

Rational Business Developer



Tutorial: Create a hello world program with EGL

Version 7.1

Rational Business Developer



Tutorial: Create a hello world program with EGL

Version 7.1

Note

Before using this information and the product it supports, read the information in "Notices," on page 9.

This edition applies to version 7.1 of Rational Business Developer and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright IBM Corporation 2000, 2008.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Create a hello world program with EGL	1	Appendix. Notices	9
Introduction	1	Trademarks	11
Lesson 1: Create an EGL project	2		
Lesson 2: Create and run an EGL program	5		
Summary	7		

Create a hello world program with EGL

In this tutorial, you will learn how to write and run your first EGL program. Although the program you create will not be powerful or complex, it will demonstrate how an EGL program works.

Learning objectives

In this tutorial, you learn to do the following tasks:

- Work with projects, files, and editors in the workbench
- Create an EGL project and an EGL source file
- Write and generate EGL code
- Run an EGL program

Time required

20 minutes

Introduction

In this tutorial, you will learn how to write and run your first EGL program. Although the program you create will not be powerful or complex, it will demonstrate how an EGL program works.

Learning objectives

In this tutorial, you learn to do the following tasks:

- Work with projects, files, and editors in the workbench
- Create an EGL project and an EGL source file
- Write and generate EGL code
- Run an EGL program

Time required

This tutorial should take approximately 20 minutes to finish. If you explore other concepts related to this tutorial, it could take longer to complete.

Skill level

Introductory

System requirements

- Enterprise Generation Language
- Rational® Software Delivery Platform

Prerequisites

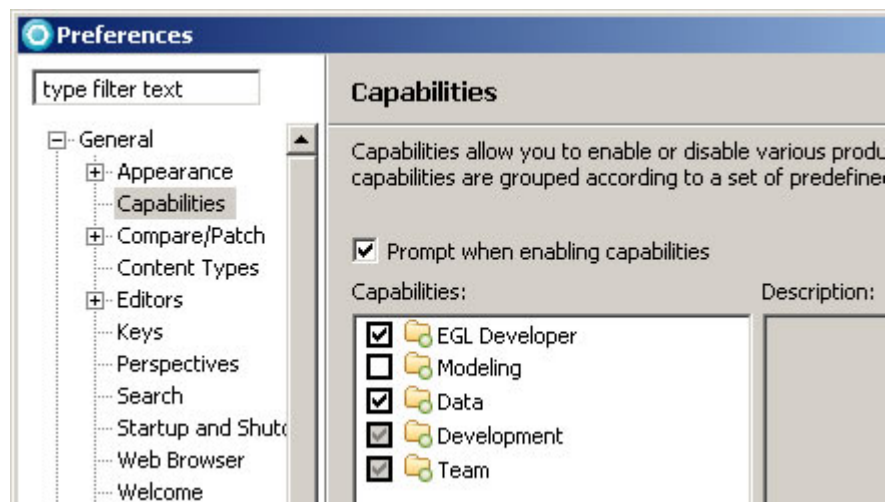
None.

Lesson 1: Create an EGL project

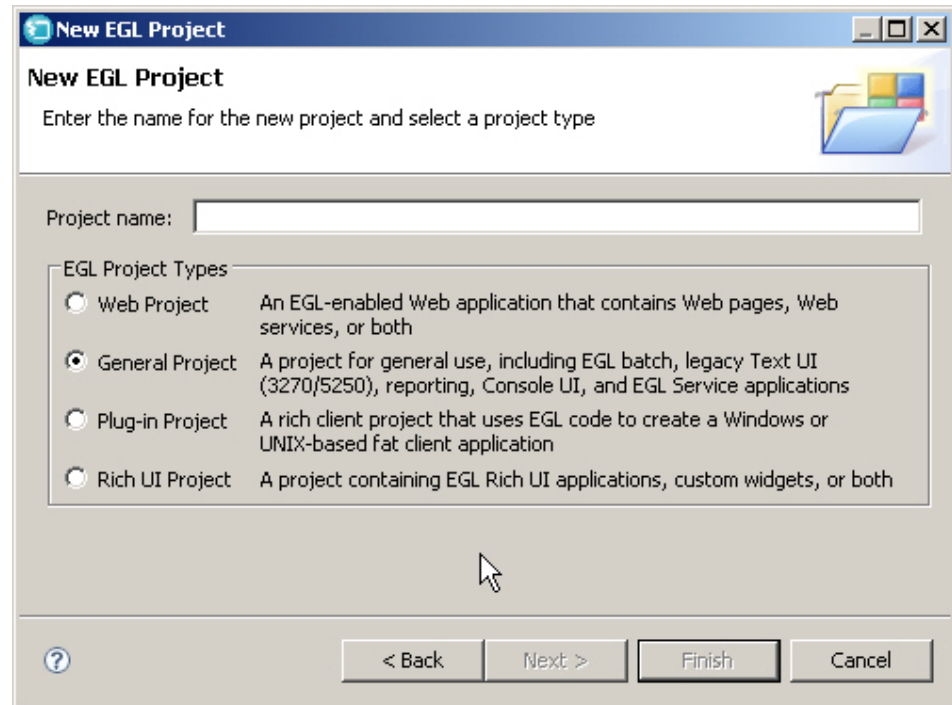
In this lesson, you will enable EGL and create an EGL project to hold your EGL source files.

All code, files, and artifacts in the workbench must be associated with a project. A project works like a top-level folder to hold your files. In this case, you will need an EGL project for your EGL application.

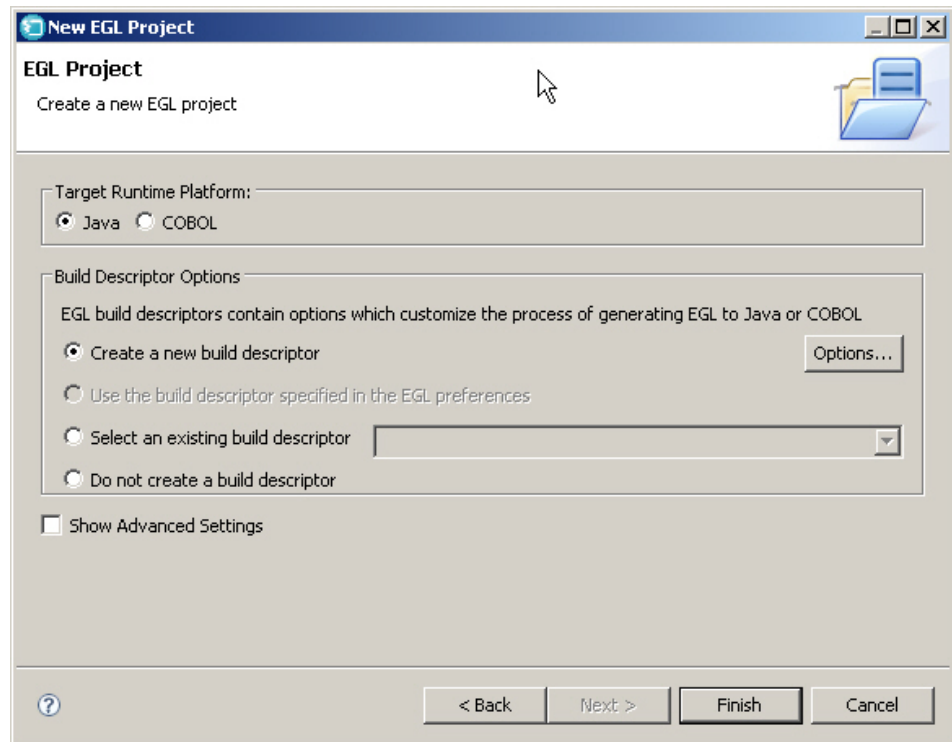
1. In the workbench, click **Window > Preferences**. The Preferences window opens.
2. In the left pane of the Preferences window, click **General > Capabilities**.
3. In the list of capabilities in the right pane, select the **EGL Developer** check box. Enabling this capability tells the workbench that you will be working with EGL. You may have many other capabilities in the list, but for now the **EGL Developer** capability is the only one that matters.



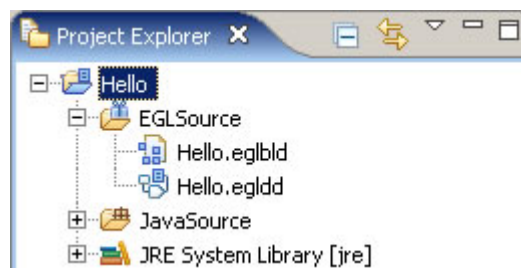
4. Click **OK**.
5. Click **File > New > Project**. The New Project window opens.
6. In the New Project window, expand **EGL** and click **EGL Project**.
7. Click **Next**.
8. In the New EGL Project wizard, assign a **Project name** such as Hello.
9. Click **General Project**. The other types of EGL project enable particular types of user interfaces, but since you don't need a dedicated user interface for this project, the general EGL project is appropriate. The first window of the New EGL Project wizard looks like this:



10. Fill in the project name, and then, click **Next**.
11. Under **Target Runtime Platform**, click **Java**. You can generate EGL code either to Java or to COBOL. You will use Java for this tutorial because the workbench can run Java code directly.
12. Make sure that **Create a new build descriptor** is selected. Build descriptors contain options for generating your program into another language. You do not need to worry about them at this point because the wizard will create an appropriate build descriptor for you. The second window of the New EGL Project wizard looks like this:



13. Click **Finish**.
14. You may see a message window that asks, "This kind of project is usually associated with the EGL Perspective. Do you want to switch to this perspective now?" If you see this window, click **Yes**. The different workbench perspectives display editors, information windows, and tools appropriate to specific tasks. The **Project Explorer** view in the upper left of the default workbench displays a new folder called **Hello**. This is your new EGL project.
15. Expand the **Hello** project by clicking the plus sign next to it and note the folders and files that EGL automatically created for you.



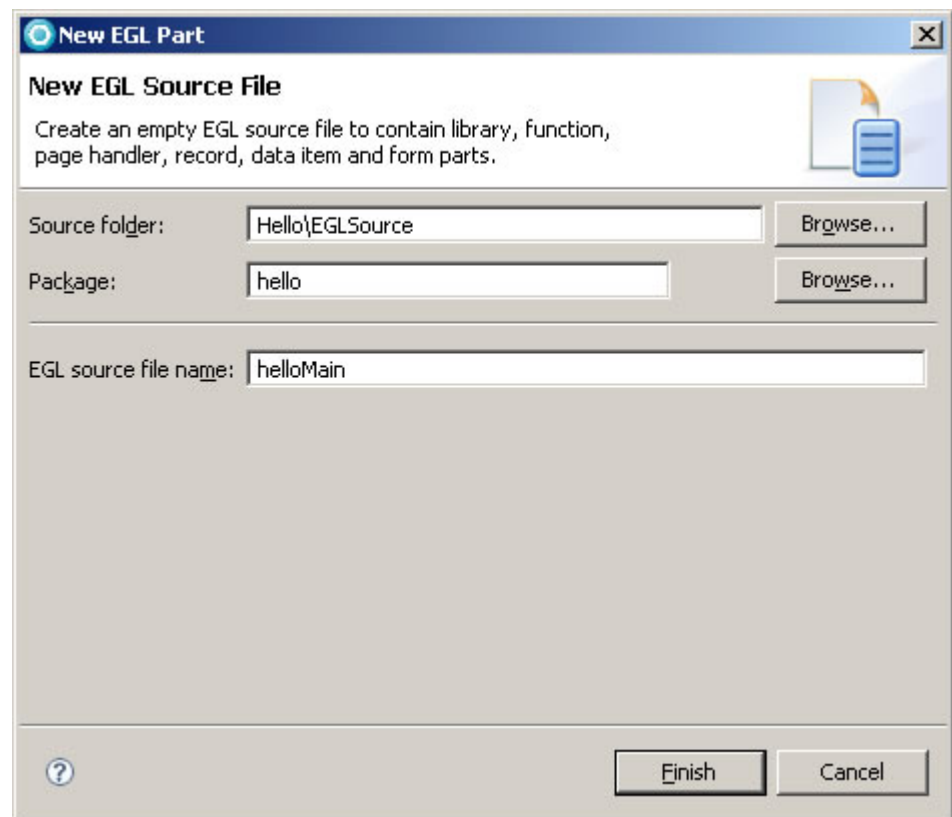
The **EGLSource** folder will contain your EGL code. Right now, it contains an EGL build file named **Hello.eglbld**, which contains an EGL Build Descriptor part. By default, the **EGLSource** folder contains an EGL deployment descriptor file named **Hello.EGLDD**. The deployment descriptor file contains information on deploying your project as a web service and information on web services in other applications that your project will use. Since your project will not contain or use any services, you can ignore this file for now.

Now you have an EGL project. In the next lesson, you will create an EGL program in this project.

Lesson 2: Create and run an EGL program

In this lesson, you will learn how to create and run a simple EGL program.

1. Click your EGL project to select it.
2. Click **File > New > EGL Source File**. You will use the contents of this file to generate a working "Hello World" program. The New EGL Source File window opens, with the **Source folder** already set to Hello\EGLSource.
3. Enter a package name such as hello. A package is both a folder and a unifying concept for a group of files that share resources. If you are only going to have one package in your project, there is no problem in using the same name for both.
4. Enter an **EGL source file name** such as helloMain. EGL will add the file extension .egl for you, because all EGL source files end with that extension. Note also that EGL by convention uses "camel case" for file, function, variable, and other names. This style capitalizes each word in the name except the first, and omits the spaces between words, allowing you to use names that are long enough and complex enough to document their contents.



5. Click **Finish**.
6. The workbench displays the new helloMain.egl file in the EGL editor. Note that it has already included the package name for you, in the line of code package hello;.
7. Highlight the text that says //Put EGL Source File Contents Here and replace it with the following line of code, exactly as it is written here:

```
program helloMain type BasicProgram
```

Moving down the elemental structure of the EGL application, a program falls below a package, representing, in most cases, the solution to a single business problem. EGL has different types of programs, of which BasicProgram is the simplest.

Note also that the name of your program must match the name of the EGL file, minus the .egl extension. If you named your file helloMain.egl, your program must be named helloMain. In most cases, EGL is not case sensitive, but for programs (and other major types of EGL part called *generatable parts*), the case of the program name must match the case of the file name.

8. Leave a blank line, then on a new line (indented, if you want to follow common style), type this code:

```
function main()
```

A program consists of one or more functions, which are comparable to atoms—they are the fundamental building blocks of EGL logic. All function declarations are followed by parentheses, which in some cases hold data passed in or out of the function. Every program must have one and only one `main()` function.

9. On another new line (indented again), type:

```
writeStdOut("Hello, Cleveland!");
```

Here the function performs its actual work. In this case, it is calling another function, one named `writeStdOut()`. This function exists in a library named `sysLib`, a system library of functions that you get free with EGL. You can call its functions from any other EGL function, and most of the time (you will learn about the rare exceptions later) you will not need to explicitly point to the library for EGL to be able to find the function.

The `writeStdOut()` function takes a single argument (the piece of data in the parentheses). In this case the argument is a literal string of characters, and `writeStdOut()` has the job of displaying this string, on a line by itself, wherever EGL thinks the standard output should go. By default, that would be in the workbench's console window. That window is associated with one of the tabs in the lower right of the default workbench screen.

The line of code, which represents a complete thought, ends with a semicolon.

10. On another new line, with the current indentation, type:

```
end
```

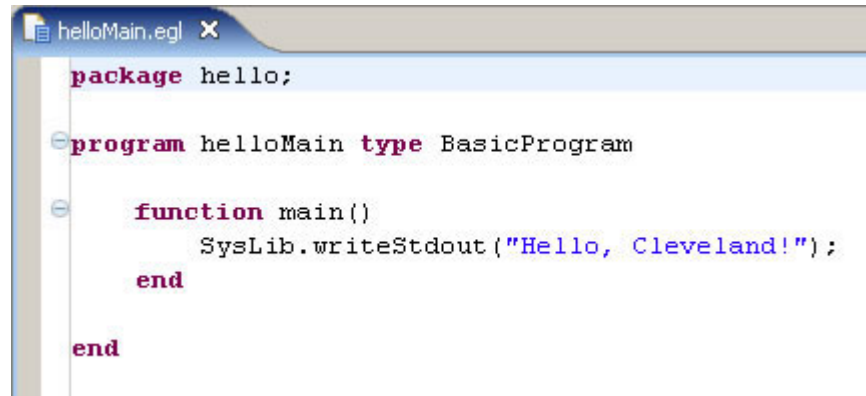
EGL is smart enough to figure out that this `end` statement refers to the `main()` function declaration, so it should match the indentation of that declaration. Other languages use braces or other contrivances to mark blocks of code; EGL, being closer to natural language, simply matches the word `end` with the beginning of a block.

11. Then on another new line, with the current indentation, type:

```
end
```

Again, EGL knows this `end` refers back to the program declaration, so it matches the indentation of that opening declaration.

12. Press **CTRL+S** to save the file. Your finished program should look like the following example:

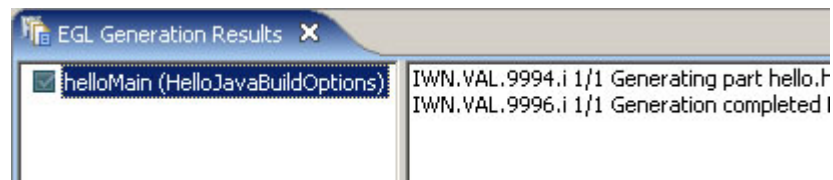


```
package hello;

program helloMain type BasicProgram

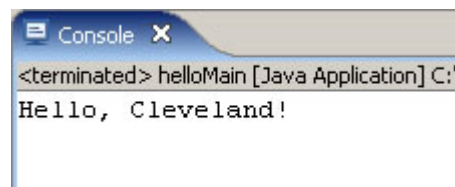
    function main()
        SysLib.writeStdout("Hello, Cleveland!");
    end
end
```

13. Press **CTRL+G**. CTRL + G will generate a native language version of the file. In this case, EGL generates a Java version of your EGL program. Your generation results will appear in a tabbed window directly below the editor in the default workbench configuration. The EGL Generation Results view looks like this:



In the Project Explorer view, you can now see the file Hello\Java Resources\hello\helloMain.java. Double-click the file name if you want to display the contents in the editor and see the considerable quantity of Java code that resulted from your one-line EGL function.

14. Right click the file name Hello\Java Resources\hello\helloMain.java in the Navigator view, and from the resulting option window click **Run > Java Application**.
15. After a moment, the lettering on the Console view's tab turns bold, indicating that it has messages.
16. Switch to the Console view by clicking its tab. The Console view displays the message from your program.



This message in the Console view demonstrates that this simple program works.

Summary

This is the end of the hello world tutorial for EGL.

One of the principles of agile programming is to start small and build up to larger things.

Lessons learned

By completing this tutorial, you learned how to do the following tasks:

- Work with projects, files, and editors in the workbench
- Create an EGL project and an EGL source file
- Write and generate EGL code
- Run an EGL program

Additional resources

You can learn more about EGL by expanding the simple program you created, by working with more advanced EGL tutorials, or by reading about EGL in the help system.

Appendix. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created

programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
3600 Steeles Avenue East
Markham, ON Canada L3R 9Z7

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.html>.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.



Printed in USA