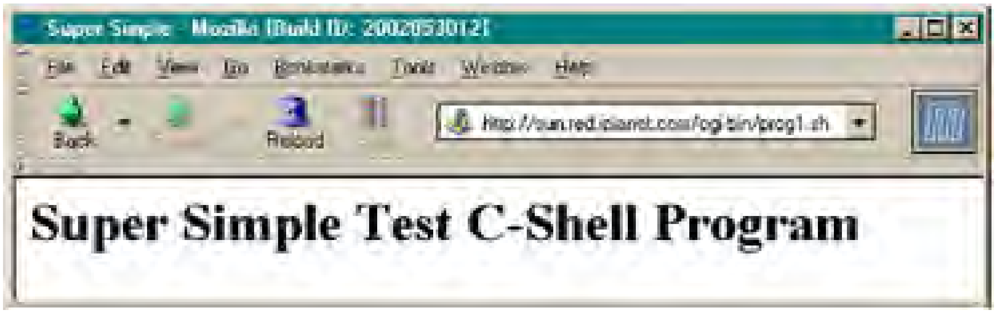
```
# cd /sunone/as7/cgi-bin
# chmod o+x prog1.sh
```

As an optional test, run the program `# ./prog1.sh`. The program echoes an HTML web page.

Now use your browser to test the program. The syntax of the URL is *http://sun.red.iplanet.com:80/cgi-bin/prog1.sh*

Figure 16.14 shows that the test is successful.

Figure 16.14. Dynamic web page from the CGI program prog1.sh (UNIX).


Super Simple - Mozilla {Build ID: 2002053012}
File
Edit
View
Go
Bookmarks
Tools
Window
Help
Back
Reload
http://sun.red.iplanet.com/cgi-bin/prog1.sh

Super Simple Test C-Shell Program

Windows CGI Program Test

To run a test on Windows, create a batch program named prog1.sh in the cgi-bin directory. Here is an example Windows batch program:

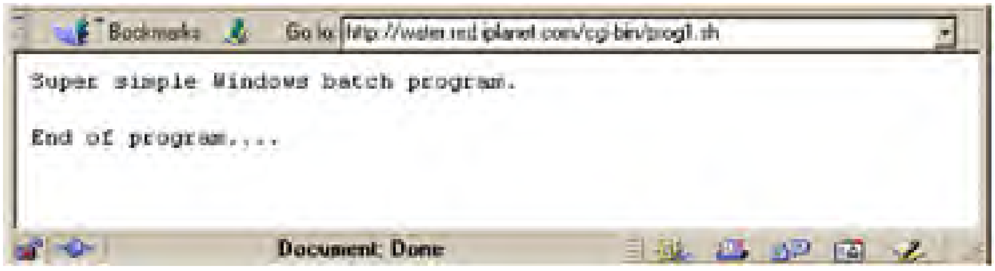
```
@echo off
echo Content-type: text/plain
echo.
echo Super simple Windows batch program.
echo.
echo End of program....
```

As an optional test, run the program prog1.sh. The program echoes an HTML web page.

Now use your browser to test the program. The syntax of the URL is *http://water.red.iplanet.com:80/cgi-bin/prog1.sh*

Figure 16.15 shows that the test is successful.

Figure 16.15. Dynamic web page from the CGI program prog1.sh (Windows).



```
Super simple Windows batch program.

End of program....
```

CGI is very popular on the Internet. This is because it is well known and so easy to set up.

Running JSPs

JSPs allow us to create powerful, dynamic web pages and are similar in concept to embedding SQL database calls into C programs. What JSPs actually do is embed Java program calls into an HTML file. In fact, a JSP is an HTML file with embedded Java code and specialized tags.

Running servlets on a web server requires only a JRE. However, to run JSPs, you need the JDK. Here are the steps the application server goes through to run a JSP for the first time.

1. The JSP is transformed into a Java servlet program.
2. The Java servlet is compiled by the compiler javac.exe into a class file. The javac.exe compiler comes with the JDK. The Sun ONE Application Server comes with the JDK version 1.4.
3. Once compiled, the class file is run by the application server.

The following is an example of a JSP.

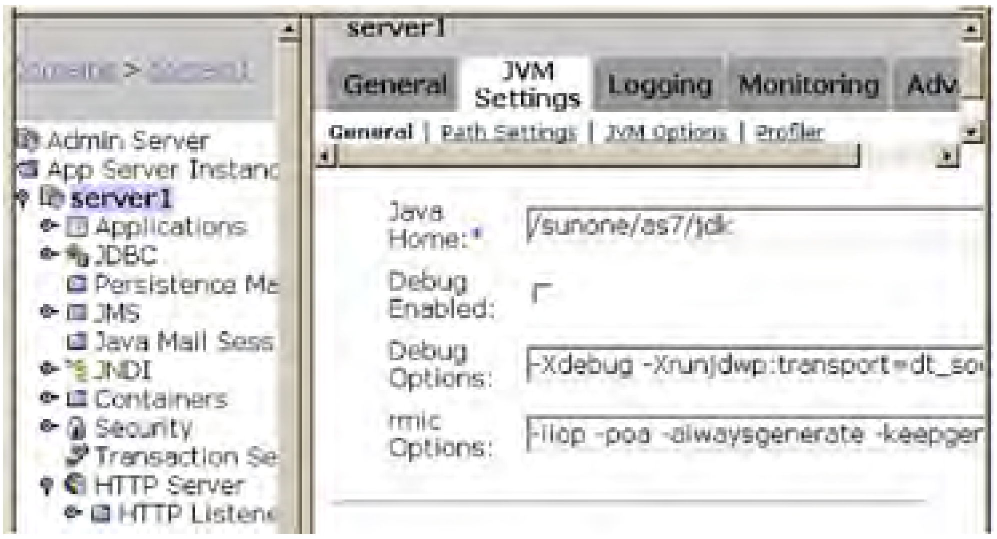
```
<head>
<title>JSP Super Simple Sample</title>
</head>
<BODY  bgcolor="#FFFFFF" text="#000000">

<h2>JSP Super Simple Sample</h2>

<% String s1 = "cool"; %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

String s1 = "cool"; is the Java command to declare a variable of type *String* and to set the variable to the value *cool*. <% and %> are the tokens used to embed Java code. <%=s1%> is a specialized JSP tag to display/print a Java program variable. The rest of the code is HTML.

Viewing the JDK Configurations

The JDK is installed during the installation of your Sun ONE Application Server. To view the settings, from a browser, log in to the Sun ONE Application Server Administration Server. Once logged in, under *App Server Instance* (on the left in Figure 16.16) click on *server1*, and then click the *JVM Settings* tab.

Figure 16.16. View configuration setting to run JSPs.



Java Home:* /sunone/as7/jdk
Debug Enabled:
Debug Options: -Xdebug -Xrunjdwp:transport=dt_so
rmic Options: -iiop -poa -alwaysgenerate -keepger

Creating and Testing a JSP

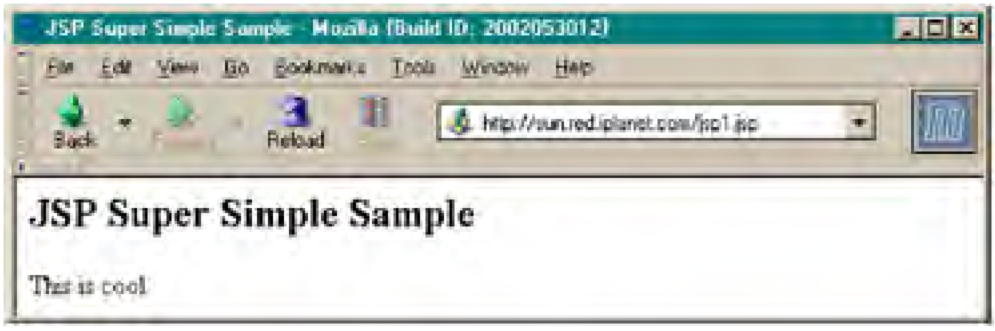
By default, the application server is configured to treat any file with the filename extension of jsp as a JSP program. Therefore, any file under the document directory with filename extension jsp is run as a JSP.

Create a JSP file in the document directory /sunone/as7/apps-erv/domains/domain1/server1/docroot. Use the filename jsp1.jsp.

```
<head>
<title>JSP Super Simple Sample</title>
</head>
<BODY  bgcolor="#FFFFFF" text="#000000">

<h2>JSP Super Simple Sample</h2>

<% String s1 = "cool"; %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

To test, from a web browser, go to URL *http://sun.red.iplanet.com/jsp1.jsp* (as shown in Figure 16.17). Figure 16.16 shows that the test is successful.

Figure 16.17. Output from the JSP jsp1.jsp.


JSP Super Simple Sample - Mozilla (Build ID: 2002053012)
File Edit View Go Bookmarks Tools Window Help
Back
Forward
Reload
http://sun.red.iplanet.com/jsp1.jsp

JSP Super Simple Sample

This is cool

Once again, setting up to run dynamic content from the application server is a straightforward process. However, when programming, something is bound to go wrong and, when it does, we will want to use the error log to help debug our program.

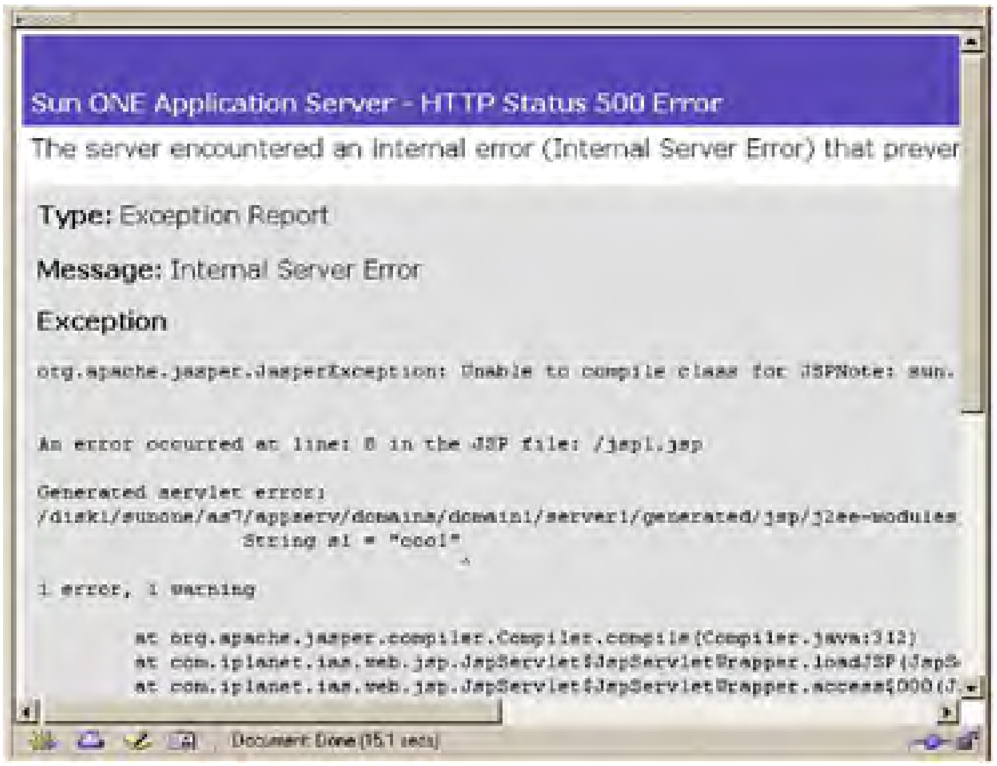
Error Log Checking

If there is a bug (or error) in the JSP program code—say, for example, the omission of the semicolon in the first Java command (<% String s1 = "cool" %>)—an error will happen. Modify jsp1.jsp (to omit the semicolon) so you can see what happens.

```
<head>
<title>JSP Super Simple Sample</title>
</head>
<BODY  bgcolor="#FFFFFF" text="#000000">

<h2>JSP Super Simple Sample</h2>

<% String s1 = "cool" %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

To test, from a web browser, go to URL >*http://<hostname>/jsp1.jsp*. For example, *http://sun.red.iplanet.com/jsp1.jsp*. The result will be the error output message in Figure 16.18.

Figure 16.18. Error output message from jsp1.jsp.



Sun ONE Application Server - HTTP Status 500 Error

The server encountered an internal error (Internal Server Error) that prever

Type: Exception Report

Message: Internal Server Error

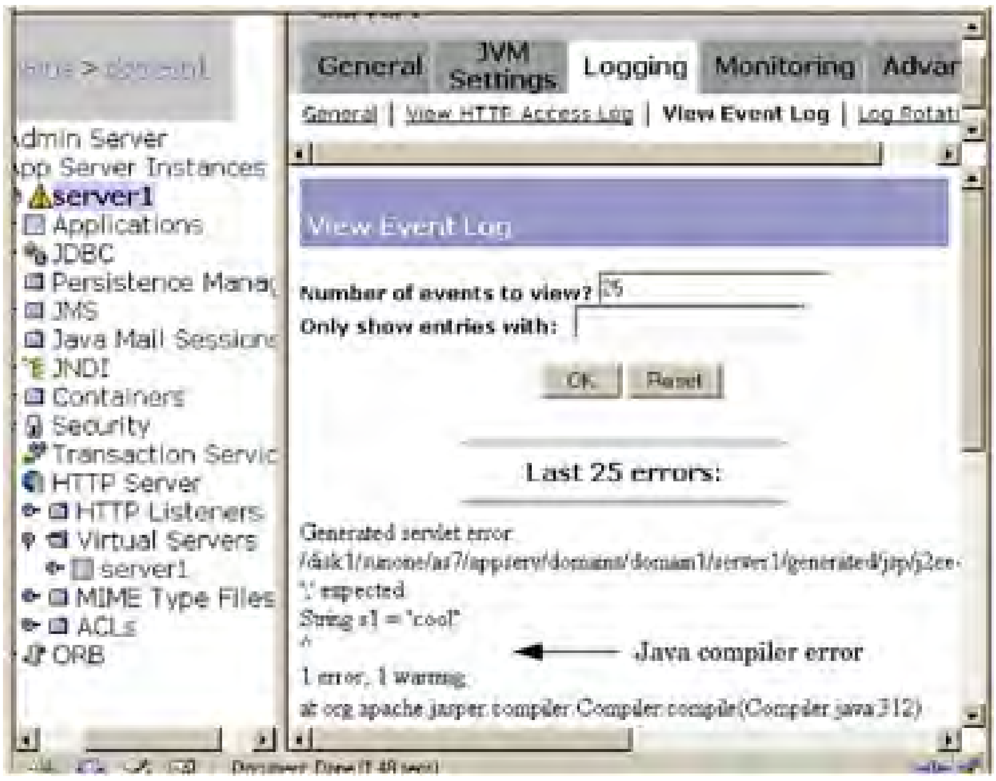
Exception

```
org.apache.jasper.JasperException: Unable to compile class for JSPNote: sun.


An error occurred at line: 8 in the JSP file: /jsp1.jsp

Generated servlet error:
/disk1/sunone/as7/appserv/domains/domain1/server1/generated/jsp/j2ee-modules
                String s1 = "cool"
                                  ^

1 error, 1 warning

        at org.apache.jasper.compiler.Compiler.compile(Compiler.java:312)
        at com.iplanet.ias.web.jsp.JspServlet$JspServletWrapper.loadJSP(JspS
        at com.iplanet.ias.web.jsp.JspServlet$JspServletWrapper.access$000(J.
```

Document: Done (15.1 secs)

Another way to view the error message is to log in to the Sun ONE Application Server Administration Server. Under *App Server Instance* (see Figure 16.19), click *server1*, click the *Logging* tab; then click *View Event Log*. Inside the log file is the same error message as the one in Figure 16.18.

Figure 16.19. Viewing error messages.



View Event Log

Number of events to view? 25

Only show entries with:

Last 25 errors:

```
Generated servlet error
/disk1/sunone/as7/appserv/domains/domain1/server1/generated/jsp/j2ee-
';' expected
String s1 = "cool"
^
1 error, 1 warning
at org.apache.jasper.compiler.Compiler.compile(Compiler.java:312)
```

Document: Done (1.48 secs)

Fix the error in jsp1.jsp and the JSP will work again.

[Team LiB]

Standalone Servlets

In the Sun ONE Web Server 6.0, standalone servlets were called *legacy servlets*. In the Sun ONE Application Server 7, legacy servlet configuration has been dropped. The only way to have servlet class files run is to put the servlet into a web application archive (war) file and deploy the war file into the application server. There will be more on this topic in a later chapter.

If you still want to run servlets without war files, you can convert your servlet into a JSP. Then let the application server compile the JSP into a servlet class. For complicated servlets, this may be difficult at best, but for simple servlets it is not that difficult. Note that this is not something I recommend; however, it does work. It is best to learn to use JSPs effectively and to create war files using the Sun ONE Studio 4 developer package. There is a free version of Studio 4, so consider investing the time to learn to use it. Again, there will be more discussion of this topic in a later chapter.

Here is a sample JSP with no HTML code; this makes it very similar to a simple servlet program.

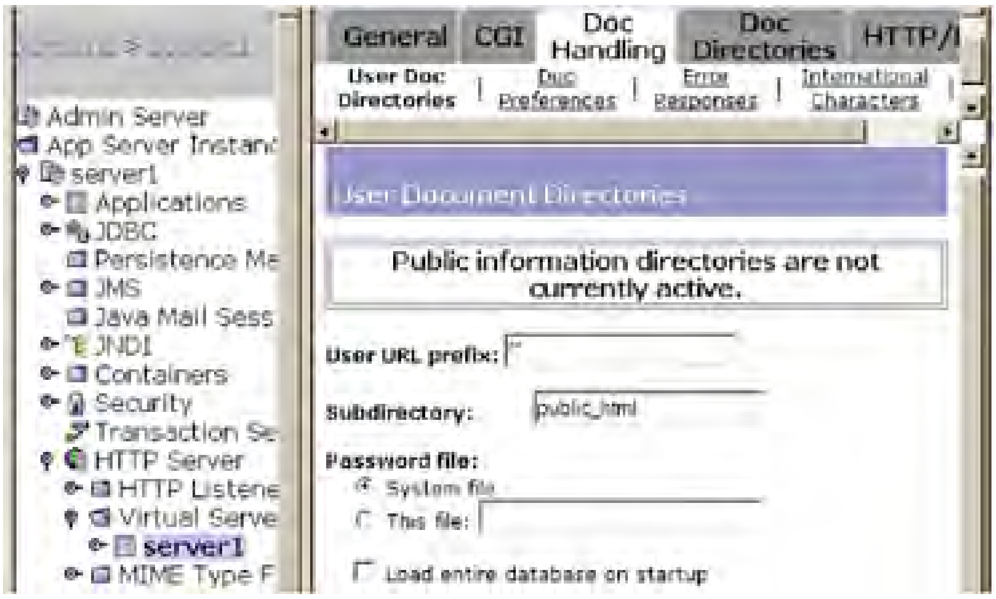
```
<% // import statement: %>
<%@page import="java.io.*" %>

<% // Start of servlet code.
response.setContentType( "text/plain" );
out.println( "* Start, servletecho, v1.01" );

out.println( "Method:    " + request.getMethod() );
out.println( "URI:       " + request.getRequestURI() );
out.println( "Protocol:  " + request.getProtocol() );
out.println( "Path Info: " + request.getPathInfo() );

String thePars;
if (request.getMethod().equals("GET"))
{
thePars = request.getQueryString();
out.println( "+ * echo servletecho, 'GET' Parameters '"
  + thePars + "'" );
}
else if (request.getMethod().equals("POST"))
{
out.println( "+ * echo servletecho, 'POST' Parameters:"
  );
DataInputStream in = new DataInputStream( request.get-
  InputStream() );
out.println( in.readLine() );
while ( in.available() > 0 ) {
out.println( in.readLine() );
}
out.println( "+ * echo servletecho, 'POST' end of
  list..." );
}
else
{
out.println( "+ * echo servletecho, unknown method '" +
  request.getMethod() + "'" );
}
out.println( "* Success, servletecho..." );
%>
```

User Document Root Directories

While surfing the Internet, I see that user document directories are still popular, so I have included the instructions to set up this feature on the Sun ONE Application Server. An example user document directory URL is *http://sun.red.iplanet.com/~stacy*, whereas *stacy* is a UNIX user on the server computer.

From a browser, log in to the Sun ONE Application Server Administration Server. Under menu item *Virtual Server* (Figure 16.20), click on *server1* and then click the *Doc Handling* tab. To turn user document directories *on*, click *OK* and the application server is configured. Next apply the changes and restart the server.

Figure 16.20. Setting up user document directories.



Public information directories are not currently active.

User URL prefix: ~

Subdirectory: public_html

Password file:

(●) System file

( ) This file:

[ ] Load entire database on startup

As an exercise to solve a problem using the error logs, follow these steps to finish setting up the user document directory for the UNIX user *stacy*:

1. After the changes have been applied and the server restarted, go to the URL *http://sun.red.iplanet.com/~stacy*.
2. Check the log file to see the error:

   ```
   [09/Aug/2002:09:14:53] WARNING ( 580): for host
   1.1.1.101 trying to GET /~stacy/, unix-home reports:
   HTTP4110: could not find home directory for user
   stacy
   ```

3. Create UNIX user *stacy* using sample command

   ```
   # useradd stacy
   ```

4. Go to *http://sun.red.iplanet.com/~stacy*
5. The log file shows error:

   ```
   [09/Aug/2002:09:16:23] WARNING ( 580): for host
   1.1.1.101 trying to GET /~stacy/, send-file reports:
   HTTP4142: can't find /home/stacy/public_html/ (File
   not found)
   ```

6. Create directory /home/stacy/public_html/ and put an index.html file into the directory.
7. Again, go to the URL *http://sun.red.iplanet.com/~stacy*. Now the index.html appears in the browser. Success!

There are many more web server features built into the Sun ONE Application Server—don't be afraid to try things out.

Summary

This chapter covered the fundamentals of configuring the Sun ONE Application Server to serve web site content. Configuring the application server is a simple matter of configuring directories, putting files in place, and testing.

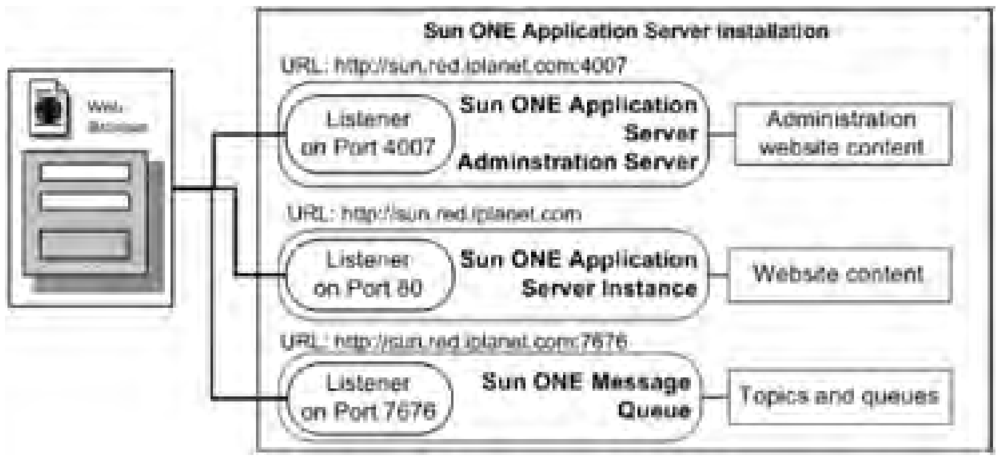
Although CGI programming is offered by most ISPs, I wish more of them would also offer a Java option. The Java program environment is so easy for the ISP to set up, and the Java program language is such an excellent language for programming Internet programs.

Chapter 17. Sun ONE Application Server and Administration Hierarchy

Up to this point, the Sun ONE Application Server chapters have been about using a single HTTP server on a single application server instance. This chapter looks at multiple configurations of instances, HTTP (web) server listeners, and virtual servers.

Figure 17.1 is a diagram of the default servers that are currently installed and running with the Sun ONE Application Server—an administration server, an application server instance, and a message queue.

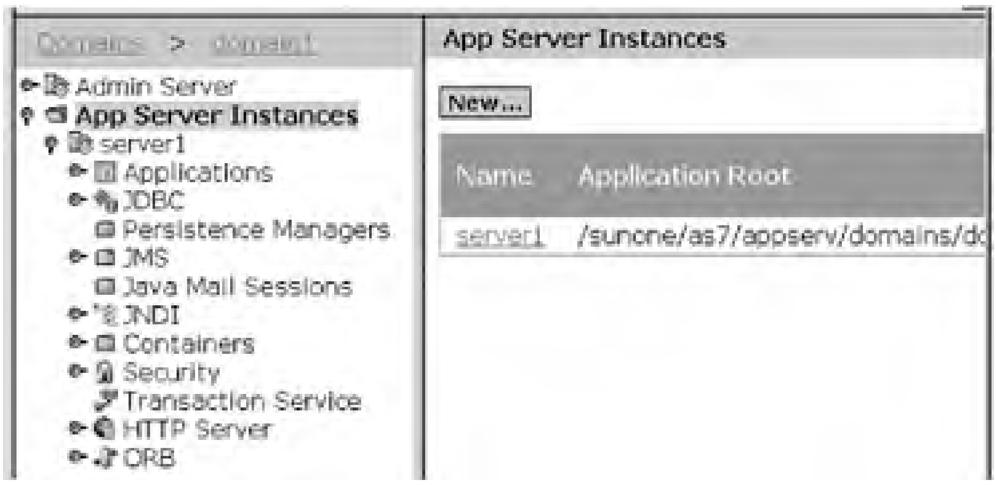
Figure 17.1. Sun ONE Application Server 7 default servers.


Web Browser
Sun ONE Application Server installation
URL: http://sun.red.iplanet.com:4007
Listener on Port 4007
Sun ONE Application Server Adminstration Server
Administration website content
URL: http://sun.red.iplanet.com
Listener on Port 80
Sun ONE Application Server Instance
Website content
URL: http://sun.red.iplanet.com:7676
Listener on Port 7676
Sun ONE Message Queue
Topics and queues

Application Server Instance

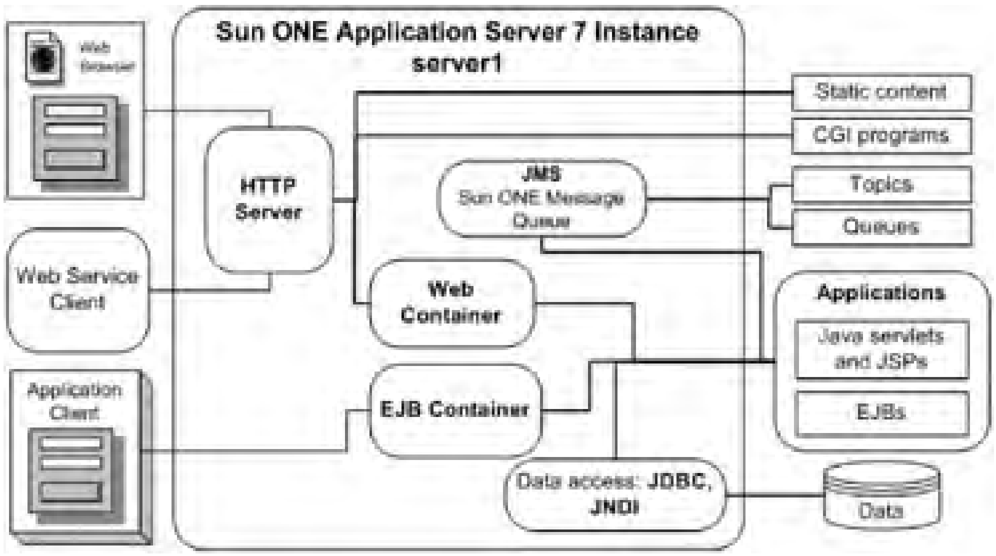
Log in to the Sun ONE Application Server Administration Server and you will see a list of the components that make up an application server instance as in Figure 17.2. Figure 17.3 shows these application server components in diagram form.

Figure 17.2. Configuration menu with the application server instance components.



Name	Application Root
server1	/sunone/as7/appserv/domains/dc

Figure 17.3. Application server instance components.



An application instance is made up of a number of components. Some of the components are individual processes, such as the HTTP server and the JMS message queue, while others are connector tools, such as Java DataBase Connectivity (JDBC) and Java Naming and Directory Interface (JNDI).

During an installation, an application server instance has components that listen to a number of ports. For example, the currently installed HTTP server listens on port 80, and the message queue listens on port 7676.

Next we add another application server instance that has listeners on ports that are different from those the current listeners are on.

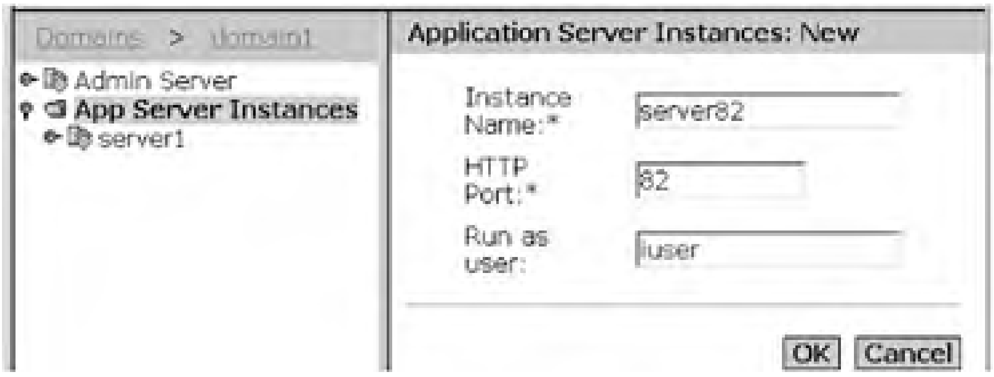
Adding an Application Server Instance

An instance runs a number of processes. The processes are run as a specific user in a specific group. When testing, I create the user *iuser* (Internet user) and *igroup*. Since a user is added to a group, create the group first. The UNIX commands to create a test user and group are

```
# groupadd igroup
# useradd -g igroup iuser
```

From the main administration web page, click the menu item *App Server Instance* and then click *New*. Enter data into the web page as shown in Figure 17.4. Then click *OK*—you have created a new application server instance.

Figure 17.4. Adding a new application server instance.



Instance Name:* server82
HTTP Port:* 82
Run as user: iuser

Now there are two instances. Figure 17.5 is a diagram of the setup, and Figure 17.6 is the administration menu tree showing the new instance and its components.

Figure 17.5. Multiple application server instances.

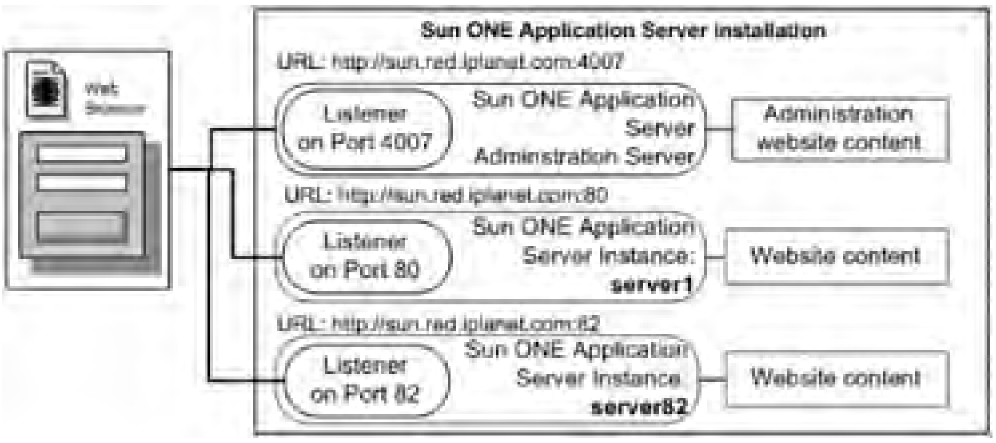
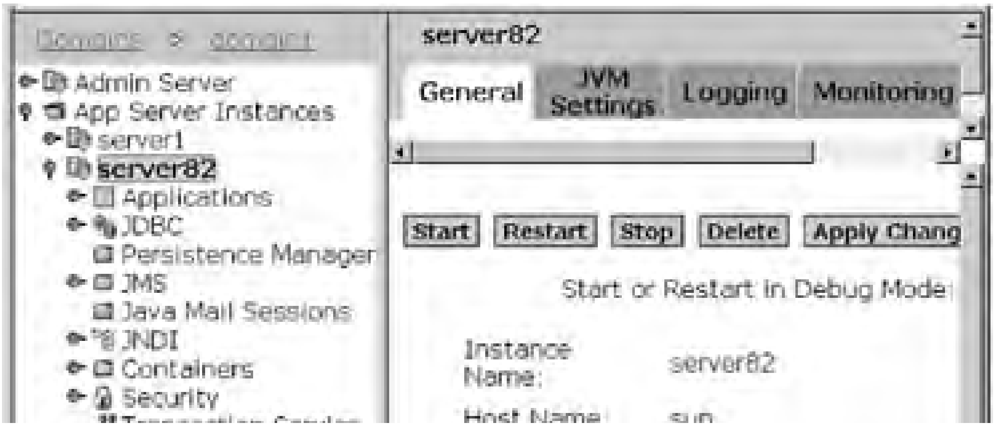
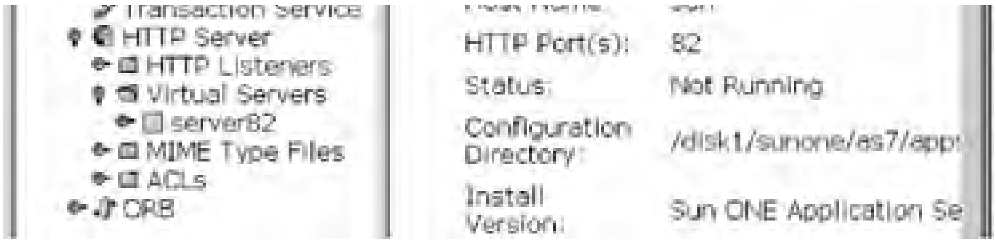

Web Browser
Sun ONE Application Server installation
URL: http://sun.red.iplanet.com:4007
Listener on Port 4007
Sun ONE Application Server Administration Server
Administration website content
URL: http://sun.red.iplanet.com:80
Listener on Port 80
Sun ONE Application Server Instance: server1
Website content
URL: http://sun.red.iplanet.com:82
Listener on Port 82
Sun ONE Application Server Instance server82
Website content

Figure 17.6. Administration menu tree for the new server instance.



Instance Name: server82
Host Name: sun

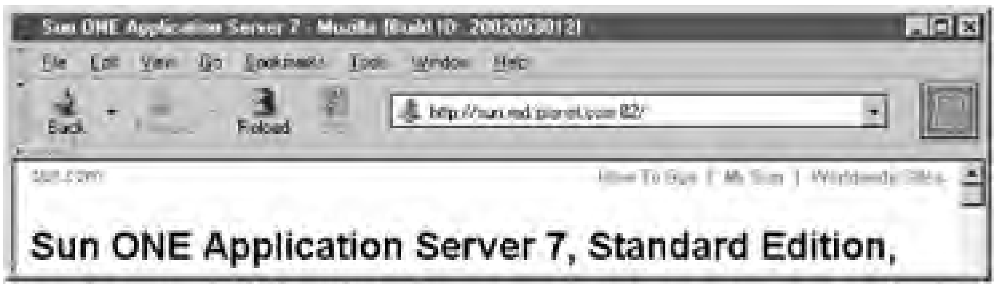


HTTP Port(s): 82
Status: Not Running
Configuration Directory: /disk1/sunone/as7/app
Install Version: Sun ONE Application Se

Testing the New Instance

From the web page in Figure 17.6, click on Start to start the new instance. Then use a web browser URL *http://sun.red.iplanet.com:82* to test the HTTP server on port 82 (Figure 17.7).

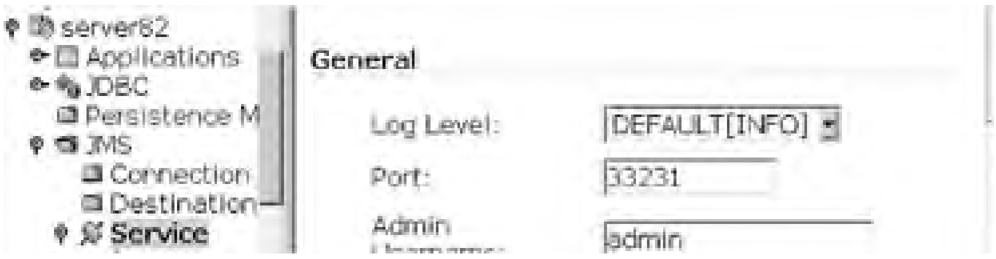
Figure 17.7. Web page from the new web site.


Sun ONE Application Server 7 - Mozilla (Build ID: 2002053012)
File
Edit
View
Go
Bookmarks
Tools
Window
Help
Back
Reload
http://sun.red.iplanet.com:82/
sun.com
How To Buy | My Sun | Worldwide Sites

Sun ONE Application Server 7, Standard Edition,

Next, test that message queue is listening. To find the listener port number of the message queue (again under menu item *App Server Instances*), under *server82* and *JMS*, click *Service* (Figure 17.8).

Figure 17.8. Finding the listener port number of the message queue.



Log Level: DEFAULT[INFO]
Port: 33231
Admin Username: admin

This shows that the port number is 33231. From a browser, go to the URL *http://sun.red.iplanet.com:33231*. The message queue responds with a default message to the browser.

The new instance was created under the domain1 file directory.

```
# cd /disk1/sunone/as7/appserv/domains/domain1
# ls
admin-server  server1      server82
```

To see the new instance processes, shut down the administration server and the application server instance *server1*. Then run the command ps -ef | grep sunone. In the listing below, note that the watchdog process (794) and the monitor process (795) are running as root and the other processes are running as iuser.

```
# ps -ef | grep sunone
root   794     1  0 19:11:16 ?        0:00 ./appservd-
  wdog -r /sunone/as7/appserv -d /disk1/sunone/as7/apps-
  erv/domains/dom
root   795   794  0 19:11:16 ?        0:01 appservd -r
  /sunone/as7/appserv -d /disk1/sunone/as7/apps-
  erv/domains/domain1/se
iuser   796   795  0 19:11:17 ?        0:51 appservd -r
  /sunone/as7/appserv -d /disk1/sunone/as7/apps-
  erv/domains/domain1/se
iuser   799   796  0 19:11:21 ?        0:00
  /disk1/sunone/as7/appserv/lib/Cgistub -f
  /tmp/server82-15341adc/.cgistub_796
iuser   800   799  0 19:11:21 ?        0:00
```

```
  /disk1/sunone/as7/appserv/lib/Cgistub -f
  /tmp/server82-15341adc/.cgistub_796
iuser   801   799  0 19:11:21 ?        0:00
  /disk1/sunone/as7/appserv/lib/Cgistub -f
  /tmp/server82-15341adc/.cgistub_796
iuser   802   796  0 19:11:32 ?        0:00 /bin/sh
  /sunone/as7/imq/bin/../bin/imqbrokerd -javahome
  /sunone/as7/jdk -name d
iuser   821   802  0 19:11:34 ?        0:10
  /sunone/as7/jdk/jre/bin/java -server -cp
  /sunone/as7/imq/bin/../bin/../lib/imqb
```

New server instances have the overhead of many processes running at once. However, there is another way to listen to more than one port without adding another application server instance: Add another listener and another virtual server to the originally installed instance.

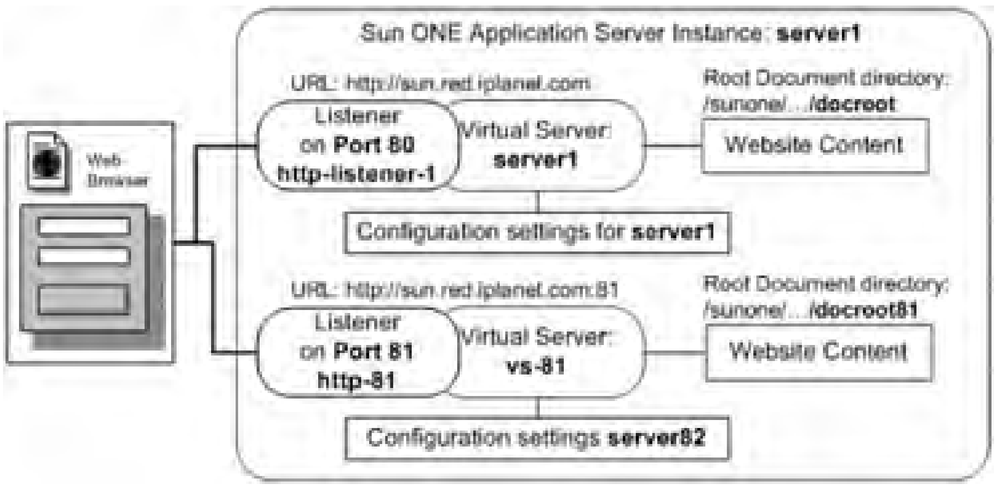
Since *server82* is not used in the rest of the book, delete the *server82* instance. Under *App Server Instance* (refer to Figure 17.6), click *server82* and then click *Stop*. After the instance is stopped, click *Delete*.

Adding a New Listener and Virtual Server

The goal in this exercise is to set up a new web site configuration that listens on port 81. This new configuration is a virtual server (we'll call it *vs-81*) within the current HTTP server; therefore, no new processes are added. The only thing added is configuration information. The new functionality is that the current HTTP server has a new web site.

As illustrated in Figure 17.9, the two virtual servers (*server1* and *vs-81*) have these features:

Figure 17.9. Multiple listeners and web sites with one HTTP server.


Sun ONE Application Server Instance: server1
URL: http://sun.red.iplanet.com
Listener on Port 80 http-listener-1
Virtual Server: server1
Root Document directory: /sunone/.../docroot
Website Content
Configuration settings for server1
Web Browser
URL: http://sun.red.iplanet.com:81
Listener on Port 81 http-81
Virtual Server: vs-81
Root Document directory: /sunone/.../docroot81
Website Content
Configuration settings server82

- Each is attached to a listener.
- Each has its own configuration information. (An example of configuration information might be setting up a cgi-bin directory. Virtual server *server1* may have a cgi-bin directory set up, whereas virtual server *vs-81* does not.)
- Each has its own root document directory.

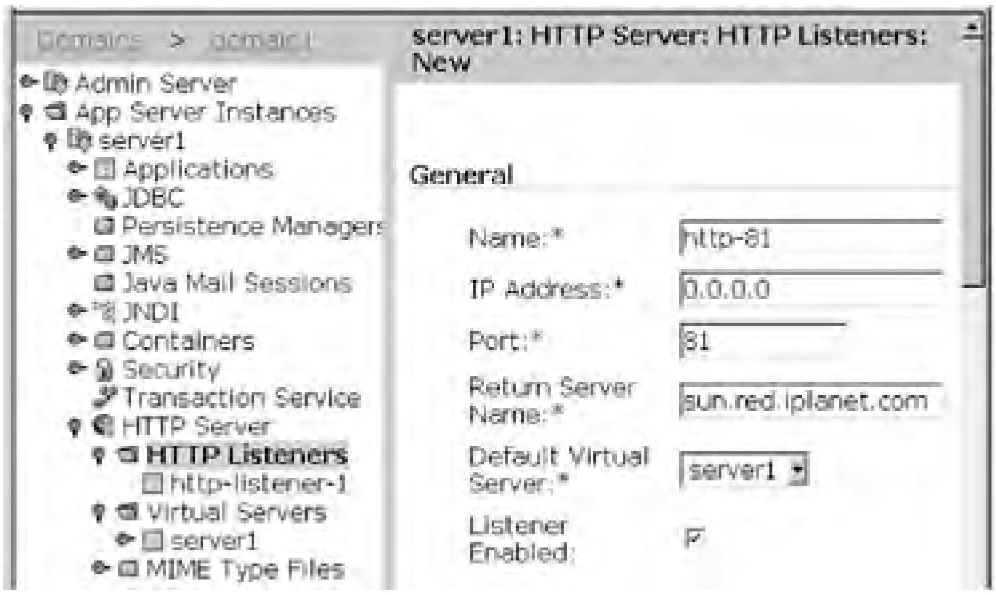
These are the steps needed to set up this configuration:

1. Add a new listener for port 81 called *http-81*.
2. Add a virtual server called *vs-81* that is attached to the new listener. Set the root document value to /sunone/as7/app-serv/domains/domain1/server1/docroot81.
3. Make the new root document file directory.
4. Create an index.html file in the new root document file directory. This serves as the home page.
5. Reconfigure the listener *http-81* to have the default virtual server *vs-81*.
6. Apply the changes and restart the server instance.
7. Use a browser to test the configuration: *http://sun.red.iplanet.com:81*.

Adding Listener for Port 81

In the window in Figure 17.10, click *HTTP Listeners and then* click *New*, as in Figure 17.11. Enter data as shown in the figure, and click *OK* at the top right. The new listener (*http-81*) now appears in the list under *Name* (Figure 17.11).

Figure 17.10. Add listener to current server instance.

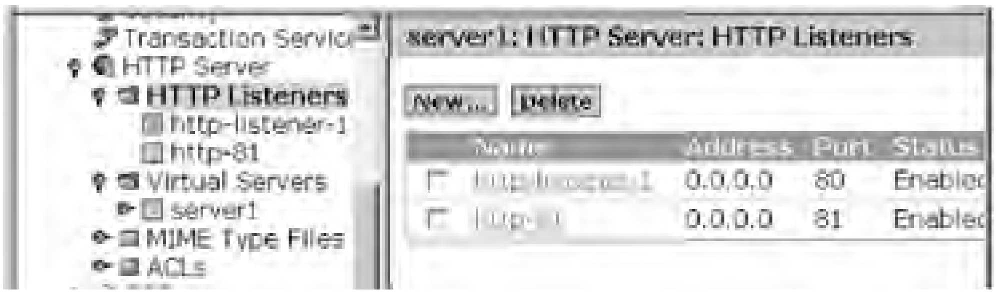


server1: HTTP Server: HTTP Listeners: New

General

Name:*	http-81
IP Address:*	0.0.0.0
Port:*	81
Return Server Name:*	sun.red.iplanet.com
Default Virtual Server:*	server1
Listener Enabled:	✓

Figure 17.11. New listener appears.



server1: HTTP Server: HTTP Listeners

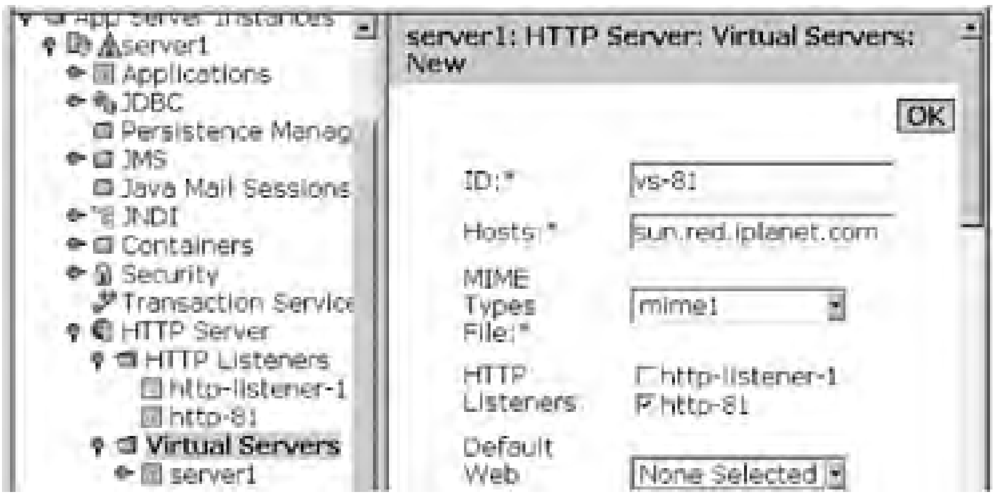
Name	Address	Port	Status
http-listener-1	0.0.0.0	80	Enabled
http-81	0.0.0.0	81	Enabled

Adding and Setting Up a Virtual Server

The steps for adding and setting up a new virtual server are as follows:

1. In the window in Figure 17.12, click *Virtual Servers* and then click *New*. Enter the data as shown in Figure 17.12. Note: The new listener on port 81 (*http-81*) is selected (there is a check in the box). Scroll down and change the virtual server *Document Root* to file directory /sunone/as7/apps-erv/domains/domain1/server1/docroot81.

Figure 17.12. Add virtual server to current HTTP server.

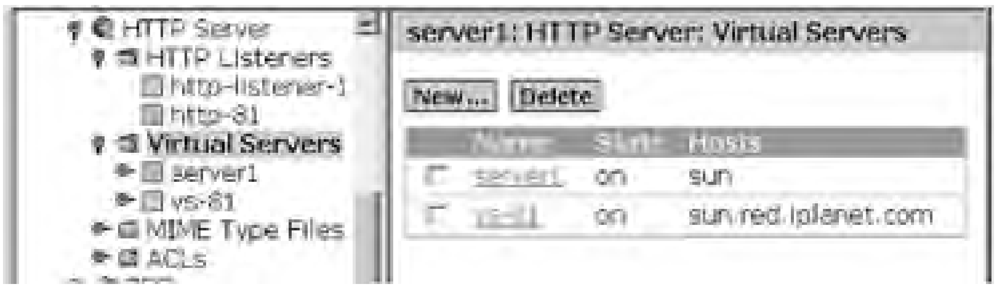


server1: HTTP Server: Virtual Servers: New

ID:*	vs-81
Hosts:*	sun.red.iplanet.com
MIME Types File:*	mime1
HTTP Listeners	☐ http-listener-1 ✓ http-81
Default Web	None Selected

2. Click *OK*. The new virtual server (*vs-81*) now appears in the list (see Figure 17.13).

Figure 17.13. New virtual server appears.



server1: HTTP Server: Virtual Servers

Name	State	Hosts
server1	on	sun
vs-81	on	sun.red.iplanet.com

3. Make the root document file directory. An example UNIX command is # mkdir /sunone/as7/appserv/domains/domain1/server1/docroot81.

4. Add an index.html file in the root directory. Here is an example index.html file.

```
<html><head>
<TITLE>Sun ONE Application Server home
   page</TITLE>
</head>
<body>
Virtual server listening on port 81.
</body>
</html>
```

5. Next, reset listener *http-81* to have the *Default Virtual Server* of *vs-81*. Click *http-81*, select *vs-81,* and click *Save* on the top right.

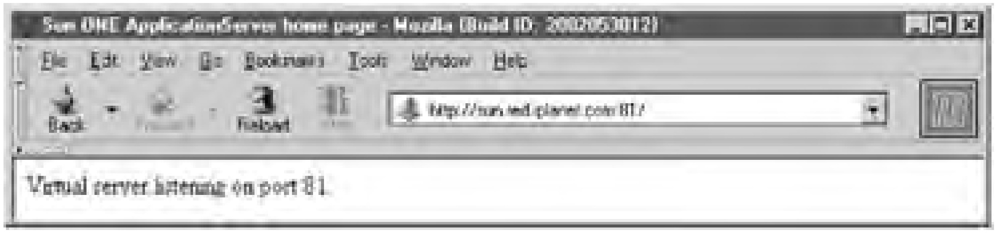
6. Click the message *Apply Changes Required* (on the top) to apply the changes, and restart the application server instance.

Testing the New Listener and Virtual Server

To test, go to URL *http://sun.red.iplanet.com:81*

As the window in Figure 17.14 shows, once again, the test is a success!

Figure 17.14. Successful testing of new listener and virtual server.



Sun ONE ApplicationServer home page - Mozilla (Build ID: 2002053012)

http://sun.red.iplanet.com:81/

Virtual server listening on port 81.

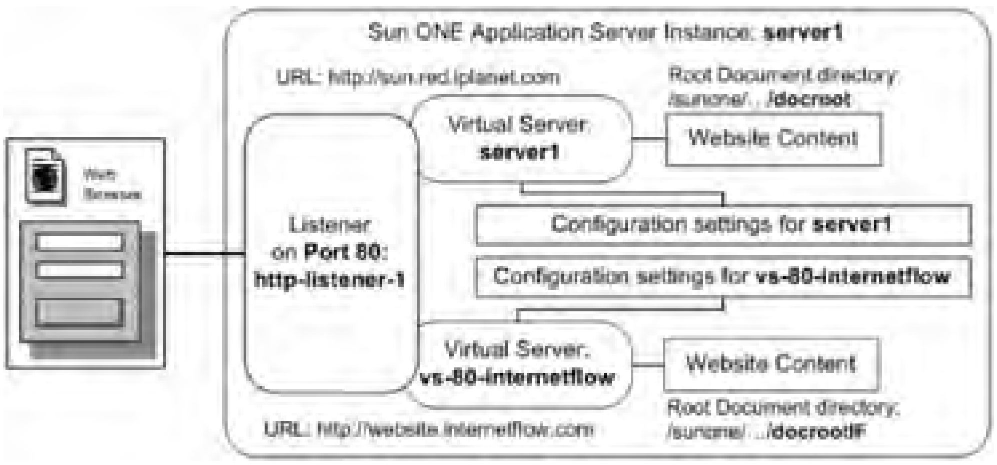
Multiple Hostnames on One IP Address on One Port

Another popular type of virtual server is a server that uses the same listener but has different web sites based on the hostnames being requested.

We will set up a new web site configuration that listens on port 80 for multiple hostnames. This new configuration is a virtual server within the current HTTP server. Again, no new processes are added; the only thing added is configuration information for a virtual server. The new functionality is that the current HTTP server has a new web site for each specific hostname.

As illustrated in Figure 17.15, each virtual server has these features:

Figure 17.15. Multiple hostnames on the same port and same IP address.


Sun ONE Application Server Instance: **server1**
URL: http://sun.red.iplanet.com
Root Document directory /sunone/...**/docroot**
Virtual Server **server1**
Website Content
Web Browser
Listener on **Port 80: http-listener-1**
Configuration settings for **server1**
Configuration settings for **vs-80-internetflow**
Virtual Server **vs-80-internetflow**
Website Content
Root Document directory /sunone/...**/docrootIF**
URL: http://website.internetflow.com

- Each is attached to the same listener that is listening to port 80 on the same IP address.
- Each has its own configuration information. (An example of configuration information might be setting up a cgi-bin directory. Virtual server *server1* may have a cgi-bin directory set up, whereas virtual server *vs-80-internetflow* does not.
- Each has its own root document directory.

These are the steps needed to set up this configuration.

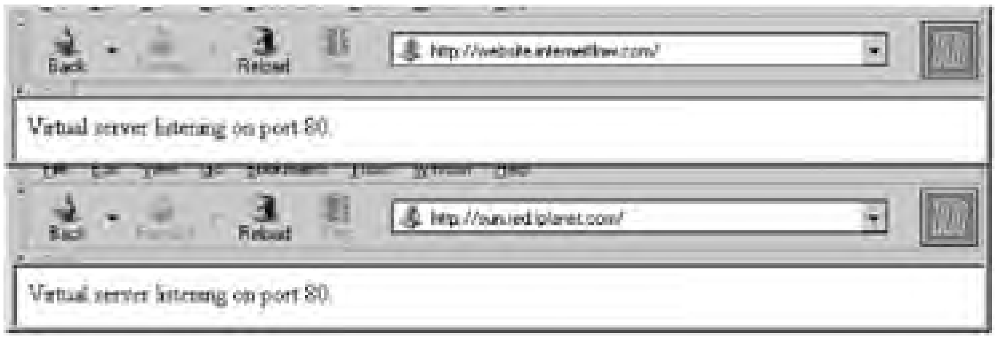
1. We need to assign another hostname to the same IP address. On my system, I added *website.internetflow.com* to my host file with the same IP address as *sun.red.iplanet.com*. On UNIX, the host file is /etc/hosts. On Windows 98, the host file is /windows/hosts. My host file entries are

   ```
   1.1.1.7 website.internetflow.com
   1.1.1.7 sun.red.iplanet.com
   ```

   There is one IP address with multiple hostnames.

   To confirm that the new hostname is working, from a browser, go to the URL *http://website.internetflow.com*. This opens the same web page as the URL *http://sun.red.iplanet.com* (Figure 17.16), thus confirming that the new hostname works.

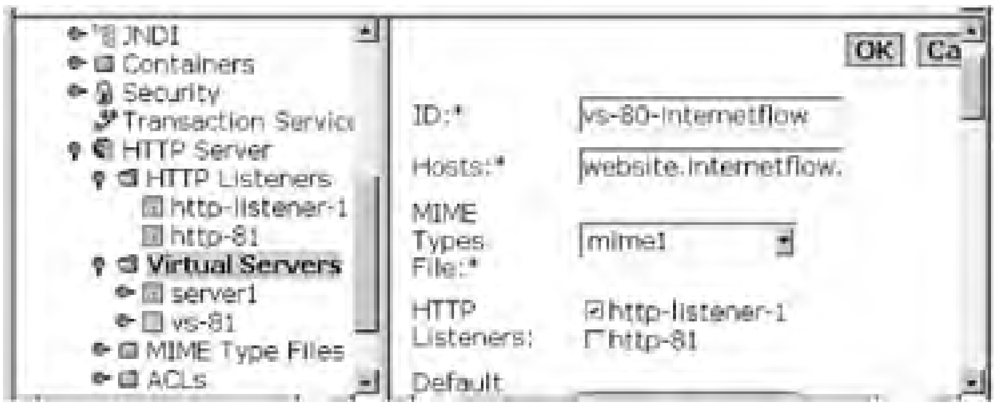
Figure 17.16. Confirming the new hostname works.


Virtual server listening on port 80.

Virtual server listening on port 80.

2. Now add another virtual server. Click *Virtual Servers* and then click *New* in Figure 17.17. Enter that data shown in the figure. Note: The original listener on port 80 (*http-listener-1*) is selected. Now scroll down (see Figure 17.17) and change the virtual server *Document Root* to file directory /sunone/as7/appserv/domains/domain1/server1/docrootIF.

Figure 17.17. Add another virtual server to the current HTTP server.



ID:* vs-80-Internetflow
Hosts:* website.internetflow.
MIME Types File:* mime1
HTTP Listeners: ☑ http-listener-1 ☐ http-81

3. Click the message *Apply Changes Required* (on the top) to apply the changes. Then restart the application server instance.

4. Make the root document file directory. An example UNIX command is

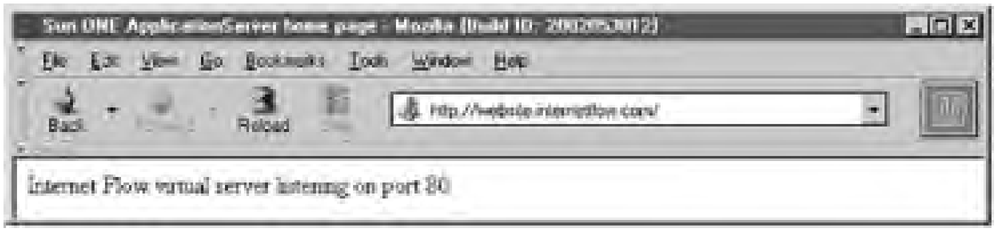
```
# mkdir /sunone/as7/appserv/domains/domain1
/server1/docrootIF.
```

5. Add an index.html file in the root directory. Here is an example index.html file.

```
<html><head>
<TITLE>Sun ONE Application Server home
  page</TITLE>
</head>
<body>
Internet Flow virtual server listening on port
  80.
</body>
</html>
```

6. To test the new virtual server, go to URL *http://website.internetflow.com*. The window in Figure 17.18 shows that the test is successful.

Figure 17.18. New virtual server test successful.


Internet Flow virtual server listening on port 80.

Now there are two hostnames on the same IP address and port number. Each hostname is configured to its own web site. You may be wondering how the HTTP server knows how to separate the requests to the proper virtual server? The answer is that the browser sends the hostname in the heading information when connecting to the HTTP server. In this exercise, if the heading value has the hostname *website.internetflow.com*, then the virtual server *vs-80-internetflow* handles the request. All other requests go to the default virtual server of the listener. In this case, the default is virtual server *server1*. Therefore, if the heading value has the hostname *sun.red.iplanet.com*, then the virtual server *server1* handles the request.

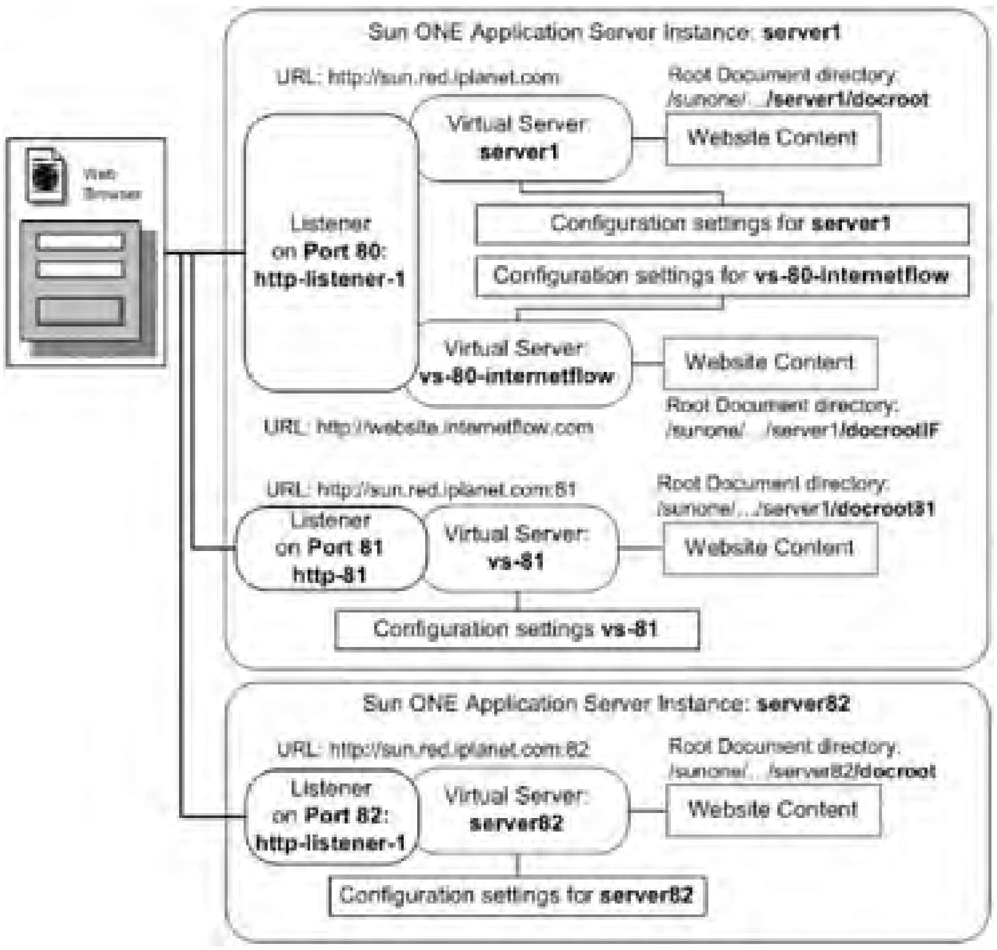
Summary

In this chapter a new application server instance was added, tested, and removed. Then new listeners and virtual servers were added, all to serve the following web sites:

1. From one application instance:
   - *http://sun.red.iplanet.com*
   - *http://website.internetflow.com*
   - *http://sun.red.iplanet.com:81*
2. From the second application server instance:
   - *http://sun.red.iplanet.com:82*

The Sun ONE Application Server's web site system architecture is amazingly easy to configure. Figure 17.19 is a diagram of the configurations that have been demonstrated in this chapter. Think about it—does it take a web site genius to configure these web sites, or can almost anybody do it with a little training? The answer is almost anybody with a little training because the Sun ONE Application Server is so fast and easy to learn to use.

Figure 17.19. The instances, listeners, and virtual servers from this chapter.


Sun ONE Application Server Instance: server1
URL: http://sun.red.iplanet.com
Root Document directory: /sunone/.../server1/docroot
Web Browser
Virtual Server: server1
Website Content
Listener on Port 80: http-listener-1
Configuration settings for server1
Configuration settings for vs-80-internetflow
Virtual Server: vs-80-internetflow
Website Content
URL: http://website.internetflow.com
Root Document directory: /sunone/.../server1/docrootIF
URL: http://sun.red.iplanet.com:81
Root Document directory: /sunone/.../server1/docroot81
Listener on Port 81 http-81
Virtual Server: vs-81
Website Content
Configuration settings vs-81
Sun ONE Application Server Instance: server82
URL: http://sun.red.iplanet.com:82
Root Document directory: /sunone/.../server82/docroot
Listener on Port 82: http-listener-1
Virtual Server: server82
Website Content
Configuration settings for server82

A closing note: The exercises in this chapter were all done using the administration web site. It is possible to use the *asadmin* command-line tool to do the same configurations. This means that scripts can be created to make the configurations even easier and faster.

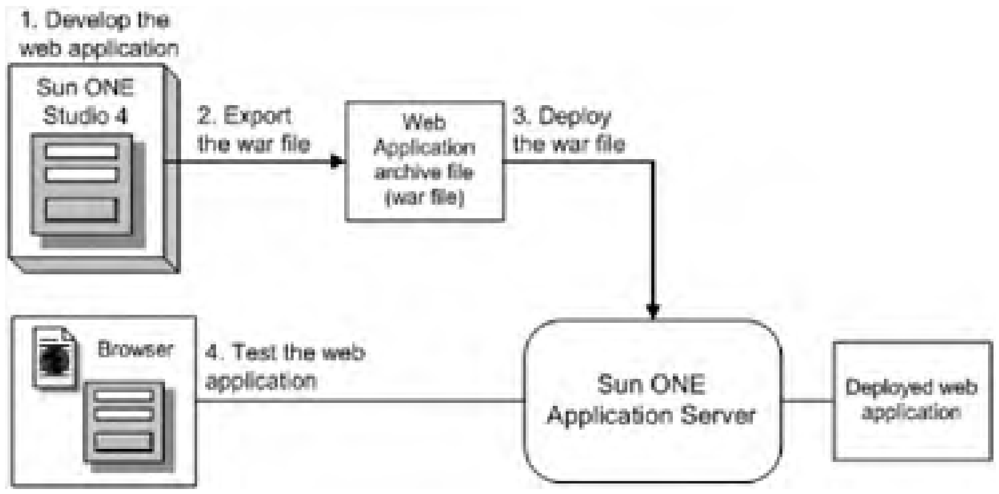
Even though this chapter has covered a number of different configurations, more are possible. I believe you will find the Sun ONE Application Server to be an excellent, complete web site management system.

Chapter 18. Using Sun ONE Studio 4 to Develop Web Applications

The Sun ONE Studio 4 is a development environment used to develop web applications and web services. This chapter will help you get started and become familiar with using the Sun ONE Studio 4. You will develop and package a web application using Studio 4. Then you will deploy the application and run it on the Sun ONE Application Server.

The steps we will perform are shown in Figure 18.1 and listed as follows:

Figure 18.1. Creating and deploying applications.


1. Develop the web application
Sun ONE Studio 4
2. Export the war file
Web Application archive file (war file)
3. Deploy the war file
Browser
4. Test the web application
Sun ONE Application Server
Deployed web application

1. Create a web application that has an HTML web page, graphics, a JSP, and a servlet (step 1).
2. Package (export) this application into a WAR file (step 2).
3. Deploy the application WAR file using the Sun ONE Application Server Administration Server system and apply the changes to the application server (step 3).
4. Test the web application (step 4).

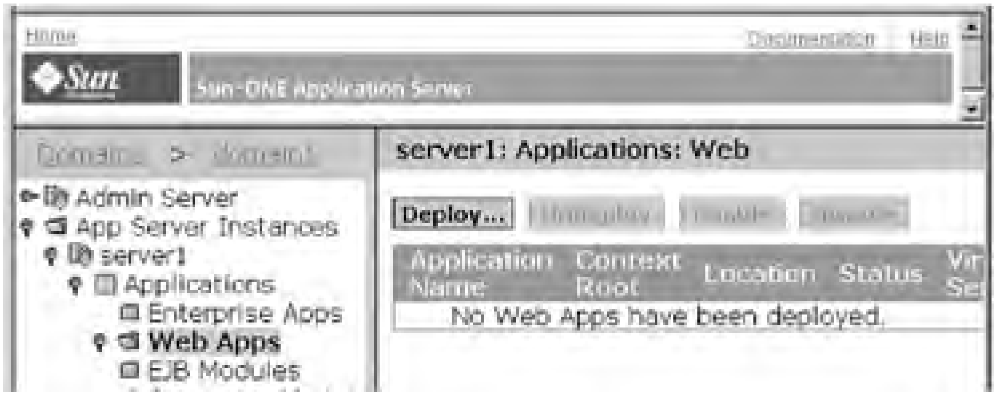
A web application is made up of web pages, graphics, JSPs, and servlet program files. These files are zipped into a web application archive (WAR) file. The WAR files are deployed to application servers that have a web container. While this chapter covers deploying these files on the Sun ONE Application Server, the WAR files created here can be deployed to the Sun ONE Web Server.

Deploying a Sample Web Application

Before moving to the development part of the chapter, you should deploy the sample web application that comes with the Sun ONE Application Server. This demonstrates steps 3 and 4 in Figure 18.1.

Start the Sun ONE Application Server Administration Server if you do not already have it running; then log in. Once logged in, under *App Server Instances/server1/Applications* on the left side of the screen, click *Web Apps*. This brings you to the web page in Figure 18.2. Click *Deploy* and the window in Figure 18.3 opens. Put the full-path filename in the *File Path* field and click *OK*. The WAR file on the application server computer is located at /sunone/as7/apps-erv/samples/webapps/simple/webapps-simple.war.

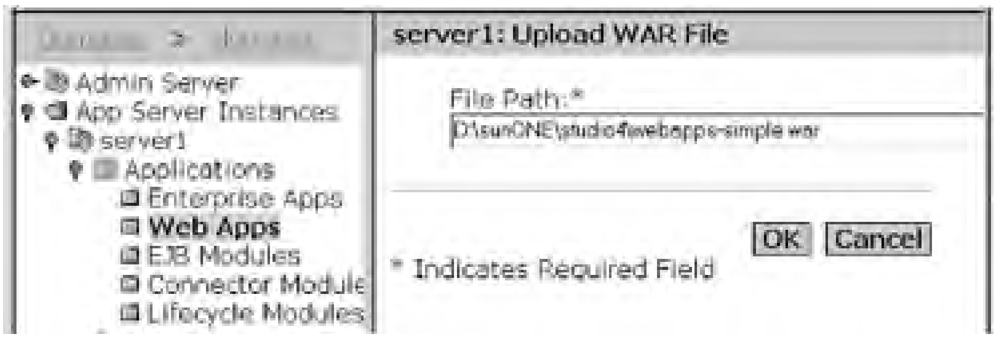
Figure 18.2. Web application management web page.



server1: Applications: Web

Application Name	Context Root	Location	Status	Vir Se
No Web Apps have been deployed.				

Figure 18.3. The WAR file full-path file name on Windows computer.



server1: Upload WAR File

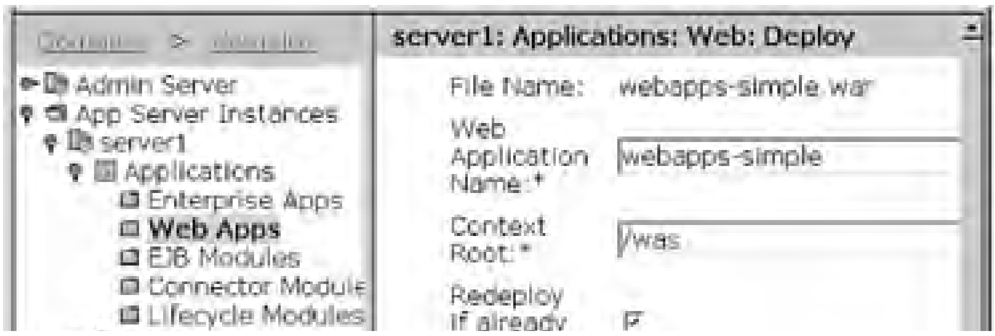
File Path:* D:\sunONE\studio4\webapps-simple.war

* Indicates Required Field

The WAR file D:\sunONE\studio4\webapps-simple.war in Figure 18.3 was copied from the UNIX computer onto my Windows computer. From the Windows computer, I used a browser to log in to the Sun ONE Application Server Administration Server (on the UNIX computer) to deploy the WAR file.

As in Figure 18.4, enter the *Context Root /was* and click OK.

Figure 18.4. Web application file deploy web page.



server1: Applications: Web: Deploy

File Name: webapps-simple.war

Web Application Name:* webapps-simple

Context Root:* /was

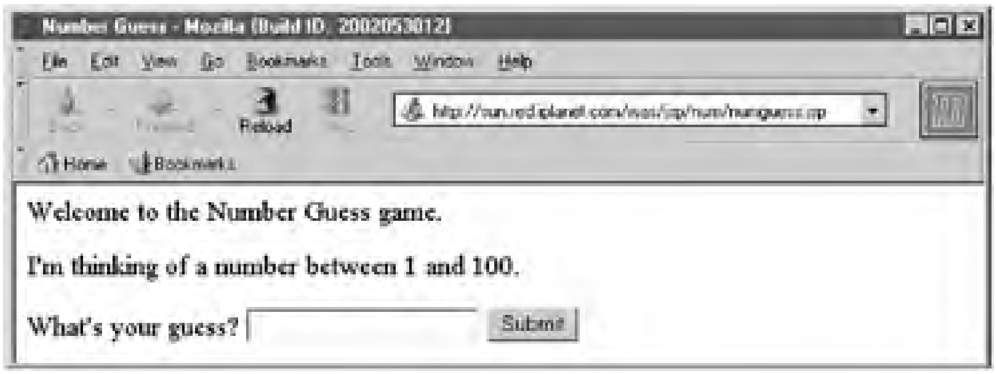
Redeploy If already ☑

The new sample application is now added to the list of web applications and is ready for testing. From a browser, go to the URL

http://sun.red.iplanet.com/was.

The web application menu appears. Click *JSP Examples* to get list of JSPs included in the sample web application. Click *Execute* beside *Number Guess* to test this JSP. After clicking *Execute*, the JSP is compiled and run; then the web page in Figure 18.5 is displayed in your browser. Play the game; try to guess the number.

Figure 18.5. Number Guess game.


Number Guess - Mozilla (Build ID: 2002053012)
File
Edit
View
Go
Bookmarks
Tools
Window
Help
Reload
http://sun.red.iplanet.com/was/jsp/num/numguess.jsp
Home
Bookmarks
Submit

Welcome to the Number Guess game.

I'm thinking of a number between 1 and 100.

What's your guess?

This web application containing the number game was packaged in the WAR file named webapps-simple.war. This WAR file has HTML web pages, graphics, JSPs, and servlets.

[Team LiB]
PREVIOUS NEXT

Before We Begin Developing...

Installing Sun ONE Studio 4

Either you installed Sun ONE Studio 4 with the Sun ONE Application Server 7 or you need to install it. To see if you have already installed it, look for the file <install directory>/studio4/bin/runide.sh (or runide.exe) in the install directory of the Sun ONE Application Server 7. If it is there, you have it installed. If you have not installed Studio yet, you will find it on the book DVD, or you can download it from *www.sun.com*.

There are two software packages required to get the Sun ONE Studio 4 to run on a computer:

1. Java 2 Platform, Standard Edition v1.4 (J2SE)
2. The Sun ONE Studio 4 software itself

The version of the Studio software used in this chapter is the Community Edition. This software is free. A second version of the Sun ONE Studio 4 software is the Enterprise Edition. It may be used free for a trial period; then there is cost to the product (consult the Sun web site for details). The basic difference between the two is that the Community Edition does not do EJBs. The Enterprise Edition has tools to make developing EJBs a straightforward process.

If you are unsure as to whether you have J2SE v1.4 on your computer, download the Sun ONE Studio 4 software and install it. During the installation, the installer checks for J2SE v1.4. If you don't have it already, download J2SE v1.4 and install it. Follow these steps:

1. Once connected to *www.sun.com*, select *Downloads.*
2. On the left select *View All.*
3. Find *Java 2 Platform, Standard Edition v1.4* (the products are listed in alphabetical order).
4. Select your operating system. Click *Download* under the heading *SDK*.
5. Follow the steps to download the software onto your computer.
6. Put the file into a working directory so that you can retain it and use it later.

To download Sun ONE Studio 4:

1. Once connected to *www.sun.com*, select *Downloads.*
2. On the left select *View All.*
3. Find *Sun ONE Studio 4* (the products are listed in alphabetical order).
4. Select your operating system. Click *Download*.
5. Follow the steps to download the software onto your computer.
6. Put the file into a working directory so that you can retain it and use it later.

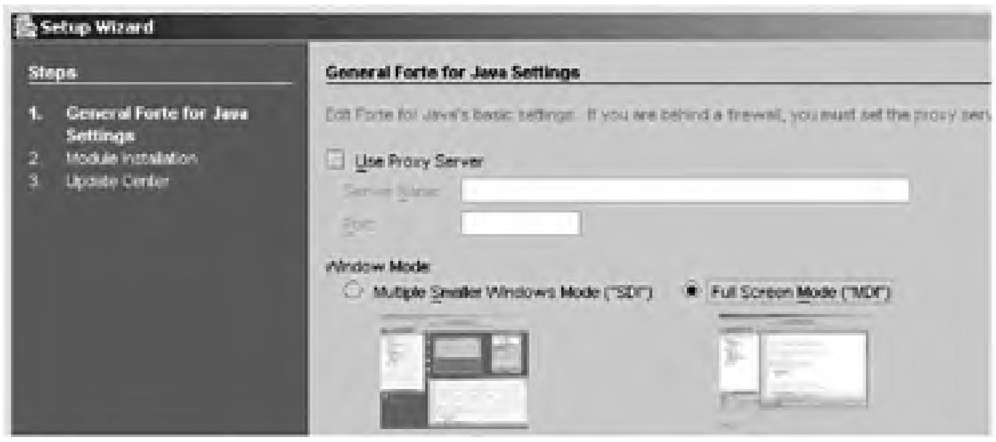
Once the software packages are downloaded, it is a straightforward process to run the installation programs. When you have Sun ONE Studio 4 installed, follow the steps outlined below to get it running.

Running Sun ONE Studio 4

If you are using Sun ONE Studio 4 (studio4) from UNIX with the Sun ONE Application Server, the startup command is /sunone/as7/forte4j/bin/fjscript.sh. From Windows, use *Start/Programs/Forte for Java 4 CE/Forte for Java*.

The first time studio4 starts up, I select *Full Screen Mode* because I like it better than the other mode (Figure 18.6). Note: You can change this from a menu option later.

Figure 18.6. Starting window for Sun ONE Studio 4.



Setup Wizard

Steps

1. General Forte for Java Settings
2. Module Installation
3. Update Center

General Forte for Java Settings

Edit Forte for Java's basic settings. If you are behind a firewall, you must set the proxy serv

☐ Use Proxy Server

Server Name:

Port:

Window Mode

○ Multiple Smaller Windows Mode ("SDI") [✓ ● Full Screen Mode ("MDI")]

Select your *Window Mode*; then click *Finish*.

Creating a Web Application with Sun ONE Studio 4

Step 1. Developing a Web Application

There are several activities that make up this step. You will

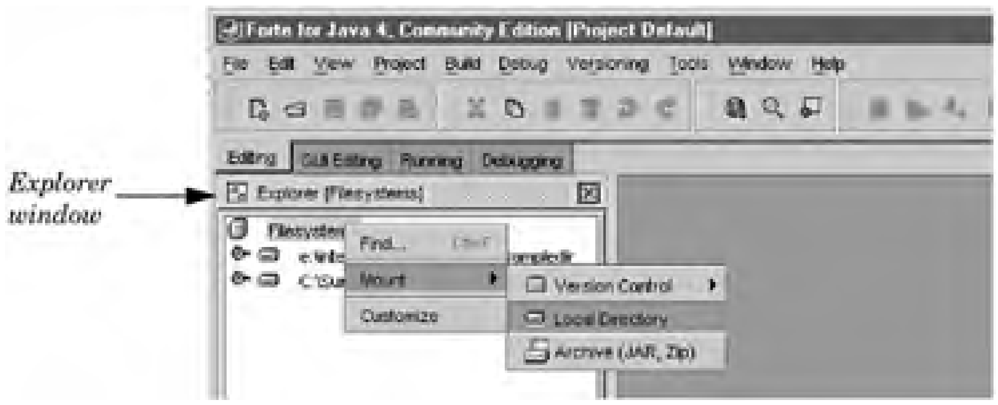
- Set up a file directory for developing a web application
- Create a web module in the file directory
- Add a servlet to the application
- Add a JSP into the application
- Add a folder and graphic file into the application
- Add an HTML web page into the application

Let's begin.

Set Up a File Directory for Developing a Web Application

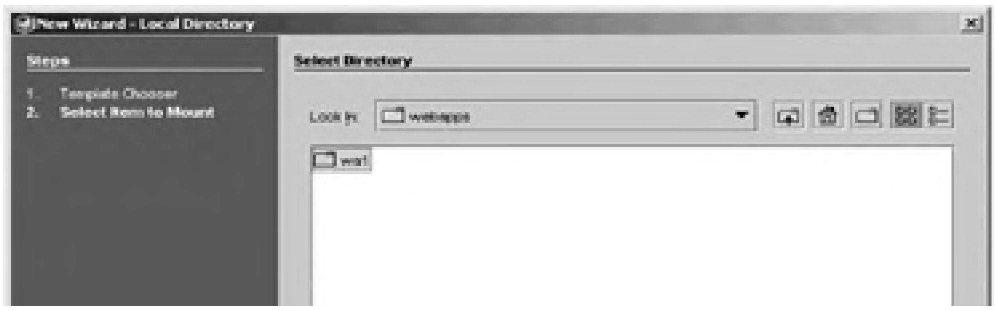
Before selecting a file directory in Sun ONE Studio 4, from a command window make a file directory named c:\sun\webapps\wa1 or /sun/webapps/wa1. Now, using the file directory *Explorer (Filesystems)* window of Sun ONE Studio 4 (Figure 18.7), right click on *Filesystems* and select *Mount/Local Directory*. This *Explorer* window has features similar to the file explorer in UNIX or Windows. Note: For Windows users, *mount* means add; *unmount* means remove.

Figure 18.7. Add a local file directory.



Select the file directory that you have just made and click *Finish* (Figure 18.8).

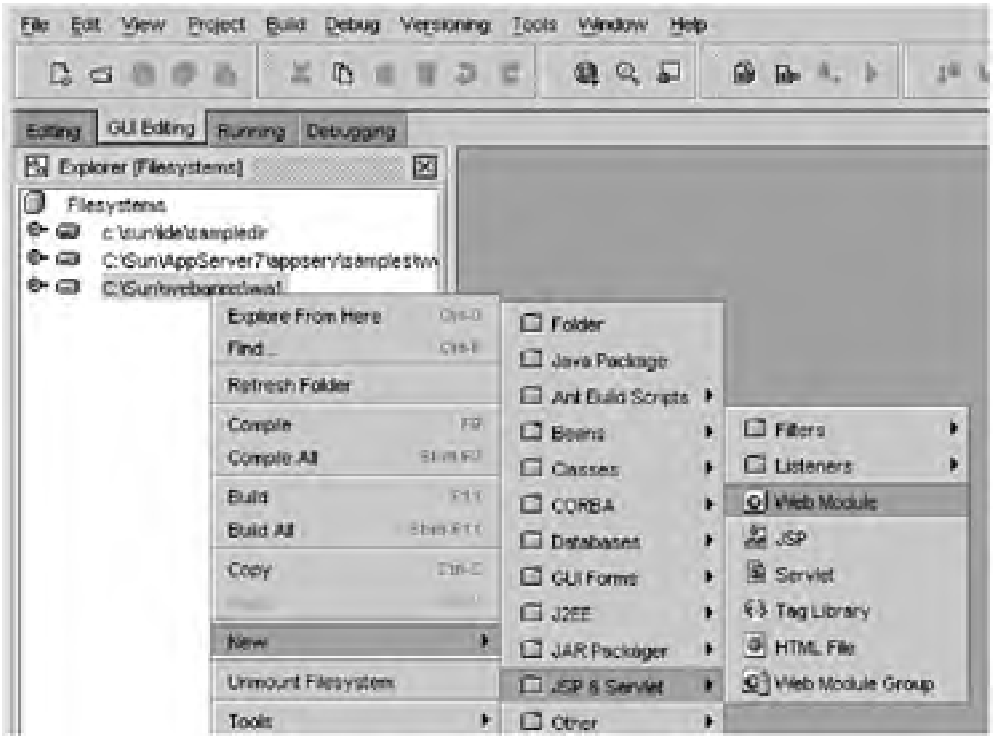
Figure 18.8. Finish adding local file directory.



Create a Web Module in the File Directory

Before we can add servlets, we need to create a *Web Module* structure. Following steps in Figure 18.9, create a web module by right clicking on the newly mounted web application file directory and selecting *New/JSP & Servlet/Web Module*.

Figure 18.9. Create a web module.



In the *New Wizard—Web Module* window (Figure 18.10), the web application directory is displayed, click *Finish*. Click *OK* to make the conversion (Figure 18.11).

Figure 18.10. Web application directory displayed.

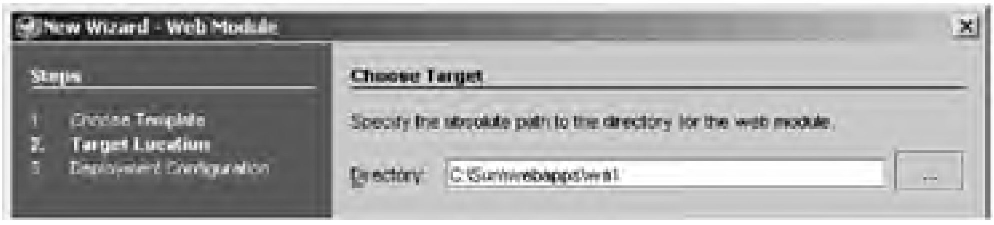
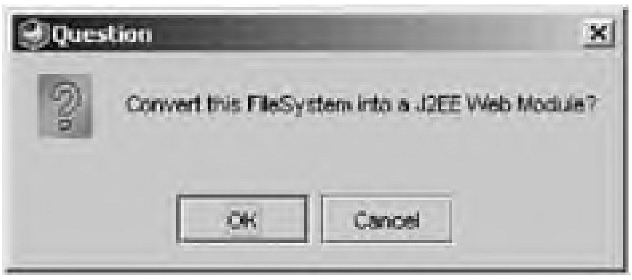

New Wizard - Web Module
Steps
1. Choose Template
2. Target Location
3. Deployment Configuration
Choose Target
Specify the absolute path to the directory for the web module.
Directory: C:\Sun\webapps\wat
...

Figure 18.11. Conversion window.

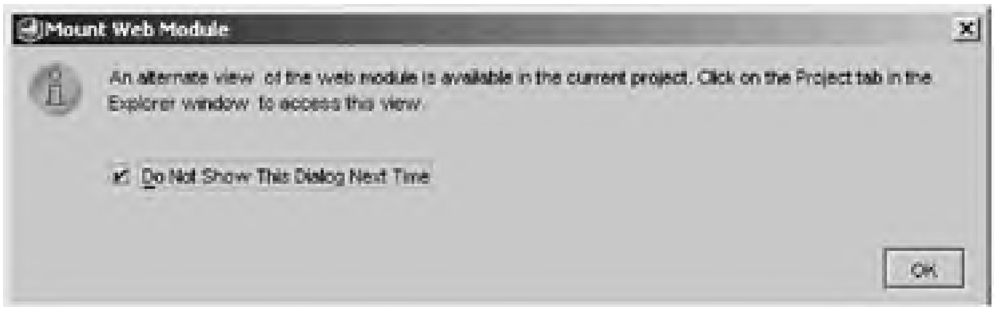


Question

Convert this FileSystem into a J2EE Web Module?

On the *Mount Web Module* prompt, check the *Do Not Show* box (Figure 18.12). This prompt is not needed. Then click *OK*.

Figure 18.12. Mounting the web module.



Mount Web Module

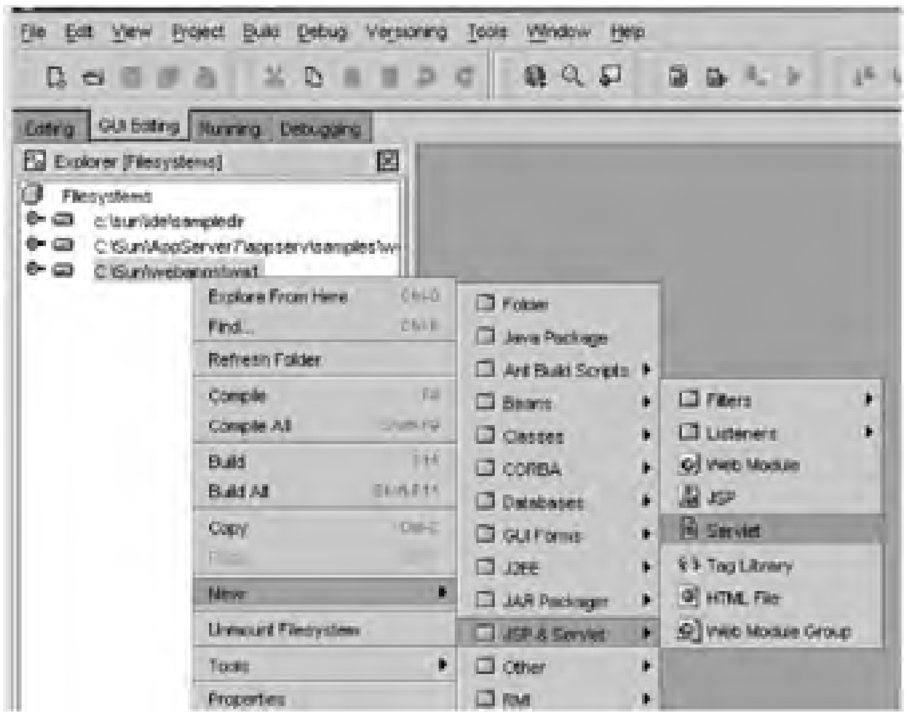
An alternate view of the web module is available in the current project. Click on the Project tab in the Explorer window to access this view.

☑ Do Not Show This Dialog Next Time

Add a Servlet to the Application

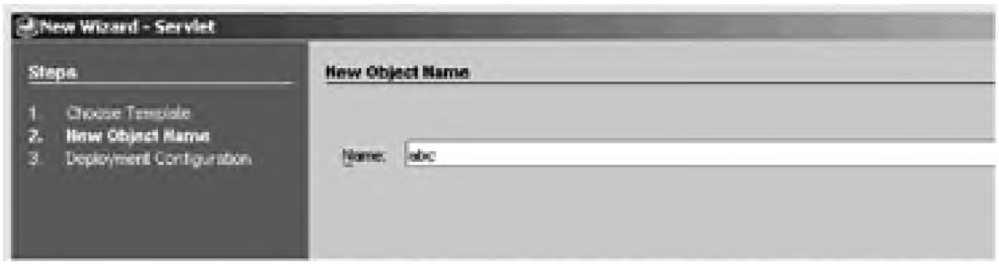
The steps taken at the beginning are visible in Figure 18.13. To add a servlet, right-click on the mounted web application directory as in Figure 18.13: C:\Sun\webapps\wa1. Then New/JSP & Servlet/JSP.

Figure 18.13. Adding a servlet.



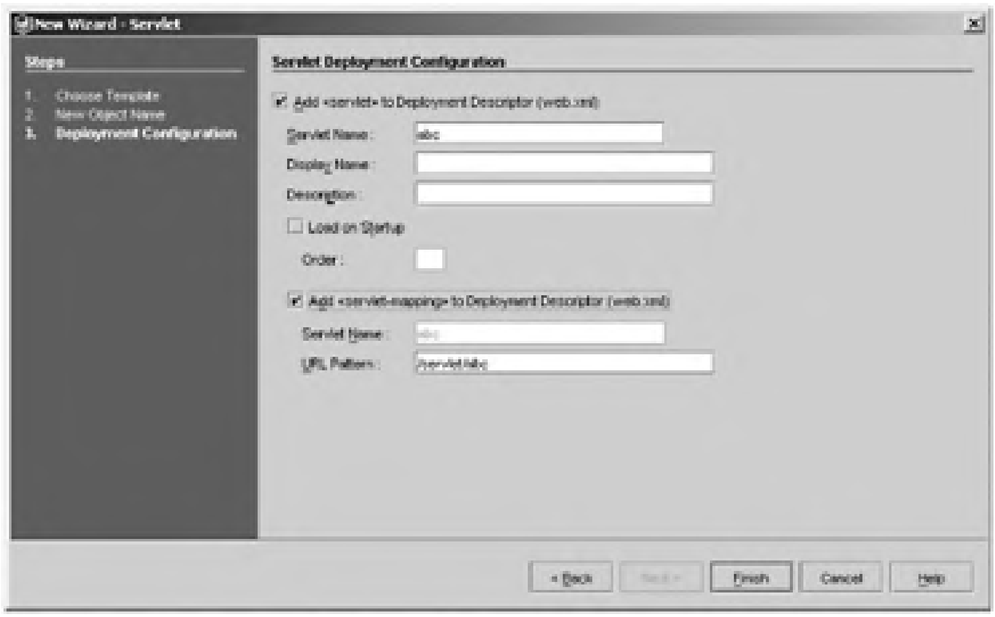
In the *New Wizard—Servlet* window (Figure 18.14), enter *abc* as the servlet name and click *Next*. The window shown in Figure 18.15 appears.

Figure 18.14. Enter servlet name.



Name: abc

Figure 18.15. Servlet deployment description.



[✓] Add <servlet> to Deployment Descriptor (web.xml)

Servlet Name: abc

Display Name:

Description:

[ ] Load on Startup

Order:

[✓] Add <servlet-mapping> to Deployment Descriptor (web.xml)

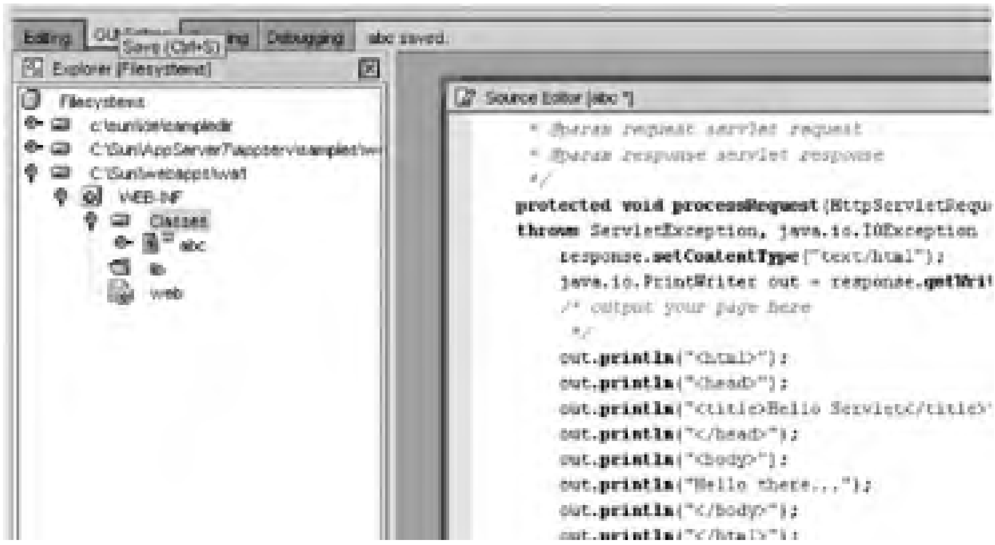
Servlet Name: abc

URL Pattern: /servlet/abc

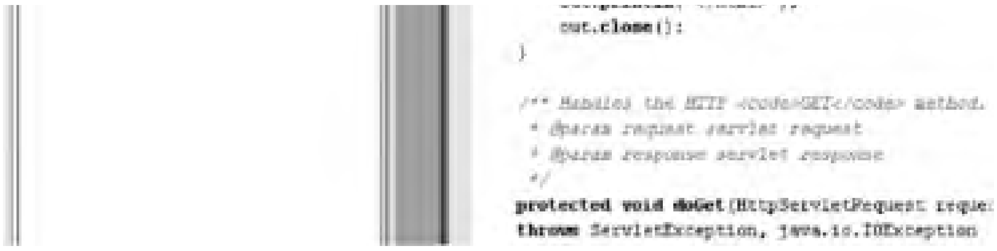
An important thing to note in Figure 18.15 is the *URL Pattern*. This pattern is used to call the servlet after the web application has been deployed. For example, this servlet is accessed by the URL *http://<hostname><Context Root><URL Pattern>*, or *http://sun.red.iplanet.com/wa1/servlet/abc*.

Now click *Finish*. A generic servlet is created, and the source code is displayed (Figure 18.16).

Figure 18.16. The Java program for servlet abc.



```
     * @param request servlet request
     * @param response servlet response
     */
    protected void processRequest(HttpServletRequ
    throws ServletException, java.io.IOException
        response.setContentType("text/html");
        java.io.PrintWriter out = response.getWri
        /* output your page here
         */
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Hello Servlet</title>
        out.println("</head>");
        out.println("<body>");
        out.println("Hello there...");
        out.println("</body>");
        out.println("</html>");
```



```
        out.close();
    }

    /** Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     */
    protected void doGet(HttpServletRequest reque
    throws ServletException, java.io.IOException
```

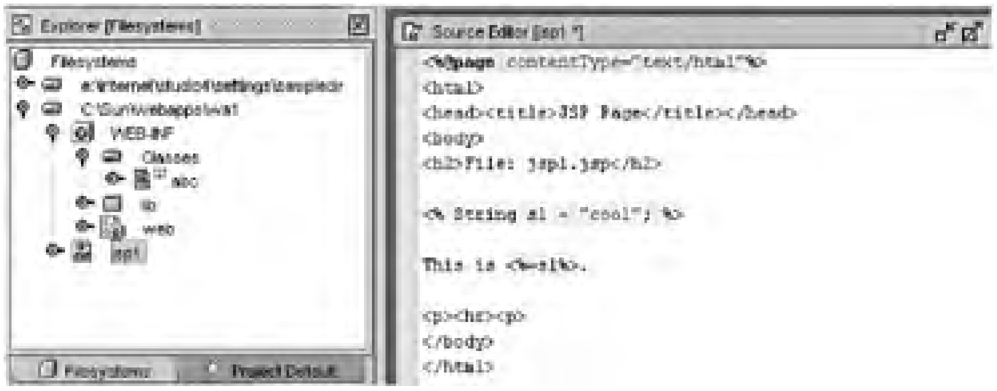
Modify the servlet source code as in Figure 18.16. Change only the *processRequest* method; leave everything else as is.

Add a JSP into the Application

This web application is organized under the file directory that has been mounted (in this example C:\Sun\webapps\wa1). The servlet programs are part of the *Web Module* (refer to Figure 18.9). Servlets are put into the WEB-INF directory under *Classes*. JSPs are web site components; web site components do not go into the WEB-INF directory.

To add a JSP, right click on the application file directory C:\Sun\webapps\wa1, and select *New/JSP & Servlet/JSP* (same as you did in Figure 18.9). When the *New Wizard—JSP* window appears (similar to the servlet wizard in Figure 18.14), enter *jsp1* in the object *Name* and click the *Finish* button. A default JSP is created. Change your JSP source code to look like the code in Figure 18.17.

Figure 18.17. Sample JSP.



```
<%@page contentType="text/html"%>
<html>
<head><title>JSP Page</title></head>
<body>
<h2>File: jsp1.jsp</h2>

<% String s1 = "cool"; %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

Add a Folder and a Graphics File into the Application

Graphics files are also web site components. Just like other web site components, they do not go into the WEB-INF directory.

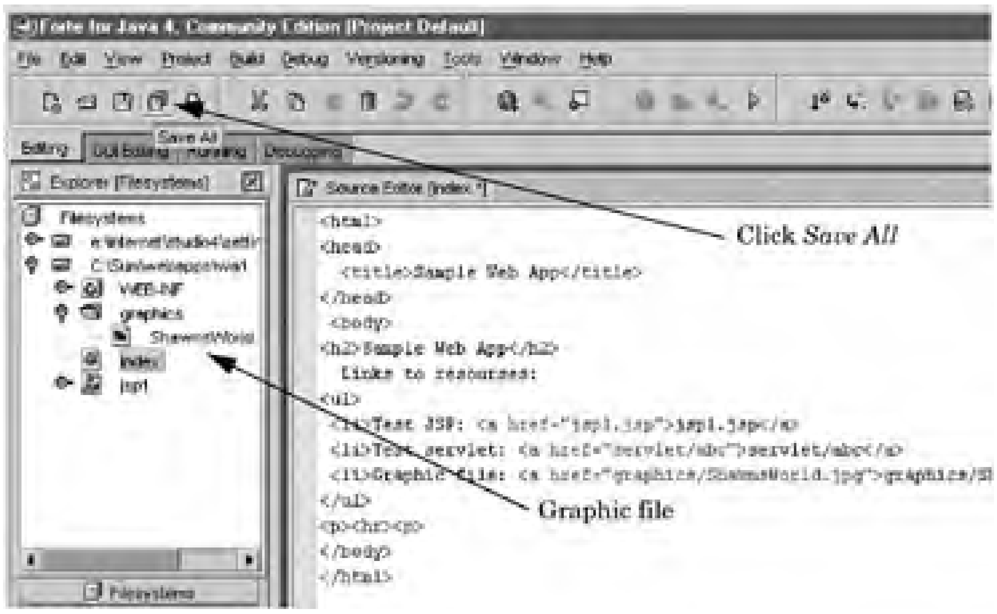
First, add a folder for the graphics file. Right click on the application file directory C:\Sun\webapps\wa1, and select *New/Folder* (again, these selections are visible in Figure 18.9). In the *New Wizard—Folder* prompt, enter *graphics* and click the *Finish* button.

Next, copy a graphics file into the folder. You may use any graphics file available on your computer. (I copied ShawnsWorld.jpg into my graphics directory.)

Add an HTML Web Page into the Application

HTML files are web site components as well, and, as we've said before, they do not go into the WEB-INF directory. To add an HTML web page, right click on the application file directory C:\Sun\webapps\wa1, and select *New/Other/HTML File* (again, refer to Figure 18.9 or Figure 18.13). On the *New Wizard—HTML File* prompt, enter *index* and click the *Finish* button. Then in the Source Editor, enter the HTML code as in Figure 18.18.

Figure 18.18. Web application index.html web page.


Forte for Java 4, Community Edition [Project Default]
File Edit View Project Build Debug Versioning Tools Window Help
Save All
Editing GUI Editing Running Debugging
Explorer [Filesystems]
Filesystems
WEB-INF
graphics
ShawnsWorld
index
jsp1
Source Editor [index *]
Click Save All
Graphic file

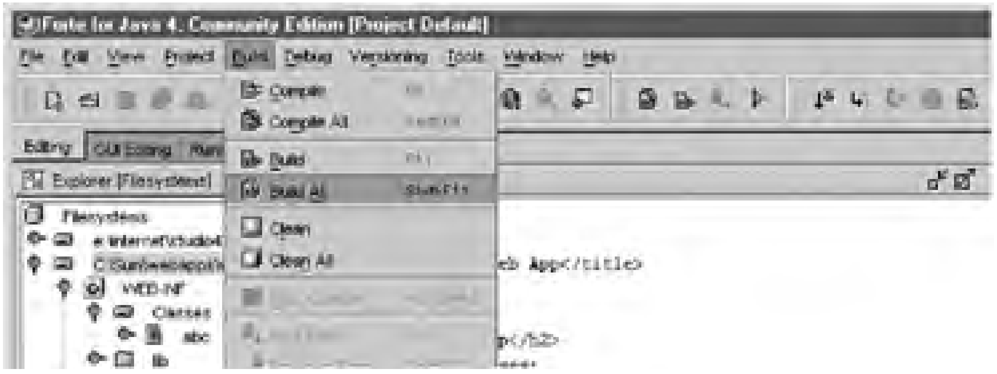
```
<html>
<head>
  <title>Sample Web App</title>
</head>
 <body>
<h2>Sample Web App</h2>
  Links to resources:
<ul>
 <li>Test JSP: <a href="jsp1.jsp">jsp1.jsp</a>
 <li>Test servlet: <a href="servlet/abc">servlet/abc</a>
 <li>Graphic file: <a href="graphics/ShawnsWorld.jpg">graphics/Sl
</ul>
<p><hr><p>
</body>
</html>
```

All the files have now been added to the web application, so click *Save All*. The web application is ready to be packaged.

Step 2. Packaging (Exporting) the Application into a WAR File

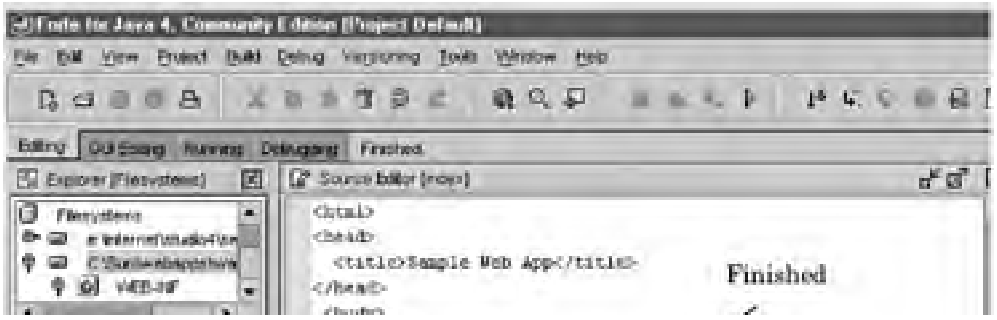
We have put together the web application, and now it is ready to be tested and packaged up. The next step is to build the application. First, single click on the application directory C:\Sun\webapps\wa1 (Figure 18.19). Next click on *Build* on the main menu and select the *Build All* item on the Build menu.

Figure 18.19. Building the application.

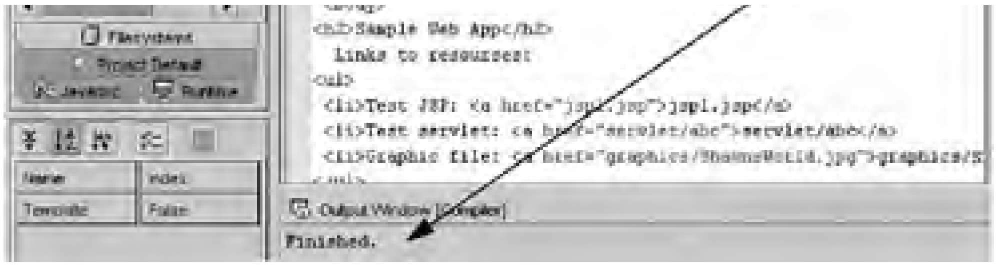

Forte for Java 4, Community Edition [Project Default]
File Edit View Project Build Debug Versioning Tools Window Help
Compile
Compile All
Build
Build All
Clean
Clean All
Editing GUI Editing Running
Explorer [Filesystems]
Filesystems
C:\Sun\webapps\wa1
WEB-INF
Classes
abc
lib

When the build is completed, a *Finished* message appears in the *Output Window* (Figure 18.20). The application is now ready to be exported into a WAR file. Single click on the application directory C:\Sun\webapps\wa1 (Figure 18.21). Click the menu option *Tools* and select the menu item *Export WAR File*.

Figure 18.20. The build is finished.


Forte for Java 4, Community Edition [Project Default]
File Edit View Project Build Debug Versioning Tools Window Help
Editing GUI Editing Running Debugging
Finished.
Explorer [Filesystems]
Source Editor [index]
Filesystems
C:\Sun\webapps\wa1
WEB-INF
Finished

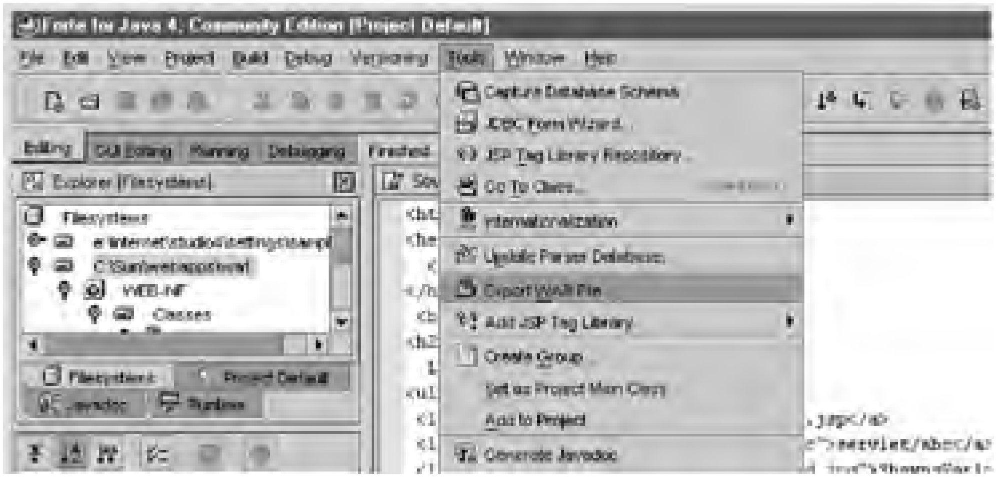
```
<html>
<head>
  <title>Sample Web App</title>
</head>
```



```
<h1>Sample Web App</h1>
  Links to resources:
<ul>
 <li>Test JSP: <a href="jsp1.jsp">jsp1.jsp</a>
 <li>Test servlet: <a href="servlet/abc">servlet/abc</a>
 <li>Graphic file: <a href="graphics/ShawnsWorld.jpg">graphics/S
```

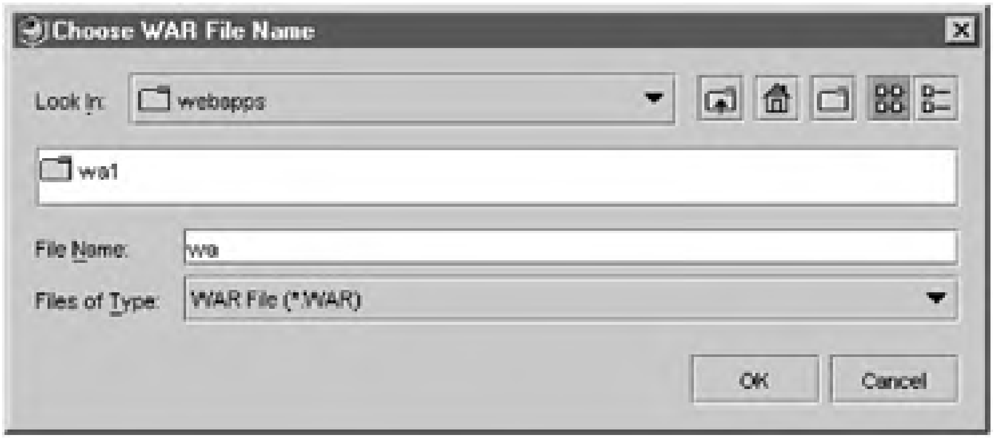
```
Finished.
```

Figure 18.21. Exporting the application.



When prompted, select a directory to export the web application to—using the same directory as the application directory. I do this to remember where to find the WAR file in the future. Now that we have chosen export options (Figure 18.22), click *OK* and the WAR file is created.

Figure 18.22. Choosing export options.



Look In: webapps

wa1

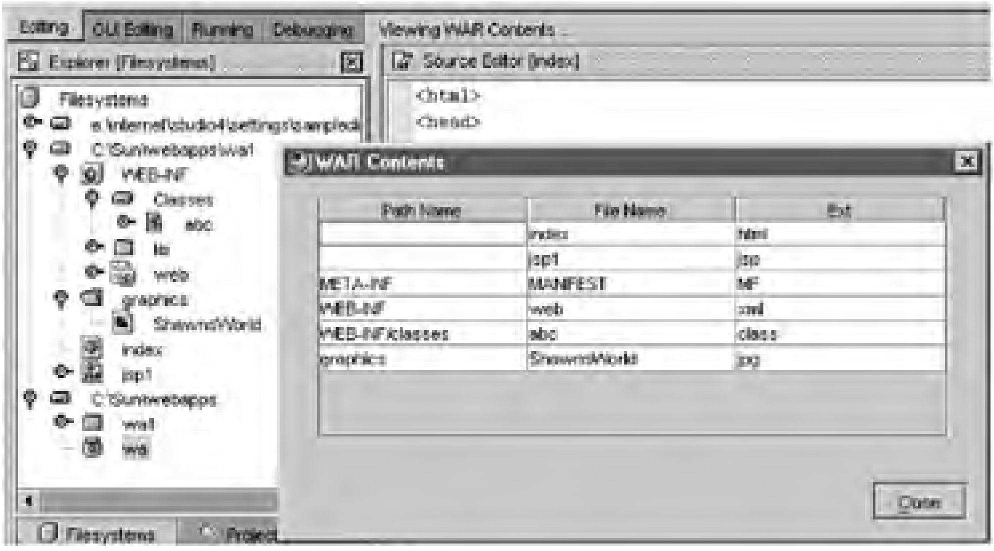
File Name: wa

Files of Type: WAR File (*.WAR)

View WAR File Contents

Let's view the contents of the WAR file you just created. To do this from Sun ONE Studio 4, mount the application directory by doing a right click on *Filesystems* (in the Explorer window) and select *Mount/Local Directory* (refer to Figure 18.7 to see these steps). Select the application directory (C:\Sun\webapps) and click *Finish* (refer to Figure 18.8)—this adds the directory to the Explorer. Now you can open the directory to see the WAR file, wa. Double-click on the WAR file and the *WAR Contents* window pops up showing the contents of the WAR file (see Figure 18.23).

Figure 18.23. Viewing contents of WAR file.



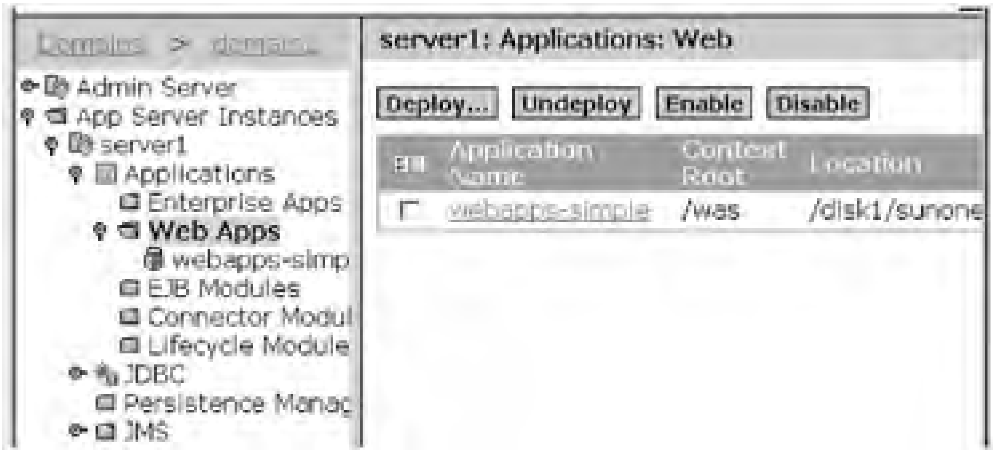
```
<html>
<head>
```

Path Name	File Name	Ext
	index	html
	jsp1	jsp
META-INF	MANIFEST	MF
WEB-INF	web	xml
WEB-INF\classes	abc	class
graphics	ShawndWorld	jpg

Step 3. Deploying the Application WAR File

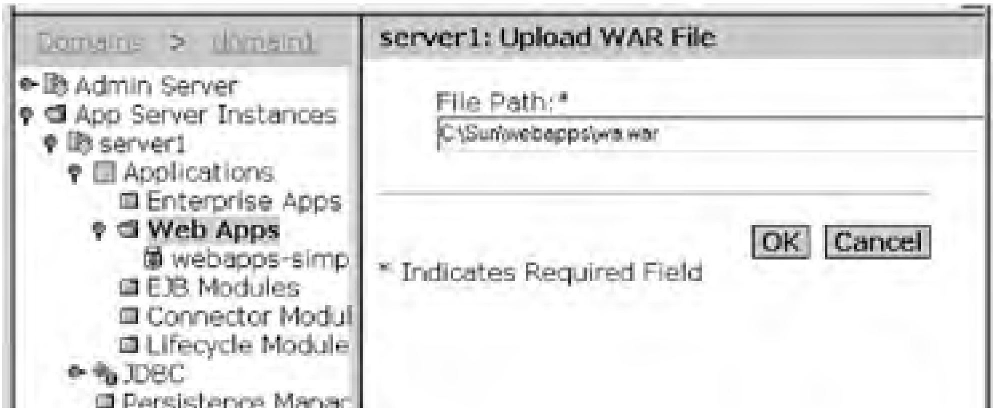
The web application has been packaged and is ready for deployment. Start the Sun ONE Application Server Administration Server if it is not already running and log in. Under *server1* and *Applications*, click on *Web Apps* (on left side of Figure 18.24). This brings up the web application mainenance web page on the right side of Figure 18.24. Click the *Deploy* button. Put the full-path filename (C:\Sun\webapps\wa.war) in the *File Path* field (Figure 18.25). Then click *OK*.

Figure 18.24. Web application maintenance web page.



	Application Name	Context Root	Location
☐	webapps-simple	/was	/disk1/sunone

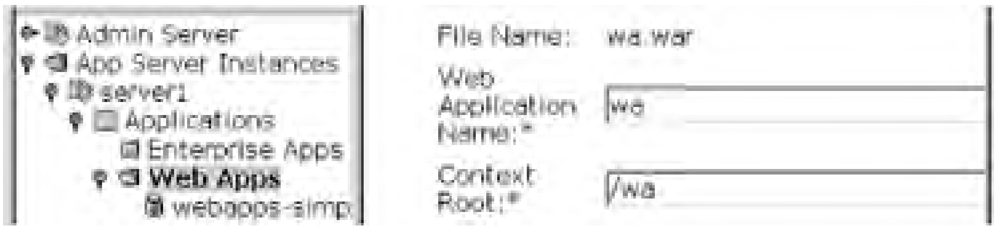
Figure 18.25. WAR file full-path file name.



File Path:* C:\Sun\webapps\wa.war

Figure 18.26 is the WAR file deployment web page. The only thing to add here is the *Context Root* value of */wa*. The context root is the root document directory for the web application wa. All the resources in wa are accessed under the context root. For example, the web application contains an index.html file. To access this file, use URL *http://sun.red.iplanet.com/wa/index.html*. Then click *OK*. Now the new web application is added to the list of web applications.

Figure 18.26. WAR file deployment web page.

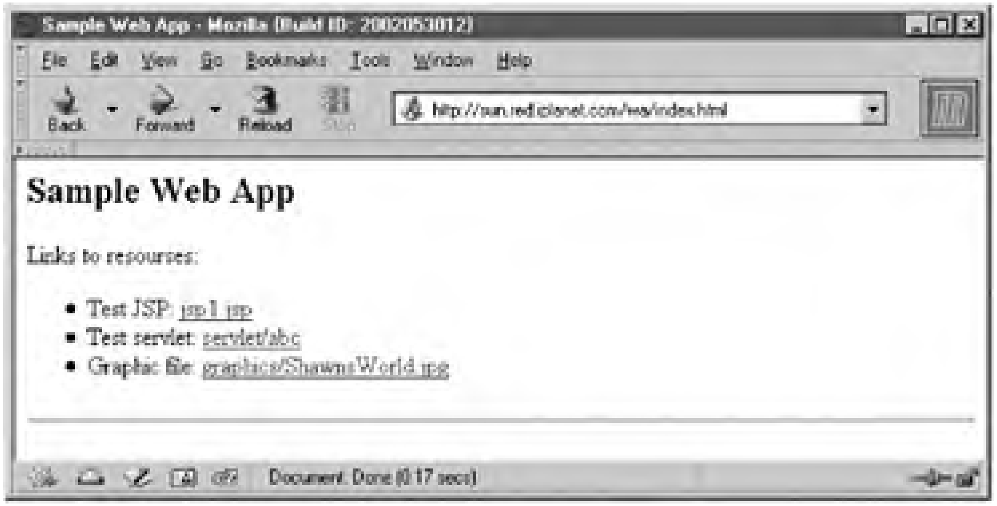


File Name: wa.war
Web Application Name:* wa
Context Root:* /wa

Step 4. Testing the Web Application

The web application is now ready for testing. Go to the URL *http://sun.red.iplanet.com/wa/index.html* (Figure 18.27).

Figure 18.27. Web application home page.


Sample Web App - Mozilla (Build ID: 2002053012)
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://sun.red.iplanet.com/wa/index.html
Document: Done (0.17 secs)

Sample Web App

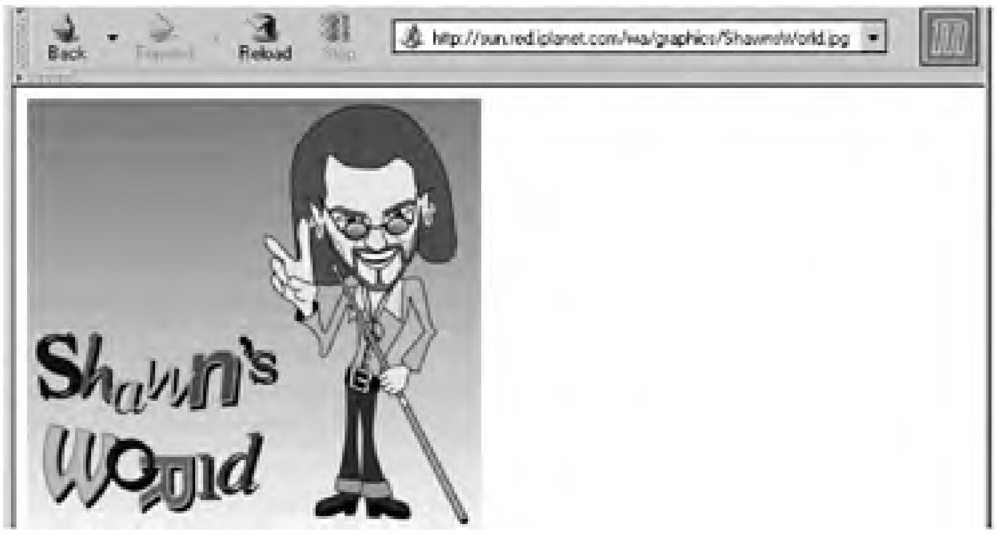
Links to resources:

- Test JSP: jsp1.jsp
- Test servlet: servlet/abc
- Graphic file: graphics/ShawnsWorld.jpg

In Figure 18.27:

- When you click *jsp1.jsp*, the Sun ONE Application Server converts the JSP into a servlet and compiles and runs the servlet.
- When you click *servlet/abc*, the servlet runs and the results are displayed in the browser.
- When you click *graphics/ShawnsWorld.jpg*, the *ShawnsWorld.jpg* graphic is displayed (Figure 18.28).

Figure 18.28. Graphics picture deployed in the web application WAR file.

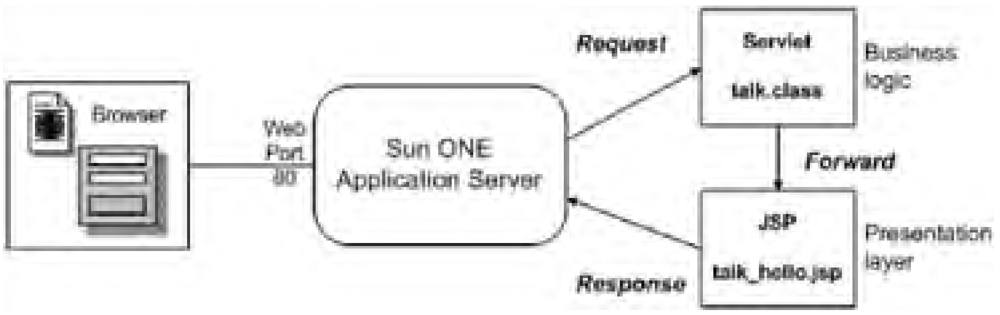


There—you've done it. You have created, packaged, deployed, and tested a web application. Have some fun with it!

Exercise: Expanding the Web Application

It is common in programming to use servlets to manage business logic and database interactions, then to forward the process control to a JSP to handle the presentation of the data. This method is illustrated in Figure 18.29.

Figure 18.29. Servlet request, JSP response.



In Figure 18.29, there are two new components that need to be added to the web application from the previous exercise: a servlet *talk.java* (*talk.class*) and a JSP *talk_hello.jsp*. To make the application more interesting, an HTML form is used to pass a parameter to the servlet *talk.class*.

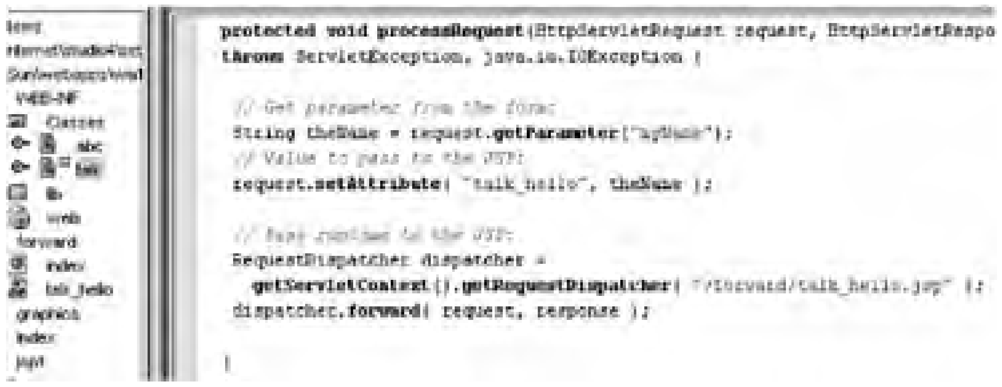
If you do not have Sun ONE Studio 4 running, start it. (Note: Some of the instructions in this second part of the chapter are the same as what we did in the first part. Therefore, I will not include a second set of the figures involved. If you need to see the screens, please refer to the earlier figures, primarily Figure 18.9 and Figure 18.14).

Add Another Servlet to the Application

To add the servlet

1. Open the web app branch and right click on *Classes* under *WEB-INF*, which is under the application file directory C:\Sun\webapps\wa1. Select *New/JSP & Servlet/Servlet*.
2. In the *New Wizard—Servlet* window that pops up, enter *talk* as the servlet name and click *Next*.
3. In the next window (refer back to Figure 18.15), for the *URL Pattern*, enter */talk*. The URL to access this servlet after the web application has been deployed is *http://sun.red.iplanet.com/wa/talk*.
4. Click the *Finish* button. A generic servlet is created, and the source code is displayed. Modify the source as shown in Figure 18.30. Change only the *processRequest* method; leave everything else as is.

Figure 18.30. The Java program for the servlet talk.



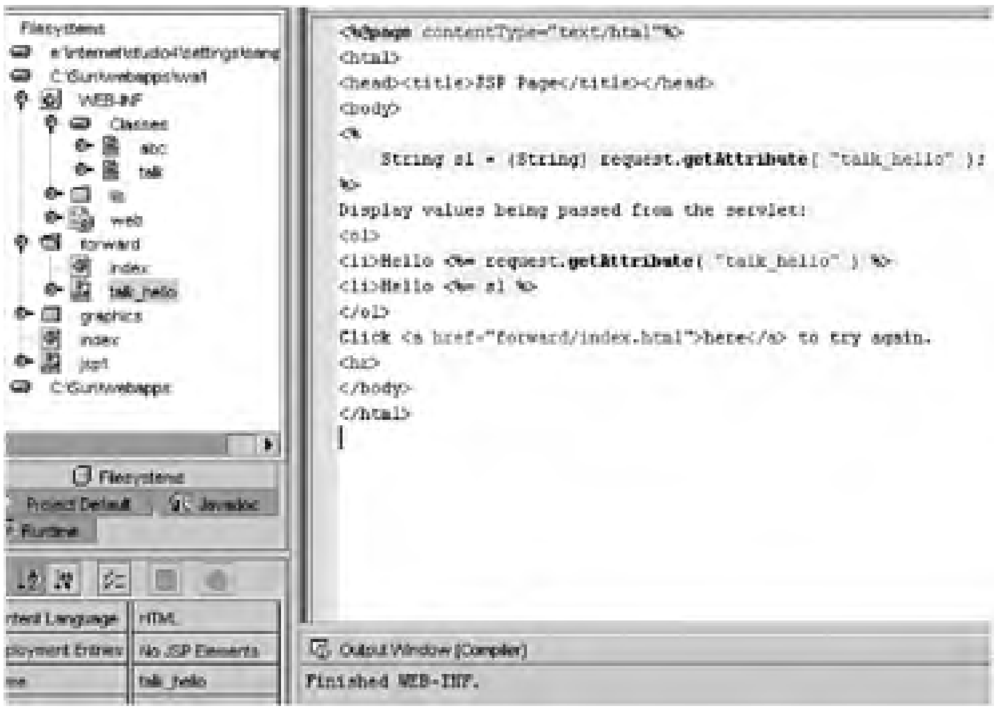
```
protected void processRequest(HttpServletRequest request, HttpServletRespo
throws ServletException, java.io.IOException {

 // Get parameter from the form:
 String theName = request.getParameter("myName");
 // Value to pass to the JSP:
 request.setAttribute( "talk_hello", theName );

 // Pass control to the JSP:
 RequestDispatcher dispatcher =
   getServletContext().getRequestDispatcher( "/forward/talk_hello.jsp" );
 dispatcher.forward( request, response );

}
```

Add Another JSP into the Application

First create a folder to hold the JSP. Right click on the application file directory C:\Sun\webapps\wa1, and select *New/Folder*. When the *New Wizard—Folder* windows appears, enter *forward* in the object *Name* and then click *Finish*.

Add a JSP into the forward folder by right clicking on the new folder *forward* and selecting *New/JSP & Servlet/JSP.* When the *New Wizard—JSP* windows appears, enter *talk_hello* in the object *Name*, and click *Finish*. A default JSP is created. Change the JSP source code to make it look like the code in Figure 18.31.

Figure 18.31. JSP to present the values in HTML.



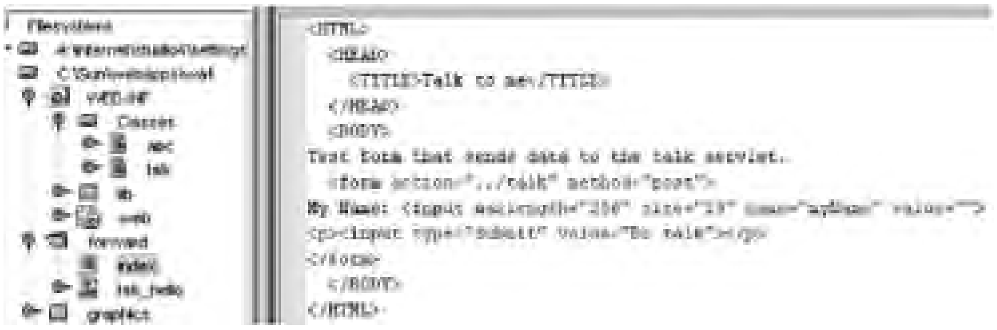
```
<%@page contentType="text/html"%>
<html>
<head><title>JSP Page</title></head>
<body>
<%
    String s1 = (String) request.getAttribute( "talk_hello" );
%>
Display values being passed from the servlet:
<ol>
<li>Hello <%= request.getAttribute( "talk_hello" ) %>
<li>Hello <%= s1 %>
</ol>
Click <a href="forward/index.html">here</a> to try again.
<hr>
</body>
</html>
```

```
Finished WEB-INF.
```

Add an HTML Web Form into the Application

To add the web page form, under the application file directory, right click on the *forward* folder and select *New/Other/HTML File*. In the *New Wizard—HTML File* prompt, enter *index* and click *Finish*. Then make the web page look like the one in Figure 18.32.

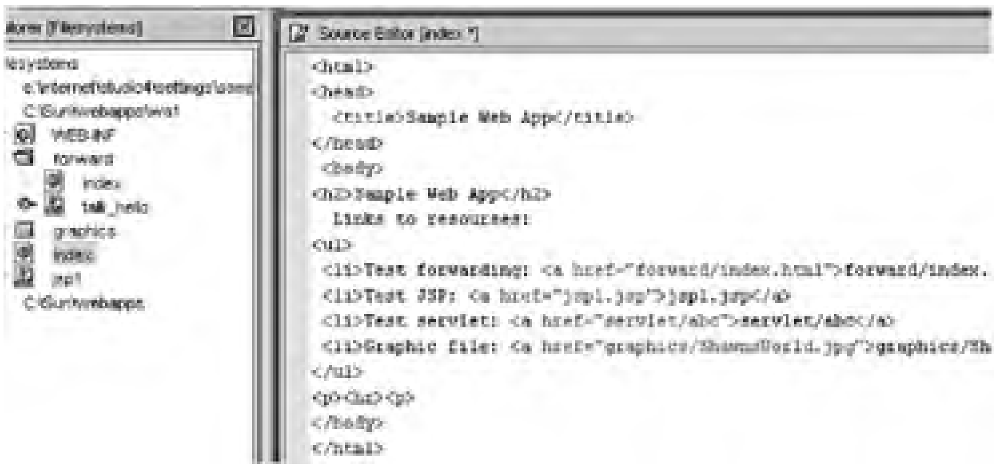
Figure 18.32. Forward web page form.



```
<HTML>
  <HEAD>
    <TITLE>Talk to me</TITLE>
  </HEAD>
  <BODY>
Test form that sends data to the talk servlet.
  <form action="../talk" method="post">
My Name: <input maxlength="256" size="15" name="myName" value="">
<p><input type="Submit" value="Do talk"></p>
</form>
  </BODY>
</HTML>
```

Update the main index.html file to call the forwarding index.html form (Figure 18.33). Save all the changes by clicking on the top menu option *File* and selecting menu item *Save All*. Then build the application (refer back to Figure 18.19 to see the screen) by clicking the top menu option *Build* and then clicking on *Build*. When the build is complete, the message *Finished* appears in the *Output Window* (similar to Figure 18.20). You are now ready for testing.

Figure 18.33. Update the file.

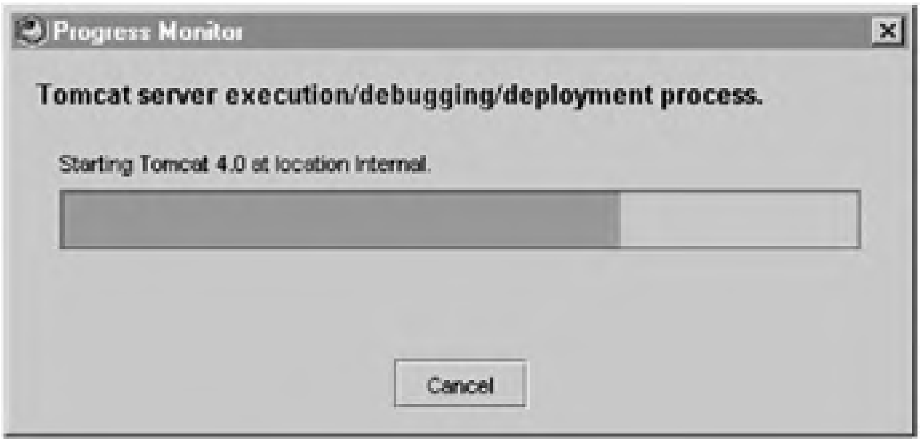


```
<html>
<head>
  <title>Sample Web App</title>
</head>
 <body>
<h2>Sample Web App</h2>
  Links to resources:
<ul>
 <li>Test forwarding: <a href="forward/index.html">forward/index.
 <li>Test JSP: <a href="jsp1.jsp">jsp1.jsp</a>
 <li>Test servlet: <a href="servlet/abc">servlet/abc</a>
 <li>Graphic file: <a href="graphics/ShamusWorld.jpg">graphics/Sh
</ul>
<p><hr><p>
</body>
</html>
```

Test the New Servlet and JSP

This time, I have added another test step. Testing is first done using the Apache Tomcat 4.0 server that comes with Sun ONE Studio 4. Test by clicking the top menu option *Build* and selecting menu item *Execute*. A progress bar appears while Tomcat is starting up (see Figure 18.34). When the progress bar completes, Apache Tomcat is running.

Figure 18.34. Progress bar.

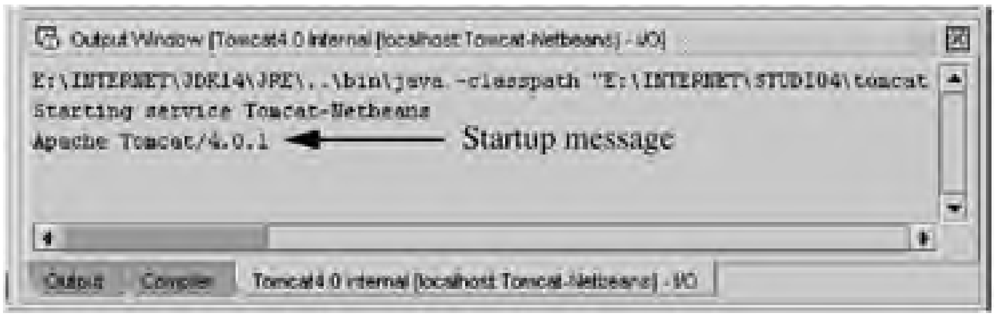


Tomcat server execution/debugging/deployment process.

Starting Tomcat 4.0 at location Internal.

Here's a useful feature: The *Output Window* (see Figure 18.35) shows the command used to run the Apache Tomcat server. If you copy the command out of the *Output Window* and put it into a batch file, you can run Tomcat at any time, independent of Sun ONE Studio. The root document directory is the directory of the web application being tested (in this case, C:\Sun\webapps\wa1).

Figure 18.35. Apache Tomcat has started.


Output Window [Tomcat4.0 internal [localhost:Tomcat-Netbeans] - I/O]
Startup message
Output
Compiler
Tomcat4.0 internal [localhost:Tomcat-Netbeans] - I/O

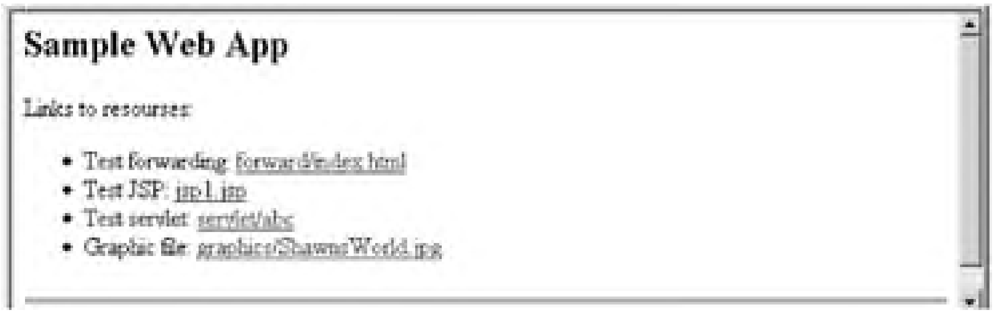
```
E:\INTERNET\JDK14\JRE\..\bin\java -classpath "E:\INTERNET\STUDIO4\tomcat
Starting service Tomcat-Netbeans
Apache Tomcat/4.0.1
```

Here is the complete command to run Tomcat from my system.

```
E:\INTERNET\JDK14\JRE\..\bin\java -classpath
  "E:\INTERNET\STUDIO4\tomcat401\bin\bootstrap.jar";
  "E:\INTERNET\JDK14\JRE\..\lib\tools.jar"
  -Dcatalina.home="E:\INTERNET\STUDIO4\tomcat401"
  -Dcatalina.base="e:\internet\studio4\set-
  tings\tomcat401_base"
  org.apache.catalina.startup.Bootstrap "start"
```

Once Tomcat is started, the default browser appears. You may now test the JSP by using the URL *http://localhost:8081/index.html* to go to the web application home page (in Figure 18.36).

Figure 18.36. Web application home page.



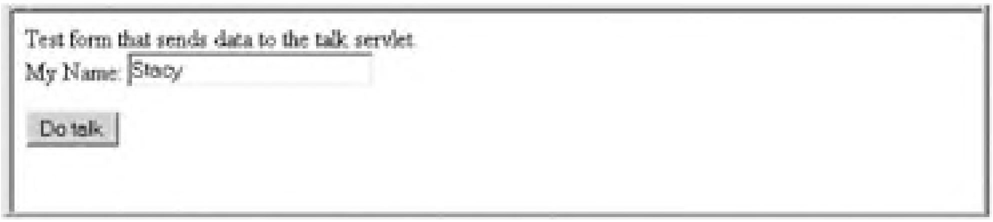
Sample Web App

Links to resourses:

- Test forwarding: forward/index.html
- Test JSP: jsp1.jsp
- Test servlet: servlet/abc
- Graphic file: graphics/ShawnsWorld.jpg

Click *forward/index.html* to call the web page form (Figure 18.37). On the web page form, enter your name and click the *Do talk* button. In my example, the parameter value of *Stacy* is passed to the servlet *talk*, which copies the parameter into a value to be passed to the *talk_hello* JSP. The servlet then forwards processing to the *talk_hello* JSP.

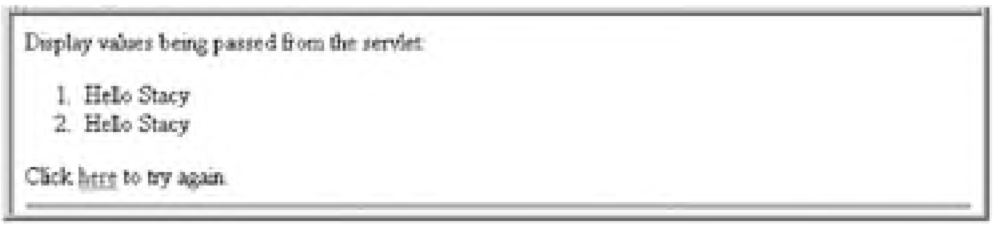
Figure 18.37. Sending data to servlet.



Test form that sends data to the talk servlet.
My Name: Stacy

The *talk_hello* JSP reads to the parameter value from the *talk* server and adds the value into the web page that is passed back to the browser (Figure 18.38).

Figure 18.38. Display values returning from servlet.



Display values being passed from the servlet:

1. Hello Stacy
2. Hello Stacy

Click here to try again.

Deploy to the Sun ONE Application Server

The revised web application has been tested successfully and is ready for deployment to an application server.

Since I have already described WAR file deployment two times in this chapter (the sample web application early in the chapter, and again in Step 4 where we deployed the application WAR file), I leave it up to you to redeploy your own application file.

Once the file is redeployed and the changes are applied to application server, to access this file, go to URL *http://sun.red.iplanet.com/wa/index.html*. Because of the simplistic nature of this web application, it functions on the Sun ONE Application Server in the same way it did when run under the Apache Tomcat server.

Summary

Sun ONE Studio 4 is a comprehensive tool for developing applications. In this chapter, you developed and packaged a web application using Sun ONE Studio 4. Then we deployed the application and ran it on the Sun ONE Application Server. The steps we used to do this were laid out in Figure 18.1:

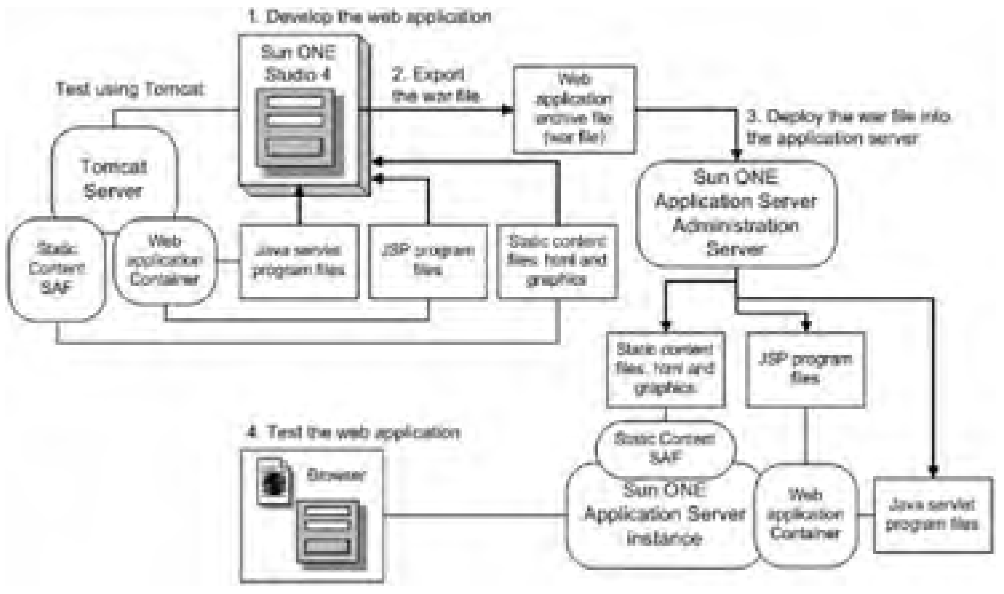
1. Create a web application that has an HTML web page, graphics, a JSP, and a servlet (step1).
2. Package (export) this application into a WAR file (step 2).
3. Deploy the application WAR file using the Sun ONE Application Server Administration Server system and apply the changes to the application server (step 3).
4. Test the web application (step 4).

In completing those steps, you performed the following functions (including the four steps):

1. Mount a file directory that is used for the project to create the application.
2. Before adding servlets, add a web module structure into the file directory. Then add servlets.
3. Add web site component files into the application directory. Types of application files that may be included with a web application are HTML web pages, graphics, sound files, and text files.
4. Save all files that have been edited.
5. Build the application. This step compiles any program files that need compiling (e.g., servlet programs are compiled into class files).
6. Package the application files into a WAR file.
7. Use a browser to log in to the Sun ONE Application Server Administration Server web site, and go to the deployment web page.
8. Select the WAR file and deploy it.
9. Apply the application configuration changes to the application server.
10. Test!

Figure 18.39 illustrates that Sun ONE Studio is the center of the application development cycle—it organizes all the files in an application. Since the Tomcat system is installed with Studio, it makes testing a straightforward process.

Figure 18.39. Web application development using Sun ONE Studio.


1. Develop the web application
Sun ONE Studio 4
Test using Tomcat
2. Export the war file
Web application archive file (war file)
3. Deploy the war file into the application server
Tomcat Server
Static Content SAF
Web application Container
Java servlet program files
JSP program files
Static content files: html and graphics
Sun ONE Application Server Administration Server
Static content files: html and graphics
JSP program files
4. Test the web application
Browser
Static Content SAF
Sun ONE Application Server instance
Web application Container
Java servlet program files

When an application is ready to be deployed to an application server, Studio packages all the web application files into a single WAR file. This is an excellent distribution system: develop and package the files into a single standard formatted file. This WAR file can then be distributed and installed easily. This installation uses standard tools to deploy files into the application server environment. These files can be deployed to any application server because a WAR file is standard format used by all application servers regardless of the vendor.

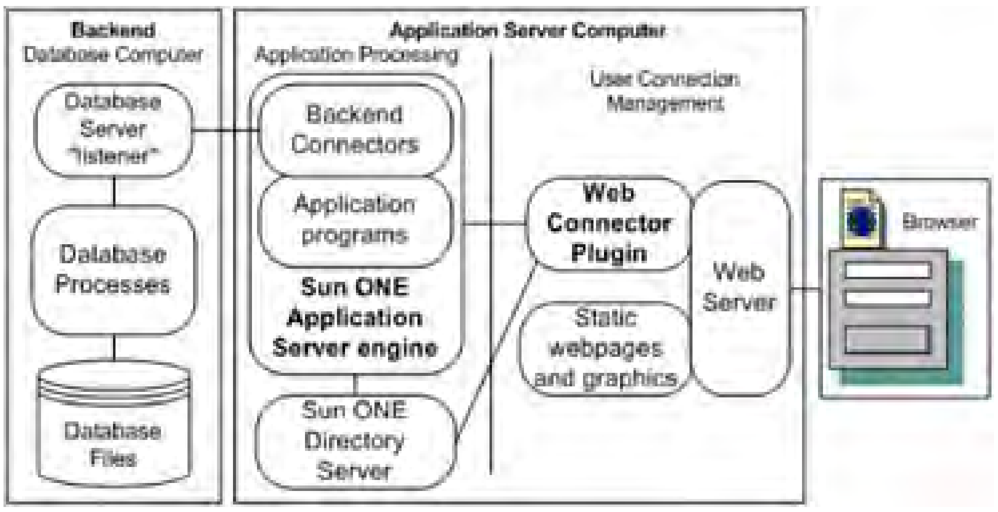
In our exercise, deployment was managed through the Sun ONE Application Server Administration Server web application. From the administration server, deployment is simply a matter of selecting the WAR file, specifying a context root for the web application, and clicking the Deploy button. The deployment program unpacks the WAR file and puts the application files into their appropriate file directory in the application server's environment. That's all there is to it; then it's ready to test. Using a browser, test the web application.

After a developer and I completed this exercise, he was ready to try more and more things with Sun ONE Studio 4. He is confident enough with the product to continue developing his own applications.

Chapter 19. The Sun ONE Application Server 6.5

Web browsers connect surfers to the Internet and to company intranets. No matter where you surf, you find web-enabled applications, i.e., applications that are accessible using a web browser. The Sun ONE Application Server 6.5 is a server that provides a system to develop, build, manage, and run web site applications. On the back end, it connects and manages connections to database systems. Figure 19.1 shows the main components of a Sun ONE Application Server 6.5 installation.

Figure 19.1. Application server development system or a small production system.



Sun ONE Application Server 6.5 Components

This application server has three main components that make it function: a piece that plugs into the Sun ONE Web Server, an application server engine, and the Sun ONE Directory Server. The web server manages connections to people using browsers, and the application server plugin (plugged into the web server) communicates with the application server engine. The engine runs the application programs and manages connections to databases. The directory server stores information related to the applications and the application server itself. Figure 19.1 gives you an overview of the components. Now let's look at sample setups.

Sun ONE Application Server and the Web Server

Browsers connect to applications, and it is the Sun ONE Web Server that manages these connections. The web server communicates with the Sun ONE Application Server through an SAF called the web connector. The web connector manages the connection between the web server and the application server engine.

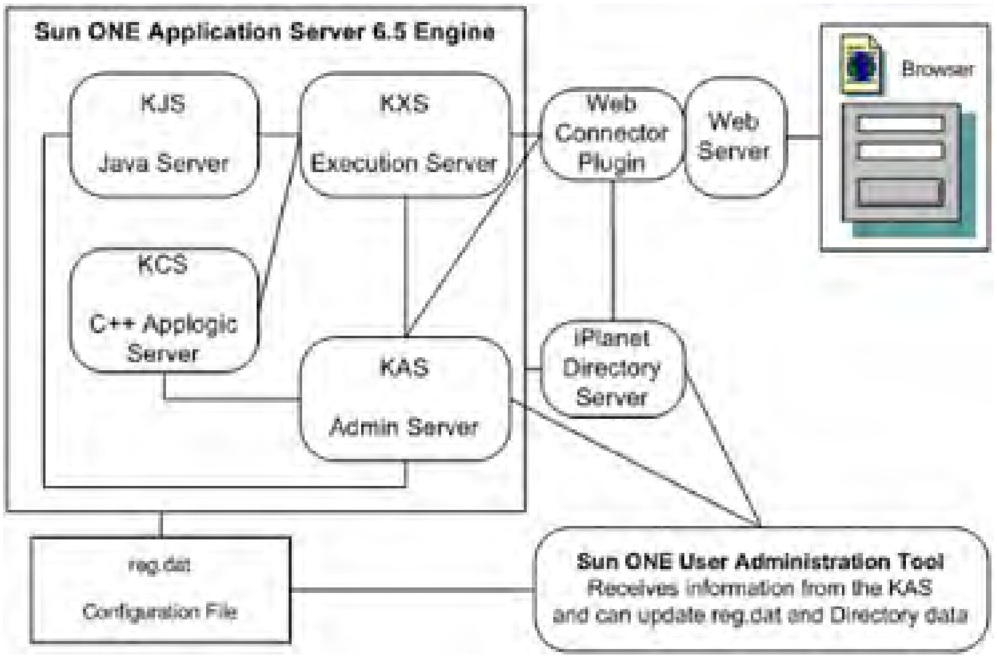
When the web server receives an application server request from a browser, the request is given to the web connector. The web connector will check with the Sun ONE Directory Server to find out which Sun ONE Application Server the request is to be sent to—that is, to find out which application server can fulfill the request. In a single application server configuration, this is obvious: The request is sent to the only Sun ONE Application Server engine in the configuration.

Now that the web connector knows which application server will handle the request, it passes the request to that application server. The web connector then waits for a response from the application server. Based on the request, the application server will run the appropriate program, which returns a response to the web connector, which in turn passes the response back to the web browser. The browser then displays the web page (from the application) to the user.

The Sun ONE Application Server Engine

The engine is made up of a number of processes working together, each with its own function and features. These include four servers (see Figure 19.2): KAS, KXS, KJS, and KCS. A historical note: The *K* in these server names stands for Kiva—the company that developed the original application server. Netscape purchased Kiva, bringing the Kiva product line into the Netscape family of products. Then, through a series of upgrades, these products transformed into the iPlanet Application Server, which is now the Sun ONE Application Server 6.5 product.

Figure 19.2. Components of the application server engine.



- **KAS—** Provides system services for application server administration and failure recovery. This administration server receives signals and statistics from the other servers that are part of the Sun ONE Application Server. If one of the other processes crashes, the administration server restarts the process.

- **KXS—** This execution server is responsible for managing and hosting system-level services such as load balancing and delegating requests to the language-specific application processes of the KJS and the KCS. The execution service receives requests from the web connector. If the request is to be run by the KJS process, then the request is given to a KJS process. If the request is to be run by the KCS process, then it is given to a KCS process.

- **KJS—** This is a Java server for processing Java program requests.

- **KCS—** These C++ application logic programs can run on the application server. If C++ processing is not needed, this server process should be shut down.

Sun ONE Application Server Java Server (KJS)

The KJS process runs Java servlets, JSPs, EJBs, and the other types of Java technology beans.

Through the KJS process, there are connection tools to connect to back-end systems and other applications using JDBC, connection pooling, transaction-based systems, and now JMS.

All of this makes the Sun ONE Application Server compliant with J2EE technology. The J2EE specification says the application server runs a specific version of each type of component; for example, the Java servlet version 1.2.

iPlanet Application Server Administration Tool

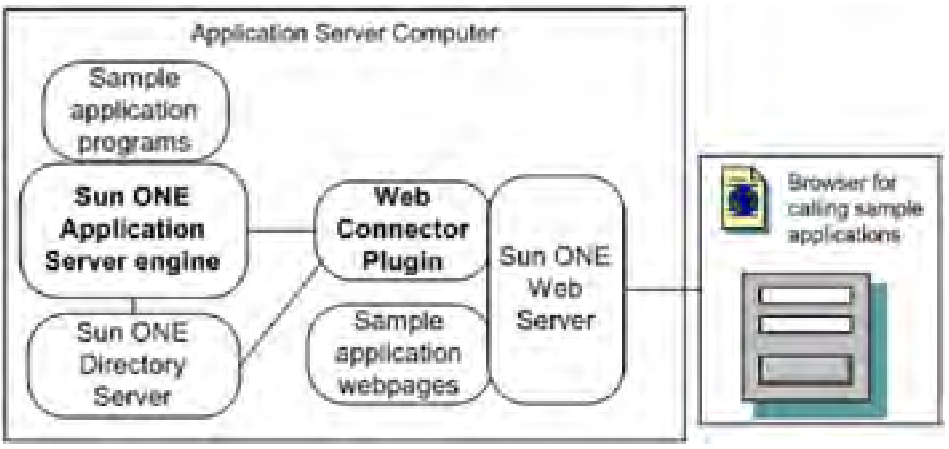
The Sun ONE Application Server Administration Tool is another included component. It is used to stop and start processes, as well as to configure and tune the server engine processes. Through this tool, database connections are configured and monitored. The tool is used to monitor what is happening on the servers on a regular basis; for example, the number of hits/requests, size of request queues, and so on.

It is possible to provide a centralized administration system by using the Sun ONE Application Server Administration Tool to administer a number of application servers. Setting this up is a matter of starting the tool and registering each of the application servers that you want to monitor. After registration, set up the monitoring system and monitor each server.

Setting Up a Sun ONE Application Server

Figure 19.3 illustrates a single computer setup. Two major steps are involved in setting this up: Install the Sun ONE Web Server; then install the Sun ONE Application Server. When the installation is complete, start the servers and test.

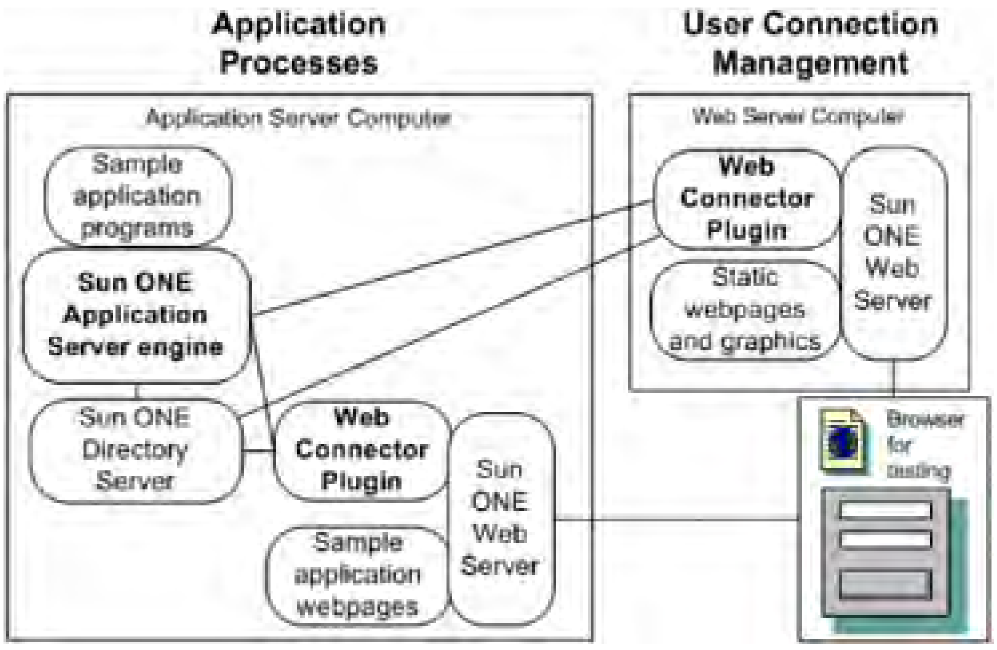
Figure 19.3. Default Installation with a Web Connector.


Application Server Computer
Sample application programs
Sun ONE Application Server engine
Sun ONE Directory Server
Web Connector Plugin
Sample application webpages
Sun ONE Web Server
Browser for calling sample applications

During the installation of the Sun ONE Application Server 6.5, you have the option of installing a web connector. This is because the application server does not require a web server on the same computer. However, I always install both a web server and a web connector on the same computer as the application server so that I can test the application server. By default, when a web connector is installed, the installation sets up sample programs. When you complete the installation, start both the application server and the web server and then call one of the sample programs—this is to test that the application server is working. After the application server is tested, you have the option of setting up a web server on another computer and front-ending this application server and shutting down the web server that was used for testing.

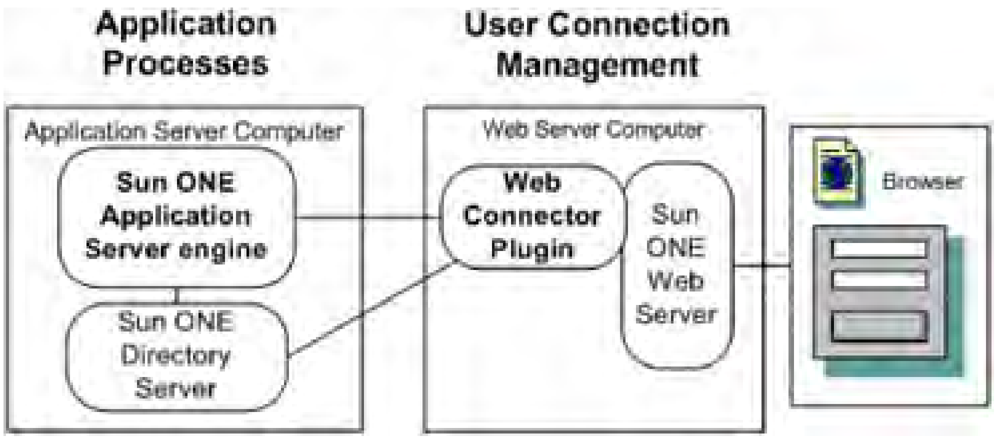
Figure 19.4 illustrates a web server that has been set up on another computer. The computer also has a Sun ONE Application Server web connector plugin installed into the web server's web instance. When this web connector was installed, it was given the information to connect to the application server engine. The web connector also has the information to connect to the iPlanet Directory Server, which contains the application server configuration information.

Figure 19.4. Two web servers communicating with the same application server.


Application Processes
User Connection Management
Application Server Computer
Sample application programs
Sun ONE Application Server engine
Sun ONE Directory Server
Web Connector Plugin
Sample application webpages
Sun ONE Web Server
Web Server Computer
Web Connector Plugin
Static webpages and graphics
Sun ONE Web Server
Browser for testing

When the new web server is working with the application server, the test web server can be shut down. If this is a production setup, remove the sample applications as well. This gives us the production-ready setup illustrated in Figure 19.5.

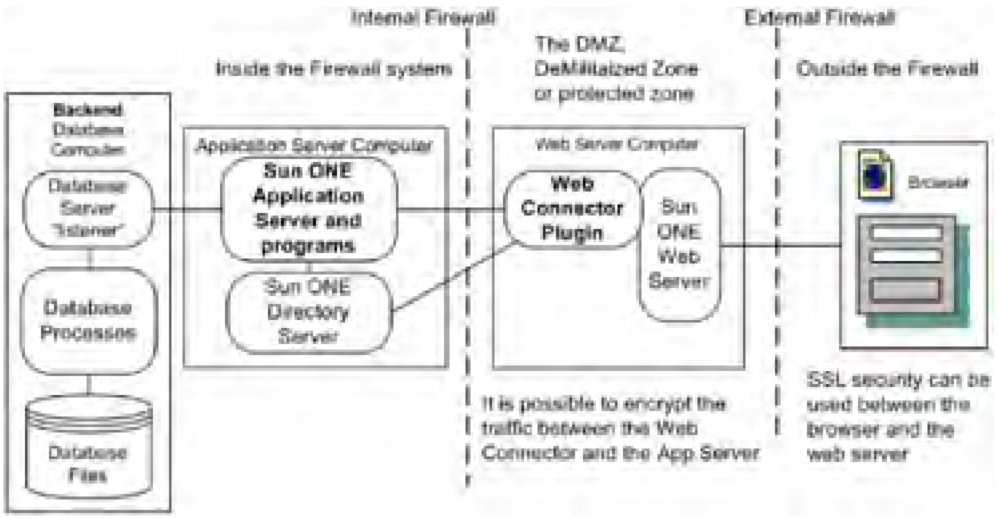
Figure 19.5. Production-ready configuration.



Secure Network Setup

To protect your computers, applications, and databases, it is advisable (and a common practice) to have two firewalls for protection. Figure 19.6 exemplifies a secure setup.

Figure 19.6. A well-protected application server and database.

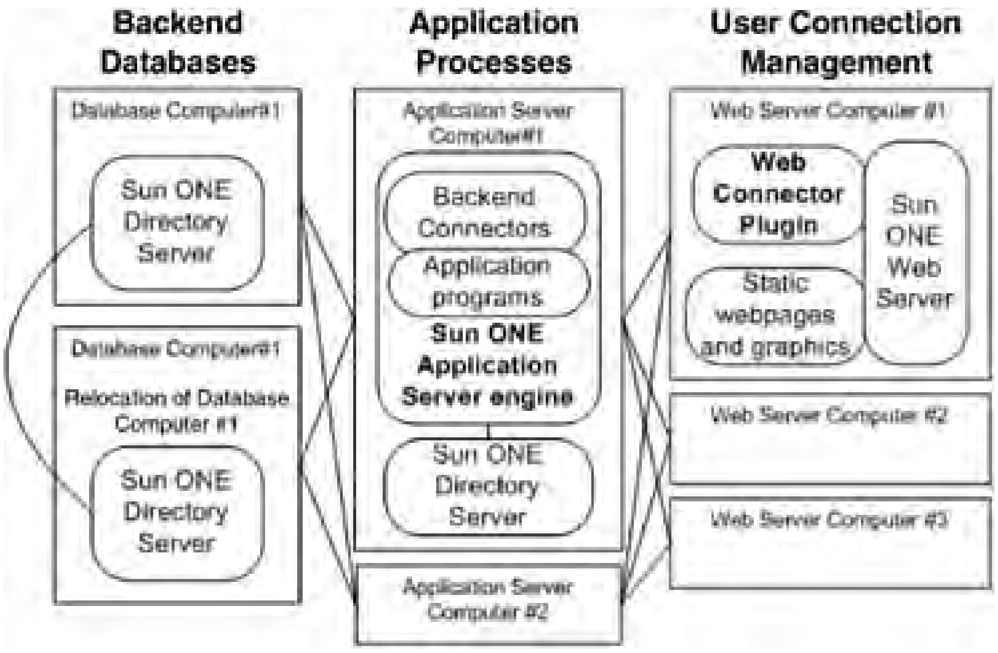

Internal Firewall
External Firewall
Inside the Firewall system
The DMZ, DeMilitaized Zone or protected zone
Outside the Firewall
Backend Database Computer
Database Server "listener"
Database Processes
Database Files
Application Server Computer
Sun ONE Application Server and programs
Sun ONE Directory Server
Web Server Computer
Web Connector Plugin
Sun ONE Web Server
Browser
It is possible to encrypt the traffic between the Web Connector and the App Server
SSL security can be used between the browser and the web server

Multiple Application Server Setup

The Sun ONE Application Server 6.5 is a system for developing and running medium- to large-scale applications When setting up and maintaining an application server system, a number of people and groups must be involved. A group of systems people sets up a development and production environment. The development team writes Java technology programs in a development environment. Another group or person—the system administrator—maintains the system and deploys the programs into the production system.

In the medium-sized system shown in Figure 19.7, multiple machines are needed for load balancing, failover, and backup. There are multiple web servers front-ending multiple application servers using multiple databases.

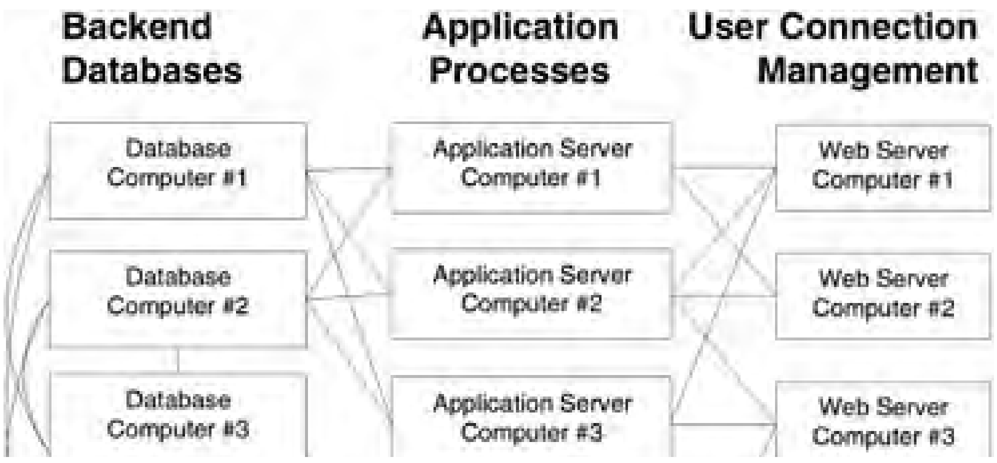
Figure 19.7. A medium-sized production system.

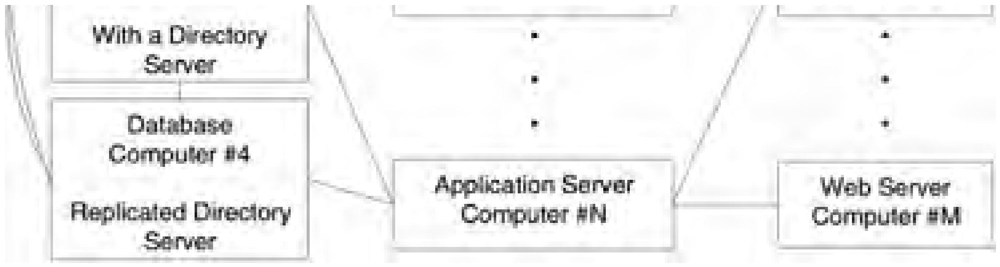


When designing an application, keep in mind the amount of computer processing unit (CPU) space you have. In this case, there are two application servers. If, while both application servers are running, neither machine's CPU use goes above 50 percent, then should one machine go down, the other machine can pick up the load and still stay below 100 percent, thereby avoiding a crash or loss of service.

While Figure 19.7 illustrated a medium-size setup, Figure 19.8 is a large-scale setup. There are multiple computers at every level, from the front to the back. In a large-scale system that is set up and designed well, there should not be any loss of service if any one of the machines is taken down for maintenance. The application should keep on running.

Figure 19.8. Large-scale production environment.





Programs, static web pages, and graphics are spread across a number of computers. The Sun ONE Application Server becomes a central processing engine for a large application production environment. When you are deciding where to locate programs and files, consider the size of each computer, as well as the number of database calls and processes. After a system is set up and running, you can fine-tune and monitor the system.

In a large Sun ONE Application Server environment, machines can multitask or they can be dedicated to particular functions. The Sun ONE Application Server can handle a multitude of complex configurations.

Summary

The Sun ONE Application Server has a web connector that runs on Sun ONE Web Server. The web connector communicates with the application server engine. Then the web connector and the engine both communicate with the Sun ONE Directory Server which stores application data and application server configuration information.

The design of the Sun ONE Application Server 6.5 allows for flexibility in setting it up. The application server and web server can both be on one computer or they can be on separate computers. The web server with a web connector can be put into the DMZ to make the setup secure. Multiple web servers can communicate with multiple application servers or vise versa. All this adds up to the ability to design a flexible, scalable application system.

At first the Sun ONE Application Server may seem overwhelming; however, once it is broken down into separate components, it is easier to understand, set up, and use.

Chapter 20. Installing Sun ONE Application Server 6.5, Test Drive Edition

The installation of the Test Drive Edition of Sun ONE Application Server 6.5 is very clear cut. Follow these steps:

1. Go to URL *http://wwws.sun.com/software/products/appsrvr/home_appsrvr.html* (to download the Sun ONE Application Server 6.5 Enterprise Edition).
2. Select *Trial Download* and download the software and put the file into a working directory. Also, copy the product license key for use during the installation.
3. Decompress the file. If you are using Windows, unzip the file into a working directory. If you are using UNIX, use these commands to decompress a tar.gz file:

   ```
   gzip -d *.gz
   tar xvf *.tar
   ```

When you install the Sun ONE Application Server, you get these major components:

1. Sun ONE Application Server
2. Sun ONE Application Server tools, command line, and GUI tools
3. Directory server engine—this is the server that maintains and serves the directory data
4. Documentation—once the software is decompressed (step 3 above), use a browser to view the included installation documentation; go to URL *file:/<decompress directory>/ias-testdrive/index.html*

Preinstallation Notes...

Here are sample computer configurations that I used for my installation:

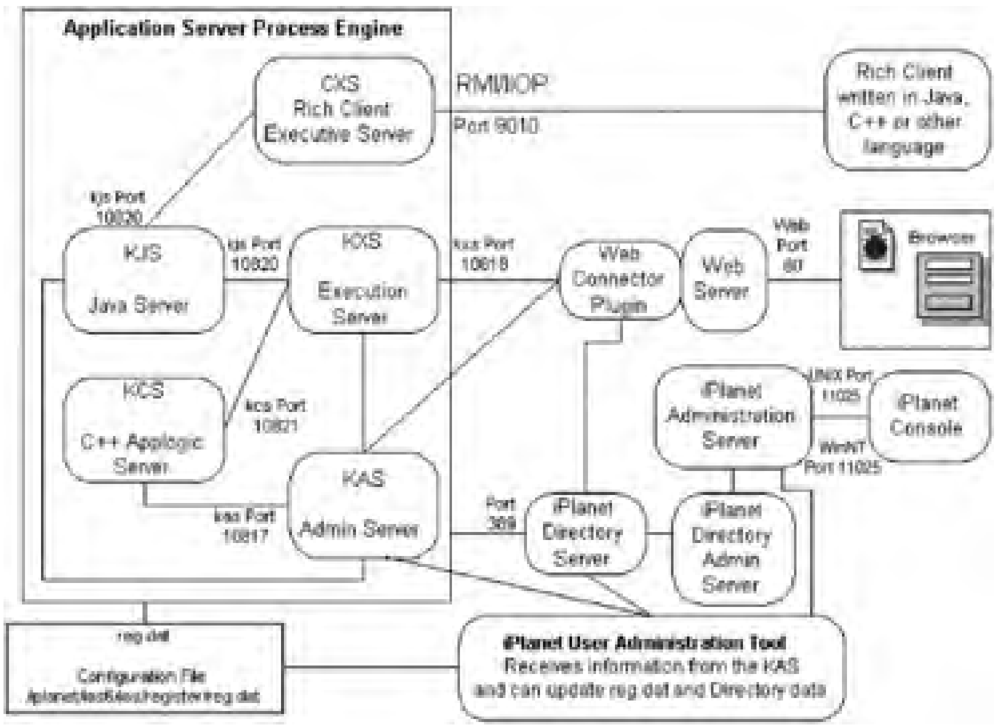
- SUN Ultra 10 computer running Solaris 8 OS, 512 megabytes (MB) memory, 1/2 gigabytes (gig) of disk space. Another computer I used is a Sun Ultra 1 computer running Solaris 8 OS, 192 MB memory, 1/2 gig of disk space.
- WinNT 4.0 SP5 computer with 256 MB memory, 250 MB disk space.
- Windows 2000, 256 MB memory, 1 gig of disk space.

Sample supporting software includes

- Netscape Browser 4.75 (or higher)
- Sun ONE Web Server 6.0 already installed, set up, and tested. During the application server installation, the application server web connector plugin is installed to run under the web server instance.

Application server components that are installed (Figure 20.1):

Figure 20.1. Sun ONE Application Server 6.5 components.


Application Server Process Engine
CXS Rich Client Executive Server
RMI/IIOP
Port 9010
Rich Client written in Java, C++ or other language
kjs Port 10830
KJS Java Server
kjs Port 10820
KXS Execution Server
kxs Port 10818
Web Connector Plugin
Web Server
Web Port 80
Browser
KCS C++ Applogic Server
kcs Port 10821
iPlanet Administration Server
UNIX Port 11025
WinNT Port 11025
iPlanet Console
KAS Admin Server
kas Port 10817
Port 389
iPlanet Directory Server
iPlanet Directory Admin Server
reg.dat
Configuration File /iplanet/ias6/ias/registry/reg.dat
iPlanet User Administration Tool
Receives information from the KAS and can update reg.dat and Directory data

1. Web connector into an iPlanet Web Server instance
2. iPlanet Application Server engine—one KJS (Java server); one KCS (C++ server); one CXS (rich client executive server); KXS (execution server); one KAS (administration server)
3. iPlanet Application Server deployment tool
4. iPlanet Application Server Administration System
5. Sample applications
6. PointBase, a third-party database system
7. iPlanet Directory Server, which will be used to store the iPlanet Application Server configuration information and user data

8. iPlanet Console
9. Java Development Kit 1.3.1

UNIX Installation

Run the setup script ./setup. Now the first screen appears:

```
>>> iPlanet Application Server Test Drive ezSetup
  install <<<

  BY INSTALLING THIS SOFTWARE YOU ARE CONSENTING TO BE
  BOUND BY AND ARE BECOMING A PARTY TO THE AGREEMENT
  FOUND IN THE LICENSE.TXT FILE. IF YOU DO NOT AGREE TO
  ALL OF THE TERMS OF THIS AGREEMENT, PLEASE DO NOT
  INSTALL OR USE THIS SOFTWARE.
  Do you agree to the license terms [NO]? y
```

As long as you agree with the terms, enter y for yes and the next window will come up.

```
>>> iPlanet Application Server Test Drive ezSetup
  install <<<

  This program will extract the server files and install
  them into a directory you specify. That directory is
  called the server root in the product documentation
  and will contain the server programs, the Administra
  tion Server, and the server configuration files.

  To accept the default shown in brackets, press the
  Enter key.
  Install location [/usr/iplanet/ias6]:  /iplanet/ias6
```

Install into a directory of your choice, making sure it has enough disk space.

Now you need to tell the installation where your Sun ONE Web Server instance is located. If you have not installed the web server, do so before continuing. Refer to chapter 4 about installing the web server that is included on the Test Drive CD if you need instructions.

```
>>> iPlanet Application Server Test Drive ezSetup
  install <<<

  iPlanet Application Server supports iPlanet Web Server
  4.1 and 6.0.

  Enter the FULL PATH of the Web Server Instance to be
  used, for example: /usr/ns-home/https-my_web_server

  Enter the FULL PATH of the Web Server Instance to be
  used:  /iplanet/iws/https-art.red.iplanet.com
```

Enter your full path to your web server instance.

```
>>> iPlanet Application Server Test Drive ezSetup
  install <<<

  To get Product key visit http://wwws.sun.com/software

  Enter Product Key:  3622750138-1397245601
```

Enter the product license key that you copied from the Sundown load web site. When you hit Enter, the software will install.

```
Installing Netscape core components
Installing Server Core Components
Installing iPlanet Directory Suite
Installing Administration Services
Installing nsPerl
Installing PerLDAP
Installing iPlanet Application Server Suite

      Starting the iAS install ...
Installing iPlanet Application Server 6.5 Core Binaries
Extracting Netscape core components ...
Extracting Server Core Components ...
Extracting Core Java classes ...
Extracting Java Runtime Environment ...
Extracting iPlanet Directory Server ...
Extracting iPlanet Directory Server Console ...
Extracting iPlanet Administration Server ...
Extracting Administration Server Console ...
Extracting nsPerl 5.005_03 ...
Extracting PerLDAP 1.4.1 ...
Extracting iPlanet Application Server Web Connector Com-
  ponent ...
Extracting iPlanet Application Server Core Server Compo-
  nent ...
Extracting iPlanet Application Server Administration
  Tool ...
Extracting iPlanet Application Server Deployment Tool
  ...
Extracting Pointbase Database Server ...
Extracting iPlanet Application Server 6.5 Core Binaries
  ...

[slapd-sun]: starting up server ...
[slapd-sun]: [28/Jul/2002:10:05:28 -0700] - iPlanet-
  Directory/5.0 ServicePack 1 B2001.264.1425 starting up
[slapd-sun]: [28/Jul/2002:10:05:34 -0700] - slapd
  started.  Listening on all interfaces port 11024 for
  LDAP requests
Your new directory server has been started.
Created new Directory Server
Start Slapd  Starting Slapd server configuration.
Success Slapd Added Directory Server information to Con-
  figuration Server.
Configuring Administration Server...
Your parameters are now entered into the Administration
  Server
database, and the Administration Server will be started.

Changing ownership to admin user root...
Setting up Administration Server Instance...
Configuring Administration Tasks in Directory Server...
Configuring Global Parameters in Directory Server...
iPlanet-WebServer-Enterprise/4.1SP3 BB1-09/17/2000 01:08

startup: listening to http://1.1.1.7, port 11025 as root

Proceeding with installation of iPlanet Application
  Server (iAS)...

Reading userinput log file .....
Done.
Configuring web server plugin .....
Changing permissions ...
Done.
The iAS cgi plug-in and/or HTMLs for Sample Applogics
  installed ...
Done.
     Steps to complete for the iAS install,
     along with the percent completed for each step:

 1.  Register iAS Deployment Tool
 2.  Make Database Access Parameter registry entries
 3.  Register iAS, Java & C++ engines
 4.  Register iAS dynamically loadable modules
 5.  Update Application Server startup scripts
 6.  Register iAS DLM
 7.  Extract Java JDK 1.3.1 into
     /iplanet/ias6/ias/usr/java
 8.  Register iAS sample applications
 9.  Register Transaction Manager
10.  Register iAS in the Directory Server
11.  Configure iAS Administration Tool
12.  Application Server ACL registration

     ---------------------------------------------------
     0% 10%  20%  30%  40%  50% 60%  70% 80%  90%  100%
     | | | | | | | | | | |

1.-12.  **************************************************
Starting PointBase integration...
Creating database users...
Database Users created.
Loading estore database...
Loaded.
```

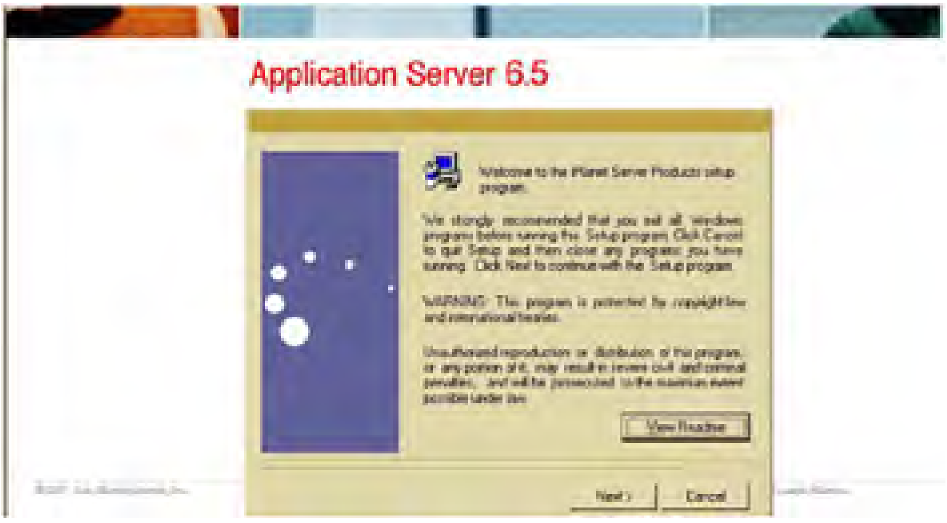
```
Loading data for j2eeguide schema...
Loading data for schema hellodb...
Loaded.
Loading data for schema sqlrunner...
Loaded.
Starting Pointbase Server...
PointBase Integration complete.
Start registering System/StaticServlet...

Start registering Bootstrap EJB...

Start registering iAS 6.5 Fortune Application...

Start Web Server
failure: server not running
iPlanet-WebServer-Enterprise/6.0SP3 B05/18/2002 02:30
[LS ls1] http://sun, port 80 ready to accept requests

Port Number Assignments
-----------------------
The following port numbers have been assigned during
  installation of iPlanet Application Server.  Please
  record these numbers so that you can specify them
  administering the application server environment.

iPlanet Directory Server Port: 11024
iPlanet Admin Server Port: 11025
iPlanet Application Server Admin Server Port: 10817

iPlanet Application Server Engine Ports:
Executive Server: 10818
Java Engine #1:10820,ORBPort:10921
C++ Engine #1:10821
CXS Engine #1: 10822, IIOP: 9010

Execute the following command to start the iPlanet Con-
  sole and begin managing your servers:
/iplanet/ias6/startconsole -a
  http://sun.red.iplanet.com:11025
iPlanet Application Server installation COMPLETED

Installation is complete.
See /iplanet/ias6/setup/setup.log for possible errors.
#
```

In chapter 21, you will administer and test what has been installed here.

Windows Installation (WinNT)

Run F:\TestDrive\NT\setup.exe. When the startup window appears (Figure 20.2), click *Next*.

Figure 20.2. Sun ONE Application Server 6.5 startup window.



Application Server 6.5

Welcome to the iPlanet Server Products setup program.

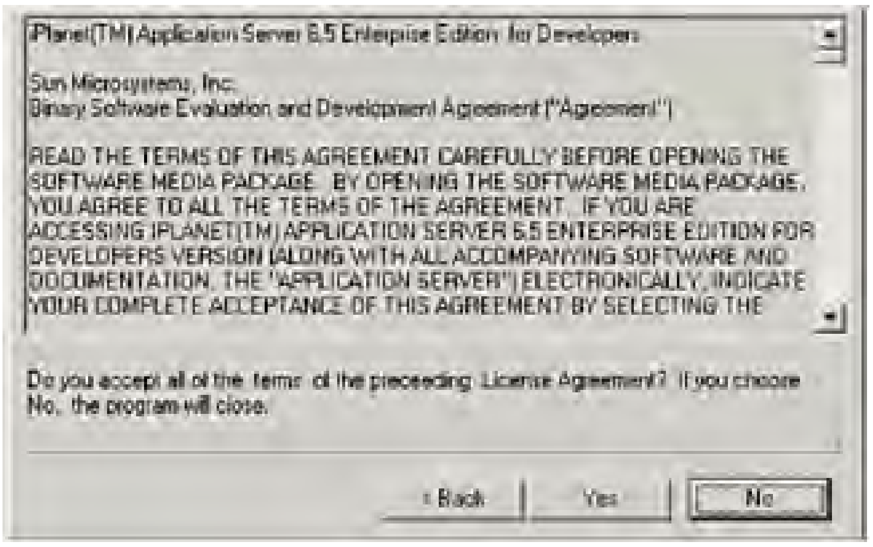
We strongly recommended that you exit all windows programs before running the Setup program. Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

The next window is the license agreement (see Figure 20.3). Read the license and click *Yes* (as long as you agree).

Figure 20.3. License agreement.



iPlanet(TM) Application Server 6.5 Enterprise Edition for Developers

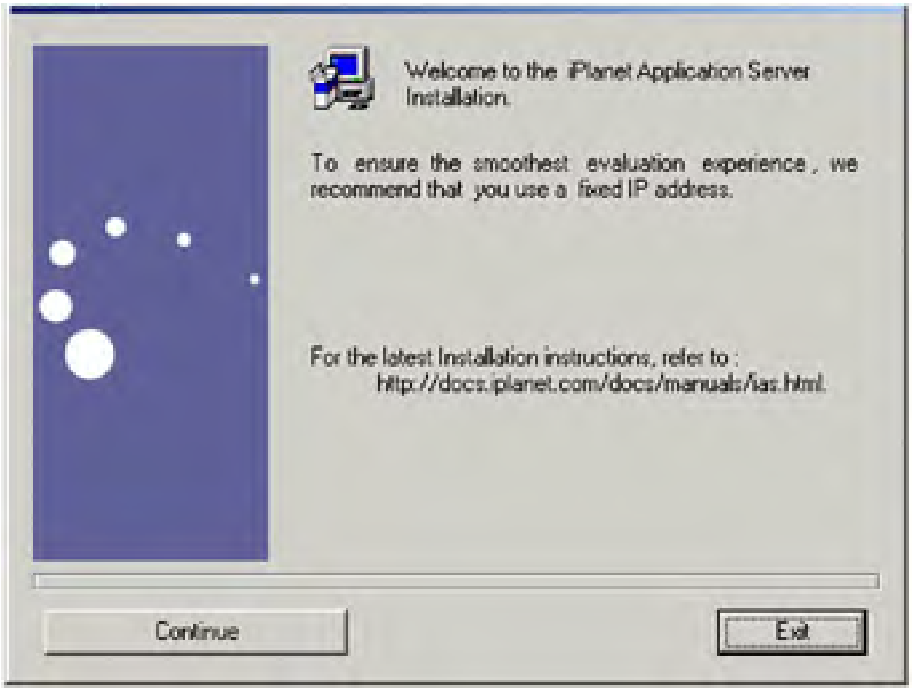
Sun Microsystems, Inc.
Binary Software Evaluation and Development Agreement ("Agreement")

READ THE TERMS OF THIS AGREEMENT CAREFULLY BEFORE OPENING THE SOFTWARE MEDIA PACKAGE. BY OPENING THE SOFTWARE MEDIA PACKAGE, YOU AGREE TO ALL THE TERMS OF THE AGREEMENT. IF YOU ARE ACCESSING IPLANET(TM) APPLICATION SERVER 6.5 ENTERPRISE EDITION FOR DEVELOPERS VERSION (ALONG WITH ALL ACCOMPANYING SOFTWARE AND DOCUMENTATION, THE "APPLICATION SERVER") ELECTRONICALLY, INDICATE YOUR COMPLETE ACCEPTANCE OF THIS AGREEMENT BY SELECTING THE

Do you accept all of the terms of the preceding License Agreement? If you choose No, the program will close.

The welcome window recommends that you use a *fixed IP address* (Figure 20.4). I have a *fixed IP address* and a valid domain name for my computer. Click *Continue*.

Figure 20.4. Welcome window.

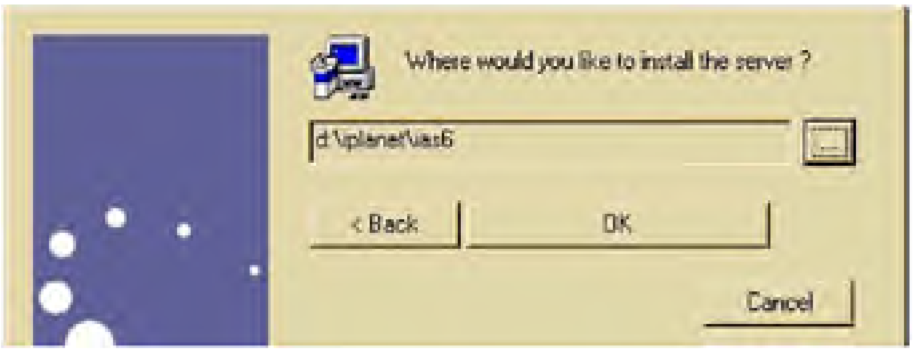

Welcome to the iPlanet Application Server Installation.

To ensure the smoothest evaluation experience, we recommend that you use a fixed IP address.

For the latest Installation instructions, refer to :
http://docs.iplanet.com/docs/manuals/ias.html

The next window asks where you would like to install the server (Figure 20.5). Enter your installation directory, in this case d:\iplanet\ias6. Click *OK*.

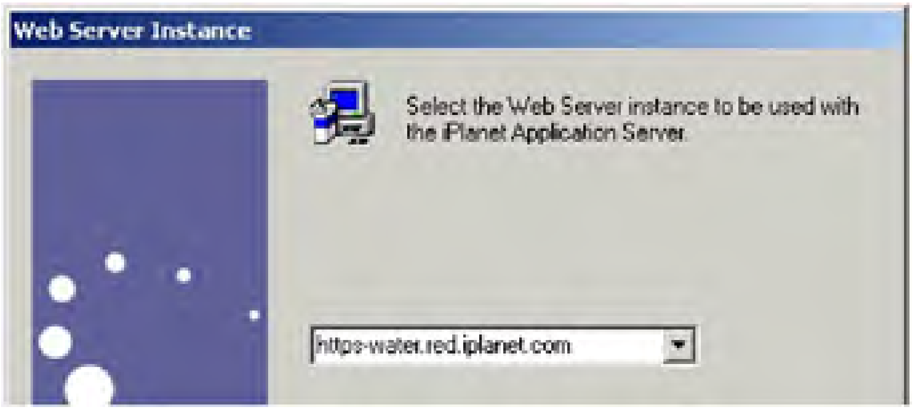
Figure 20.5. Enter installation directory.


Where would you like to install the server ?

d:\iplanet\ias6

Before you began installing this application server, you installed a Sun ONE Web Server. Now the web server instance window is displayed (Figure 20.6). Select the web server instance (in this case, *https-water.red.iplanet.com*). Click *OK*.

Figure 20.6. Select the web server instance.


Web Server Instance

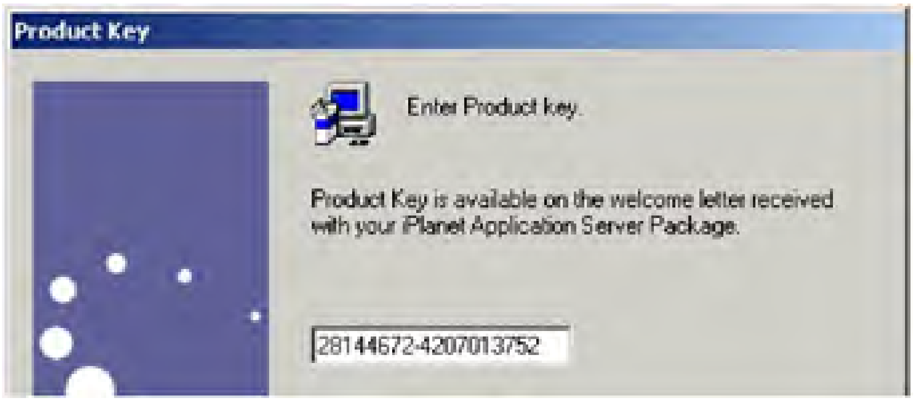
Select the Web Server instance to be used with the iPlanet Application Server.

https-water.red.iplanet.com

In the product key window that now opens (Figure 20.7), enter the product key that you already have (28144672-

4207013752) and click OK. If you don't have a product key, get one now at *www.iplanet.com/testdrive*. It takes only about three minutes to get a product key.

Figure 20.7. Enter product key.



Product Key

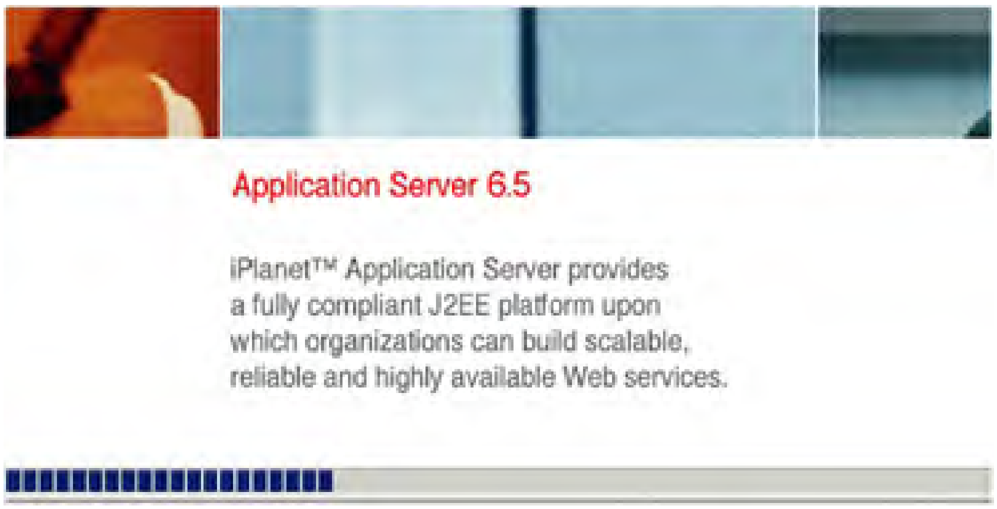
Enter Product key.

Product Key is available on the welcome letter received with your iPlanet Application Server Package.

28144672-4207013752

Now the software will start installing. There is a progress bar at the bottom of the screen (Figure 20.8).

Figure 20.8. Progress bar during installation.

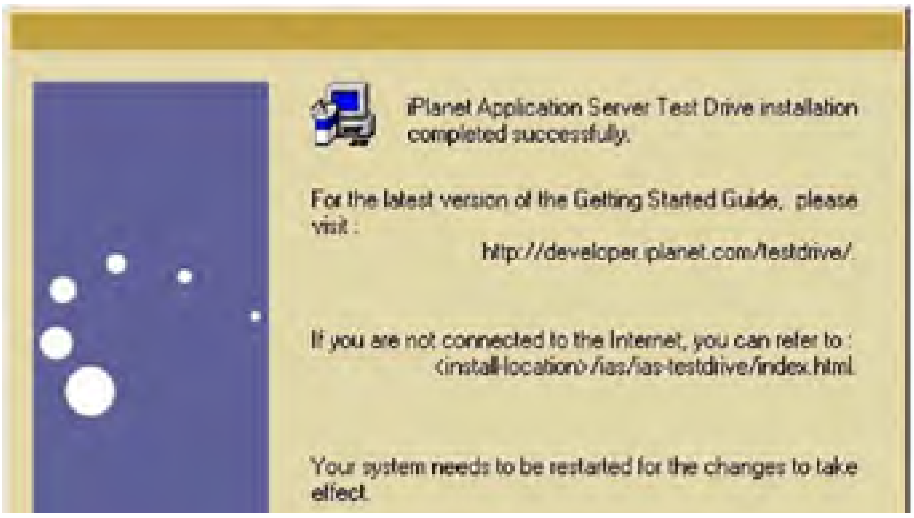


Application Server 6.5

iPlanet™ Application Server provides a fully compliant J2EE platform upon which organizations can build scalable, reliable and highly available Web services.

That's all there is to it. Do not reboot your machine at this time. Click the *Reboot—Later* button in Figure 20.9.

Figure 20.9. Installation completion window.

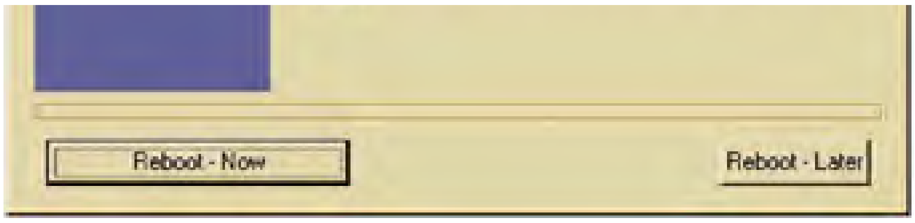


iPlanet Application Server Test Drive installation completed successfully.

For the latest version of the Getting Started Guide, please visit :
http://developer.iplanet.com/testdrive/.

If you are not connected to the Internet, you can refer to :
<install-location>/ias/ias-testdrive/index.html.

Your system needs to be restarted for the changes to take effect.



I repeat: Click *Reboot—Later*, do *not* reboot at this time.

Configure Processes to Start Manually

Before moving forward, set the *Startup Type* to *Manual*, so that the processes will not start up automatically when the computer is rebooted.

Click *Start / Settings / Control Panel*.

- Windows NT—click *Services*.
- Windows 2000—click *Administration Tools*; then click *Services*.

After double-clicking on each of the following services, set the *Startup Type* to *Manual*. The services are

- iPlanet Administration Server 5.0
- iPlanet Application Server 6.5
- iPlanet Directory Server 5 (<*hostname*>)
- iWS Administration Server (6.0)
- iWS (<*hostname*>)

When you have selected manual startup of processes, reboot your computer and move to the next chapter to start working with this Sun ONE Application Server 6.5.

Chapter 21. Testing and Administration

This is the list of components we installed in the previous chapters:

- Sun ONE Web Server
- Sun ONE Application Server
- Sun ONE Directory Server
- Sample programs and applications
- Documentation

This chapter covers the basics of getting started with the Sun ONE Application Server:

1. Starting and stopping the servers that have been installed
2. A simple test of the application server
3. Using/viewing log files
4. Using the Sun ONE Application Server Administration Tool

In the previous chapter, the last step in the installation process was to reboot your UNIX or Windows computer. If you are using a UNIX system, none of the servers listed above are running. If you are using a Windows computer and you have taken my advice in the previous chapter as to setting your Sun ONE Application Server processes to manual, then none of the servers listed above are running.

Since there are multiple independent software components involved in running the Sun ONE Application Server, we will focus next on starting and stopping all the software components.

Starting, Viewing, and Stopping Processes

Both the UNIX environment and the Windows environment have the same server process requirements. Since UNIX is more command-line oriented than Windows, I have elected to show the command-line commands in UNIX and the GUI tools in Windows.

Starting, Viewing, and Stopping Processes in UNIX

Starting Processes (UNIX)

There are dependencies between the software components, so I recommend that you start the components in appropriate order.

First, let's start the runtime components. Each component in this list is followed by its startup command line.

1. Sun ONE Directory Server instance

   ```
   # /iplanet/ias6/slapd-art/start-slapd
   ```

2. Sun ONE Application Server

   ```
   # /iplanet/ias6/ias/bin/iascontrol start
   ...
   iascontrol for iPlanet Application Server 6.5 Test
     Drive
   iAS started successfully on art.red.iplanet.com
   ```

3. Sun ONE Web Server instance

   ```
   # /iplanet/iws60/https-art.red.iplanet.com/start
   iPlanet-WebServer-Enterprise/6.0SP3 B05/18/2002
     02:30
   http://art.red.iplanet.com,port 80 ready for
     requests
   startup: server started successfully
   ```

The web server administration server needs to be running only when you need to administer the web server instances.

Starting the Sun ONE Web Server Administration Server:

```
# /iplanet/iws60/https-admserv/start
iPlanet-WebServer-Enterprise/6.0SP3 B05/18/2002
  02:30
warning: daemon is running as super-user
http://art.red.iplanet.com, port 4006 ready for
  requests
```

Now that all of the processes are running, we will look at the use of the `ps` command to list the processes.

Viewing Processes

List the processes:

```
# ps -ef | grep iplanet
```

1. The Sun ONE Application Server 6.5 processes:

   ```
   /iplanet/ias6/ias/bin/.kas
   /bin/ksh /iplanet/ias6/ias/bin/kas

   /iplanet/ias6/ias/bin/.kxs -cset CCS0 -eng 0
   /bin/ksh /iplanet/ias6/ias/bin/kxs -cset CCS0 -eng 0

   /iplanet/ias6/ias/bin/.kcs -cset CCS0 -eng 2
   /bin/ksh /iplanet/ias6/ias/bin/kcs -cset CCS0 -eng 2

   /bin/ksh /iplanet/ias6/ias/bin/kjs -cset CCS0 -eng 1
   /iplanet/ias6/ias/bin/.kjs -DPATH=/iplanet/ias6/
     ias/bin:/
   iplanet/ias6/ias/APPS/
   ```

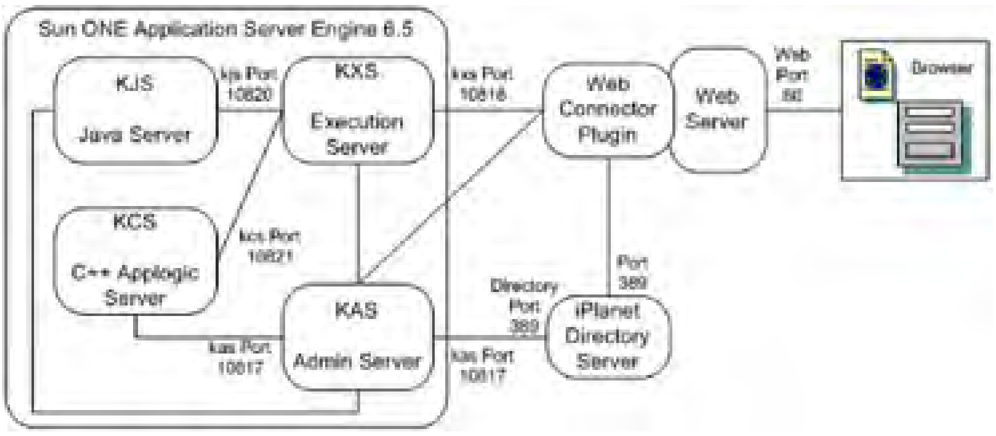
   a. kas— administration processes.
   b. kxs— execution processes. These processes receive requests from the web connector on the web server. The requests are passed on to the Java program process or the C++ process.
   c. kcs— C++ process to execute C++ programs.
   d. kjs— Java program process to execute Java class files.
2. The Sun ONE Directory Server process:

   ```
   ./ns-slapd -D /iplanet/ias6/slapd-art -i /iplanet/ias6/slapd-
     sun/logs/pid
   ```
3. The Sun ONE Web Server instance that has the application server web connector:

   ```
   ./uxwdog -d /iplanet/iws/https-ias65/config
   ns-httpd -d /iplanet/iws/https-ias65/config
   ns-httpd -d /iplanet/iws/https-ias65/config
   ```

Each of these processes, as well as each of the listener ports, is represented in Figure 21.1.

Figure 21.1. Sun ONE Application Server processes and related processes.



Stopping Processes (UNIX)

The Sun ONE Application Server connects to both the web server and the directory server. It should be shut down first.

```
# /iplanet/ias6/ias/bin/iascontrol stop
iascontrol for iPlanet Application Server 6.5 Test
  Drive
Connected to LDAP server on art.red.iplanet.com
  port 11024
iascontrol: iAS stopped successfully on art.red.
  iplanet.cm
```

The Sun ONE Web Server instance connects with the web connector. The web connector connects to the directory server, so it should be shut down next.

```
# /iplanet/iws60/https-art.red.iplanet.com/stop
```

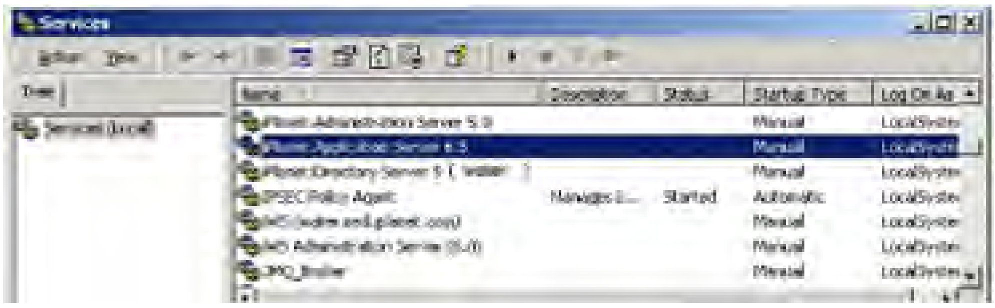
The Sun ONE Directory Server instance should be shut down last.

```
# /iplanet/ias6/slapd-art/stop-slapd
```

Starting, Viewing, and Stopping Processes in Windows

Under Windows, in the *Services window* (Figure 21.2), processes have been set to *Manual* after the installation so that the processes do not automatically start after the computer starts. It's set up this way because I do not want extra processes running on my computer unless I am testing them.

Figure 21.2. Windows 2000 Services window.



Services

Name	Description	Status	Startup Type	Log On As
iPlanet Administration Server 5.0			Manual	LocalSystem
iPlanet Application Server 6.5			Manual	LocalSystem
iPlanet Directory Server 5 (water)			Manual	LocalSystem
IPSEC Policy Agent	Manages I...	Started	Automatic	LocalSystem
iWS (water.red.iplanet.com)			Manual	LocalSystem
iWS Administration Server (6.0)			Manual	LocalSystem
JMQ_Broker			Manual	LocalSystem

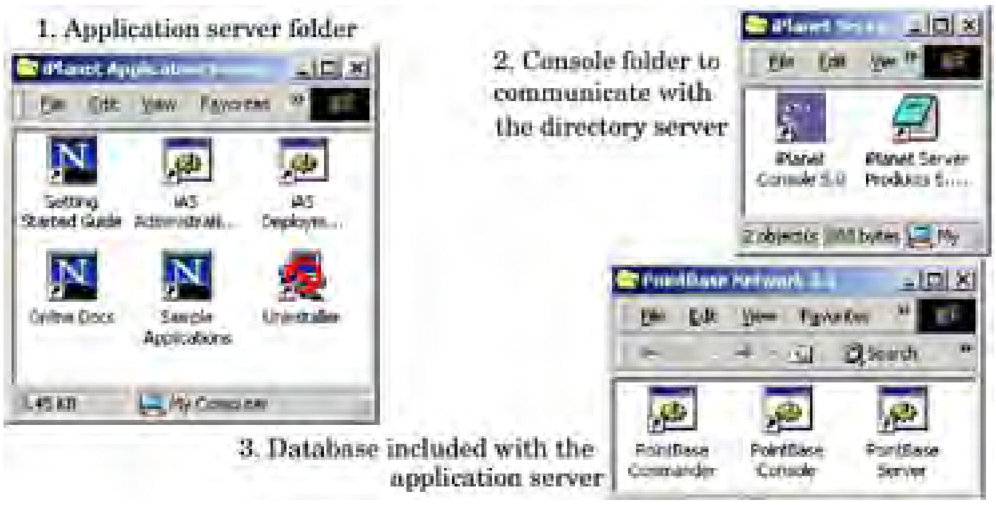
After the Installation (Windows)

In the *Services* window, the services related to the application server are

1. The application server iPlanet Application Server 6.5
2. The web server instance iWS (water.red.iplanet.com)
3. The web server administration server iWS Administration Server (6.0)
4. Administration server iPlanet Administration Server 5
5. Directory server instance iPlanet Directory Server 5 (water)

There are three new application server folders on your desktop, as shown in Figure 21.3.

Figure 21.3. New application server folders on desktop.

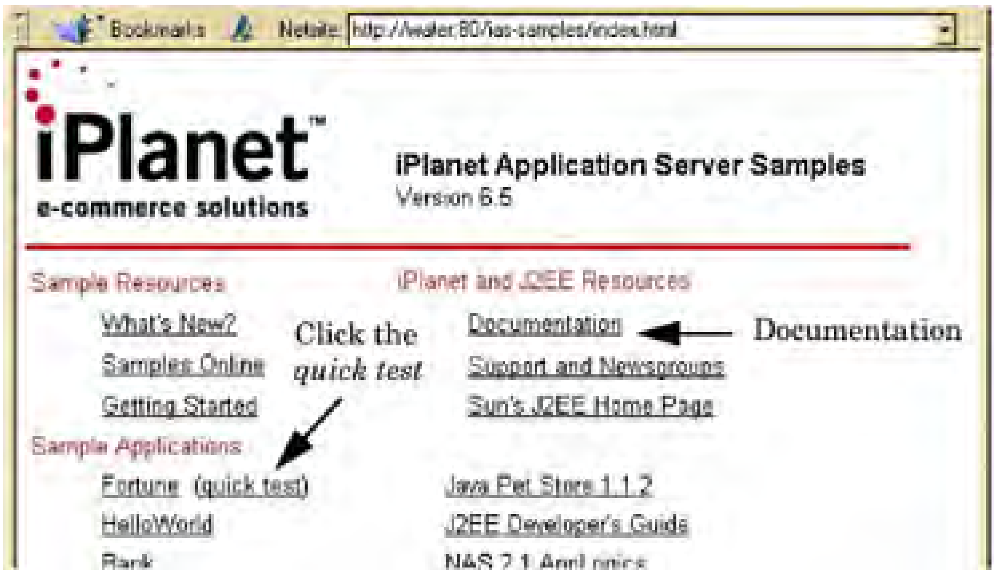


[Team LiB]

Testing the Installation

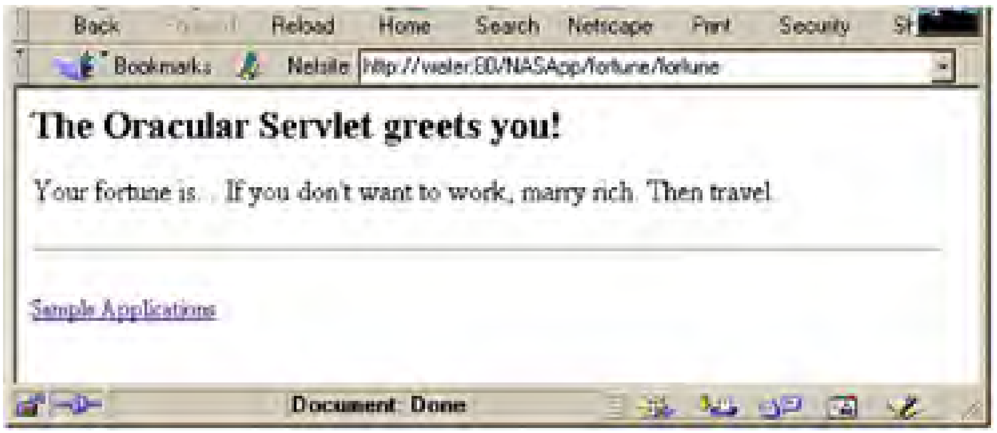
Start the Sun ONE Application Server and related processes (the directory server and the web server). Then use a browser to load the sample applications web page (URL *http://water.red.iplanet.com/ias-samples/index.html*) shown in Figure 21.4. Click the *Fortune quick test* link. This runs the fortune Java program.

Figure 21.4. Sample applications web page.



If your fortune is displayed, then your Sun ONE Application Server has a future. If you see a fortune come up as in Figure 21.5, your test has been successful—you have a running Sun ONE Application Server. Next let's look at working with the server.

Figure 21.5. Application server fortune.



The Oracular Servlet greets you!

Your fortune is... If you don't want to work, marry rich. Then travel.

Viewing the Log Files

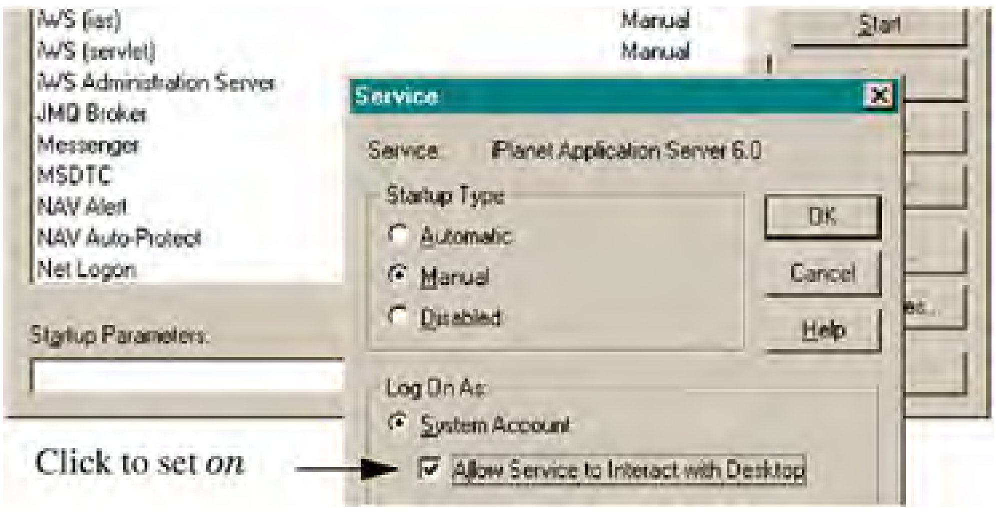
At times it is convenient to watch the log files while the programs are starting or running. In UNIX, there is the tail command; for example, tail -f /iplanet/ias6/ias/logs/kjs*.

```
# tail -f /iplanet/ias6/ias/logs/kjs*
07/Aug/2001 17:15:33:7] info: REQ-012: thread add
[07/Aug/2001 17:15:33:7] info: REQ-012: thread add
[07/Aug/2001 17:15:33:8] info: ENGINE-ready:
  ready: 10820
[07/Aug/2001 17:15:34:4] info: PROT-006: new con-
  nection established
[07/Aug/2001 17:15:34:4] info: PROT-006: new con-
  nection established
[07/Aug/2001 17:16:22:7] info: -------------------
[07/Aug/2001 17:16:22:7] info: FortuneServlet:
  init
[07/Aug/2001 17:16:22:7] info: -------------------
[07/Aug/2001 17:16:22:8] info: -------------------
[07/Aug/2001 17:16:22:8]info:jsp.APPS.fortune.for-
  tune:init
[07/Aug/2001 17:16:22:8] info: -------------------
```

This command will allow you to watch the kjs engine in action. As calls are made to the application server, the log listing continues to show the new lines that are being added to the log file. This is a property of the tail command.

It is possible to do something similar in WinNT. From the *Services* window (Figure 21.6), double-click on *Sun ONE Application Server 6.0*. Next, click the check box labeled *Allow Service to Interact with Desktop* to select it, and save the change.

Figure 21.6. Turning on viewing of log files.



Service: iPlanet Application Server 6.0

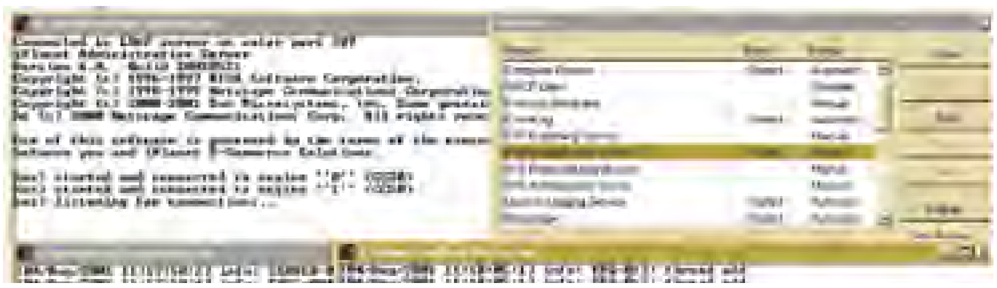
Startup Type: Automatic [✓ Manual] Disabled

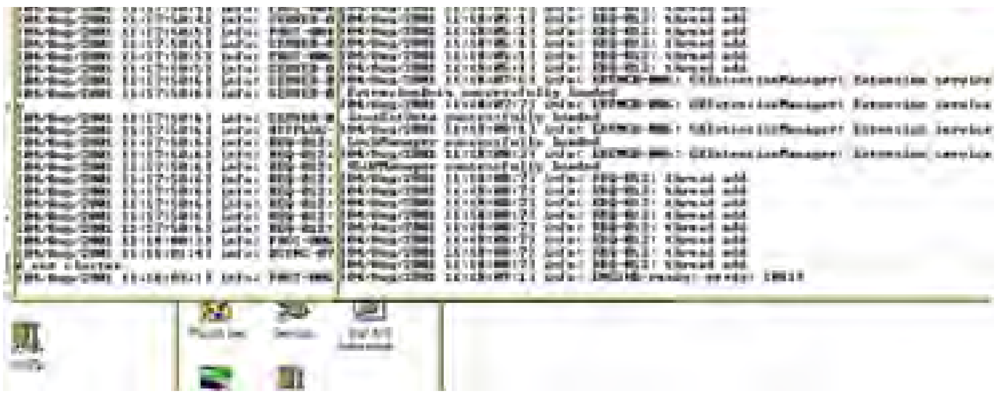
Log On As: [✓ System Account]

[✓] Allow Service to Interact with Desktop

When the *Sun ONE Application Server* is restarted, each of the processes will open a command window and log messages will scroll on the screen (Figure 21.7).

Figure 21.7. Display of the application server log-in information.





Note: If you close the command window, the process is shut down.

To stop the log messages in the windows, stop the service *Sun ONE Application Server 6.5*, deselect *Allow Service to Interact with Desktop*, and save the change.

Sun ONE Application Server Administration Tool

Controlling Process Startup

If you are going to run only Java programs, and if interaction with the application server is only through the browser, then you need to run only the kas, kxs, and kjs processes. You don't need the other processes.

Make sure the iPlanet Directory Server is running, that the iPlanet Application Server administration program is running, and that the application server itself is running. Then start the Admin Tool

- from UNIX: */iplanet/ias6/ias/bin/ksrvadmin &*
- from WinNT: *Start /Programs/iPlanet Application Server 6.5/iAS Administration Tool* (see Figure 21.8). Then the Admin Tool will start and its window will open (see Figure 21.9). The buttons across the top of the Admin Tool window are the major things that the Admin Tool can do.

Figure 21.8. Starting the Admin Tool.

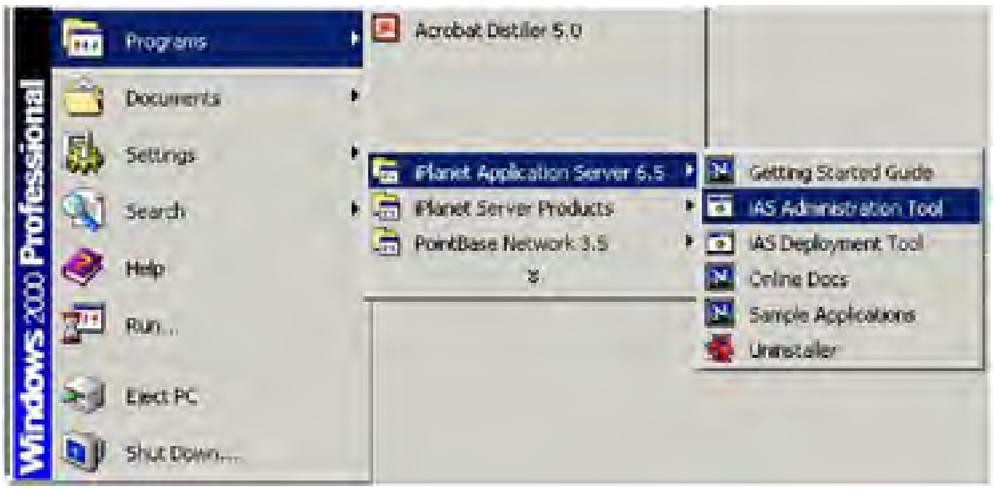
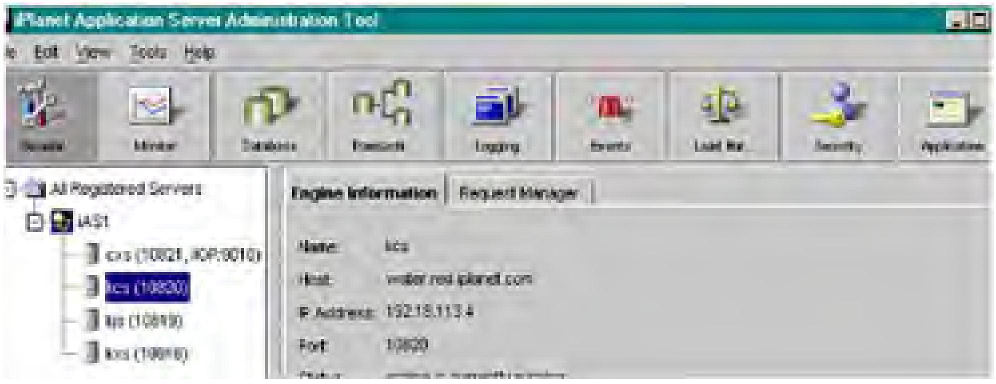


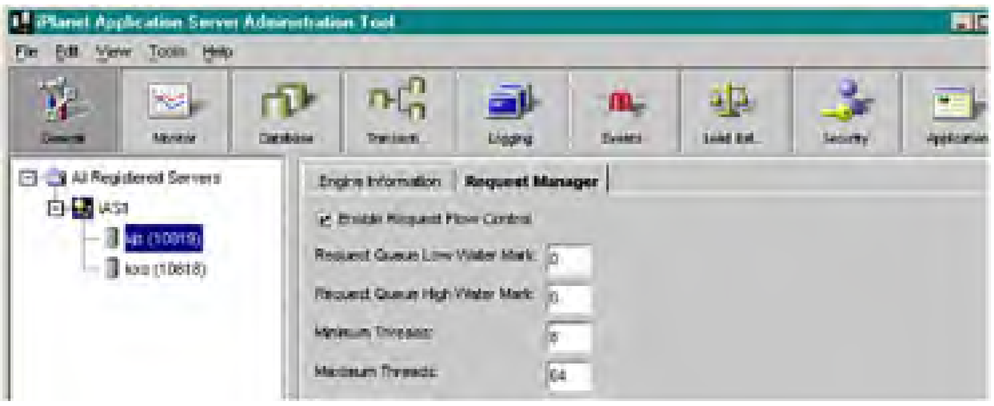
Figure 21.9. Administration Tool window.



Name: kcs
Host: [illegible]
IP Address: 192.18.113.4
Port: 10820

In Figure 21.9, the *kcs (10820)* process is selected. If you do not plan to run any C++ programs under the application server, select *kcs (10820)* and then (from the main menu) select *Edit/Delete* to remove it. From this point on, the process will not start. Since we will not be doing any rich client interaction with the application server, we can delete the *cxs* process as well, thereby giving us the window in Figure 21.10.

Figure 21.10. Minimum servers to run Java programs.



Enable Request Flow Control

Request Queue Low Water Mark: 0

Request Queue High Water Mark: 0

Minimum Threads: 0

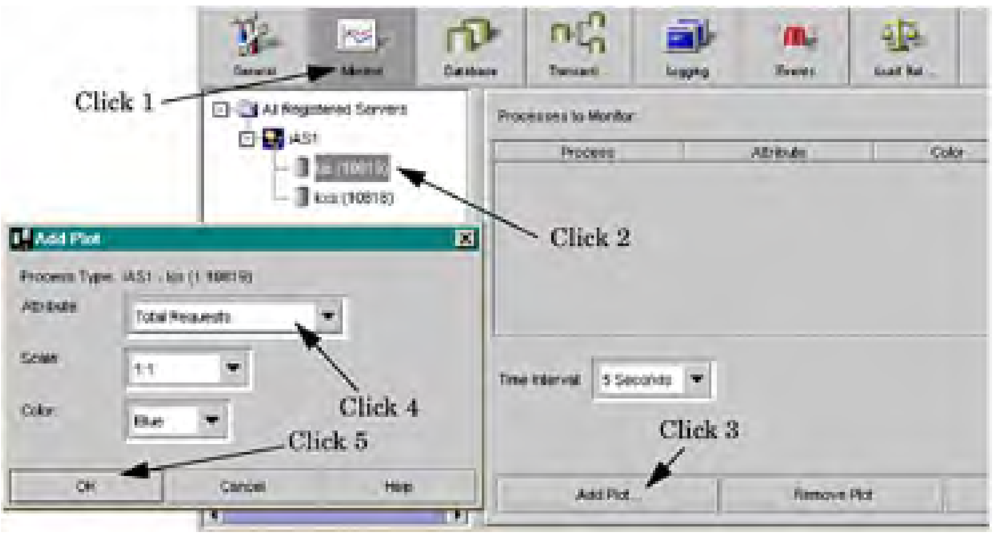
Maximum Threads: 64

Figure 21.10 shows only the processes required to run Java programs—*kjs (10819)*—from the application server.

Notice in Figure 21.10 that, by selecting the *Request Manager* tab, tuning can be managed for the kjs process; for example, the number of threads this process can manage can be set up right here.

Now set up a graph to monitor the kjs process. On the screen in Figure 21.11, click *Monitor* on the menu bar, the *kjs (10819)* process, then *Add Plot...*. In the *Add Plot* screen, select *Total Requests*, *1:1*, and *Blue*. Then click *OK*.

Figure 21.11. Setting up a graph to monitor kjs progress.



Process Type: iAS1 - kjs (1 10819)

Attribute: Total Requests

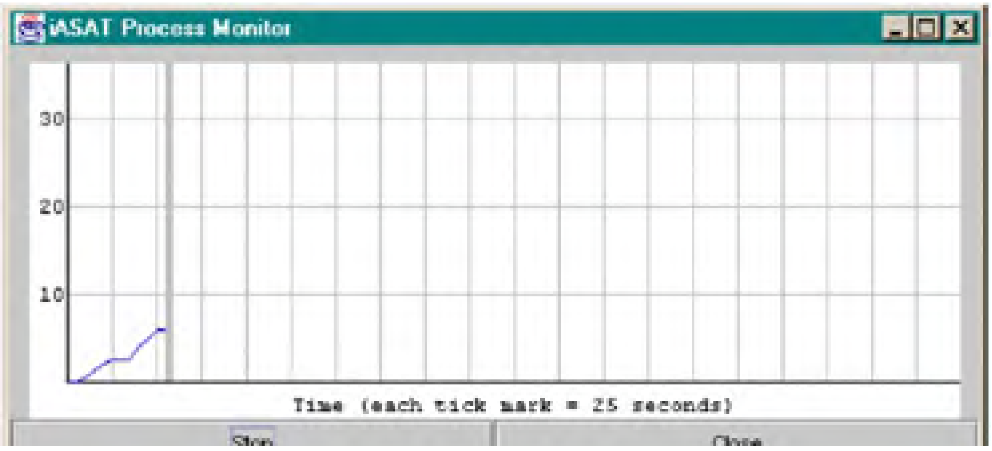
Scale: 1:1

Color: Blue

Time Interval: 5 Seconds

Now, from the browser call the sample Fortune program *http://water.red.iplanet.com/NASApp/fortune/fortune*. Then keep clicking the *Reload* button and watch the stats line rise in Figure 21.12.

Figure 21.12. Graph showing kjs progress.



The *Monitor* menu option in the Admin Tool is great for watching what is happening on the application server. Yes, there are many other things that can be done, e.g., send the data to a file or send the log data to a database and write custom reports.

Have a look around the Administration Tool and try things.

Summary

To administer the Sun ONE Application Server 6.5, you must also manage the Sun ONE Directory Server and the Sun ONE Web Server. Previous chapters have all the information needed regarding these other servers.

Once the application server is running, a simple test using the browser demonstrates that it is working. Watching the log files is invaluable when it comes to debugging an installation or application.

The Sun ONE Application Server Administration Tool makes it easy to configure the processes and monitor the server.

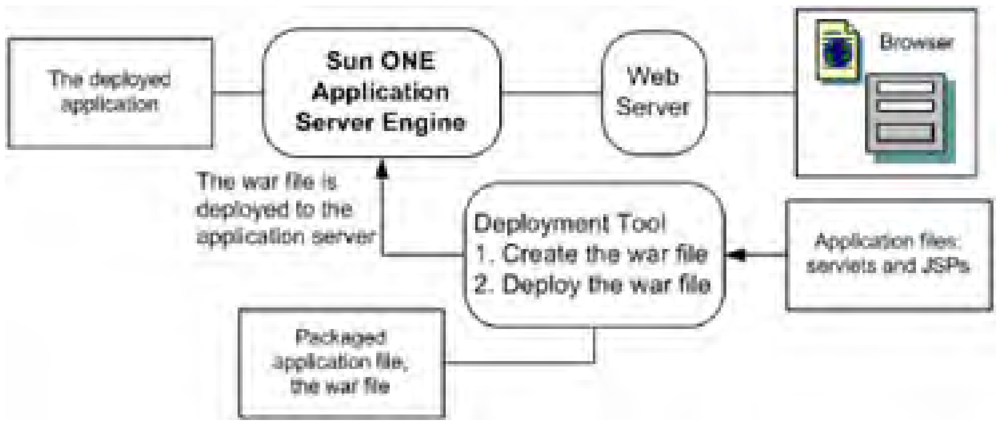
All in all, this is a very straightforward package.

Chapter 22. Deploying Programs into the Sun ONE Application Server 6.5

In this chapter, you'll see the steps needed to get you started installing and running programs in the Sun ONE Application Server environment. We'll look at three types of programs: a servlet, a JSP, and a servlet forwarding a request to a JSP. You will deploy a servlet into the Sun ONE Application Server. Then we'll add a JSP into the Hello World application. Finally, create and deploy an application that demonstrates the separation (into separate Java programs) of processing a request and formatting a response.

This chapter also introduces the concepts of packaging program files into a zipped WAR file. The WAR files are deployed into a Sun ONE Application Server using the Sun ONE Application Server Deployment Tool as illustrated in Figure 22.1. Once deployed, programs are tested, then modified, redeployed, and retested.

Figure 22.1. Create and deploy applications.


The deployed application
Sun ONE Application Server Engine
Web Server
Browser
The war file is deployed to the application server
Deployment Tool
1. Create the war file
2. Deploy the war file
Application files: servlets and JSPs
Packaged application file, the war file

Deploying a Servlet into the Application Server

In this section, a servlet is deployed into the Sun ONE Application Server. Here is an overview of the steps to set up and run a servlet in the application server:

1. Start the Deployment Tool.
2. When the Deployment Tool is running, create a new application.
3. Add the servlet file, HelloWorldServlet.class, into the new application, creating a WAR file. (The servlet sample file HelloWorldServlet.class comes with the Sun ONE Web Server and was installed into the Sun ONE Web Server environment and tested in a previous chapter.)
4. Deploy the WAR file.
    a. Next, register the Sun ONE Application Server with the Deployment Tool.
    b. Then deploy the WAR file to the newly registered application server.
    c. Use a browser to test the servlet.

Start the Deployment Tool

The Deployment Tool has three main functions: organizing application files, packaging application files, and deploying the packaged files to the application server.

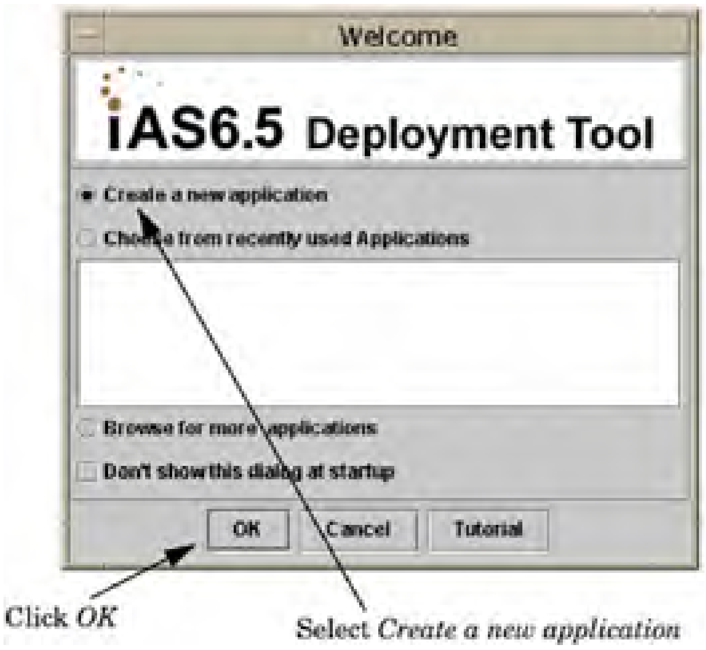
To start the Deployment Tool from UNIX, use the command `/iplanet/ias6/ias/bin/deploytool &`.

To start the Deployment Tool from Windows, click on *Start/Programs/iPlanet Application Server 6.5/iAS Deployment Tool*.

To start the Deployment Tool from Windows, click *Start/Programs/iPlanet Application Server/iAS Deployment Tool*.

Once the tool starts (see *Welcome* window in Figure 22.2), the instructions are the same for both UNIX and Windows. The following are the steps for creating a new application with one Java program—the classic Hello World servlet. On the *Welcome* web page, select *Create a new application* and then click *OK*.

Figure 22.2. The Sun ONE Application Server Deployment Tool.

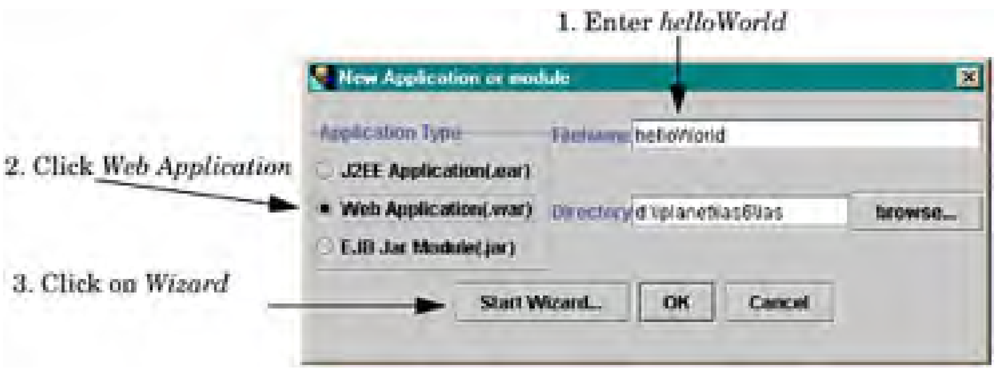

Welcome
iAS6.5 Deployment Tool
Create a new application
Choose from recently used Applications
Browse for more applications
Don't show this dialog at startup
OK
Cancel
Tutorial
Click OK
Select Create a new application

Create a New Application

Now create a new application that will contain the HelloWorldServlet.class file.

First, enter the filename *helloWorld* in the *FileName* field (Figure 22.3). Then click on the radio button next to *Web Application (.war).* Now click on *Start Wizard* to create a WAR file.

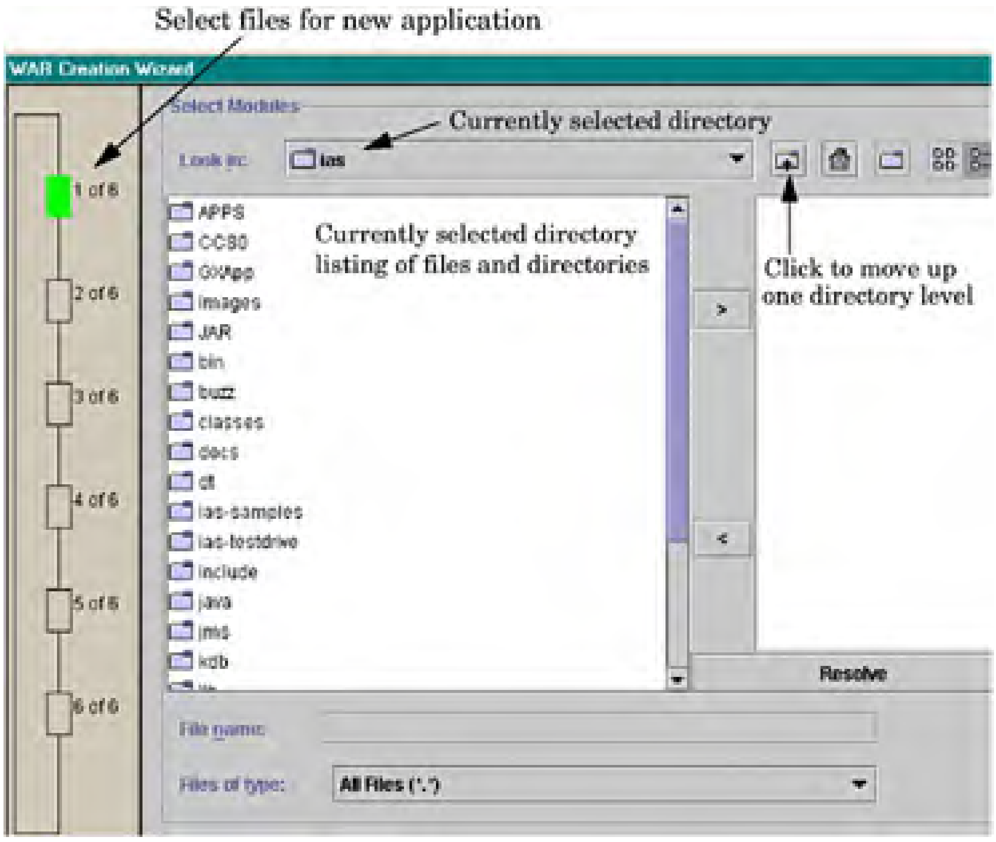
Figure 22.3. Creating a new application.



Filename: helloWorld
Directory: d:\iplanet\ias6\ias

In the *WAR Creation Wizard* window (Figure 22.4), click on *1 of 6* on left side to select files for the new application. You will see a list of files and directories in the directory that is currently selected (in the *Look in* field). Click on the Directory button to the right of the currently selected directory to move up one directory level.

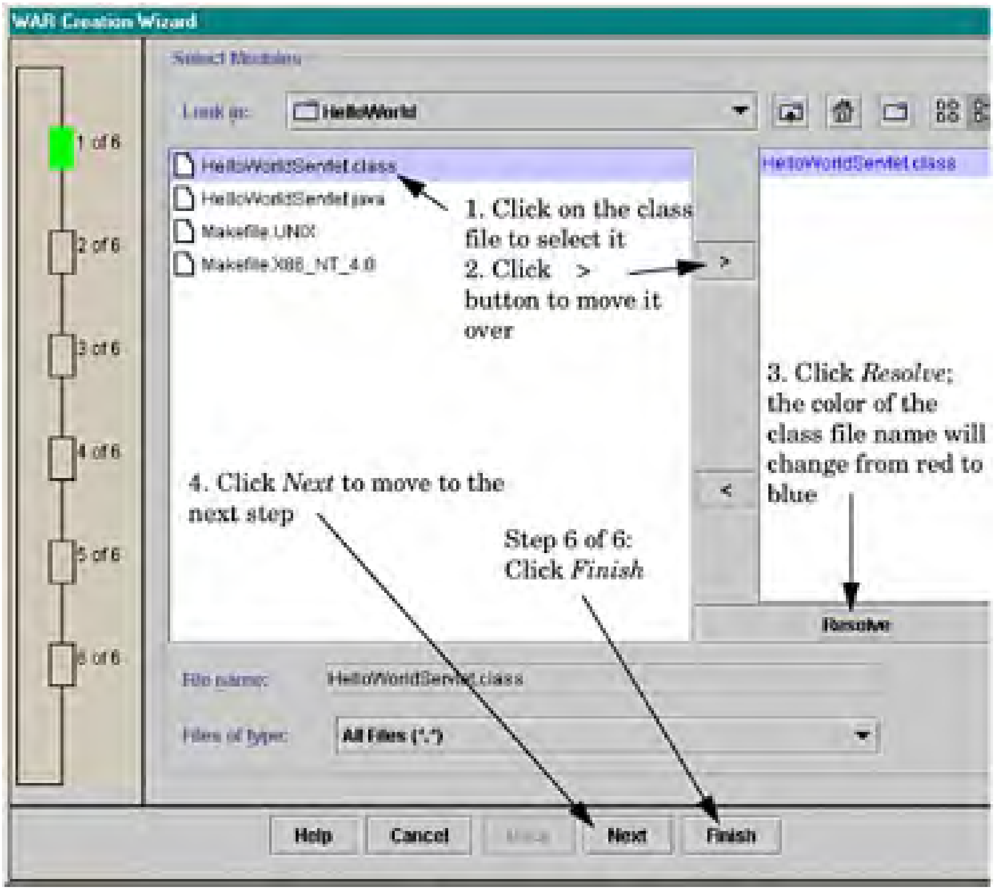
Figure 22.4. Creating a WAR file.



Add the Servlet File into the New Application

As demonstrated in Figure 22.5, click on *HelloWorldServlet.class* file from the Sun ONE Web Server sample files. In the step *1 of 6* window, use the directory buttons and file directory list to locate the *HelloWorldServlet.class* file in D:\iplanet\iws\plugins\samples\servlets\servlets\HelloWorld or /iplanet/iws/plugins/samples/servlets/servlets/HelloWorld; i.e., <web server install directory>/plugins/samples/servlets/servlets /HelloWorld.

Figure 22.5. Add the servlet file into the application.


WAR Creation Wizard
Select Modules
Look in: HelloWorld
HelloWorldServlet.class
HelloWorldServlet.java
Makefile.UNIX
Makefile.X86_NT_4.0
1 of 6
2 of 6
3 of 6
4 of 6
5 of 6
6 of 6
1. Click on the class file to select it
2. Click > button to move it over
3. Click *Resolve*; the color of the class file name will change from red to blue
4. Click *Next* to move to the next step
Step 6 of 6: Click *Finish*
Resolve
File name: HelloWorldServlet.class
Files of type: All Files (*.*)
Help
Cancel
Back
Next
Finish

Step 1 of 6, Figure 22.5

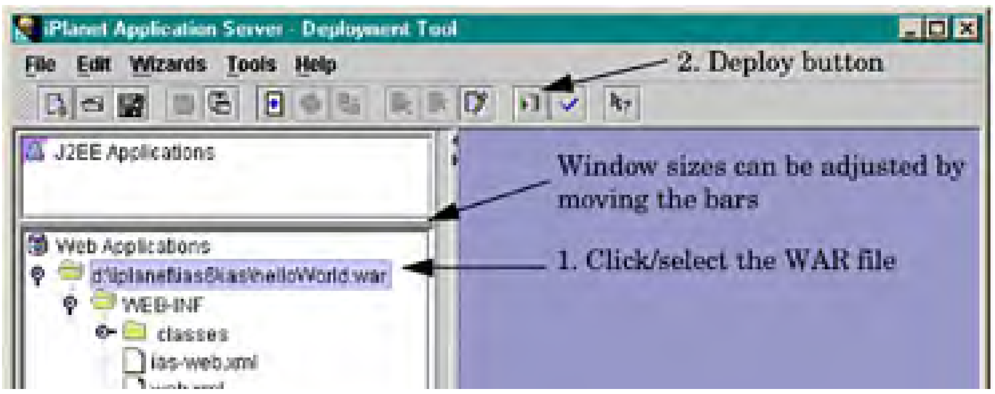
To select the HelloWorldServlet.class file:

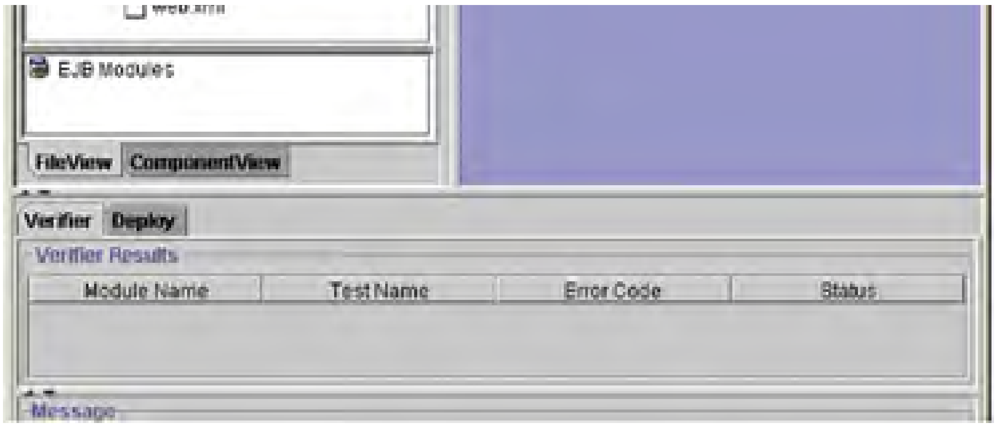
1. Click the class file name HelloWorldServlet.class.
2. Click the > in the middle to add the file to the right-hand column.
3. Click the Resolve button.

The servlet has been added.

On each of the next steps (Step 2 of 6, Step 3 of 6, Step 4 of 6, Step 5 of 6) just keep clicking the *Next* button. Then, on Step 6 of 6, click *Finish* and Figure 22.6 will appear. That's it—the WAR file has been created.

Figure 22.6. Deploying the WAR file.


iPlanet Application Server - Deployment Tool
File Edit Wizards Tools Help
2. Deploy button
J2EE Applications
Window sizes can be adjusted by moving the bars
Web Applications
d:\iplanet\ias6\ias\helloWorld.war
1. Click/select the WAR file
WEB-INF
classes
ias-web.xml

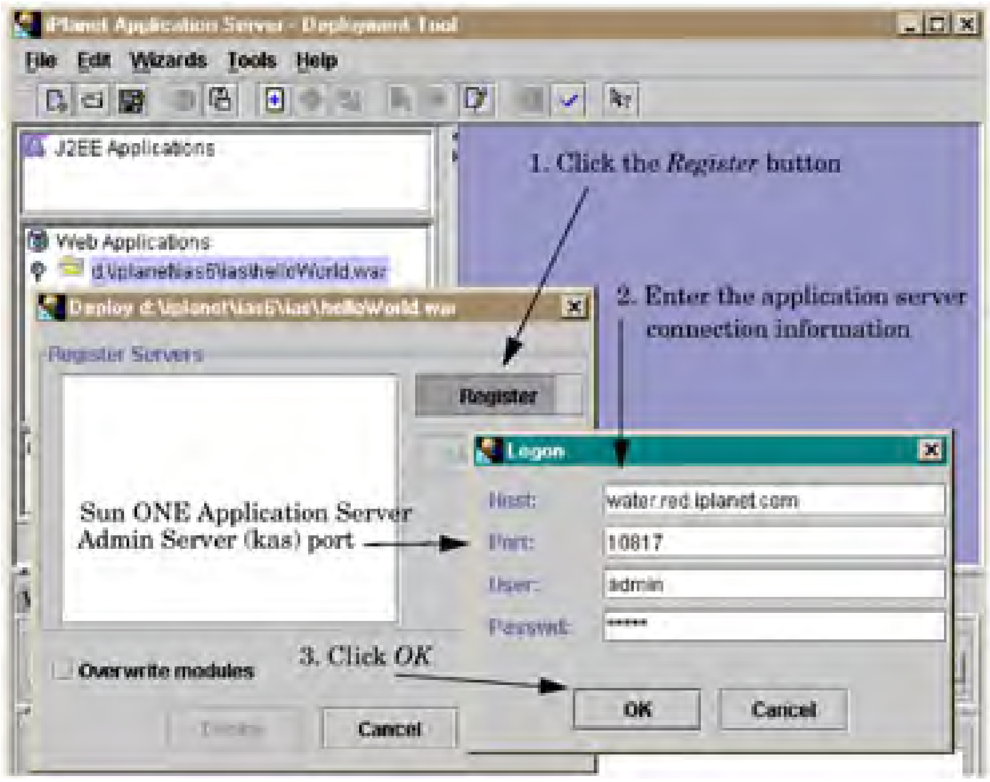


Deploy the WAR File

Deployment is a straightforward process: Select an application to deploy; select an application server to deploy to; deploy the application to the selected application server.

From the Deployment Tool main page (Figure 22.6), under *Web Applications* select the *HelloWorld* WAR file (*d:\iplanet\ias6\ias\helloWorld.war*) to be deployed. This is the file that was created by the *WAR Creation Wizard* earlier in the chapter. Click the Deploy button to deploy the *HelloWorld* WAR file (Figure 22.7) into the application server.

Figure 22.7. Registering a new application server.


iPlanet Application Server - Deployment Tool
File
Edit
Wizards
Tools
Help
J2EE Applications
Web Applications
d:\iplanet\ias6\ias\helloWorld.war
Deploy d:\iplanet\ias6\ias\helloWorld.war
Register Servers
Register
1. Click the *Register* button
2. Enter the application server connection information
Logon
Host:
Port:
User:
Passwd:
Sun ONE Application Server Admin Server (kas) port
3. Click *OK*
Overwrite modules
Deploy
Cancel
OK
Cancel

Host: water.red.iplanet.com
Port: 10817
User: admin
Passwd: ******

Register the Sun ONE Application Server with the Deployment Tool

The next step is to select an application server to deploy to. Since this is the first deployment, a new application server needs to be registered with the Deployment Tool. Make special note that the application server *must* be running for this deployment step.

In the Deploy window, click on the *Register* button. When the *Logon* window opens, enter the application server connection information. Then click *OK*.

Now that the new server has been registered, you can select the target of our deployment. In Figure 22.8, click on the server name to which you wish to deploy (*water.red.iplanet.com:10817*). Then click *Deploy*. Watch the messages (Figure 22.9) while the *Hello World* application is being deployed.

Figure 22.8. Choosing an application server to deploy to.

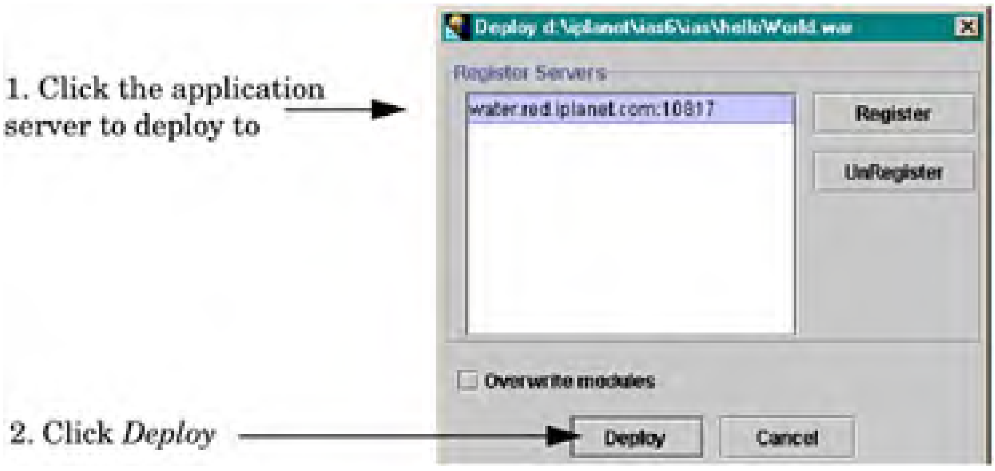
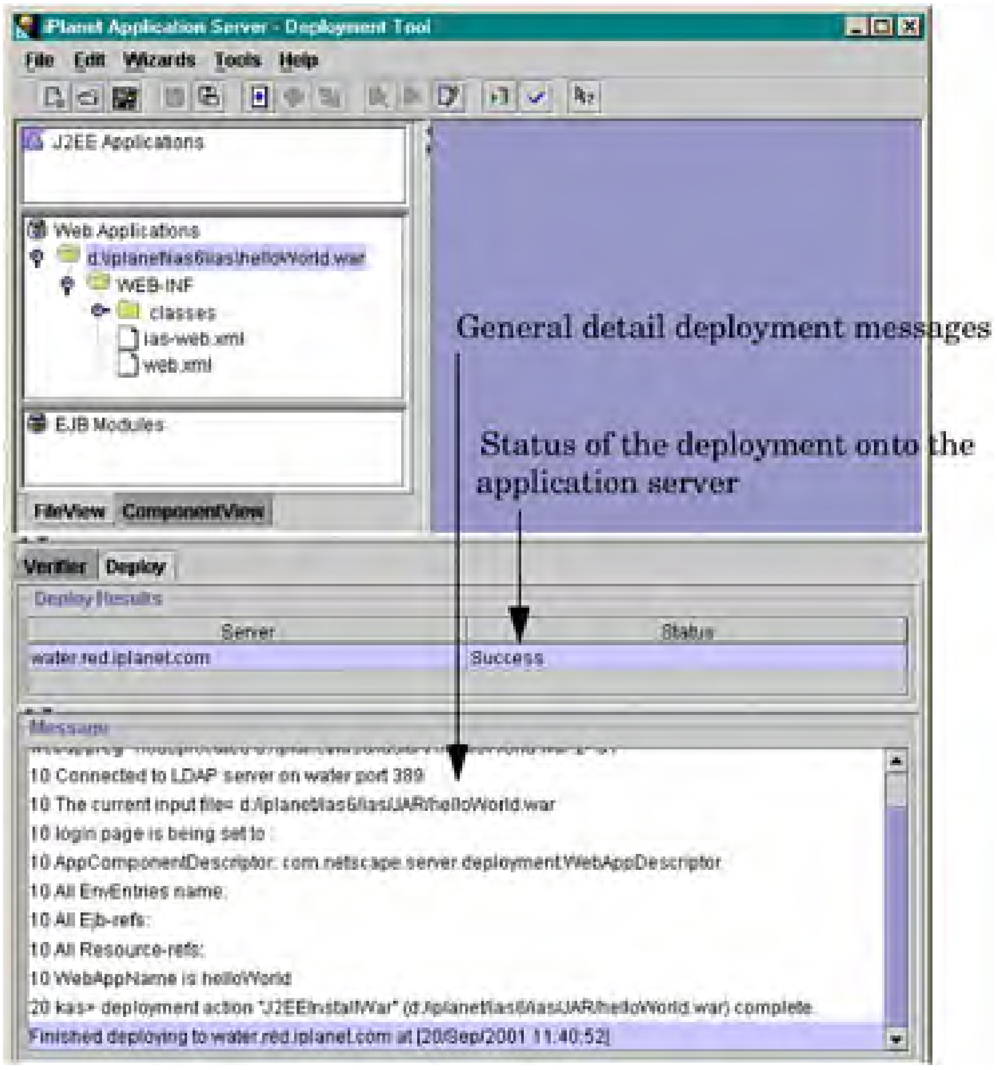

Deploy d:\iplanet\ias6\ias\helloWorld.war
Register Servers
water.red.iplanet.com:10817
Register
UnRegister
Overwrite modules
Deploy
Cancel
1. Click the application server to deploy to
2. Click *Deploy*

Figure 22.9. Deployment messages.


iPlanet Application Server - Deployment Tool
File Edit Wizards Tools Help
J2EE Applications
Web Applications
d:\iplanet\ias6\ias\helloWorld.war
WEB-INF
classes
ias-web.xml
web.xml
EJB Modules
FileView ComponentView
General detail deployment messages
Status of the deployment onto the application server
Verifier Deploy
Deploy Results
Message

Server	Status
water.red.iplanet.com	Success

```
10 Connected to LDAP server on water port 389
10 The current input file= d:\iplanet\ias6\ias\JAR\helloWorld.war
10 login page is being set to :
10 AppComponentDescriptor: com.netscape.server.deployment.WebAppDescriptor
10 All EnvEntries name:
10 All Ejb-refs:
10 All Resource-refs:
10 WebAppName is helloWorld
20 kas> deployment action "J2EEInstallWar" (d:\iplanet\ias6\ias\JAR\helloWorld.war) complete
Finished deploying to water.red.iplanet.com at [20/Sep/2001 11:40:52]
```

When deploying a new application, a new directory under the application modules directory is created:

```
<app.server install directory>/ias/APPS/modules/<appli-
  cation name>
```

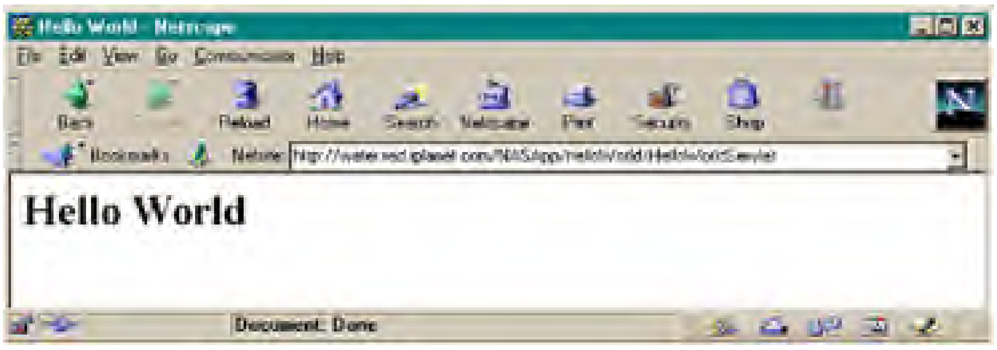
For example:

```
/iplanet/ias6/ias/APPS/modules/helloWorld
```

Test the Servlet

Start a browser and test. The syntax for the URL is *http://water.red.iplanet.com/NASApp/helloWorld/HellowWorldServlet*. When the opening window in Figure 22.10 appears, you know the test succeeded.

Figure 22.10. Test successful.


Hello World - Netscape
File Edit View Go Communicator Help
Back
Reload
Home
Search
Netscape
Print
Security
Shop
Bookmarks
Netsite: http://water.red.iplanet.com/NASApp/helloWorld/HelloWorldServlet
Hello World
Document: Done

[Team LiB]
PREVIOUS NEXT

Add a JSP into the Hello World Application

There are two ways to add a JSP into a deployed application:

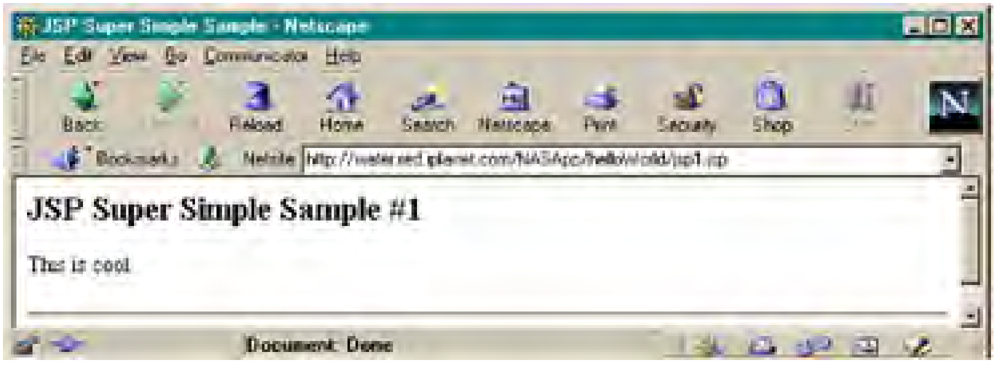
1. Create a JSP in the application file directory (you did this earlier).
2. Add the JSP into the application and redeploying the application.

The first method works because JSPs do not need to be registered with the application server. Therefore, the Sun ONE Application Server does not require that JSPs be deployed using the Deployment Tool. An example of the first method would be to add the following JSP file jsp1.jsp in the application directory /iplanet/ias6/ias/APPS/modules/helloWorld. Follow these steps:

```
<head>
<title>JSP Super Simple Sample</title>
</head>
<BODY  bgcolor="#FFFFFF" text="#000000">

<h2>JSP Super Simple Sample #1</h2>

<% String s1 = "cool"; %>

This is <%=s1%>.

<p><hr><p>
</body>
</html>
```

Then test to make sure the JSP works using URL *http://water.red.iplanet.com/NASApp/helloWorld/jsp1.jsp* (Figure 22.11).

Figure 22.11. Server test successful.

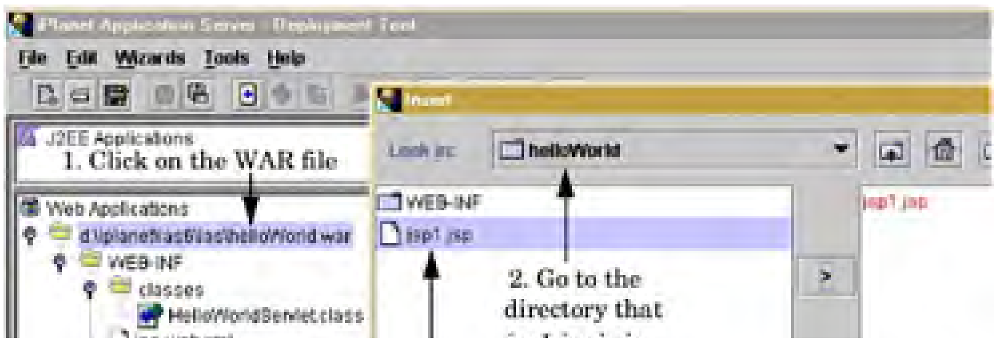

JSP Super Simple Sample - Netscape
File Edit View Go Communicator Help
Back Reload Home Search Netscape Print Security Shop
Bookmarks Netsite http://water.red.iplanet.com/NASApp/helloWorld/jsp1.jsp
Document: Done

JSP Super Simple Sample #1

This is cool.

Again, the second method is to add the JSP into the application—the *helloWorld* application—and redeploy the application. On the Deployment Tool main page (Figure 22.12), first click on the WAR file *d:\iplanet\ias6\ias\helloWorld.war* under *Web Applications*. In the *Look in* field, go to the directory containing *jsp1.jsp* and click on *jsp1.jsp* in the box below. Then click *Resolve*, which will open the *Update Dest. Paths* window (Figure 22.13).

Figure 22.12. Adding JSP into the application.



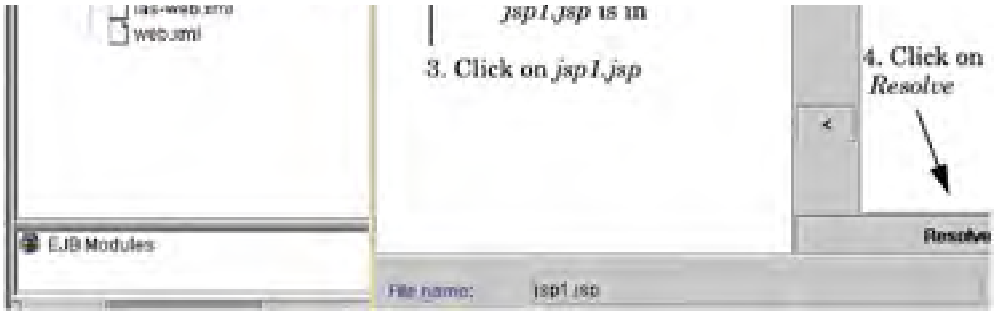
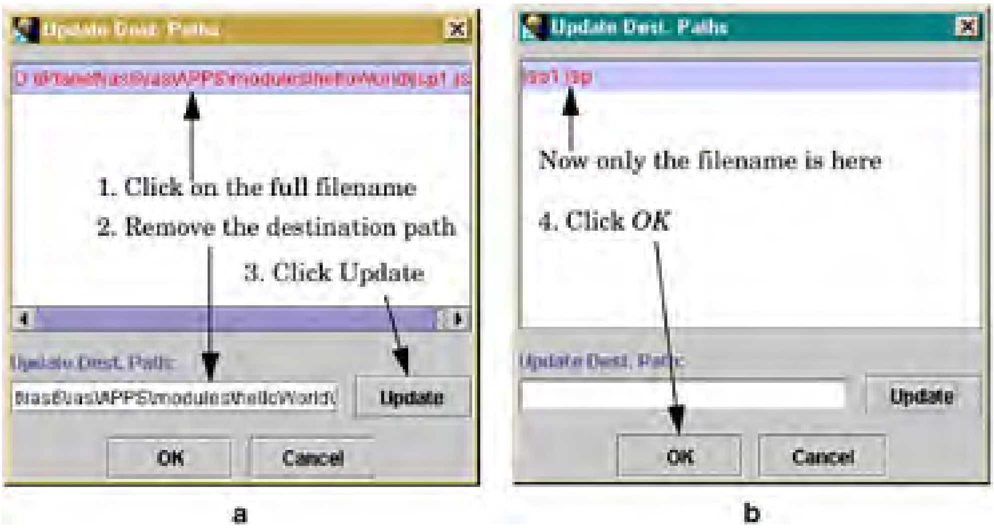


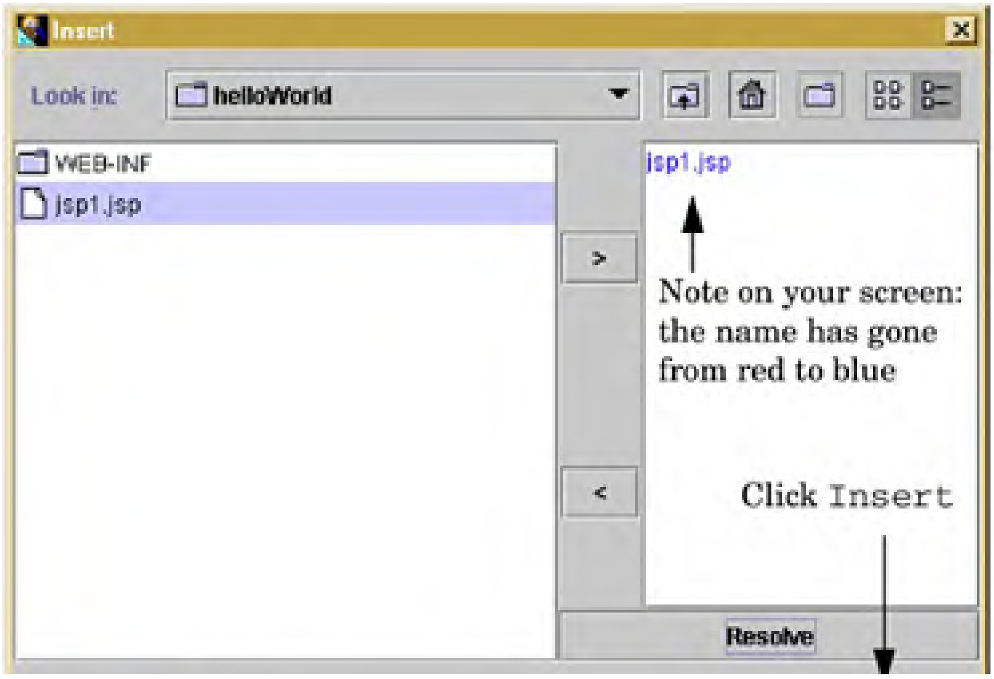
Figure 22.13. Update the destination path.



In Figure 22.13, first click on the full filename at the top. Then remove the destination path by deleting what's in the field under *Update Dest. Path.* Now, in Figure 22.13, only filename *jsp1.jsp* appears where the full filename had been. Click *OK*.

The *Insert* window (Figure 22.14) opens; on your screen, note that filename *jsp1.jsp* has gone from red to blue. Click *Insert*. *jsp1.jsp* will now be inserted into the *helloWorld* application.

Figure 22.14. Inserting jsp1.jsp into helloWorld.



Now that the file has been added to the application, you are back at the Deployment Tool main page to redeploy the application (Figure 22.15). First, click on the WAR file under *Web Applications*, and then click *Save* to save the updates to the application. Now click on the WAR file again to select the application (as shown in Figure 22.16). Then click on the deploy button on the toolbar. Click the check box to select *Overwrite modules* and click *Deploy*. You have just redeployed the application with the added JSP.

Figure 22.15. Saving the application updates.

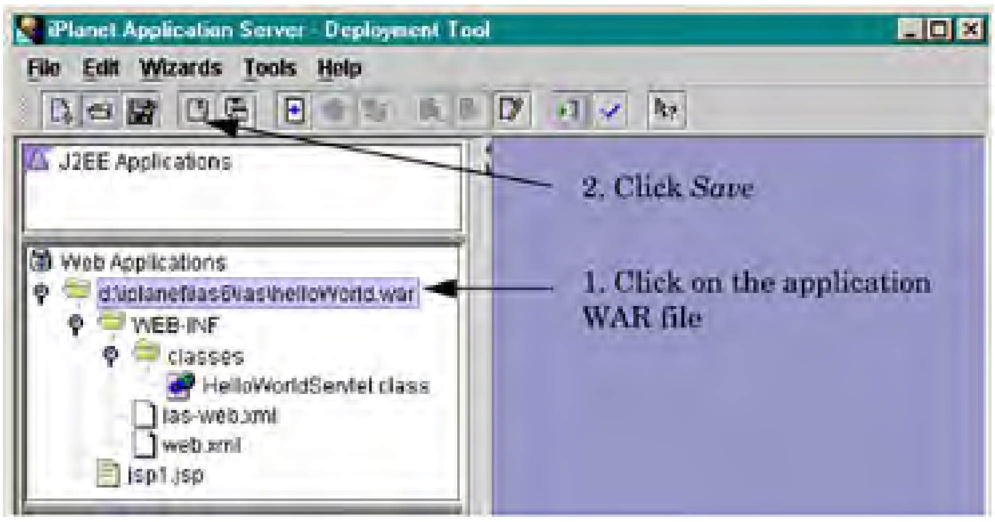
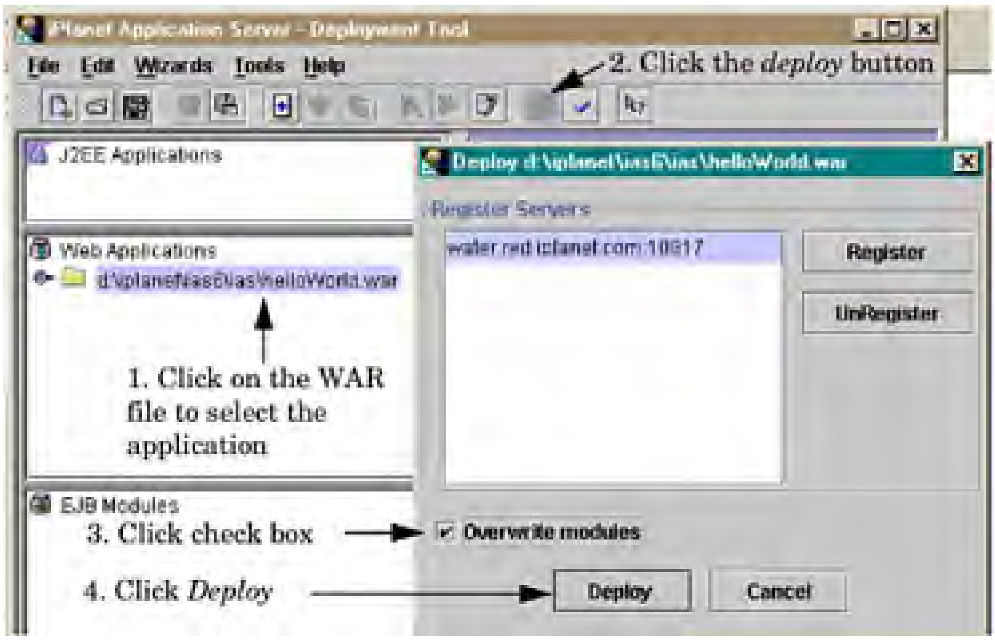
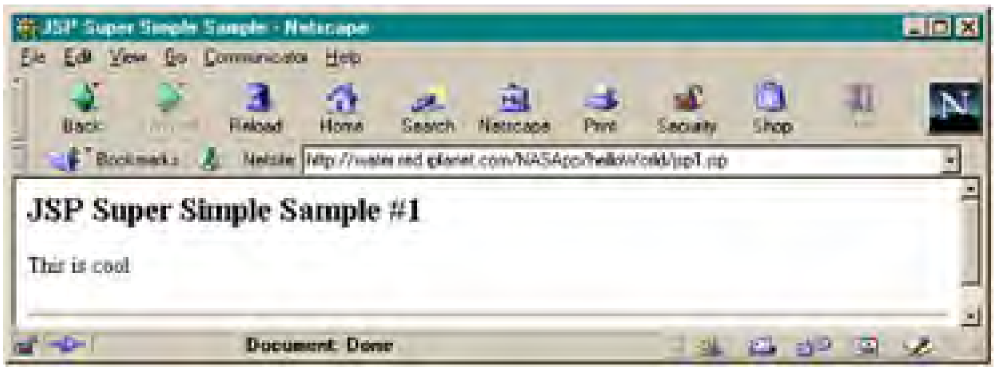

iPlanet Application Server - Deployment Tool
File Edit Wizards Tools Help
J2EE Applications
2. Click *Save*
Web Applications
d:\iplanet\ias6\ias\helloWorld.war
1. Click on the application WAR file
WEB-INF
classes
HelloWorldServlet.class
ias-web.xml
web.xml
jsp1.jsp

Figure 22.16. Redeploying the application with added JSP.


iPlanet Application Server - Deployment Tool
File Edit Wizards Tools Help
2. Click the *deploy* button
J2EE Applications
Deploy d:\iplanet\ias6\ias\helloWorld.war
Register Servers
water.red.iplanet.com:10817
Register
UnRegister
Web Applications
d:\iplanet\ias6\ias\helloWorld.war
1. Click on the WAR file to select the application
EJB Modules
3. Click check box
Overwrite modules
4. Click *Deploy*
Deploy
Cancel

Now it is time to test to find out whether the JSP works. Use a browser to test the JSP, go to URL *http://water.red.iplanet.com/NASApp/helloWorld/jsp1.jsp*. When the window shown in Figure 22.17 opens, you know you've been successful.

Figure 22.17. Adding JSP to Hello World is successful.


JSP Super Simple Sample - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop
Bookmarks Netsite
Document: Done

http://water.red.iplanet.com/NASApp/helloWorld/jsp1.jsp

JSP Super Simple Sample #1

This is cool

Developer Notes

Updating Class Files Already Deployed

When changes are made to a Java class file that has already been deployed to the application server, you can use one of two options to update the class file on the application server:

1. Replace the old class file that is in the application directory with a new class file (i.e., copy your new class file on top of the old class file). Note, in this case, the WAR file still has the old class file.
2. In the deployment tool, remove the current class file reference. Then reinsert the reference to the class file and redeploy the application. In this case, the WAR file contains the most up-to-date class file.

Recreating WAR Files

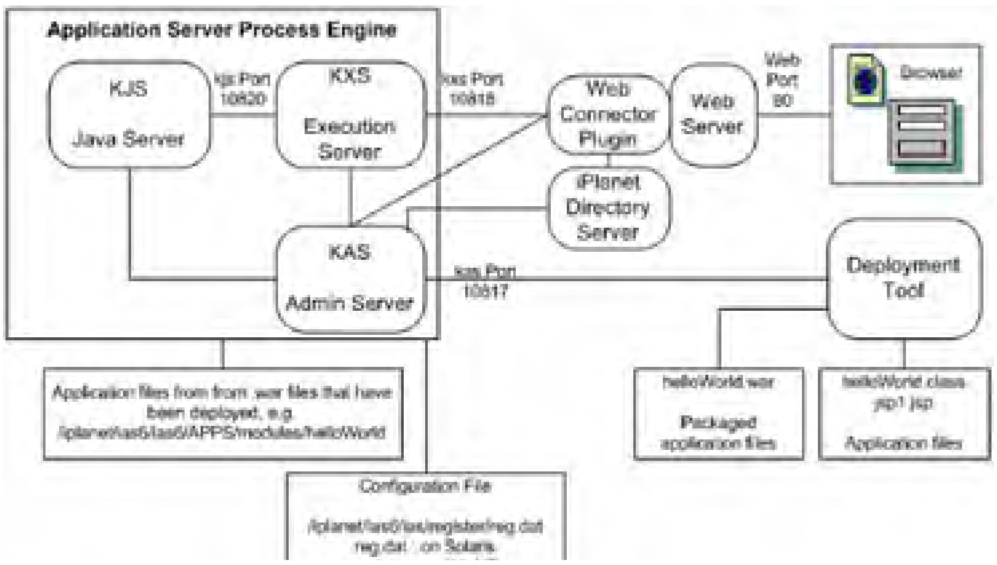
When creating applications on your own computer, recreate the WAR file only when necessary.

There is an alternative to using the GUI Deployment Tool: Create script files that will recreate the WAR files. When the WAR file is created, run the command-line deployment tool to deploy the WAR file. There is information regarding this on the Sun ONE Application Server 6.5 Test Drive CD. Also, you can find more information at *http://developer.iplanet.com/ias-samples/docs/build.html*.

Communications Between Deployment Tool and Application Server

The Deployment Tool communicates with the Sun ONE Application Server Administration Server when deploying files. The processes and services of the application server are diagramed in Figure 22.18.

Figure 22.18. The many parts of the Sun ONE Application Server 6.5 system.



When an application is deployed, information is stored in the application server's registry (reg.dat file on Solaris [UNIX] computers and in the Windows directory on Windows computers). Also, application information is stored in the directory server under the Sun ONE Application Server configuration branch.

Creating WAR Files to Sell on the Internet

When a WAR file has been created, it can be given to someone else to deploy on their own application server. This means that WAR files can be created and sold on the Internet. You can create an application, package the application into a single file, and put it onto a web site for people to download. If the web site has the ability to get people's credit card information, you should get their credit card info and bill them before the download.

People who download the WAR file will deploy it into their application servers and then run the application programs that you have created for them.

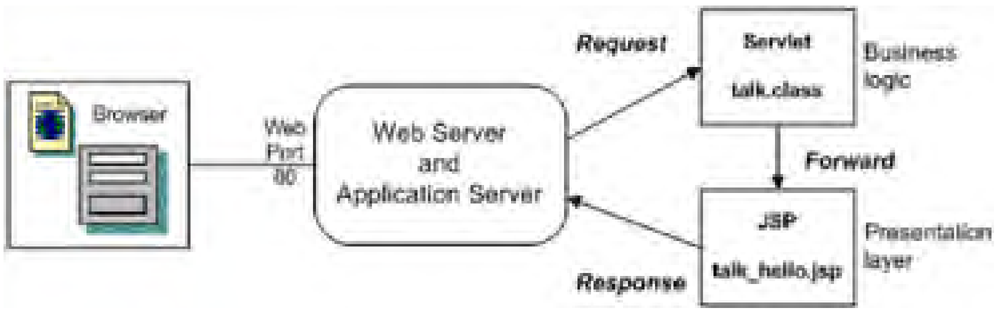
J2EE application WAR files are an excellent method of distributing software.

Demonstrating Separation of Processing a Request and Formatting a Response (a Servlet Forwarding a Request to a JSP)

In this exercise, a complete application is created and deployed. The application demonstrates the separation of processing a request and formatting a response into separate Java programs.

Figure 22.19 illustrates how a request is processed. The browser sends a request to the web server/application server. The web server/application server passes the request to the servlet. The servlet receives the request, does some processing, and puts data information into the session memory.

Figure 22.19. Processing a request.



The processing of the request is forwarded to the JSP. The sample JSP in this chapter is HTML with embedded Java code. The JSP uses the process data from the servlet to complete an HTML web page that is sent back to the web server/application server. The web server/application server returns the response to the browser with the web page from the JSP.

The exercise goes through these steps:

1. Create a working file directory.
2. Create the programs and other files for development and testing.
3. Compile the servlet.
4. Add the servlet files into the application *helloWorld*.
5. Redeploy *helloWorld*.
6. Test.

Create a Working File Directory

First create a working directory:

```
/iplanet/servlet-jsp
```

Create Programs and Files

In the working directory create this servlet file: filename talk.java.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class talk extends HttpServlet
{
   public void service (HttpServletRequest req, Http-
  ServletResponse res)
   throws ServletException, IOException
   {
      // Value to pass to the JSP:
      req.setAttribute( "talk_hello", "there..." );
// Pass runtime to the JSP:
      RequestDispatcher dispatcher =
         getServletContext().getRequestDispatcher(
  "/talk_hello.jsp" );
      dispatcher.forward( req, res );
   }

   public String getServletInfo() { return "Straight
  forward program."; }
}
```

In the working directory create this JSP file: filename talk_hello.jsp.

```
<%
    String s1 = (String) request.getAttribute(
  "talk_hello" );
%>
<html>
<head><title>Talk Hello</title></head>
<body>
2 methods to display values being passed from the servlet:
<ol>
<li>Hello <%= request.getAttribute( "talk_hello" ) %>
<li>Hello <%= s1 %>
</ol>
version 1
</body>
</html>
```

In the working directory create an HTML file: filename talk-readme.html.

```
<head>
<title>testing</title>
</head>
<body>
talk-readme.html file
</body>
</html>
```

Compile the Servlet

The command to compile a servlet for the Sun ONE Application Server is the same as the command for the Sun ONE Web Server.

```
javac -g -classpath.;\apps\java\jdk\lib\classes.zip;d:
  \iplanet\iws/bin/https/jar/servlets.jar talk.java
```

Here are some notes regarding servlet compilation:

1. In UNIX

   ```
   javac -g -classpath.:/usr/java1.1/lib/classes.zip:
   /iplanet/iws/bin/https/jar/servlets.jar talk.java
   ```

The jar file servlets.ja*r* is installed with the Sun ONE Web Server.

The file classes.zip was installed with the Java JDK.

The JDK was installed with the operating system (e.g., Solaris 8, /usr/java1.2). If you don't have it, it can be downloaded from the Sun web site (*http://java.sun.com*), or you can use the JDK that was installed with the Sun ONE Application Server. Here is where you can look for the JDK from the application server install:

```
<install directory>/ias6/ias/usr/java/
/iplanet/ias6/ias/usr/java/
/iplanet/ias6/ias/usr/java/lib/classes.zip
```

1. In Windows:

   ```
   javac -g -classpath.;\apps\java\jdk\lib\classes.zip;
   d:\iplanet\wsserver4/bin/https/jar/servlets.jar talk.java
   ```

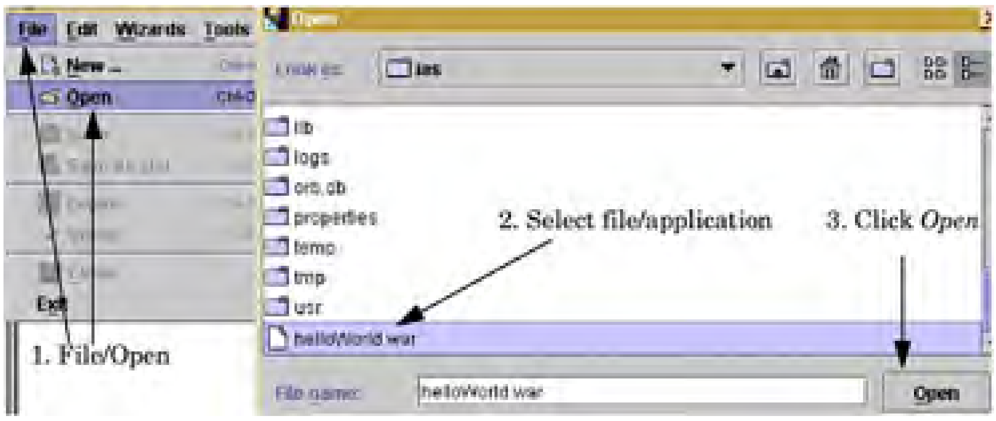
Compile the talk.java servlet now.

Add the Servlet Files into the Hello World Application

The next step is to update the application WAR file and deploy the application.

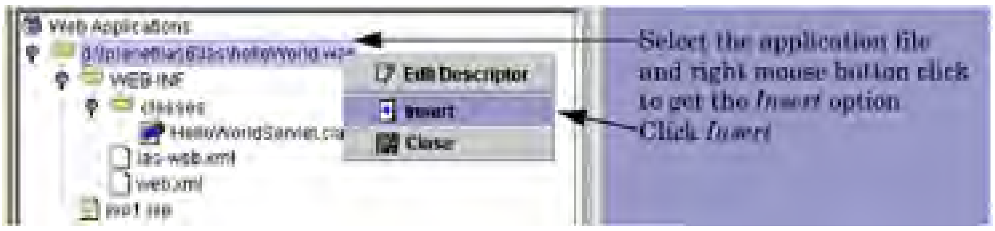
Start the Deployment Tool and click on *File/Open*. Then select the *helloWorld.war* file and click *Open*. These steps are in Figure 22.20.

Figure 22.20. Opening helloWorld.war file.



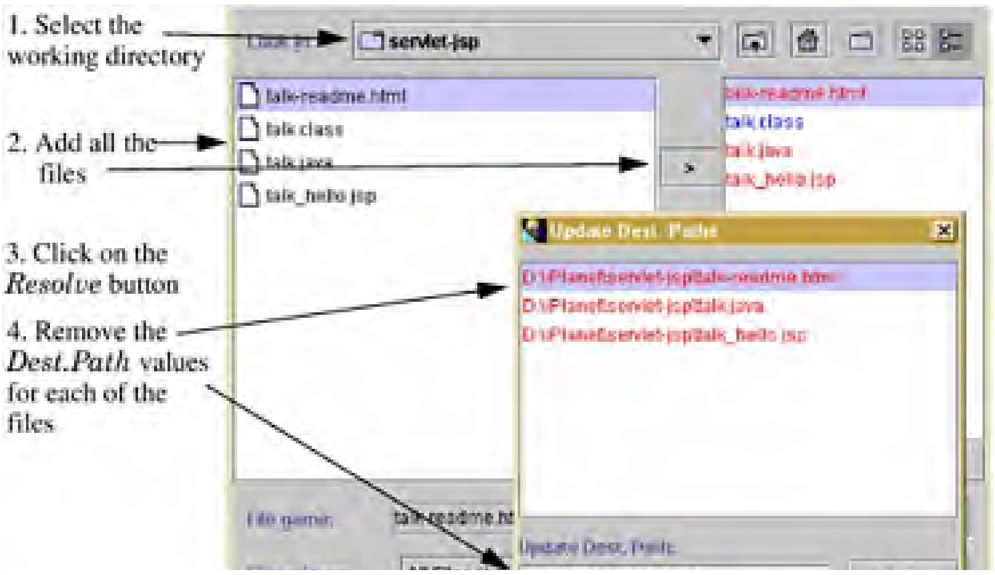
Insert the new files into the application WAR file by clicking on the destination file *d:\iplanet\ias6\ias\helloWorld.war* to select it and right clicking to get the *Insert* option (Figure 22.21). Click *Insert*.

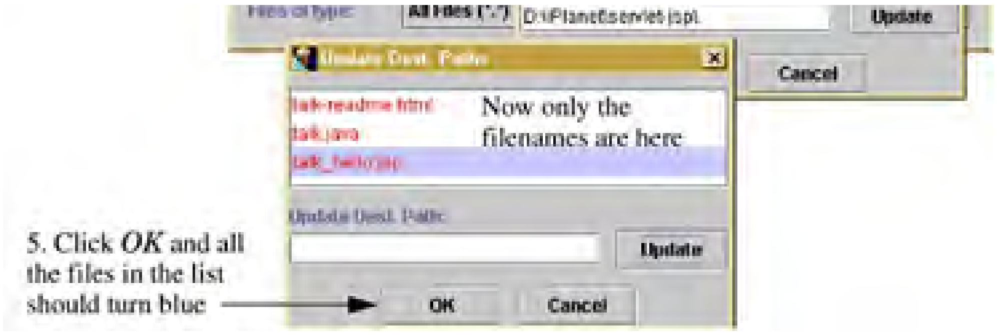
Figure 22.21. Choosing the application where files will be inserted.


Web Applications
d:\iplanet\ias6\ias\helloWorld.war
WEB-INF
classes
HelloWorldServlet.cla
ias-web.xml
web.xml
post.jsp
Edit Descriptor
Insert
Close
Select the application file and right mouse button click to get the *Insert* option
Click *Insert*

Now you are ready to update the destination paths of the files to be inserted into the destination application. First, select the working directory *servlet.jsp* (step 1 in Figure 22.22). Add all the files in the left-hand side of the window by selecting them and then clicking on the arrow pointing to the right-hand side (step 2 in Figure 22.22). The files will appear on the right side of the arrow. Do this for each file (or select all the files and move them together). Then click *Resolve* (step 3, not visible here, but *Insert* window is visible in Figure 22.12).

Figure 22.22. Updating destination paths.

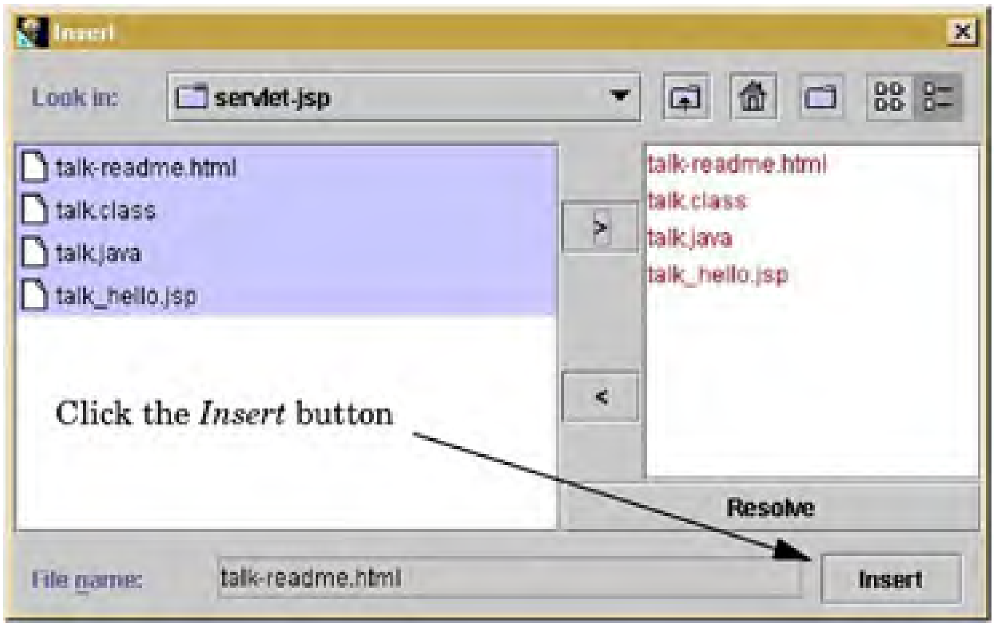

1. Select the working directory
Look in: servlet-jsp
talk-readme.html
talk.class
talk.java
talk_hello.jsp
>
talk-readme.html
talk.class
talk.java
talk_hello.jsp
2. Add all the files
3. Click on the *Resolve* button
4. Remove the *Dest.Path* values for each of the files
Update Dest. Paths
D:\iPlanet\servlet-jsp\talk-readme.html
D:\iPlanet\servlet-jsp\talk.java
D:\iPlanet\servlet-jsp\talk_hello.jsp
File name: talk-readme.ht
Update Dest. Path


Update Dest. Paths
talk-readme.html
talk.java
talk_hello.jsp
Now only the filenames are here
Update Dest. Path:
Update
5. Click OK and all the files in the list should turn blue
OK
Cancel

Now the *Update Dest. Paths* window opens. For each of the files, remove the destination path value. Click on the full filename, remove the *Dest. Path* value, and click *Update* (step 4 in Figure 22.22. The result of the update is the *Update Dest. Path* window with filenames only (nothing in the *Path* field; step 5 in Figure 22.22). Finally, click *OK*. On your screen, all of the files in the list should turn blue.

Now you are ready to insert the new files into the application WAR file. To do this, click *Insert* in the *Insert* window (Figure 22.23).

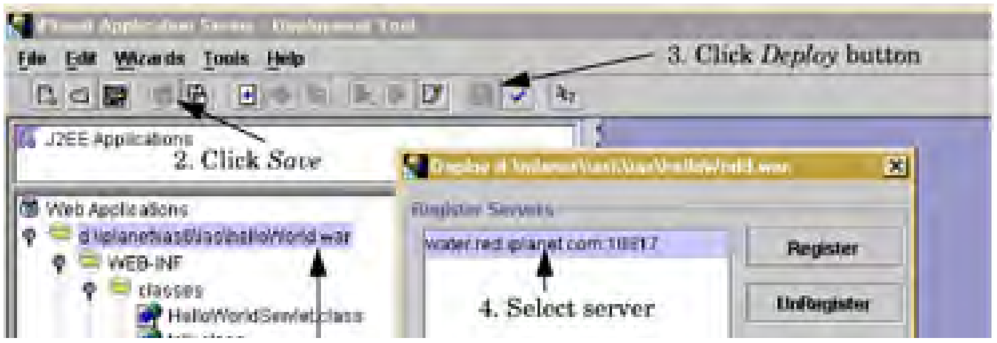
Figure 22.23. Inserting the new files.

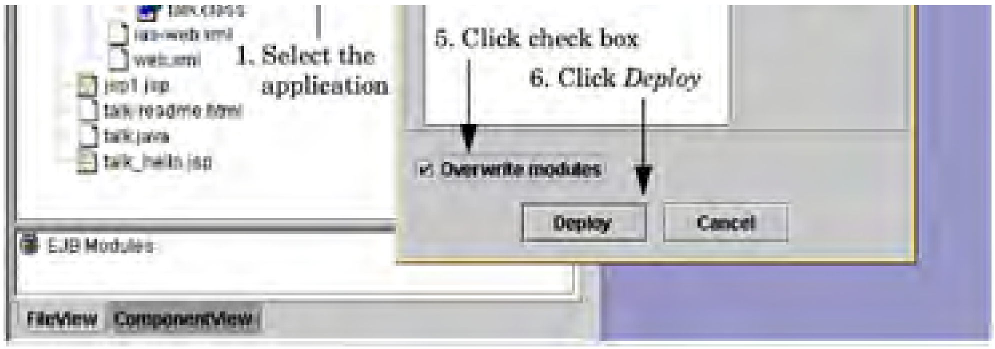


Redeploy Hello World

The new files you inserted in the last section now appear under *Web Applications* in the Deployment Tool main page (Figure 22.24). You need to save the changes you made to the application before redeploying it. First select the application under *Web Applications*. Then click the Save button on the toolbar.

Figure 22.24. Redeploying the updated application WAR file.



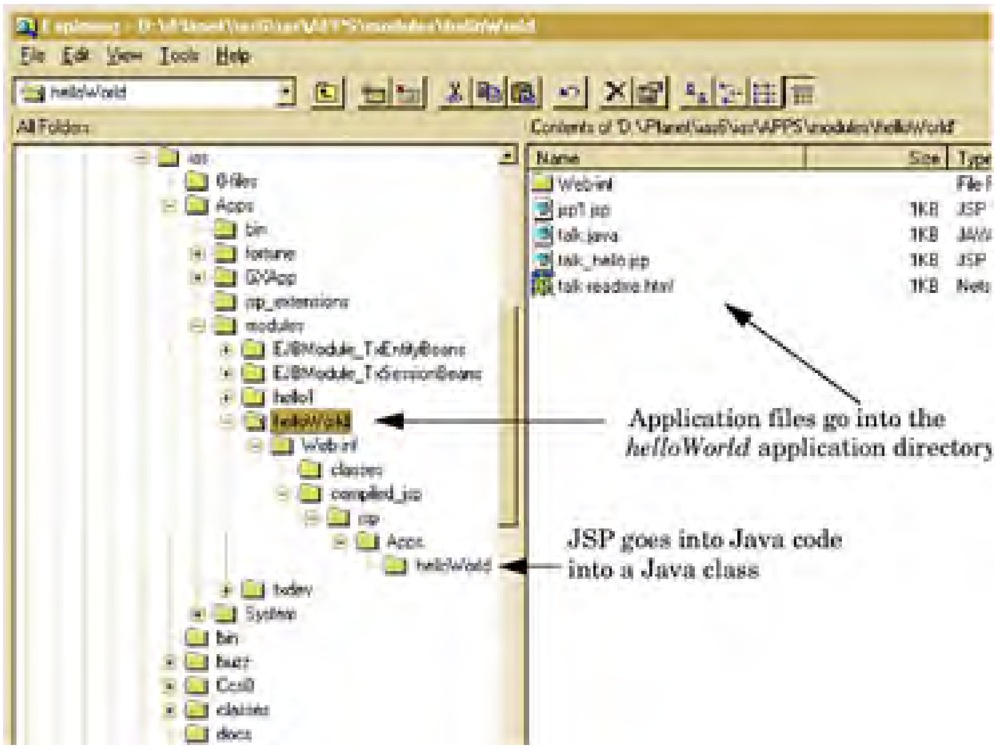


Finally you are ready to deploy the updated application WAR file. (Note: The application server *must* be running for the deployment step.) First, click the Deploy button on the toolbar (see Figure 22.24).

In the *Deploy d:\iplanet\ias6\ias\helloWorld.war* window that opens, select the *water.red.iplanet.com: 10817* server. Then click on the check box to select *Overwrite modules*, and click *Deploy*.

When you have redeployed the application WAR file, have a look at where each of the files ends up (see Figure 22.25, for example).

Figure 22.25. Exploring the application WAR file.



Test

Notes Regarding Testing

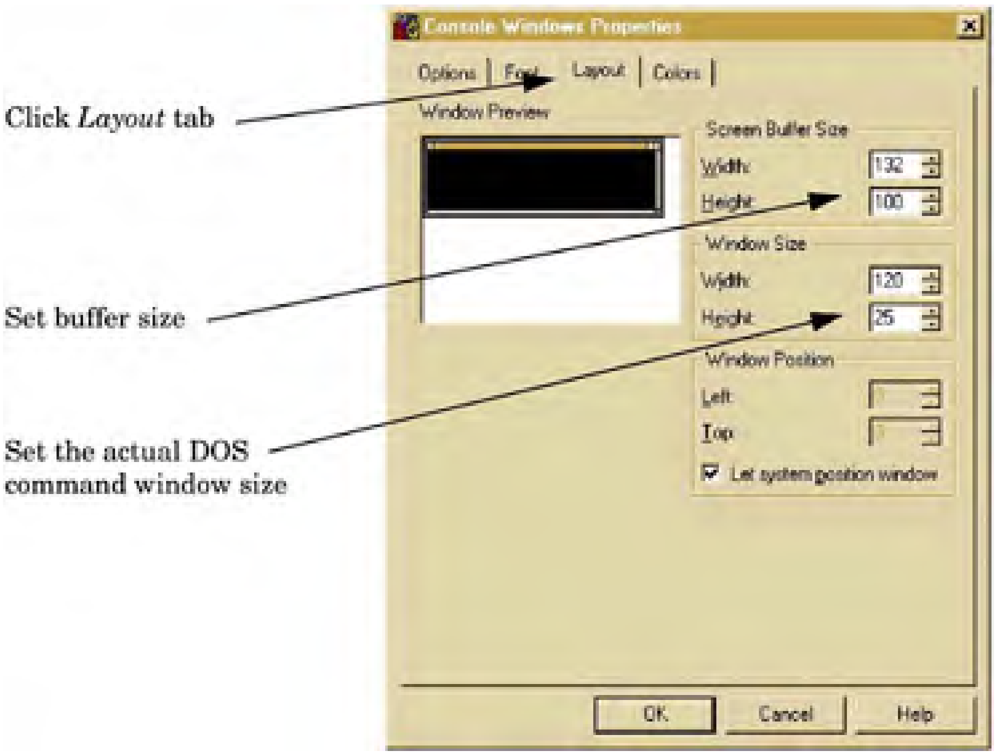
You may need to restart the application server to get the talk servlet to work.

When testing a program on the application server, if debugging is required, check the KJS log files.

- On UNIX, the command is `tail -f /iplanet/ias6/ias/logs/kjs*`.
- On Windows, the KJS log file will show up in a Windows command window. To make a DOS command window more useful, from the Control Panel options (Start/Settings/Control Panel), select Console. When the *Console Window Properties* window opens (Figure 22.26), click the *Layout* tab. Next, set the buffer size (132W x 100H)

and then set the actual DOS command window size (120W x 25W). This will give the DOS command window features that are similar to a UNIX terminal screen.

Figure 22.26. Setting buffer size and DOS command window size.



Screen Buffer Size
Width: 132
Height: 100
Window Size
Width: 120
Height: 25
Window Position
Left:
Top:
☑ Let system position window

Restart the application server, and a DOS command window will come up for each of the server processes, i.e., KAS, KXS, and KJS (depending on your application server process configuration settings). The KJS process is the process that needs to be monitored for debugging purposes.

An alternative is to start the KXS and KJS processes from a batch file. A sample run-kxs.bat file follows:

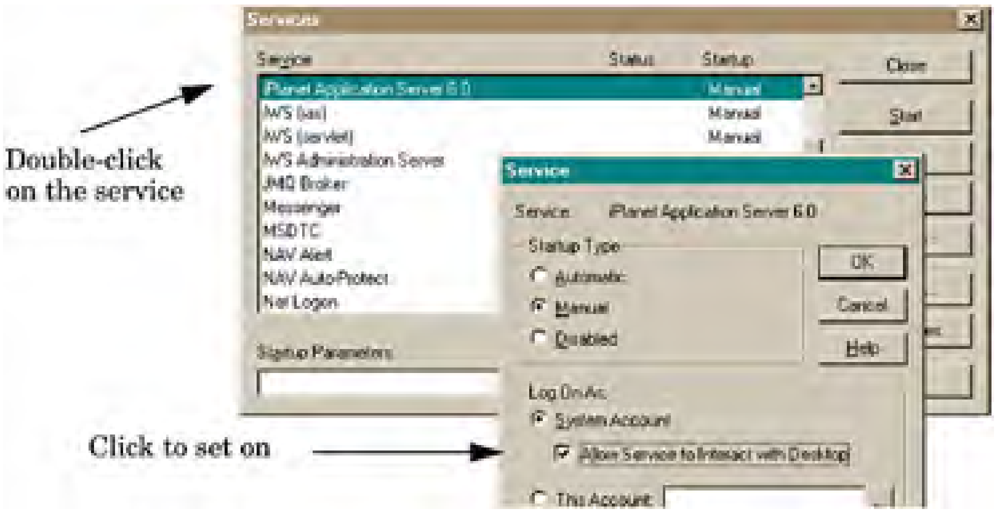
```
@echo off
echo run kxs
echo ...
d:\iplanet\ias6\ias\bin\kxs

Here is a sample run-kjs.bat file:
@echo off
echo run kjs
echo ...
d:\iplanet\ias6\ias\bin\kjs
```

Start KXS (from Windows File Explorer, double-click on *run-kxs.bat*); wait for the process to start running. Then start the KJS process.

Viewing Application Server Log Messages

To turn screen logging on in Windows NT, go to the Services window, as in Figure 22.27, Start/Control Panel/Services. Double-click on *iPlanet Application Server*, and click on *Allow Service to Interact with Desktop*.

Figure 22.27. Allowing services to interact with desktop.

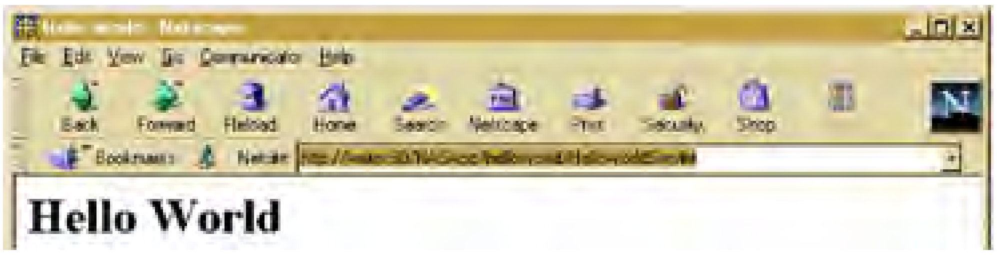


Testing the Servlets that Have Been Deployed

Test each of the files that have been deployed.

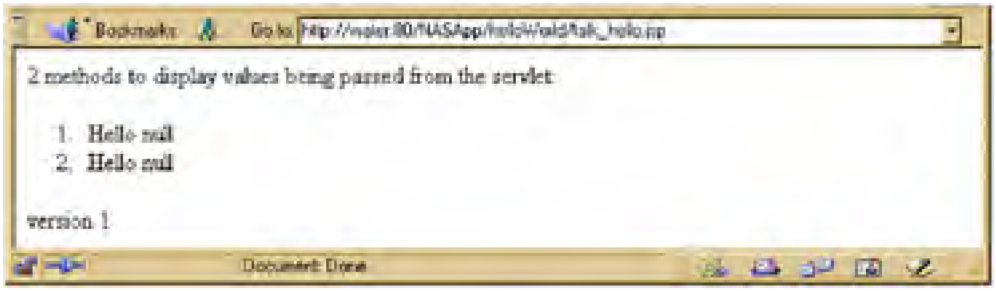
To make sure the previous servlet is still working, use URL *http://water.red.iplanet.com/NASApp/helloWorld/HelloWorldServlet*. If it's working, the Hello World home page will open (Figure 22.28).

Figure 22.28. Hello World home page shows servlet still working.


Hello World

Call the JSP *http://water.red.iplanet.com/NASapp/helloWorld/talk_hello.jsp* (the window in Figure 22.29 will open). The variables for the JSP have the value *null* because the talk servlet has not passed any data to this JSP.

Figure 22.29. The JSP variables are null.


http://water:80/NASApp/helloWorld/talk_hello.jsp

2 methods to display values being passed from the servlet

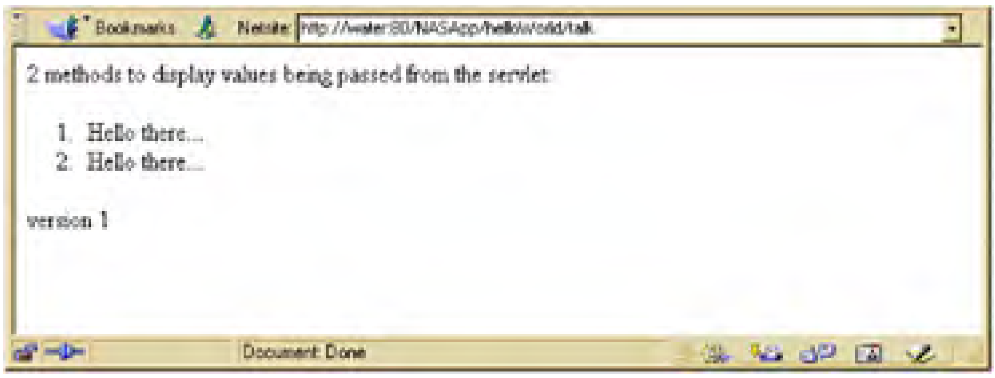
1. Hello null
2. Hello null

version 1

Document: Done

Test the new servlet class by using the URL *http://water.red.iplanet.com/NASApp/helloWorld/talk*. On the web page in Figure 22.30, the variables are set by the talk servlet to *there....*

Figure 22.30. Testing the new servlet.

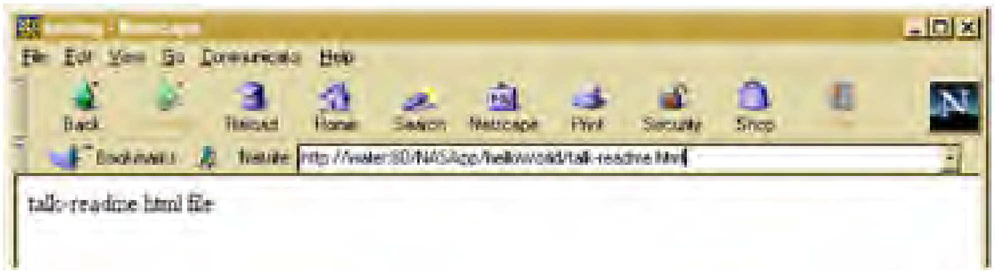

2 methods to display values being passed from the servlet

1. Hello there...
2. Hello there...

version 1

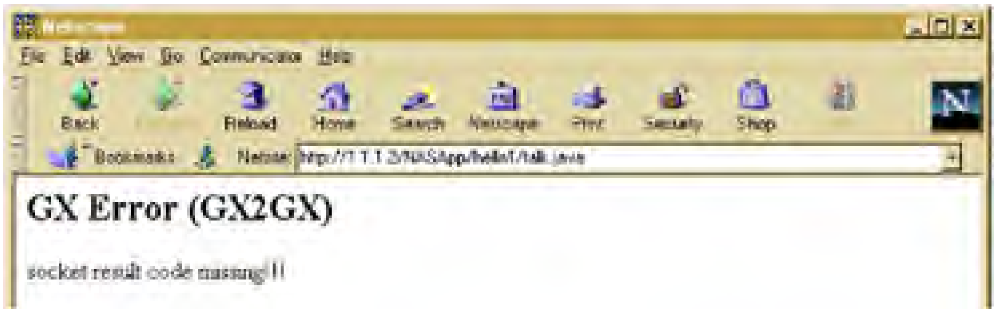
Now test viewing the HTML file using URL *http://water.red.iplanet.com/NASApp/helloWorld/talk-readme.html*. If the window in Figure 22.31 opens, you have a success.

Figure 22.31. Testing viewing the HTML file.


talk-readme html file

The Sun ONE Application Server handles some filename extensions but not others. As we've seen, .class files work and .html files work; however, Java program files with the .java file extension do not work (Figure 22.32). Nor do files with the .txt extension.

Figure 22.32. An incompatible file extension test result.


GX Error (GX2GX)

socket result code missing!!!

Even though the application server does not serve files with extension .java, it may be still desirable to include the source files anyway. This keeps the source with the binary file.

Development Note

When developing a servlet or a JSP, I worked right in the application server application file directory. For example, I first created the initial program files; then I created the application WAR file; then I deployed the application file. When the application file was deployed, I went to the application file directory and continued development from there. An example directory:

```
d:\iplanet\ias6\ias\APPS\modules\helloWorld\WEB-
  INF\classes
```

or a file directory for the JSPs:

```
d:\iplanet\ias6\ias\APPS\modules\helloWorld
```

On Windows, I created batch files to do the compilations. Here is a sample batch file that I created to compile the servlet on my WinNT computer and to copy the file(s) into the application live file directories:

```
@echo off
echo Compile talk.java
echo ...
javac -g -classpath
  .;\apps\java\jdk\lib\classes.zip;d:\iplanet\wsserver4/
  bin/https/jar/servlets.jar talk.java
echo.
echo Next, copy the file to the servlet directory.
pause
copy talk.class d:\iplanet\ias6\ias\APPS\modules\hel-
  loWorld\WEB-INF\classes
copy talk_hello.jsp d:\iplanet\ias6\ias\APPS\mod-
  ules\helloWorld
echo.
echo Next, test.
pause
```

I also created an index.html web page that has all my test URLs. I put this file in the document directory of the application server web instance, e.g., /iplanet/wsserver4/doc/index.html. This helped me remember all my test links. It is also good documentation for other people who are helping to test. I also used the index.html page when demonstrating applications. Here is a sample index.html that works for all the URLs we just tested.

```
<html>
<head><title>App.Server Testing</title></head>
<body>
<h1>App.Server Testing</h1>
<ol>
<li><a href="http://water.red.iplanet.com/NASApp/for-
  tune/fortune">Fortune</a>
<p>
<li><a href="http://water.red.iplanet.com/NASApp/hel-
  loWorld/helloServlet">helloServlet</a>
<li><a href="http://water.red.iplanet.com/NASApp/hel-
  loWorld/talk">talk - hello</a>
<li><a href="http://water.red.iplanet.com/NASApp/hel-
  loWorld/talk">talk - readme</a>
</ol>
</body>
</html>
```

Ultimately, however, it's up to you to decide how to develop your own style and way of doing things.

23. Epilogue

If you have finished the book and done the exercises included in it, then you have used Sun Microsystems software, seen the architectures, worked with the products, and gotten started using Sun ONE!

Powerful and Free

I talked about this in chapter 2, and I would like to mention again, Sun Microsystems is offering powerful, free software with the new product releases of the Sun ONE Studio and the Sun ONE Application Server 7. The Community Edition of the Sun ONE Studio 4 is free of charge, and the Platform Edition of Sun ONE Application Server 7 is free. This is an incredible combination of free and powerful software.

It is possible to set up a computer on which all of the software is totally free. First, you develop applications using the Sun ONE Studio 4 Community Edition. Then you run the applications on the Platform Edition of the Sun ONE Application Server 7. Run the Sun ONE Application Server 7 on Linux or the Sun Solaris operating environment—both of these operating systems are available for free as well! Your only cost is hardware and, when you choose, technical support and training.

Now It Is Time to Do Some Real Work

The goal of the book was to get you started using Internet servers and creating web sites and web site applications. Now that you have started, I hope you will continue, becoming confident with the products by working with them in some form of production environment. By production environment, I mean creating web sites for the Internet or setting up a web site for your corporate intranet. For example, I have created my own web site: *www.internetflow.com*.

To make the *www.internetflow.com* web site, I used many concepts from this book. Here are some examples:

1. I installed the Sun ONE Web Server to create a development environment. This environment mirrors the CGI environment of my ISP.
2. When designing the hierarchical organization structures used to maintain the web site database, I used directory structure concepts from the Sun ONE Directory Server.
3. To display the data in an HTML format, I wrote a CGI system with JSP features.

The thing I like best about *www.internetflow.com* is its usefulness. I use it every day to go to my favorite web sites, and I use it to store information such as a list of UNIX commands to help when I am working. The *www.internetflow.com* web site uses Sun ONE concepts.

Be Confident

Internet servers are like any other software products. To become confident using them, you need to:

1. Install the software.
2. Test the software.
3. Remove the software.
4. Go back to step 1 and repeat steps 1–3 five times.
5. Get some new computers with different operating system versions. Go back to step 1 for each operating system.
6. Do some production work with the servers—some *real* work, i.e., make a web application; for example, *www.internetflow.com*.
7. Get feedback from people that are using the web site and revise your web site.
8. Now, if you are up to it, go back to step 1 and repeat the whole thing ten times.
9. Then write a book.

Repetition is one sure way to become confident and successful with software. As a Netscape/AOL/iPlanet/Sun ONE trainer, I taught courses all over the world. Before I left to teach a course, I prepared myself because I wanted to be successful. To get prepared, I installed the products over and over again on a number of different computers. This way, when I traveled to the training location, I was ready for almost anything.

On one occasion, I flew to Singapore to teach a course and, when I got there, the only computers available to me had Windows NT OS. Unfortunately, the course material and most of the software I had was for Sun Solaris UNIX. The students I was scheduled to train were from India, Malaysia, and Singapore, and I did not want to let them down. Since I have worked with many operating systems, I was confident I could do this. From the Internet, I downloaded products that would run on Windows NT; I talked to my tech support team in the U.S. and Australia. Then I tested the setup. While teaching, I modified the instructions for the different operating systems, and everything worked—we had a very successful class and everybody left happy.

Install, test, remove, get new computers, install, test, remove, install...

Be prepared, be confident, be successful.

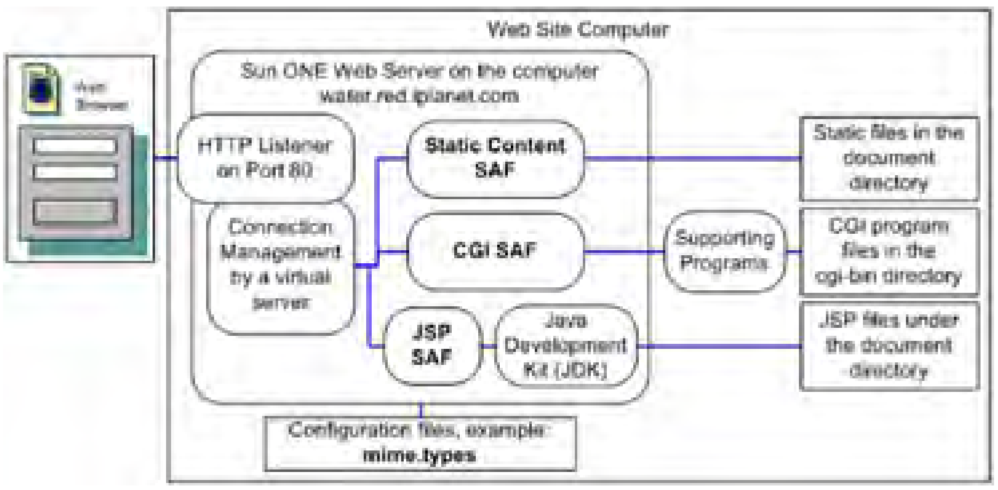
Tasks Completed in This Book

This section will review some of the exercises carried out in this book. If you have worked along with the exercises, you have accomplished some of these tasks.

Setting Up a Web Site

Both the Sun ONE Web Server and the Sun ONE Application Server 7 have features enabling them to host a web site with static content and dynamic content. Dynamic content is created using CGI programs and Java Server Pages (JSPs) (illustrated in Figure E.1). Configuring the web or application server for dynamic content is a very straightforward process:

Figure E.1. Sun ONE Application Server web server application functions (SAFs).

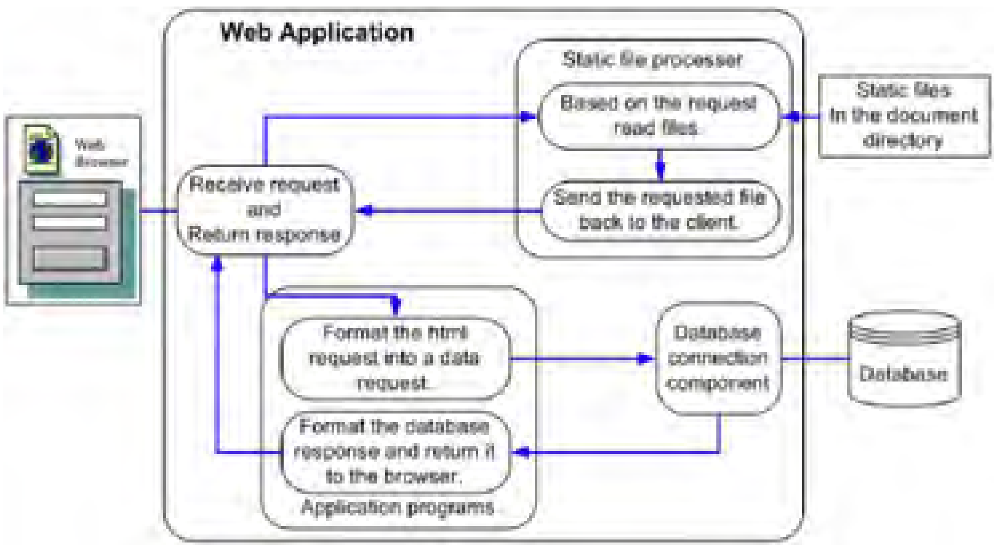

Web Site Computer
Web Browser
Sun ONE Web Server on the computer water.red.iplanet.com
HTTP Listener on Port 80
Connection Management by a virtual server
Static Content SAF
CGI SAF
JSP SAF
Java Development Kit (JDK)
Supporting Programs
Static files in the document directory
CGI program files in the cgi-bin directory
JSP files under the document directory
Configuration files, example: mime.types

1. Configure a program directory using the administration system.
2. Make a file directory to store programs on the server computer. This file directory is the program directory configured in step 1.
3. Write a program in the program directory, or write the program somewhere else and load or copy the program into the program directory.
4. If it is a CGI program on UNIX, make the program file executable.
5. Test the program by using a web browser.

Web Applications

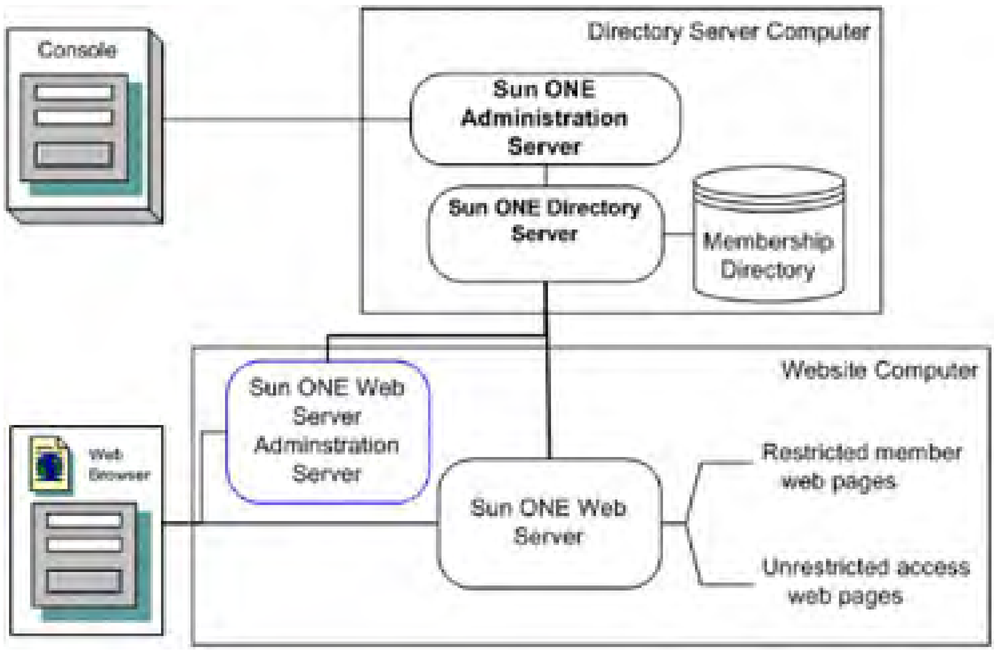
Web applications are extremely common on the Internet. Examples of web site applications include search engines, shopping applications, and web mail (doing email from a browser). Browser clients communicate with web applications using the HyperText Transfer Protocol (HTTP). The applications respond by returning HyperText Markup Language (HTML) files, pictures, and other types of media files as illustrated in Figure E.2. Web applications use the open standards of HTTP and HTML, and they work over the most open network in the world—the Internet.

Figure E.2. Structure of a web application.


Web Application
Static file processer
Based on the request read files.
Send the requested file back to the client.
Static files In the document directory
Web Browser
Receive request and Return response
Format the html request into a data request.
Format the database response and return it to the browser.
Application programs
Database connection component
Database

Web applications are web sites whose prime function is to format data into HTML web pages. When moving from a simple web site to a web application, it is common to add a directory server to control user access to the application. The directory stores member information such as user ID, password, email address, and other contact information. Another type of data stored in the directory is access control information; for example, which particular users are allowed to access which parts of the web site application. Figure E.3 shows a Sun ONE Directory Server connected to a Sun ONE Web Server. This configuration is also possible with the Standard Edition of the Sun ONE Application Server 7.

Figure E.3. Connecting the Sun ONE Directory Server to the Sun ONE Web Server.


Console
Directory Server Computer
Sun ONE Administration Server
Sun ONE Directory Server
Membership Directory
Website Computer
Sun ONE Web Server Adminstration Server
Web Browser
Sun ONE Web Server
Restricted member web pages
Unrestricted access web pages

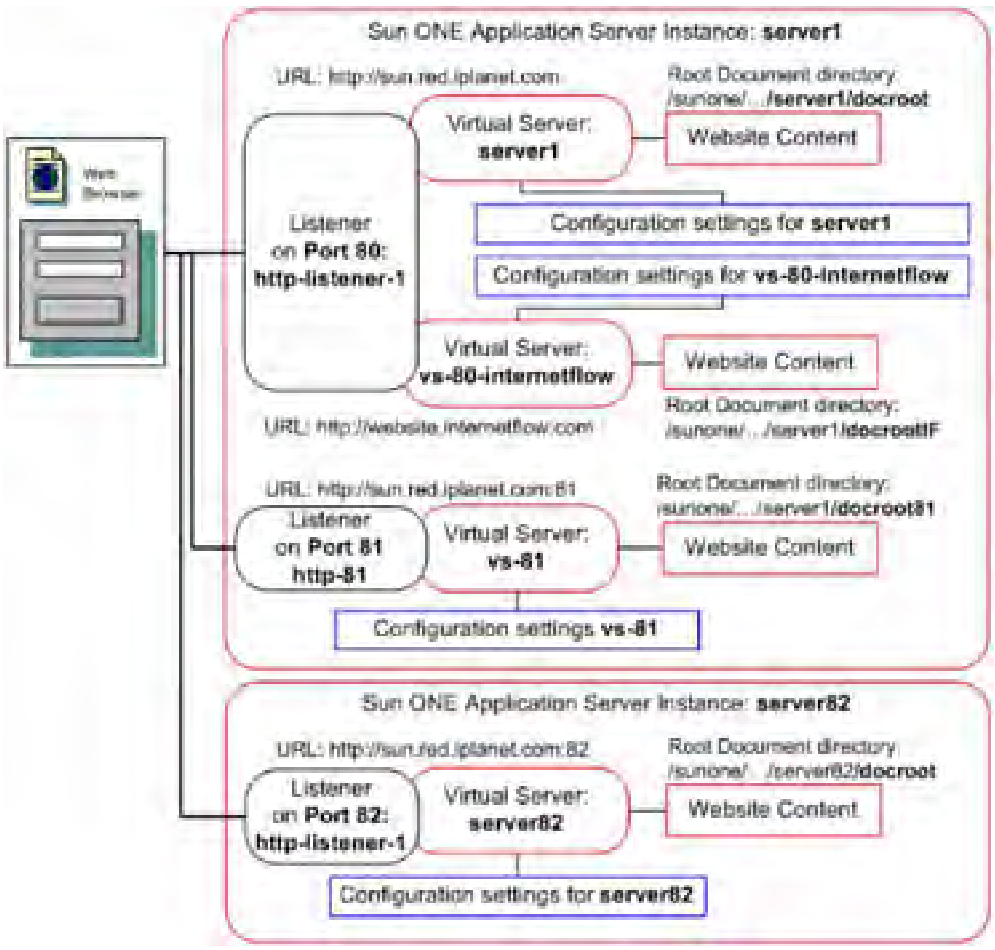
In Figure E.3, member information is maintained by the Sun ONE Web Server Administration Server or by the Sun ONE Administration Server Console. When member data has been added into the membership directory, it is used by the Sun ONE Web Server to control access to the restricted member web pages.

Another method of controlling access to areas of the web application is to have multiple web sites and then to separate application functionality across the web sites. When doing this, each of the web sites would be connected to the same directory server system, so that you have multiple web sites and one central directory of member information.

Multiple Web Sites

The Sun ONE Application Server 7's web site system architecture is amazingly easy to configure. Figure E.3 is a diagram of the configurations that have been demonstrated in this book. In Figure E.4, four web sites are set up under one Sun ONE Application Server administration system (the Sun ONE Web Server has the same abilities of multiple servers, instances, and virtual servers).

Figure E.4. The instances, listeners, and virtual servers from this book.

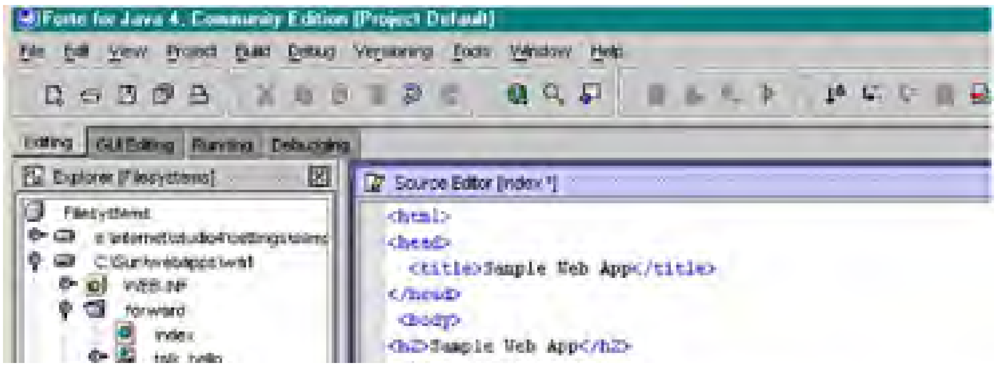

Web Browser
Sun ONE Application Server Instance: **server1**
URL: http://sun.red.iplanet.com
Root Document directory /sunone/.../server1/docroot
Listener on Port 80: http-listener-1
Virtual Server: server1
Website Content
Configuration settings for server1
Configuration settings for vs-80-internetflow
Virtual Server: vs-80-internetflow
Website Content
Root Document directory: /sunone/.../server1/docrootIF
URL: http://website.internetflow.com
URL: http://sun.red.iplanet.com:81
Root Document directory: /sunone/.../server1/docroot81
Listener on Port 81 http-81
Virtual Server: vs-81
Website Content
Configuration settings vs-81
Sun ONE Application Server Instance: **server82**
URL: http://sun.red.iplanet.com:82
Root Document directory /sunone/.../server82/docroot
Listener on Port 82: http-listener-1
Virtual Server: server82
Website Content
Configuration settings for server82

Whether you have a single web site application or multiple web site applications that span across multiple locations, you need to have an application that allows you to create and package applications. The Sun ONE Studio is such an application.

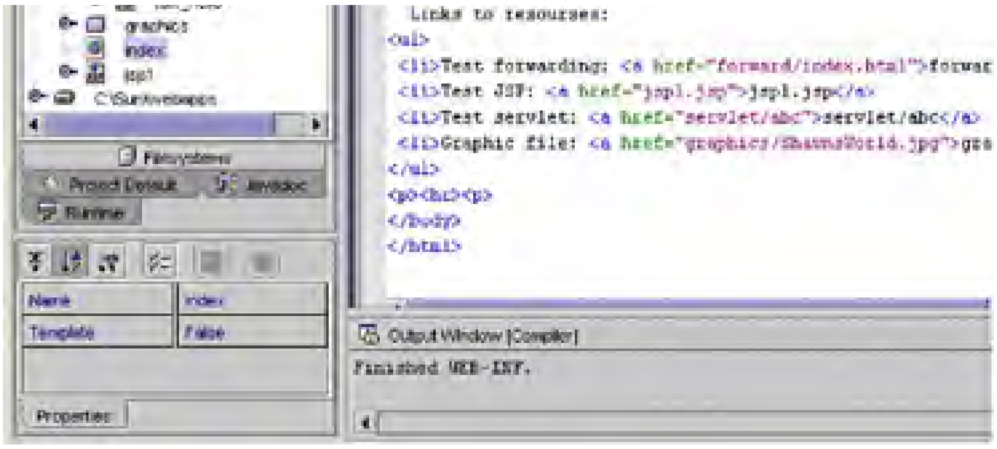
The Sun ONE Studio

Sun ONE Studio is a comprehensive tool for developing applications. Figure E.5 is a screen shot of the Community Edition of Sun ONE Studio. It has the *Explorer* subwindows for organizing files, the *Source Editor* window for editing files, an output window for viewing compilation messages, and many of the windows that are accessible through tabs. The Community Edition is available for free from the Sun web site: *www.sun.com*.

Figure E.5. Sun ONE Studio, an application to create applications.


Forte for Java 4, Community Edition [Project Default]
File Edit View Project Build Debug Versioning Tools Window Help
Editing GUI Editing Running Debugging
Explorer [Filesystems]
Filesystems
WEB-INF
forward
index
Source Editor [index *]

```
<html>
<head>
  <title>Sample Web App</title>
</head>
 <body>
<h2>Sample Web App</h2>
```

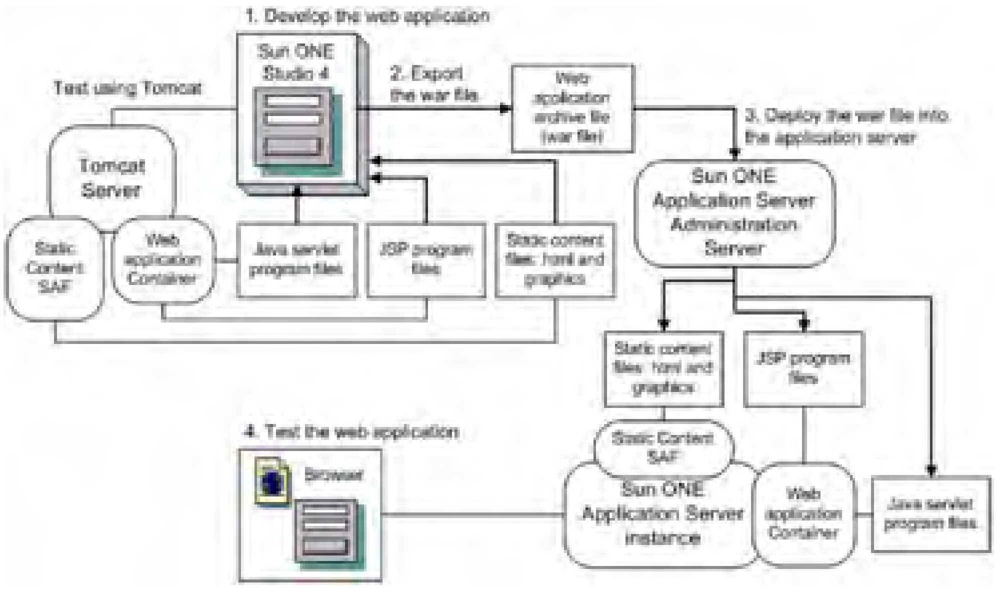


```
 Links to resources:
<ul>
 <li>Test forwarding: <a href="forward/index.html">forwar
 <li>Test JSP: <a href="jsp1.jsp">jsp1.jsp</a>
 <li>Test servlet: <a href="servlet/abc">servlet/abc</a>
 <li>Graphic file: <a href="graphics/ShawnsWorld.jpg">gra
</ul>
<p><hr><p>
</body>
</html>
```

```
Finished WEB-INF.
```

Sun ONE Studio is the center of the application development cycle as illustrated in Figure E.6. Sun ONE Studio organizes all the files in an application. Since the Tomcat system is installed with Sun ONE Studio, it makes testing an application a straightforward process—create an application, then select a menu option, and the application runs under Tomcat. To test the application, use a browser to call the application as it runs on Tomcat.

Figure E.6. Web application development using Sun ONE Studio.


1. Develop the web application
Sun ONE Studio 4
2. Export the war file
Web application archive file (war file)
3. Deploy the war file into the application server
Test using Tomcat
Tomcat Server
Static Content SAF
Web application Container
Java servlet program files
JSP program files
Static content files: html and graphics
Sun ONE Application Server Administration Server
Static content files: html and graphics
JSP program files
4. Test the web application
Browser
Static Content SAF
Sun ONE Application Server instance
Web application Container
Java servlet program files

When an application is ready to be deployed to an application server, Studio packages up all the web application files into a single WAR file. This is an excellent distribution system: you develop the application and then package it into a single file. This single file can be deployed to any application servers because the WAR file is in standard format that is used by all application servers regardless of the vendor. This is all part of the open standards initiatives by Sun Microsystems.

Here are the steps needed to create a web application using Sun ONE Studio:

1. From Sun ONE Studio's *Explorer* window, mount (add) a file directory to use as a storage area for project files.
2. Add web site component files into this application file directory. Types of application files that may be included with a web application are HTML web pages, graphics, sound files, and text files.
3. Before adding servlets, a web module structure is added into the file directory. Then servlets are added.
4. Save all files that have been edited.
5. Build the application. This step compiles any program files that need compiling (e.g., servlet programs are compiled into class files).
6. Package the application files into a web application archive (WAR) file.
7. Use a browser to log in to the Sun ONE Application Server Administration Server website, and go to the deployment web page.
8. Select the WAR file you have just created and deploy it.

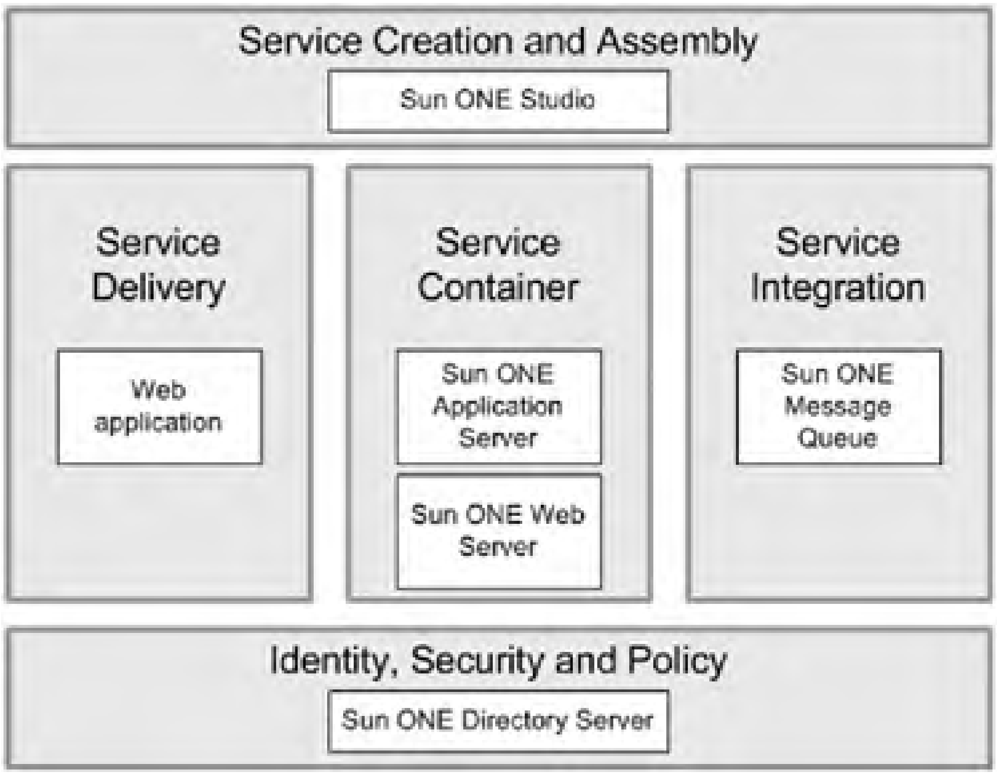
9. Test!

This WAR file can then be distributed and installed to as many application servers as you want. The installation of the WAR file uses a standard method to deploy files into the application server environment. And there you have it—a deployed application.

Summarizing Sun Microsystems Sun ONE Middleware Software

The Sun ONE middleware products discussed in this book fit into the Sun ONE architecture as shown in Figure E.7. The middleware products are as follows:

Figure E.7. Sun ONE server middleware product architecture.



- The Sun ONE Web Server is designed for hosting web sites. The Sun ONE Application Server 7 has many of the same features as the Sun ONE Web Server, and thus it is designed for hosting web sites as well.
- The Sun ONE Directory Server is to manage member data information. The member information is used by other servers or applications. The steps to link the web server and the directory server together to give the web server the ability to control access to web sites are in this book.
- The Sun ONE Studio is used to create and package web applications. It has a complete development environment, including the Tomcat server.
- Web applications are deployed and tested on the Sun ONE Application Server. Also, the Sun ONE Web Server has a web container and is capable of deploying and hosting web applications.
- Although not used in this book, the Sun ONE Message Queue is installed with the Sun ONE Application Server 7. The queue is used to communicate messages between applications.