Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle WebLogic Server 11*g*: Administration Essentials

Volume I • Student Guide

D58682GC20

Edition 2.0

July 2010

D68356

ORACLE

Authors

Shankar Raman

Steve Friedberg

Technical Contributors and Reviewers

Anand Rudrabatla

Angelika Krupp

Bala Kothandaraman

David Cabelus

Holger Dindler Rasmussen

Matthew Slingsby

Mike Blevins

Mike Lehmann

Nagavalli Pataballa

Serge Moiseev

Shailesh Dwivedi

Steve Button

Takyiu Liu

TJ Palazzolo

Werner Bauer

William Albert

Will Hopkins

Graphic Designer

Satish Bettegowda

Editors

Amitha Narayan

Malavika Jinka

Publishers

Shaik Mahaboob Basha

Veena Narasimhan

Copyright © 2010, Oracle and/or it affiliates. All rights reserved.

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS
The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Contents

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carl.mcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Preface

1 Introduction
Objectives 1-2
Course Prerequisites 1-3
Course Objectives 1-4
Course Schedule 1-6
Facilities in Your Location 1-8
Summary 1-9

2 Introducing Oracle Fusion Middleware Platform
Objectives 2-2
Oracle Fusion Middleware 2-3
Oracle SOA and Oracle WebCenter Suites 2-5
Oracle Identity and Access Management 2-6
Oracle Business Intelligence 2-7
Oracle Portal, Forms, and Reports 2-8
Web Tier Components 2-9
Oracle Fusion Middleware Management Infrastructure 2-10
Oracle Coherence: In-Memory Data Grid 2-11
Relationship of Fusion Middleware Products to WebLogic Server 2-12
Typical Oracle Fusion Middleware Environment 2-13
Summary 2-14
Practice 2 Overview: Logging In to the Lab Environment 2-15

3 Defining Java Enterprise Edition Terminology and Architecture
Objectives 3-2
Distributed Systems 3-3
How Standards Help 3-5
Java EE Standard 3-6
Java EE Architecture 3-8
Java Servlets 3-9
`SimplestServlet.java` 3-10
JavaServer Pages (JSP) 3-11
`realsimple.jsp` 3-12

Enterprise JavaBeans (EJBs) 3-13
Java Database Connectivity (JDBC) 3-14
Java Naming and Directory Interface (JNDI) 3-15
JNDI Tree 3-16
JNDI Contexts and Subcontexts 3-17
Java Transaction API (JTA) 3-18
Java Message Service (JMS) 3-19
Java Authentication and Authorization (JAAS) 3-20
Java Management Extension (JMX) 3-21
Java EE Connector Architecture (JCA) 3-22
Client Application 3-23
Web Client 3-24
Proxy Server 3-25
Web Server 3-26
Firewalls 3-27
Application Servers 3-28
Web Application Server Configuration 3-29
Application Server Configuration 3-30
Quiz 3-31
Summary 3-32
Practice 3 Overview: Defining Terminology and Architecture 3-33

4 Installing Oracle WebLogic Server 11*g*
Objectives 4-2
Road Map 4-3
Oracle WebLogic Server Installation 4-4
System Requirements 4-5
GUI Mode Installation 4-6
Choosing or Creating a Home Directory 4-7
Registering for Support 4-8
Choosing an Installation Type and Products 4-9
Choosing the JDK and Product Directory 4-10
Installation and Summary 4-11
QuickStart 4-12
Road Map 4-13
Console and Silent Mode Installations 4-14
Postinstallation: Oracle Home 4-15
Oracle WebLogic Server Directory Structure 4-16
Setting Environment Variables 4-18
Defining Environment Variables 4-19
List of Environment Variables and Their Meanings 4-21

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCrinder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Documentation 4-23
Road Map 4-24
Specialized Installations 4-25
Downloading Software from OTN 4-27
Quiz 4-28
Summary 4-31
Practice 3 Overview: Installing Oracle WebLogic Server 11*g* 4-32

5 Configuring a Simple Domain
Objectives 5-2
Road Map 5-4
Domain: Overview 5-5
Domain Diagram 5-7
Configuring a Domain 5-8
Starting the Domain Configuration Wizard 5-10
Creating a Domain Using the Domain Configuration Wizard 5-11
Creating a New WebLogic Domain and Selecting the Domain Source 5-12
Configuring Administrator Settings 5-13
Configuring Startup Mode and JDK 5-14
Customizing Optional Configuration 5-15
Configuring the Administration and Managed Servers 5-16
Configuring Clusters and Assigning Servers to Clusters 5-17
Creating an HTTP Proxy Application and Configuring Machines 5-19
Assigning Servers to Machines 5-21
Configuring JDBC Data Sources 5-22
Testing Data Source Connections 5-25
Running Database Scripts 5-26
Configuring the JMS File Store 5-27
Customizing Application and Service Targeting Configuration 5-29
Configuring RDBMS Security Store Database 5-30
Reviewing the WebLogic Domain 5-32
Creating the WebLogic Domain 5-33
Domain Directory Structure 5-34
Road Map 5-36
JVM Run-Time Arguments 5-37
Oracle WebLogic Server Dependencies 5-38
Configuring `CLASSPATH` 5-39
Starting Oracle WebLogic Administration Server 5-41
Starting Administration Server Using `startWebLogic.sh` 5-43
Starting the Administration Server by Using the `java weblogic.Server` Command 5-45

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carl.mcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Stopping the Administration Server 5-46
Quiz 5-47
Summary 5-52
Practice 5 Overview: Configuring a Simple Domain 5-53

6 Configuring a Domain Using Templates
Objectives 6-2
Road Map 6-3
Custom Domain Templates 6-4
Domain Template Builder 6-6
Creating a Domain Template 6-7
Comparing Domain Templates 6-8
`wls.jar` Template 6-9
`MedRecTemplate.jar` Template 6-10
Extending an Existing Domain 6-11
Select Which Existing Domain to Extend 6-12
Select What You Want to Extend it With 6-13
Resolving Conflicts 6-14
Summary Before Extending 6-15
Road Map 6-16
Templates for SOA, JDeveloper, and Others 6-17
`oracle.soa_template_11.1.1.jar` Template 6-18
WLS Configuration in the Context of Other Products in the Fusion Middleware Suite 6-20
RCU 6-21
SOA Installation 6-22
Quiz 6-23
Summary 6-26
Practice 6 Overview: Using a Domain Template 6-27

7 Using Administration Console and WLST
Objectives 7-2
Road Map 7-4
Benefits of Using the Administration Console 7-5
Accessing the Administration Console 7-6
Administration Console Login 7-7
Basic Navigation 7-8
Using the Help System 7-9
General Administration Console User Preferences 7-10
Advanced Console Options 7-11
Setting Basic Properties 7-12

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCrudden (carlmccrudden@deloitte.com) has a non-transferable license to use this Student Guide.

Administration Console Monitoring 7-13
Configuration Change Management 7-14
Configuration Change Management Using the Administration Console Change Center 7-15
Domain Configuration Repository 7-16
Configuration Change Process 7-17
Configuration Management Architecture 7-18
XML Schema for `config.xml` 7-20
Road Map 7-22
WebLogic Scripting Tool (WLST) 7-23
Jython 7-24
Using Jython 7-25
WLST Modes 7-26
WLST Example 7-27
WLST Command Requirements 7-28
Running WLST Scripts 7-29
Importing WLST as a Jython Module 7-30
General WLST Commands 7-31
Offline WLST Commands 7-32
Creating a Domain: Example 7-33
Online WLST Commands 7-34
WebLogic JMX: Overview 7-35
Navigating JMX MBeans 7-36
Generating a WLST Script 7-37
Quiz 7-38
Summary 7-44
Practice 7 Overview: Using the Administrative Console and WLST 7-45

8 Configuring Managed Servers
Objectives 8-2
Road Map 8-3
Configuring Managed Servers 8-4
Creating a Managed Server with WLST 8-5
Starting Oracle WebLogic Managed Servers 8-7
Starting a Managed Server Using `startManagedWebLogic.sh` 8-8
Command-Line Requirements for Starting the Managed Server Using `java weblogic.Server` 8-10
Starting a Managed Server Using the Administration Console 8-12
Shutting Down a Server 8-13
Shutting Down a Domain 8-14

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Boot Identity File 8-16
Monitoring All Servers 8-18
Customizing the View for All Servers 8-20
Monitoring Individual Servers 8-21
Demonstration 8-22
Road Map 8-23
Creating a Managed Server on a Remote Computer 8-24
`pack` and `unpack`: Examples 8-26
Road Map 8-27
Managed Server Independence (MSI) 8-28
MSI Search Order 8-29
When the Administration Server Is Down 8-31
Running Multiple WLS Instances 8-32
Quiz 8-33
Summary 8-38
Practice 8 Overview: Configuring a Managed Server 8-39

9 Configuring Node Managers
Objectives 9-2
Road Map 9-3
What Node Managers Can Do 9-4
Road Map 9-6
What Is a Machine? 9-7
Relationship of Machines to Other Components 9-8
Creating a Machine 9-9
Defining Names and OS of Machines 9-10
Assigning Servers to a Machine 9-11
Monitoring Machines and Servers 9-12
Configuring a Machine to Use a Node Manager 9-13
Node Manager Architecture 9-14
How a Node Manager Starts an Administration Server 9-15
How a Node Manager Starts a Managed Server 9-16
How a Node Manager Restarts an Administration Server 9-17
How a Node Manager Restarts a Managed Server 9-18
How a Node Manager Shuts Down a Server Instance 9-19
Versions of Node Managers 9-20
Road Map 9-22
Node Manager Default Behaviors 9-23
Configuring a Java-Based Node Manager 9-24
Reconfiguring the Startup Service for a Windows Installation 9-26
Node Manager as a UNIX Daemon 9-27

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCracken (teamcgroder@deloitte.com) has a non-transferable license to use this Student Guide.

Reviewing `nodemanager.properties` 9-28
Configuring a Script-Based Node Manager 9-30
Creating Management OS Users 9-31
Additional Configuration Information 9-32
Configuring the `nodemanager.domains` File 9-33
Defining the Administration Server Address 9-34
Setting Node Manager Environment Variables 9-35
Node Manager Configuration and Log Files 9-36
Quiz 9-40
Summary 9-43
Practice 9 Overview: Configuring Machines and Node Managers 9-44

10 Viewing and Managing Logs in Oracle WLS Environment
Objectives 10-2
Road Map 10-3
Oracle WebLogic Server Logs 10-4
Server and Domain Logs 10-6
Configuring Server Logging 10-7
Configuring Server Logging: Advanced 10-8
HTTP Access Logs 10-10
Apache Commons Logging API 10-11
Using the Console to View Logs 10-12
Using WLST to View Logs 10-13
Message Attributes 10-14
Message Severity 10-15
Message Catalog Using the Web 10-16
Message Catalog Cross-Reference 10-17
Road Map 10-18
Log Filters 10-19
Creating a Log Filter 10-20
Applying a Log Filter 10-21
Using the Console to Monitor 10-22
Monitoring Running Servers 10-23
Customizing Views 10-24
Monitoring Individual Servers 10-25
Network-Addressing Environment 10-26
Network-Addressing Features 10-27
Quiz 10-28
Summary 10-29
Practice 10 Overview: Viewing and Managing WLS Logs 10-30

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

11 Deployment Concepts
Objectives 11-2
Road Map 11-3
Overview of Deployment 11-4
What Is Deployed? 11-5
Deployment Process 11-7
Deployment Methods 11-8
Deployment Tools 11-9
Console Deployment Method 11-10
Console Deployment Production Mode 11-11
Preparing a New Application 11-12
Preparing a New Application: Targeting 11-13
Preparing a New Application: Settings 11-14
Deploying or Undeploying Applications 11-15
Redeploying an Application 11-16
Starting and Stopping an Application 11-17
Editing Deployment Descriptors 11-18
Monitoring an Application 11-19
Application Testing 11-20
Deleting Applications 11-21
Command-Line Deployment 11-22
Deployment with `weblogic.Deployer` 11-23
More `weblogic.Deployer` Examples 11-24
Deploying Applications with WLST 11-25
Deploying an Application with WLST 11-26
Deployment with WLST 11-27
Road Map 11-28
Production Mode Flag 11-29
Autodeployment 11-30
Autodeploying Using an Expanded Directory 11-31
FastSwap and On-Demand Deployment 11-32
Road Map 11-34
Role of Web Servers 11-35
A Typical Web Interaction 11-36
MIME Types 11-38
HTTP Status Codes 11-39
Static Content 11-40
Dynamic Content 11-41
Configuring Oracle HTTP Server to Serve Multiple WebLogic Servers 11-42
`mod_wl_ohs.conf` 11-43

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGrudor (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Verifying Ports Used by OHS 11-44
Quiz 11-45
Summary 11-48

12 Deploying Java EE Applications
Objectives 12-2
Road Map 12-3
Java EE Web Applications 12-4
Packaging Web Applications 12-6
Web Application Structure 12-7
Web Application Archive 12-8
Optional Configuration of Web Applications 12-9
`web.xml` 12-10
`weblogic.xml` 12-11
`weblogic.xml` Deployment Descriptor 12-12
URLs and Web Applications 12-13
Web Service Applications 12-14
Virtual Directory Mappings 12-15
Virtual Directory Mapping: Example 12-16
Road Map 12-17
EJB Applications 12-18
Types of EJBs 12-19
EJB Application Structure 12-21
`weblogic-ejb-jar.xml` 12-22
Administrator Tasks with EJBs 12-23
Road Map 12-24
What Is an Enterprise Application? 12-25
Typical Java EE System 12-26
Java EE Enterprise Application 12-27
Why Enterprise Applications? 12-29
Enterprise Application Structure 12-30
`weblogic-application.xml` 12-31
Application Scoping 12-32
EAR Class Libraries 12-33
Java EE Library Support 12-34
WebLogic Java EE Shared Libraries 12-35
Shared Library References 12-36
Quiz 12-37
Summary 12-40
Practice 12 Overview: Web Application Deployment Concepts 12-41

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carl.mcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

13 Advanced Deployment
Objectives 13-2
Road Map 13-3
What Is a Deployment Plan? 13-4
Configuring an Application for Multiple Deployment Environments 13-5
Sample Deployment Plan 13-7
Creating a Deployment Plan 13-8
Creating a New Deployment Plan 13-10
`weblogic.PlanGenerator` 13-11
Using the Administration Console to Generate a Deployment Plan 13-12
Modifying and Saving Data to Create a New Plan 13-13
New Deployment Plan Shows Changed Values 13-14
Using an Existing Deployment Plan to Configure an Application 13-15
Using an Existing Deployment Plan 13-17
Directory Structure for Easier Production Deployment 13-18
Generic File-Loading Overrides 13-19
Performing a Sanity Check in Production Without Disruption to the Clients 13-20
Road Map 13-21
Staged Deployment 13-22
Road Map 13-23
Application Availability 13-24
Production Redeployment and Application Versioning 13-25
WebLogic Production Redeployment 13-27
Production Redeployment 13-28
Advantages of Production Redeployment 13-29
Requirements and Restrictions for Production Redeployment 13-30
Redeploying a New Application Version 13-31
Redeploying Versus Distributing 13-32
Distributing a New Version of the Production Application 13-33
Distributing a New Application Version 13-35
Production Redeployment 13-36
Quiz 13-37
Summary 13-40
Practice 13 Overview: Deploying Production Applications 13-41

14 Understanding JDBC and Configuring Data Sources
Objectives 14-2
Road Map 14-3
JDBC Review 14-4
JDBC Data Sources 14-5

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl Mcarter (carl.cgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Data Source Scope 14-6
Multi-Tier Architecture 14-7
Type 4 Drivers 14-8
WebLogic JDBC Drivers 14-9
Road Map 14-10
What Is a Connection Pool? 14-11
JDBC Connection Pooling 14-12
Benefits of Connection Pools 14-13
Modular Configuration and Deployment of JDBC Resources 14-14
How Data Source Connection Pools Are Used 14-15
Creating a Data Source Using the Administration Console 14-16
Non-XA Configuration 14-17
Data Source Connection Properties 14-18
Test Configuration 14-19
Connection Pool Configuration 14-20
Connection Pool Advanced 14-21
Targeting a Data Source 14-22
Viewing the Server JNDI Tree via the Administration Console 14-23
Listing the JNDI Contents Via WLST 14-24
Demonstration 14-25
JDBC URLs 14-26
Connection Properties 14-27
Specifying Connection Properties 14-28
Road Map 14-29
Monitoring and Testing a Data Source 14-30
Connection Pool Life Cycle 14-31
Fixing an Offline Data Source 14-32
Quiz 14-33
Summary 14-36
Practice 14 Overview: Configuring JDBC Data Sources 14-37

15 Setting Up Java Message Service (JMS) Resources
Objectives 15-2
Road Map 15-3
Message-Oriented Middleware 15-4
Point-To-Point Queue 15-5
Publish/Subscribe Topics 15-6
Oracle WebLogic Server JMS Features 15-7
WebLogic JMS Architecture 15-9
Typical JMS Messaging Process 15-10
Transacted Messaging 15-11

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

JMS Administrative Tasks 15-12
Oracle WLS JMS Implementation 15-13
Road Map 15-14
Creating a JMS Server 15-15
Configuring a JMS Server 15-16
Targeting a JMS Server to a Managed Server 15-17
JMS Modules 15-18
Creating a JMS Module 15-20
Modular JMS Resource Configuration and Deployment 15-21
Connection Factories 15-22
Creating a Connection Factory 15-24
Configuring a Connection Factory 15-25
Destination 15-26
Queue Destinations 15-27
Topic Destinations 15-28
Creating a Destination (Topic) 15-29
Threshold, Quota, and Paging 15-31
Configuring Thresholds and Quotas 15-32
Road Map 15-33
Durable Subscribers and Subscriptions 15-34
How a Durable Subscription Works 15-35
Configuring a Durable Subscription 15-36
Persistent Messaging 15-37
Creating a JMS Store 15-38
Creating a JDBC Store for JMS 15-39
Creating a JMS JDBC Store 15-40
Assigning a Store to a JMS Server 15-41
Persistent Connection Factory 15-42
Configuring Destination Overrides 15-43
Road Map 15-44
Monitoring JMS Servers 15-45
Monitoring and Managing Destinations (Active Queues and Topics) 15-46
Monitoring Queues 15-47
Managing Messages in a Queue 15-48
Quiz 15-49
Summary 15-51
Practice 15 Overview: Configuring JMS Resources 15-52

16 Introduction to Clustering
Objectives 16-2
Road Map 16-3

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCrudden (carmccrudden@deloitte.com) has a non-transferable license to use this Student Guide.

What Is a Cluster? 16-4
Benefits of Clustering 16-5
What Can Be Clustered 16-6
Proxy Servers for HTTP Clusters 16-7
High Availability for EJBs 16-8
Road Map 16-9
Selecting a Cluster Architecture 16-10
Cluster Architecture 16-11
Basic Cluster Architecture 16-12
Basic Cluster Architecture: Advantages and Disadvantages 16-13
Multitier Cluster Architecture 16-14
Multitier: Advantages and Disadvantages 16-15
Basic Cluster Proxy Architecture 16-17
Multitier Cluster Proxy Architecture 16-18
Proxy Web Server Plug-In Versus Load Balancer 16-19
Proxy Plug-Ins 16-20
OHS as Proxy Web Server 16-21
Request Flow When Using OHS 16-22
WLS `HttpClusterServlet` 16-23
Road Map 16-24
Server Communication in a Cluster 16-25
One-To-Many Communications 16-27
Considerations When Using Unicast 16-29
Peer-To-Peer Communications 16-30
Clusterwide JNDI Naming Service 16-31
Name Conflicts and Resolution 16-32
Quiz 16-33
Summary 16-35

17 Configuring a Cluster
Objectives 17-2
Road Map 17-3
Preparing Your Environment 17-4
Hardware 17-5
IP Addresses and Host Names 17-6
Cluster Address 17-7
Road Map 17-8
Methods of Configuring Clusters 17-9
Creating a Cluster by Using the Administration Console 17-10
Setting Cluster Attributes 17-12
Configuring Cluster Communication 17-13

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carl.mcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Adding Cluster Members: Option 1 17-14
Adding Cluster Members: Option 2 17-15
Creating a Cluster with the Configuration Wizard 17-16
Creating a Cluster Using the Cluster MBean 17-17
Clusters and WLST 17-18
Synchronizing When Starting Servers in a Cluster 17-19
Configuring OHS as Proxy Server 17-21
Starting and Stopping OHS Manually 17-22
Verifying Access Through OHS 17-23
Successful Access of OHS Splash Page 17-24
Quiz 17-25
Summary 17-26
Practice 17 Overview: Configuring Clusters 17-27

18 Managing Clusters
Objectives 18-2
Road Map 18-3
Deploying Applications to a Cluster 18-4
Two-Phase Deployment 18-5
Considerations for Deploying to Cluster 18-6
Production Redeployment in a Cluster 18-7
Road Map 18-8
HTTP Session Failover 18-9
HTTP Session State Replication 18-10
HTTP Session In-Memory Replication 18-11
In-Memory Replication and Proxy Servers 18-12
In-Memory Replication 18-13
In-Memory Replication: Example 18-14
Requirements for In-Memory Replication 18-15
Configuring In-Memory Replication 18-16
Failover with Load Balancer 18-18
Replication Groups 18-19
Configuring Replication Groups 18-21
Replication Groups for Different Criteria 18-22
HTTP Session Persistence Using JDBC 18-23
Configuring JDBC Persistence 18-25
JDBC Persistent Table Configuration 18-26
HTTP Session Persistence Using Files 18-28
Configuring File Persistence 18-29
HTTP State Management Best Practices 18-31
Road Map 18-32

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Clustering EJB Objects: Replica-Aware Stub 18-33
EJB: Server Failure Situations 18-34
Load-Balancing Clustered EJB Objects 18-35
Stateless Session Bean Failover 18-36
Configuring EJB Clustering in Deployment Descriptors 18-37
Configuring EJB Clustering Using the Administration Console 18-38
Configuring Clusterable Stateless Session EJBs 18-39
Stateful Session Beans 18-40
Configuring Clusterable Stateful Session EJBs 18-41
Read/Write Versus Read-Only 18-42
Entity Bean Cluster-Aware Home Stubs 18-43
EJB Best Practices 18-44
Quiz 18-45
Summary 18-49
Practice 18: Overview Managing Clusters 18-50

19 Security Concepts and Configuration
Objectives 19-2
Road Map 19-3
Introduction to Oracle WebLogic Security Service 19-4
Oracle Platform Security Services 19-5
Oracle WLS Security Architecture 19-6
Security Services 19-7
Overview of Security Concepts 19-8
Confidentiality 19-9
Credential Mapping 19-11
Road Map 19-12
Security Realms 19-13
Security Model Options for Applications 19-14
How WLS Resources Are Protected 19-16
Users and Groups 19-17
Configuring New Users 19-18
Groups 19-19
Configuring New Groups 19-20
Configuring Group Memberships 19-21
Embedded LDAP Server 19-22
Configuring an Embedded LDAP 19-23
Road Map 19-25
Security Roles 19-26
Configuring the Global Security Role 19-28
Security Policies 19-29

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Policy Conditions 19-30
Protecting Web Applications 19-31
Specifying Protected Web Resources 19-32
Defining Policies and Roles for Other Resources 19-33
Configuring Authentication 19-34
Authentication Examples 19-35
Migrating Security Data 19-36
Exporting the WLS Default Authenticator Provider 19-38
Importing into a Different Domain 19-39
Summary 19-40
Practice 19: Overview Configuring Security for WLS Resources 19-41

20 Protecting Against Attacks
Objectives 20-2
Road Map 20-3
What Is SSL? 20-4
Trust and Identity 20-5
Using an SSL Connection 20-6
Enabling Secure Communication 20-8
Oracle WebLogic Server SSL Requirements 20-10
`keytool` Utility 20-11
Obtaining a Digital Certificate: `keytool` Examples 20-12
Configuring Keystores 20-14
Configuring SSL for an Oracle WebLogic Server 20-15
Road Map 20-16
Protecting Against Attacks 20-17
Man-in-the-Middle Attacks 20-18
Man-in-the-Middle: Countermeasures 20-19
Configuring a Hostname Verifier 20-21
Denial of Service Attacks 20-22
Denial of Service Attacks: Countermeasures 20-23
Filtering Network Connections 20-24
Connection Filter 20-25
Excessive Resource Consumption 20-26
Large Buffer Attacks 20-27
Setting the Post Size 20-28
Connection Starvation 20-29
User Lockout 20-31
Configuring User Lockout 20-32
Unlocking Users 20-33
Protecting the Administration Console 20-34

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Quiz 20-35
Summary 20-37
Practice 20: Overview Configuring Keystores 20-38

21 Backup and Recovery Operations
Objectives 21-2
Road Map 21-3
Review of Terms and Components 21-4
Homes: Oracle, Middleware, WebLogic 21-6
Understanding Backup and Recovery 21-7
Types of Backups 21-9
Backup Recommendations 21-11
Limitations and Restrictions for Backing Up Data 21-12
Performing a Full Offline Backup 21-13
Backing Up a Domain Configuration 21-15
Backing Up an Instance Home 21-16
Creating a Record of Installations 21-17
Road Map 21-18
Directories to Restore 21-19
Recovery After Disaster 21-20
Recovery of Homes 21-21
Recovery of a Managed Server 21-22
Recovery of the Administration Server Configuration 21-23
Restarting an Administration Server on a New Computer 21-24
Recovery of a Cluster 21-25
Restoring OPMN-Managed Components to a New Computer 21-26
Quiz 21-27
Summary 21-32
Practice 21 Overview: Backing Up and Restoring Configuration and Data 21-33

Index

Glossary

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Preface

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Profile

Before You Begin This Course

Before you begin this course, you should be able to

- Issue basic UNIX user-level commands
- Perform UNIX desktop navigation tasks
- Describe basic XML concepts
- Edit basic XML documents
- Describe basic TCP/IP networking client/server concepts

How This Course Is Organized

Oracle WebLogic Server 11g: Administration Essentials Edition 2 (based on version 11gR1 PS2 [11.1.1.3.0] - WLS 10.3.3), is an instructor-led course featuring lectures and hands-on exercises. Online demonstrations and written practice sessions reinforce the concepts and skills that are introduced.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Related Publications

Oracle Publications

Title	Part Number
Oracle Fusion Middleware Online Documentation Library 11g Release 1 (11.1.1.3)	E14571_01
Oracle Fusion Middleware Administrator's Guide 11g Release 1 (11.1.1.3)	E10105-04
Oracle Fusion Middleware Upgrade Planning Guide 11g Release 1 (11.1.1.3)	E10125-04
Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache 11g Release 1 (11.1.1.3)	E10143-03
Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server 11g Release 1 (11.1.1.3)	E10144-03

Additional Publications

- System release bulletins
- Installation and user's guides
- *read.me* files
- International Oracle User's Group (IOUG) articles
- *Oracle Magazine*

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Typographic Conventions

The following two lists explain Oracle University typographical conventions for words that appear within regular text or within code samples.

1. Typographic Conventions for Words Within Regular Text

Convention	Object or Term	Example
Courier New	User input; commands; column, table, and schema names; functions; PL/SQL objects; paths	Use the `SELECT` command to view information stored in the `LAST_NAME` column of the `EMPLOYEES` table. Enter `300`. Log in as `scott`
Initial cap	Triggers; user interface object names, such as button names	Assign a When-Validate-Item trigger to the ORD block. Click the Cancel button.
Italic	Titles of courses and manuals; emphasized words or phrases; placeholders or variables	For more information on the subject see *Oracle SQL Reference Manual* Do *not* save changes to the database. Enter *`hostname`*, where *`hostname`* is the host on which the password is to be changed.
Quotation marks	Lesson or module titles referenced within a course	This subject is covered in Lesson 3, "Working with Objects."

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcruder@deloitte.com) has a non-transferable license to use this Student Guide.

Typographic Conventions (continued)

2. Typographic Conventions for Words Within Code Samples

Convention	Object or Term	Example
Uppercase	Commands, functions	`SELECT employee_id` `FROM  employees;`
Lowercase, italic	Syntax variables	`CREATE ROLE` *`role`*`;`

Initial cap	Forms triggers	`Form module: ORD`
		`Trigger level: S_ITEM.QUANTITY item`
		`Trigger name: When-Validate-Item`
		`. . .`

Lowercase	Column names, table names, filenames, PL/SQL objects	`. . .`
		`OG_ACTIVATE_LAYER`
		`(OG_GET_LAYER ('prod_pie_layer'))`
		`. . .`
		`SELECT last_name`
		`FROM  employees;`

Bold	Text that must be entered by a user	`CREATE USER` **`scott`** `IDENTIFIED BY` **`tiger`**`;`

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carl.mcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

1 Introduction

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

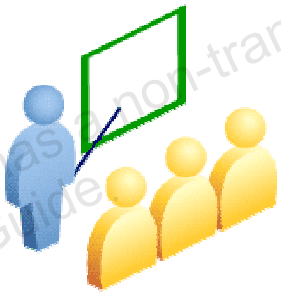
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- List the course prerequisites
- List the course objectives
- List the course schedule
- Identify the facilities in your location



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

The objectives of each lesson have a business scenario describing a real-life situation that can be solved by an Oracle Fusion Middleware solution. For this course, assume that you are the administrator of middle-tier computing resources at Example Corporation (www.example.com). You run a Linux shop although your end users can be anything with a browser (for example, laptops, cell phones, and so on). Your programmers want to deploy Java applications that access a back-end Oracle Database and have front-end Web clients. Initially, your entire middle tier is one computer (just like the lab), but you expect to grow substantially in the very near future. Future scaling might be performed using separate Test and Production servers; later ProdServer1 and ProdServer2 may be used for failover protection. Because you are not the database administrator (DBA), the back-end tier is already installed and configured for you. Because you are not in the programming department, it is assumed that the Java code is a black box that is delivered to you as either ZIP or JAR or packaged in some self-contained way. Your job is to deploy these applications to the middle tier. You use the sample applications medrec (Medical Records) available for Oracle WebLogic Server. As the administrator, you have full life-cycle responsibilities including deploying applications and resources (for example, data sources and message queues), starting or stopping the resources, configuring failover and high availability (for example, clustering and persisting stores), and performing backup and recovery of data in the event of a disaster.

Course Prerequisites

Required

- Basic knowledge of UNIX user-level commands and desktop navigation
- Basic familiarity with Extensible Markup Language (XML) concepts
- Basic TCP/IP networking knowledge of client/server concepts

Suggested

- Basic knowledge of Java EE concepts and constructs including Servlet, JavaServer Pages (JSP), Enterprise JavaBeans (EJB), and Java Message Service (JMS)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Course Prerequisites

These prerequisites can be met by Oracle courses, or by other means of experience and training.

There is a Glossary in the Appendix with many of the acronyms used throughout the course.

Course Objectives

After completing this course, you should be able to:

- Describe the architecture of WebLogic Server including domains, servers, and machines
- Install, configure, and use WebLogic Server
- Perform routine Oracle WebLogic Server administration functions
- Set up a cluster of servers and distribute applications and resources to the cluster
- Configure Oracle HTTP Server as the WebTier front end for Oracle WebLogic Server instances and clusters
- Deploy different types of Java EE applications to Oracle WebLogic Server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Course Objectives

In addition to these high-level terminal objectives, each lesson has a lower-level set of enabling objectives.

Course Objectives

After completing this course, you should be able to:

- Monitor the application server by using GUI and command-line tools, such as automation scripts
- Deploy and manage large-scale Java EE applications to servers or clusters through the entire development and production life cycle
- Configure basic resource and application security
- Back up and recover from various failures

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Course Objectives (continued)

In addition to these high-level terminal objectives, each lesson has a lower-level set of enabling objectives.

Course Schedule

Day One

1. Introduction
2. Fusion Middleware Overview
3. Terms and Architecture
4. Installation
5. Configure Domains

Day Two

6. Configure Using Templates
7. Using Administration Console and WLST
8. Configure Managed Servers
9. Node Manager

Day Three

10. Logs and Filters
11. Deployment Concepts
12. Deploying Web Applications
13. Advanced Deployment

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Course Schedule

This schedule is approximate and subject to change.

Course Schedule

Day Four

14. JDBC
15. JMS
16. Introduction to Clusters
17. Configure Clusters

Day Five

18. Manage Clusters
19. Securing WLS
20. Protecting WLS
21. Backup and Recovery

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

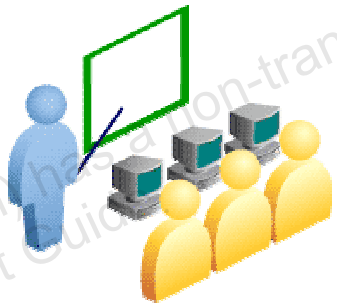
Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Course Schedule (continued)

This schedule is approximate and subject to change.

Facilities in Your Location

- Enrollment/Registration/Sign in
- Badges
- Parking
- Phones
- Internet
- Restrooms
- Labs
- Lunch
- Kitchen/Snacks
- Hours
- Materials (paper, pens, and markers)



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Facilities in Your Location

Contact your instructor or the education coordinator for site-specific information. This may not be applicable for a Live Virtual Class (LVC).

Summary

In this lesson, you should have learned how to:

- List the course prerequisites
- List the course objectives
- List the course schedule
- Identify the facilities in your location

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Summary

There are no practices or exercises for this lesson.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Introducing Oracle Fusion Middleware Platform

2

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Describe the Oracle Fusion Middleware environment
- Describe how WebLogic Server supports various Fusion Middleware Suites
- Describe how various Fusion Middleware Suites augment the functions of WebLogic Server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

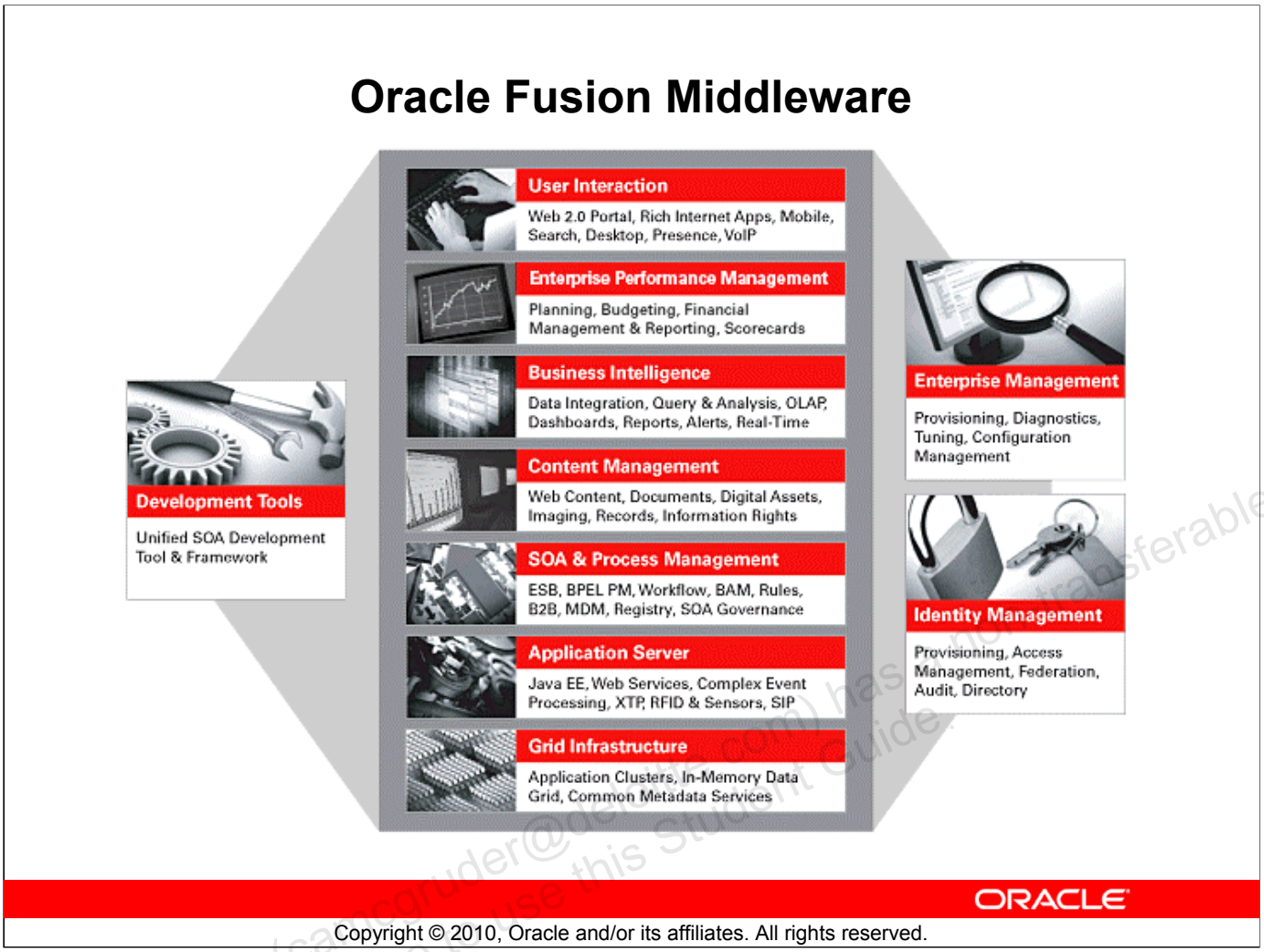
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

You are a new WebLogic Server Administrator in Example Corporation. You have come to know that WebLogic Server is one among many of the Oracle Fusion Middleware products that Example Corporation uses. You would like to know more about the role of WebLogic Server in the Oracle Fusion Middleware environment.



Oracle Fusion Middleware

Middleware is a term used to describe computer software that connects software components or applications. Middleware is used most often to support complex, distributed business software applications. Middleware includes Web servers, application servers, content management systems, and similar tools that support application development and delivery. Middleware is especially integral to information technology based on Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web services, and Service-Oriented Architecture (SOA).

Oracle Fusion Middleware is a collection of standards-based software products that spans a range of tools and services, from a Java Enterprise Edition 5 (Java EE)–compliant environment and developer tools to integration services, business intelligence, collaboration, and content management. Oracle Fusion Middleware offers complete support for development, deployment, and management.

Oracle Fusion Middleware offers solutions and support to support complex, distributed business software applications. It includes Web servers, application servers, content management systems, and similar tools that support application development and delivery.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle Fusion Middleware (continued)

Oracle Fusion Middleware offers the following solutions to middleware implementation:

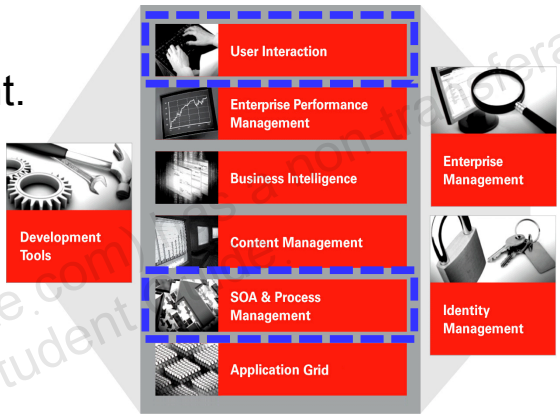
- **Design and Development Tools:** Is a single integrated, but modular, design tool to build a complete application (rather than too many specialized tools). The design tool includes a single design environment for user interface, business logic, service composition, business process or workflow, business rules, and business intelligence. The design tool enables you to simplify design and debugging and improve productivity.
- **Application Server:** Is a standards-based Java EE application server to run enterprise applications and provide the Web Services infrastructure for interoperability
- **User Interaction:** Is a single, end-user environment that allows users to access their enterprise applications, business processes, and business intelligence and share information with each other. This end-user environment is multichannel, so it can be accessed from a variety of different clients (mobile clients, desktop clients, Voice-over IP clients, and so on).
- **Integration and Business Process Management (BPM):** Is a standards-based service bus to connect applications with each other and legacy systems using messaging. A BPM or workflow engine connects the application into a business process or workflow. Business activity monitoring monitors and optimizes business processes in real time.
- **Business Intelligence:** Is a suite of business intelligence tools from extract, transform, and load (ETL) to integrate data into warehouses; query, analysis, and reporting tools for decision support; and scorecards to compare how the business is doing against key performance indicators and alerting to drive notifications to users based on problems in the business
- **Enterprise Content Management:** Is a repository within which to manage documents, digital assets, scanned images, and other forms of content to integrate this content with a company's enterprise applications, Web sites, and business processes
- **SOA:** Leverages existing investments in applications and systems, so an organization can focus more resources and budget on innovation and delivering new business services
- **Security and Identity Management:** Lowers the cost of security administration across multiple applications and systems in an enterprise by centralizing how you create and provision users, their identities, and roles and enable them to have single sign-on access
- **Enterprise Management:** Lowers the cost of operations and administration by running middleware on a grid architecture with grouping, backup, and other high availability technologies and integration with Oracle Enterprise Manager for systems management
- **Grid Infrastructure:** Allows an in-memory data cache for high-volume database transactions and/or Java messages to be spread across several servers, even spread across several domains

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrady (carl.mccrady@deloitte.com) has a non-transferable license to use this student guide.

Oracle SOA and Oracle WebCenter Suites

- Oracle SOA Suite enables services to be created, managed, and orchestrated into SOA-composite applications.
- Oracle WebCenter is designed to help integrate structured and unstructured content, and Web 2.0-style services into the applications.
- Both SOA and WebCenter run in the Oracle WLS environment.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle SOA and Oracle WebCenter Suites

Oracle SOA Suite is a middleware component for service-oriented applications. SOA runs within a Java environment and provides a complete set of service infrastructure components for designing, deploying, and managing SOA-composite applications. It also enables services to be created, managed, and orchestrated into SOA-composite applications.

Oracle WebCenter is an integrated suite of technology designed to deliver a unified, context-aware user experience that integrates structured and unstructured content, business intelligence, business processes, communication, and collaboration services.

Before installing SOA or WebCenter, you should have installed a:

- WebLogic Server environment
- Database and configured a Metadata Repository (schemas) in it

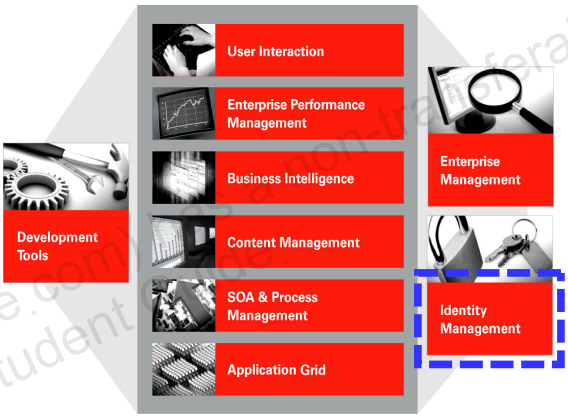
Administration of SOA environment is explained in the *Oracle SOA Suite 11g: Administration* course.

For further information, you can refer to documentation guides for SOA and WebCenter.

Oracle Identity and Access Management

Oracle Identity and Access Management solution:

- Enables you to centralize identity and access management
- Requires a Java EE environment for operation
- Can be easily configured as the central system for use in the WebLogic environment



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle Identity and Access Management

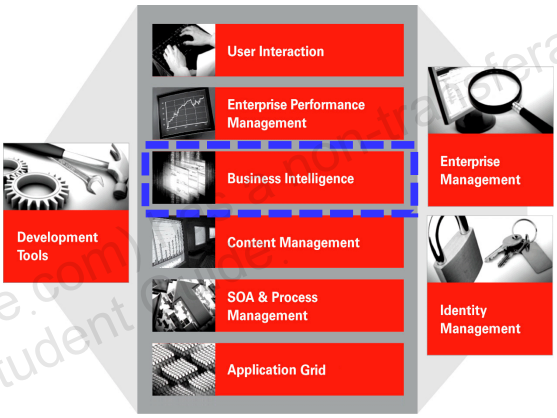
The Oracle Identity and Access Management solution enables you to manage users and their access privileges in your enterprise. Using Oracle Identity Management, you can manage the entire identity management life cycle from the initial creation of users, their access privileges to dynamically adapting to changes in enterprise business requirements.

Oracle Identity and Access Management products provide for a shared infrastructure for all Oracle applications. It also provides services and interfaces that facilitate third-party enterprise application development. These interfaces are useful for application developers who need to incorporate identity management into their applications.

There is a symbiotic relationship between Oracle Identity and Access Management products and the Oracle WebLogic Server environment. Most of the Identity and Access Management products run in a Java EE platform, such as WebLogic Server. Similarly, WebLogic can be configured to make use of the centralized identity and access management functions provided by the Oracle Identity and Access Management suite.

Oracle Business Intelligence

- Oracle Business Intelligence (BI) enables you to view and analyze data from data warehouses and data marts.
- BI enables you to gather data from various sources and to extract, analyze, and report in appropriate formats.
- BI is easily integrated with Oracle Identity Management solutions, such as SSO.
- BI can also be integrated with SOA.



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Oracle Business Intelligence

Oracle Business Intelligence (BI) provides an integrated Enterprise Performance Management System; an array of query, reporting, analysis, alerting, mobile analytics, data integration, and management; desktop integration; as well as leading financial performance management applications, operational BI applications, and data warehousing.

BI systems provide historical, current, and predictive views of business operations, using data that has been gathered into a data warehouse or a data mart.

BI provides tools to extract, analyze, and report information gathered from various sources including sales, production, financial, and many other sources of business performance management. Information may be gathered on comparable companies to produce benchmarks.

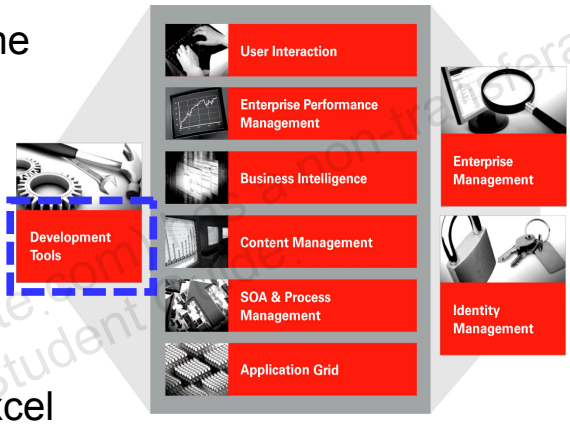
BI can be integrated with SOA that requires WebLogic Server support. BI can also integrate with Oracle Identity Management solutions, such as Single Sign On (SSO).

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle Portal, Forms, and Reports

- Portal:
  - Is a Web-based interface that uses wizards
  - Requires Oracle Database
  - Uses a Portal Developer Kit to extend via programming
- Forms:
  - Are created by developers using Oracle Forms Builder
  - Can be compiled into run-time services for deployment
  - Run as run-time services in a Java EE environment
- Reports:
  - Dynamically retrieve, format, and publish information in several formats
  - Are also available via MS Excel



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can McCrudder (cancrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle Portal, Forms, and Reports

Oracle Portal offers a complete and integrated framework for building, deploying, and managing enterprise portals. Oracle Portal delivers a unified and secure point of access to all enterprise information and services to improve business visibility and collaboration, reduce integration costs, and ensure investment protection.

Oracle Forms comprises a run-time environment:

- Oracle Forms Services is a middle-tier application framework for deploying complex, transactional forms applications to a network such as an intranet or the Internet.
- Oracle Forms Builder is a development environment that developers use to build Forms applications and deploy them with Oracle Forms Services. Developers can also take applications that were previously deployed in a client/server and move them to a three-tier architecture.

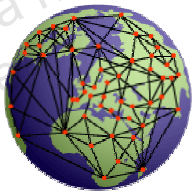
Oracle Reports provides you with access to different data sources out of the box. It supports SQL, PL/SQL (`ref cursor`), XML, JDBC, Oracle OLAP, and text files. You can combine queries from different data sources in a single report and link them to produce master-detail relationships.

Oracle Reports offers various output formats for paper-based publishing. You can use your report definition to generate output in PDF, HTML, HTMLCSS, RTF, Excel-compatible spreadsheet, as well as XML and DelimitedData.

Web Tier Components

The Web tier contains system components that are managed using Oracle Process Management Server.

- Oracle HTTP Server:
  - Provides scalability and supports Web services functionality
  - Contains modules specific to Oracle FMW to support integration with other FMW components
  - Acts as a proxy server for Oracle WebLogic Servers
- Oracle Web Cache:
  - Runs in the Web tier and speeds up requests for applications being served by WebLogic Server



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Web Tier Components

The Web tier is responsible for interacting with the end user, such as Web browsers primarily in the form of HTTP requests and responses. It is the outermost tier in the application server, closest to the end user. Two components make up the Web tier:

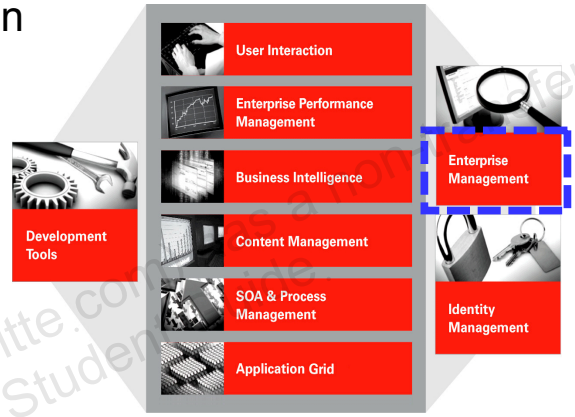
1. **Oracle HTTP Server:** Oracle HTTP Server (OHS) can greatly increase the scalability and functionality of Web services. Oracle HTTP Server is based on Apache 2.2.8 and includes modules developed specifically by Oracle for single sign-on, clustered deployment, and high availability. OHS includes mod_wl_ohs that enables OHS to be configured as an HTTP proxy for Oracle WebLogic Server. Configuration of OHS as proxy is covered in this course.
2. **Oracle Web Cache:** Oracle Web Cache runs in the Web tier and speeds up requests for applications being served by WebLogic Server. Oracle Web Cache is a content-aware server accelerator or reverse proxy for the Web tier that improves the performance, scalability, and availability of Web sites that run on Oracle HTTP Server.

Oracle Web Cache is the primary caching mechanism provided with Oracle Fusion Middleware. Caching improves the performance, scalability, and availability of Web sites that run on Oracle WebLogic Server or Oracle Application Server by storing frequently accessed URLs in memory.

Oracle Fusion Middleware Management Infrastructure

Oracle offers the following primary tools for managing your Oracle Fusion Middleware installations:

- Oracle WebLogic Server Administration Console
- Oracle Enterprise Manager Fusion Middleware Control
- Oracle Fusion Middleware command-line tools such as OPMNCTL, WLST, and so on
- Fusion Middleware Control MBean browser
- Oracle Web Services Manager



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) is a non-transferable license to use this Student Guide.

Oracle Fusion Middleware Management Infrastructure

Oracle WebLogic Server Administration Console is a Web browser-based, graphical user interface that you use to manage an Oracle WebLogic Server domain. It is accessible from any supported Web browser with network access to the administration server. In this course, you learn to administer a WebLogic domain using the Administration Console.

Oracle Enterprise Manager Fusion Middleware Control is a Web browser-based, graphical user interface that you can use to monitor and administer Fusion Middleware installations. Fusion Middleware Control organizes a wide variety of performance data and administrative functions into distinct, Web-based home pages for the different Fusion Middleware elements.

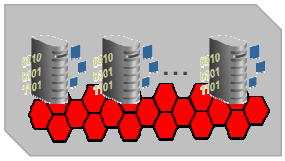
Oracle WebLogic Scripting Tool (WLST) is a command-line scripting environment that you can use to create, manage, and monitor Oracle WebLogic Server domains. It is based on the Java scripting interpreter, Jython. WLST provides a set of scripting functions (commands) that are specific to WebLogic Server. You can extend the WebLogic scripting language to suit your needs by following the Jython language syntax. In this course, you learn to use WLST for performing administrative tasks in a WebLogic domain.

Oracle Process Manager and Notification Server (OPMN) manages and monitors system components of Oracle Fusion Middleware. The command-line interface to OPMN is `opmnctl`. Using `opmnctl`, you can perform life-cycle management (start, stop, and status) for system components.

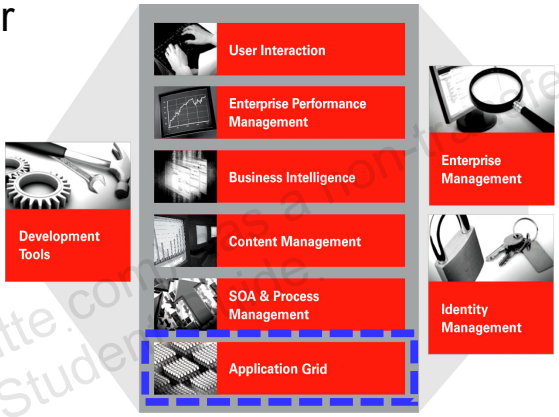
Oracle Coherence: In-Memory Data Grid

Oracle Coherence:

- Automatically and dynamically partitions data in memory across multiple servers
- Enables continuous data availability and transactional integrity, even in the event of a server failure
- Installs with WebLogic Server



Data Grids



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) is a non-transferable license to use this Student Guide.

Oracle Coherence: In-Memory Data Grid

Oracle Coherence Suite is an in-memory application grid that extends the Grid Infrastructure capabilities in Coherence Grid Edition further into the real time data analytics, compute intensive and high performance transaction space to offer a comprehensive system for building highly reliable, available, and scalable middleware infrastructure. It is installed by default with WebLogic Server, though by default WebLogic Server does not require it. Based on the leading Coherence Grid Edition, the addition of JRockit Real Time, WebLogic Operations Control and Enterprise Manager Diagnostics Pack increases dramatically the scale and breadth of applications infrastructure supportable by Oracle Coherence Suite. Oracle Coherence Suite works with any application server, including Oracle WebLogic Server, IBM WebSphere Application Server, and JBoss Application Server, or in a pure grid environment without an application server. Oracle Coherence Suite provides extreme and predictable application scalability and performance with the following capabilities:

- Data grid capabilities
- Java run-time capabilities
- Application management capabilities
- Enhances the traditional clustering capabilities of WLS by:
  - Enabling in-memory session replication across WLS domains
  - Caching Object-Relational Mapping (ORM) across multiple servers transparently
  - Improving performance as well as availability

Relationship of Fusion Middleware Products to WebLogic Server

- Common installation/configuration steps:
  - Install Database (if needed).
  - Install Repository Creation Utility (if needed).
  - Install WebLogic Server.
  - Install Fusion Middleware Component (for example, OID).
  - Configure all software.
- Common operations:
  - The components run in a Java EE Container such as WebLogic Server.
- Common administration:
  - Enterprise Manager oversees all the products at a high level.
  - Each component has its own administration console for finer detail.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

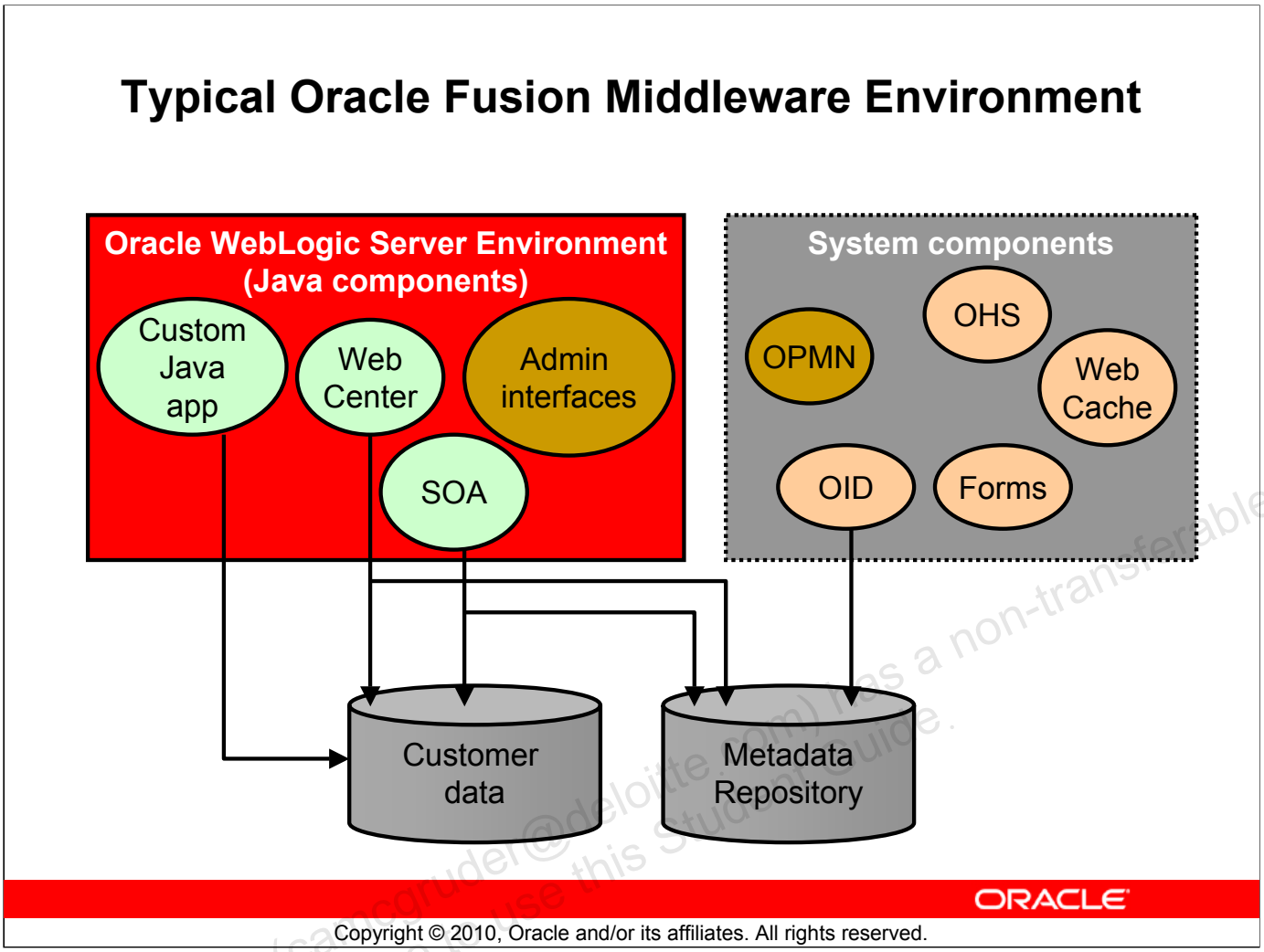
Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Relationship of Fusion Middleware Products to WebLogic Server

The Fusion Middleware products can be interdependent and integrated. Some of them require certain components; others can optionally take advantage of certain components. For example, SOA requires a database to be present; WebLogic does not. However, if WebLogic has a database at its disposal, it can use it. Similarly, Oracle Internet Directory (OID) requires WebLogic, whereas WebLogic does not require OID. But if OID is present, WebLogic can use it. After all the software pieces are installed and aware of each other, configuration specifies how they interact. Other Fusion Middleware products go through a similar set of steps for installation and configuration.

Because the products are Java Enterprise applications, they need to run in a Java container. WebLogic Server provides the Java EE environment for all these applications to run. The life cycle of deploying and starting or stopping the applications can be performed from the WebLogic Administration Console.

There are other administration tools such as Enterprise Manager that can manage the middleware products at a high level (for example, checking whether they are running or stopped). Each of the products in turn has its own console that can perform finer levels of configuration.



Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Typical Oracle Fusion Middleware Environment

From an administrator's perspective, Oracle Fusion Middleware components can be classified into two main types:

- **Java components:** Java components are deployed as one or more Java EE applications and a set of resources. Java components are deployed to an Oracle WebLogic Server domain as part of a domain template. Examples of Java components are the Oracle SOA Suite and Oracle WebCenter components.
- **System components:** This is a manageable process that is not deployed as a Java application. Instead, a system component is managed by Oracle Process Manager and Notification (OPMN) Server. Some of the system components are Oracle HTTP Server, Oracle Web Cache, Oracle Internet Directory, Oracle Virtual Directory, Oracle Forms Services, and so on.

The arrows indicate that data from the two components (Java and System/OPMN) may have database dependencies as well. For instance, SOA and OID require a Metadata Repository, whereas Forms and OHS do not. Customer Java apps has customer data, whereas OHS does not. SOA is managed through the Admin interfaces, whereas OID is managed through the OPMN interfaces (this is a slight oversimplification, but good enough for an overview). And, the database management can be Oracle or anything (for example, DB2, Sybase, Microsoft, and so on) and therefore is outside the control of the FMW environment as such.

Summary

In this lesson, you should have learned how to:

- Describe the Oracle Fusion Middleware environment
- Describe how WebLogic Server supports various Fusion Middleware Suites
- Describe how various Fusion Middleware Suites augment the functions of WebLogic Server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 2 Overview: Logging In to the Lab Environment

This practice covers the following topics:

- IP address assignments
- User IDs and passwords
- NoMachine (NX) configuration

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 2 Overview: Logging In to the Lab Environment

See Appendix A for the complete steps to do the practice.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

3

Defining Java Enterprise Edition Terminology and Architecture

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Explain the motivation behind distributed systems
- List the major components of the Java Platform Enterprise Edition 5* (Java EE) specification

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

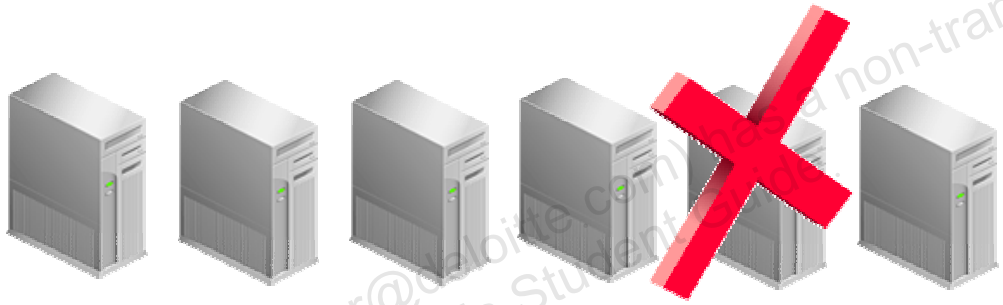
Scenario

The non-IT department managers of Example Corporation have asked you to put together a brief presentation outlining why you need a middle tier for computing. They grew up in the mainframe days where there was one giant host computer in the basement and many terminals connected to the host. They want to know if this middleware will help them process orders more quickly, or whether it just introduces another layer of complexity unnecessarily. Many terms have been used by Oracle marketing representatives in proposals, and some of the terms *sound* like terms they heard before, but in a different context, and perhaps with a different meaning. So a clarification of terms and acronyms would be in order for all involved.

* Wait until page 3-6 to ask whether WebLogic uses version 5 or 6 of Java.

Distributed Systems

- Distributed systems divide the work among several independent modules.
- The failure of a single module has less impact on the overall system, which makes the system more:
  - Available
  - Scalable
  - Maintainable



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Distributed Systems

The main goal of distributed systems is to better manage the complexity and resulting cost of providing highly available, scalable, and maintainable systems. Distributed systems can achieve these objectives by using many, more simple and self-contained systems that work together to perform the same functions as a single system that does everything.

- *Availability* is a measure of a system's ability to actively process client requests.
- *High availability* requires that a system is up and running as close to 24/7/365 as possible.
- *High availability* is achieved using load balancing and failover techniques.

The 24/7/365 availability is needed by applications such as ATM banking, stock and commodities trading, and e-commerce.

When a business grows beyond the available equipment, it is easier and more cost effective to add system components than replace the old ones. Distributed systems outshine in this context because they rely on independent system components that can be added or removed depending on the demands on the overall system.

This approach has the added advantage of lowering the initial costs of a new system because there is no need to purchase a big system up front that scales later. The costs can be deferred until the demands on the system warrant them (that is, you can purchase new system components as needed).

Distributed Systems (continued)

Maintainability (such as, software maintainability) is achieved by better managing the complexity of the software. This is done in the same way as for hardware: by using small, simple software components that work together to perform the same task as a single, large piece of software that does many things. For example, if the structure of your database has to change, the user software can remain unaffected.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

How Standards Help

- Many advantages of distributed systems come from standards.
- Standards allow:
  - Modularization of complex hardware and software
  - A larger portion of the project costs to go toward solving business software needs

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

How Standards Help

Many advantages of distributed systems come from the standardization of the hardware and software components that they are built on. The underlying complexity may not go away, but by having standards in different areas, many problems can be handled separately. For example, network standards have made communication between machines in a distributed system much easier, not because networking became any easier, but because you no longer have to manage all the complexity.

Software standards that deal with the complexities of application development are also very important because they can provide you with the same benefits at a higher level. They make it possible to focus on business-specific application components by removing many of the complexities associated with the underlying application infrastructure.

Java EE Standard

- Java Platform Enterprise Edition 5 (Java EE) helps you to overcome distribution liabilities.
- Applications deployed with Java EE technologies are:
  - Standardized
  - Adherent to specification guidelines
  - Written in Java
  - Deployable in any compliant application server
- Java Community Process (JCP) is the oversight (custodial) process for moderating Java's future direction.
  - http://jcp.org/en/home/index
  - http://jcp.org/en/introduction/faq

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java EE Standard

Java Platform Enterprise Edition 5 (Java EE, formerly known as J2EE), developed by Sun Microsystems, is a superset of the Java Platform Standard Edition (SE) 6 Java Development Kit (JDK). Java SE's product version number is 6 and the developer version number is 1.6.0. SE is primarily concerned with the Java Virtual Machine (JVM), whereas EE is primarily concerned with the application server or container.

Java SE

When most people think of the Java programming language, they think of the Java SE API. Java SE's API provides the core functionality of the Java programming language. It defines everything from the basic types and objects of the Java programming language to high-level classes that are used for networking, security, database access, graphical user interface (GUI) development, and XML parsing.

In addition to the core API, the Java SE platform consists of a virtual machine, development tools, deployment technologies, and other class libraries and toolkits commonly used in Java applications.

Java EE

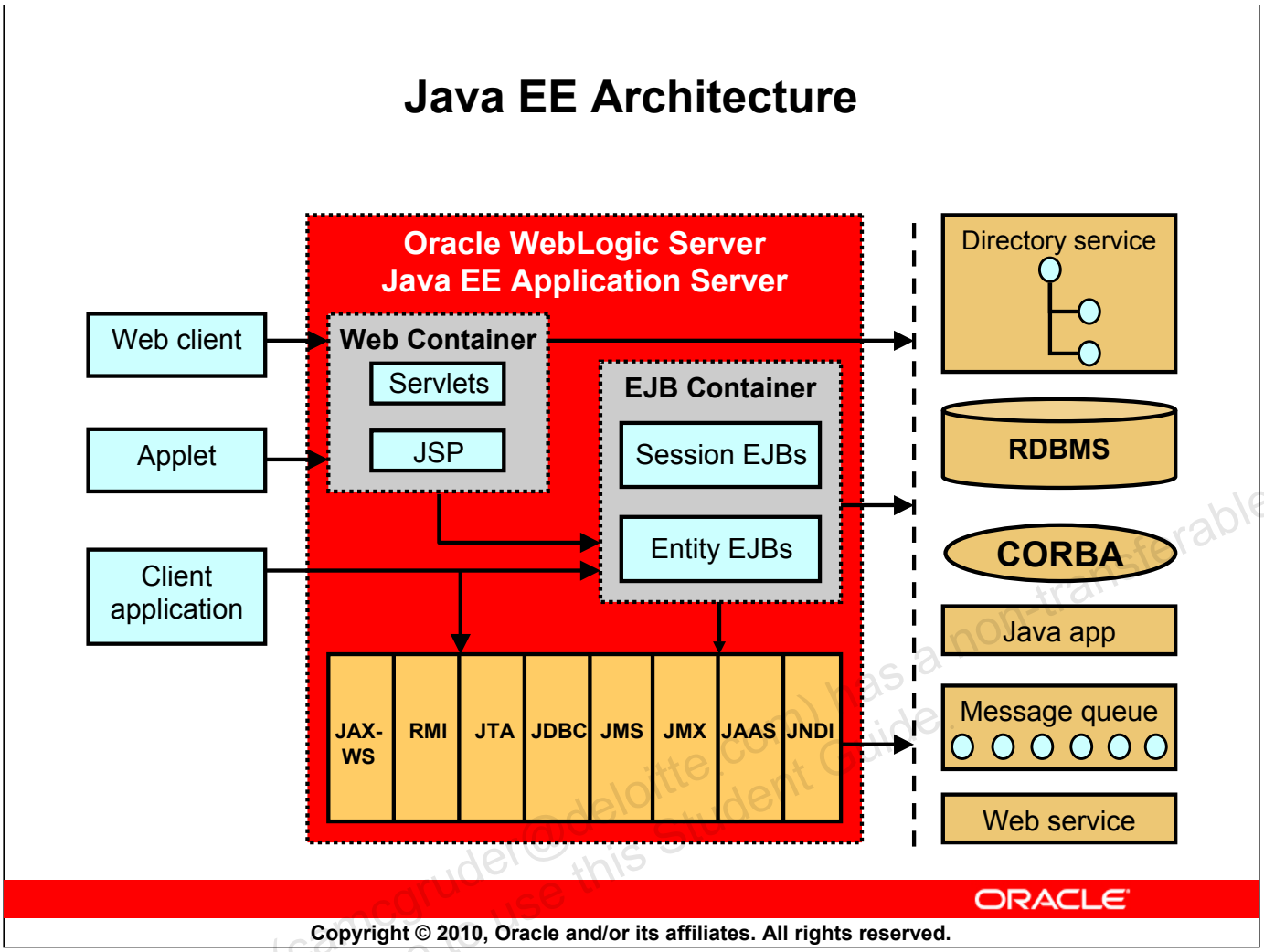
The Java EE platform is built on top of the Java SE platform. Java EE provides an API and run-time environment for developing and running large-scale, multitiered, scalable, reliable, and secure network applications.

Java EE Standard (continued)

Oracle WebLogic Server (WLS) is an application server that provides an implementation of the Java EE specification. By supporting all the specifications of Java EE, Oracle WebLogic Server can provide a scalable, fault-tolerant environment for the compliant applications to execute within. By standardizing the process for how pieces of software are developed, the Java EE specification has paved the way for companies such as Oracle to create application servers that automate many of the complicated distribution services that you had to tackle yourself in the past.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.



Java EE Architecture

The graphic in the slide shows the general architecture that must be supported by a Java EE platform. The JCP has defined a specification for classifying the entire platform and application servers as Java EE compliant.

The Java EE platform consists of the following:

- Application components, including application clients, applets, servlets and JSP, and EJBs
- Containers that provide run-time support for the components
- Resource manager drivers that implement network connectivity to an external resource manager
- A database that is used for storing business data

The Java EE specification also includes standard services such as HTTP, HTTPS, Java Transaction API, Remote Method Invocation-Internet Inter-ORB Protocol (RMI-IIOP), Java Database Connectivity (JDBC), Java Message Service (JMS), and Java Naming and Directory Interface (JNDI). Many of these technologies are discussed in the slides that follow.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) Has a non-transferable license to use this Student Guide.

Java Servlets

- A servlet is a Java program that executes on the server, accepting client requests and generating dynamic responses.
- The most prevalent type of servlet is an HttpServlet that accepts HTTP requests and generates HTTP responses.
- Servlets:
  - Do not just generate HTML
  - Can also be used to generate other MIME types, such as images

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java Servlets

A servlet is an independent thread of control that runs in the context of a server. The server acts as the “environment” in which the servlet lives. The server controls the life cycle, security, and execution of the servlets within its environment. That is, the server is responsible for the creation, access, and destruction of the servlet.

Because the servlet is a server-side resource, it has access to all the resources available on that server. This includes databases, transaction monitors, files, directory structures, and naming services. The servlet uses the available resources to generate a dynamic response for the client. The generated response is dynamic because the data being pulled from the resources is nondeterministic and unpredictable.

Multipurpose Internet Mail Extensions (MIME) types are used to describe the type of data being transferred in an HTTP request or response. MIME types include text, images, sound, and so on.

SimplestServlet.java

Creates HTML

```
package mypackage;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class SimplestServlet extends HttpServlet {
  public void service(HttpServletRequest request,
    HttpServletResponse response)
      throws ServletException, IOException {
       response.setContentType("text/html");
       PrintWriter out = response.getWriter();
       out.println("<HTML><BODY>");
       out.println("<H1>Hello, World!</H1>");
       out.println("</BODY></HTML>");
  }
}
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

SimplestServlet.java

There are better and certainly more complicated ways to write servlets, but this example illustrates a simple servlet. Tools such as JDeveloper and Eclipse assist in the writing and packaging of such Java objects.

JavaServer Pages (JSP)

- Are HTML documents that are interwoven with Java
- Provide a dynamic response that is based on the client's request
- Provide for the separation of responsibilities between the Web presentation and the dynamic content
- Are portable (write once, run anywhere)
- Compile and run as servlets
- May include JavaServer Faces tags

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

JavaServer Pages (JSP)

JavaServer Pages (JSP) are HTML documents that are embedded with special tags to insert Java code, thereby providing dynamic content. When the user makes a request on a JSP, the server executes the Java code and generates an HTML document, which is sent to the client. One of the benefits of JSP is the separation of responsibilities between the presentation to the client and the dynamic content. The Web artists can easily do their job and make the Web page look aesthetically pleasing, and then the Java programmers can add their code to make the page dynamic. Because JSP are written in the Java programming language, they are portable; they can be written once and deployed in any JSP-compliant server.

In addition, JavaServer Pages Standard Tag Library (JSTL) encapsulates as a single tag many of the functions that would have taken several tags. It also has an expression language for doing page development.

Related to JSP are JavaServer Faces. This is a server-side technology that supports user interface components, their states, and input validation. A JavaServer Faces page is a JavaServer Page with JavaServer Faces tags in it. The advantage of this is to separate the application behavior from the presentation.

realsimple.jsp

The Web container takes the JSP tags and creates HTML.

```
<!-- this is a comment -->
<HTML>
  <HEAD>
    <TITLE>My title</TITLE>
  </HEAD>
  <BODY>
    <H1>A big heading</H1>
    <P>Blah blah blah blah blah.</P>
    <% for (int i=0; i<3; i++) { %>
    <H3>Say it again, Sam.</H3>
    <% } %>
  </BODY>
</HTML>
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

realsimple.jsp

JavaServer Pages are a mix of plain old HTML and plain old Java plus some special tags to glue it all together. When the page in the slide is run, the `for` loop is replaced with three lines of "`<H3>Say it again, Sam.</H3>`" and the Java code is not shown in the final HTML source.

- **Expression:** It prints out:

```
<%= some-snippet-of-Java-code %>
<%= myEmployee.getName( ) %>
```

- **Scriptlet:** It is code that runs:

```
<% real-Java-statements %>
<% for (int i=0; i<50; i++) {
  System.out.println(i);
  } %>
```

- **Page import directive:** Do something at an OS file level:

```
<%@ page import="mypackage.junk.*" %>
```

JavaServer Pages and other related files are often packaged together into a Java Archive (JAR) or Web Archive (WAR), which is very similar to a ZIP or TAR file.

Enterprise JavaBeans (EJBs)

- Are distributed components written in the Java programming language
- Provide distributable and deployable business services (logic) to clients
- Have well-defined interfaces
- Are reusable across application servers
- Execute within a container that provides management and control services
  - Oracle WebLogic Server 10.3.3 supports the EJB 3.0 specification.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

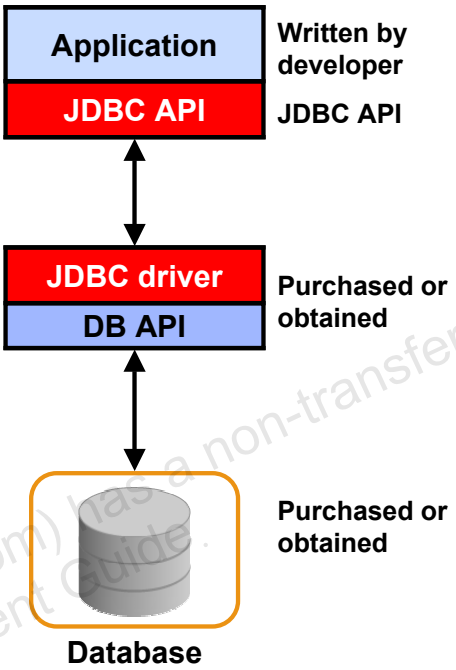
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Enterprise JavaBeans (EJBs)

EJBs make it possible for relatively new programmers to develop useful parts of highly complex, transaction-aware, and scalable enterprise applications. They also facilitate the creation of numerous front-end applications based on a common set of middleware (that is, EJBs). When you build applications using EJBs, you rely on tools such as Oracle WebLogic Server to do the hard work for you.

Java Database Connectivity (JDBC)

- The standard Java interface for accessing heterogeneous databases
- The specification that defines four driver types for connecting to databases



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java Database Connectivity

The diagram shows the flow between a database and a Java application through API drivers. The JDBC specification defines an interface for Java programs to use and an interface for which database vendors can develop custom drivers. Application developers will have to learn only a single JDBC API. The JDBC API is designed to work with any JDBC driver. Any proprietary driver that is developed by a database vendor will be compatible with a JDBC program.

Oracle implements two types of JDBC drivers:

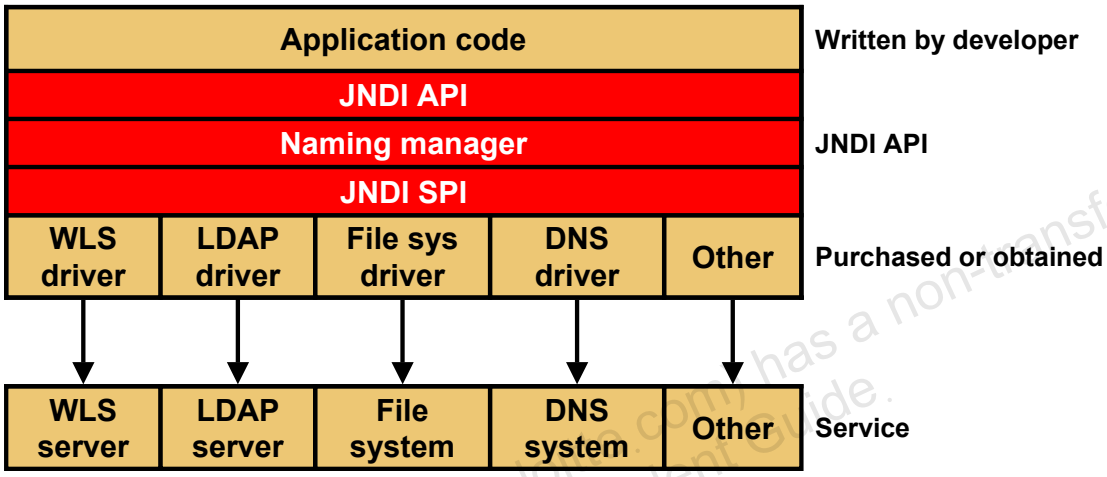
- **Thick** JDBC drivers built on top of the C-based Oracle Net client
- **Thin** (pure Java) JDBC driver to support downloadable applets

WebLogic supports Oracle's thin drivers.

Related to JDBC is the Java Persistence API (JPA), which provides object-to-relational persistence architecture for storing Java objects and Enterprise Java Beans (EJBs) in relational database tables.

Java Naming and Directory Interface (JNDI)

- Java API for accessing naming and directory services
- Built as a layer over DNS, LDAP, and so on



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

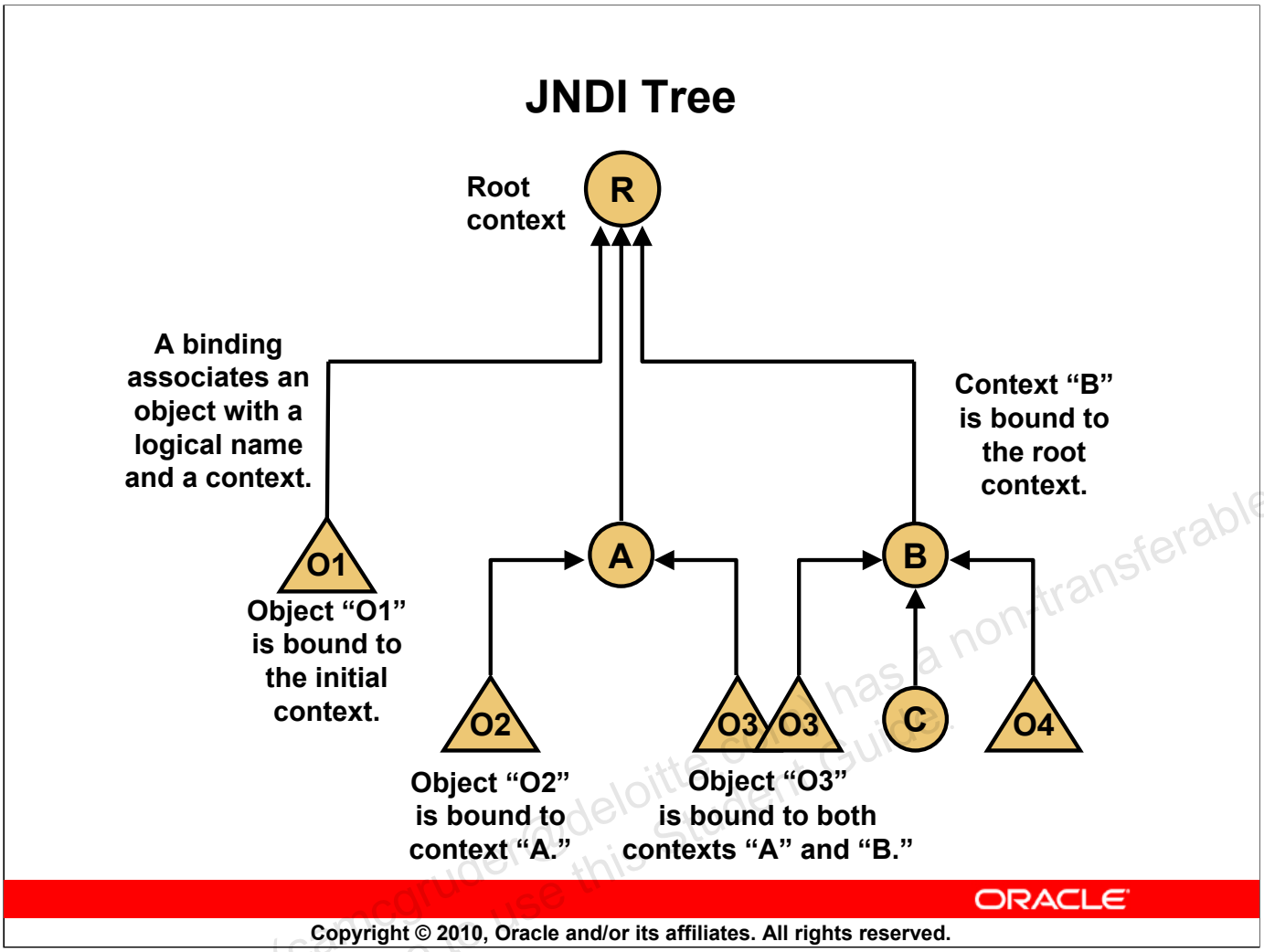
Java Naming and Directory Interface

The diagram shows the programming layers and API interfaces for managing named objects. Naming and directory services are used to hierarchically structure items that need to be made available to distributed programs. Naming and directory services provide lookup, search, and binding features to their clients. Clients can navigate the trees and contexts of a naming or directory service in search of the object they require. A variety of naming and directory services are available.

JNDI (pronounced “jenn-dee”) is an API and a standard that Java programs use to access existing naming or directory services. It is not a naming or directory service. It is merely the mapping. Different naming and directory services store objects in different ways. Oracle WebLogic Server has its own proprietary naming service that it uses to store configurable Java objects. Client programs connect to the naming service and download various Java objects that they need to use.

In Oracle WebLogic Server, JNDI serves as a repository and lookup service for Java EE objects, including:

- Enterprise JavaBeans (EJB) home stubs
- JDBC data sources
- JMS connection factories, queues, and topics
- Remote Method Invocation (RMI) stubs



Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

JNDI Tree

The terms associated with a JNDI tree include the following:

- **Context:** A node in the JNDI tree. It can contain only a list of objects and contexts.
- **Object:** A leaf in the JNDI tree, which is bound to a context. It cannot contain other objects or contexts.
- **Root context:** The context at the top in the entire tree
- **Initial context:** (not shown) A context that is chosen as the starting point for all future operations. This is somewhat like the "current directory" that you choose. It does not always have to be the root context of your directory structure. The initial context is merely the starting point that you select to traverse through the application.

Take a look at object O3 in the diagram in the slide. You bind it to two contexts: A and B. Does the object that is bound to two different contexts exist once or twice in the naming service? Or, worded differently, is the object accessed by value or by reference?

The answer is that JNDI stores objects by value, copying them into the tree. That is, there are multiple copies in the tree. To create a "by reference" binding, you need to use the LinkRef class.

Modifying an object under a context does not change the same object that is under a different context. In the tree in the slide, object O3 actually has two instances. The first instance lives under context A, and the second under context B.

JNDI Contexts and Subcontexts

- Subcontexts are referenced through dot or slash delimiters (`.` or `/`).
- Subcontexts must be created before objects are placed into them.
- Typically, when objects are bound to a JNDI tree, subcontexts are automatically created based on the JNDI name.
- Example:
  If the following context exists:
  `com.oracle.examples`
  You cannot bind:
  `com.oracle.examples.ejb.SomeObject`
  Without first creating:
  `com.oracle.examples.ejb`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

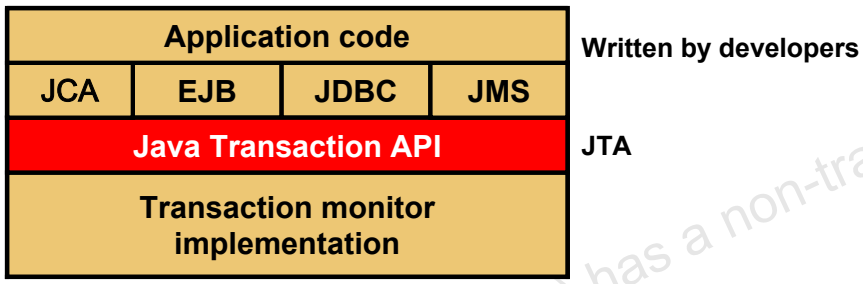
JNDI Contexts and Subcontexts

To avoid name collisions, it is recommended to make subcontexts. Subcontexts are referenced through simple dot delimitation (for example, the name `com.oracle.examples` versus simply `examples`). So if you have a JNDI object representing a JDBC data source such as Human Resources for Department 10 in the USA, rather than just naming it HR, you may want to name it with higher qualifiers (for example, `us.dept10.HR` as opposed to `uk.dept23.HR`).

You cannot insert an object into the naming service unless its subcontext exists. Suppose *`SomeObject`* refers to an object in the *`examples`* context, which is a subcontext of *`com.oracle`*. In this example, you cannot bind an object named `com.oracle.examples.ejb.SomeObject` unless the *`ejb`* subcontext is first created. JNDI will not automatically create the subcontext because the name of the bound object has a subcontext listed in it.

Java Transaction API (JTA)

JTA is a standard Java API for demarcating transactions within a program.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

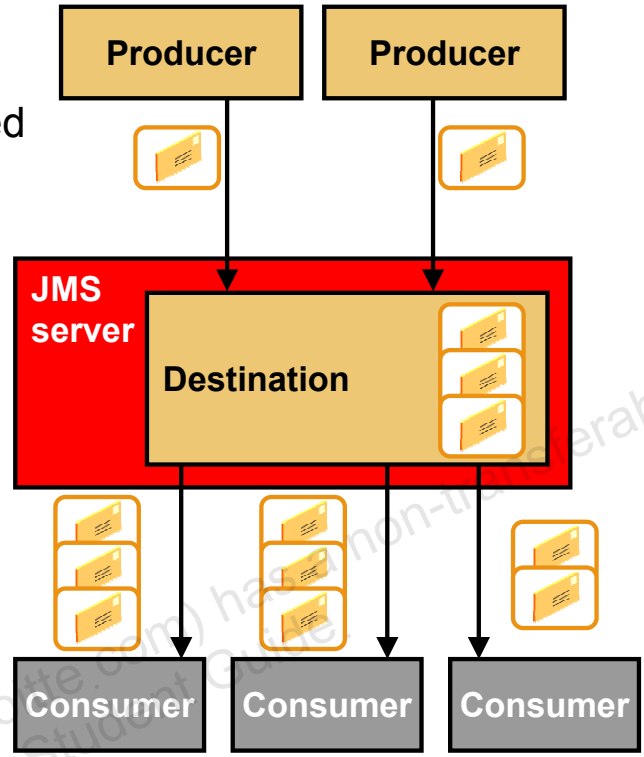
Java Transaction API

The diagram shows the programming layers and API interfaces for managing transactions as units of work that commit or roll back together. JTA specifies standard Java interfaces between a transaction manager and the parties involved in a distributed transaction system: the resource manager, the application server, and the transactional applications. The JTA specification was developed by Sun Microsystems in cooperation with leading industry partners in the transaction processing and database system arena. JTA is meant for bean-managed persistence.

WLS also supports Java Transaction Service (JTS), which is a lower-level interface specification for transaction managers. This interface, in fact, follows the JTA specification for managing transactions in Java and also supports the Object Transaction Service (OTS) specification to provide Common Object Request Broker Architecture - Object Request Broker (CORBA ORB)–type transaction management.

Java Message Service (JMS)

- JMS is a Java API for accessing message-oriented middleware.
- The interface supports:
  - Point-to-point model
  - Publish/subscribe ("pub/sub") model
  - Guaranteed message delivery
  - Transactional participation
  - Dynamically configurable services
  - Application- or system-scoped resources
  - Interoperability with other messaging systems



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java Message Service

The diagram shows the programming layers and API interfaces for managing messages end to end from producers to consumers. Enterprise messaging is now recognized as an essential tool for building enterprise applications. By combining Java technology with enterprise messaging, the JMS API provides a tool for writing enterprise applications that communicate with each other asynchronously.

Enterprise messaging provides a service for the asynchronous exchange of events and business data throughout an organization. The JMS API adds to this a common API and provider framework that enables the writing of portable, message-based applications in the Java language.

The common concepts of subscriber, publisher, producer, consumer, and queues are supported by all JMS technology–compliant messaging systems. The configuration decision for JMS is if and where to store messages—memory, database, or plain files—or not store them at all. This storage is known as persistence.

Java Authentication and Authorization (JAAS)

- JAAS is a Java-based security management framework.
- JAAS supports:
  - Single sign-on
  - A Pluggable Authentication Module (PAM)
- JAAS enables flexible control over authorization whether it is based on:
  - Users
  - Groups
  - Roles

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

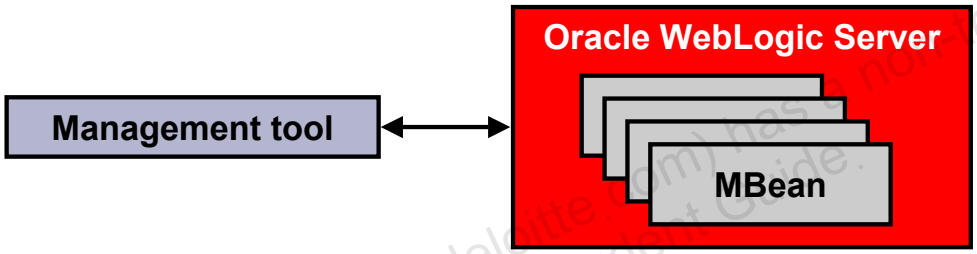
Java Authentication and Authorization

Single sign-on is a very important authentication feature specified by JAAS, because in a heterogeneous system, many authentication systems may exist. It is not acceptable to force a user to enter multiple passwords and to synchronize these passwords. The single sign-on framework specified by JAAS solves this problem. Oracle WebLogic Server by itself does not support single sign-on currently. However, there are several ways to achieve this using the Oracle Identity Management suite.

With the PAM framework, you can add multiple authentication technologies without having to change existing login services. You can use the PAM framework to integrate login services with different authentication technologies.

Java Management Extension (JMX)

- JMX:
  - Defines a standard infrastructure to manage a device from Java programs
  - Decouples the managed device from the management tools
- The specification describes MBeans, which are the building blocks of JMX.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java Management Extension

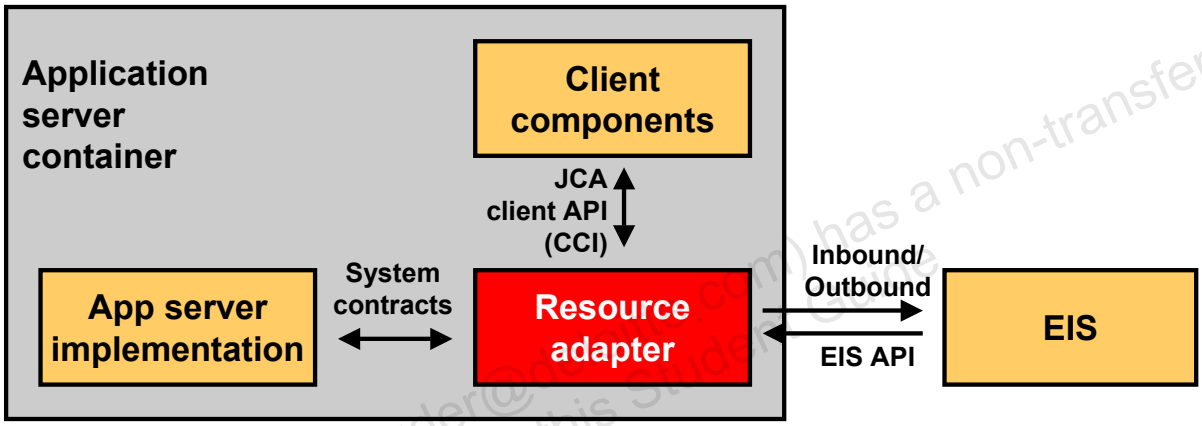
JMX is a standard way to automate the management of devices using Java as the control language. You can compare JMX to other technologies, such as JDBC. JDBC enables a database vendor to develop a product without worrying about how the database will be accessed. All that is needed is a JDBC driver, which acts as a contract between the vendor and the clients. Similarly, the client can switch from one database to another without worrying about compatibility.

For JMX, it is a similar situation. In theory, JMX enables decoupling of the managed device from its management software. The device vendor develops the management beans (MBeans) along with its device, so clients can automate the management of the device seamlessly.

WebLogic Server uses JMX extensively internally to handle the configuration and state of the system.

Java EE Connector Architecture (JCA)

- JCA connects Enterprise Information Systems (EIS) with resource adapters.
- Resource adapters can be deployed in a Resource Adapter Archive (RAR).



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Java EE Connector Architecture

Oracle's adapters are implemented as JCA resource adapters. JCA is a Java standard that provides a generic architecture to connect Java EE–compliant application servers to EIS. The current JCA version 1.5 standard supports bidirectional communication and can be deployed on any Java EE–compliant application server, including Oracle WebLogic Server and Oracle Application Server. You can use the uniform JCA APIs and contracts to programmatically connect to legacy systems instead of having to learn and use each EIS's API or other interface mechanism. You can think of JDBC as a special kind of JCA.

JCA features include:

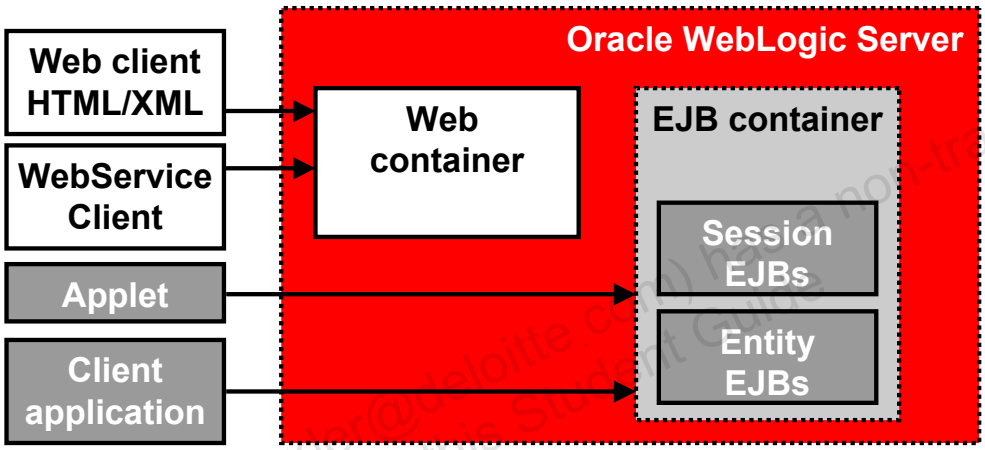
- A set of system-level contracts between an application server and EIS
- The Common Client Interface (CCI) that defines the API client components, which can be used to interact with the EIS through a JCA resource adapter
- A deployment and packaging mechanism for resource adapters
- Transaction and message inflow capabilities enabling asynchronous inbound or outbound communication, or both, with an EIS

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@oracle.com) has a non-transferable license to use this Student Guide.

Client Application

- The client application interacts with WLS through JRMP/T3, IIOP, and jCOM.
- The types of clients include:
  - Stand-alone Java applications
  - Applets within a browser



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Client Application

The diagram shows a Web client interacting with an EJB container. An application client is any client that can communicate with a distributed system. These clients can be heavyweight or lightweight, based on the amount of business processing that is performed on the local client. “Heavyweight” and “lightweight” are relative terms. Some people may differ on whether a browser application such as Java Swing could be heavyweight, or whether all browser applications are by definition lightweight. Although Web clients are technically application clients, a distinction is made so that you understand the context that is being discussed (you will know that Web client instantly implies HTTP communication). The WebService client is running the Simple Object Access Protocol (SOAP) protocol.

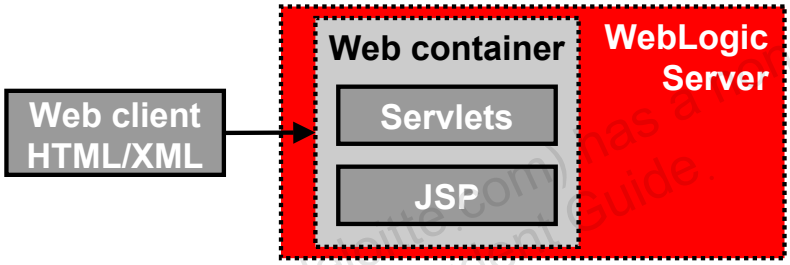
An EJB container is an environment that manages session beans (stateful or stateless), entity beans, and message-driven beans.

Note: The Oracle WebLogic Server implementation of the Remote Method Invocation (RMI) specification uses a proprietary wire-protocol, known as T3. JavaSoft’s reference implementation of RMI uses a proprietary protocol called Java Remote Method Protocol (JRMP).

jCOM allows bidirectional access between Java/J2EE objects and Microsoft Active X components.

Web Client

- A Web client interacts with Oracle WebLogic Server via HTTP using servlets or JSP.
- The types of Web clients include:
  - Browser
  - WebService (SOAP over HTTP)



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

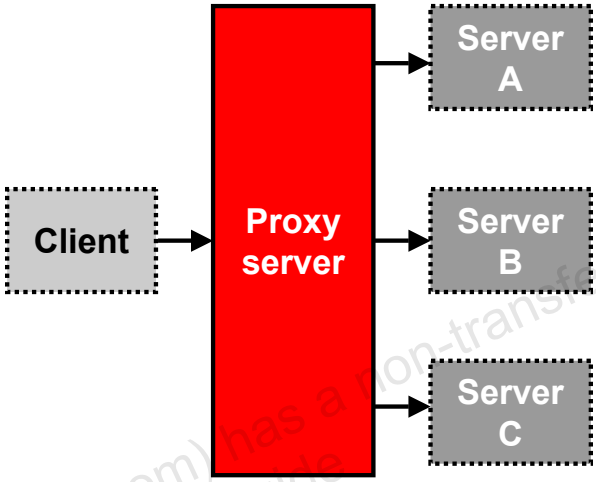
Web Client

The diagram shows a Web client interacting with a Web container. A Web client is a piece of software or computing device that communicates with a Web server over the HTTP protocol. Typically, a Web client receives HTTP packets that contain HTML pages. A browser is a piece of software that can interpret and display an HTML page in its correct format. Web clients interact with Oracle WebLogic Server by sending requests for pages. These page requests get mapped to the HTML pages, servlets, or JSP that are deployed on Oracle WebLogic Server.

Also, advanced Web clients can display XML pages. XML provides a more generic format for data that frees Web developers from many of the design constraints of HTML. Oracle WebLogic Server provides a wide range of services for processing XML. Do not assume that a Web client is a PC; it can be a phone or kiosk or any other kind of device.

Proxy Server

- The proxy server:
  - Forwards requests to other machines
  - Can be used as a level of indirection and security
  - Can be used to load-balance a system
- A reverse proxy is a Web page cache.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Proxy Server

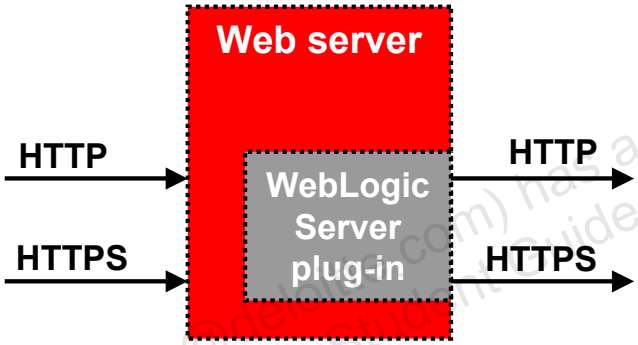
The diagram shows a proxy server distributing traffic among several servers. A proxy server is a piece of software that accepts a request on behalf of a client and forwards the request onto a machine that satisfies the request. The proxy server typically maintains some sort of mapping mechanism to determine which type of client requests gets forwarded to which servers. A proxy server decouples clients from the server that provides the implementation for the client. This level of indirection gives administrators and architects a level of security and the ability to reconfigure the back-end network without impacting the existing clients.

A reverse proxy (also known as a page cache) is a form of proxy that caches page requests in memory, and then when repeat requests come for the same pages, the cache looks for recently used URLs and returns those pages without having to fetch them again. This greatly speeds up the response time. In addition, reverse proxies can have a request filter (for example, allow if match, deny if match).

Web Server

Web servers:

- Provide Web content
- Communicate via HTTP, FTP, and so forth
- Can handle CGI requests
- Proxy some requests to application servers



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

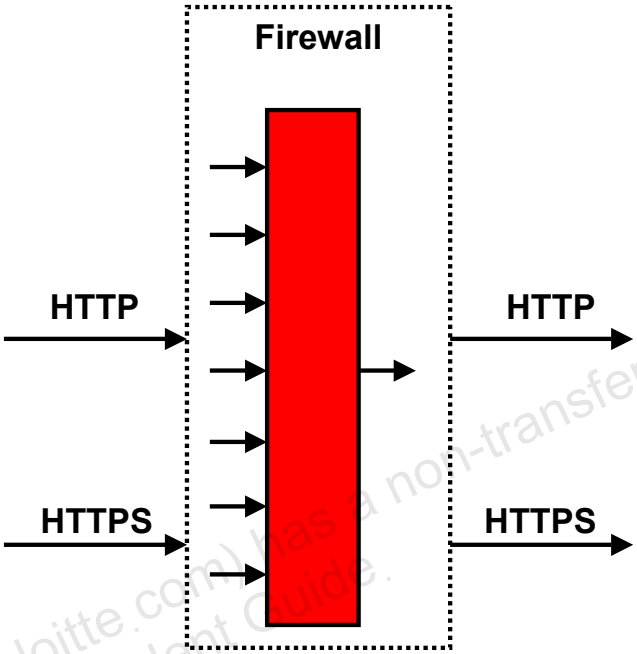
Web Server

A Web server is a software that communicates over the HTTP protocol. Modern-day Web servers are optimized for the rapid distribution of pages to many concurrent users. To accomplish this, a Web server may provide special file mappings, frequently accessed caches, or other features. Web servers also commonly handle Common Gateway Interface (CGI) requests. Many Web servers support some sort of technology in which a client-side page can invoke the operation of a server-side CGI program.

Web servers can also proxy a request that they receive to application servers that run the Java EE applications. Oracle WebLogic Server provides proxy plug-ins to popular Web servers that allow the Web server to proxy requests that are intended for the Oracle WebLogic Server elements. The Oracle WebLogic Server plug-in can also round-robin these requests to a variety of machines in a cluster, thereby providing a basic level of scalability. WebLogic Server includes a built-in Web Server. Oracle recommends the use of a separate Web Server such as Oracle HTTP Server (OHS) rather than the built-in Web server of WebLogic Server.

Firewalls

- Provide filtering, authorization, and authentication services
- Help keep out hackers
- Map port requests
- Can act as proxy servers
- Can decrease back-end network activity



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

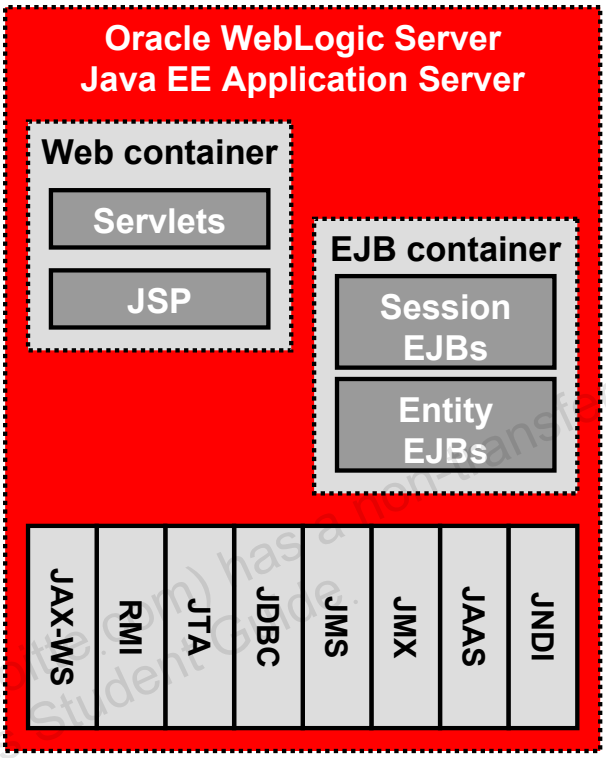
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Firewalls

The diagram shows a firewall passing and blocking certain traffic—in this case, Web traffic. Firewalls are a security measure. Typically, your client's equipment is behind a firewall and your Web server is behind another firewall. Often, in large applications, there is a firewall between your Web server and your application server. Physically, a firewall is often part of an IP router.

Application Servers

- Provide services that support the execution and availability of deployed applications
- Handle heavier processing chores than Web servers



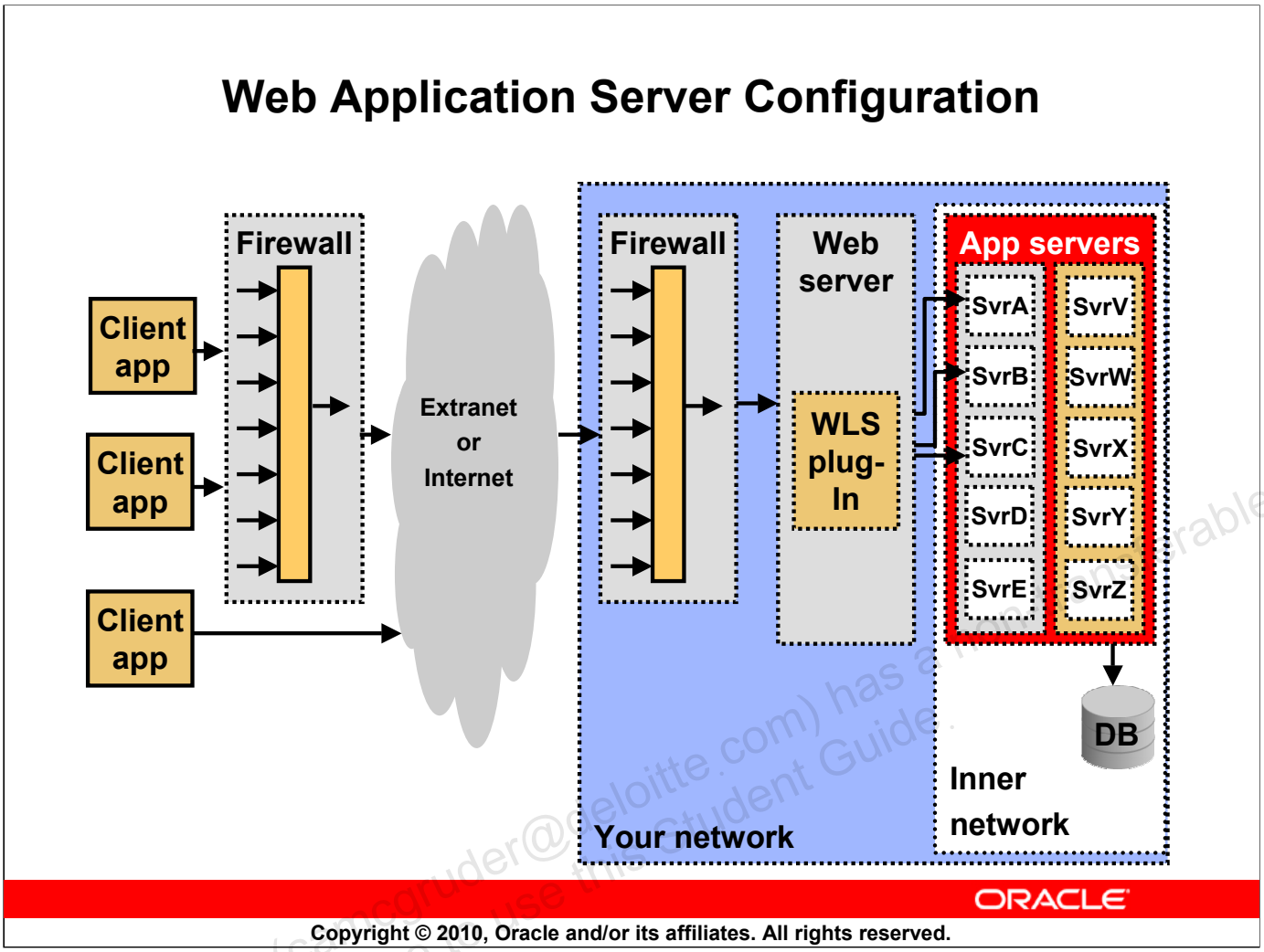
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Application Servers

The diagram shows WebLogic Server with several containers and services as blocks. An application server is a software that can service the requests of applications. You can think of an application server as the CPU for a large distributed system. An application server is a coordinator, facilitator, supporter, and manager of the distributed architecture activities of a distributed system. To perform these tasks, popular application servers have support for components, CGI programs, message-oriented middleware, transaction monitoring, persistence management, timing services, email support, and many other technologies that an enterprise application may require for full-featured support.

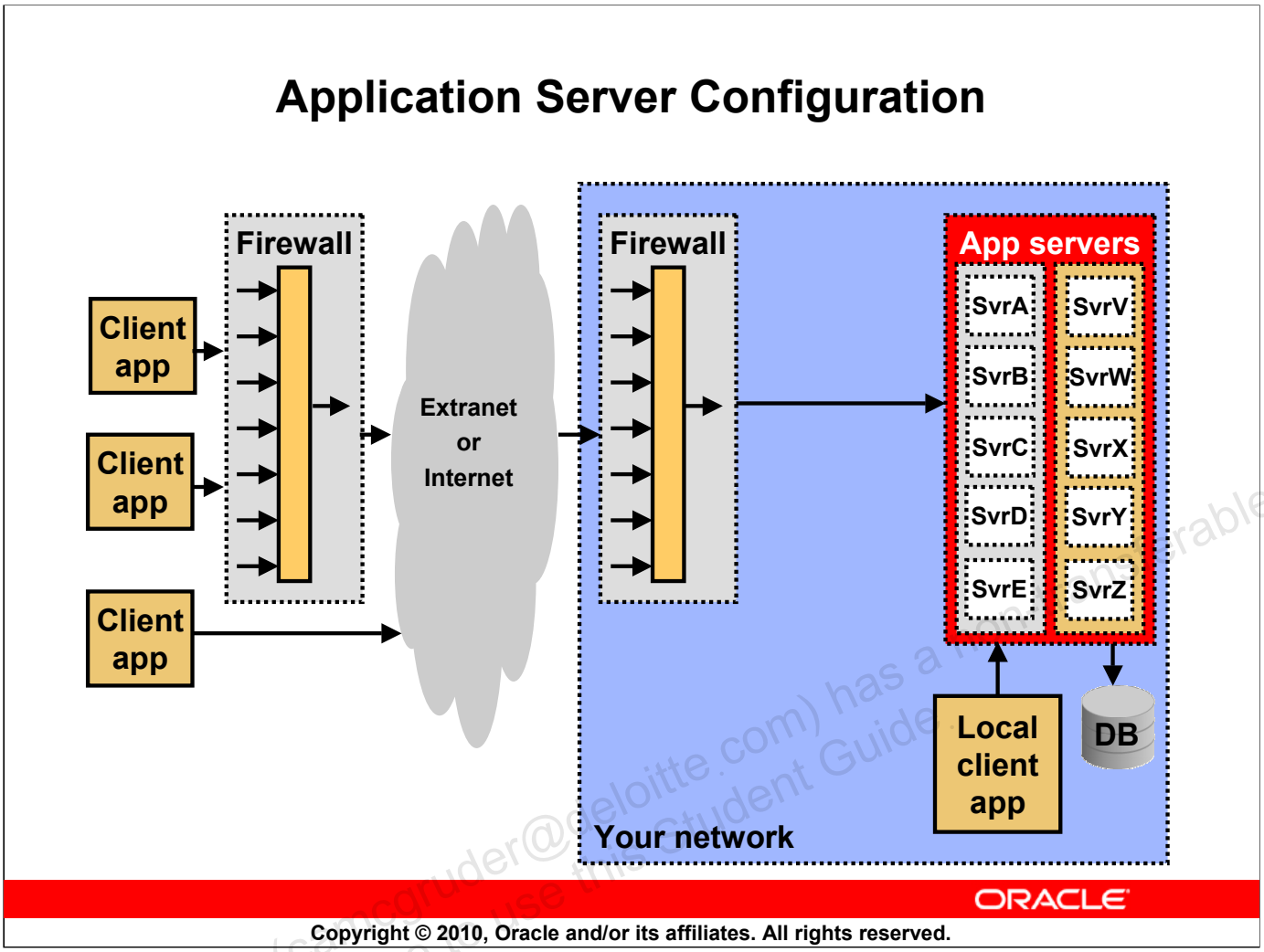


Web Application Server Configuration

The slide shows all the previous components together in one block diagram: clients, firewalls, Web servers, plug-ins, and applications servers. In this environment, you see the usual invisible firewalls. Often, your client has one and your Web server usually has one. Presuming that you have relatively sensitive data in your application server, you probably would add another firewall to further isolate your enterprise data.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.



Application Server Configuration

The slide shows all the previous components together in one block diagram: clients, firewalls, applications servers, and databases. Here, you see that there are still applications that do not run on browsers. They use the World Wide Web infrastructure to access your application server but they probably get to your server via some EJB that is exposed through a registry. The same service is available to local clients as well.

Firewalls, which normally are not shown in these diagrams, are shown here to emphasize that you cannot rely on just any protocol making a round trip through the system. That is why you have HTTP tunneling where other protocols are encapsulated in an HTTP wrapper so that they can get past the firewalls.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Quiz

Oracle WebLogic Server 10.3.3 is certified with JDK 1.6.

a. True

b. False

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

You can use Sun JDK, JRockit, or any other JDK 1.6 with Oracle WebLogic Server.

Summary

In this lesson, you should have learned how to:

- Explain the motivation behind distributed systems
- List the major components of the Java EE specification

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 3 Overview: Defining Terminology and Architecture

There is no practice for this lesson.

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Installing Oracle WebLogic Server 11*g*

4

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Explain Oracle WebLogic Server installation steps
- Install WebLogic Server using both the graphical user interface (GUI) and command line
- Describe the organization and contents of the WebLogic Server directory structure
- Navigate the WebLogic Server online and offline documentation

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

As the administrator of middle-tier computing for Example Corp, you install Oracle WebLogic Server on a single Linux machine using many of the default options to test out the basic functionality of simple configurations. To aid others in replicating your work, you make templates of the domain configurations you created. These are not copies as such, but templates to let others pick up where you left off.

Road Map

- Setting up an Oracle WebLogic Server environment using the GUI
- Setting up an Oracle WebLogic Server environment using the command line
- Specialized installations

ORACLE

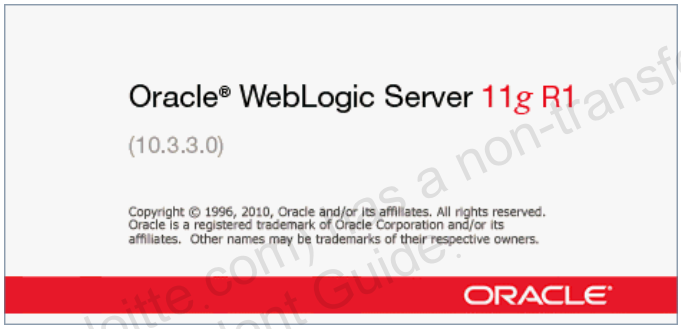
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has non-transferable license to use this Student Guide.

Oracle WebLogic Server Installation

- Install Oracle WebLogic Server in three different ways:
  - GUI mode (local and network)
  - Console mode
  - Silent mode
- Oracle WebLogic Server supports a number of platforms including:
  - Linux
  - Sun Solaris
  - HP-UX
  - Windows 2000, 2003 Server, XP



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle WebLogic Server Installation

Oracle WebLogic Server 10.3.3 installs on a variety of platforms. The installation uses the `wls1033_win32.exe`, `wls1033_ccjk_linux32.bin`, or `wls1033_linux32.bin` file to install on Windows, Sun/Sparc Solaris, HP-UX, or Linux platforms. The `_ccjk` version is for international alphabets.

The Oracle WebLogic Server installation program can be used in the following modes:

- **GUI mode:** The GUI mode installation is an interactive, GUI-based method for installing your software. It can be run on both Windows and UNIX systems.
  Note: If you want to run the graphical mode installation, the console attached to the machine on which you are installing the software must support a Java-based GUI. All the consoles for Windows systems support Java-based GUIs, but not all the consoles for UNIX systems do. If you attempt to start the installation program in graphical mode on a system that cannot support a graphical display, the installation program automatically starts the console mode installation.
- **Console mode:** The console mode installation is an interactive, text-based method for installing your software from the command line, on either a UNIX system or a Windows system.
- **Silent mode:** The silent mode installation is a noninteractive method of installing your software that requires the use of an XML properties file for selecting installation options. You can run the silent mode installation in either of two ways: as part of a script or from the command line. The silent mode installation is a way of setting installation configurations only once, and then using those configurations to duplicate the installation on many machines.

System Requirements

- Processor:
  - At least one 1 GHz CPU is recommended.
  - Intel and UNIX processors are supported.
- Hard disk drive:
  - A full installation requires approximately 2 GB of disk space.
  - The Linux value for file descriptors must be 4096 or greater.
  - Samples are optional (download from OTN).
- Memory:
  - A minimum of 2 GB RAM is recommended for WebLogic Server.
  - Consider the number of simultaneous users and sessions.
  - Consider in-memory programs, such as Coherence.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

System Requirements

The following are some of the basic system requirements for Oracle WebLogic Server 10.3.3:

- The Oracle WebLogic Server installer requires a Java Runtime Environment (JRE) to run. Oracle WebLogic Server is certified with JDK6.0. As part of the installation, it gives the option to install the JRockit JDK 6.0 version. As part of postinstallation, prefix the `bin` directory of the JDK to the `PATH` environment variable.
- The Oracle WebLogic Server installer requires a temporary location in which to unpack the files. Typically, the installer requires approximately 2.5 times the amount of temporary space that is required by the installed files.
- The Linux file descriptors are set in `/etc/security/limits.conf`.

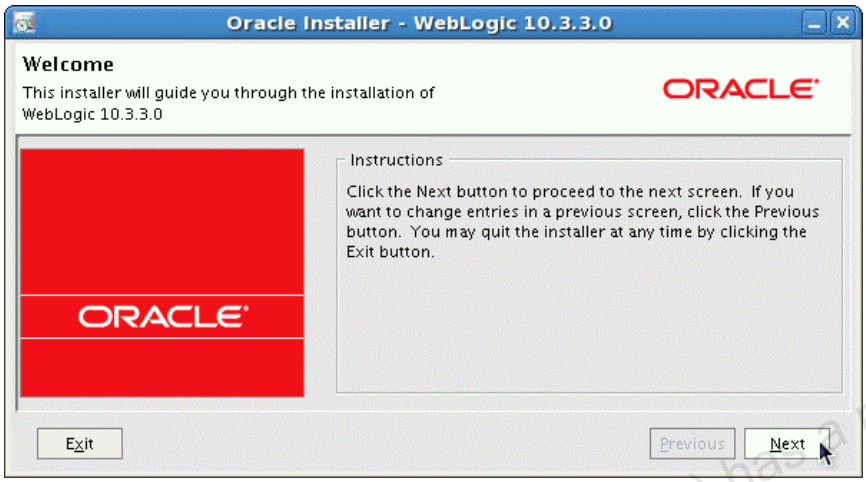
To specify the temporary directory location (*tmpdirpath*), perform the following steps:

1. Set the `TMP`/`TMPDIR` environment variable.
2. Set the `java.io.tmpdir` Java system property in the Installer command (any platform): `-Djava.io.tmpdir=`*`tmpdirpath`*

Note: In this release of WebLogic Server, users can choose which components of WebLogic Server they use. Specifically, this release allows users to choose whether the Enterprise JavaBeans (EJB), Java Message Service (JMS), and J2CA services are started when WebLogic Server is started. The benefit of excluding some services is reduced memory footprint and reduced startup time.

GUI Mode Installation

Start the installer (net or package).



`wls1033_linux32.bin`, `wls1033_win32.exe` or
`java -jar wls1033_generic.jar`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

GUI Mode Installation

The screenshot shows the splash page for the installation. The `_ccjk` addition to the name is for international alphabets (Chinese, Japanese, Korean). The GUI mode installation is the graphics-based method of executing the Oracle WebLogic Server installation program. It can be run on both Windows and UNIX systems. You cannot reinstall Oracle WebLogic Server on a previously installed version of Oracle WebLogic Server. You must first uninstall the Oracle WebLogic Server installation or install it at another location or run the upgrade installer.

Alternatively, if you have an existing JDK (such as the 64-bit version), you can type

```
java -jar ./wls1033_generic.jar
```

to start the installation of the WebLogic Server without any bundled JDK.

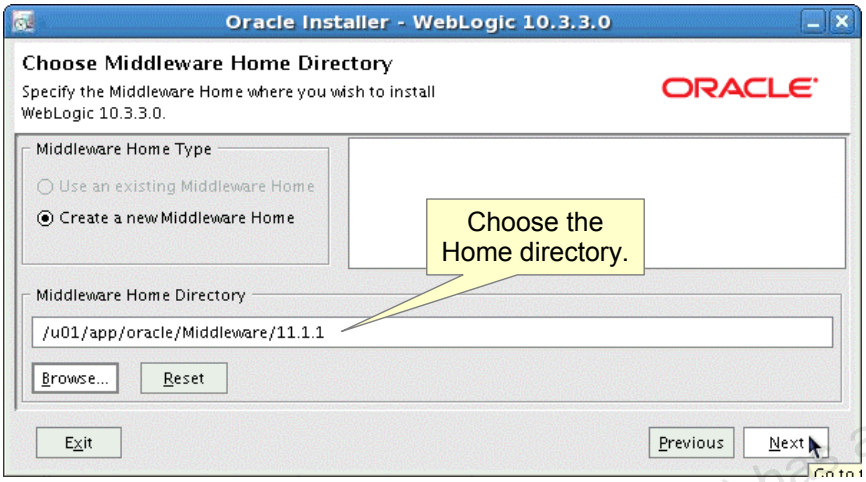
The Net installer involves initially downloading a small piece of software, selecting the installation options, and then downloading and installing only the components that you select. The Net installer eliminates the need to download a large single executable binaries file before actually installing the product.

Welcome Screen

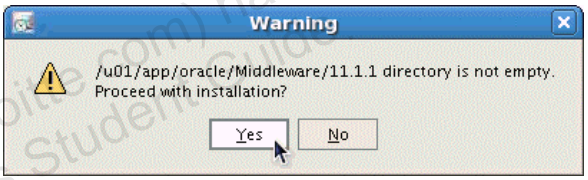
After the Welcome screen, click the **Next** button to proceed with the installation. You may cancel the installation at any time by clicking **Exit**.

Choosing or Creating a Home Directory

Browse or enter a destination.



You may see a warning if the directory already contains other applications, such as Web Tier.



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@yahoo.com) has a non-transferable license to use this Student Guide.

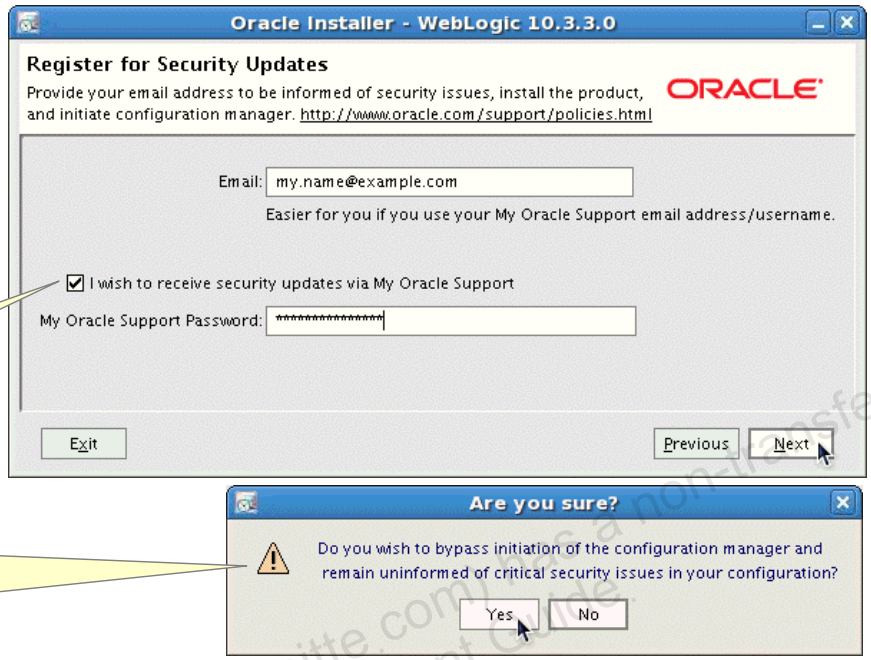
Choosing or Creating a Home Directory

The screenshot shows the choosing of the home directory for the installation. Specify the Oracle Middleware Home directory that will serve as the central directory for all the Oracle products installed on the target system. If you already have an Oracle Middleware Home directory on your system, you can select that directory (recommended) or create a new Oracle Middleware Home directory. If you choose to create a new directory, the Oracle WebLogic Server installer program creates the directory for you.

The best practice is to locate the WLS binaries away from the configuration data and away from the OS itself. In this example, the binaries are being installed on `/u01` (as though it were a separate physical disk), the configuration data will be on `/home`, and the OS on `/sys`, `/etc`, and other directories. The Oracle 11*g* database is also already installed on `/u01`.

Registering for Support

Optional, but highly recommended



If you deselect the checkbox…

…you will be warned that you will miss out on critical information in the future.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

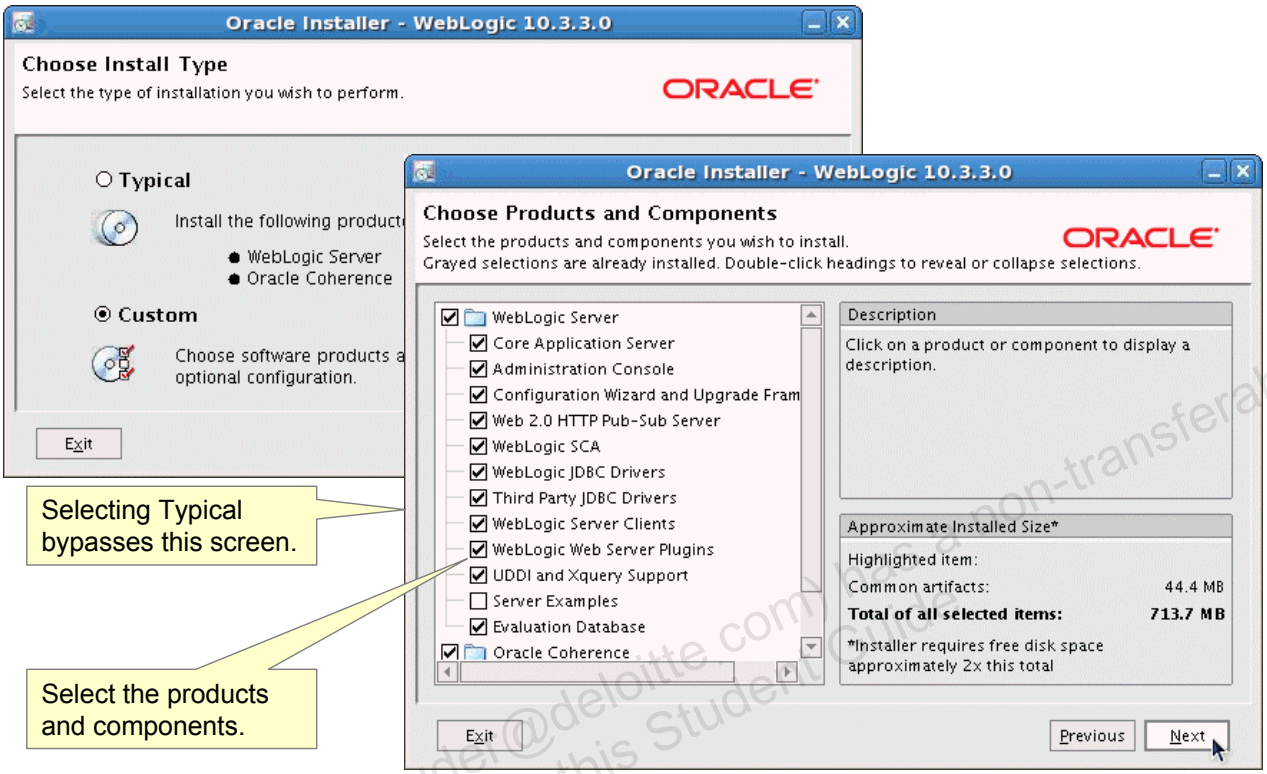
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Registering for Support

The screenshot shows how to enter your email and account information for online support. Follow the link http://www.oracle.com/support/policies.html on the screen to find out about current support policies.

Choosing an Installation Type and Products



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Choosing an Installation Type and Products

The screenshot shows choosing which components to install. A typical Oracle WebLogic Server 10.3.3 installation includes both the server components and the examples for the Oracle WebLogic Server platform. If you want to customize this installation, select the Custom installation option.

The Oracle WebLogic Server installer includes enhancements that enable you to select only the software components that you need, thus reducing the disk and run-time footprint. For Typical, Coherence is installed by default; for Custom, Coherence is also selected by default, but the Coherence Examples are not.

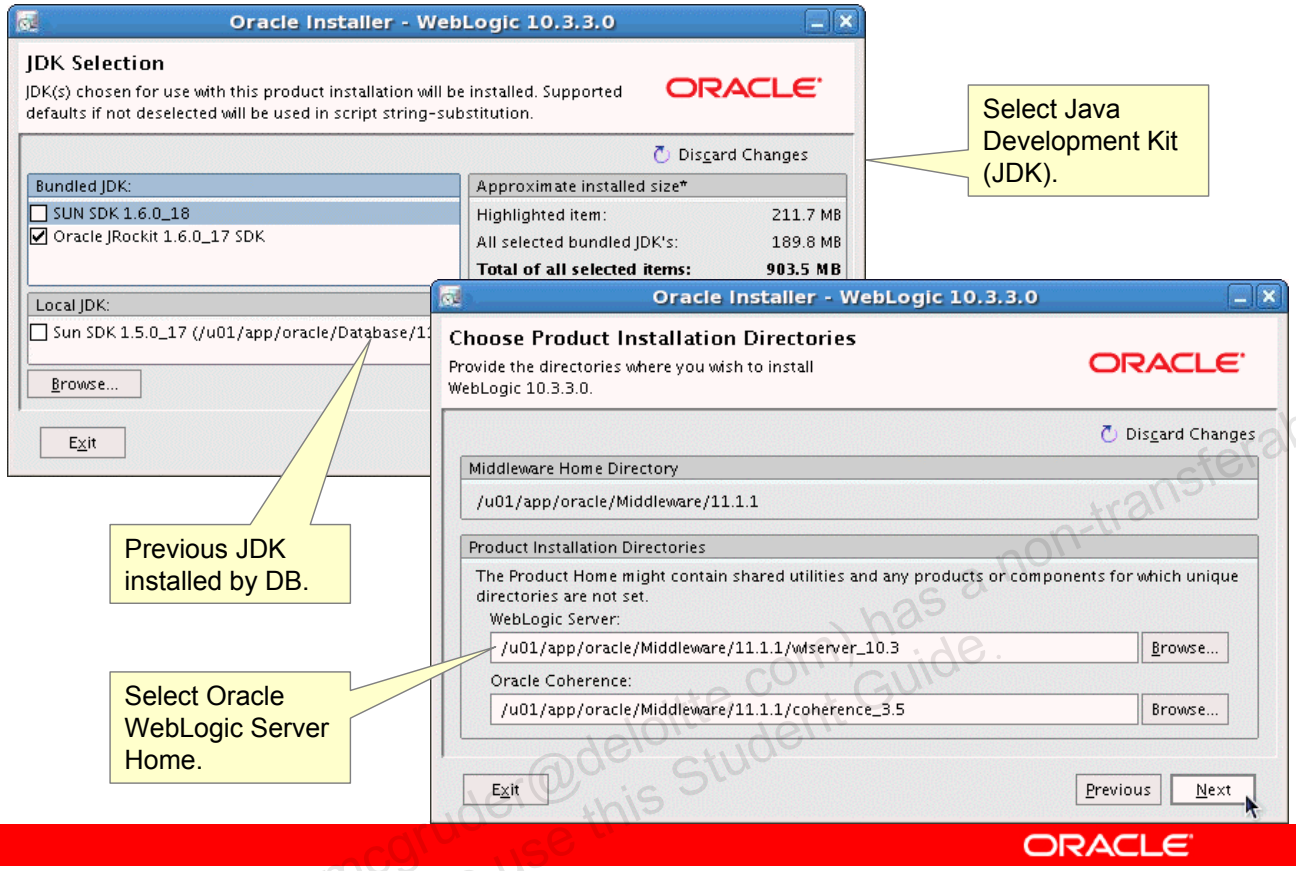
The server examples are not selected by default. If you want them in addition to the other items, select the check box here to install a medrec (Medical Records) sample that uses the Derby database. (In previous versions of WLS, the medrec sample application used the PointBase database.)

In the previous version WLS 10.3.0, the installation types were Complete or Custom. Also, there used to be an option for Workshop and WorkSpace Studio products in addition to WebLogic Server.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Choosing the JDK and Product Directory



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

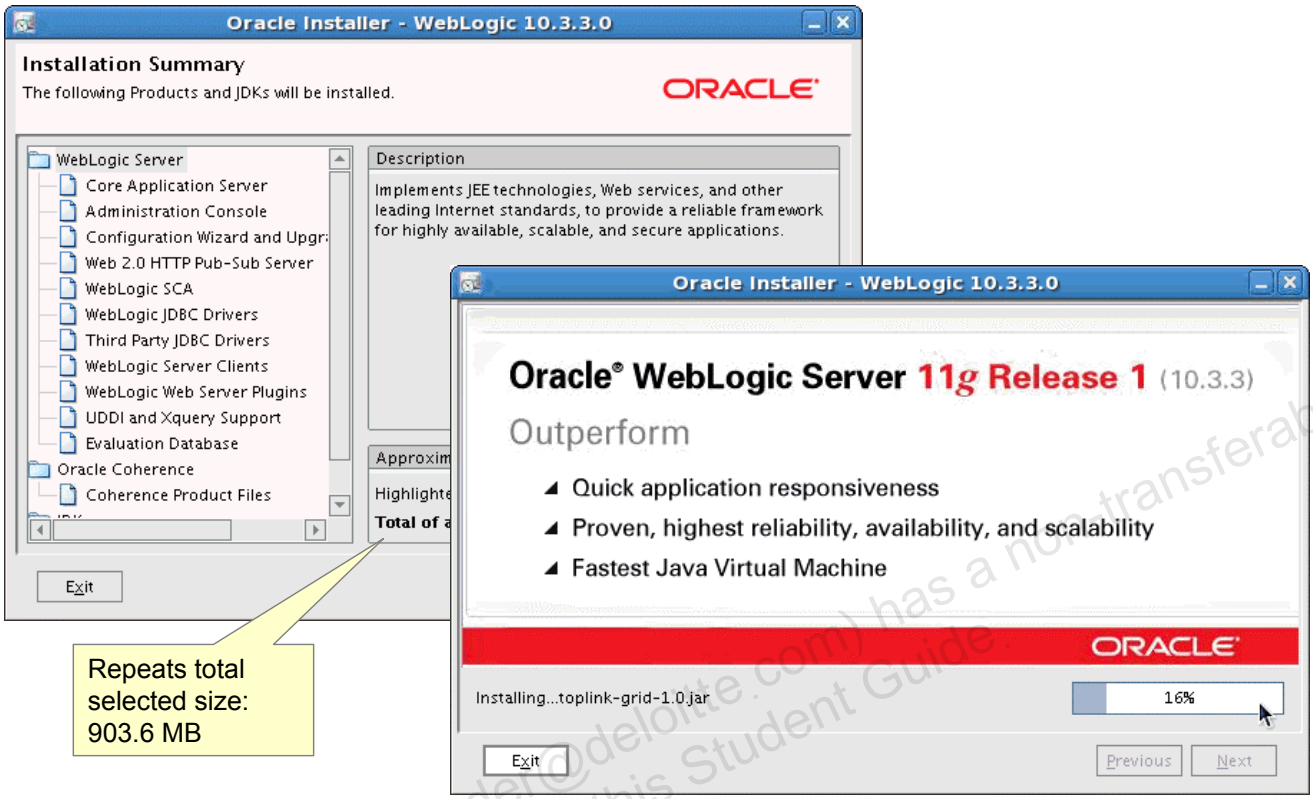
Choosing the JDK and Product Directory

The screenshot shows choosing JDKs. WebLogic Server requires a compatible Java Runtime Environment (JRE). Oracle WebLogic Server installers for Windows and Linux offer both the Sun and Oracle JRockit JDKs. By default, both are selected. If a suitable Java Runtime is already installed on your machine, you may deselect both JDKs and click Browse to select the Java Runtime you want to use. Your selected JDK location appears under Local JDK. This may be helpful if you have multiple WLS installations in the same computer.

Choose the product installation directory and select which JDK you want to register (JRockit or any certified JDK—for example, Sun JDK 1.6). There was a back-level SDK from a previous Oracle 11*g* Database installation in the default location of `<ORACLE_HOME>/jdk`.

Note: The Installer requires free disk space that is approximately twice the total shown.

Installation and Summary



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

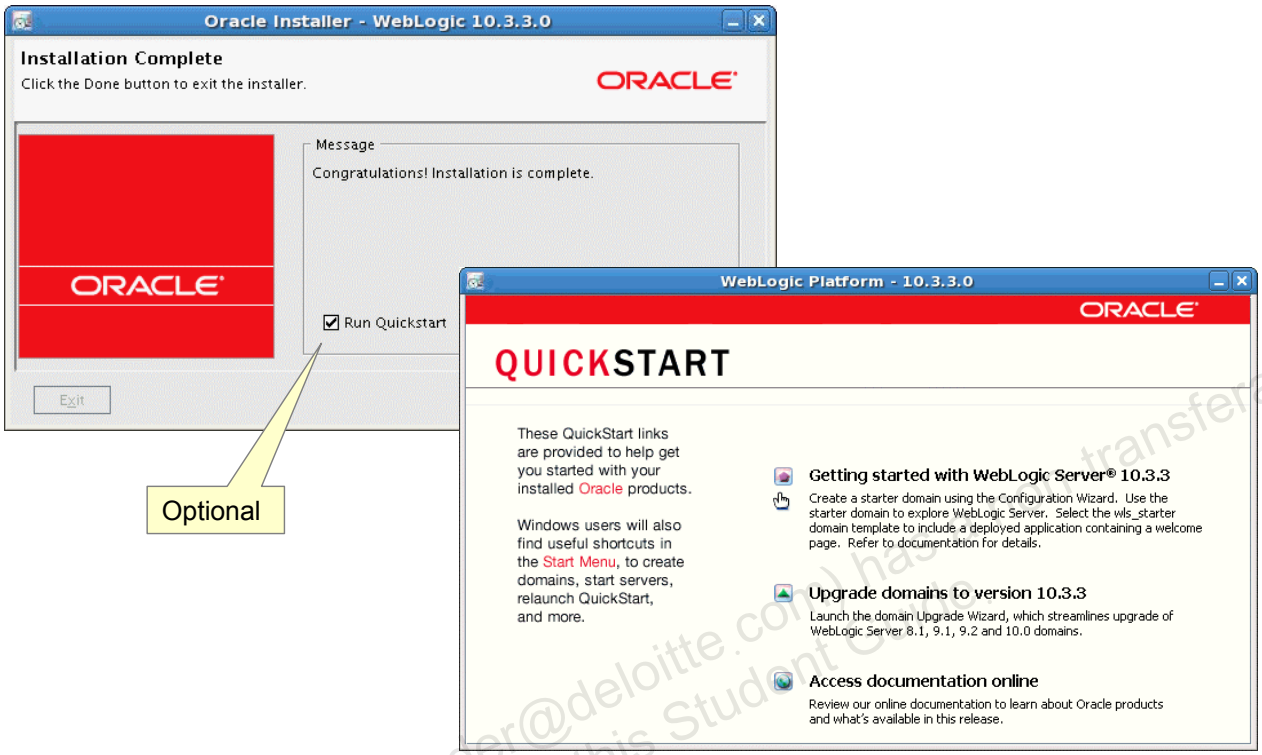
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Installation and Summary

The screenshot shows the progress bar as the installation happens. Note that only one of the two available JDKs is getting installed. When the product installation is completed using the graphical mode installation, the QuickStart application is launched automatically, by default. If you do not want to run QuickStart after the installation process is over, you can deselect the Run QuickStart check box in the Installation Complete dialog box shown in the following slide. QuickStart is not invoked for console mode or silent mode installations.

QuickStart is designed to help first-time users evaluate, learn, and use Oracle Products software. If you installed your software using a complete installation, or if you used a custom installation to install the examples, the sample domains that are installed are automatically configured to run with the PointBase database, a database that is installed with Oracle WebLogic Server.

QuickStart



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

QuickStart

The screenshot shows postinstallation options, such as QuickStart.

You can use QuickStart to perform the following operations:

- Upgrade the domains to 10.3.3.
- Access the documentation online. In particular, if you are going to upgrade, access *Oracle Fusion Middleware Upgrade Guide for Oracle WebLogic Server 11g Release 1 (10.3.3)* http://download.oracle.com/docs/cd/E14571_01/web.1111/e13754/toc.htm

After installation, you can launch QuickStart as follows:

1. On Window systems, select **Start > Programs > Oracle Products > QuickStart**.
   On UNIX systems, perform the following steps:
   a. Log in to the target UNIX system.
   b. Go to the `<WL_HOME>/common/quickstart` subdirectory of your Oracle Products installation. For example:
      ```
      cd /u01/app/oracle/Middleware/11.1.1/wlserver_10.3/
      common/quickstart
      ```
2. Enter the following command:
   ```
   ./quickstart.sh
   ```

Road Map

- Setting up an Oracle WebLogic Server environment using the GUI
- Setting up an Oracle WebLogic Server environment using the command line
- Specialized installations

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has not transferable license to use this Student Guide.

Console and Silent Mode Installations

- Console mode:
  - `wls1033_linux32.bin -mode=console`
  - The installation steps are similar to the GUI-based installation.
- Silent mode:
  - Create a `silent.xml` file.
  - `wls1033_linux32.bin -mode=silent -silent_xml=path_to_silent.xml`

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Console and Silent Mode Installations

If you attempt to start the GUI mode installation on a system that cannot support the graphical display, the installation program starts the console mode installation.

To start the console mode installation process for `wls1033_linux32.bin`, do the following:

- Navigate to the directory where you downloaded the installer and invoke the installation procedure by entering the following command:

  `wls1033_linux32.bin -mode=console`

  where the `wls1033_linux32.bin` file name is the name of the Oracle WebLogic Server installer.

  You can also include the `-log=full_path_to_log_file` option in the command line to create a verbose installation log.

For the silent mode installation, create a `silent.xml` file that defines the configuration settings normally entered by a user during an interactive installation process, such as the graphical mode or console mode installation. Use the sample `silent.xml` file from the documentation as a starting point to edit the file. Along with the `-mode` and `-silent_xml` parameters, you can also include the `-log=full_path_to_log_file` option in the command line to create a verbose installation log.

Postinstallation: Oracle Home

- `/u01/app/oracle/Middleware/11.1.1/`: Oracle Home
- `registry.dat/registry.xml`: Record of all Oracle Middleware products
- `utils`: Additional or utility JAR files
- `wlserver_10.3`: Oracle WebLogic Server 10.3.3 Home
- `logs`: Installation logs
- `modules`: Modules (`.jar`) installed in Oracle Home

```
[oracle@wls-sysadm /]$ ls /u01/app/oracle/Middleware/11.1.1
coherence_3.5        logs      oracle_common  user_projects  wlserver_10.3
domain-registry.xml  modules   registry.dat   utils
jrockit_160_17_R28…  ocm.rsp   registry.xml   webtier

[oracle@wls-sysadm /]$ ls /u01/app/oracle/Middleware/11.1.1/wlserver_10.3/
common   inventory   L10N   server   sip   uninstall
```

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Postinstallation: Oracle Home

Oracle WebLogic Server installs into a root directory, which is referenced as the environment variable <`ORACLE_HOME`> in script files that you select during installation. Under this directory, Oracle WebLogic Server sets up a `wlserver_10.3` directory where most of the content is stored. Both 10.3.0 and 10.3.3 versions use the same directory name of 10.3. Also in <`ORACLE_HOME`> are the Oracle modules, installation registration information, and `utility` directories.

Registry

The `.dat` file is binary and the `.xml` file is readable. These contain records of some of the installed Fusion Middleware products, such as, WebLogic Server, Coherence, and JRockit. This is analogous, but not identical, to `oraInventory` for the database.

Note: By default, the installer creates the Oracle WebLogic Server home directory under the Oracle Home directory. But the Oracle WebLogic Server Home does not have to be installed there. Home can be located anywhere. Oracle WebLogic Server home is referred to as the <`WL_HOME`> environment variable.

The `samples` directory is no longer installed by default, it is an option under Custom installation.

Oracle WebLogic Server Directory Structure

- `common`: The files shared by the Oracle WebLogic Server 10.3.3 components, including the template JAR files used by the Configuration Wizard when creating domains
- `samples`: Sample code and resources
- `server`: Server software components (executables, database files, XML JAR files, alternative JDBC drivers, Oracle WebLogic Server JAR files, and plug-ins)
- `uninstall`: The code required to uninstall Oracle WebLogic Server 10.3.3

```
[oracle@wls-sysadm /]$ ls /u01/app/oracle/Middleware/11.1.1
coherence_3.5        logs      oracle_common  user_projects  wlserver_10.3
domain-registry.xml  modules   registry.dat   utils
jrockit_160_17_R28…  ocm.rsp   registry.xml   webtier

[oracle@wls-sysadm /]$ ls /u01/app/oracle/Middleware/11.1.1/wlserver_10.3/
common   inventory   L10N   server   sip   uninstall
```

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmccrudder@dubitte.com) has a non-transferable license to use this Student Guide.

Oracle WebLogic Server Directory Structure

The `wlserver_10.3` directory is the location for most of the content that is distributed with Oracle WebLogic Server.

The common directory contains files shared by the product components, including scripts used for setting the environment attributes that are common to all the WebLogic domains running on the machine, the template JAR files used by the Configuration Wizard and WLST offline when creating domains, and the evaluation software from third-party vendors.

The `samples` directory contains sample code, resources, and preconfigured sample domains that are designed to help you to learn how to develop your own applications by using the product software. The sample domains are organized by the components installed on the system. For example, the server folder contains the source code for examples and the medrec (Medical Records) sample applications that use the PointBase (BEA) or Derby (Apache) database. Samples are available in the Custom installation.

The `server` directory contains the core files that are needed to run Oracle WebLogic Server 10.3.3. First, the `bin` directory is the location for all Windows convenience programs, native performance packs (threading, sockets, and security), and shared libraries that are needed for the Web server proxy plug-ins. Also, the `bin` directory contains the executables needed for setting up Oracle WebLogic Server as a Windows service.

Oracle WebLogic Server Directory Structure (continued)

`L10N` is part of language localization. In many cases it is empty.

The `ext` directory contains the XML JAR files and also a JDBC subfolder, which contains classes for third-party JDBC drivers. The `lib` directory contains the JAR files needed to run Oracle WebLogic Server, as well as the libraries for UNIX systems to support JDBC drivers, Web server proxy plug-ins, and other native code packages.

The `uninstall` directory contains the executables necessary for removing Oracle WebLogic Server from the existing computer.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Setting Environment Variables

- Run `setWLSEnv.sh` under `<WL_HOME>/server/bin` to set `WL_HOME`, `JAVA_HOME`, `PATH`, and `CLASSPATH`.
- `setWLSEnv.sh` makes a call to `commEnv.sh` under `<WL_HOME>/common/bin` to set common environment variables, such as `ORACLE_HOME`, `ANT_HOME`, and `DERBY_HOME`.
- Check the version of JDK:
  - `java -version`

```
[oracle@wls-sysadm /]$ java -version
java version "1.6.0_17"
Java(TM) SE Runtime Environment (build 1.6.0_17-b04)
Oracle JRockit(R) (build R28.0.0-679-130297-1.6.0_17-20100312-2128-linux-ia32,
     compiled mode)
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Setting Environment Variables

Oracle WebLogic Server is installed in a root directory, which is referenced as the `<ORACLE_HOME>` environment variable in script files that you select during installation. Under this directory, Oracle WebLogic Server sets up a `wlserver_10.3` directory where most of the content is stored. Also in `<ORACLE_HOME>` are the Oracle modules, installation registration information, and `utility` directories. Be aware that there can be multiple `<ORACLE_HOME>`s: one for each Oracle product. Because of this, do not add the variable to the profile, but rather set it as needed in scripts.

Make sure that the Java version is picked up from the latest `PATH`, not necessarily the previous `PATH` before the installation. You may need to alter the `PATH` statement to place the WebLogic Server `/bin` (version 1.6.0) ahead of the Oracle 11*g* Database `/bin` (version 1.5.0). In Linux, to see the `PATH` in use, you can also enter:

```
which java
```

PointBase is a small database included with WebLogic Server. If you are using Oracle Database, set those environment variables such as `<ORACLE_SID>`.

Defining Environment Variables

- OS specific:
  - Windows: `%SOME_VAR%`
  - UNIX: `$SOME_VAR`
  - Generic: `<SOME_VAR>`
- Kinds of environment variables:
  - Location variables: `*_HOME`
  - Path variables: `*PATH`
  - Names
  - Miscellaneous
- Scope:
  - Preinstallation
  - Running session

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Defining Environment Variables

Some Oracle installers require some variables to be set before the installation can proceed. These variables can be set in the user's profile. You can see all the variables that are already set by entering `env` for UNIX or `set` for Windows at a command prompt or you can see individual ones by entering `echo <SOME_VAR>` where the syntax differs slightly by operating system. File names differ by operating system (slashes versus backslashes, C: and D: drives versus mount points, reserved characters, and so on). Remember that UNIX is case-sensitive.

The kinds of environment variables that are used fall into four broad categories:

Locations: These are where things are stored on disk: binaries, scripts, configuration data, work areas, and so on. The installation program prompts for many of these locations while suggesting defaults. Often, the locations are built one on the next, so that if `<ORACLE_BASE>` is defined as `/a/b/c`, `<ORACLE_HOME>` might be defined as `<ORACLE_BASE>/d/e` so that `<ORACLE_HOME>` resolves to `/a/b/c/d/e`. Occasionally, (very rare) you may see a location defined with a missing intermediate directory, so that `<VERY_RARE_EXAMPLE>` might be defined as `<ORACLE_BASE>../d/e`, which resolves to `/a/b/d/e` (notice "`..`" causes the missing "`c/`"). Some examples of location variables are `HOME`, `WEBLOGIC_HOME`, `WL_HOME`, `MW_HOME`, `BEA_HOME`, `ORACLE_HOME`, `JAVA_HOME`, `DERBY_HOME`, and so on. Some variables are equivalent to each other, and exist only for backwards compatibility (for example, `BEA_HOME` is the same as `MW_HOME`, and `HOME` is the same as ~ [tilde] in UNIX).

Defining Environment Variables (continued)

Paths: When searching for something in directories, what is the order that the system should search? Executables use the `PATH` variable and Java uses the `CLASSPATH` variable. You can see the order that will be used by entering `PATH` in Windows or `echo $PATH` in UNIX. You can test this in UNIX by entering `which java`, which will show the path that would be used to run that executable. Note that UNIX does not assume that you want to look in the current directory. So to run `someapp` in the current directory, if that directory was not in the `PATH`, enter `./someapp`. This syntax changes when you want to run a script that alters session variables. Some examples of path variables are `PATH`, `PATCH_PATH`, `CLASSPATH`, `PRE_CLASSPATH`, `POST_CLASSPATH`, `PATCH_CLASSPATH`, and so on.

Names: Variables that are not related to physical directories indicate the logical operations at a given time. Some examples of name variables are `ORACLE_SID` (the name of the database service identifier or instance), `SERVER_NAME`, and so on.

Miscellaneous: These variables are all of the others that are defined while the WebLogic Server is running. They are properties, options, and arguments. Some examples of these kinds of variables are: `PRODUCTION_MODE`, `PROXY_SETTINGS`, `USER_MEM_ARGS`, `CLUSTER_PROPERTIES`, `JAVA_PROPERTIES`, `JAVA_OPTIONS`, and so on.

Preinstallation: You need to define some of these variables before installation so that the installer can know the defaults of where to put files. These variables are often stored in the user profiles (`.profile` or `.bash_profile` for Linux or for Windows in My Computer > Properties (System Properties) > Advanced > Environment Variables.) Some of the installation information from previous products is stored in Oracle Inventory at `<ORACLE_BASE>/../oraInventory` (notice the capital "I") or in the Windows registry. The Universal Installer (`runInstaller` or `setup.exe`) can pick up this information. The WebLogic Server does not use the Universal Installer. The complete list of suggested (none are required) preinstallation variables is `JAVA_HOME`, `ORACLE_BASE`, `ORACLE_HOME`, `ORACLE_SID`, `BEA_HOME`, `WEBLOGIC_HOME`, and `PATH`. Often, these exist already (with the exception of `BEA_HOME`) as the result of an Oracle Database installation. Be aware that multiple `ORACLE_HOME` variables may exist for other products.

Running Session: The largest set of variables is created only when the WebLogic Server is running. Most of the scripts to start the servers call the three scripts that set these variables. The main environment variable scripts are `commEnv`, `setWLSEnv`, and `setDomainEnv`. Together, they set over 30 environment variables. The contents of these scripts is created by the installation itself. To run these scripts in a UNIX session, you use a slightly different syntax than a regular script:

```
. ./setWLSEnv.sh
```

or

```
source ./setWLSEnv.sh
```

Note the leading period and space in the first form. For more information about these three scripts, refer to the lesson titled, "Configuring a Simple Domain."

Examples

In Linux, you can set an environment variable `ORACLE_SID` to a value of `orcl` in a script using the `export` command:

```
export ORACLE_SID=orcl
```

In Windows, go to My Computer > Properties > Advanced > Environment Variables > New or Edit.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can Recognize Unauthorized Reproduction. Oracle has a non-transferable license to use this Student Guide.

List of Environment Variables and Their Meanings

- BEA_HOME
- MW_HOME
- WL_HOME
- PATH
- CLASSPATH
- JAVA_HOME
- JAVA_OPTIONS
- JAVA_VENDOR
- JAVA_VM
- JAVA_USE_64BIT
- LD_LIBRARY_PATH
- LIBPATH
- SHLIB_PATH
- MEM_ARGS
- SERVER_NAME
- WEBLOGIC_CLASSPATH
- CLASSPATHSEP
- PATHSEP
- POINTBASE_HOME
- POINTBASE_TOOL
- POINTBASE_CLASSPATH
- POINTBASE_CLIENT_CLASSPATH
- PRODUCTION_MODE
- PATCH_CLASSPATH
- PATCH_LIBPATH
- PATCH_PATH
- WEBLOGIC_EXTENSION_DIRS
- USER_MEM_ARGS
- PRODUCTION_MODE
- DOMAIN_PRODUCTION_MODE
- ANT_HOME
- ANT_CONTRIB
- PRE_CLASSPATH
- POST_CLASSPATH
- PRE_PATH
- POST_PATH
- DERBY_HOME
- DERBY_OPTS
- DERBY_TOOLS

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

List of Environment Variables and Their Meanings

The following environment variables are set or used by `commEnv`, `setWLSEnv`, and `setDomainEnv`. Usually, you do not have to call these scripts directly; they are called from other scripts that start servers and other processes. The main ones you care about are the first six.

`BEA_HOME`: Is the home directory of all your BEA installations. `BEA_HOME` has been deprecated, but it continues to exist for purposes of backward compatibility. Use `MW_HOME` instead. It is identical to `MW_HOME`.

`MW_HOME`: Is the home directory of all your Oracle Middleware installation. It is identical to `BEA_HOME`.

`WL_HOME`: Is the root directory of your WebLogic installation

`PATH`: Adds the JDK and WebLogic directories to the system path

`CLASSPATH`: Adds the JDK and WebLogic JARs to the classpath

`JAVA_HOME`: Is the location of the version of Java used to start WebLogic Server

`JAVA_OPTIONS`: Are the Java command-line options for running the server (These will be tagged to the end of `JAVA_VM` and `MEM_ARGS`.)

`JAVA_VENDOR`: Is the vendor of the JVM (that is, BEA, HP, IBM, Sun, and so on)

`JAVA_VM`: Is the Java argument that specifies the VM to run (for example, `-server`, `-hotspot`, and so on)

`JAVA_USE_64BIT`: Indicates whether JVM uses 64-bit operations

`WEBLOGIC_CLASSPATH`: Is the classpath required to start WebLogic Server

List of Environment Variables and Their Meanings (continued)

`ANT_HOME`: Is the Ant Home directory

`ANT_CONTRIB`: Is the Ant contrib directory

`LD_LIBRARY_PATH`, `LIBPATH`, and `SHLIB_PATH`: Locate native libraries

`CLASSPATHSEP`: Is the `CLASSPATH` delimiter

`PATHSEP`: Is the `PATH` delimiter

`POINTBASE_HOME`: Is the PointBase home directory if a PointBase PID is detected

`POINTBASE_TOOL`: Is the PointBase tools JAR

`POINTBASE_CLASSPATH`: Is the classpath needed to start PointBase

`POINTBASE_CLIENT_CLASSPATH`: Is the PointBase client classpath

`PATCH_CLASSPATH`: Is the WebLogic system classpath patch

`PATCH_LIBPATH`: Is the library path used for patches

`PATCH_PATH`: Is the path used for patches

`WEBLOGIC_EXTENSION_DIRS`: Are the extension directories for the WebLogic classpath patch

`MEM_ARGS`: Is the variable to override the standard memory arguments passed to Java

`USER_MEM_ARGS`: Is the variable to override the standard memory arguments passed to Java

`PRODUCTION_MODE`: Is the variable that determines whether WebLogic Server is started in production mode

`DOMAIN_PRODUCTION_MODE`: Is the variable that determines whether the workshop-related settings such as the `debugger`, `testconsole`, or `iterativedev` should be enabled; settable only using the command-line parameter named production

`SERVER_NAME`: Is the name of the WebLogic server

`PRE_CLASSPATH`: Is the path style variable to be added to the beginning of the `CLASSPATH`

`POST_CLASSPATH`: Is the path style variable to be added to the end of the `CLASSPATH`

`PRE_PATH`: Is the path style variable to be added to the beginning of the `PATH`

`POST_PATH`: Is the path style variable to be added to the end of the `PATH`

`DERBY_HOME, DERBY_TOOLS, DERBY_CLASSPATH, DERBY_OPTS`: Are the variables for the included evaluation all-Java database

`HOME`: Is the user's home directory; different for each user

Other components such as Oracle HTTP Server may require and set their own environment variables—for example, `ORACLE_INSTANCE`. Note that *`domain_name`* is *not* an environment variable; you must replace it with the actual name of the domain.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can McCrider (cammccrider@deloitte.com) has a non-transferable license to use this Student Guide.

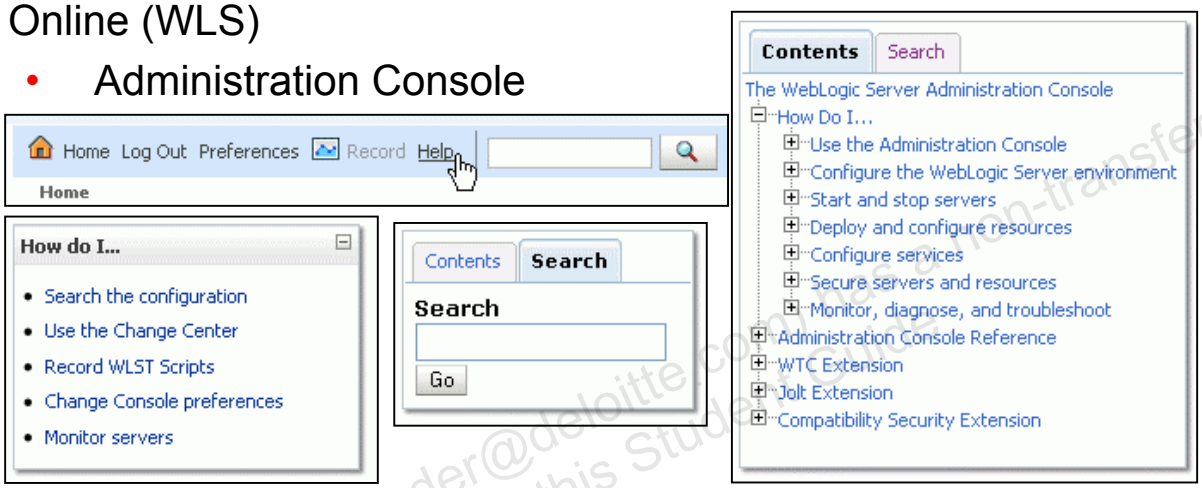
Documentation

Offline (Web)

- http://download.oracle.com/docs/cd/E12840_01/wls/docs103/sitemap.html

Online (WLS)

- Administration Console



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Documentation

Documentation is available on the Web in both HTML and PDF formats from Oracle and BEA Web sites for free. Both sites contain almost identical documentation. After the Administration Console is installed and running, you have access to the online Help from the administration server. As you navigate the menus of the Console, the "How do I" panel changes to reflect the topics that you are currently working on. On any screen in the Console, you can click Help or search for help on any topic.

Road Map

- Setting up an Oracle WebLogic Server environment using the GUI
- Setting up an Oracle WebLogic Server environment using the command line
- Specialized installations

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has non-transferable license to use this Student Guide.

Specialized Installations

- Net Installer from the Web
- Development-Only ZIP file installer
- Coherence installation
  - Typical
  - Custom
- Server Examples
  - SIP Server
  - Medrec

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl Mc Cruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Specialized Installations

Net Installer

Allows you to download only the parts you need, not the whole package. This saves on download time and space.

Development-Only Installer

Oracle provides a complete WebLogic Server installation in a ZIP file only for development use. This installation is supported on Windows, Linux, and Mac OS X systems. The extracted installation contains all the necessary artifacts you need to develop applications on WebLogic Server, but uses less disk space than a WebLogic Server installation performed in Typical mode.

Coherence Installation

You can install Oracle Coherence directly from the WebLogic Server installation program by selecting Coherence Product Files. Oracle Coherence is installed by default if you select a Typical installation. If you select a Custom installation, you also have the option to install Oracle Coherence code examples.

Server Examples

The WebLogic Server Medrec and Medrec-Spring server examples have been modified to use the evaluation Derby database that is included with WebLogic Server (see the next section). They have also been modified to use Oracle TopLink as the Java Persistence Architecture (JPA) persistence provider, where such a provider is used. The WebLogic Server installation includes additional

Specialized Installations

- Derby database
  - Evaluation license (just as PointBase was evaluation only)
  - Apache open source, entirely in Java
- 64-bit installation
  - Does not include the JDK
  - Special flag for 32/64 hybrid

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Specialized Installations (continued)

server examples for SIP server. The SIP server examples are installed if you select the Server Examples option during installation.

Evaluation Database

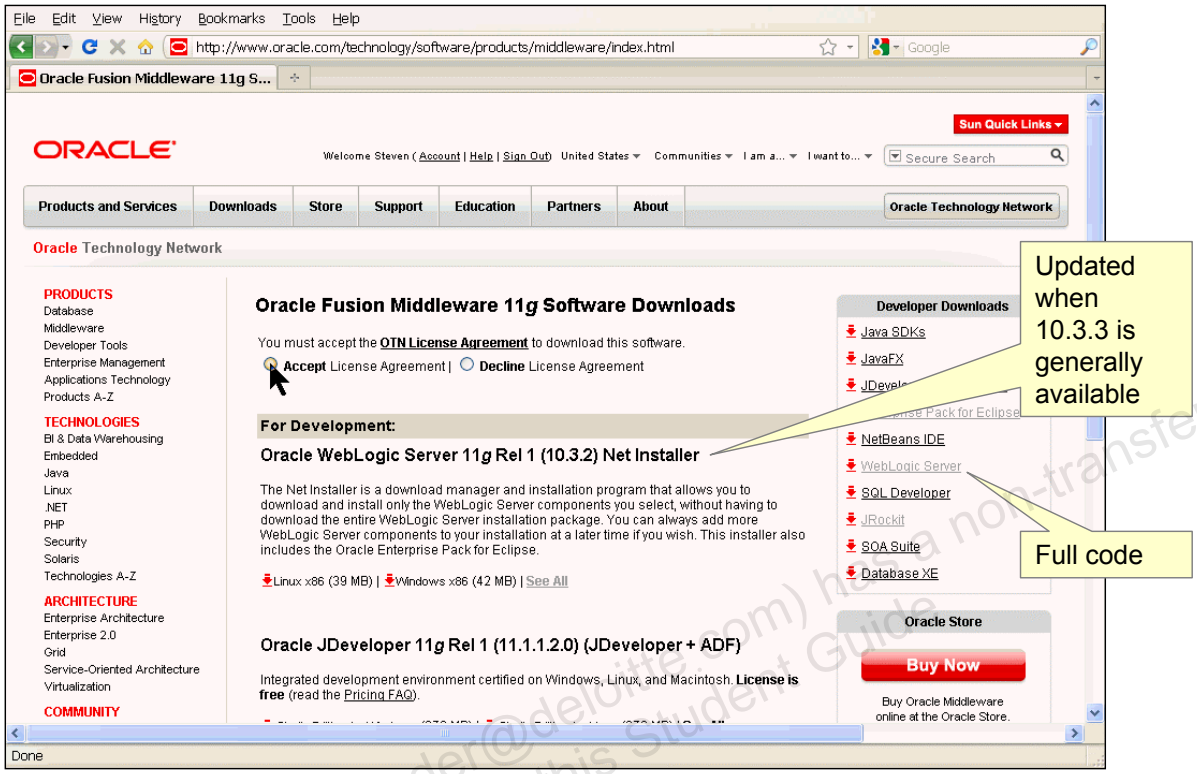
The WebLogic Server installation program includes an Evaluation Database option on the "Choose Products and Components" screen. If selected, an evaluation Derby database is installed with WebLogic Server in the `WL_HOME\common\derby` directory. If you select a Typical installation, this component is installed by default. If you choose to install the Server Examples component, the Evaluation Database option cannot be deselected, because some of the server examples use the evaluation database.

64-bit Installation

The 64-bit version of the software comes only as a JAR file with no bundled JDK, so you have to install the 64-bit JDK first and then run the command `java -jar ./wls1033_generic.jar` to start the installation of the WebLogic Server. Include the `-d64` flag in the installation command when using a 32/64-bit hybrid JDK (such as for the HP-PA, HPIA, and Solaris64 platforms). For example, if installing in graphical mode using the package installer:

```
java -d64 -jar ./wls1033_generic.jar
```

Downloading Software from OTN



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Downloading Software from OTN

Oracle Technology Network (available from http://www.oracle.com/technology) is the world's largest community of developers, DBAs/admins, and architects using Oracle products and industry-standard technologies. Millions of members collaborate via OTN every day to share real-world expertise and best practices about how to best design, build, deploy, manage, and optimize applications. Membership is free.

As an OTN member, you can:

- Get started with Java, PHP, Linux, and other industry-standard technologies
- Download and use software such as WebLogic Server (subject to license terms)
- Explore and download product documentation in PDF or HTML formats
- Read technical articles and notes authored by OTN members
- Join discussion forums to get advice from Oracle engineers and other OTN members
- Listen to podcast interviews with Oracle engineers, customers, and partners
- Bookmark technology and developer centers devoted to your area of interest
- Subscribe to developer email newsletters

Quiz

Net installer is preferred over the package installer if you want to install select components using only the Custom option and have access to the Internet.

a. True

b. False

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

Net installer has a smaller footprint and is preferred if you want to use the Custom option and install a selected number of products. Package installer is good for the default (all components selected) option. Net installer requires access to the Internet to incrementally download and install software bytes.

Quiz

You want to install WebLogic Server to use 64-bit JDK and JVM because you intend to use more than 4 GB heap sizes. Which WLS installation package would you use?

a. `./wls1033_linux64.bin`

b. `java -jar wls1033_generic.jar`

c. `./runInstaller`

d. `setup.exe`

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

There is no bundled JDK, so install that first, then the generic JAR file.

Quiz

Which JDK does Oracle WebLogic Server 10.3.3 come bundled with for a 32-bit Linux platform?

a. Sun SDK 1.6
b. JRockit SDK 1.6
c. Both

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: c

Both are shipped with 32-bit Oracle WebLogic Server 10.3.3. There are 64-bit JDK versions available for download from OTN.

Summary

In this lesson, you should have learned how to:

- Install Oracle WebLogic Server 10.3.3
- Describe the organization and contents of the WebLogic Server directory structure
- Navigate the WebLogic Server online and offline documentation

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 3 Overview: Installing Oracle WebLogic Server 11*g*

This practice covers the following topics:

- Installing Oracle WebLogic Server with JRockit
- Navigating through the installed folder structure

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 3 Overview: Installing Oracle WebLogic Server 11*g*

See Appendix A for the steps to complete this practice.

5

Configuring a Simple Domain

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Describe the common elements in a WebLogic Server domain
- Describe how domains are used in the enterprise
- Compare administration and managed servers
- Configure a domain
- Describe the organization and contents of the WLS directory structure
- Describe the use of WLST offline to manage domains
- Create a simple domain with one managed server
- Check the port numbers that are used for components

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

As the administrator of the middleware, you get to name the domains and servers. The application is a Medical Records system for a doctor's office, so you decide on a "MedRec" prefix for most names. The application is from a software company named Avitek, so you will see that name shows up on Web page banners. This system uses Web clients and a back-end database. Your first job is to create the total application environment: a "domain." The domain *references* the database but does not *include* the database. All domains require some common elements, so if the creation of a domain can also make those other pieces (servers of various sorts), then you could benefit from time-saving procedures. Besides, you can always come back later and either modify the servers created at this time or create other servers at a later date.

Objectives

After completing this lesson, you should be able to:

- Identify the location of process management scripts
- Describe the hierarchy of scripts and the setting of environment variables
- Use scripts to start and stop the administration server and the managed servers

ORACLE

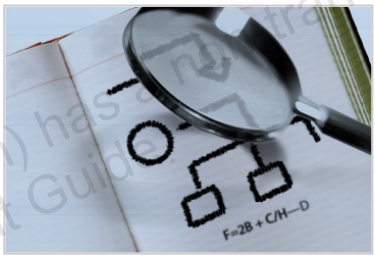
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

- Domains
  - Explaining how the domain works
  - Describing the domain directory structure
  - Configuring a domain
- Starting and stopping the Oracle WebLogic Server



ORACLE

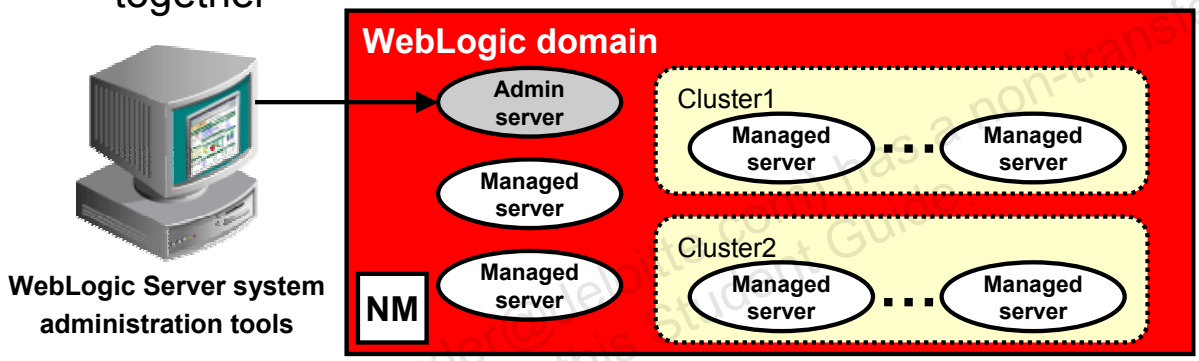
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Domain: Overview

- Is the basic administration unit for Oracle WebLogic Server
- Always includes one Oracle WebLogic Server instance configured as an administration server
- May include optional Oracle WebLogic Server instances in a domain called managed servers
- May also include clusters of server instances that work together



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@yahoo.com) has a non-transferable license to use this Student Guide.

Domain: Overview

A domain is the basic administration unit for Oracle WebLogic Server. It consists of one or more Oracle WebLogic Server instances and logically related resources and services that are managed collectively as one unit. As shown in the graphic, the basic domain infrastructure consists of one administration server and optional managed servers and clusters.

A domain always includes one Oracle WebLogic Server instance that is configured as an administration server. The administration server provides a central point for managing the domain and providing access to the Oracle WebLogic Server administration tools. These tools include, but are not limited to, the following:

- **Oracle WebLogic Server Administration Console:** Graphical user interface (GUI) to the administration server
- **Oracle WebLogic Server Node Manager (NM):** A Java program that enables you to remotely start and stop the administration and managed server instances. It also monitors and automatically restarts servers after an unexpected failure. The Node Manager is covered in more detail in the lesson titled "Configuring Node Managers."

Domain: Overview (continued)

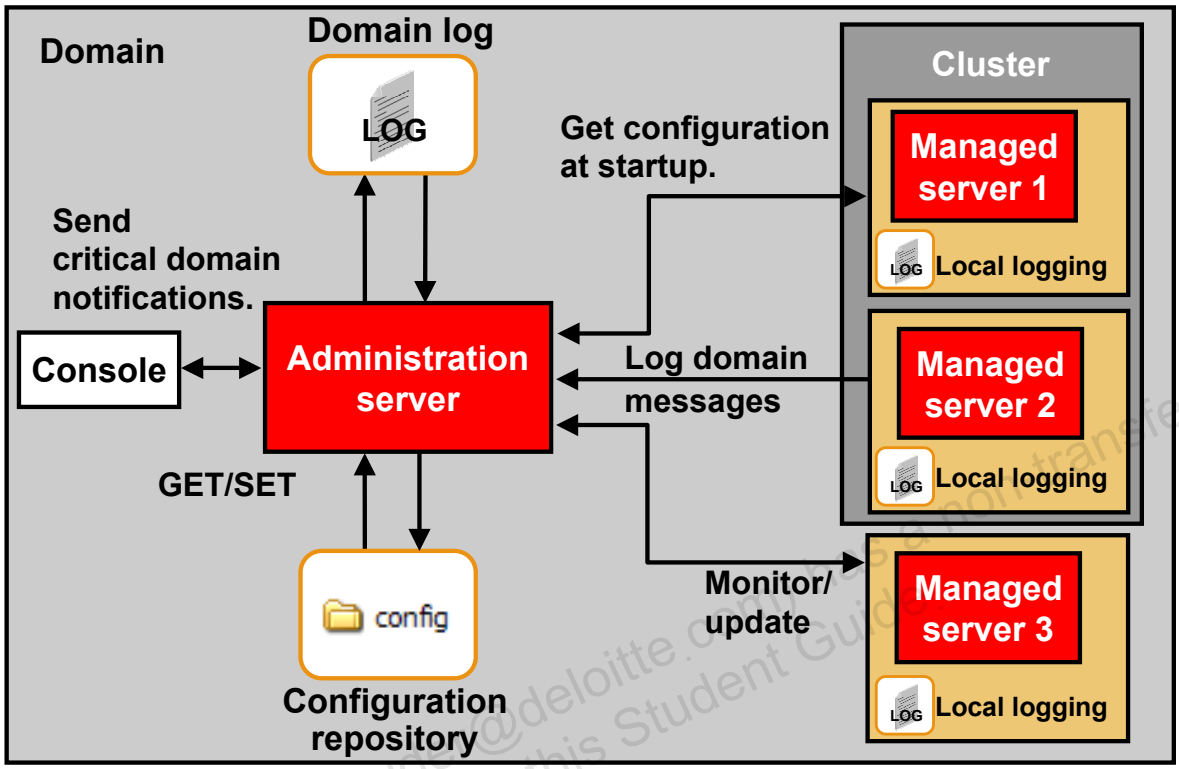
All other Oracle WebLogic Server instances in a domain are called managed servers. Managed servers host application components and resources, which are also deployed and managed as part of the domain. In a domain with only a single Oracle WebLogic Server instance, that server functions as both the administration server and the managed server.

A domain may also include Oracle WebLogic Server clusters, which are groups of server instances that work together to provide scalability and high availability for applications. Clusters can improve performance and provide failover if a server instance becomes unavailable. The servers within a cluster can run on the same machine, or they can reside on different machines. To the client, a cluster appears as a single Oracle WebLogic Server instance.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Domain Diagram



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) is a non-transferable license to use this Student Guide.

Domain Diagram

The diagram in the slide shows the critical components of a domain. A domain is an arbitrary logical administration unit that is managed by one administration server. A domain can encompass clusters in different geographies.

The administration server is responsible for providing configurations for all servers of a specific domain and for logging critical (configurable) notifications of the domain's servers. It is also responsible for monitoring managed servers. A managed server is responsible for performing the business logic.

A managed server gets its configuration from the administration server at boot time. The managed server is then able to execute independently of the administration server. The administration server registers itself with each managed server so that it can receive critical notifications and run-time server state changes. Note that the only configuration folders and files that play a role are those of the administration server. The configuration folders and files of the managed servers are normally ignored because they download all configuration information from the administration server.

Configuration management is implemented through Java Management Extension (JMX). The communication between the managed servers and the administration server is via Remote Method Invocation (RMI), which also uses JMX.

Note: The diagram in the slide is a simplification; all the servers would be monitored by and updated from the administration server. Also, all the managed servers retrieve their configuration information at startup time from the administration server and log critical data to the administration server.

Configuring a Domain

- After the installation, configure a domain on which to develop and deploy applications.
- By creating a domain, you define a collection of resources, such as:
  - Managed servers
  - Clusters
  - Database connections
  - Security services
  - Java EE applications
- Configuration Wizard creates and configures domains.
- Common domain configurations are:
  - Development or test
  - Production

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring a Domain

All managed servers in a domain must run the same version of Oracle WebLogic Server. The administration server can run either the same version as the managed servers in the domain or a later service pack.

In addition to infrastructure components, a domain defines the basic network configuration for the server instances that it contains. Specifically, a domain defines application deployments, supported application services (such as database and messaging services), security options, and physical host machines. Configuration information for a domain is maintained in an XML file named `config.xml` that is located in the domain's root directory.

In production environments that require increased performance, throughput, or availability for an application, several managed servers might be grouped in a cluster. In such a case, the domain consists of one or more clusters with the applications they host and an administration server to perform management operations.

Note: In production environments, it is recommended that you deploy applications only on managed servers; the administration server should be reserved for management tasks.

Configuring a Domain (continued)

In development or test environments, a single application and server might be deployed independently without managed servers. In such a case, you can have a domain consisting of a single administration server that also hosts the applications you want to test or develop.

Configuration Checklist

To prepare for the configuration, you may want to collect a list of names and numbers so that you do not get a resource conflict. You need to know (if applicable):

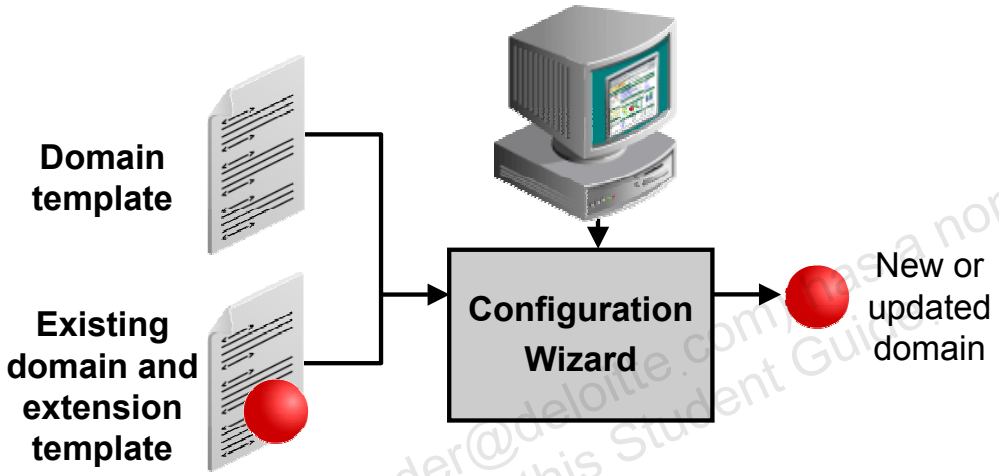
- Resource names
- IP addresses
- DNS names
- Non-SSL ports
- SSL ports
- User IDs
- Passwords
- Destinations on disk for storage

You possibly also need to know other information. Names need to be unique within their scope (either application scope or domain scope) and the combination of IP address and port needs to be unique. Although you are allowed to have two very different kinds of entities with the same name—for example, an application and a cluster both named Benefits—this is not considered the best practice. You are not allowed to have two similar entities with the same name—for example, a server and a cluster both named HR is an error. This checklist then serves as your documentation record as well.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting the Domain Configuration Wizard

- Scripts are in the `<WL_HOME>/common/bin` directory.
- Two modes:
  - Graphical: `config.sh`
  - Console: `config.sh -mode=console`



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

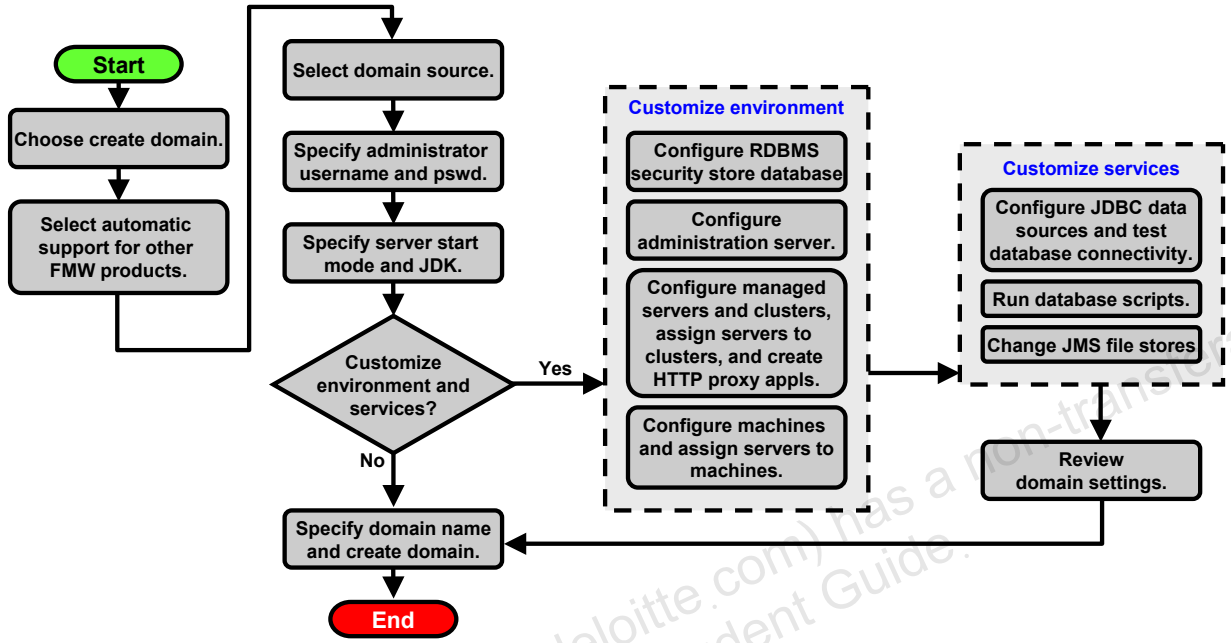
Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting the Domain Configuration Wizard

Before you can develop and run a WebLogic application, you must first create a domain. The Configuration Wizard simplifies the process of creating and extending a domain. To create or extend a domain by using the Configuration Wizard, you select the product components to be included in the domain (or choose a template that best meets your requirements), and provide basic configuration information. The Configuration Wizard then creates or extends the domain by using the settings from the templates.

Note: The Domain Template Builder simplifies the process of creating templates by guiding you through the process of creating custom domain and extension templates. You can use these templates for creating and extending domains with the Configuration Wizard or the WebLogic Scripting Tool (WLST) or the `unpack` command.

Creating a Domain Using the Domain Configuration Wizard



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

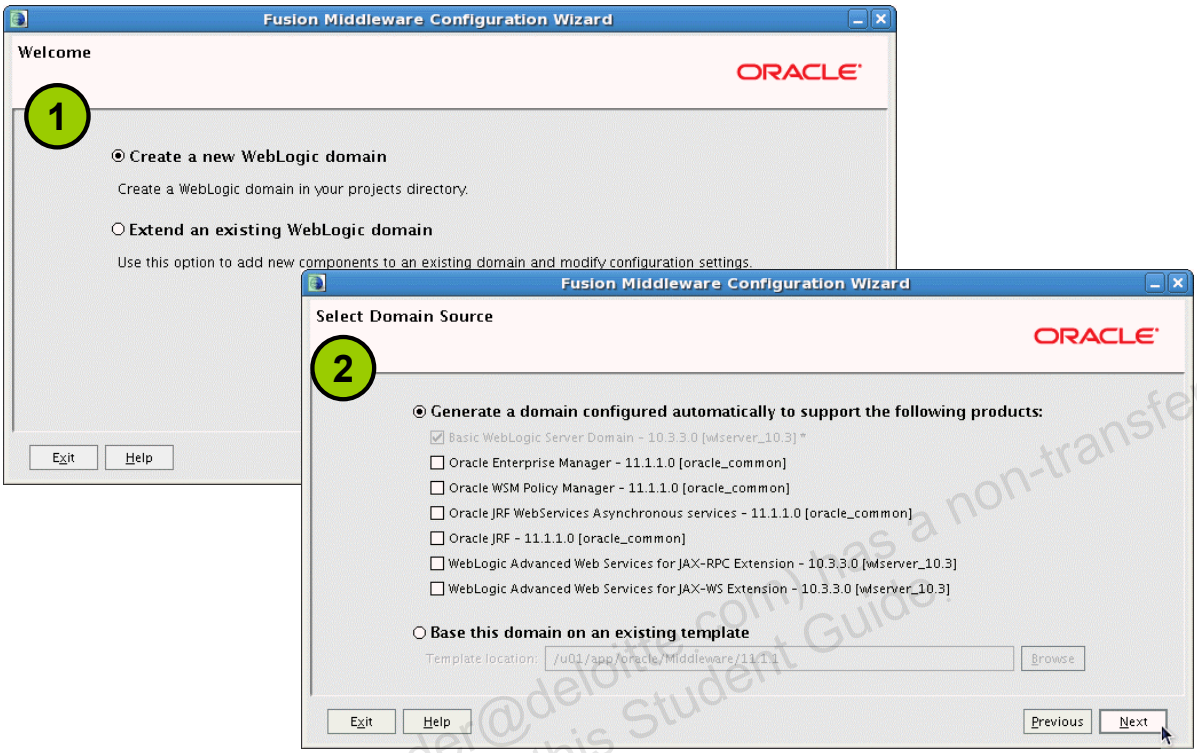
Creating a Domain Using the Domain Configuration Wizard

The Configuration Wizard guides you through the process of creating a domain for your target environment by selecting the product components that you want to include in your domain, or by using domain templates. If required, you can also customize the domain to suit your environment by adding and configuring managed servers, clusters, and machine definitions, or customizing predefined Java Database Connectivity (JDBC) data sources and JMS file store directories.

You might want to customize your domain in the following circumstances:

- To create a multiserver or clustered domain when using the default settings. All the predefined templates delivered with the product create single-server domains.
- To use a database that is different from the default database in the domain or extension template. Here, you need to customize the JDBC settings to point to the appropriate database.
- To customize the listen port and the SSL port
- To create a test environment using a domain template that you received, and to modify the domain configuration to work in the test environment based on your requirement

Creating a New WebLogic Domain and Selecting the Domain Source



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Cam McCruder (camcruder@deloitte.com) has a non-transferable license to use this Student Guide.

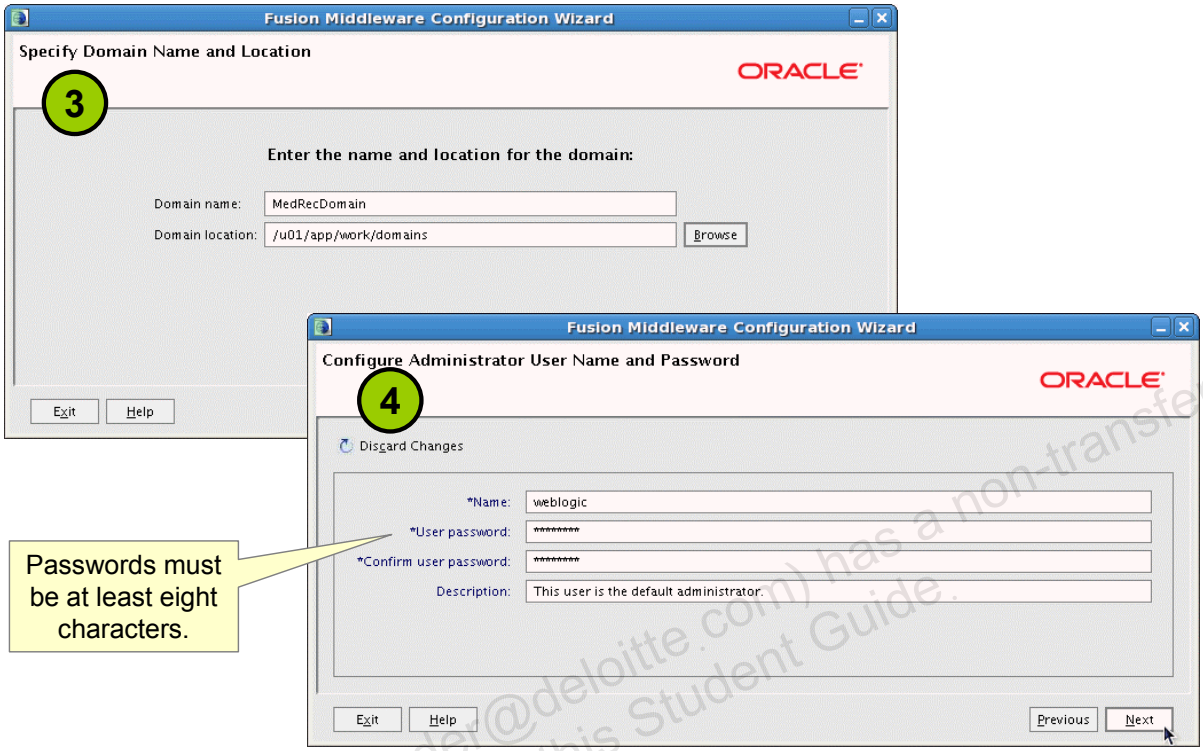
Creating a New WebLogic Domain and Selecting the Domain Source

The screenshots show the start of the Configuration Wizard.

1. If you want to create a new domain, select the "Create a new WebLogic domain" option. If you want to extend or customize a preexisting domain for the JDBC data source, run SQL scripts, or change the JMS store settings, and select the "Extend an existing WebLogic domain" option.
2. If you choose to have the domain configured automatically to support the Oracle WebLogic Server, the domain is based on the default template `wls.jar` that is found in the `<WL_HOME>/common/templates/domains` directory.

   If you have an existing domain that you want to use as a source to create the new domain, select the "Base this domain on an existing template" option (which by default is pointed to the same preceding `wls.jar` location).

Configuring Administrator Settings



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

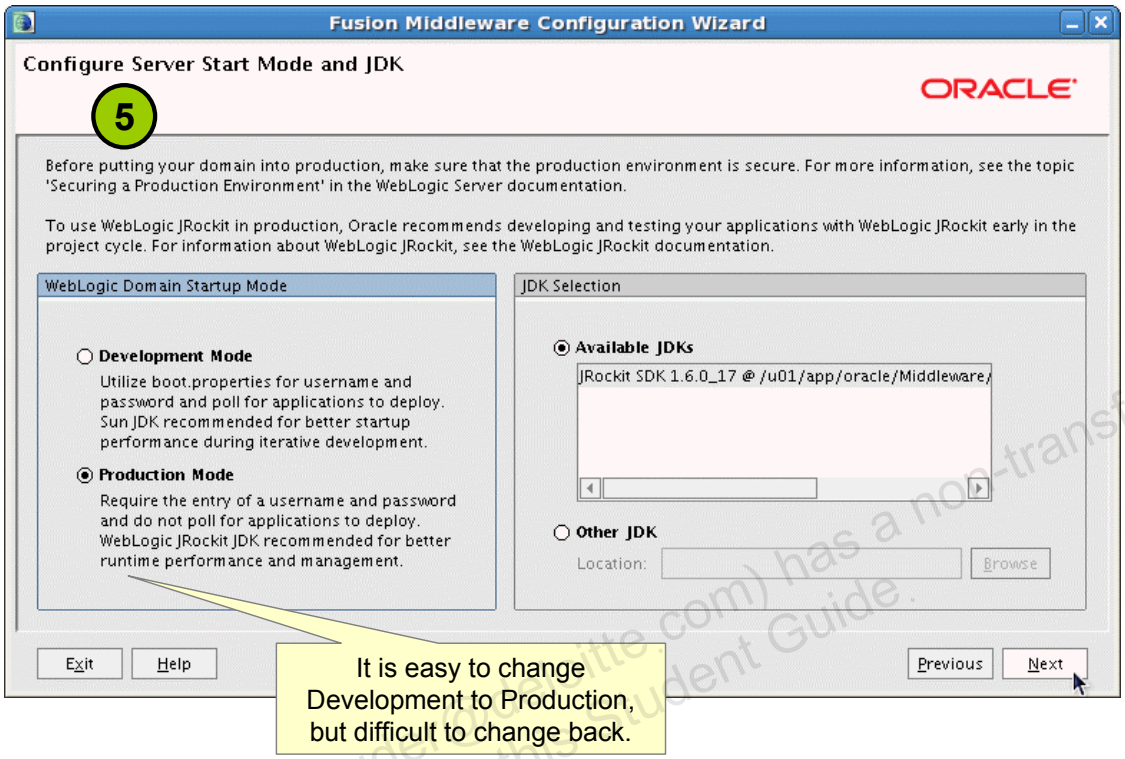
Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring Administrator Settings, Start Mode, and JDK

The screenshots show the username, password, mode, and JDKs of the Configuration Wizard.

3. The Domain name creates many directories under the specified location. Some of the directories do not get created until after the servers start the first time.
4. The name and password is stored in an internal Lightweight Directory Access Protocol (LDAP) directory. You can later move to an external LDAP-based identity store if you want. Using the administrator username and password, you can log in to the Administration Console to manage the Oracle WebLogic Server domain.

Configuring Startup Mode and JDK



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

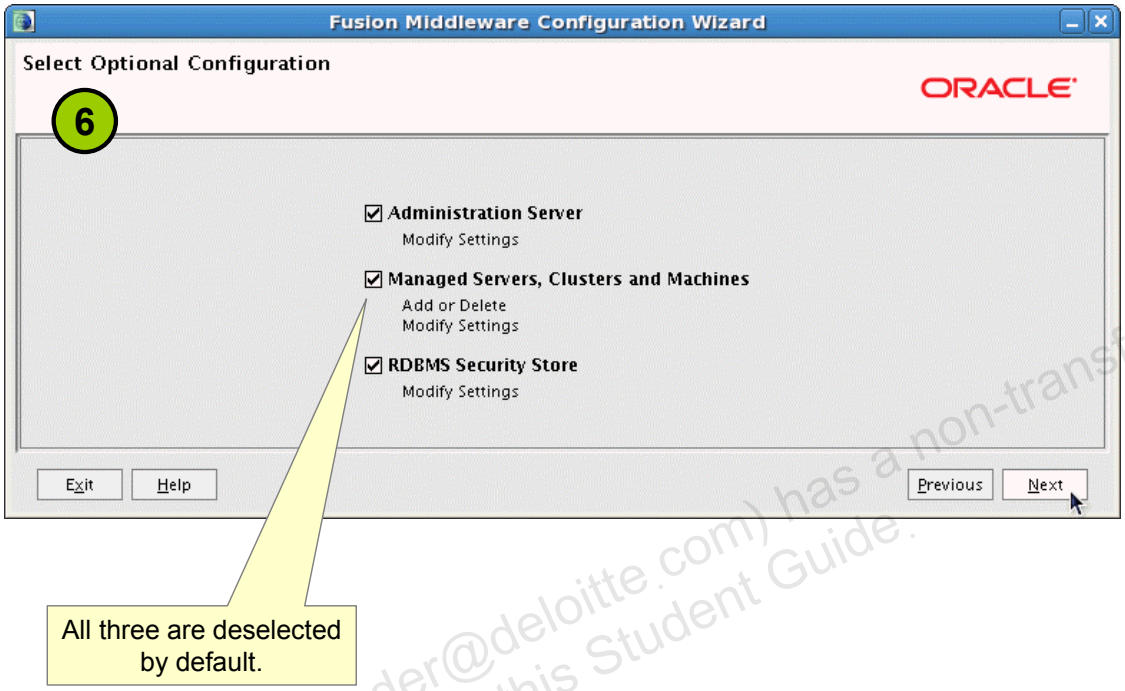
Carl McCruder (carl.mccruder@oracle.com) has a non-transferable license to use this Student Guide.

Configuring Administrator Settings, Start Mode, and JDK (continued)

The screenshots show the username, password, mode, and JDKs of the Configuration Wizard.

5. Select the Java Development Kit (JDK) from the options available. (If you do not see the JDK that you would like to use, make sure that the `PATH` and `<JAVA_HOME>` environment variables are pointing to the correct JDK and `JDK/bin` directory. As mentioned in the lesson titled "Installing Oracle WebLogic Server 11*g*," Oracle WebLogic Server 10.3 is certified with JDK 6.0.)
   The Domain Startup Mode is used to make a domain more or less suitable for development or production. Developers would choose Development, and administrators would choose Production. A Development Mode domain can be easily turned into Production Mode by a selecting a check box in the Administration Console that you will see later, but to turn the domain back into Development requires editing the `config.xml` file and changing `<production-mode-enabled>` to false for the domain. In Development mode, you can use auto-deployment and Fast-Swap, both covered in the lesson titled "Deployment Concepts" in this course.

Customizing Optional Configuration



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

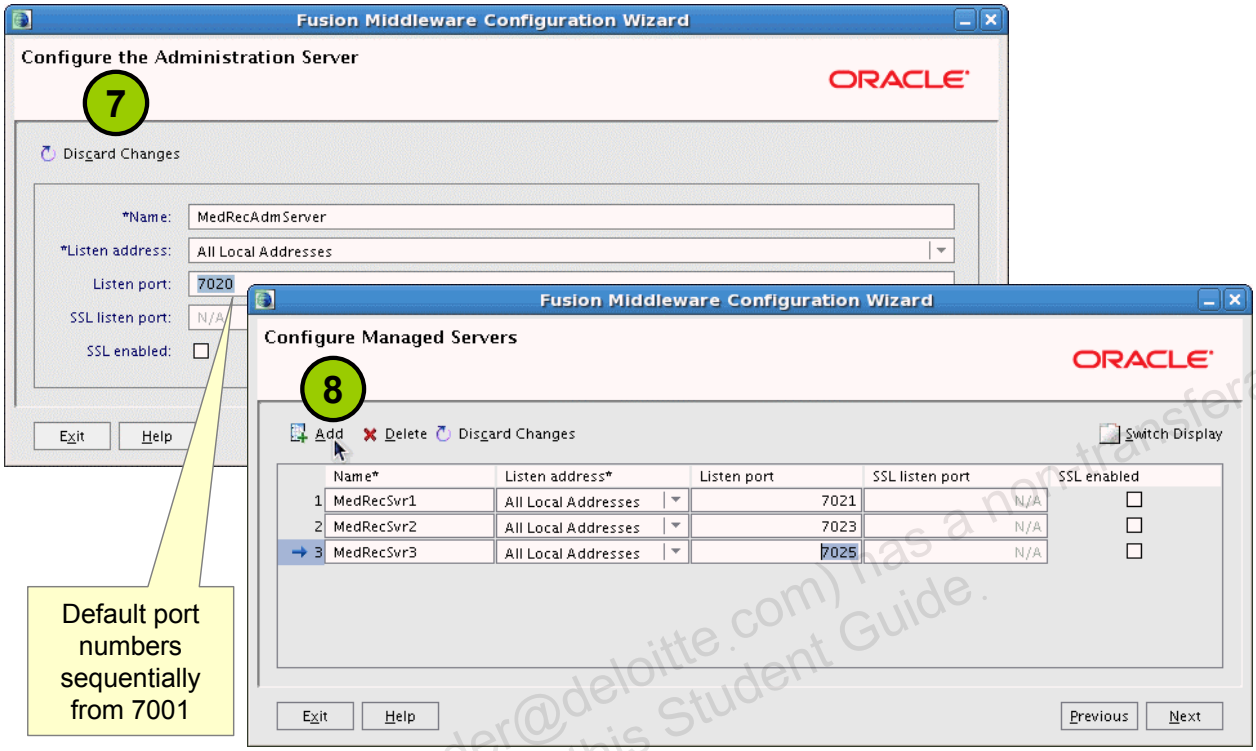
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Customizing Advanced Configuration

6. The screenshot asks if you want to optionally go down this configuration path. Select the check boxes to customize the environment or services settings, such as modifying (configuring) the administration server, managed servers, clusters, machines, JDBC data sources, RDBMS Security Stores, or JMS store settings. By default, none of the check boxes are selected. If you leave the check boxes deselected, it would skip the next few screens.

Configuring the Administration and Managed Servers



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a nontransferable license to use this Student Guide.

Configuring the Administration and Managed Servers

The screenshots show creating servers. In every domain, one server must be designated as the administration server: the central point from which the entire domain is managed.

You can access the administrator server by using the URL protocol://listen-address:listen-port. The protocol can be any of the following: t3, t3s, http, or https. (t3 and t3s are proprietary protocols that are analogous to the industry-standard HTTP or HTTPS protocols.)

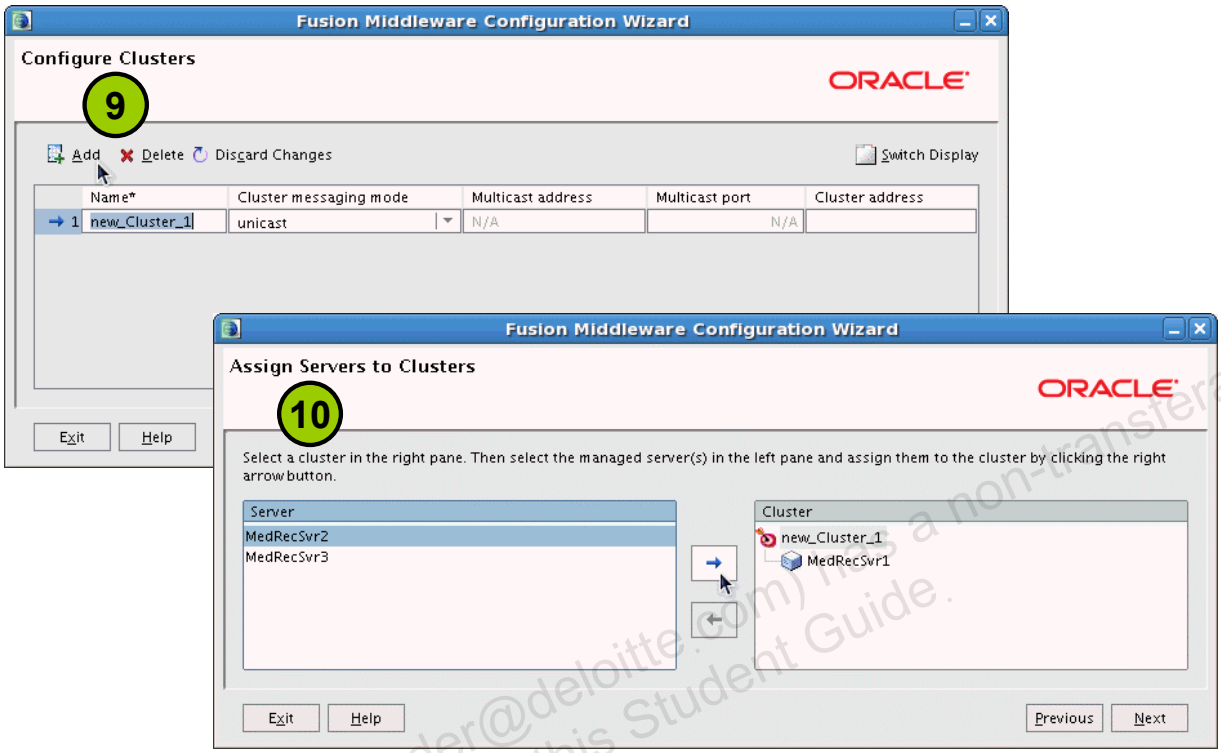
You can define the listen-address and listen-port on the “Configure the Administration Server” page of the Configuration Wizard. The valid port range for non-SSL ports is 1 through 65534. The default non-SSL port is 7001. The valid port range for SSL ports is 1 through 65535. The default SSL port is 7002.

On multihomed Windows machines, a server instance can bind to all available IP addresses. On a multihomed computer, you can use the same listen port, but you must configure each server to use a unique IP address as the listen address. If your computer does not support multiple IP addresses, you must use a different listen port for each active instance.

In production environments, enterprise applications are hosted typically on one or more managed servers; typically, there are no applications deployed to the administration server.

You can add and delete managed servers on the Configure Managed Servers page, which is displayed when you click Next on the “Configure the Administration Server” page of the Configuration Wizard.

Configuring Clusters and Assigning Servers to Clusters



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring Clusters and Assigning Servers to Clusters

The screenshots show optional cluster parameters. On the Configure Clusters page, enter a valid name for the cluster: a string of characters that can include spaces. The name of the cluster must be unique among all the component names within the domain. The default value in this field is `new_Cluster_n`, where *n* is a numeric value that is used to differentiate among all the default cluster names; the value of *n* for the first cluster is 1. The value is incremented by 1 for each cluster that you add.

Cluster Messaging Mode: This could be “multicast” or “unicast.” If you select “unicast,” the next two options are disabled. When you would pick unicast over multicast or multicast over unicast is discussed briefly on the next page and also in more detail in the lesson titled “Introduction to Clustering.” The default mode of cluster messaging is unicast.

Multicast address: Enter the multicast address for the cluster. This address is used by cluster members to communicate with each other. The default value is 239.192.0.0. The valid multicast address range is 224.0.0.1 through 239.255.255.255. Avoid using any address ending in .0.0.1.

Multicast port: Enter the multicast port for the cluster. The multicast port is used by cluster members to communicate with each other. The default value is 7001. Valid values for multicast ports range from 1 through 65534.

Configuring Clusters and Assigning Servers to Clusters (continued)

Cluster address: Enter the addresses to identify the managed servers in the cluster. A cluster address can be one of the following:

- A comma-separated list of IP addresses or DNS names and ports (for example, dns_name:port, dns_name:port)
- A DNS name that maps to multiple IP addresses
- A localhost, DNS name, or IP address if the listen address of all the managed servers is listening to the same address with unique port numbers

The cluster address is used in entity and stateless Enterprise JavaBeans (EJBs) to construct the host name portion of the URLs. If the cluster address is not set, EJB handles may not work properly.

Assigning Servers to Clusters

A managed server may belong to zero or one cluster. A cluster may have zero, one, or more servers. There is no point in having a cluster with zero or one server.

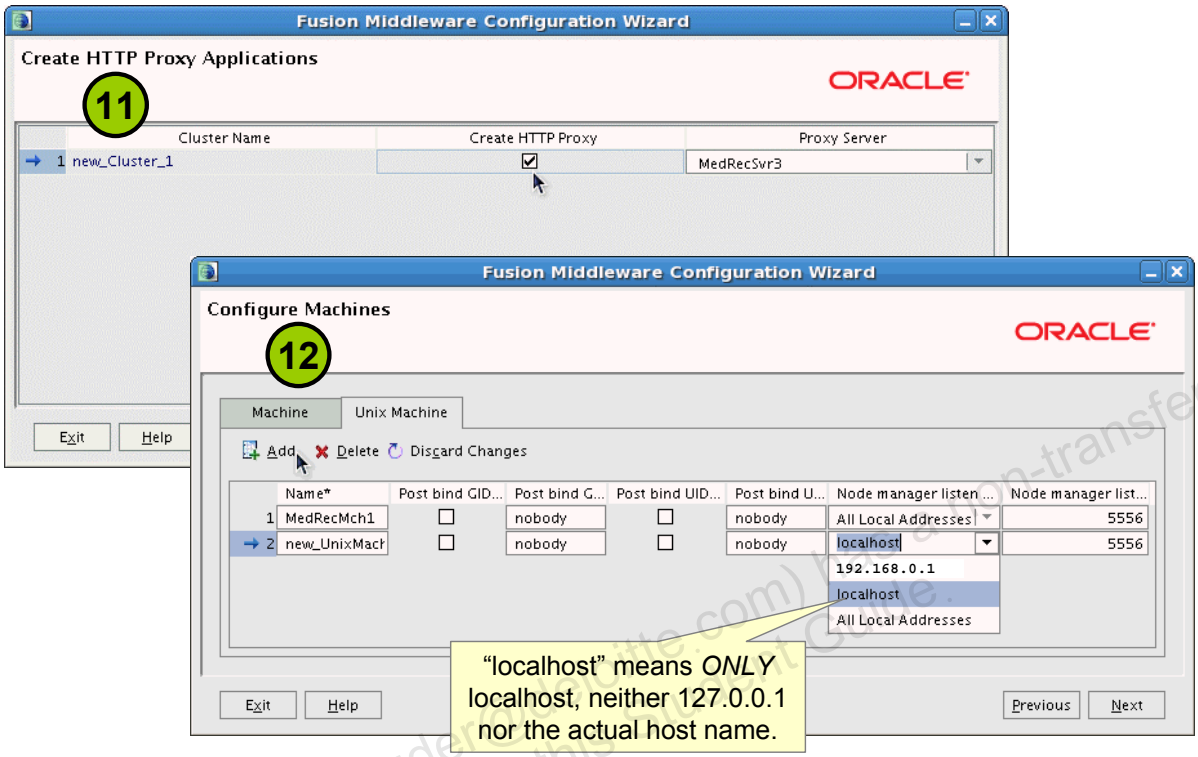
When to Use Multicast and When to Use Unicast

If your machines are not on the same IP subnet, so that communications from one machine to another need to go across several routers, possibly even over the Internet, multicast is not an option. Unicast is more reliable across subnets. But if the machines are all on the same IP subnet, multicast is preferable.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating an HTTP Proxy Application and Configuring Machines



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmccrudder@yahoo.com) has a non-transferable license to use this Student Guide.

Creating an HTTP Proxy Application and Configuring Machines

The screenshots show creating proxies and machines. An HTTP proxy application acts as an intermediary for HTTP requests.

On the Create HTTP Proxy Applications page of the Configuration Wizard, you can create an HTTP proxy application for each cluster and specify the managed server on which the proxy application must be deployed.

This page is displayed when you click Next on the "Assign Servers to Clusters" page only if both of the following statements are true:

- At least one managed server is assigned to a cluster.
- At least one managed server is not assigned to any cluster.

Creating HTTP Proxy Applications

1. If multiple clusters are defined, click the tab corresponding to the cluster for which you want to create the HTTP proxy applications.
2. Select the "Create HTTP proxy for cluster <*cluster_name*>" check box.
   A list of the managed servers that are not assigned to any cluster is displayed in the Proxy Server list.

3. From the **Proxy Server** list, select a managed server on which the proxy applications must be deployed.
   A proxy application named `OracleProxy4_`*`clustername_servername`* is created and targeted at the managed server.
4. Repeat steps 1 through 3 for each cluster for which you want to create the HTTP proxy applications.
5. Click **Next** to proceed.

Note: Although WebLogic Server has the built-in function of Web and HTTP server, you may want to use the Oracle HTTP Server covered in the lesson titled "Deployment Concepts."

Creating Machines

In a domain, the machine definitions identify the physical units of hardware and are used to associate computers with the managed servers that they host.

You might want to create machine definitions in situations such as (but not limited to) the following:

- The administration server uses the machine definition with the Node Manager application to start remote servers.
- Oracle WebLogic Server uses configured machine names when determining the server in a cluster that is best able to handle certain tasks, such as HTTP session replication. Oracle WebLogic Server then delegates those tasks to the identified server.

Note: You must configure machines for each product installation that runs a Node Manager process. The machine configuration must include values for the listen address and port number parameters.

The machine name is used to identify the machine within the Oracle WebLogic Server domain; it is not required to match the network name for the machine.

Node manager listen address: Select a value from the list for the listen address used by the Node Manager to listen for connection requests. By default, the IP addresses defined for the local system and localhost are shown in the list. The default value is localhost.

Node manager listen port: Enter a valid value for the listen port used by the Node Manager to listen for connection requests. The valid Node Manager listen port range is from 1 through 65534. The default value is 5556.

Post bind GID enabled: This check box is displayed only on the **Unix Machine** tab. Select this check box to enable a server running on this machine to bind to a UNIX group ID (GID) after it finishes all the privileged startup actions. By default, this check box is not selected.

Post bind GID: This field is displayed only on the **Unix Machine** tab. Enter a valid UNIX group ID (GID) that a server running on this machine will use after it finishes all the privileged startup actions. Otherwise, the server continues to run under the group from which it was started (requires you to enable a post bind GID).

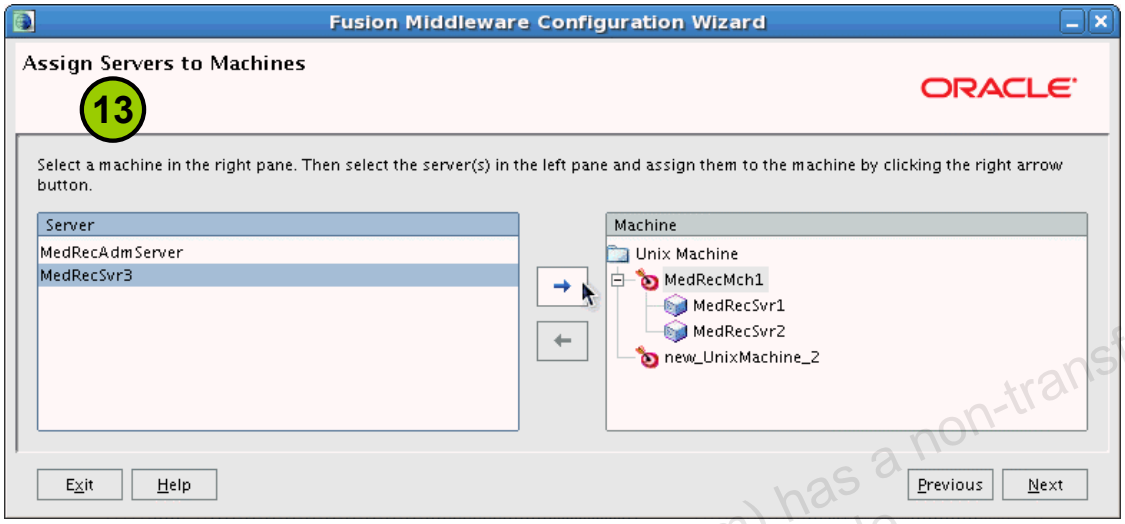
Post bind UID enabled: This field is displayed only on the **Unix Machine** tab. Select this check box to enable a server running on this machine to bind to a UNIX user ID (UID) after it finishes all the privileged startup actions. By default, this check box is not selected.

Post bind UID: This field is displayed only on the **Unix Machine** tab. Enter a valid UNIX user ID (UID) that a server running on this machine will run under after it finishes all the privileged startup actions. Otherwise, the server continues to run under the account from which it was started (requires you to enable a post bind UID).

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@lakecitte.com) has a non-transferable license to use this Student Guide.

Assigning Servers to Machines



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

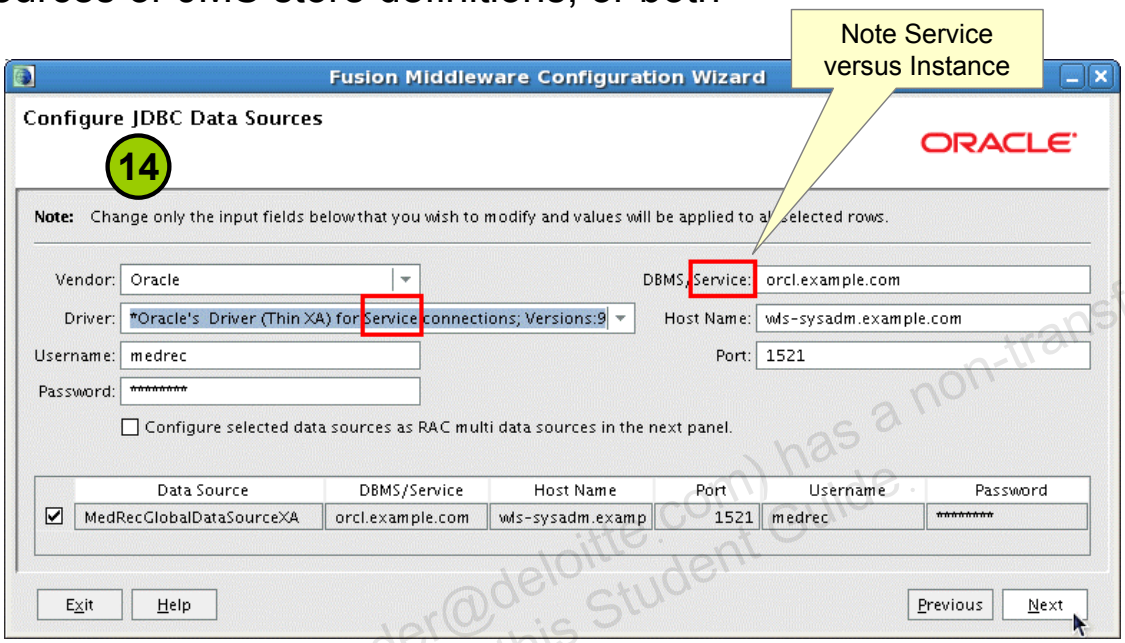
Assigning Servers to Machines

The screenshot shows assigning servers to machines. Machines are optional and assigning servers is optional. It is possible to create a machine and then not assign any servers to it, but there is no point in that. Sometimes the Administration Server is not assigned to a machine.

No databases are defined in the `wls.jar` template. Therefore, the Configure JDBC Data Sources page would not normally be displayed. For purposes of completeness, Configure JDBC Data Sources is shown next.

Configuring JDBC Data Sources

Shown only if you choose a template that contains JDBC data sources or JMS store definitions, or both



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring JDBC Data Sources

The screenshot shows JDBC configuration. In this example, you create a new domain based on the default template (`wls.jar`). This default template does not contain any JDBC data source or JMS store definitions. So, if you select the Oracle WebLogic Server template as the basis for the domain, this Configure JDBC Data Sources page is not displayed.

To see the screens pertaining to these options, you can run the Domain Configuration Wizard and select the "Extend my domain using existing extension template" option. If the template contains the existing JDBC data sources or the JMS store (for example, use templates under the `<WL_HOME>/common/templates/applications` directory), it will allow you to configure the JDBC data source, test the data source connections, and configure the JMS file store.

After step 13, if the domain source on which you base your domain contains the JDBC data source and JMS file store definitions, you are presented with the option to modify them; otherwise, you are presented with the option to review the domain settings and create the domain.

When you create or extend a domain using the Configuration Wizard, you can change the JDBC data source and JMS file store settings if they are defined in the domain or template that you selected as the source for the domain that you are creating.

Configuring JDBC Data Sources (continued)

Note: If you select **Yes** on the "Customize Environment and Services Settings" page, the Configuration Wizard does not run any database automatically, even though the domain JDBC is configured for whichever database is specified in the data source. In the Run Database Scripts dialog box, click the **Run Scripts** button to load the respective database.

Note: If you select **No** on the "Customize Environment and Services Settings" page, the Configuration Wizard automatically populates the required database, whichever database is specified in the data source.

A JDBC data source contains a pool of database connections that are created when the data source instance is created—when the data source is deployed or targeted, or at server startup. Applications look up a data source on the Java Naming and Directory Interface (JNDI) tree, and then request a connection. When the applications no longer need the connection, they return the connection to the connection pool in the data source.

The following are the fields on the Configure JDBC Data Sources page:

- **Name:** Enter a valid name for the JDBC data source. The name of the JDBC data source must be unique among all the component names within the domain.
- **JNDI name:** From the list, select the JNDI name to which this data source is bound.
  - To add a new JNDI name, select **Add New** and enter a valid JNDI path.
  - To change an existing JNDI name, select the name and edit it.

  You can associate multiple JNDI names with a single data source. When an application looks up a JNDI path, a `javax.sql.DataSource` instance corresponding to the data source is returned.
- **Database type:** From the list, select the type of database to which you want to connect. If your DBMS is not listed, select **Other**.
- **Driver:** From the list, select the JDBC driver that you want to use to connect to the database. The list includes common JDBC drivers for the selected DBMS. If you selected **Other** in the **Database type** field, this field is not available.
- **Class name:** If you selected a DBMS in the **Database type** field, no action is required. If you selected **Other** in the **Database type** field, enter the full package name of the class that implements the `java.sql.Driver` interface for your DBMS.
- **DBMS name:** Enter the name of the database. If you selected **Other** in the **Database type** field, this field is not available.
- **DBMS host:** Enter the name of the server machine that hosts the database. If you selected **Other** in the **Database type** field, this field is not available.
- **DBMS port:** Enter the port to be used to connect to the server. The default setting associated with the database selected is displayed. If you selected **Other** in the **Database type** field, this field is not available.
- **JDBC URL:** If you selected a DBMS in the **Database type** field, no action is required. If the **Driver name** is set and a default URL exists, that URL is displayed in this field. If you selected **Other** in the **Database type** field, enter the URL that should be used to create the connections in the connection pool in the data source.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCrindle (carl.mccrindle@deloitte.com) has a non-transferable license to use this Student Guide.

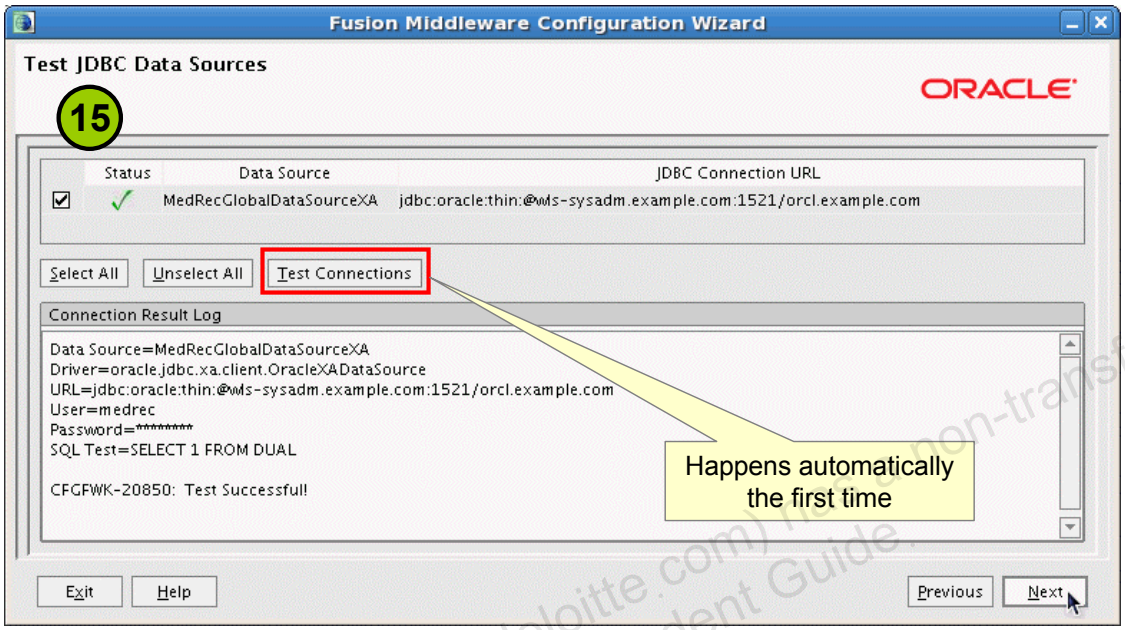
Configuring JDBC Data Sources (continued)

- **User name:** Enter the account login name that is required for connecting to the database. If you selected **Other** in the **Database type** field, this field is not available. You can specify this value in the **Additional Properties** field.
- **User password:** Enter a valid password for accessing the database. Valid values consist of a string of alphanumeric characters. The hyphen (-) and underscore (_) characters are supported. This password overrides the password entered as part of the JDBC properties. The value is encrypted.
- **Known properties:** If you selected a DBMS in the **Database type** field, no action is required. This field displays the properties list that is passed to the JDBC driver for creating physical database connections. If you selected **Other** in the **Database type** field, this field is blank and not available.
- **Additional properties:** Enter any additional properties to be passed to the JDBC driver. If you specified **Other** in the **Database type** field, enter any properties to be passed to the JDBC driver, such as the property needed to specify the user.
- **Supports global transactions:** If you selected an XA driver in the **Driver** field, **Supports global transactions** and the **Two-phase commit** protocols are selected automatically. You cannot change the protocol. If you selected a non-XA driver in the **Driver** field, you can, if required, select **Supports global transactions**. Then you can select one of the following protocols:
  - **Logging last resource:** With this option, the transaction branch in which the connection is used is processed as the last resource in the transaction and is processed as a one-phase commit operation. The result of the operation is written in a log file on the resource itself, and the result determines the success or failure of the prepare phase of the transaction. This option offers some performance benefits with greater data safety than Emulate Two-Phase Commit, but it has some limitations.
  - **Emulate two-phase commit:** With this option, the transaction branch in which the connection is used always returns success for the prepare phase of the transaction. It offers performance benefits, but also exposes data to risks in some failure conditions. Select this option only if your application can tolerate heuristic conditions.
  - **One-phase commit (default):** With this option, a connection from the data source can be the only participant in the global transaction and the transaction is completed by using a one-phase commit optimization. If multiple resources participate in the transaction, an exception is thrown.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carl.mcgruder@dbtuts.com) has a non-transferable license to use this Student Guide.

Testing Data Source Connections



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Testing Data Source Connections

The screenshot shows testing connections. To test the data source connections to the specified databases, click **Test Connections**. The Test Data Source Connections page is displayed. The Test Data Source Connections page allows you to test the connection to the database for each of the data sources defined in your domain, by using the JDBC URL defined for the database.

A list of the data sources and the associated JDBC URLs is displayed.

1. Make sure that the database and listener to which you want to test the connections is running.
2. Make sure that the user (in this case medrec) exists and is unlocked. Test the user with sqlplus.
3. Select the connection(s) that you want to test.
4. Click **Test Connections**.

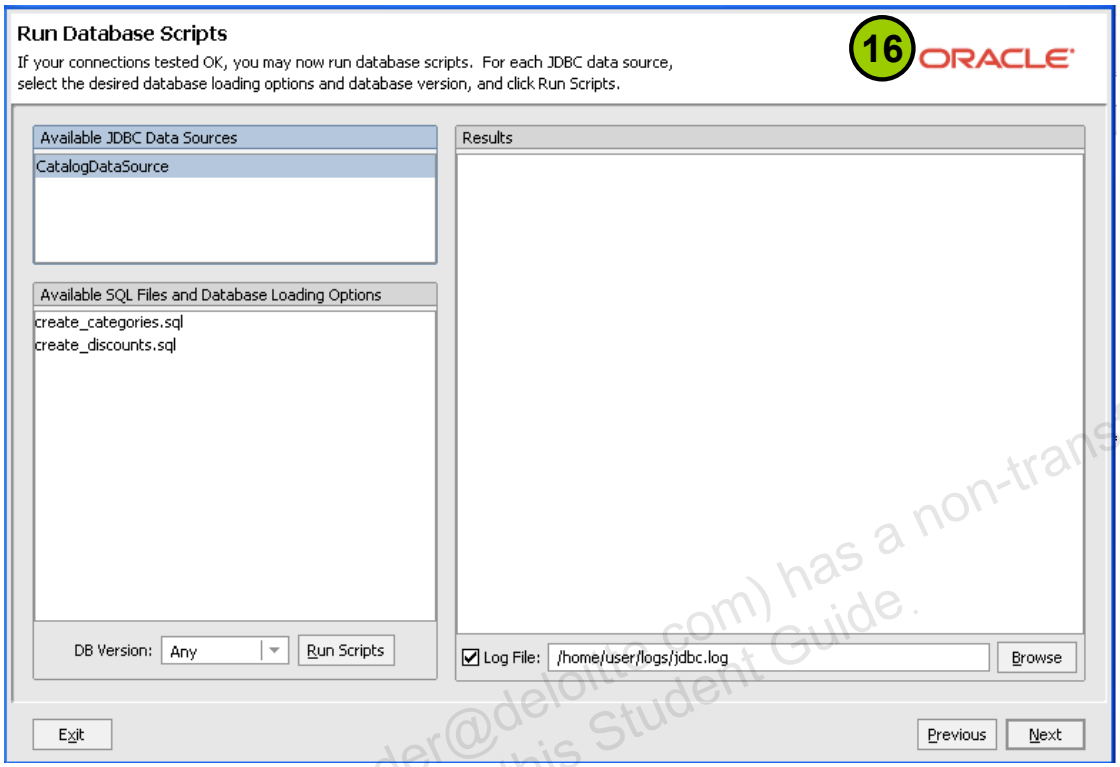
The button changes to **Cancel**. You can click **Cancel** at any time to cancel the test. After a test is in progress, all the other buttons are disabled.

The **Status** field is blank for connections for which the test has not been initiated.

1. Review the results of the test in the **Connection Result Log** pane.
2. Perform the test for each data source, as required.
3. Click **OK** to return to the Configure JDBC Data Sources page.

A domain template may contain a set of SQL files organized by database type. If the domain template contains SQL files, you can run them while creating the domain on the Run Database Scripts page. The database content for each of the data sources defined in your domain is set up by using preexisting SQL or database-loading files.

Running Database Scripts



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Running Database Scripts

Note: No databases are defined in the Oracle WebLogic Server template (`wls.jar`). If you select the Oracle WebLogic Server template as the basis for the domain, the Configure JDBC Data Sources page and the Run Database Scripts page are not displayed. Selecting a Portal template would have displayed this page.

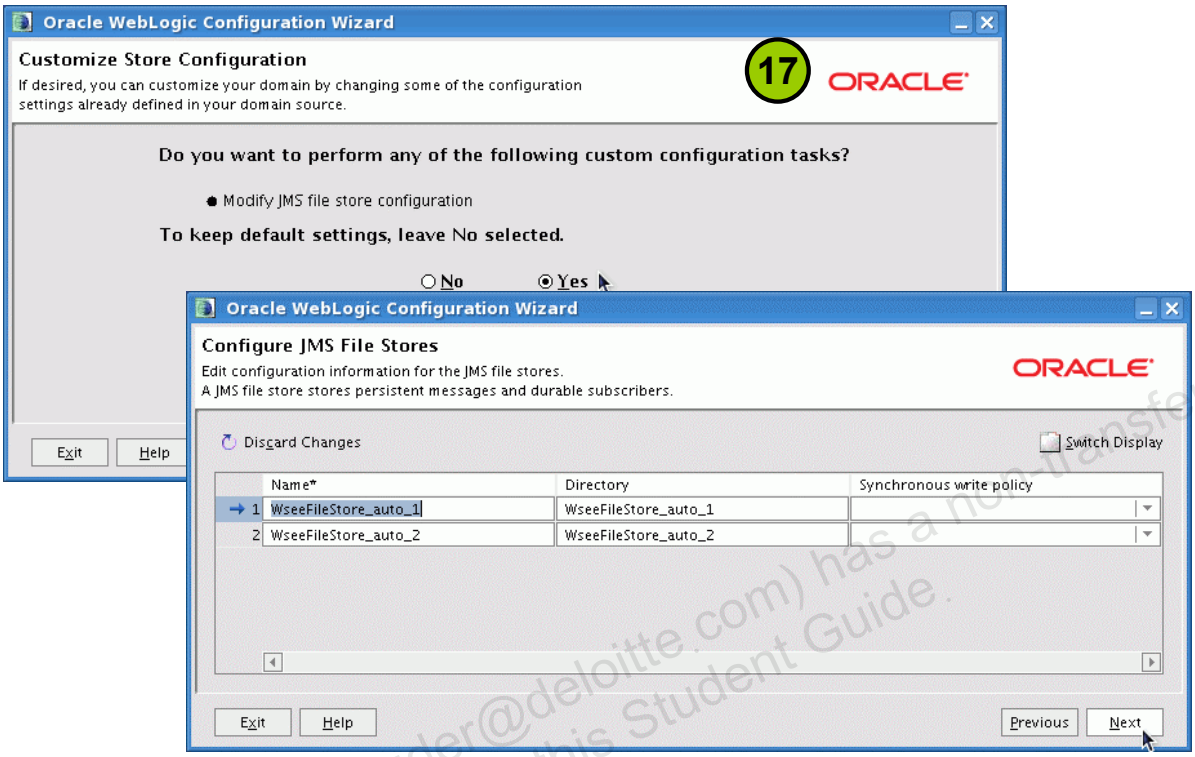
For all databases, the database server must be running to execute the SQL scripts.

1. In the Available JDBC Data Sources pane, select the data source for which you want to run the scripts. The scripts that can be executed are displayed in the Available SQL Files and Database Loading Options pane.
2. Select the database version from the DB Version menu.
3. Click Run Scripts.

All the scripts that are displayed in the Available SQL Files and Database Loading Options pane for the selected data source are executed, and the results are displayed in the Results pane. If you want to capture the test output in a log file, select the Log File check box and specify the location of the log file.

- Repeat steps 1 through 3 for each data source for which you want to execute SQL scripts.
- Click Next.

Configuring the JMS File Store



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring the JMS File Store

These screenshots configure the optional Java Message Service (JMS). A JMS file store is a disk-based file in which persistent messages can be saved.

You can modify the JMS file stores that are configured in your domain on the Configure JMS File Stores page, which is displayed when you click Next on the Run Database Scripts page. This step is optional.

Review the current list of JMS file stores. The default values may vary based on the domain source that you selected earlier.

Note: The wizard provides two display modes: a concise tabular view of all the defined components and an individual view, in which each component is represented by a tab. You view a particular component by clicking the corresponding tab. To toggle the display mode between table and tab formats, click Switch Display.

Modify the settings as required for your domain.

Name: Enter a name for the JMS file store. The name must be a string of characters and can include spaces. The name of the JMS file store must be unique among all the component names within the domain.

Directory: Enter the path of the directory (in your system) in which the JMS file store resides.

Configuring the JMS File Store (continued)

Synchronous write policy: From the list, select one of the following synchronous write policies to determine how the file store writes data to the disk:

- **Cache-Flush:** Transactions cannot be completed until all their write operations have been flushed to disk.
- **Direct-Write:** Write operations are performed directly to disk. Direct I/O is supported on most platforms. For a potential performance boost, file stores in direct I/O mode will automatically load a native I/O wlfileio2 driver. This driver is available on Windows, Solaris, HP, AIX, and Linux platforms. If this policy is active on an unsupported platform, the file store switches automatically to the cache-flush policy.
- **Disabled:** Transactions are complete as soon as the writes are cached in memory. When this policy is active, completion of transactions does not depend on waiting for writes to reach the disk.

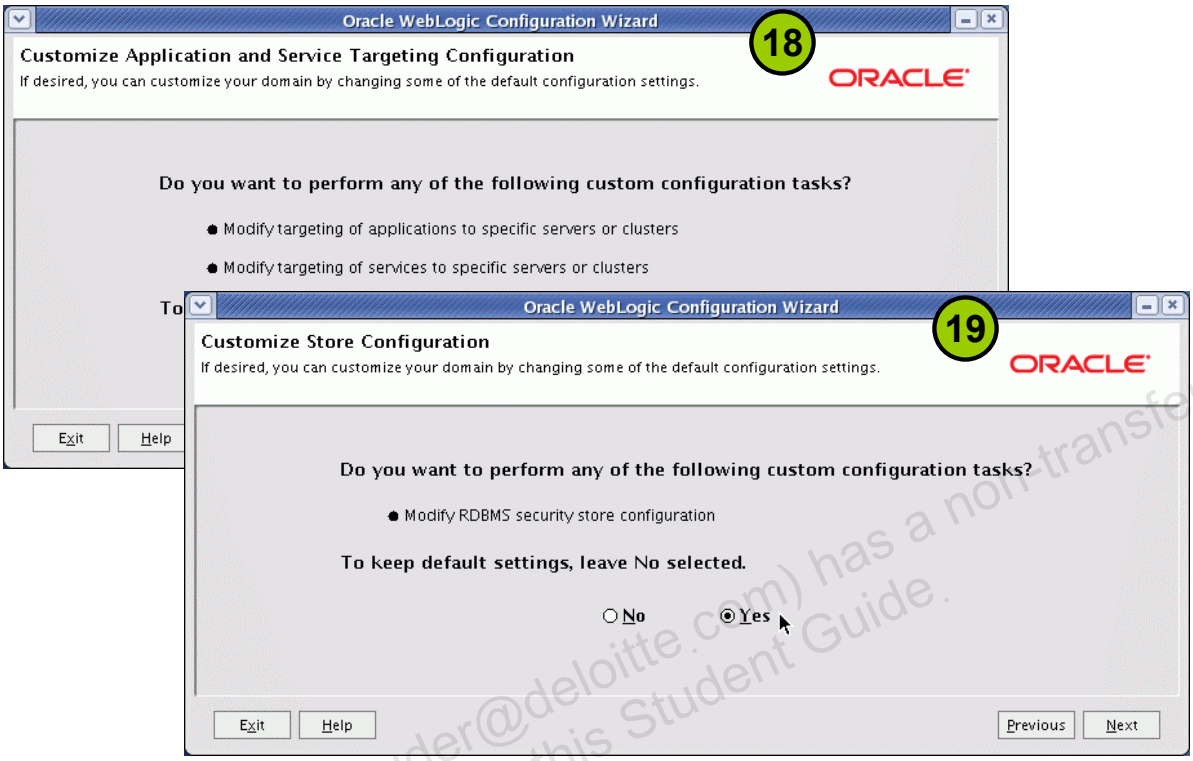
This setting affects performance, scalability, and reliability.

Note: The use of the direct-write policy is transactionally reliable in Solaris systems, but Windows systems may leave transaction data in the on-disk cache without writing it to disk immediately. This is not considered to be transactionally reliable because a power failure can cause loss of on-disk cache data, possibly resulting in lost or duplicate (or both) messages. For reliable writes using direct-write on Windows, either disable all write caching for the disk (enabled by default) or use a disk with a battery-backed cache. Some file systems, however, do not allow this value to be changed (for example, a redundant array of independent disks [RAID] disk system that has a reliable cache).

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Customizing Application and Service Targeting Configuration



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

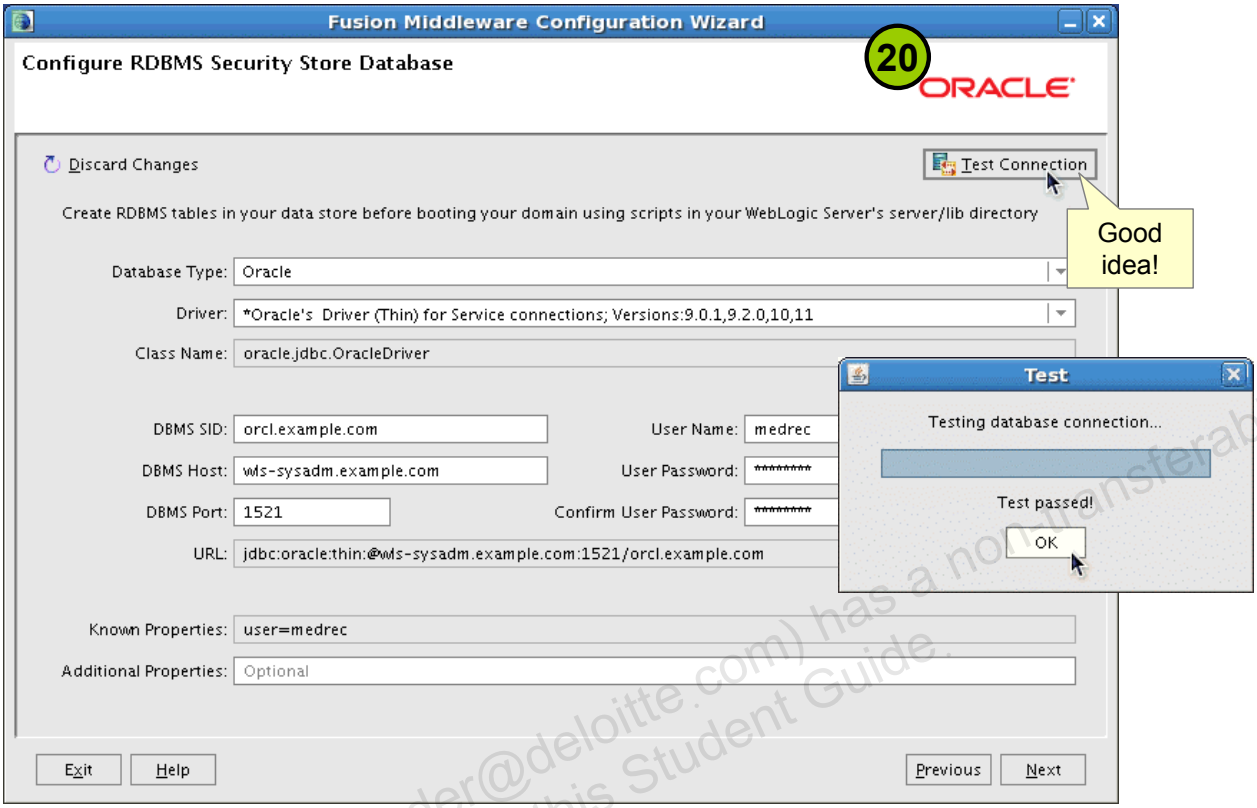
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Customizing Application and Service Targeting Configuration

The screens shown in the slide ask if you want to go down an optional path to configure more parameters. By default, both of these screens have No selected. If you leave it as No, you will skip the next few screens.

Configuring RDBMS Security Store Database



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring RDBMS Security Store Database

By default, database security is disabled—that is, all users, groups, and roles are stored in the embedded Lightweight Directory Access Protocol (LDAP) store of the administration server and this panel is disabled.

If you choose to use any of the supported databases to store this information, select the database type and populate the DBMS name, host and port, and username and password. The remaining fields are prepopulated for you based on this information. Before you start the server, you must load the necessary SQL scripts (`<WL_HOME>/server/lib`) for the RDBMS security store.

The following RDBMS systems can be used for the RDBMS security store:

- Oracle 9*i* Release 9.0.1 and 9.2.0
- Oracle 10*g*, Oracle 11*g*
- MS-SQL 7.0, 2000, and 2005
- DB2
- PointBase RDBMS
- Sybase
- MySQL
- Informix
- Ingres

See WebLogic Release notes for particular versions supported from other vendors.

Configuring RDBMS Security Store Database (continued)

Note: The Configuration Wizard has been modified to allow you to create the RDBMS security store when you create a domain. When you boot the domain, you can set additional configuration options for the RDBMS security store from the WebLogic Administration Console.

Oracle WebLogic Server provides a set of SQL scripts for each supported RDBMS system that must be run before booting the domain. These scripts create the tables in the data store that are used by the security providers.

Oracle WebLogic Server provides the option of using an external RDBMS as a data store that is used by the following security providers:

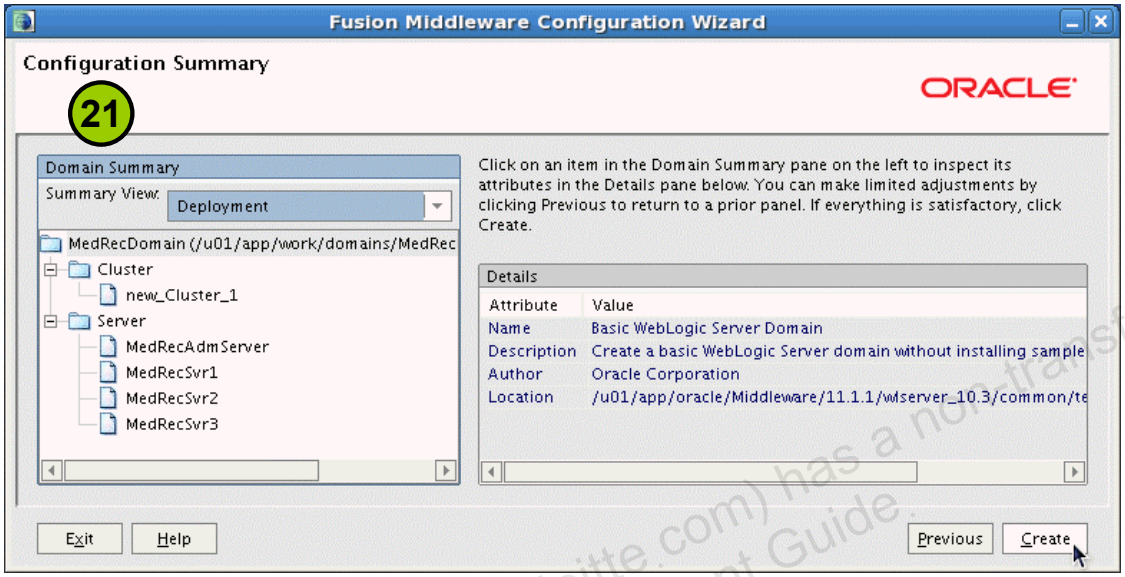
- XACML Authorization and Role Mapping providers
- WebLogic Credential Mapping provider
- PKI Credential Mapping provider
- The following providers for SAML 1.1:
  - SAML Identity Assertion provider V2
  - SAML Credential Mapping provider V2
- The following providers for SAML 2.0:
  - SAML 2.0 Identity Assertion provider
  - SAML 2.0 Credential Mapping provider
- Default Certificate Registry

The RDBMS security store is required if you want to use SAML 2.0 services in two or more Oracle WebLogic Server instances in a domain. Security store helps implement Single Sign On (SSO) between systems.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Reviewing the WebLogic Domain



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

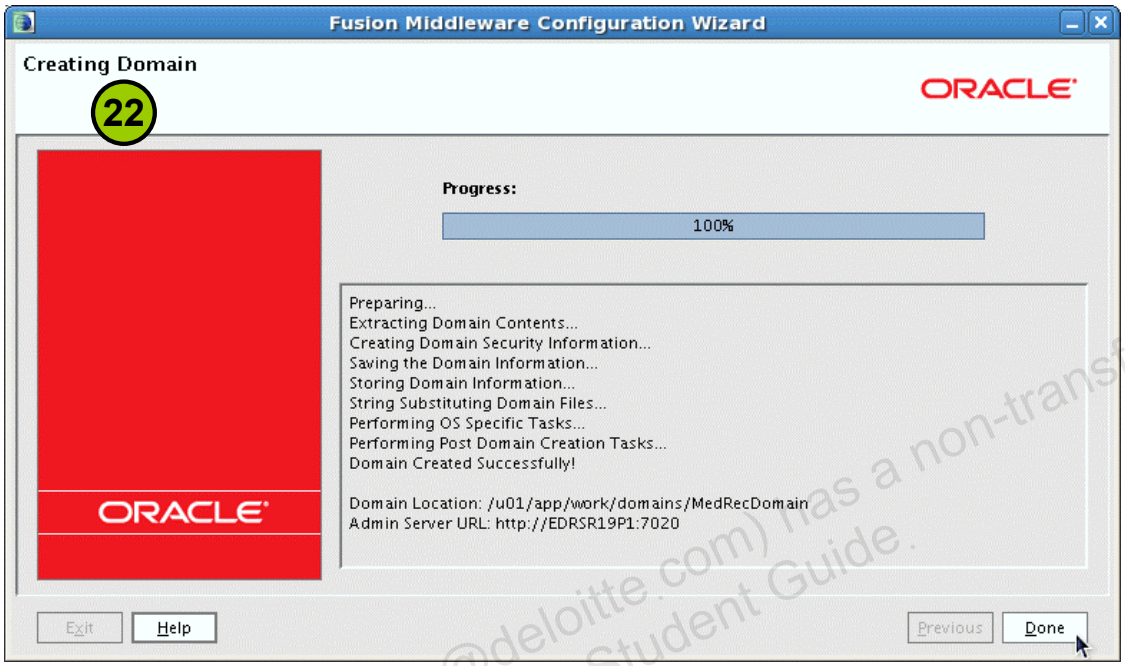
Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Reviewing the WebLogic Domain

The screenshot shows the summary so far. The choices in the Summary View drop-down list are the following:

- Deployment
- Application
- Service
- Cluster
- Machine
- JMS Server

Creating the WebLogic Domain



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating the WebLogic Domain

These screenshots show the progress indicator as the domain builds. The domain directory can be located anywhere in the system. By default, it resides in `<MW_HOME>/user_projects/domains/`*`DOMAIN_NAME`*, where `<MW_HOME>` is the directory that contains the product installation, and *`DOMAIN_NAME`* is the name of the domain directory defined by the selected template.

The Configuration Wizard stores the `config.xml` file and all other generated components in the domain directory that you specify. The domain is not started automatically, you must start it by hand later.

Note: You cannot overwrite an existing domain. If a domain with the name that you specify exists in the selected location, you must either delete the existing domain, or specify a different name or location for the new domain.

Domain Directory Structure

Directory	Description
🗁 *domain-name*	The name of this directory is the name of the domain.
🗀 autodeploy	In development mode, WLS automatically deploys any applications or modules that you place in this directory.
🗀 bin	The scripts for starting and stopping the administration server and the managed servers in the domain
🗀 config	The current configuration and deployment state of the domain; **config.xml**
🗀 console-ext	Console extensions
🗀 init-info	Domain initialization information
🗀 lib	JAR files added to CLASSPATH of each server instance
🗀 pending	Domain configuration changes that have been requested, but not yet been activated
🗀 security	Domainwide security-related files
🗁 servers	One subdirectory for each server in the domain
🗀 *server-name*	The server directory for the WLS instance with the same name

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@apache.org) has a non-transferable license to use this Student Guide.

Domain Directory Structure

The domain directory also contains the following directories:

- `tmp`: A directory for temporarily storing files. You should not modify any files in this directory.
- `user_staged_config`: This directory is an alternative to the `config` directory if the domain is set up such that the configuration information is "user-staged"—that is, the administrator is responsible for staging (copying to the managed servers) the configuration information.
- `pending`: This directory contains the domain configuration files that represent the configuration changes that have been requested, but not yet been activated. After the configuration changes are activated, the configuration files are deleted from this directory.
- `servers`: The `servers` directory that contains the subdirectories for the administration and managed servers is created the first time the servers are started. This directory contains one subdirectory for each Oracle WebLogic Server instance in the domain. The subdirectories contain data that is specific to each server instance.
- `lib`: Any JAR files that you put in this directory are added to the Java system `CLASSPATH` of each server instance in the domain when the server's Java Virtual Machine starts.
- **`init-info`:** This directory contains files that are used for WebLogic domain provisioning. You should not modify any files in this directory.

Domain Directory Structure (continued)

- security: This directory holds the security-related files that are the same for every Oracle WebLogic Server instance in the domain: SerializedSystemIni.dat
  - This directory also holds the security-related files that are needed only by the domain's administration server:
    DefaultAuthorizerInit.ldift
    DefaultAuthenticatorInit.ldift
    DefaultRoleMapperInit.ldift
- console-ext: This directory contains extensions to the Administration Console, which enable you to add content to Oracle WebLogic Server Administration Console, replace content, and change the logos, styles, and colors without modifying the files that are installed with Oracle WebLogic Server. For example, you can add content that provides custom monitoring and management facilities for your applications.
- config: This directory contains the current configuration and deployment state of the domain. The central domain configuration file, config.xml, resides in this directory. Ignore the /lib directory under /config.
- bin: This directory contains the scripts that are used for starting and stopping the administration server and the managed servers in the domain. These scripts are generally provided as .sh files for UNIX and .cmd files for Windows. The bin directory can optionally contain other scripts of domainwide interest, such as scripts to start and stop database management systems, full-text search engine processes, and so on.
- autodeploy: This directory provides a quick way to deploy applications in a development server. When the Oracle WebLogic Server instance runs in development mode, it automatically deploys any applications or modules that you place in this directory.

The files that you place in this directory can be Java EE applications, such as:

- An EAR file
- A WAR, EJB JAR, RAR, or CAR archived module
- An exploded archive directory for either an application or a module

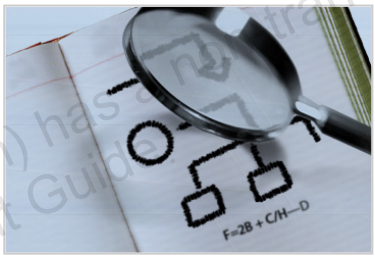
domain-name: The name of this directory is the name of the domain.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can McCrudden (camcrudden@gmail.com) has a non-transferable license to use this student guide.

Road Map

- Domains
- Starting and stopping the Oracle WebLogic Server



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

JVM Run-Time Arguments

- Oracle WebLogic Server can be executed with most Java Virtual Machines, such as Sun JVM or JRockit.
- Oracle WebLogic Server supports JDK 1.6.
- The syntax for running a virtual machine is:

```
java options FullyQualifiedJavaClass
   ProgramOptions
```

- Some virtual machine options:
  - `-Xms`: The minimum size of the dynamic heap
  - `-Xmx`: The maximum size of the dynamic heap
  - `-Dprop=val`: An environment variable that is accessible by the program
  - `-classpath CLASSPATH`: The list of files or directories that contain the dependent classes

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carl.mccruder@oracle.com) has a non-transferable license to use this Student Guide.

JVM Run-Time Arguments

You can see the default memory sizes used to start the WebLogic environment by entering `echo $MEM_ARGS`. You specify the startup minimum heap size with the `-Xms` flag. The `startWebLogic` utility gives a default value `-Xms256m` (that is, 256 megabytes). You specify the maximum heap size with the `-Xmx` flag. The `startWebLogic` utility gives a default value `-Xmx512m` (that is, 512 megabytes).

Options can be set using the `-D` options. `startWebLogic`, for example, has an option that sets the policy file.

Oracle WebLogic Server Dependencies

- To run Oracle WebLogic Server *itself*, `PATH` and `CLASSPATH` environment variables are usable without needing any additional modification.
  - Environment variables are set properly in start scripts, which call `setWLSEnv.sh`.
  - Start scripts are created during installation.
  - You may want to modify the start scripts' environment variables based on deployed applications' requirements.
- To run your *deployed applications* on Oracle WebLogic Server, configure the following environment variables:
  - `PATH` to include all executable programs (including the Java interpreter)
  - `CLASSPATH` to include dependencies for the applications

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle WebLogic Server Dependencies

The `PATH` environment variable provides the list of directories that your operating system uses to search for executable programs. Because the JVM is an executable program, the directory containing the virtual machine should be included in the `PATH` environment variable.

`CLASSPATH` is the list of directories that the virtual machine uses to search for dependent classes in a program. You can set `CLASSPATH` in an environment variable or as a command-line parameter when you execute the virtual machine.

Configuring CLASSPATH

- The Oracle WebLogic Server CLASSPATH is configured by the Java system CLASSPATH environment variable.
- Files that *must* be in CLASSPATH:
  - <WL_HOME>/server/lib/weblogic.jar
  - Any additional service pack JAR files
- Files that *can* be in CLASSPATH:
  - <WL_HOME>/common/derby/lib/derby.jar
  - <WL_HOME>/server/lib/log4j.jar
  - <WL_HOME>/server/lib/wlepool.jar
  - <WL_HOME>/server/lib/wleorb.jar
- Additional related environment variables:
  - PRE_CLASSPATH
  - POST_CLASSPATH

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring CLASSPATH

In Oracle WebLogic Server 10.3.3, the only file that needs to be configured in CLASSPATH is weblogic.jar, which is the primary distribution file. If there are patches that need to be in the CLASSPATH, those are handled with the setPatchEnv.sh script. You can optionally include other files in CLASSPATH depending on whether or not you would be using Java EE Connector Architecture (JCA) connectivity or another database.

Note: To use any of the command-line utilities that are distributed with Oracle WebLogic Server, such as WLST, you must have your CLASSPATH set with the files listed in the slide. The files are needed for Oracle WebLogic Server to boot and to use utility operations.

The real name for a Java class is always its package name prepended to its class name. This is often called the fully qualified name of the Java class. When referencing a class from within a Java program or through a virtual machine, you should always refer to its fully qualified class name.

Also, in order for the virtual machine to locate the class, the root directory of the package should be included in CLASSPATH. The root directory is included because the virtual machine knows which subdirectories to look under when it sees the fully qualified name of the class. Only root class directories and JAR files should be in the Java system CLASSPATH.

Configuring CLASSPATH (continued)

If you built a class named AWorkingClass.class and if the class is in a package named com.bea.examples and the file is located at c:\student\files\com\bea\examples\AWorkingClass.class, your CLASSPATH should include c:\student\files\ to execute the java com.bea.examples.AWorkingClass program.

After installation, Oracle WebLogic Server's CLASSPATH is already set, but you may choose to modify it for a number of reasons such as adding a patch to Oracle WebLogic Server, updating the version of Derby that you are using, or adding support for Log4j logging.

To apply a patch to all your Oracle WebLogic Server domains without the need to modify the CLASSPATH of a domain, give the patch JAR file the name, weblogic_sp.jar, and copy it into the <WL_HOME>/server/lib directory. The commEnv.cmd/sh script automatically includes a JAR named weblogic_sp on the CLASSPATH for you.

If you would rather not use the name weblogic_sp.jar for your patch file or you would just like to ensure that a JAR file, such as the one mentioned as follows, comes before weblogic.jar on the CLASSPATH, perform the following:

- For all domains, edit the commEnv.cmd/sh script in <WL_HOME>/common/bin and prepend your JAR file to the WEBLOGIC_CLASSPATH environment variable.
- To apply a patch to a specific Oracle WebLogic Server domain, edit the setDomainEnv.cmd/sh script in that domain's bin directory and prepend the JAR file to the PRE_CLASSPATH environment variable.

If you use the trial version of Derby, an all-Java database management system, include the following files on the CLASSPATH:

- <WL_HOME>/common/derby/lib/derby.jar

If you use WebLogic Enterprise Connectivity, include the following files on the CLASSPATH:

- <WL_HOME>/server/lib/wlepool.jar
- <WL_HOME>/server/lib/wleorb.jar

If you use Log4j logging, include the following file on the CLASSPATH:

- <WL_HOME>/server/lib/log4j.jar: The shell environment in which you run a server determines which character you can use to separate the path elements. On Windows, you typically use a semicolon (;). In a BASH shell, you typically use a colon (:).

Example

To add the path for the Derby database to the end of the CLASSPATH variable, edit setWLSEnv.sh and add the lines:

```
POST_CLASSPATH="${WL_HOME}/common/derby/bin/derby.jar"
export POST_CLASSPATH
```

immediately below "export WL_HOME" (approximately line 39).

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrude (carlmgruder@deloitte.com) has a non-transferable license to use this Student Guide

Starting Oracle WebLogic Administration Server

Start the Administration Server by using the following:

- *DOMAIN_NAME*/bin/startWebLogic.sh
- WebLogic Scripting Tool (WLST) and Node Manager
- WLST without Node Manager
- Start menu (only Windows)
- A custom script calling `weblogic.Server` (only in development)

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting Oracle WebLogic Administration Server

You can invoke the class directly in a command prompt (shell), indirectly through provided or custom scripts, or through the Node Manager.

There are a variety of ways to execute and start Oracle WebLogic Server. Oracle WebLogic Server provides several ways to start and stop server instances. The method you choose depends on whether you prefer using the Administration Console or a command-line interface, and on whether you are using Node Manager to manage the server's life cycle. Eventually, regardless of the technique that you use, the `weblogic.Server` class is executed by a virtual machine installed on the machine via a command script. The server instance runs within the JVM, and the JVM can host only one server instance.

Executing the class directly through a virtual machine or through the `startWebLogic.cmd` file is the preferred method for starting and executing Oracle WebLogic Server. Executing Oracle WebLogic Server through one of these techniques allows Oracle WebLogic Server to pick up your Java system `CLASSPATH` and other settings.

If you already set up your Java system `CLASSPATH`, one option is to just run the `weblogic.Server` class. If you want to include a special `CLASSPATH`, you can use the `-cp` option with the virtual machine. You would need to configure the `java.security.policy` environment properties. The `weblogic.policy` file provided with your Oracle WebLogic Server installation sets up any authorizations required to execute Oracle WebLogic Server.

Starting Oracle WebLogic Administration Server (continued)

In addition to entering the entire command manually (which not likely; it is very long), the Oracle WebLogic Server installation provides a number of default scripts for starting up Oracle WebLogic Server and setting environment variables for command-line use. Under each installed domain is a `startWebLogic.cmd` or `.sh` script for starting an instance of Oracle WebLogic Server manually. These scripts are customized by the installation and appropriately reflect the location in which Oracle WebLogic Server is installed.

Starting the Administration Server Using the Start Menu in Windows

When you create an administration server on a Windows computer, the Configuration Wizard creates a shortcut on the Start menu for starting the server. (The menu option is: User Projects > `DOMAIN_NAME` > Start Admin Server for WebLogic Domain.) The command that the Configuration Wizard adds to the Start menu opens a command page and calls the startup script (`startWebLogic.cmd`).

Note: Starting the administration server using WLST and Node Manager or using WLST by itself is covered later in the lesson titled "Configuring Node Managers."

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting Administration Server Using `startWebLogic.sh`

Run *DOMAIN_NAME*/bin/startWebLogic.sh.

- Sets the environment by using `setDomainEnv.sh`
- Invokes `java weblogic.Server`

```
<Feb 2, 2010 11:27:29 AM EST> <Info> <Management> <BEA-141107> <Version: WebLogic
     Server 10.3.3.0  Mon Jan 19 23:37:46 EST 2010 1185576 >
<Feb 2, 2010 11:27:31 AM EST> <Info> <Security> <BEA-090065> <Getting boot identity
     from user.>
Enter username to boot WebLogic server:weblogic
Enter password to boot WebLogic server:********
<Feb 2, 2010 11:27:43 AM EST> <Notice> <WebLogicServer> <BEA-000365> <Server state
     changed to STARTING>
<Feb 2, 2010 11:27:43 AM EST> <Info> <WorkManager> <BEA-002900> <Initializing self-
     tuning thread pool>
```

```
<Feb 2, 2010 11:28:01 AM EST> <Notice> <WebLogicServer> <BEA-000329> <Started
     WebLogic Admin Server "MedRecAdmSvr" for domain "MedRecDomain" running in
     Production Mode>
<Feb 2, 2010 11:28:02 AM EST> <Notice> <WebLogicServer> <BEA-000365> <Server state
     changed to RUNNING>
<Feb 2, 2010 11:28:02 AM EST> <Notice> <WebLogicServer> <BEA-000360> <Server
     started in RUNNING mode>
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting Administration Server Using `startWebLogic.sh`

Navigate to the directory in which you located the domain. By default, this directory is <MW_HOME>/user_projects/domains/*DOMAIN_NAME*, where *DOMAIN_NAME* is the root directory of the domain. (The name of this directory is the name of the domain.) Run one of the following scripts:

- `bin\startWebLogic.cmd` (Windows)
- `bin/startWebLogic.sh` (UNIX and Windows. On Windows, this script supports the MKS and Cygnus BASH UNIX shell emulators.)

Replace the username (default is `weblogic`) and password (shown as asterisks) as appropriate. Remember that passwords must have at least one uppercase letter, one lowercase letter, and a number, for example, **`Welcome1`**.

Note: If you use a Configuration Wizard template that is provided by Oracle WebLogic Server, your domain directory includes a start script named `startWebLogic`. If you use a domain template from another source, the wizard might not create a start script, or it might create a script with a different name. The template designer determines whether the wizard creates a start script and the name of the script.

Starting Administration Server Using `startWebLogic.sh` (continued)

The `startWebLogic` script performs the following steps:

1. It sets the environment variables by invoking *DOMAIN_NAME*`/bin/setDomainEnv.sh` (or `.cmd`), where *DOMAIN_NAME* is the directory in which you located the domain—for example, `<WL_HOME>/user_projects/domains/`*DOMAIN_NAME*, where `<WL_HOME>` is the location in which you installed Oracle WebLogic Server. The `setDomainEnv` script sets environment variables such as `WL_HOME`, `BEA_HOME`, `SUN_JAVA_HOME`, `JAVA_HOME`, `SAMPLES_HOME`, `DOMAIN_HOME`, `PRODUCTION_MODE`, `PROXY_SETTINGS`, `MEM_ARGS`, `CLUSTER_PROPERTIES`, `JAVA_PROPERTIES`, `JAVA_OPTIONS`, `CLASSPATH`, `PRE_CLASSPATH`, `POST_CLASSPATH`, and others. The `setDomainEnv` script has no screen output.
2. It invokes the `java weblogic.Server` command, which starts a JVM that is configured to run an Oracle WebLogic Server instance.

When the server successfully completes its startup process, it writes the following message to standard out (which, by default, is the command page):

```
<Notice> <WebLogicServer> <BEA-000360> <Server started in RUNNING mode>
```

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting the Administration Server by Using the `java weblogic.Server` Command

- Run `<WL_HOME>/server/bin/setWLSEnv.sh`.
- Run `java weblogic.Server`.
- Optionally, run `java weblogic.Server` with these additional options:

```
java -server -Xms256m -Xmx512m -classpath
"CLASSPATH"
-Dweblogic.Name=SERVER_NAME
-Dplatform.home=<WL_HOME>
-Dweblogic.management.username=WLS_USER
-Dweblogic.management.password=WLS_PW
-Djava.security.policy=
<WL_HOME>/server/lib/weblogic.policy
 weblogic.Server
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting the Administration Server by Using the `java weblogic.Server` Command

There are several ways to specify the system administrator's password at startup time:

- Enter the password when prompted.
- Use the `boot.properties` file (stores the password encrypted).
- Specify the password in the startup command (for example, in a start script*) using:
  `-Dweblogic.management.password=`*`weblogic`*

*Leaving the password in clear text in a file can be a security hazard. But this is necessary when starting WLS unattended (for example, as a service or daemon). Make sure that the file containing the password is protected against unauthorized access. Do not make the file readable to all.

The `weblogic.Server` class is the main class for an Oracle WebLogic Server instance. You start a server instance by directly invoking `weblogic.Server` in a Java command.

Note: It is recommended that you use `java weblogic.Server` primarily for initial development (if at all) but not as a standard mechanism for starting production systems for the following reasons:

- The `java weblogic.Server` class will not function if you select a product directory outside of the Oracle home directory.
- When executing `java weblogic.Server`, patches will not be recognized by the Oracle WebLogic Server run time.
- There is way too much to enter each time. This would be useful to include if you wanted to build your own start script. Alternatively, use `startWebLogic.sh` as a template to modify your own start script.

Stopping the Administration Server

- Gracefully
  - Via GUI:
    - Stop the server from the Administration Console.
    - This also closes the Administration Console, so restarting requires a command-line action.
  - Via CLI: `domain_name/bin/stopWebLogic.sh`
- Abruptly
  - Press Ctrl + C to interrupt the running `startWebLogic` program.
  - Usually, your applications are running on managed servers, not on the Administration Server; so even though this is abrupt, it is not disruptive.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Stopping the Administration Server

Stopping the WebLogic part of the domain can be done in several ways:

- If the process is running in a terminal session, you can press Ctrl + C to stop the process.
- The `domains/DOMAIN_NAME/bin/stopWebLogic.sh` script will shut down only the administration server.
- You can shut down servers using the Administration Console, including the administration server.

The Middleware environment may consist of more than just the domain. It may consist of other components that are not managed from the Administration Console—for example, the Oracle HTTP Server and Web Cache. Those non-Java components may be managed by Oracle Process Manager and Notification (OPMN), which has its own mechanisms for graceful shutdowns.

Stopping the Administration Server does not stop the managed servers. There is no single command to stop a whole domain; you have to stop the individual components separately. Nevertheless, this is a very good script to create for yourself early in the rollout of a domain. The Web-based Administration Console has a single check box that can stop everything.

Quiz

Which directory within a domain directory is used to maintain its configuration repository?

a. `/console`
b. `/cache`
c. `/config`
d. `/logs`
e. `/AdminServer`

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: c

Changes made to a domain's configuration are recorded on the file system in the `/config` directory.

Quiz

Invoke the domain Configuration Wizard by using _____.

a. `config.sh` under `<WL_HOME>/common/bin`

b. `config_builder.sh` under `<WL_HOME>/common/bin`

c. `pack.sh`

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

`config_builder.sh` is used to invoke the Domain Template Builder.

Quiz

What is the main configuration file for the domain called?

a. `configuration.xml`
b. `wlsconfig.xml`
c. `wls.xml`
d. `config.xml`

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: d

The central domain configuration file is called `config.xml` and it resides in the `config` subdirectory of the domain's root directory.

Quiz

Which of the following statements is *NOT* true?

a. Managed servers in a domain may run a different OS version of Oracle WebLogic Server (for example, Windows + Linux).

b. A domain comprises only the administration server, only the managed server, or the administration and managed servers.

c. The administration server stores the configuration information and logs for a domain.

d. The administration server in a domain must run the same or later version number of Oracle WebLogic Server as the managed servers in the domain.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

A domain must consist of one administration server. Managed servers are optional.

Quiz

Where are all users, groups, and roles stored by default?

a. Oracle Database 11*g*
b. Derby database
c. Oracle Internet Directory
d. LDAP store of the administration server

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: d

By default, database security is disabled—that is, all users, groups, and roles are stored in the embedded LDAP store of the administration server.

Summary

In this lesson, you should have learned how to:

- Describe the common elements in a WebLogic Server domain
- Describe how domains are used in the enterprise
- Compare administration and managed servers
- Configure a domain
- Describe the organization and contents of the WLS directory structure
- Describe the use of WLST offline to manage domains
- Create a simple domain with one managed server
- Check the port numbers that are used for components
- Use scripts to start and stop the servers

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 5 Overview: Configuring a Simple Domain

This practice covers the following topics:

- Creating a minimal domain from scratch
- Creating a new domain to support a particular application template
- Starting and stopping the administration server and the managed servers

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 5 Overview: Configuring a Simple Domain

See Appendix A for the complete steps to do the practice.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

6

Configuring a Domain Using Templates

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Describe the internal process used to create or update a domain using a template
- Describe the process of extending a domain template
- Explain the domain templates provided for setting up JDeveloper, SOA, and WebCenter

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

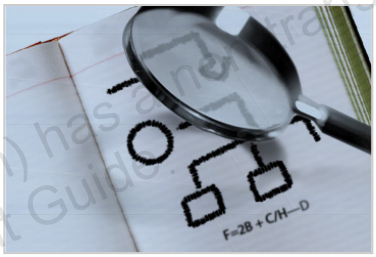
Objectives

Scenario

As the administrator of the middleware, you finally got the domain the way you like it, and eventually you plan to roll this out to dozens of sites. To make the job of rolling out the implementation easier at the remote site, you want to create a template that names and sizes all the required components beforehand. Also, these new sites use JDeveloper. You want to use tools to help automate the creation of these new domains.

Road Map

- Creating a custom domain template
  - Building a template
  - Extending a domain
- Supporting other environments



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Custom Domain Templates

- A domain template defines the full set of resources within a domain.
- Oracle provides sample templates for creating any platform domain.
- There are three ways to create domain templates:
  - WLST offline command-line tool
  - `pack` command
  - Domain Template Builder
- You can use the Domain Template Builder to create a domain template or an Extension template.
- Using the Domain Template Builder, you can:
  - Define a domain and replicate it across multiple projects
  - Distribute a domain packed with an application that has been developed to run in it

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Custom Domain Templates

A domain template defines the full set of resources within a domain, including infrastructure components, applications, services, security options, and general environment and operating system parameters. You can create a domain template from an existing template or from a domain. A domain template is used to create a new domain. Oracle provides a base Oracle WebLogic Server domain template. Domain templates are available in the `<WL_HOME>/common/templates/domains` directory, and extension templates are available in the `<WL_HOME>/common/templates/applications` directory.

The domain template defines the core set of resources within a domain, including an administration server and basic configuration information, infrastructure components, and general environment and operating system options. It does not include sample applications. You can use this template to create a basic Oracle WebLogic Server domain that you can then extend using an extension template that contains applications and services, or additional product component functionality.

The term template refers to a Java Archive (JAR) file that contains the files and scripts required to create or extend a domain.

Custom Domain Templates (continued)

With the Domain Template Builder, you can create two types of templates: Domain templates or Extension templates. An Extension template defines the applications and services that can provide additional product component functionality, such as Apache Beehive, product sample applications, or Java Database Connectivity (JDBC) or Java Message Service (JMS) components. This type of template can be used to extend an existing domain.

Any template that you create with the Configuration Template Builder is used as input to the Configuration Wizard. The Configuration Wizard uses it, in turn, as the basis for creating a domain that is customized for your target environment.

The `pack` command enables you to create a template archive (`.jar`) file that contains a snapshot of either an entire domain or a subset of a domain. You can use a template that contains the subset of a domain to create a managed server domain directory hierarchy on a remote machine.

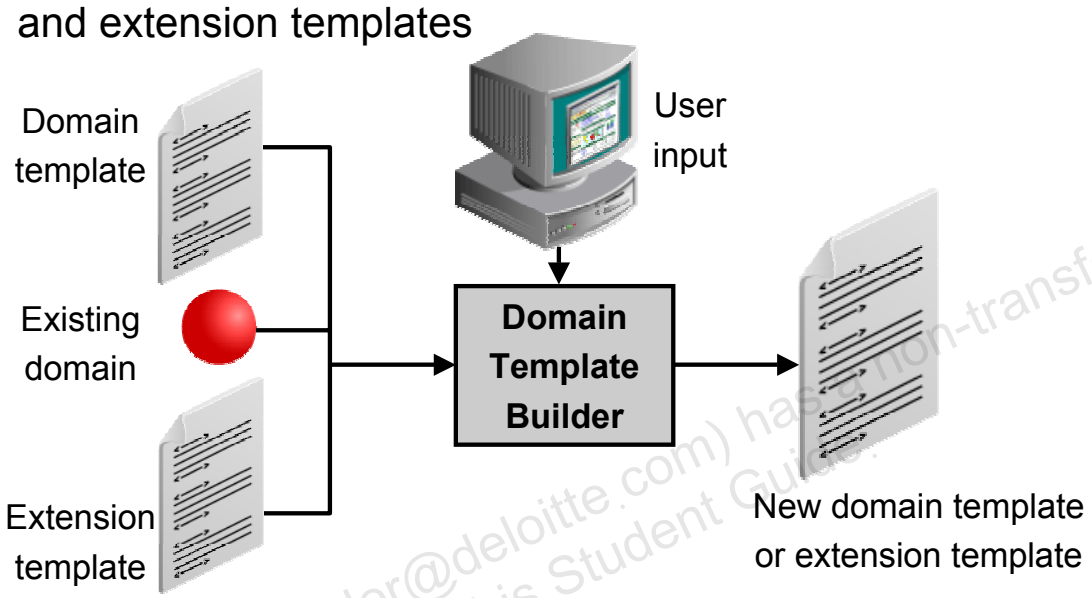
The WLST offline command-line scripting interface is used to create a new domain or update an existing domain without connecting to a running instance of Oracle WebLogic Server. The `writeTemplate` functionality of WLST offline provides the same capability for creating a template as the `pack` command. The `pack` command is covered in more detail in the lesson titled "Configuring Managed Servers."

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Domain Template Builder

- Available only in GUI mode and not in console mode
- A stand-alone Java application to create custom domain and extension templates



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Domain Template Builder

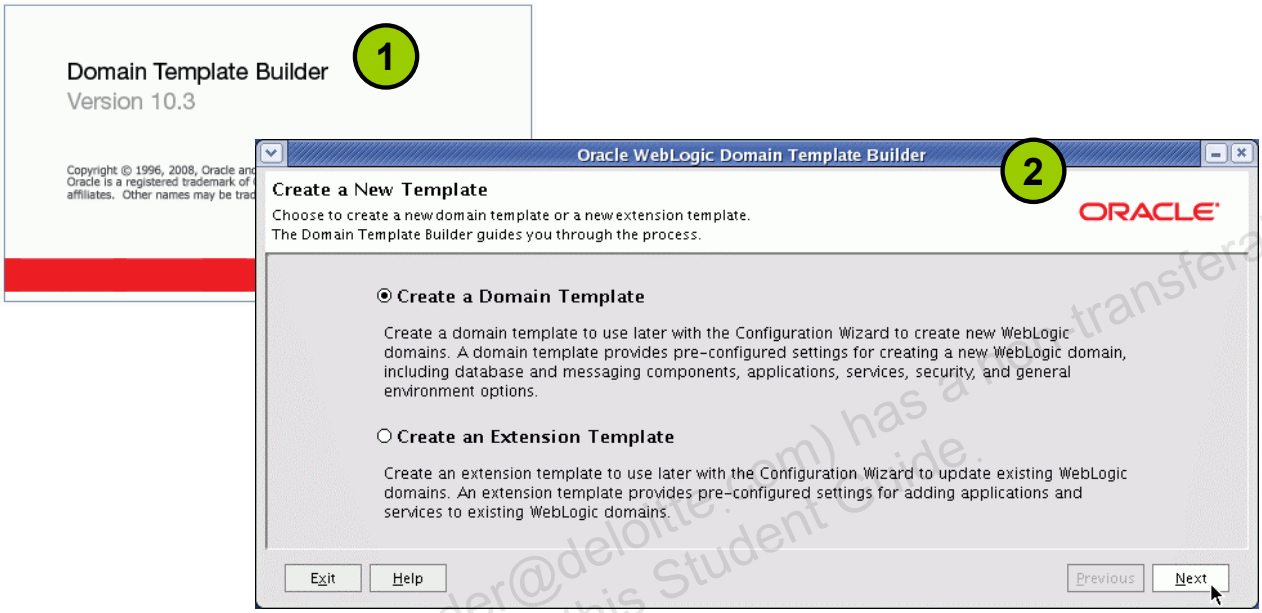
The Domain Template Builder is designed to be used offline, only in graphical mode. It is not supported in console mode. However, you can create templates from the command line by using the `pack` command.

After you create your domain and add new resources and applications to it, you can use the Domain Template Builder to create a custom domain template. You can also use the Domain Template Builder to customize an existing template. For example, you may want to remove applications or add SQL scripts for additional databases. To do so, select the existing domain or template as the source for your new custom template. When your custom domain templates are complete, you can start using them to create domains using the Configuration Wizard, WLST offline, or the `unpack` command.

Creating a Domain Template

Start the Domain Template Builder:

```
<WL_HOME>/common/bin/config_builder.sh
```



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Cam McCruder (camcruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Domain Template

Creating a Domain Template: Create a template that defines the full set of resources within a domain, including infrastructure components (for example, administration servers and managed servers), applications, services, security options, and general environment and operating system parameters. You can use the template that you create as the basis for creating a domain by using the Configuration Wizard, WLST, or the `unpack` command.

Creating an Extension Template: Create a template that defines the applications and services that can be used to extend existing domains, but does not include any server information. You can import the applications and services that are stored in extension templates into an existing domain by using the Configuration Wizard.

Comparing Domain Templates

- The template that comes with WebLogic Server:
  - Located in `<MW_HOME>/wlserver_10.3/common/templates/domains` by default, or wherever you want to store it
- The template that was just created by Domain Template Builder
  - Located in `<MW_HOME>/user_templates` by default, or wherever you want to store it

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Comparing Domain Templates

The following slides show the directory structure of the JAR and a portion of the `config.xml` file.

wls.jar Template

Users, groups, ACLs

For Windows

Substitutions

Shown below

Placeholders

WinZip Pro - wls.jar

Name	Path	Size
fileRealm.properties		336
security.xml		2,846
startmenu.xml		1,205
startscript.xml		61,759
stringsubs.xml		1,254
template-info.xml		488
readme.txt	autodeploy\	2,345
wlscontrol.sh	bin\nodemanager\	29,533
wlsifconfig.cmd	bin\server_migration\	5,812
wlsifconfig.sh	bin\server_migration\	13,838
readme.txt	bin\service_migration\	1,621
config.xml	config\	542
readme.txt	config\deployments\	248
readme.txt	config\diagnostics\	103
readme.txt	config\jdbc\	135
readme.txt	config\jms\	133
readme.txt	config\lib\	133
readme.txt	config\security\	200
readme.txt	config\startup\	162
readme.txt	console-ext\	76
readme.txt	lib\	708
Manifest.mf	meta-inf\	148

Selected 0 files, 0 bytes | Total 22 files, 121KB

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrider (carlmccrider@deloitte.com) has a non-transferable license to use this Student Guide.

wls.jar Template

The schema in the slide is validated as from BEA and version 9.0 even though it ships with the current 10.3.3 version. Many of the `readme.txt` files exist only to preserve the required but empty directories.

config.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<domain xmlns="http://www.bea.com/ns/weblogic/90/domain"
xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.bea.com/ns/weblogic/90/domain.xsd">
  <name>base_domain</name>
  <server>
      <name>AdminServer</name>
  </server>
  <configuration-version>9.0.1.0</configuration-version>
  <admin-server-name>AdminServer</admin-server-name>
</domain>
```

MedRecTemplate.jar Template

Calls setDomainEnv

Missing Manifest.mf

WinZip Pro - MedRecTemplate.jar

Name	Path	Size
fileRealm.properties		336
security.xml		2,976
startmenu.xml		918
startscript.xml		54,450
stringsubs.xml		1,279
template-info.xml		422
readme.txt	autodeploy\	2,345
wlscontrol.sh	bin\nodemanager\	29,533
wlsifconfig.cmd	bin\server_migration\	5,812
wlsifconfig.sh	bin\server_migration\	13,838
readme.txt	bin\service_migration\	1,621
config.xml	config\	980
readme.txt	config\deployments\	248
readme.txt	config\diagnostics\	103
readme.txt	config\jdbc\	135
readme.txt	config\jms\	133
readme.txt	config\lib\	133
readme.txt	config\security\	200
readme.txt	config\startup\	162
readme.txt	console-ext\	76
readme.txt	lib\	708

Selected 0 files, 0 bytes | Total 21 files, 114KB

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

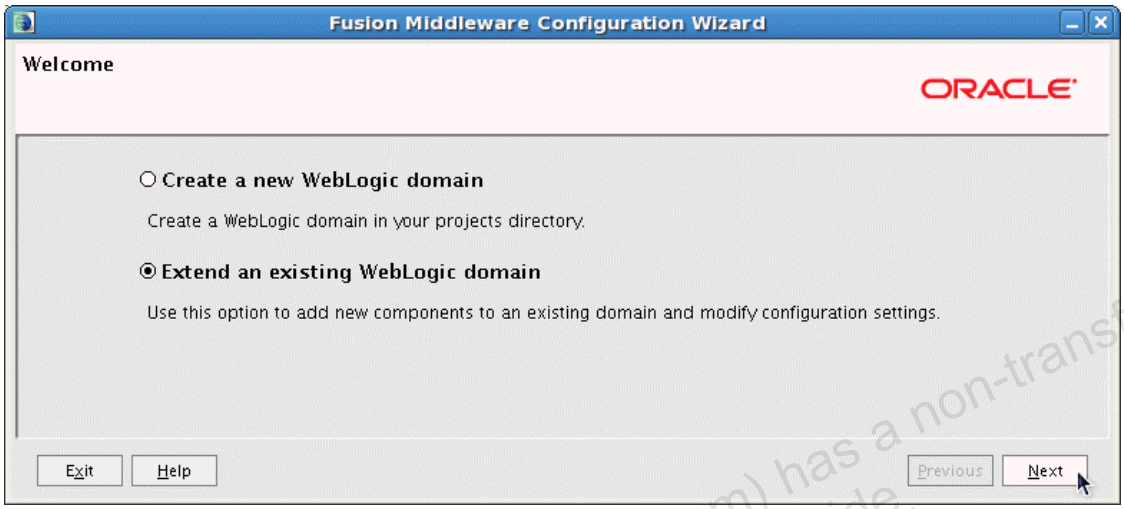
MedRecTemplate.jar Template

The schema in the slide is from Oracle samples via OTN and the version is 10.3.1.

config.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<domain
xsi:schemaLocation="http://xmlns.oracle.com/weblogic/security/wls
http://xmlns.oracle.com/weblogic/security/wls/1.0/wls.xsd
http://xmlns.oracle.com/weblogic/domain
…snip four lines…
http://xmlns.oracle.com/weblogic/security/xacml/1.0/xacml.xsd"
xmlns="http://xmlns.oracle.com/weblogic/domain"
xmlns:sec="http://xmlns.oracle.com/weblogic/security"
xmlns:wls="http://xmlns.oracle.com/weblogic/security/wls"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <name>base_domain</name>
  <domain-version>10.3.1.0</domain-version>
  <server>
    <name>MedRecAdmSvr</name>
    <listen-port>7020</listen-port>
    <listen-address/>
  </server>
  <configuration-version>10.3.1.0</configuration-version>
  <admin-server-name>MedRecAdmSvr</admin-server-name>
</domain>
```

Extending an Existing Domain



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

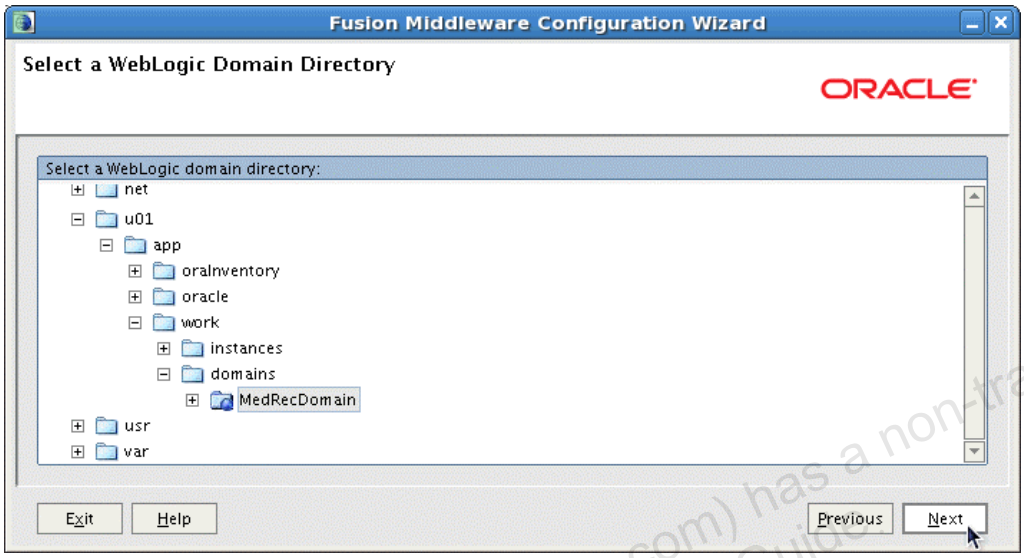
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Extending an Existing Domain

The first time you used the `./config.sh` Configuration Wizard, you selected to create a new domain. After you have a minimal functional domain, which might be as small as just an Administration Server, or as large as dozens of managed servers, you can come back to it and extend it to support additional functions such as Oracle SOA, or Oracle Identity Management, or an application that you provide. Select Extend and click Next.

Select Which Existing Domain to Extend



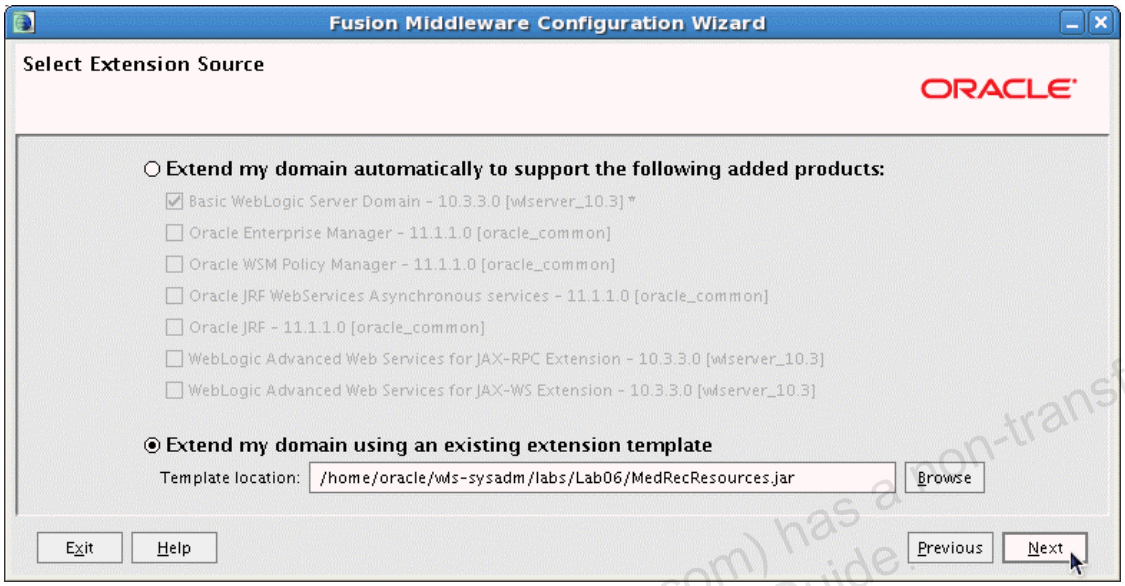
ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Select Which Existing Domain to Extend

The domain should be idle (no servers running). Select a domain that was created (and presumably tested) earlier. When the domain gets extended, it will overlay the old one, so you may want to back up or copy the old domain before proceeding.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Select What You Want to Extend it With



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

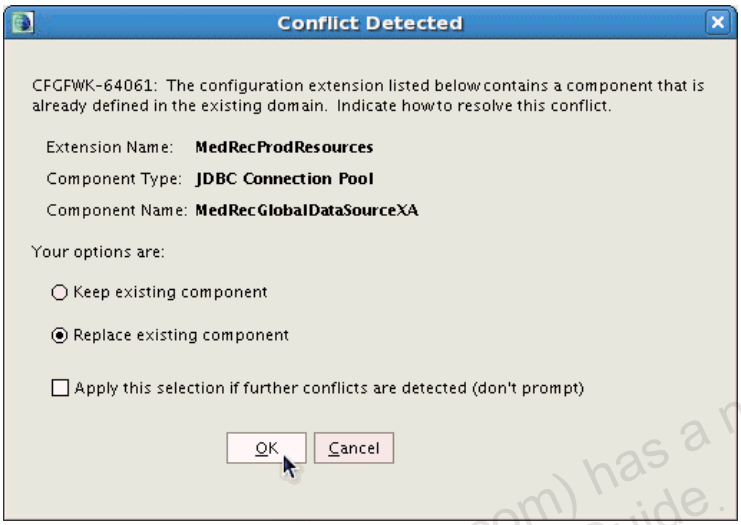
Select What You Want to Extend it With

You can see in the disabled section at the top a list of Oracle products that you can select to extend your domain. They show up because there are templates shipped with WebLogic Server that describe the functions that they need in a domain in order to run. If Oracle shipped more products not in this list, those products would also supply their own templates that would show up here as well. This option is covered later in this lesson.

You can specify your own template in the lower selection box. Navigate to the MedRec template that is part of the samples included with this class. This template (and other samples) is available on OTN at http://www.oracle.com/technology/products/webcenter/avitek_sample_portal.html (mind the underscores in the URL).

The template includes not only the application itself, but libraries needed to support the application, and also the definitions for the managed servers and other resources, such as JMS and JDBC for the application.

Resolving Conflicts



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

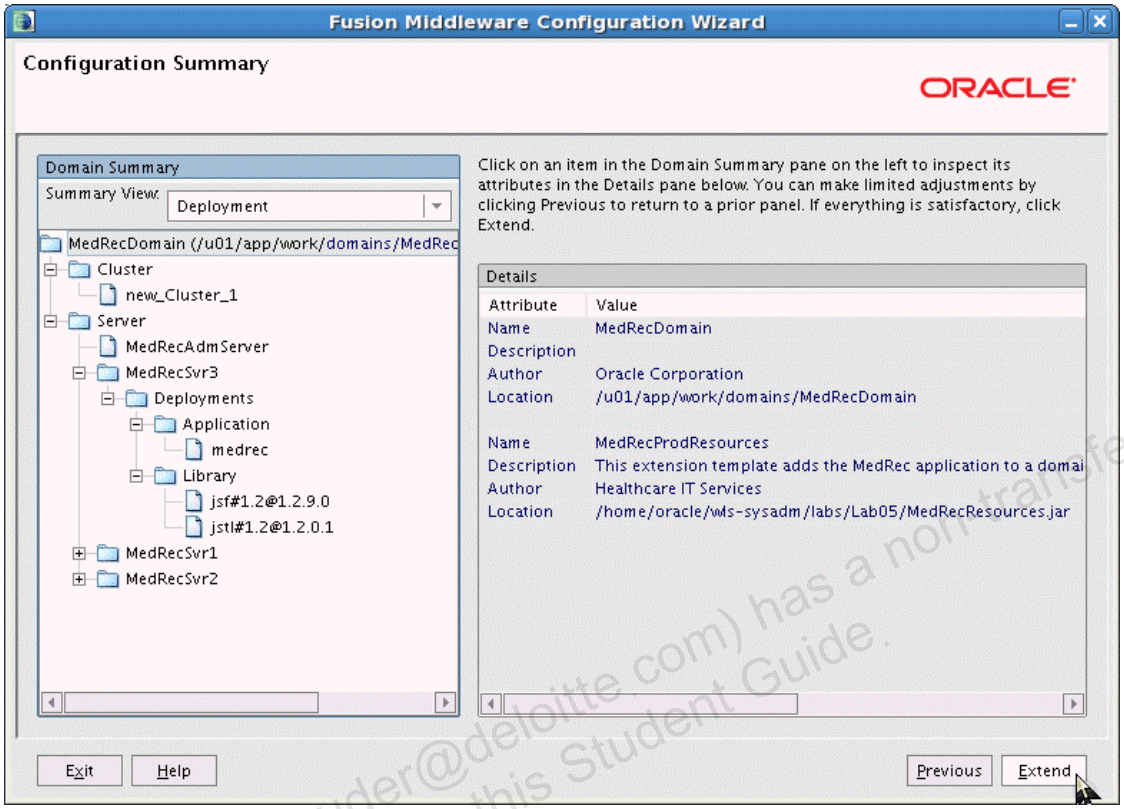
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Resolving Conflicts

It is possible that your existing domain specifies a resource such as a managed server named Server1 and that the template also specifies a server named Server1. If there is a conflict, you have a choice to let the existing structure remain, or to replace it with the template definitions. This decision can be made for each item or can be applied to all future conflicts in that domain.

Summary Before Extending



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Summary Before Extending

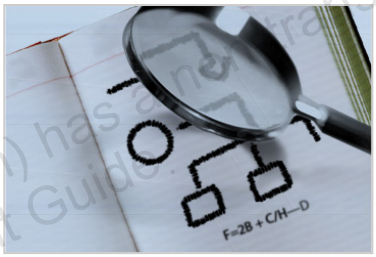
The previous domain only had MedRecSvr1 and MedRecSvr2, this template added another managed server MedRecSvr3, defined clusters, deployed applications and libraries. If you changed the pull-down from Deployment to Service, you would also see other resources added such as JDBC Data Sources and Connection Pools. There is no way to see just what the changes will be, you can only see what the final results will be.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

- Creating a custom domain template
- Supporting other environments
  - JDeveloper
  - SOA
  - WebCenter



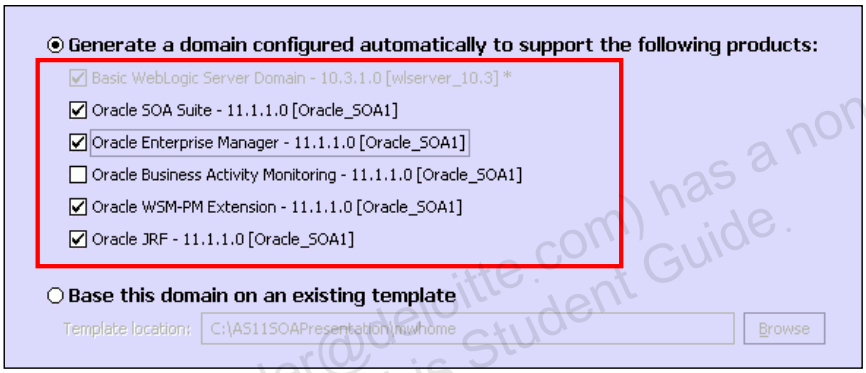
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Templates for SOA, JDeveloper, and Others

- Templates are created by installing the other products.
- The SOA template is located by default in: `<SOA_HOME>common/templates/applications/oracle.soa_template_11.1.1.jar`
- Templates for Oracle Middleware products are automatically detected by the domain wizards.



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Templates for SOA, JDeveloper, and Others

The screenshot shows the option for configuring WebLogic Server for other Fusion Middleware components. Installing those other components will point their templates from their respective directories to the WebLogic Server directory, or you can extend the current WebLogic Server domain by pointing to the JAR files in the SOA directory. In addition, you can copy your own templates into the WebLogic Server template directory at `<WL_HOME>/common/templates`.

SOA and WebCenter components are configured using templates created on WebLogic Server. When you install SOA and WebCenter, the additional WebLogic Server templates are also installed along with the product. You do not need to directly use templates for extending domains for the Oracle function. After you install the Oracle components—for example, SOA, JRF, and JDeveloper—support for those functions is available in the form of a check box on the domain wizard. The primary purpose for templates is to extend a domain with the functionality that you write for your applications (that do not show up as wizard check boxes as shown in the slide).

oracle.soa_template_11.1.1.jar Template

WinZip Pro - oracle.soa_template_11.1.1.jar

File Actions View Jobs Options Help

Name	Path	Size
config-groups.xml		3,808
security.xml		2,061
startscript.xml		3,083
stringsubs.xml		6,219
template-info.xml		702
build.xml	_project_\	2,068
setSOADomainEnv.cmd	bin\	1,003
setSOADomainEnv.sh	bin\	4,972
config.xml	config\	9,704
EDNDataSource-jdbc.xml	config\jdbc\	1,244
EDNLocalTxDataSource-jdbc.xml	config\jdbc\	1,312
mds-soa-jdbc.xml	config\jdbc\	1,299
SOADataSource-jdbc.xml	config\jdbc\	1,415
SOALocalTxDataSource-jdbc.xml	config\jdbc\	1,312
soajmsmodule-jms.xml	config\jms\	3,251
b2b-config.xml	config\soa-infra\configuration\	275
bpel-config.xml	config\soa-infra\configuration\	1,009
businessrules-config.xml	config\soa-infra\configuration\	1,663
cep-config.xml	config\soa-infra\configuration\	439
edn-config.xml	config\soa-infra\configuration\	455
logmetadata.xml	config\soa-infra\configuration\	1,080
mediator-config.xml	config\soa-infra\configuration\	1,969
soa-infra-config.xml	config\soa-infra\configuration\	1,431
workflow-config.xml	config\soa-infra\configuration\	3,476
workflow-identity-config.xml	config\soa-infra\configuration\	416
workflow-notification-config.xml	config\soa-infra\configuration\	808
.dummy	gmds\applications\soa-infra\	0
final.py	script\	11,879

Selected 0 files, 0 bytes | Total 28 files, 67KB

For Ant

WLST and Jython

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carl.mccruder@dell.com) has a non-transferable license to use this Student Guide

oracle.soa_template_11.1.1.jar Template

The `config.xml` (224 lines long, a subset of which is shown on the following page) included in the `.jar` file performs the following actions:

- Deploys several applications:
  - soa-infra
  - worklistapp
  - b2bui
  - FileAdapter
  - DbAdapter
  - JmsAdapter
  - AqAdapter
  - FtpAdapter
  - SocketAdapter
  - MOSeriesAdapter
  - OracleAppsAdapter
  - OracleBamAdapter
- Sets up libraries
  - bpel, worklist, workflow, mediator, uddi
  - ruleseditor
- Sets up file store for JMS named SOAJMSFileStore using JDBC named SOADataSource

`oracle.soa_template_11.1.1.jar` Template (continued)

Even though the XML is validated against a BEA 9.0 schema, this is the current `config.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>
<domain xsi:schemaLocation="http://www.bea.com/ns/weblogic/90/security
http://www.bea.com/ns/weblogic/90/security.xsd
http://www.bea.com/ns/weblogic/90/security/wls
http://www.bea.com/ns/weblogic/90/security/wls.xsd
http://www.bea.com/ns/weblogic/90/security/xacml
http://www.bea.com/ns/weblogic/90/security/xacml.xsd
http://www.bea.com/ns/weblogic/920/domain
http://www.bea.com/ns/weblogic/920/domain.xsd"
   xmlns="http://www.bea.com/ns/weblogic/920/domain"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <name>soainfra</name>
  <server>
      <name>AdminServer</name>
  </server>
  <server>
      <name>soa_server1</name>
      <ssl>
        <name>soa_server1</name>
        <enabled>false</enabled>
        <listen-port>8002</listen-port>
      </ssl>
      <machine>LocalMachine</machine>
      <listen-port>8001</listen-port>
      <listen-address/>
   </server>
   <machine>
     <name>LocalMachine</name>
     <node-manager>
       <name>LocalMachine</name>
       <listen-address>localhost</listen-address>
     </node-manager>
   </machine>
  <admin-server-name>AdminServer</admin-server-name>
  <app-deployment>
    <name>soa-infra</name>
    <module-type>ear</module-type>
    <source-path>$ORACLE_HOME$/soa/applications/soa-infra-wls.ear</source-path>
    <deployment-order>311</deployment-order>
    <security-dd-model>DDOnly</security-dd-model>
    <staging-mode>nostage</staging-mode>
  </app-deployment>
  <app-deployment>
    <name>worklistapp</name>
    <module-type>ear</module-type>
    <source-path>$ORACLE_HOME$/soa/applications/worklist-wls.ear</source-path>
    <deployment-order>312</deployment-order>
    <security-dd-model>DDOnly</security-dd-model>
    <staging-mode>nostage</staging-mode>
  </app-deployment>
  <app-deployment>
:
…snip 100+ lines…
:
</domain>
```

Several of the more interesting names have been highlighted for illustration purposes.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCurdy (carlmccgruder@deloitte.com) has a non-transferable license to use this Student Guide.

WLS Configuration in the Context of Other Products in the Fusion Middleware Suite

- Repository Creation Utility (RCU)
  - Requires the database to be installed first
  - Is used for SOA
- WebTier
  - Can be installed independent of WLS in “opmn-managed mode”
  - Can be installed after WLS with Java Required Files (JRF) enabled
  - Includes Oracle HTTP Server and Oracle Web Cache

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

WLS Installation in the Context of Other Products in the Fusion Middleware Suite

RCU

RCU creates schemas in an existing Oracle database. Install RCU into `rcu_home` by running `./runInstaller` for UNIX or `setup.exe` for Windows. These files are located on Disk1 of the software distribution.

After the installation, to configure RCU, enter:

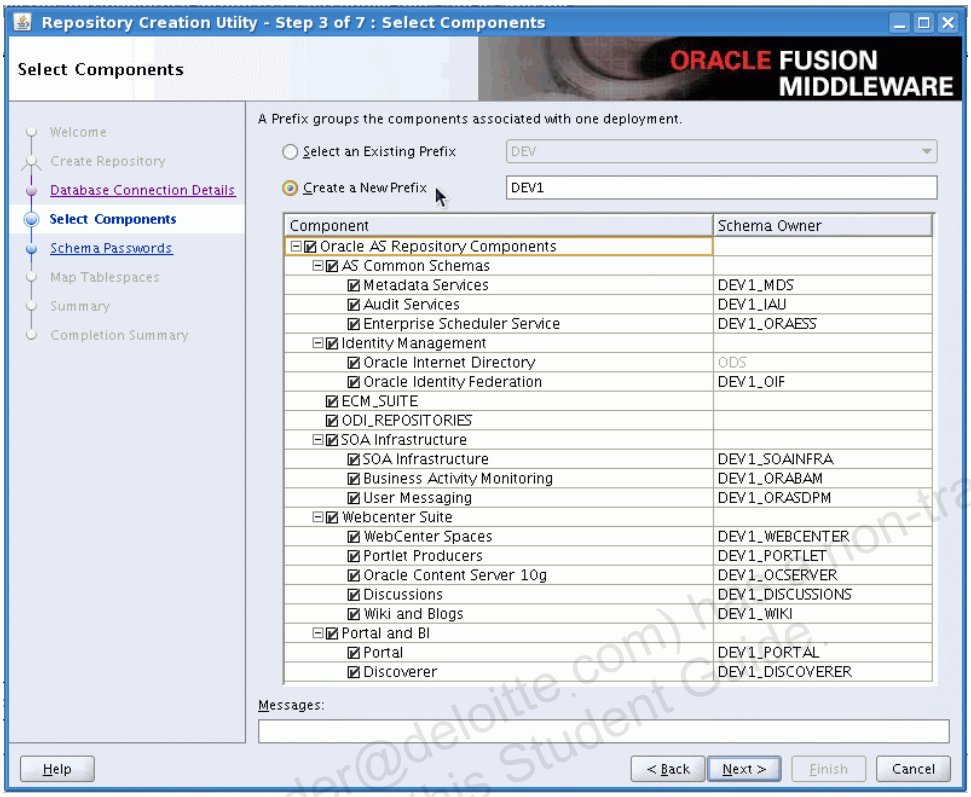
```
cd <ORACLE_BASE>/product/11.1.0/rcuhome/bin
./rcu
```

This allows you to specify the database target, schema names, and their common prefix.

WebTier

- Includes Oracle HTTP Server (OHS) based on Apache Web server. Oracle HTTP Server is covered later in this course in the lesson titled “Deployment Concepts.”
- Includes Web Cache

RCU



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

RCU

You can create schemas in the database for the following components:

- **AS Common Schema:** Metadata Services, Audit Services, and Enterprise Scheduler Service
- **Identity Management:** Oracle Internet Directory and Oracle Identity Federation
- **ECM_Suite**
- **ODI_REPOSITORIES**
- **SOA Infrastructure:** SOA Infrastructure, Business Activity Monitoring, and User Messaging
- **WebCenter Suite:** WebCenter Spaces, Portlet Producers, Oracle Content Server, Discussions, Wiki and Blogs
- **Portal and BI:** Portal and Discoverer

The schemas are prefixed with a prefix of your choice—for example, DEV1_. (The underscore is added by the system; you enter `DEV1`.)

WebLogic Server does not use the RCU directly.

SOA Installation

- Before proceeding with the SOA installation:
  - Install the database
  - Create DB schemas for the suites to install using RCU
  - Install and configure a WebLogic Server
- Perform the SOA installation.
- After the SOA installation, perform one or both of the following tasks:
  - Extend the existing WebLogic domain to support SOA functions.
  - Create a new WebLogic domain to support SOA functions.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

SOA Installation

From the SOA Installer:

Welcome to Oracle Fusion Middleware 11*g* SOA Suite Installer.

Before proceeding, create schemas for the suites you want to install. For more information, refer to the chapter titled "Repository Creation Utility (RCU)" in the *Oracle Fusion Middleware Enterprise Installation Guide*.

Before proceeding, install and configure a WebLogic server.

After all necessary schemas have been created and a WebLogic server is installed and configured, click Next to begin the installation.

Quiz

Invoke the Template Configuration Wizard using _____.

a. `config.sh` under `<WL_HOME>/common/bin`

b. `config_builder.sh` under `<WL_HOME>/common/bin`

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

`config_builder.sh` is used to invoke the Domain Template Wizard.

Quiz

When you create a new domain to automatically support Oracle WebLogic Server, which template is it based on by default?

a. `wlst.jar`
b. `ws.jar`
c. `web_server.jar`
d. `server.jar`
e. `wls.jar`

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: e

If you create a new domain and if you choose to have the domain configured automatically to support the WebLogic Server option, the domain is based on the default `wls.jar` template that is found in the `<WL_HOME>/common/templates/domains` directory.

Quiz

Which of the following can you use to create a domain template?

a. The `pack` command

b. `config_temp.sh`

c. The Administration Console

d. `config_builder.sh`

e. The WLST offline command-line tool

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a, d, e

You can create domain templates using the following:

- The WLST offline command-line tool
- The `pack` command
- The Domain Template Builder, also known as `config_builder.sh`

Summary

In this lesson, you should have learned how to:

- Create custom domain templates
- Create a new domain using the custom domain template for JDeveloper, SOA, and WebCenter

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 6 Overview: Using a Domain Template

This practice covers the following topics:

- Examining an existing template
- Extending an existing domain using a supplied application template

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 6 Overview: Using a Domain Template

See Appendix A for the complete steps to do the practice.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

7

Using Administration Console and WLST

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Start the WebLogic Server Administration Console
- Explain the layout and navigation of the Administration Console
- Access context-sensitive help within the Administration Console
- Customize Administration Console preferences
- Use the Administration Console breadcrumb trail
- Create, commit, undo, and monitor a change session using the Administration Console
- Access advanced resource attributes in the Administration Console

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

Although the command line is useful for repetitive tasks that lend themselves to scripts, the Web-based Administration Console is faster for one-off tasks. Because you will use these tools frequently, you need to customize them to show only the tasks and columns that you use most often and suppress the columns that are not applicable to your environment. You will need to identify the tools that are available and choose which one is appropriate based on the tasks and functionalities performed by each task.

Objectives

After completing this lesson, you should be able to:

- Identify dynamic and nondynamic attribute changes in the Administration Console
- Customize monitoring tables within the Administration Console
- Use the WebLogic Scripting Tool (WLST) command-line utility
- Invoke WLST and navigate through the domain structure
- Describe how change management is performed internally using Java Management Extension (JMX) and MBeans

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

- Performing Administration Console configuration
  - Navigation
  - Help
  - Setting preferences and properties
- Performing command-line configuration

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Benefits of Using the Administration Console

By using the Administration Console, you can:

- Configure attributes of servers and their resources
- Deploy and secure applications
- Configure, collect, and view diagnostic information
- Start and shut down servers or perform other management actions

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Benefits of Using the Administration Console

The Oracle WebLogic Server Administration Console is a Web browser–based graphical user interface that you use to manage an Oracle WebLogic Server domain. An Oracle WebLogic Server domain is a logically related group of Oracle WebLogic Server resources that you manage as a unit. A domain includes one or more Oracle WebLogic Servers and may also include Oracle WebLogic Server clusters.

Use the Administration Console to:

- Configure, start, and stop Oracle WebLogic Server instances
- Configure Oracle WebLogic Server clusters
- Configure Oracle WebLogic Server services, such as database connectivity and messaging
- Configure security parameters, including managing users, groups, and roles
- Configure and deploy your applications
- Monitor server and application performance
- View server and domain log files
- Edit selected run-time application deployment descriptor elements

Accessing the Administration Console

After starting the administration server, you can access the Administration Console in a Web browser of your choice.

```
http://[hostname]:[port]/console
https://[hostname]:[secureport]/console
```

hostname **= The name or IP address of the Administration Server**
port **= The port number that the Administration Server listens on**
secureport **= The SSL port number that the Administration Server listens on**

```
http://localhost:7001/console
http://adminDNSName:7001/console
https://127.0.0.1:7002/console
```

SSL on a different port

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Accessing the Administration Console

When started, the Administration Console prompts for a password. The first time the Administration Console is started, you can use the username and password with which the administration server was started.

You can use the Administration Console to create a list of users with administration privileges. When designated, these users can also perform administrative tasks via the Administration Console.

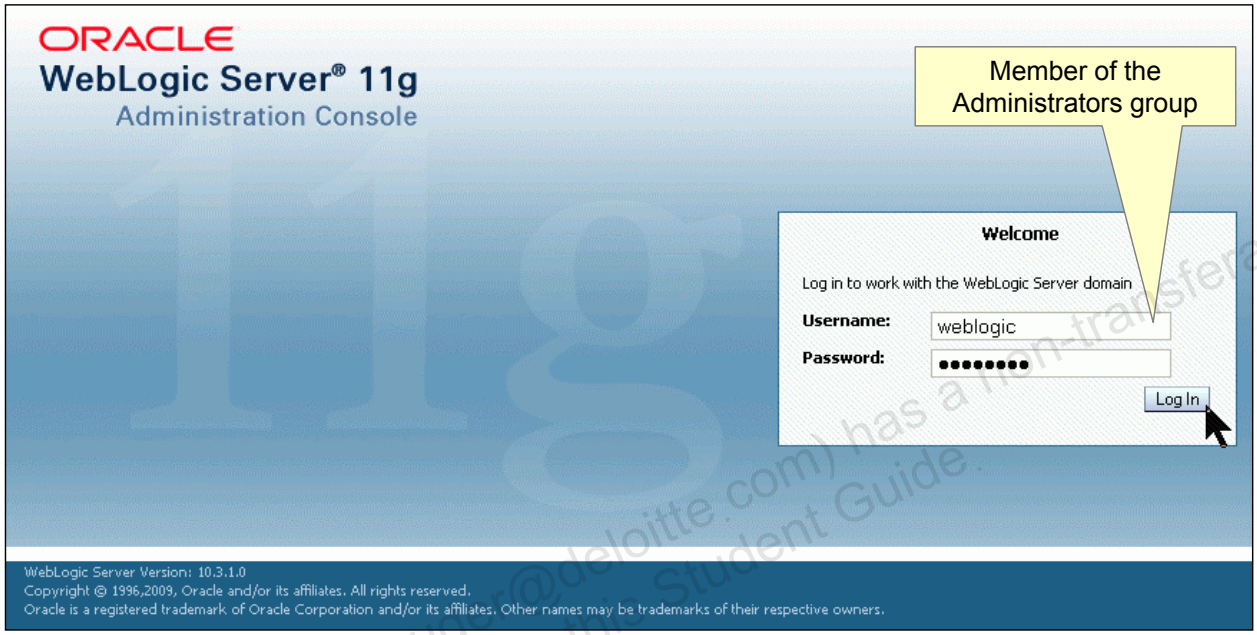
If you configured a domainwide administration port, use that port number. If you configured the administration server to use secure sockets layer (SSL), you must add “s” after “http.” A domainwide administration port always uses SSL.

If you have your browser configured to send HTTP requests to a proxy server, you may want to configure your browser so that it does not send administration server requests through the proxy. For example, if the administration server is on the same machine as the browser, you may want to ensure that requests sent to localhost or 127.0.0.1, or both, are not sent to the proxy.

The first time you access the console on the administration server, there may be a brief delay because the application has to be loaded and initialized.

Administration Console Login

Enter the username and password that you set when creating your domain. The password is not displayed.



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (ccmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Administration Console Login

This screenshot shows the login page. When the login page appears, enter the username and password that you used to start the administration server (you may have specified this username and password during the installation process) or enter a username that belongs to one of the following security groups:

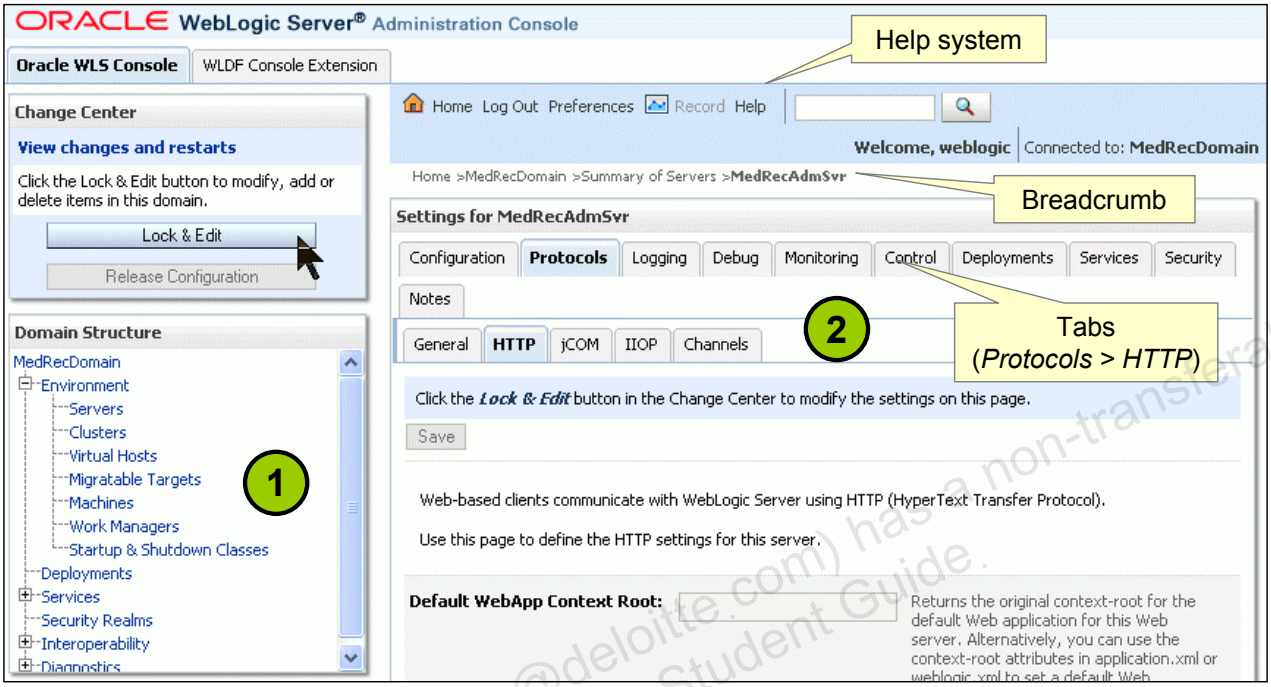
- Administrators
- Operators
- Deployers
- Monitors

These groups provide various levels of access to the system administration functions in the Administration Console. Using the security system, you can add users to or delete users from one of these groups to provide controlled access to the console.

By default, each time the administration server starts, it automatically deploys the Administration Console. If you want to prevent access to the Administration Console (for example, as an added security measure in a production environment), you can prevent the administration server from deploying it:

1. Select your domain name in the Domain Structure panel of the console.
2. Select Configuration > General, and click Advanced at the bottom of the page.
3. Deselect Console Enabled and click Save.

Basic Navigation



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

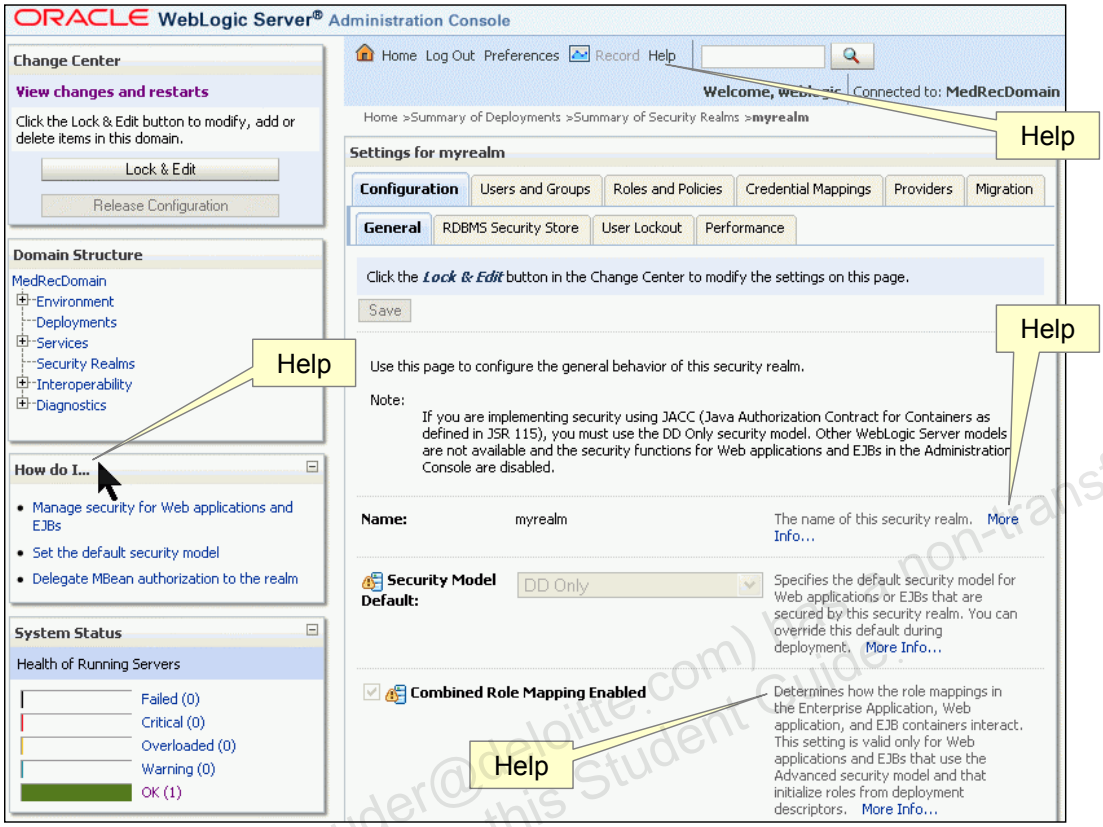
Basic Navigation

This screenshot shows a typical Administration Console page illustrating common navigation elements. The left panel (1) is a tree that you can use to navigate to pages in the Administration Console. Click any of the nodes in the Domain Structure tree to go to that page. Click the "+" (plus) symbol in Domain Structure to expand a node and the "–" (minus) symbol to collapse the node. The right panel of the Administration Console (2) shows detailed configuration information about the selection in the left panel. The Administration Console includes a complete help system described in the next slide. There are two lines of tabs shown (not three). The Notes tab in this example is wrapped from the top line.

The console includes a "breadcrumb" or "locator link" navigation feature, which presents a series of links that show the path you have taken (menu history) through the Administration Console's pages. You can click any of the links to return to a previously visited page.

The WebLogic Diagnostics Framework (WLDF) tab is a monitoring and diagnostic framework that defines and implements a set of services that run within the Oracle WebLogic Server process and participate in the standard server life cycle. Using WLDF, you can create, collect, analyze, archive, and access diagnostic data generated by a running server and the applications deployed within its containers. This data provides insight into the run-time performance of servers and applications and enables you to isolate and diagnose faults when they occur. WLDF is not used directly in this course.

Using the Help System



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Using the Help System

There are many opportunities to get help both online and offline. The screenshot shows four major help facilities:

- How do I…
- Help (documentation)
- Prompt help
- More Info…

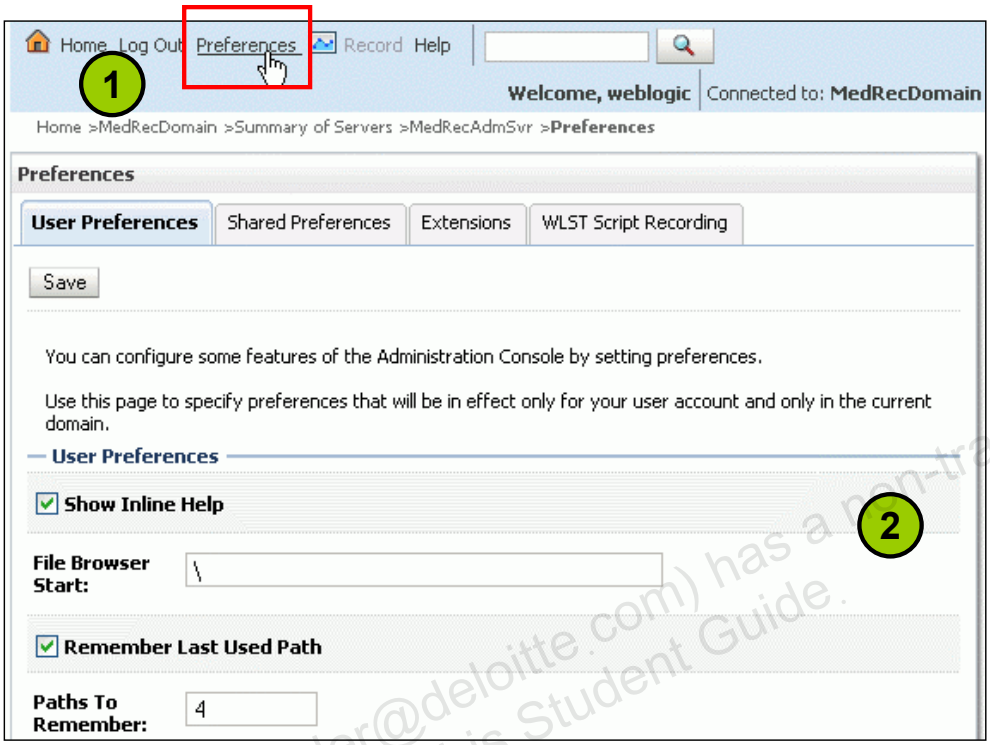
The "How do I" changes as you navigate through the menus to show the most frequently asked questions for the particular screen that you are using at the moment. The Help on the top menu bar opens a separate browser window displaying the same documentation that is available online from Oracle support. The text to the right of the prompts shows a brief explanation of the field and what is expected. Those paragraphs can be turned on and off globally in the Preferences section. Each of those paragraphs may have additional information that can be displayed in a separate window by clicking More Info.

The entire documentation library or portions of it can be downloaded from Oracle Technology Network (OTN) in either PDF or HTML format.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

General Administration Console User Preferences



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl MCGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

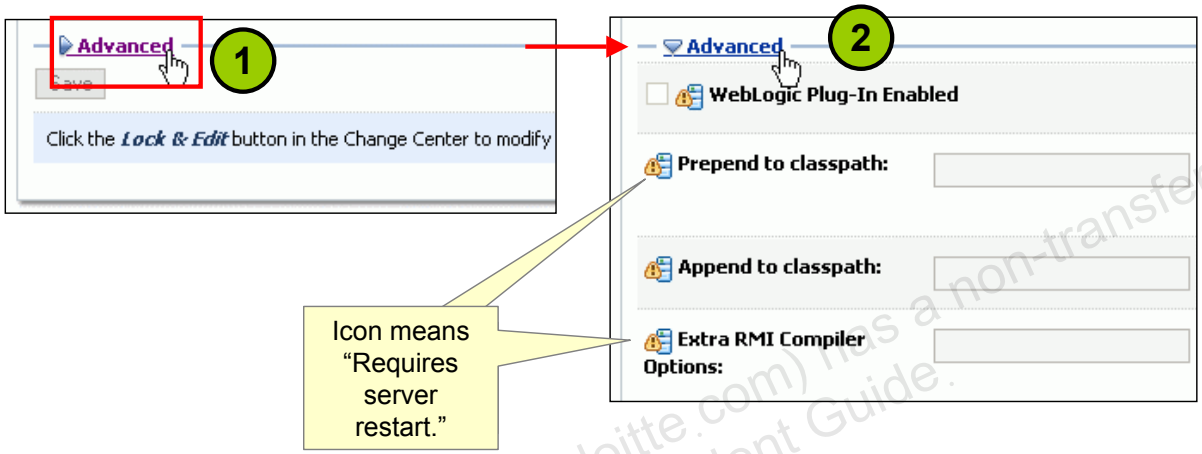
General Administration Console User Preferences

This screenshot shows how to set the Administration Console User Preferences.

1. The upper-right pane in the Administration Console contains a Preferences option, which can be selected to change console preferences.
2. Change the value for one or more preferences and click Save. The preferences that are available include:
   - **Show Inline Help:** Determines whether inline help appears for forms
   - **File Browser Start:** Sets the directory that the deployment file browser starts in
   - **Remember Last Used Path:** Causes the deployment file browser to remember the last path browsed to before selecting an application
   - **Warn If User Holds Lock:** Causes a warning message to be issued when the user logs out, reminding the user that he or she is currently the owner of the domain configuration lock
   - **Perform Asynchronous Activation:** Causes changes to be activated asynchronously when the user clicks the **Activate** button
   - **Show Advanced Sections:** Causes advanced form sections to be displayed by default
   - **Warn User Before Taking Lock:** Causes a warning message to be issued when the user attempts to take the domain lock, reminding the user that another user is currently the owner of the domain configuration lock

Advanced Console Options

In a screen's Advanced section, the console shows or hides the options that are not frequently used.



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

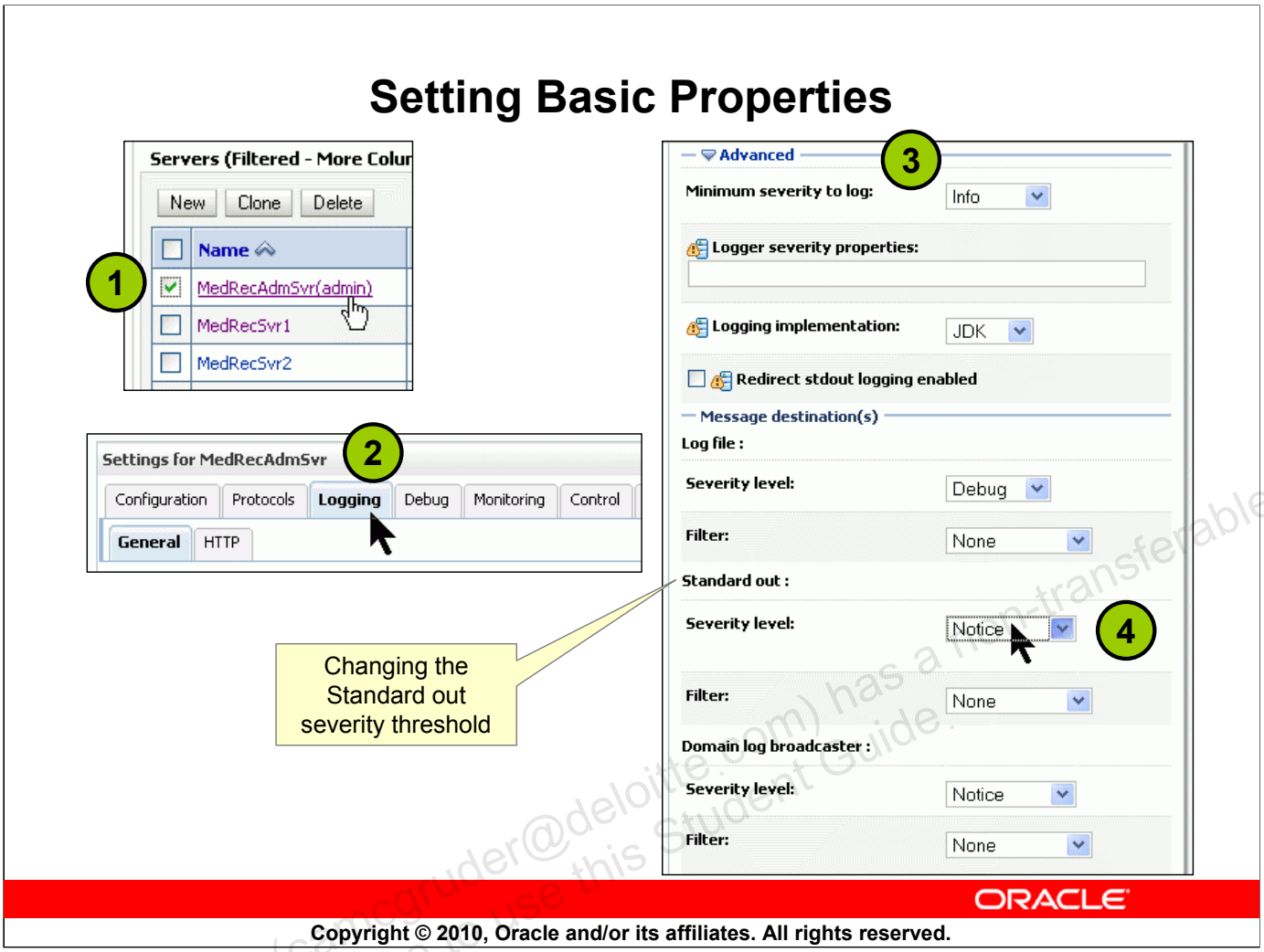
Advanced Console Options

Many pages have an Advanced section at the bottom. This screenshot shows toggling between Advanced hiding and showing the additional options for a given page. Note the triangle or arrowhead pointing to the right (Advanced options are hidden) and pointing down (Advanced options are exposed). By default, only the most commonly changed configuration attributes are shown for a given resource, such as a server, cluster, data source, application, or security provider. To see the full list of available attributes for a page, click the Advanced link at the bottom of the page, if applicable.

Some of the changes require the server to be shut down and restarted as indicated by a Warning icon. That restart can be deferred until a convenient time—for example, during a maintenance window at midnight. A recap of what needs to be restarted can be viewed at any time in the Change Center described in a few slides from now in this lesson.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.



Setting Basic Properties

This screenshot is an example of how to set some basic properties using the Oracle WebLogic Server Administration Console. In this example, you see how to set the amount of logging information Oracle WebLogic Server displays on the command line.

To change the logging severity threshold, perform the following steps in order:

1. Select Servers from the tree view at the left of the console, and find the server that you want to update.
2. Click the Logging tab at the right of the console.
3. Scroll down and expand the Advanced section.
4. Select the severity that you want from the list. This determines how much logging information is output to the command shell.

You can also change the File Name attribute.

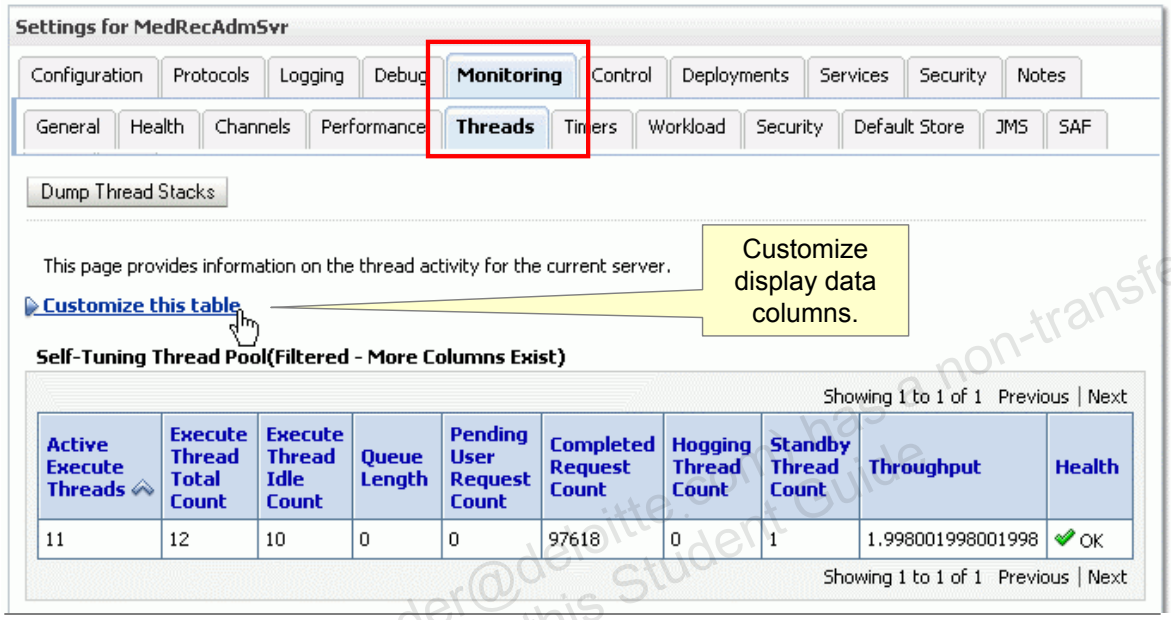
File Name: This is the name of the file that is used to write the log messages to disk. The file name may contain up to 256 alphanumeric characters. The default value is `servername.log`. The server must be restarted if you change the log file name. This particular example is covered in more detail in the lesson titled "Viewing and Managing Logs in Oracle WLS Environment."

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Administration Console Monitoring

The Administration Console offers many monitoring capabilities for servers, application deployments, and Java EE services.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Administration Console Monitoring

This screenshot shows some of the monitoring statistics available with the WebLogic Server. Every time a service or application object can be monitored, a Monitoring tab is available in the console for that object. When you click this tab, it displays the available monitoring information for the selected object.

Moreover, when the monitoring page shows information in a tabular format, you can change the way the information is displayed. To do this, click “Customize this table” and choose which columns to display and on what columns to sort the table. Some table views give you the option to filter the data.

However, you cannot monitor the activity of one domain through another domain. For example, you cannot open the Administration Console for domainY and try to monitor servers within domainZ.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

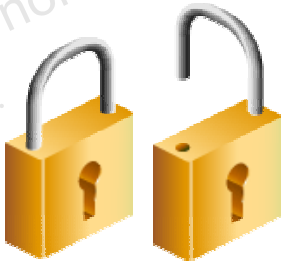
Carl McCrudder (ccmccrudder@deloitte.com) has a non-transferable license to use this Student Guide

Configuration Change Management

For change management, use the Change Center in the WLS Administration Console to lock the configuration files.

The change management features of WLS:

- Enable you to distribute configuration changes throughout a domain securely, consistently, and predictably
- Are the same, regardless of whether you are using:
  - The WLS Administration Console
  - The WebLogic Scripting Tool (WLST)
  - The JMX APIs



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuration Change Management

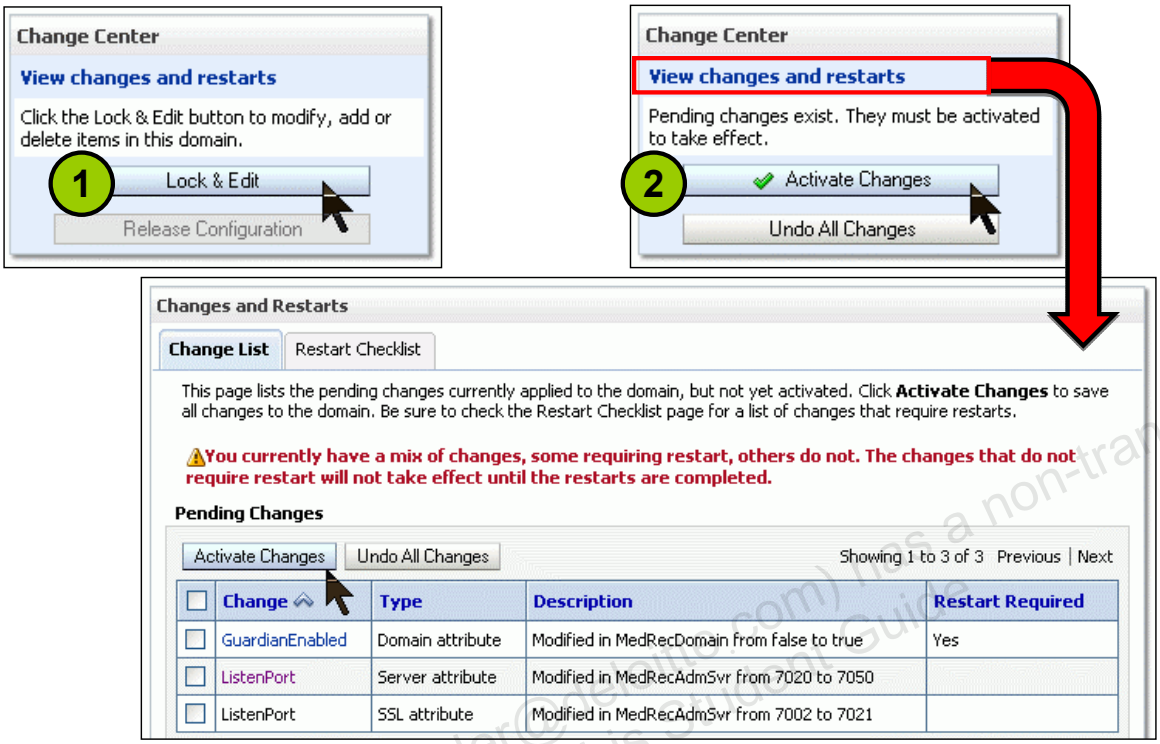
Because the Oracle WebLogic Server management system is based on Java EE and other standards, it integrates with systems that are frequently used to manage other software and hardware components. In addition, Oracle WebLogic Server includes several of its own standards-based, extensible utilities. Alternatively, you can use APIs to create custom management utilities.

JMX is the Java EE solution for monitoring and managing resources on a network. Like Simple Network Management Protocol (SNMP) and other management standards, JMX is a public specification, and many vendors of commonly used monitoring products support it.

The Administration Console, WLST, and other Oracle WebLogic Server utilities use JMX APIs. However, system administrators can easily perform all Oracle WebLogic Server management tasks without having to learn the JMX API or the underlying management architecture.

The Administration Server enables you to change the configuration attributes of domain resources dynamically—that is, while the Oracle WebLogic Servers are running. For many attributes, you need not restart the servers for your change to take effect. Therefore, a change in configuration is reflected in both the current run-time value of the attribute as well as the persistently stored value in the configuration file.

Configuration Change Management Using the Administration Console Change Center



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@yahoo.com) has a non-transferable license to use this Student Guide.

Configuration Change Management Using the Administration Console Change Center

This screenshot shows a list of pending configuration changes. The starting point for using the Administration Console to make changes in your Oracle WebLogic Server domain is the Change Center. To change a domain's configuration, you must:

1. Locate the Change Center at the upper left of the Administration Console screen. Click the Lock & Edit button to lock the configuration edit hierarchy for the domain. Make the changes that you desire on the relevant page of the console. Click Save on each page where you make a change.
2. When you finish making the desired changes, click Activate Changes in the Change Center.

You can undo any pending (saved, but not yet activated) changes by clicking Undo All Changes in the Change Center. Stopping the administration server does not release the configuration lock. When the administration server starts again, the configuration lock is in the same state it was in when the administration server was shut down, and any pending changes are preserved.

You can view any changes that you have saved, but not yet activated, by clicking the View Changes and Restarts link in the Change Center. The Change List tab presents all the changes that have been saved, but not yet been activated. Restart Checklist lists all the servers for which nondynamic changes have been activated, but which require restarts before the changes become effective.

Domain Configuration Repository

Directory	Description
config	Root folder of domain configuration files
configCache	Cached configuration data
deployments	Staging area for deployment applications
diagnostics	Configuration modules for diagnostics framework
jdbc	Configuration modules for data services
jms	Configuration modules for messaging services
lib	Not currently used. See `<domain>/lib`.
nodemanager	Node Manager configuration files
security	Configuration modules for the security framework
startup	Any scripts to run as part of server startup
config.lok	Lock file used for change management
config.xml	Primary domain configuration file

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudden (carlmccrudden@yahoo.com) has a non-transferable license to use this Student Guide.

Domain Configuration Repository

Each domain describes its configuration in an XML document that is located in the domain's configuration directory. At run time, each Oracle WebLogic Server instance in a given domain creates an in-memory representation of the configuration.

The central configuration file for a domain is *DOMAIN*`/config/config.xml`. This file specifies the name of the domain and the configuration of each server instance, cluster, resource, and service in the domain. The file includes references to additional XML files that are stored in subdirectories of the *DOMAIN*`/config` directory. These included files are used to describe major subsystems of Oracle WebLogic Server. Also note that the security credentials for domain security and the embedded LDAP server are stored in the `config.xml` file in encrypted form.

Each managed server maintains a copy of the domain's configuration files. This copy is read-only and can be updated only as part of a change management process.

You can configure Oracle WebLogic Server to make backup copies of the configuration files. This facilitates recovery in cases where configuration changes need to be reversed or the unlikely case that configuration files become corrupted.

The configCache folder contains data that is used to optimize performance when validating changes in the domain's configuration documents. This data is internal to Oracle WebLogic Server and does not need to be backed up.

The *DOMAIN*`/lib` directory contains any JAR files that should be added to the system `CLASSPATH` of each server instance in the domain when the server's Java Virtual Machine (JVM) starts.

Configuration Change Process

- Domain configuration is represented in two ways:
  - On the file system by a set of XML configuration files, including `config.xml`
  - At run time by a hierarchy of in-memory JMX objects
- When you activate changes, it is a two-phase process:
  - Each server determines whether it can accept the change.
  - If all servers are able to accept the change, they update their working configuration hierarchy and the change is completed. Otherwise, the pending changes are rolled back and nothing happens.

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

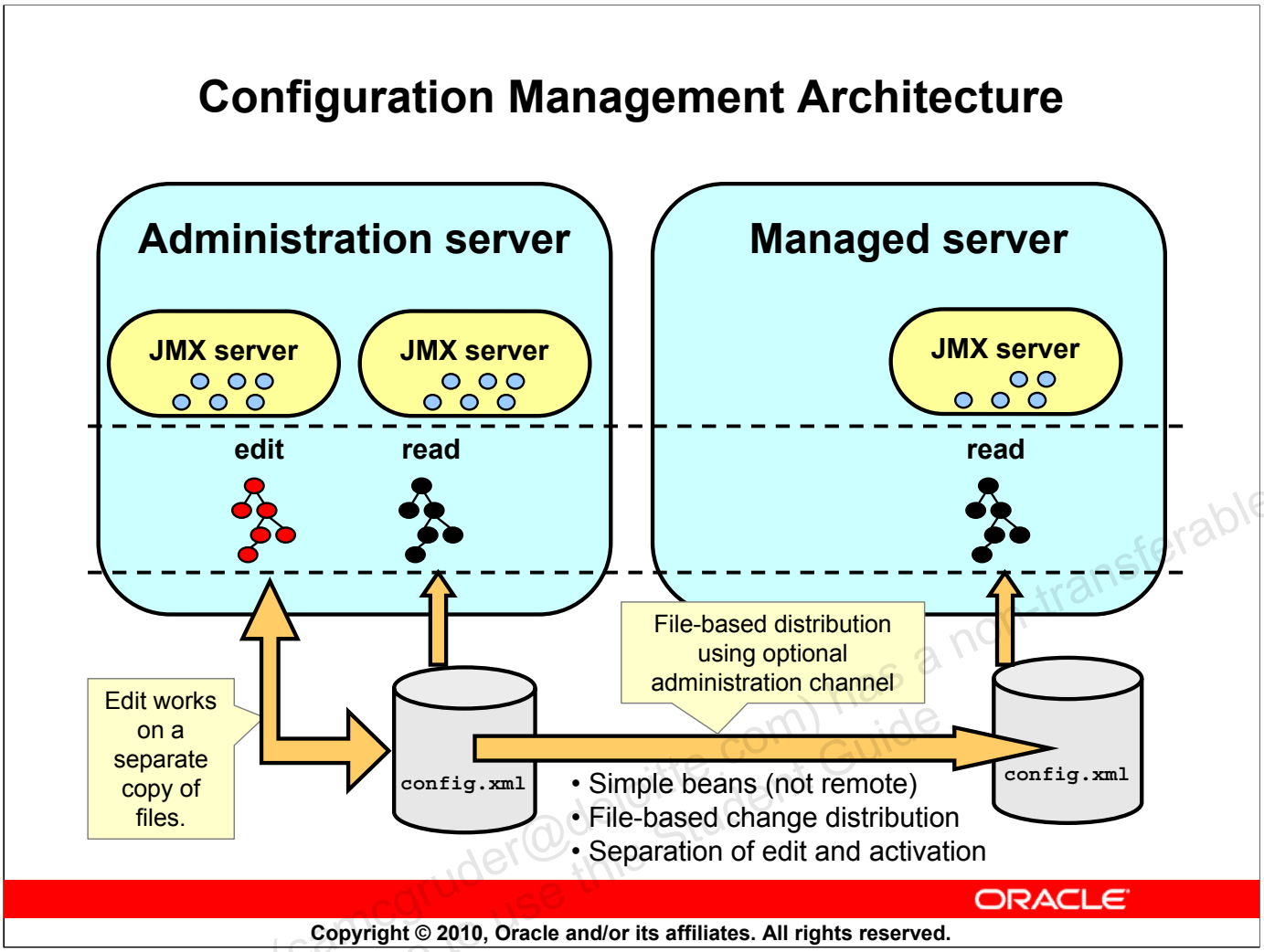
Carl MCGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuration Change Process

To provide a secure, predictable means for distributing configuration changes in a domain, Oracle WebLogic Server imposes a change management process that loosely resembles a database transaction.

The configuration of a domain is represented on the file system by a set of XML configuration files, centralized in the `config.xml` file, and at run time by a hierarchy of JMX Configuration Managed Beans (MBeans). When you edit the domain configuration, you edit a separate hierarchy of Configuration MBeans that reside on the administration server.

The configuration change management process of Oracle WebLogic Server loosely resembles a database transaction. To start the edit process, you first obtain a lock on the edit hierarchy to prevent other people from making changes. When you finish making changes, you save the changes to the edit hierarchy. For the changes to take effect, you need to distribute them to one or more servers in the domain, and then activate them. When you activate the changes, each server determines whether or not it can accept the changes. If all the servers are able to accept the changes, they update their working configuration hierarchy. If any server cannot accept a change, all changes are rolled back from all the servers in the domain. The changes are left in a pending state; you can then either edit the pending changes to resolve the problem or revert to the original configuration.



Configuration Management Architecture

This graphic depicts MBeans reading and writing to the `config.xml` files. Each domain describes its configuration in an XML document that is located in the domain's configuration directory. At run time, each Oracle WebLogic Server instance in a given domain creates an in-memory representation of the configuration described in this document. The in-memory representation of a domain's configuration is a collection of read-only managed beans (MBeans) called Configuration MBeans.

In addition to the read-only Configuration MBeans, the administration server maintains another collection of Configuration MBeans that you can edit. To edit these Configuration MBeans, you use a JMX client (either the Administration Console, WLST, or a client that you create) to obtain a lock.

While you have the lock on the editable Configuration MBeans, you can save your in-memory changes, which causes the administration server to write the changes to a set of pending configuration documents in the domain directory. The Oracle WebLogic Server instances do not consume (commit) the changes until you activate them.

When you activate the changes, each server in the domain determines whether it can accept the change. If all the servers can accept the change, they update their copy of the domain's configuration document. Then they update their working copy of the Configuration MBeans and the change is completed.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmcgruder@yahoo.com) has a non-transferable license to use this Student Guide

Configuration Management Architecture (continued)

Note that Oracle WebLogic Server's change management process applies to changes in the domain and server configuration data, and not to security or application data.

Some configuration changes can take effect without disturbing normal operations, whereas others require the affected servers to be restarted before they take effect. The configuration changes that can take effect without a server restart are sometimes referred to as dynamic changes, whereas those that require a server restart are sometimes referred to as nondynamic changes.

Edits to dynamic configuration attributes become available after they are activated, without restarting the affected server or system resource. However, edits to nondynamic configuration attributes require that the affected servers or system resources be restarted before they become effective.

If an edit is made to a nondynamic configuration setting, no edits to the dynamic configuration settings will take effect until after restart. This is to ensure that a batch of updates with a combination of dynamic and nondynamic attribute edits are not partially activated.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

XML Schema for `config.xml`

- The `config.xml` file adheres to an XML schema that can be used for validation.
- The `config.xml` file aggregates configuration information from other configuration files representing Oracle WebLogic Server subsystems, which adhere to their own XML schemas.
- The `config.xml` file is located (by default) in the `user_projects/domains/domain_name/config` directory.
- Subsidiary configuration files are located in subdirectories.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

XML Schema for `config.xml`

The following is an extract from the `config.xml` schema definition. Notice that most of them have a minimum occurrence (`minOccurs`) of zero, meaning that most of these are optional and have reasonable defaults. Because of that, the `config.xml` for a simple domain can be fairly small. A sample of the schema follows:

```
<xs:element name="replication-group" type="xs:string" minOccurs="0"
   nillable="true"/>
<xs:element name="preferred-secondary-group" type="xs:string"
   minOccurs="0" nillable="true"/>
<xs:element name="auto-migration-enabled" type="xs:boolean"
   minOccurs="0" nillable="false"/>
<xs:element name="web-server" type="dom:web-serverType" minOccurs="0"
   nillable="true"/>
<xs:element name="jdbc-logging-enabled" type="xs:boolean" minOccurs="0"
   nillable="false"/>
<xs:element name="j2ee12-only-mode-enabled" type="xs:boolean"
   minOccurs="0" nillable="false"/>
<xs:element name="j2ee13-warning-enabled" type="xs:boolean"
   minOccurs="0" nillable="false"/>
<xs:element name="iiop-enabled" type="xs:boolean" minOccurs="0"
   nillable="false"/>
```

XML Schema for `config.xml` (continued)

```
<xs:element name="default-iiop-user" type="xs:string" minOccurs="0"
    nillable="true"/>
<xs:element name="default-iiop-password-encrypted" type="xs:string"
    minOccurs="0" nillable="true"/>
<xs:element name="tgiop-enabled" type="xs:boolean" minOccurs="0"
    nillable="false"/>
<xs:element name="default-tgiop-user" type="xs:string" minOccurs="0"
    nillable="true"/>
<xs:element name="default-tgiop-password-encrypted" type="xs:string"
    minOccurs="0" nillable="true"/>
<xs:element name="com-enabled" type="xs:boolean" minOccurs="0"
    nillable="false"/>
<xs:element name="jrmp-enabled" type="xs:boolean" minOccurs="0"
    nillable="false"/>
<xs:element name="com" type="dom:comType" minOccurs="0" nillable="true"/>
<xs:element name="server-debug" type="dom:server-debugType" minOccurs="0"
    nillable="true"/>
<xs:element name="httpd-enabled" type="xs:boolean" minOccurs="0"
    nillable="false"/>
<xs:element name="system-password-encrypted" type="xs:string"
    minOccurs="0" nillable="true"/>
<xs:element name="console-input-enabled" type="xs:boolean" minOccurs="0"
    nillable="false"/>
<xs:element name="listen-thread-start-delay-secs" type="xs:int"
    minOccurs="0" nillable="false"/>
<xs:element name="listeners-bind-early" type="xs:boolean" minOccurs="0"
    nillable="false"/>
<xs:element name="listen-address" type="xs:string" minOccurs="0"
    nillable="true"/>
<xs:element name="external-dns-name" type="xs:string" minOccurs="0"
    nillable="true"/>
<xs:element name="interface-address" type="xs:string" minOccurs="0"
    nillable="true"/>
```

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

- Performing Administration Console configuration
- Performing command-line configuration
  - WLST
  - Jython

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

WebLogic Scripting Tool (WLST)

- The WLS command-line tools are useful:
  - For automating common administration activities
  - As an alternative to the Administration Console
  - When graphical tools are not supported
- WLST provides a command-line interface for:
  - Creating new WLS domains
  - Retrieving and updating WLS domain configurations
  - Deploying applications
  - Obtaining run-time server statistics

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

WebLogic Scripting Tool

The WLST is a command-line scripting environment that you can use to create, manage, and monitor Oracle WebLogic Server domains. It is based on the Java scripting interpreter, Jython. In addition to supporting standard Jython features such as local variables, conditional variables, and flow control statements, WLST provides a set of scripting functions (commands) that are specific to Oracle WebLogic Server. You can extend the WebLogic scripting language to suit your needs by following the Jython language syntax.

Jython

Jython is a Java implementation of the popular Python scripting language:

- Simple and clear syntax
- Indentation to structure code (white space critical!)
- Interactive command mode
- Custom commands
- Integration with any existing Java libraries

```
list = ['ab','cd','ef']

if len(list) >= 3:
   for x in list:
      print x, len(x)
   print 'done'
```

```
from java.util import ArrayList

list = ArrayList()
list.add('ab')
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@oracle.com) has a non-transferable license to use this Student Guide.

Jython

The latest Jython release (2.2) implements the same language as CPython 2.2 and many of the CPython standard library modules.

Jython is an implementation of the high-level, dynamic, object-oriented language Python, seamlessly integrated with the Java platform. Jython is freely available for both commercial and noncommercial use, and is distributed with source code. Jython is complementary to Java and is especially suited for the following tasks:

- **Embedded scripting:** Java programmers can add Jython libraries to their system to allow end users to write simple or complicated scripts that add functionality to the application.
- **Interactive experimentation:** Jython provides an interactive interpreter that can be used to interact with Java packages or with running Java applications. This allows programmers to experiment and debug any Java system using Jython.
- **Rapid application development:** Python programs are typically two to ten times shorter than the equivalent Java program. This translates directly to increased programmer productivity. The seamless interaction between Python and Java enables developers to freely mix the two languages both during development and in shipping products.

Python (and therefore, Jython) uses leading white space to format structures such as conditions, loops, and functions, instead of braces ("{") like in Java.

Using Jython

Jython can interpret commands in three ways:

Interactive: Supply commands one at a time from a command prompt. Enter a command in the WLST console and view the response immediately.

Batch: Provide a series of commands in a script file (`.py`) when you create a text file with the `.py` extension that contains a series of WLST commands.

Embedded: Run the Jython interpreter within a Java class when you instantiate an instance of the WLST interpreter in your Java code and use it to run WLST commands.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using Jython

Interactive mode, in which you enter a command and view the response at a command-line prompt, is useful for learning the tool, prototyping the command syntax, and verifying configuration options before building a script. Using WLST interactively is particularly useful to get immediate feedback after making a critical configuration change.

The WLST scripting shell maintains a persistent connection with an instance of Oracle WebLogic Server. WLST can also write all the commands that you enter during a WLST session to a file. You can edit this file and run it as a WLST script.

Scripts invoke a sequence of WLST commands without requiring your input, much like a shell script. Scripts contain WLST commands in a text file that by convention ends with a `.py` file extension—for example, `myscriptfile.py`. Several sample scripts can be found in the WLST documentation online and at `<WL_HOME>\ samples\server`.

In embedded mode, you instantiate the WLST interpreter in your Java code and use it to run WLST commands and scripts. All WLST commands and variables that you use in the interactive and batch modes can be run in embedded mode. Refer to the Java class `weblogic.management.scripting.utils.WLSTInterpreter`. (This is a wrapper class to the Jython interpreter. It adds WebLogic-specific extensions that enable you to run WLST in embedded mode from a Java client.)

WLST Modes

- Online mode:
  - Connected to a running server
  - Access to all WLS configuration and run-time attributes
  - To create and activate change sessions similar to the WLS console
- Offline mode:
  - Domain not running
  - Access to only persisted domain configuration (`config.xml`)
  - To create or update domains similar to using the Configuration Wizard

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

WLST Modes

You can use WLST as the command-line equivalent to the Oracle WebLogic Server Administration Console (WLST online) or as the command-line equivalent to the Configuration Wizard (WLST offline).

You can use WLST to connect to a running administration server and manage the configuration of an active domain, view performance data about the resources in the domain, or manage security data (such as adding or removing users). You can also use WLST to connect to managed servers, but you cannot modify configuration data from managed servers. WLST online is a JMX client. It interacts with a server's in-memory collection of MBeans, which are Java objects that provide a management interface for an underlying resource.

Without connecting to a running Oracle WebLogic Server instance, you can use WLST to create domain templates, create a new domain based on existing templates, or extend an existing, inactive domain. However, you cannot use WLST offline to view performance data about the resources in a domain or modify security data (such as adding or removing users). WLST offline provides read and write access to the configuration data that is persisted in the domain's `config` directory or in a domain template JAR that is created using the Domain Template Builder.

WLST Example

```
[oracle@edvmr1p0 /]$ java weblogic.WLST

Initializing WebLogic Scripting Tool (WLST) ...
Welcome to WebLogic Server Administration Scripting Shell
Type help() for help on available commands

wls:/offline> connect('weblogic','password','t3://localhost:7001')
Connecting to t3://localhost:7001 with userid system ...
Successfully connected to Admin Server 'myAdmin' that belongs to domain 'mydomain'.

Warning: An insecure protocol was used to connect to the server. To
ensure on-the-wire security, the SSL port or Admin port should be used instead.

wls:/mydomain/serverConfig> cd('Servers')
wls:/mydomain/serverConfig/Servers> ls()
dr--   myAdmin
dr--   myserver1
dr--   myserver2

wls:/mydomain/serverConfig/Servers> cd('myserver1')
wls:/mydomain/serverConfig/Servers/myserver1> get('StartupMode')
'RUNNING'
wls:/mydomain/serverConfig/Servers/myserver1> exit()

Exiting WebLogic Scripting Tool.

[oracle@edvmr1p0 /]$
```

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

WLST Example

In this example, the administrator:

1. Starts WLST (case-sensitive)
2. Connects to the administration server “myAdmin”
3. Changes to the Servers MBean directory
4. Checks the directory to see which servers exist
5. Changes to the directory of the myserver1 MBean
6. Checks the value of the `StartupMode` attribute for the server “myserver1”
7. Exits WLST

WLST Command Requirements

- Use case-sensitive names and arguments of commands.
- Use arguments enclosed within single or double quotation marks.
- Precede the quoted string with **r** while specifying the backslash (\) in the string.
  - Example: `readTemplate(r'c:\mytemplate.jar')`
- Do not use the following invalid characters in object names when you use WLST offline:
  - Period (.)
  - Slash (/)
  - Backslash (\)
- Use the display help:
  - Example: **`wls:/mydomain/serverConfig> help('disconnect')`**

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

WLST Command Requirements

Note

If you need to change directory (`cd`) to an object name that includes a slash (/) in its name, include the configuration object name in parentheses—for example:
`cd('JMSQueue/(jms/REGISTRATION_MDB_QUEUE)')`

You cannot access security information through WLST while updating a domain. When you use WLST and a domain template, you can create and access security information only when you create a new domain.

Running WLST Scripts

- Use the `setWLSEnv` script to initialize the `PATH` and `CLASSPATH` required for `WLST`.
  - If no script file is supplied, `WLST` runs in interactive mode.
- Use the `execfile()` command to run additional scripts.

```
>. ./setWLSEnv.sh
>java weblogic.WLST [scriptfile.py]
```

To support SSL connection to a server:

```
>java -Dweblogic.security.SSL.ignoreHostnameVerification=true
-Dweblogic.security.TrustKeyStore=DemoTrust weblogic.WLST
```

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Running WLST Scripts

Add the Oracle WebLogic Server classes to the `CLASSPATH` environment variable and `WEBLOGIC_HOME\server\bin` to the `PATH` environment variable. You can use the `WEBLOGIC_HOME\server\bin\setWLSEnv` script to set both variables. (This script is also used indirectly by other domain scripts, such as `setDomainEnv`.)

For convenience, the following additional options are available for launching WLST in interactive mode:

- The `WEBLOGIC_HOME\common\bin\wlst` script
- On Windows, the Tools > WebLogic Scripting Tool from the Start menu

Use the following system properties if you plan to connect WLST to an Oracle WebLogic Server instance through an SSL listen port, and if the server instance is using the WebLogic demonstration SSL keys and certificates:

```
Dweblogic.security.SSL.ignoreHostnameVerification=true
Dweblogic.security.TrustKeyStore=keystorename
```

Use this option to run a WLST script, where `filePath.py` is an absolute or relative path name for the script:

```
[-i] filePath.py
```

By default, WLST exits (stops the Java process) after it executes the script. Include `-i` to prevent WLST from exiting. Instead of using this command-line option, you can use the following command after you start WLST:

```
execfile('filePath.py')
```

Importing WLST as a Jython Module

- Invoke WLST:
```
[OS prompt] java weblogic.WLST
wls:/offline>
```
- Use the `writeIniFile` command to convert the WLST definitions and method declarations to a `.py` file:
```
wls:/offline> writeIniFile("wl.py")
```
- Open a new command shell and invoke Jython directly by entering the following command:
```
[OS prompt] java org.python.util.jython
```
- Import the WLST module into your Jython module by using the Jython import command:
```
>>>import wl
```
- Now you can use the WLST methods in the module. For example, to connect WLST to a server instance:
```
wl.connect('username','password')
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@dcsitte.com) has a non-transferable license to use this student guide.

Note

When using WLST as a Jython module, in all WLST commands that have a block argument, block is always set to `True`, specifying that WLST will block user interaction until the command completes.

General WLST Commands

Most of the commands need parameters in parentheses, even if null. The **`help()`** command shows this; the others are understood to be the same way.

Command	Description
`help()`	Get help for a given WLST command.
`exit`	Quit WLST.
`dumpVariables`	Display all variables used by WLST.
`dumpStack`	Display the stack trace for the last error that occurred in WLST.
`redirect / stopRedirect`	Redirect all WLST output to a file.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

General WLST Commands

See the WLS documentation for a complete list of available WLST commands, with examples for each.

To display information about the WLST commands and variables, enter the `help` command. If you specify the `help` command without arguments, WLST summarizes the command categories. To display information about a particular command, variable, or command category, specify its name as an argument to the `help` command. To list a summary of all online or offline commands from the command line, use the following commands, respectively:

```
help('online')
help('offline')
```

The `help` command supports a query—for example, `help('get*')` displays the syntax and usage information for all commands that begin with `get`.

To redirect WLST information, error, and debug messages from standard out to a file, enter:

```
redirect(outputFile,[toStdOut])
stopRedirect()
```

This command also redirects the output of the `dumpStack()` and `dumpVariables()` commands.

Offline WLST Commands

Command	Description
`createDomain`	Create a domain by using a given template.
`readDomain`	Open an existing domain on the file system.
`readTemplate`	Open an existing domain template.
`addTemplate`	Apply a template file to the current domain.
`updateDomain`	Save changes to the current domain.
`writeDomain`	Save changes to the current domain to a specified directory.
`writeTemplate`	Save the current domain to a template file.
`assign / unassign`	Target applications or services to servers.
`setOption`	Configure domain creation options (domain name, Java home, start mode, and so on).

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrynder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Offline WLST Commands

WLST offline provides read and write access to the configuration data that is persisted in the domain's `config` directory or in a domain template JAR that is created using the Domain Template Builder. This data is a collection of XML documents and expresses a hierarchy of management objects. The offline commands include:

- `createDomain(domainTemplate,domainDir, user, password)`
- `readDomain(domainDirName)`
- `readTemplate(templateFileName)`
- `addTemplate(templateFileName)`
- `writeTemplate(templateName)`
- `assign(sourceType, sourceName, destinationType, destinationName)`
- `setOption(optionName, value)`

WLST also includes the `configToScript` command that reads an existing domain and outputs a WLST script that can re-create the domain.

Set the password for the default user, if it is not already set. The default username and password must be set before you can write the domain template. For example:

```
cd('/Security/domainname/User/username')
cmo.setPassword('password')
```

Creating a Domain: Example

```
readTemplate('mybasetemplate.jar')

setOption('DomainName','mydomain')
setOption('JavaHome','/home/myjdk')
setOption('ServerStartMode','prod')

writeDomain('/home/mydomains')
closeTemplate()

readDomain('/home/mydomains/mydomain')
addTemplate('myjms.jar')
addTemplate('myapps.jar')
updateDomain()
closeDomain()
exit()
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Domain: Example

See the WLS documentation for the `setOption` command to view a list of available domain creation options.

The following example writes a domain configuration to a domain template:

```
readDomain('/home/mydomains/mydomain')
writeTemplate('myTemplate.jar')
```

The following are several examples of the `assign` command:

```
assign("Server", "myServer,myServer2", "Cluster", "myCluster")
assign("Server", "*", "Cluster", "myCluster")
assign("AppDeployment", "myAppDeployment", "Target",
"newServer")
assign("User", "newUser", "Group", "Monitors")
assign('JMSSystemResource.SubDeployment',
'myJMSResource.myQueueSubDeployment', 'Target', 'newServer')
```

Online WLST Commands

Command	Description
`connect`	Connect to a server by using supplied credentials.
`disconnect`	Disconnect from the current server.
`shutdown`	Shut down servers.
`start`	Use the Node Manager to start servers.
`startEdit`	Begin a new change session.
`stopEdit`	Release the edit lock and discard any changes.
`activate`	Commit all changes in the current session.
`showChanges`	List all changes made in the current session.
`isRestartRequired`	Determine if any changes require a server restart.
`deploy/redeploy`	Deploy an application to servers.
`undeploy`	Shut down a running application on servers.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmccrudder@yahoo.com) has a non-transferable license to use this Student Guide.

Online WLST Commands

If you run the `connect` command without specifying the username and password or the user configuration file and key file, WLST attempts to process the command by using one of the following methods (in order of precedence):

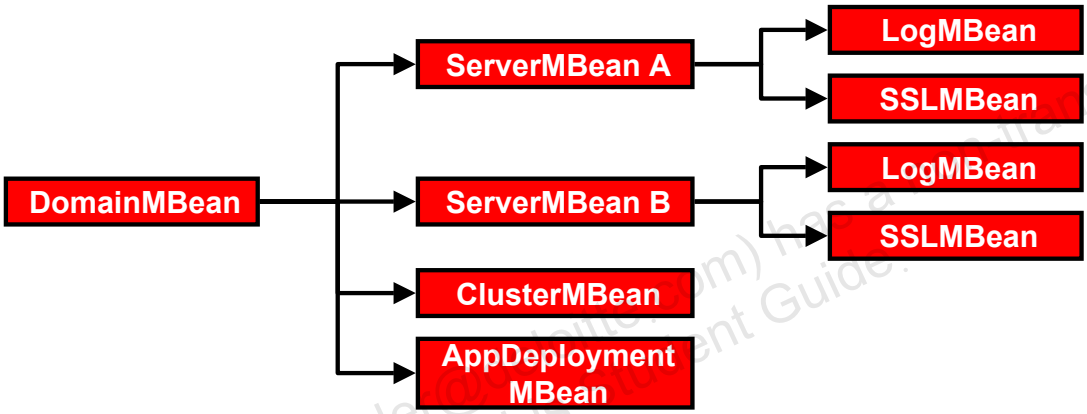
1. If a user configuration file and default key file exist in your home directory, use those files. The location of the home directory depends on the type of operating system on which WLST is running. For information about the default location, see the `storeUserConfig` command.
2. If the *`adminServerName`* argument is not specified, look for the `boot.properties` file in `./boot.properties` or `./servers/myserver/security/boot.properties`.
3. If the *`adminServerName`* argument is specified, look for the `boot.properties` file in `./servers/`*`adminServerName`*`/security/boot.properties`, where *`adminServerName`* is the value of the *`adminServerName`* argument.

It is strongly recommended that you connect WLST to the server through the SSL port or the administration port. If you do not, a warning message is displayed.

WebLogic JMX: Overview

JMX MBeans:

- Are hierarchical Java objects found on the server
- Have attributes and operations
- Support the configuration, management, and monitoring of all types of server resources



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmccrudder@yahoo.com) has a non-transferable license to use this Student Guide.

WebLogic JMX: Overview

This diagram shows the hierarchical structure of MBeans.

The JMX specification does not impose a model for organizing MBeans. However, because the configuration of an Oracle WebLogic Server domain is specified in an XML document, Oracle WebLogic Server organizes its MBeans into a hierarchical model that reflects the XML document structure. For example, the root of a domain's configuration document is `<domain>` and below the root are child elements such as `<server>` and `<cluster>`. Each domain maintains a single MBean of the DomainMBean type to represent the `<domain>` root element. Within DomainMBean, the JMX attributes provide access to the MBeans that represent child elements such as `<server>` and `<cluster>`.

All Oracle WebLogic Server MBeans can be organized into one of the following general types:

- Run-time MBeans contain information about the run-time state of a server and its resources. They generally contain data only about the current state of a server or resource, and they do not persist this data. When you shut down a server instance, all run-time statistics and metrics from the run-time MBeans are lost.
- Configuration MBeans contain information about the configuration of servers and resources. They represent the information that is stored in the domain's XML configuration documents.

Navigating JMX MBeans

- Use the `cd`, `ls`, and `pwd` commands to navigate the server configuration or run-time MBeans, similar to a file system.
- Use the `get` or `set` commands to read or update the MBean attributes.

The `cmo` variable refers to the current MBean.

```
>connect('myuser','mypass','localhost:7001')
>cd('Servers')
>ls()
dr-- AdminServer
dr-- ServerA
>cd('ServerA')
>ls()
dr-- Log
dr-- SSL
-r-- ListenPort    7011
-r-- StartupMode   RUNNING
>cd('Log/ServerA/StdoutFilter')
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Navigating JMX MBeans

WLST online provides simplified access to MBeans. The JMX APIs require you to use JMX object names to interrogate MBeans, whereas WLST enables you to navigate a hierarchy of MBeans in a similar fashion to navigating a hierarchy of files in a file system. For example, to navigate back to a parent MBean, enter the `cd('..')` command.

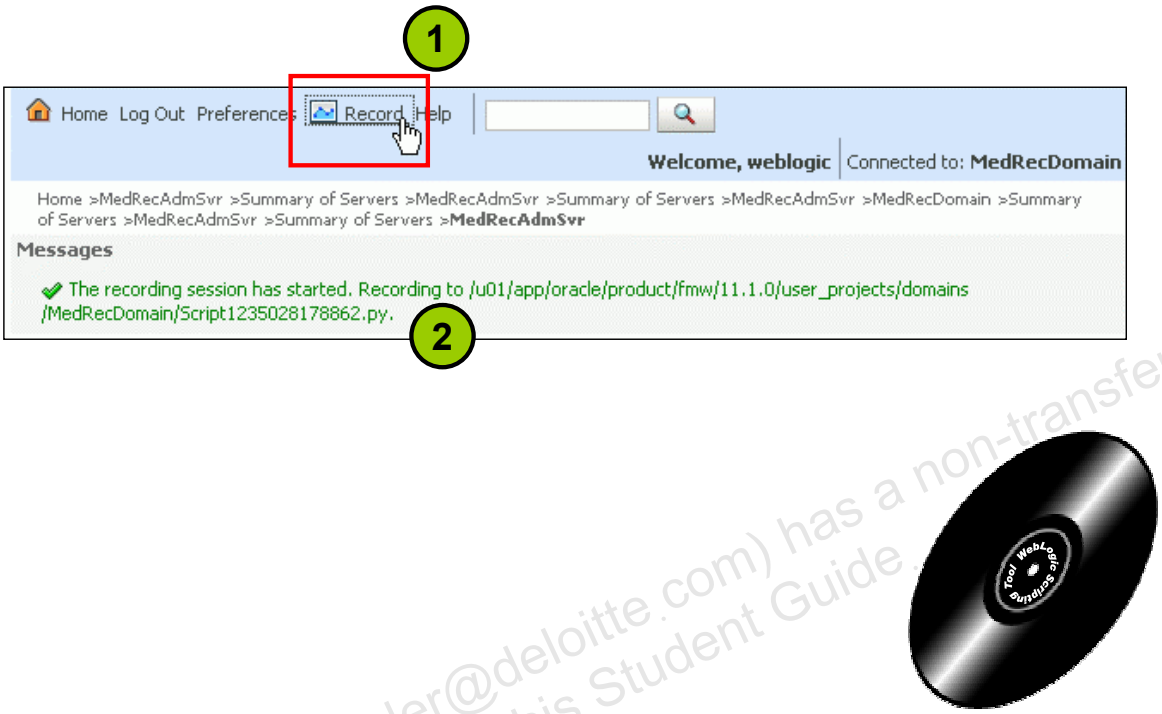
Oracle WebLogic Server organizes its MBeans in a hierarchical data model. In the WLST file system, MBean hierarchies correspond to drives, MBean types and instances are directories, and MBean attributes and operations are files. WLST traverses the hierarchical structure of MBeans by using commands such as `cd`, `ls`, and `pwd` in a similar way that you would navigate a file system in a UNIX or Windows command shell. After navigating to an MBean instance, you interact with the MBean by using WLST commands.

WLST online provides a variable, `cmo`, which represents the current management object. You can use this variable to perform any `get`, `set`, or `invoke` method on the management object. WLST sets the value of `cmo` to the current WLST path. Each time you change directories, WLST resets the value of `cmo` to the current WLST path.

In the `ls` command output information, `'d'` designates an MBean with which you can use the `cd` command (analogous to a directory in a file system), `'r'` indicates a readable property, `'w'` indicates a writeable property, and `'x'` an executable operation.

To locate a particular MBean and attribute, you use the `find` command. WLST returns the path name to the MBean that stores the attribute and its value. You can use the `getMBean` command to return the MBean specified by the path.

Generating a WLST Script



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Generating a WLST Script

This screenshot shows how to activate the Recording feature for scripts. To help automate the task of configuring a domain, you can record your configuration actions in the Administration Console as a series of WLST commands, and then use WLST to replay the commands.

1. In Change Center, click the Lock & Edit button. Then click the Record button in the toolbar region at the top of the right pane of the console.
2. A message displays in the console indicating where the generated WLST script will be written. The console records all actions that change the domain's configuration. As it records each command, it writes the command to the script file. To end the recording, click Activate Changes in the Change Center.

Several console preferences affect the WLST recording. Under Preferences, click the **WLST Script Recording** tab. The available preferences include:

- **Base Script Directory:** Specifies an existing directory into which the Administration Console writes the script file, user configuration file, and key file. Configuration and key files are used to script attributes that require encrypted values, such as passwords. By default, the scripts are stored in the `domains/`*`domain_name`* directory.
- **Automatic Recording:** Starts recording when you obtain a lock on the domain configuration and stops recording when you activate your changes, undo your changes, release the lock, or lose the lock

Quiz

Which link would you click in the console to add or remove columns to or from a monitoring page?

a. Customize this table
b. Change Monitoring View
c. Update Columns
d. Manage Preferences

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

Remember that the Monitoring tab for resources such as servers and connection pools displays statistics in a tabular format. The "Customize this table" link enables you to control the columns displayed in one of these tables.

Quiz

WLST communicates with Oracle WebLogic Server's ____ to retrieve and update resources on a running server.

a. Templates
b. Logs
c. MBeans
d. Scripts

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: c

WLST is based on JMX, which supports remote server management through MBean objects.

Quiz

The _____ panel in the Administration Console uses a tree to represent your domain resources.

a. Preferences
b. Domain Structure
c. How do I?
d. Change Center

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

The Domain Structure panel organizes your domain's resources, such as servers and application deployments.

Quiz

The Administration Console is unavailable if the administration server is shut down.

a. True

b. False

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

The administration server runs the Administration Console. Therefore, the Administration Console is unavailable whenever the administration server is unavailable.

Quiz

Both the administration server and the managed servers can be started by using the Administration Console.

a. True

b. False

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

You cannot start the administration server using the Administration Console. The Administration Console is an application that runs on the administration server.

Quiz

Using WLST's _____ mode, you can supply commands one at a time and get immediate feedback.

a. Management
b. Operational
c. Sequential
d. Template
e. Interactive

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: e

In Interactive mode, you can quickly prototype or troubleshoot some WLST commands.

Summary

In this lesson, you should have learned how to:

- Customize the console and tables
- Use the breadcrumbs to repeat recent tasks
- Operate WLST
- Describe how change management is performed internally using JMX and MBeans

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 7 Overview: Using the Administrative Console and WLST

This practice covers the following topics:

- Invoking the Administration Console and navigating through common pages
- Modifying properties of WebLogic Server using the Administration Console
- Configuring and viewing log file entries for a WebLogic Server using the Administration Console.
- Performing tasks with the WLST
- Shutting down a server or an entire domain using WLST

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 7 Overview: Using the Administrative Console and WLST

See Appendix A for the complete steps to do the practice.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

8

Configuring Managed Servers

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Configure managed servers using the Administrative Console
- Configure managed servers using WebLogic Scripting Tool (WLST)
- Start managed servers
- Shut down a server or an entire domain using WLST or the Administrative Console
- Configure managed servers on a computer separate from the administration server
- Explain administration and Managed Server Independence (MSI)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

When you created a domain, you had the option to create an administration server and a managed server. At the time, you only had one host. Since then budget money has come through. Now that you have more and bigger hosts, you want to make more managed servers and modify the existing ones. The department that runs the medical records has money to buy its own host to be used as an application server. Although “server” does not necessarily equal “computer,” in this case, the department wants to do a proof of concept by making its next few managed servers on the same computer as the other managed server and the administration server.

Road Map

- Managed servers
  - Configuring managed servers
  - Starting managed servers
  - Stopping managed servers
- Remote managed servers
- Managed Server Independence (MSI)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

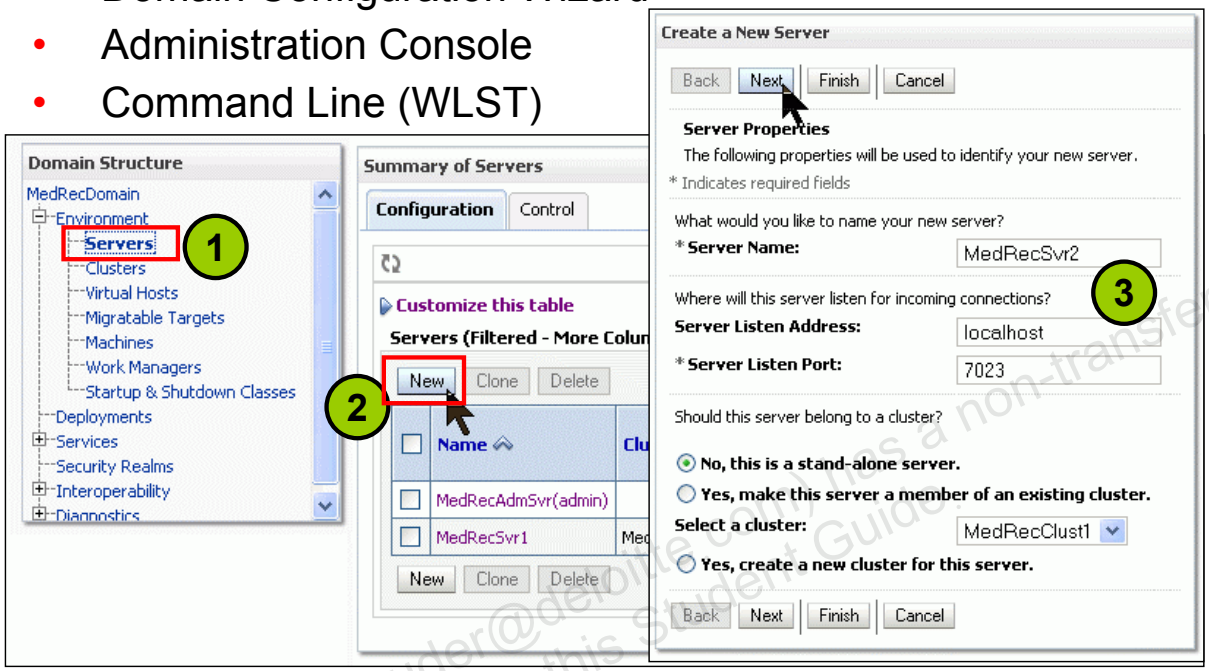
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring Managed Servers

You can configure managed servers by using the following:

- Domain Configuration Wizard
- Administration Console
- Command Line (WLST)



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring Managed Servers

The screenshot shows creating a managed server with the Administration Console. You have already seen how to use the Domain Configuration Wizard to create a new managed server in the lesson titled "Using Administration Console and WLST."

To create a managed server by using the WebLogic Administration Console, perform the following steps:

1. Click Servers in the Explorer pane (to the left of the Administration Console).
2. Click New to configure a new server.
3. Enter the server details in the WebLogic console. If there are multiple servers on the same host, each server needs to have different listen ports. There is a difference from an SSL Certificate standpoint between localhost and 127.0.0.1, or between any DNS name or alias and an IP address. If you leave Server Listen Address blank, any host or alias name for that server will work in a URL. For example, for the lab, null (blank) is the same as localhost, wls-sysadm, edvmr1p0, and the FQN. However, if you enter `localhost`, http://wls-sysadm and http://edvmr1p0 would not work.
4. Review the choices *(not shown)* and click Finish.

Even though the new server is created, it is not yet started.

Creating a Managed Server with WLST

```
[oracle@wls-sysadm /]$ java weblogic.WLST
wls:/offline> connect('weblogic','mypassword','t3://localhost:7020')
 Connecting to t3://localhost:7020 with userid weblogic ...
 Successfully connected to Admin Server 'MedRecAdmSvr' that belongs to
 domain 'MedRecDomain'.
wls:/MedRecDomain/serverConfig> cd('Servers')
wls:/MedRecDomain/serverConfig/Servers> edit()
wls:/MedRecDomain/edit> startEdit()
wls:/MedRecDomain/edit !> server1=create('MedRecSvr3','Server')
 MBean type Server with name MedRecSvr3 has been created successfully.
wls:/MedRecDomain/edit !> server1.getName()
 'MedRecSvr3'
wls:/MedRecDomain/edit !> ls('Servers')
 drw-   MedRecAdmSvr
 drw-   MedRecSvr1
 drw-   MedRecSvr2
 drw-   MedRecSvr3
wls:/MedRecDomain/edit !> save()
wls:/MedRecDomain/edit !> activate()
wls:/MedRecDomain/edit !> stopEdit()
wls:/MedRecDomain/edit> exit()
[oracle@wls-sysadm /]$
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Managed Server with WLST

Use the WLST `create` command to create a new managed server:

`create`

The `create` command works only in the context of `edit()` and `startedit()`. It can be used with WLST in either online (connected) or offline mode.

Description

The `create` command creates a configuration bean of the specified type for the current bean; it returns a stub for the newly created configuration bean.

Note: You must create child types under an instance of their parent type. You can create only configuration beans that are children of the Current Management Object (CMO) type.

The `activate` command supports two optional arguments:

- **Timeout:** The time (in milliseconds) that WLST waits for the activation of configuration changes to complete before canceling the operation. A value of `-1` indicates that the operation will not time out. This argument defaults to 300,000 ms (or 5 minutes).
- **Block:** Whether WLST should block user interaction until the command completes. This argument defaults to false, indicating that user interaction is not blocked.

Creating a Managed Server with WLST (continued)

Note the following when you use the `create` command with **WLST online:**

- You must be connected to an administration server. You cannot use the `create` command for run-time MBeans or when WLST is connected to a managed server instance.
- You must navigate to the edit configuration MBean hierarchy using the `edit` command before issuing this command.
- You can use the `create` command to create an Oracle WebLogic Server configuration MBean that is a child of the current MBean type.
- Note the following when using the `create` command with **WLST offline:**
  - When using WLST offline, the following characters are not valid in object names: period (.), slash (/), or backslash (\).

Syntax

```
create(name, childMBeanType, [baseProviderType])
```

Argument Definition

- **`name`:** Name of the configuration bean that you are creating
- **`childMBeanType`:** Type of configuration bean that you are creating. You can create instances of any type that are defined in the `config.xml` file except custom security types.
- **`baseProviderType`:** When creating a security provider, it specifies the base security provider type (for example, `AuthenticationProvider`). This argument defaults to None.

Using WLST, you can create a child configuration bean of type Server, named `newServer` for the current configuration bean, and store the stub as `server1`:

```
wls:/mydomain/edit !> server1=create('newServer','Server')
MBean Server with name newServer has been created successfully.
wls:/mydomain/edit !> server1.getName()
'newServer'
```

Here is an alternative way to programmatically create a series of servers and ports using a `for` loop:

```
servers = {'serverB':7021,'serverC':7031}

connect('myAdminuser','Welcome1','localhost:7001')
edit()
startEdit()

for name, port in servers.items():
   cd('/Servers')
   create(name,'Server')
   cd(name)
   set('ListenPort',port)

save()
activate()
disconnect()
exit()
```

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting Oracle WebLogic Managed Servers

You can start managed servers using:

- *DOMAIN_DIR*/bin/startManagedWebLogic.sh
- weblogic.Server
- WLST and Node Manager
- Administration Console
  - Requires Node Manager on each machine
  - Requires additional configuration. Examples:
    - Username and password
    - Listen ports
    - CLASSPATH, JAVA_PATH
    - Security type (plain versus SSL)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting Oracle WebLogic Managed Servers

There are four ways to start a managed server as listed in the slide. The first two are covered in detail in the next few slides.

startManagedWebLogic

The startManagedWebLogic script is the easiest to run (only one line). It has no special requirements.

weblogic.Server

Calling weblogic.Server from Java has many options to type (and is prone to typos). It is probably used only inside of a script, not typed from the command line. For example, it is used by startManagedWebLogic.

WLST and Node Manager

WLST requires that Node Manager be set up on each machine beforehand. This is useful for scripts, but involves some work to get it working the first time.

Administration Console

The Administration Console is the most powerful and easiest way to manage servers through the GUI, but it requires the most setup work with Node Manager running on each machine beforehand.

Note: You learn how to start managed servers using WLST and Node Manager in the lesson titled, "Configuring Node Managers."

Starting a Managed Server Using `startManagedWebLogic.sh`

- Start the domain's Administration server.
- Type *DOMAIN_NAME*/bin/startManagedWebLogic.sh *managed_server_name [admin_url]*.

```
[oracle@wls-sysadm /]$ cd /u01/app/work/domains/MedRecDomain/bin
[oracle@wls-sysadm bin]$ ls
nodemanager          setDomainEnv.sh            startWebLogic.sh
server_migration     startManagedWebLogic.sh    stopManagedWebLogic.sh
service_migration    startPointBaseConsole.sh   stopWebLogic.sh
[oracle@wls-sysadm bin]$ ./startManagedWebLogic.sh MedRecSvr1 http://myAdminSvr:7020
```

```
<Feb 2, 2010 11:46:22 AM EST> <Info> <Security> <BEA-090065> <Getting boot identity
      from user.>
Enter username to boot WebLogic server:weblogic
Enter password to boot WebLogic server:********
```

```
<Feb 2, 2010 12:01:32 PM EST> <Notice> <WebLogicServer> <BEA-000330> <Started WebLogic Managed Server
      "MedRecSvr1" for domain "MedRecDomain" running in Production Mode>
<Feb 2, 2010 12:01:36 PM EST> <Notice> <Cluster> <BEA-000102> <Joining cluster MedRecClust1 on
      192.168.0.1:7009>
<Feb 2, 2010 12:01:36 PM EST> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to RUNNING>
<Feb 2, 2010 12:01:36 PM EST> <Notice> <WebLogicServer> <BEA-000360> <Server started in RUNNING mode>
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can McCrudder (cancrudder@yahoo.com) has a non-transferable license to use this Student Guide.

Starting a Managed Server Using `startManagedWebLogic.sh`

Your domain directory includes a start script named `startManagedWebLogic` that you can use to start managed servers. You can use this script to start all the managed servers in a cluster. This script does not use Node Manager to start and manage the server. Instead, it uses a Java command to invoke the `weblogic.Server` class, which is the main class for an Oracle WebLogic Server instance.

To start managed servers, perform the following steps:

1. Start the domain's administration server.
2. In a shell (command prompt) on the computer that hosts the managed server, change to the directory that contains the `startManagedWebLogic` script: *DOMAIN_NAME*/bin/startManagedWebLogic.sh

   Here, *DOMAIN_NAME* is the directory in which you located the domain. By default, this directory is <MW_HOME>\user_projects\domains*DOMAIN_NAME*. If the administration server is on the same computer as the managed server, you need not type in `admin_url`.

Starting a Managed Server Using `startManagedWebLogic.sh` (continued)

Enter the following command:

```
startManagedWebLogic.sh managed_server_name [admin_url]
```

Here, *managed_server_name* (required) specifies the name of the managed server and *admin_url* (optional) specifies the listen address (host name or IP address) and port number of the domain's administration server.

For each managed server that you want to start, open a separate command shell and follow the preceding steps. If you are starting managed servers on another machine, log in to that machine (remotely or locally), and then perform the same steps.

The `startManagedWebLogic` script does the following:

- It calls the `startWebLogic` script, which sets the environment variables by invoking *DOMAIN_NAME*`\bin\setDomainEnv.sh`.
- It invokes the `java weblogic.Server` command, which starts a JVM that is configured to run an Oracle WebLogic Server instance.

When the server successfully completes its startup process, it writes the following message to standard out (which, by default, is the command page):

```
<Notice> <WebLogicServer> <000360> <Server started in RUNNING mode>
```

After completing the startup, the process creates the server directory structure in the domain if it did not exist before.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide

Command-Line Requirements for Starting the Managed Server Using `java weblogic.Server`

- Run `<WL_HOME>/server/bin/setWLSEnv.sh`.
- Start the administration server:

```
java weblogic.Server
```

- Start a managed server:

```
java
   -Dweblogic.Name=managed_server_name
   -Dweblogic.management.server=url_Admin_Server
   weblogic.Server
```

Substitute name such as `MedRecSvr2`

Substitute IP or DNS address such as
`192.168.0.1:7020`
or `localhost:7020`
or `myAdminSvr:7020`
or `myAdminSvr.us.example.com:7020`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can Moeruder (canmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Command-Line Requirements for Starting the Managed Server Using `java weblogic.Server`

The `weblogic.Server` class is the main class for an Oracle WebLogic Server instance. If you were writing your own start script (you would not *enter* this each time), you can start a server instance by directly invoking `weblogic.Server` in a Java command.

- In a command shell, set up the required environment variables by running the following script:
  `<WL_HOME>/server/bin/setWLSEnv.sh`
  where `<WL_HOME>` is the directory in which you installed the Oracle WebLogic Server software

In the command shell, change to the root of the domain directory, usually `<MW_HOME>/user_projects/domains/`*`DOMAIN_NAME`*. For example, change to `/u01/app/work/domains/MedRecDomain`.

- Start an administration server using the following command:
  `java weblogic.Server`
- If the domain's administration server is already running, and if you have already defined a managed server in the `config.xml` file, you can start a managed server as follows:
  `java -Dweblogic.Name=`*`managed-server-name`*
  `-Dweblogic.management.server=`*`url-for-Admin-Server:port`*
  `weblogic.Server`

Command-Line Requirements for Starting the Managed Server Using `java weblogic.Server` (continued)

When you start a managed server, you must specify the administration server URL that maintains the configuration information for the managed server. The `weblogic.Server` command that allows managed servers to point to their administration server is:

```
-Dweblogic.management.server
```

If you exclude this command, by default, you start the server as an administration server and not as a managed server. In the preceding example, you would specify the following in the `weblogic.Server` command to start the managed servers:

```
-Dweblogic.management.server=http://192.168.0.1:7001
```

You can also connect using HTTPS:

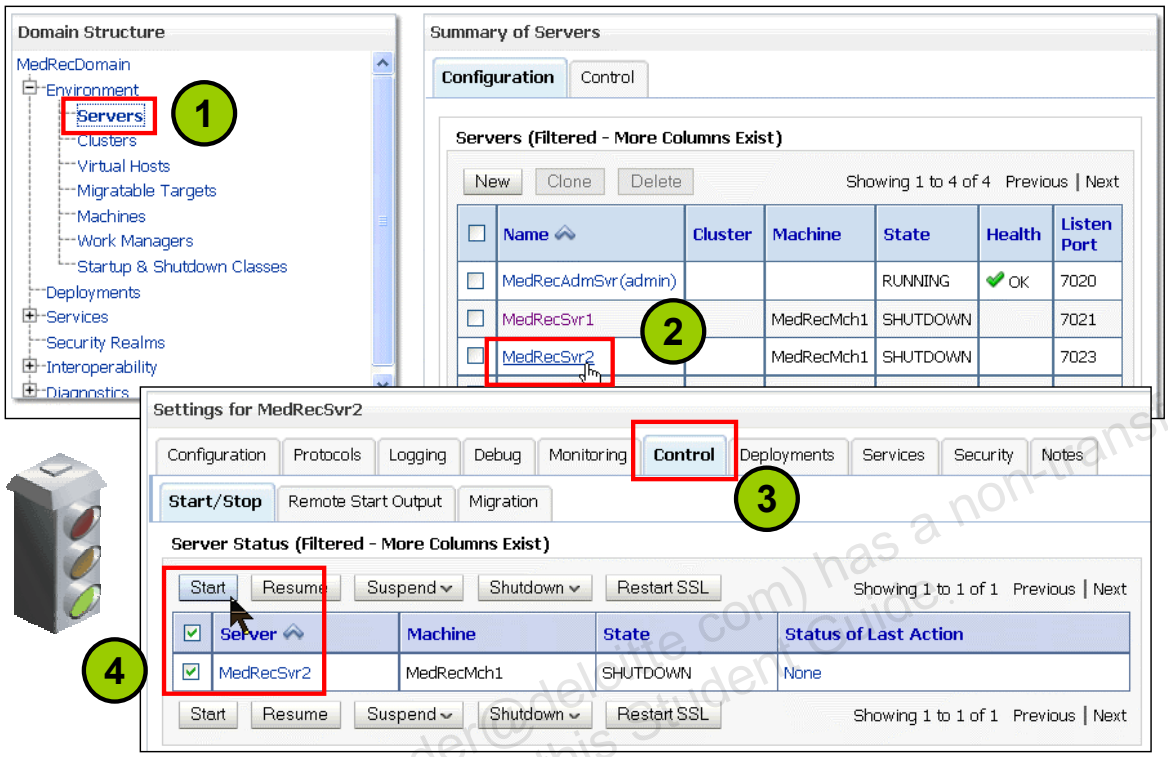
```
-Dweblogic.management.server=https://192.168.0.1:7002
```

There should be no spaces surrounding the = sign, and in the event that embedded spaces must be present in the value, the value must be enclosed within double quotation marks.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting a Managed Server Using the Administration Console



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCluder (carl.mccluder@deloitte.com) has a non-transferable license to use this Student Guide.

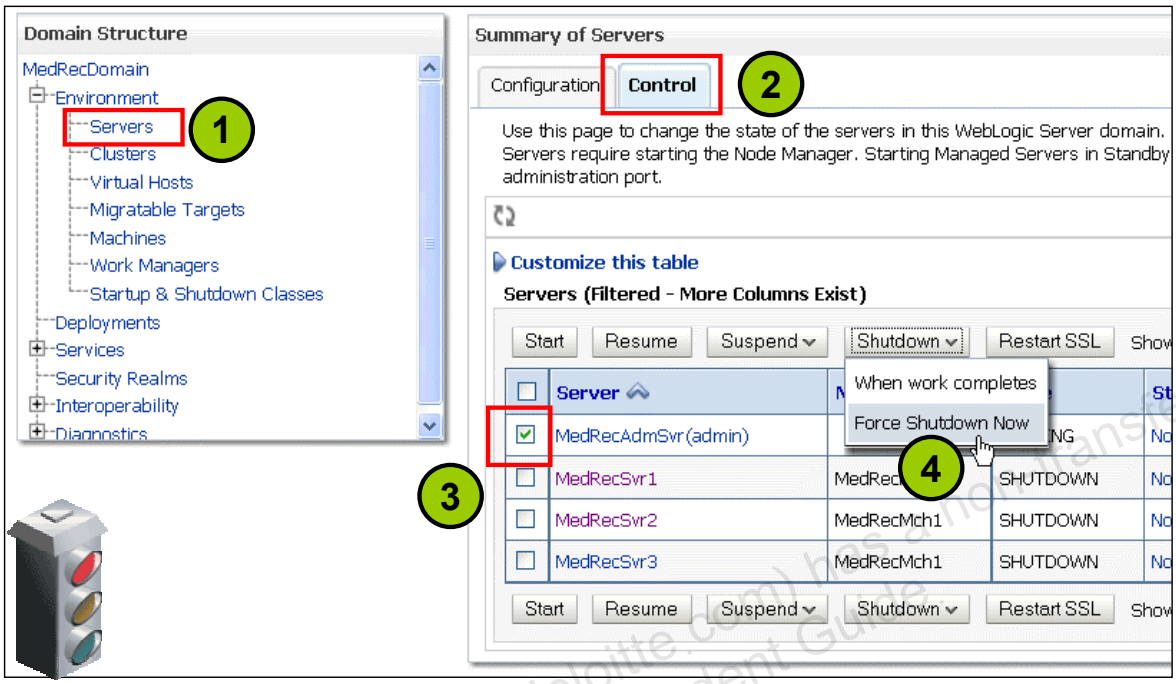
Starting a Managed Server Using the Administration Console

The screenshots show how to start a managed server. For obvious reasons, you cannot use the Administration Console to start the administration server, which runs the Administration Console. To use the Administration Console to start managed servers, perform the following steps:

1. In the left pane of the Console, expand **Environment** and click **Servers**.
2. In the **Servers** table, click the name of the managed server instance that you want to start.
3. Click the **Control** tab and then the **Start/Stop** subtab.
4. In the **Server Status** table, select the check box next to the name of the server that you want to start and click **Start**. Note that MedRecSvr2 is associated with a machine MedRecMch1, whereas MedRecSvr4, for example, might not be associated with a machine (blank in the Machine column). Therefore, MedRecSvr4 could not be started from the Administration Console.
5. On the Server Life Cycle Assistant page, click **Yes** to confirm. The Node Manager starts the server on the target machine. When the Node Manager finishes its start sequence, the server's state is indicated in the **State** column in the **Server Status** table.

Note: The Node Manager must be configured before you start the managed server by using the Administration Console. You learn how to configure the Node Manager in the lesson titled "Configuring Node Managers."

Shutting Down a Server



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Shutting Down a Server

This screenshot shows how to shut down a server by using the console:

1. Select Environment > Servers in the left pane.
2. Click the **Control** tab in the right pane.
3. Select the check box for the server that you want to shut down.
4. Select either "Force Shutdown Now" or "When work completes" from the Shutdown menu. A confirmation message appears in the right pane. Click **Yes** to proceed.

If you shut down the administration server, you will lose the browser console display. You will need to restart the administration server from a command line.

A server cannot be started by using the console out of the box. The Oracle WebLogic Server Node Manager must be configured to start a managed server from the console. For more information about the Node Manager, visit the Oracle WebLogic Server online documentation.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Shutting Down a Domain

1. Connect to the administration server.
2. Obtain a list of servers.
3. Shut down the servers using the options; shut down the managed servers first.
4. Shut down the administration server to which you are connected.

```
connect('weblogic','weblogic','t3://wls-sysadm.example.com:7020')
ls('Servers')
shutdown('MedRecSvr1')
shutdown('MedRecAdmSvr')
exit()
```

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Shutting Down a Domain

WLST `shutdown('')` gracefully shuts down a running server instance or a cluster. This command can wait for all the in-process work to be completed before shutting down the server or cluster.

WLST uses the Node Manager to shut down a managed server. When shutting down a managed server, the Node Manager must be running.

Syntax: `shutdown([name], [Type], [ignoreSessions], [timeOut], [force], [block])`

name = Optional. Name of the server or cluster to shut down. Default is the server to which WLST is currently connected.
Type = Type, Server, or Cluster. Default is Server.
ignoreSessions = Boolean value specifying whether WLST should drop all HTTP sessions immediately or wait for HTTP sessions to complete or time out while shutting down. Default is false, indicating that all HTTP sessions must complete or time out.
timeOut = Time (in milliseconds) that WLST waits for subsystems to complete in-process work before shutting down the server. Default is 0 seconds, or no timeout.
force = Boolean value specifying whether WLST should terminate a server instance or a cluster without waiting for the active sessions to complete. Default is false.
block = Boolean value specifying whether WLST should block user interaction until the server is shut down. Default is false. If Block = false, WLST returns control to the user immediately after issuing the command and assigns the MBean associated with the current task to a variable that you can use to check its completion status.

Shutting Down a Domain (continued)

In the following example, many of the steps are optional (for example, all of the **cd** steps) and are included only to show the changing prompts and the responses from the system while the shutdown is in progress.

```
[oracle@wls-sysadm /]$ java weblogic.WLST

Initializing WebLogic Scripting Tool (WLST) ...
Welcome to WebLogic Server Administration Scripting Shell
Type help() for help on available commands

wls:/offline> connect('weblogic','weblogic','t3://wls-sysadm.example.com:7020')
Connecting to t3://wls-sysadm.example.com:7020 with userid system ...
Successfully connected to Admin Server 'MedRecAdmSvr' that belongs to domain
'MedRecDomain'.

Warning: An insecure protocol was used to connect to the server. To ensure on-
the-wire security, the SSL port or Admin port should be used instead.

wls:/MedRecDomain/serverConfig> cd('Servers')
wls:/MedRecDomain/serverConfig/Servers> cd('MedRecSvr1')
wls:/MedRecDomain/serverConfig/Servers/MedRecSvr1> help('shutdown')

Description:
Gracefully shuts down a running server instance or a cluster.
...(many more lines with syntax examples)

wls:/MedRecDomain/serverConfig/Servers/MedRecSvr1> shutdown('MedRecSvr1')
Shutting down the server MedRecSvr1 with force=false while connected to
MedRecAdmSvr ...
wls:/MedRecDomain/serverConfig/Servers/MedRecSvr1> cd('..')
wls:/MedRecDomain/serverConfig/Servers> cd('MedRecAdmSvr')
wls:/MedRecDomain/serverConfig/Servers/MedRecAdmSvr> shutdown('MedRecAdmSvr')
Shutting down the server MedRecAdmSvr with force=false while connected to
MedRecAdmSvr ...
WLST lost connection to the WebLogic Server that you were connected to, this may
happen if the server was shutdown or partitioned. You will have to re-connect to
the server once the server is available.
Disconnected from weblogic server: MedRecAdmSvr
Disconnected from weblogic server:
wls:/offline> exit()

Exiting WebLogic Scripting Tool.

[oracle@wls-sysadm /]$
```

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Boot Identity File

- Create a file called `boot.properties` in the *DOMAIN_NAME*`/servers/<server_name>/security` directory that contains two lines:
  - `username=`*`username`*
  - `password=`*`password`*
- The first time you start the server, the server reads the boot identity file and overwrites it with an encrypted version of the username and password.
- Thereafter, the server remembers the credentials for subsequent startup cycles.

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Boot Identity File

A boot identity file is a text file that contains user credentials for automatically starting and stopping an instance of Oracle WebLogic Server. Technically you are not "logging in" as anybody, but simply validating that you have the credentials (authority) to start the instance. An administration server can refer to this file for user credentials instead of prompting you to provide them. Because the credentials are encrypted, using a boot identity file is much more secure than storing unencrypted credentials in a startup or shutdown script. If there is no boot identity file when starting a server, the server instance prompts you to enter a username and password.

If you use the Configuration Wizard to create a domain in development mode, the Configuration Wizard creates an encrypted boot identity file in the security directory of the administration server's root directory.

If you start a managed server from a script that invokes the `java weblogic.Server` command (or if you invoke the `java weblogic.Server` command directly), the managed server can also refer to a boot identity file. If a managed server's security directory contains a valid `boot.properties` file, it uses this file during its startup process by default. The `boot.properties` file will be different for each server instance in the domain (after encryption).

Creating a Boot Identity File (continued)

If you use the Node Manager to start a managed server, the Node Manager encrypts and saves the credentials with which it started the server in a server-specific `boot.properties` file for use in automatic restarts. This file is located in *DOMAIN_NAME*/`servers`/*SERVER_NAME*/`data/nodemanager`, where *DOMAIN_NAME* is the name of the directory in which you created the domain and *SERVER_NAME* is the name of the server.

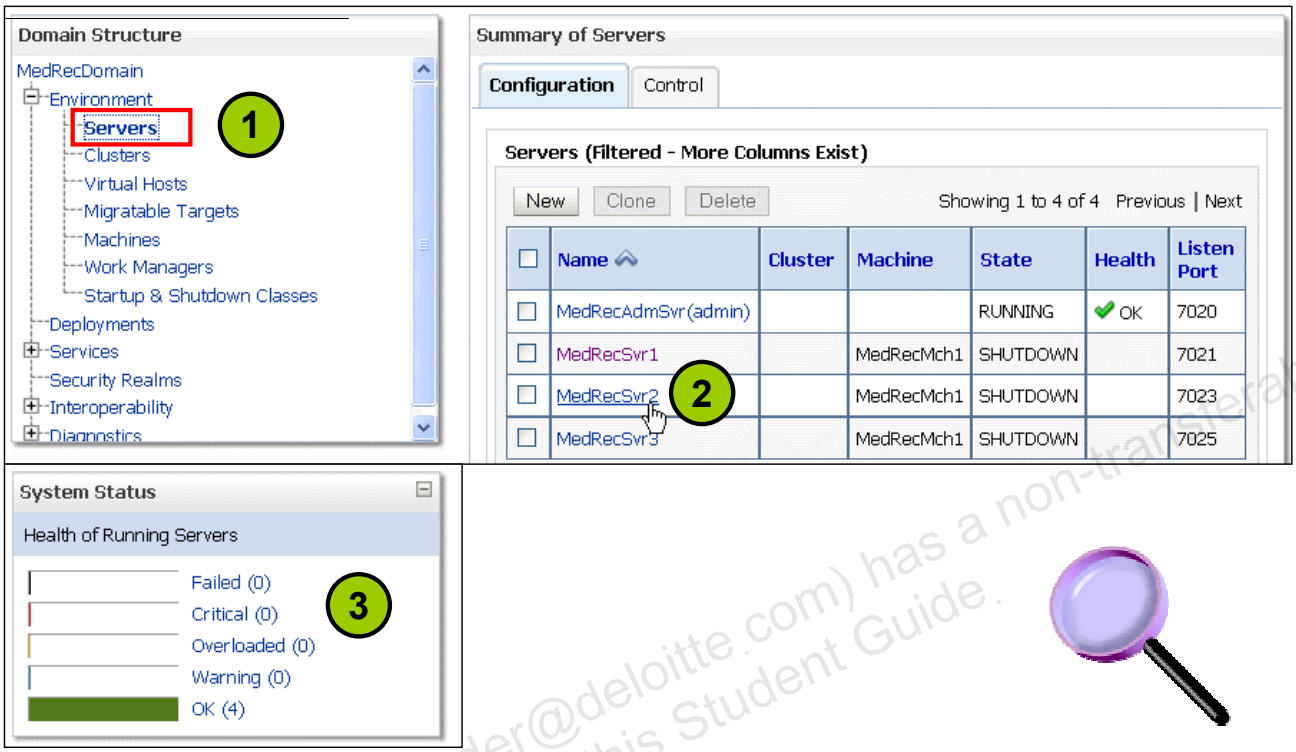
For a given server instance, use only the boot identity file that the instance created. Oracle WebLogic Server does not support copying a boot identity file from the server root directory to another directory.

For example, if you use ServerA to generate a boot identity file, use only that boot identity file with ServerA. Do not copy ServerA's boot identity file into the security directory of ServerB. Instead, create a boot identity file for ServerB.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Monitoring All Servers



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Monitoring All Servers

This screenshot shows a list of all the servers, running and otherwise. To monitor the state of all servers, perform the following steps:

1. Select Environment > Servers in the left pane.
2. This page summarizes each server that has been configured in the current Oracle WebLogic Server domain. The available table columns shown (others not shown) include:
   - **Cluster:** The cluster or group of Oracle WebLogic Server instances to which this server belongs
   - **Machine:** The Oracle WebLogic Server host computer (machine) on which this server is meant to run. If you want to use a Node Manager to start this server, you must assign the server to a machine and configure the machine for the Node Manager.
   - **State:** The current life cycle state of the server. For example, a server can be in a RUNNING state in which it can receive and process requests, or in an ADMIN state in which it can receive only administrative requests.
   - **Health:** The health state of the server as reported by the server's self-health monitoring. For example, the server can report if it is overloaded with too many requests, if it needs more memory resources, or if it will soon fail for other reasons.

Monitoring All Servers (continued)

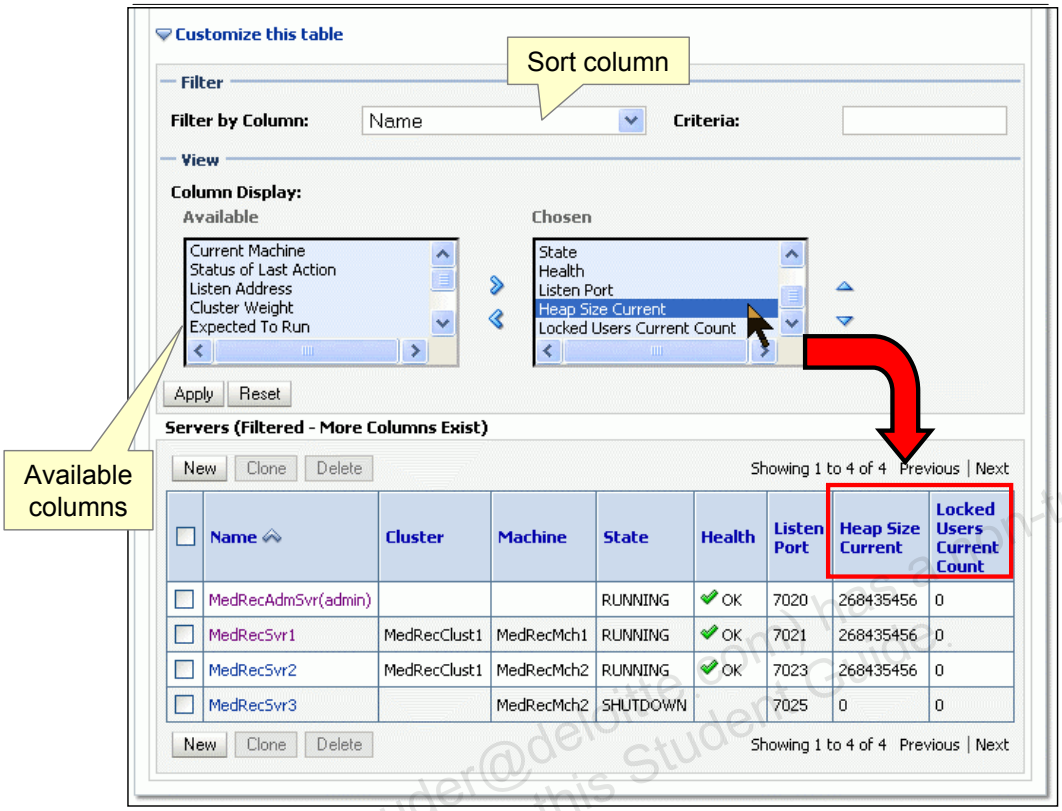
- **Listen Port:** The default TCP port that this server uses to listen for regular (non-SSL) incoming connections
- **Listen Address:** The IP address or DNS name that this server uses to listen for incoming connections. Any network channel that you configure for this server can override this listen address. If a server's listen address is undefined, clients can reach the server through an IP address of the computer that hosts the server, a DNS name that resolves to the host, or the localhost string.

3. The System Status will show if a server is Failed, Critical, Overloaded, or Warning. Simply being Shutdown is not a "bad" thing as far as the Status is concerned because it only shows Running Servers. The number of servers in that state is shown in parentheses.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Customizing the View for All Servers



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

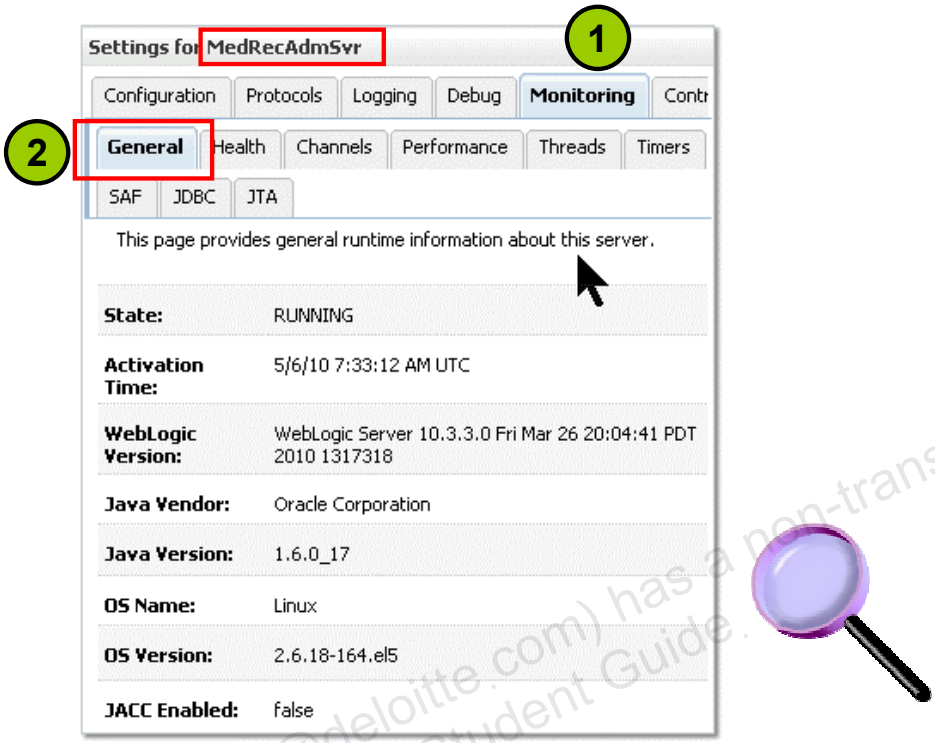
Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Customizing the View for All Servers

This screenshot shows adding two columns to the table display. When the data in the console is displayed as a table that lists objects of a particular type, you can customize the table by filtering the data displayed and adding or subtracting columns and rows. You can also sort the data tables by clicking the column headers.

1. Navigate to an object table and click “Customize this table,” located at the top of the table.
2. Select a column for filtering the data displayed.
3. Specify an optional filtering criteria (a text string) in the Criteria text box. The filtering criteria is specified as a string in the WebLogic Diagnostics Framework (WLDF) Query Language. The query language supports the Boolean operators: `AND`, `OR`, and `NOT`. It supports the relational operators: `>`, `<`, and `LIKE`.
4. Add attributes to display by moving them from the list of Available items (on the left) to the Chosen items (on the right).
5. Select the number of rows to display and the maximum number of rows displayed.

Monitoring Individual Servers



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Monitoring Individual Servers

This screenshot shows how to monitor information about one of the servers.

1. Select an individual server from within the console, and click its **Monitoring** tab.
2. A large amount of detailed monitoring information can be obtained about any running instance of Oracle WebLogic Server. By default, the **General** tab is displayed.

The following are some of the tabs that you can view:

- The **Performance** tab, for example, shows a running numerical output of request throughput, waiting request, and memory in use, and gives a fast indication of server performance. For graphical performance, use the WLDF Console Extension tab.
- The **Security** tab provides statistics that are specific to security, such as invalid logins and locked-out users.
- The **Threads** tab provides information about the thread activity for the current server. The first table provides general information about the status of the thread pool. The second table provides information about the individual threads.
- Service-specific tabs, such as JTA and JDBC, provide in-depth statistics about specific WLS subsystems.

Demonstration

- Enable automated start using the boot properties file.
- Go to OTN > Tutorials > Fusion Middleware > Oracle WebLogic Server 10.3 > Installation and Configuration > Enable Auto Login using the Boot Properties File.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

- Managed servers
- Remote managed servers
- Managed Server Independence (MSI)

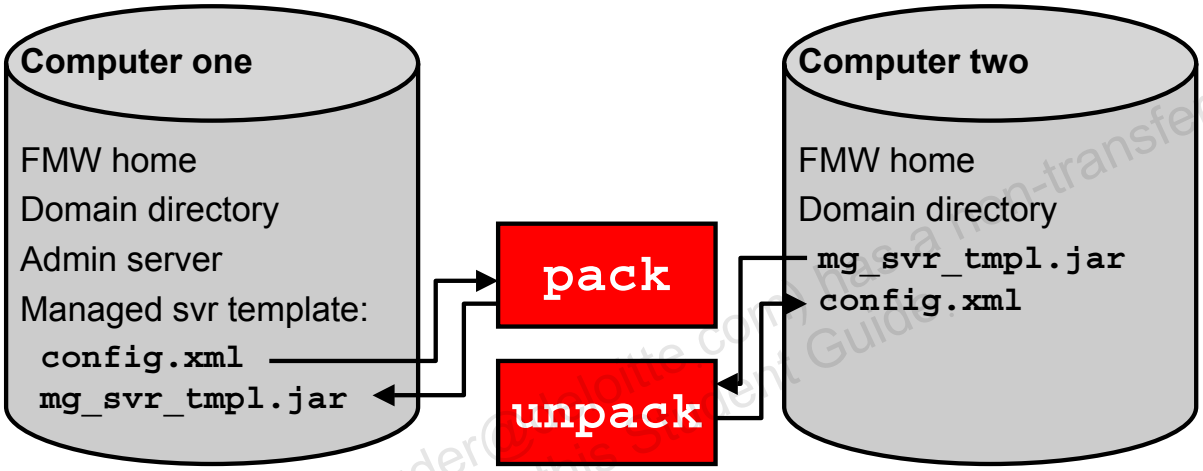
ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Managed Server on a Remote Computer

1. Install WLS on both computers.
2. Create the domain on the Admin Server.
3. Start the Admin Server.
4. Create a managed server using the Administration Console.
5. Create a managed server template using `pack`.
6. Create a managed server on a remote computer using `unpack`.
7. Start the remote managed server.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Creating a Managed Server on a Remote Computer

In some domains, you may want to run a managed server on a machine that is remote from the administration server for the domain. You can do so by using the `pack` command to create a managed server template (a `.jar` file that, by default, contains only the files necessary to create a managed server domain directory), copying the template to the remote machine, and unpacking the template to create a domain directory customized for a managed server.

The following are the steps to create a managed server domain directory on a remote machine:

1. Install WLS software on both computers. They need to be the same version of WLS software, but not necessarily the same OS.
2. Create the domain on the Administration Server computer using Configuration Wizard or `pack`.
3. Start the Administration Server.
4. Create a managed server on the local computer using the Administration Console that has the characteristics you want on the remote computer.
5. Create a managed server template using the `pack` command. A managed server template, by default, contains only those files necessary to create a managed server on a remote machine.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Managed Server on a Remote Computer (continued)

6. Create a managed server on the remote computer by copying the `*.jar` file to the remote (managed server) computer and `unpack` the managed server on the remote computer. This also creates the remote managed server `config.xml`. The two computers do not need to have the same directory structure for WebLogic Server. The use of templates allows you to have your WLS installation at different locations on each machine if you want. (`pack` tokenizes all the scripts with `@WL_HOME`, and so on, and then `unpack` fills them back in.)
7. Start the managed server on the remote machine.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

pack and unpack: Examples

On computer one (administration server):

```
[oracle@wls-sysadm]$ cd $WL_HOME/common/bin
[oracle@wls-sysadm]$ pack -managed=true
    -domain=/u01/app/oracle/domains/mydomain
    -template=/home/oracle/work/mydomain_managed.jar
    -template_name="My Managed Server Domain"
```

On computer two (remote managed server):

```
[oracle@wls-mgdsvr2]$ cd $WL_HOME/common/bin
[oracle@wls-mgdsvr2]$ unpack
    -domain=/u01/app/oracle/domains/mydomain
    -template=/home/oracle/work/mydomain_managed.jar
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

pack and unpack: Examples

You must create the managed server domain directory on the remote machine that matches the definition of the managed servers specified in the template. In the command line:

- `-domain=`: The full or relative path to the domain from which the template is to be created. Substitute `mydomain` with your domain (for example, `MedRecDomain`).
- `-template=`: The full or relative path to the template, and the file name of the template JAR file to be created. Substitute `mydomain_managed.jar` with your template (for example, `mg_svr_tmpl.jar`).
- `-template_name=`: Descriptive name for the template enclosed within double quotation marks. This need not match anything; it is only for self-documentation purposes.

Start the managed server just as you would start any other managed server by using, for example, `./startManagedWebLogic.sh MedRecSvr3`.

Road Map

- Managed servers
- Remote managed servers
- Managed Server Independence (MSI)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Managed Server Independence (MSI)

- By default, managed servers can function independently of the administration server.
- A managed server instance can start in MSI mode if the administration server is unavailable.
- Configure MSI mode from the Administration Console.
- To start a managed server in MSI mode:
  - Ensure that the managed server's root directory contains the `config` subdirectory.
  - If the `config` subdirectory does not exist, copy it from the administration server's root directory.
  - Start the managed server at the command line or by using a script.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Managed Server Independence

Usually, a managed server contacts the administration server during its startup sequence to retrieve its configuration information. If a managed server cannot connect to the administration server during startup, it can retrieve its configuration by reading its locally cached configuration data from the `config` directory.

A managed server that starts in this way is running in MSI mode.

To configure a managed server to replicate a domain's configuration files, perform the following steps:

1. If you have not already done so, in the Change Center of the Administration Console, click Lock & Edit.
2. In the left pane of the Console, select Environment > Servers and click the name of the managed server instance.
3. Select Configuration > Tuning and click Advanced at the bottom of the page.
4. Make sure that the Managed Server Independence Enabled check box is selected.
5. Click Save.
6. To activate these changes, in the Change Center of the Administration Console, click Activate Changes.

MSI Search Order

- If the administration server is unavailable at boot time, the managed servers search for:
  - `config.xml`
  - `SerializedSystemIni.dat`
  - `boot.properties(optional)`
- Each managed server looks in its local `config` directory for `config.xml`.
- You cannot change the configuration of the managed server that is running in MSI mode until it restores communication with the administration server.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

MSI Search Order

If a managed server cannot connect to the administration server during startup, it can retrieve its configuration by reading its locally cached configuration data from the `config` directory. A managed server that starts in this way is running in MSI mode.

In MSI mode, a managed server looks in its:

- Local `config` directory for `config.xml`, a replica of the domain's `config.xml`
- Security directory for `SerializedSystemIni.dat` and for `boot.properties`, which contains an encrypted version of your username and password

If `config.xml` and `SerializedSystemIni.dat` are not in these locations in the server's domain directory, you can copy them from the administration server's domain directory.

Each Oracle WebLogic Server instance writes log messages to its local log file and a domainwide log file. The domain log file provides a central location from which to view messages from all servers in a domain.

MSI Search Order (continued)

Usually, a managed server forwards messages to the administration server and the administration server writes the messages to the domain log file. However, when a managed server runs in MSI mode, it continues to write messages to its local server log file but does not forward messages to the domain log file.

A managed server that starts in MSI mode deploys its applications from its staging directory: `serverroot\stage\appName`.

You cannot use the Node Manager to start a server instance in MSI mode, but you can use it only to restart it. For a routine startup, the Node Manager requires access to the administration server. If the administration server is unavailable, you must log on to a managed server's host machine to start the managed server.

Note: The first time you start a managed server instance, it must be able to contact the administration server to get its copy of the `config.xml`. Thereafter, the managed server instance can start even if the administration server is unavailable.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

When the Administration Server Is Down

- The administration server can:
  - Go down without affecting the operation of the managed servers
  - Be restarted when the managed servers are still running
- When an administration server goes down:
  - The domain log entries are unavailable while it is down
  - Managed servers can start in independent mode
  - The Administration Console and the management tools are unavailable
  - WebLogic SNMP Agent *may* become unavailable

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

When the Administration Server Is Down

One misconception about the administration server is that it represents a single point of failure. However, this is not true. Administration servers are needed only to start managed servers. After the managed servers are started, the administration server can go down without affecting them.

The only thing that may be lost in such a scenario is the domain-specific log entries, only while the administration server is down. Also, if a new managed server tries to start, it fails because it would not be able to connect to the administration server. Lastly, the Administration Console and other management tools would not be available.

SNMP Agent *May* Become Unavailable

In each WebLogic Server domain, you can optionally create multiple SNMP agents and organize them into a decentralized or centralized model for SNMP monitoring and communication:

- In a **decentralized** model, you create SNMP agents on each managed server. SNMP managers communicate with the agents on individual managed servers. Losing the administration server does not affect SNMP on the managed servers.
- In a **centralized** model, you create an SNMP agent only on the administration server. SNMP managers communicate only with the SNMP agent on the administration server and the agent gathers monitoring data from all managed servers in the domain. This model is convenient and enables a single request to retrieve data for the entire domain. But, if the administration server is unavailable, you cannot monitor the domain through SNMP.

Running Multiple WLS Instances

- You can run multiple instances of WLS using different configurations on the same physical computer at the same time by doing either of the following:
  - Assigning multiple IP addresses to a computer (multihoming) and defining each server to use a unique IP address
  - Specifying the same IP address but using different listen ports
- A multihomed computer:
  - Is a computer with multiple IP addresses
  - Can run a different WLS instance that is bound to each IP address
  - Can be used to configure a cluster on a single computer

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Running Multiple WLS Instances

A multihomed computer is one in which multiple IP addresses are bound to a single computer. A multihomed computer may have multiple network cards or simply multiple IP addresses bound to a single network card. In either case, a computer with more than one IP address is considered to be multihomed. Multihomed computers can be configured to run multiple instances of Oracle WebLogic Server, each bound to a specific IP address and thus can simulate a multiple computer cluster environment on a single computer.

Quiz

Under the `servers` directory of WLS domain, there are subdirectories for administration and managed servers. The `servers` directory contains one subdirectory for each WebLogic Server instance in the domain. If you do not see the subdirectory for each WebLogic Server instance in your domain, it means that:

a. The WebLogic Server instance is not correctly configured.
b. The patch level is not correct.
c. The administration server is unable to communicate with the managed servers.
d. The WebLogic Server instance has not been started since it was created.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: d

The `servers` directory that contains the subdirectories for the administration and managed servers is created the first time the servers are started. This directory contains one subdirectory for each WebLogic Server instance in the domain. The subdirectories contain data that is specific to each server instance.

Quiz

Which of the following will happen if you run `startWebLogic.sh` without any options?

a. It invokes `java weblogic.Server`.

b. It starts the managed servers associated with the administration server.

c. It sets the environment using `setDomainEnv.sh`.

d. It starts the administration server.

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a, c, d

`startWebLogic.sh` does not start the managed servers. It sets the environment variable using `setDomainEnv.sh` and invokes `java weblogic.Server`, which starts the administration server.

Quiz

Which of the following options would you use to create a managed server?

a. Domain Configuration Wizard
b. Administration Console
c. Command line (WLST)

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a, b, c

You can create the managed server using either the Domain Configuration Wizard, the Administration Console, or the WLST `create` command.

Quiz

You can use boot identity files to start the following without being prompted for the administrator username and password.

a. Managed servers
b. Administration server
c. Both

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: c

You can use the `boot.properties` file to start both the administration and managed servers. The `boot.properties` file can be different for each server instance in the domain.

Quiz

Which of the following is true when the administration server is down?

a. Domain log entries are unavailable.
b. Managed servers can start in MSI mode.
c. The Administration Console and management tools are unavailable.
d. At boot time, managed servers read a local copy of `config.xml`, `SerializedSystemIni.dat`, and `boot.properties` (optional).
e. You cannot change the configuration of the managed servers that are running in MSI mode until communication with the administration server is restored.
f. The Node Manager can start the managed servers in MSI mode.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Answer: a, b, c, d, e

You cannot use the Node Manager to start a server instance in MSI mode, but you can use it only to restart it. For a routine startup, the Node Manager requires access to the administration server. If the administration server is unavailable, you must log on to a managed server's host machine to start the managed server.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Summary

In this lesson, you should have learned how to:

- Start or stop the Oracle WebLogic Server
- Configure managed servers
- Start managed servers
- Create a remote managed server
- Describe administration and MSI

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 8 Overview: Configuring a Managed Server

This practice covers the following topics:

- Creating and deleting managed servers
- Starting and stopping managed servers
- Monitoring managed servers

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 8 Overview: Configuring a Managed Server

See Appendix A for the complete steps to do the practice.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

9

Configuring Node Managers

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Define the Oracle WebLogic Server machine
- Configure a machine and assign servers to it by using the console and WebLogic Scripting Tool (WLST)
- Explain the Node Manager architecture
- Describe the organization and contents of a Node Manager directory structure
- Configure, start, and stop Node Managers
- Describe how to start and stop procedures

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

You have several servers located in a remote PC. Occasionally, the power goes out in that location and the PC comes back on by itself, but the servers need to be restarted in some fashion. To do that, you need to define the PC as a machine, assign a Node Manager to the machine, daemonize the Node Manager on that PC so it auto-restarts, and then assign the servers to that machine. This will solve the problem of the servers auto-restarting in the event of a power failure.

Road Map

- Node Managers
- Machines
- Configuring a Node Manager

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Node Managers Can Do

- You can use Node Managers to:
  - Start, shut down, and restart an administration server
  - Start, shut down, suspend, and restart managed servers
  - Automatically restart the administration and managed servers on failure
  - Monitor servers and collect log data
- Node Managers:
  - Run on the same computers as the managed servers
  - Can be run automatically in the background, as Windows services or UNIX daemons
  - Are available as either Java-based or (for UNIX only) script-based processes

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Node Managers Can Do

Starting and stopping WLS on managed servers can be a long process if many machines are involved. At best, you must start each managed server individually, and you can use the Administration Console to stop them one by one. In practice, managed servers sometimes hang and cannot be stopped the usual way. In this case, you must log on to each machine and kill the process. Again, these tasks can be tedious. It would be more convenient to start all machines on a domain simultaneously.

Starting, Shutting Down, and Restarting an Administration Server

Using the WebLogic Scripting Tool (or the SSH client for script-based Node Manager only), you connect to a Node Manager process on the machine that hosts the administration server and issue commands to start, shut down, or restart an administration server. The relationship of an administration server to a Node Manager varies for different scenarios.

- An administration server can be under a Node Manager's control. You can start it, monitor it, and restart it using the Node Manager.
- An administration server can be a Node Manager client. When you start or stop managed servers from the Administration Console, you access the Node Manager by using the administration server.
- An administration server supports the process of starting up a managed server with a Node Manager. When you start a managed server with a Node Manager, the managed server contacts the administration server to obtain the outstanding configuration updates.

What Node Managers Can Do (continued)

Starting, Shutting Down, Suspending, and Restarting Managed Servers

From the WLST command line or scripts, you can issue commands to a Node Manager to start, shut down, suspend, and restart the managed server instances and clusters.

A Node Manager can restart a managed server after failure even when the administration server is unavailable if Managed Server Independence (MSI) mode is enabled for that managed server instance. This is enabled by default.

Note

A Node Manager cannot start a managed server for the first time in MSI mode because the administration server for the domain must be available so that the managed server can obtain its configuration settings.

A Node Manager uses the same command arguments that you supply when you start a managed server with a script or at the command line. For information about the startup arguments, see "weblogic.Server Command-Line Reference" in *Oracle WebLogic Server Command Reference*.

Restarting the Administration and Managed Servers

If a server instance that was started using the Node Manager fails, the Node Manager automatically restarts it.

Note: A Node Manager can restart only a server that was started using a Node Manager.

The restart feature is configurable. The Node Manager's default behavior is to:

- Automatically restart server instances under its control that fail. You can disable this feature.
- Restart failed server instances no more than a specific number of times. You define the number of restarts by setting the `RestartMax` property in the Node Manager `startup.properties` file.

If a Node Manager fails or is explicitly shut down, upon restart, it determines the server instances that were under its control when it exited. A Node Manager can restart any failed server instances as needed.

Note: It is advisable to run Node Managers as operating system services, so that they restart automatically if its host machine is restarted.

Monitoring Servers and Viewing Log Data

A Node Manager creates a log file for the Node Manager process and a log file of server output for each server instance that it controls. You can view these log files, as well as the log files for a server instance by using the Administration Console or WLST commands.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrosky (carl.mccrosky@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

- Node Managers
- Machines
- Configuring a Node Manager

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Is a Machine?

- The main purpose of a machine is to administratively manage servers.
- A machine is required by a Node Manager.
- Machines are optionally used by clusters (described later in the course).
- A machine is a logical description, not a physical entity.

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Is a Machine?

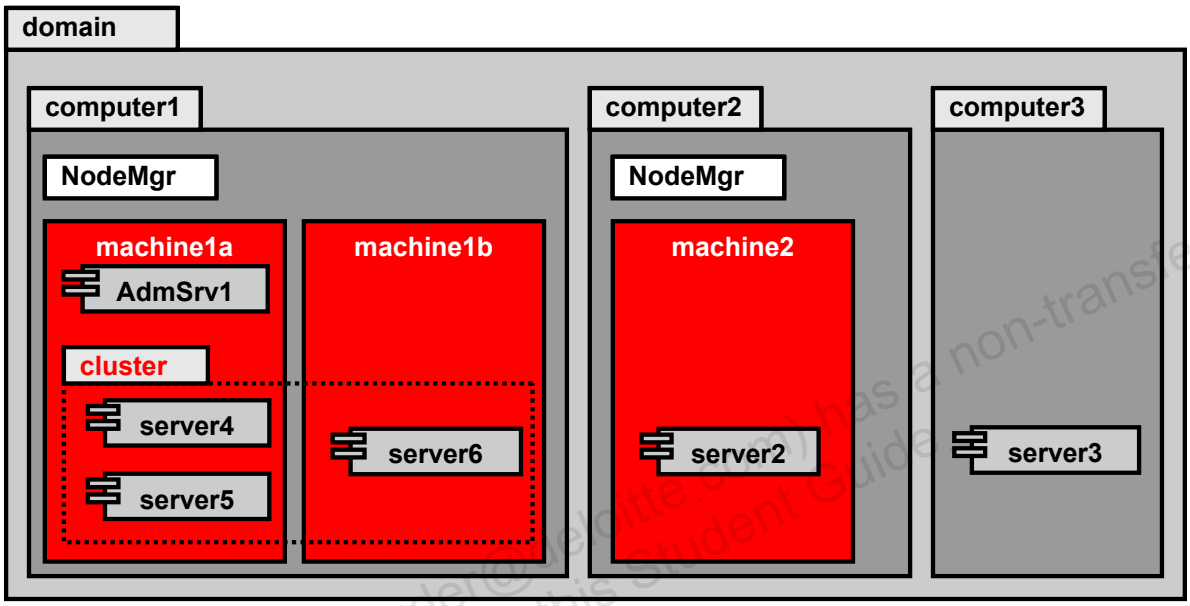
The term “machine” is a logical description that usually represents one physical computer that hosts one or more Oracle WebLogic Server instances. A machine identifies a particular, physical piece of hardware. A machine is used to associate a computer with the managed servers that it hosts. There are many other possible variations described on the next page, but this is the normal relationship between a machine and a computer. Even though multiple machines can be configured for one physical computer, Oracle recommends one machine per physical computer on production environments.

It is used by a Node Manager to restart a failed or stopped managed server and by a clustered managed server to select the best location for storing replicated session data. A Node Manager is a utility or a process running on a physical server with which you can start, stop, suspend, or restart the administration and managed servers remotely. This topic is covered in detail later in this lesson.

Oracle WebLogic Server uses configured machine names to determine the optimum server in a cluster to which to delegate tasks such as HTTP session replication. The administration server uses the machine definition with the Node Manager application to start remote Oracle WebLogic Server instances.

Relationship of Machines to Other Components

A possible (perhaps unusual) topology for WebLogic environments contains several components.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

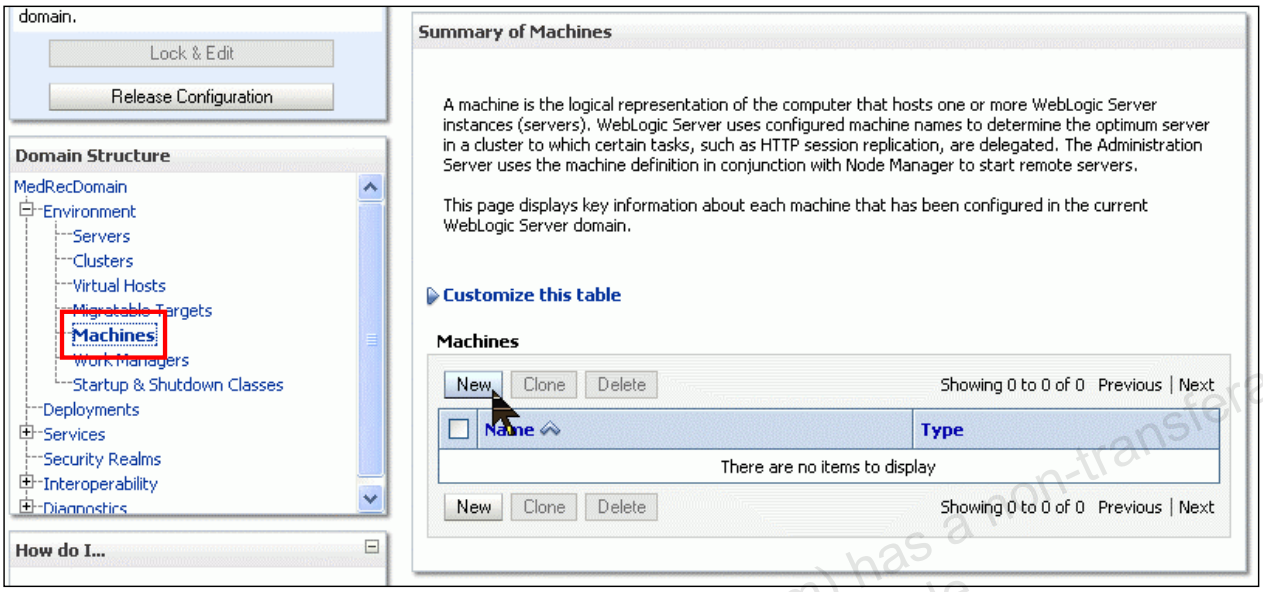
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Relationship of Machines to Other Components

This graphic illustrates some of the relationship of domains, hosts, machines, clusters, and servers. A domain is made up of one or more hosts (computers, either real or virtual). A host can contain zero, one, or more domains. A host can contain zero, one, or more machines. A machine can contain zero, one, or more servers, but it is unusual to have a machine with zero servers. A server can belong to zero or one machine. A machine cannot span hosts. A cluster can span hosts or machines, or both. A machine can have zero, one, or more clusters. A machine can have zero or one Node Manager. Although it is possible to have a machine with no Node Manager, that severely limits the purpose of having defined a machine. If you have no Node Manager on a machine, the machine can still be used for determining application load balancing, but cannot be used to remotely start servers. One Node Manager can only manage one host, but can manage servers in multiple machines or even multiple domains as long as they are on the same host as the Node Manager. The Node Manager in host1 can manage both machine1a and machine1b, and therefore, can restart server4, server5, and server6. If you have a server that does not belong to a machine (for example, server3), it cannot be started from the Administration Console, but server3 can be stopped from the Administration Console.

Creating a Machine




```
wls:/mydomain/edit> startEdit()
wls:/mydomain/edit !> create('MedRecMch3','Machine')
wls:/mydomain/edit !> save()
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Machine

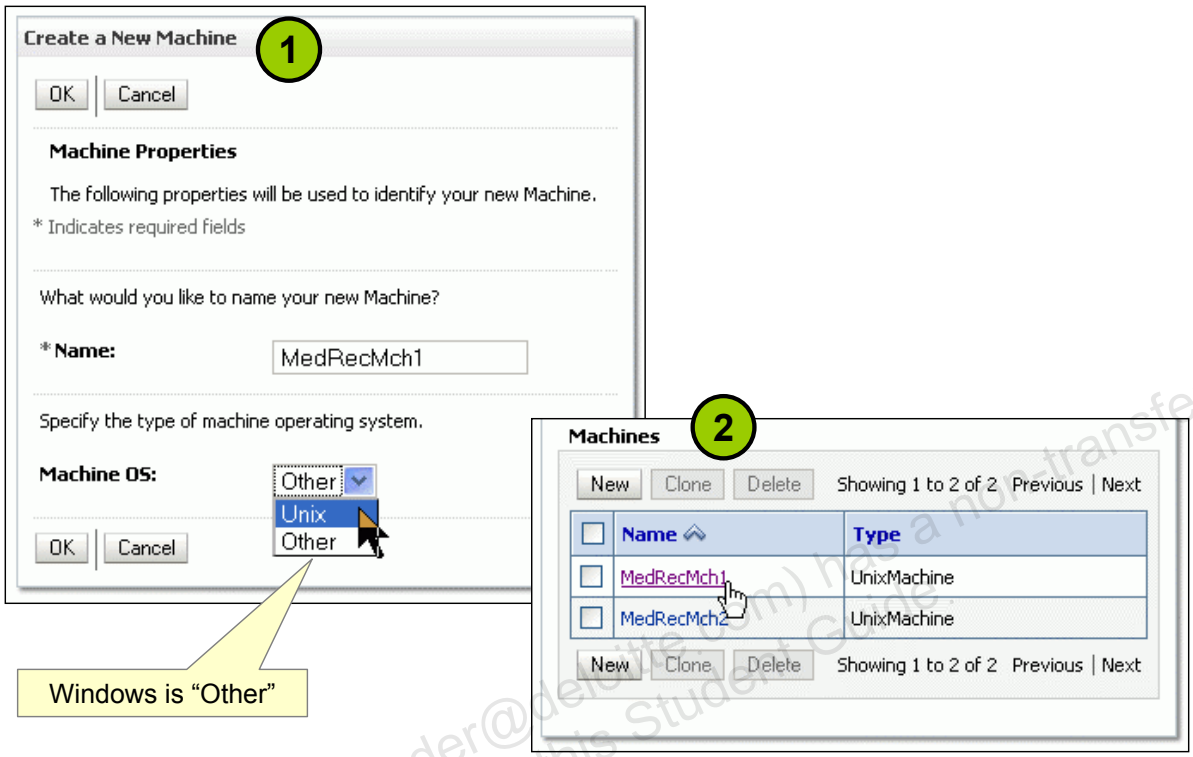
The screenshot shows the first panel for creating machines via the Administration Console as well as the WLST command line. The Administration Console can start and stop servers remotely if they are defined in a machine that is controlled by a Node Manager. You can create a machine at any time, but servers can be assigned to a machine only when those servers are in a SHUTDOWN state. In the WebLogic Administration Console, perform the following steps:

1. Select **Machines** under Domain Structure in the left pane.
2. Click **New** to configure a new machine.

Alternatively, you can create a machine using WLST. This example creates a machine named `MedRecMch3` in the `MedRecDomain` domain using WLST:

```
[oracle@wls-sysadm ~]$ java weblogic.WLST
wls:/offline> connect('weblogic','weblogic','t3://wls-sysadm:7020')
Connecting to t3://wls-sysadm:7020 with userid weblogic ...
Successfully connected to Admin Server 'MedRecAdmSvr' that belongs to
domain 'MedRecDomain'.
wls:/MedRecDomain/edit/> startEdit()
wls:/MedRecDomain/edit/ !> create('MedRecMch3','Machine')
MBean type Machine with name MedRecMch3 has been created successfully.
wls:/MedRecDomain/edit/ !> ls('Machines')
wls:/MedRecDomain/edit/ !> save()
wls:/MedRecDomain/edit/ !> activate()
```

Defining Names and OS of Machines



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Defining Names and OS of Machines

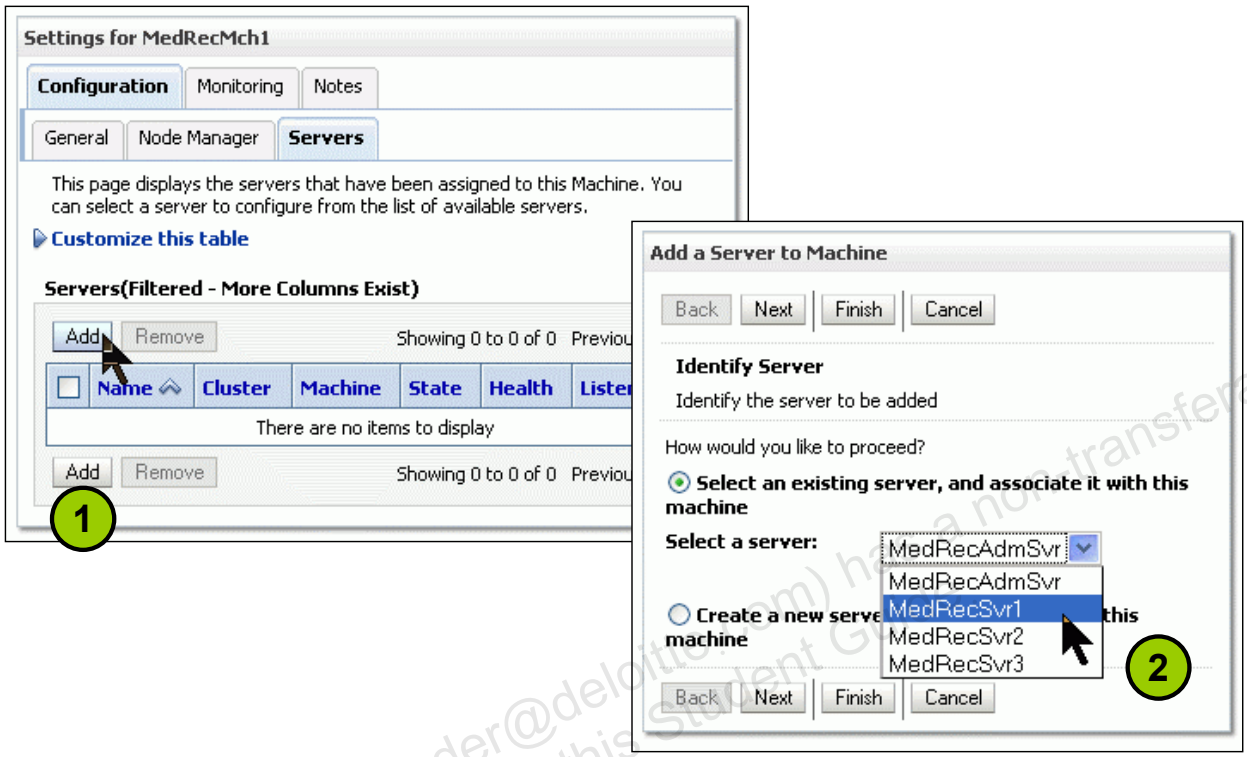
The screenshot shows the panel that is used for naming the new machine. The name can match the DNS name, but it does not have to. You may want to incorporate part of the domain name in the machine name as a prefix. For example, `myHRdomain` may include the `myHRhost1` and `myHRhost2` machines, and `myHRhost2` may include the `myHR2a` and `myHR2b` servers. The machine names (as are most WLS object names) are case-sensitive.

For Windows servers, select Other as the operating system (OS) from the drop-down list.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Assigning Servers to a Machine



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

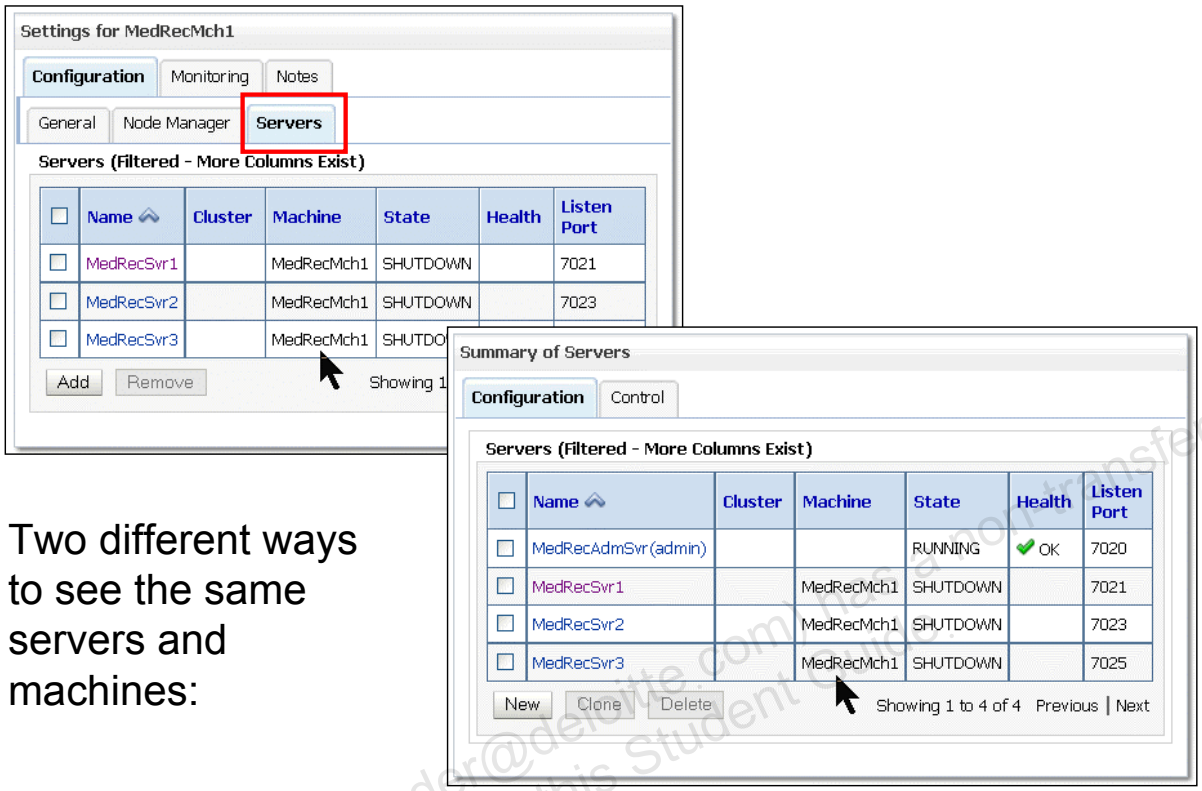
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Assigning Servers to a Machine

The screenshot shows how to add servers to a particular machine. You may want to assign a server to a machine so that a Node Manager can be used to start and stop it. A server need not belong to any machine, and a machine need not have any servers, though the typical relationship is that each server belongs to one machine. Because you select servers from the drop-down list, the next time you add servers to a machine, those servers will not show in the list of available servers to assign.

Monitoring Machines and Servers



Two different ways to see the same servers and machines:

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

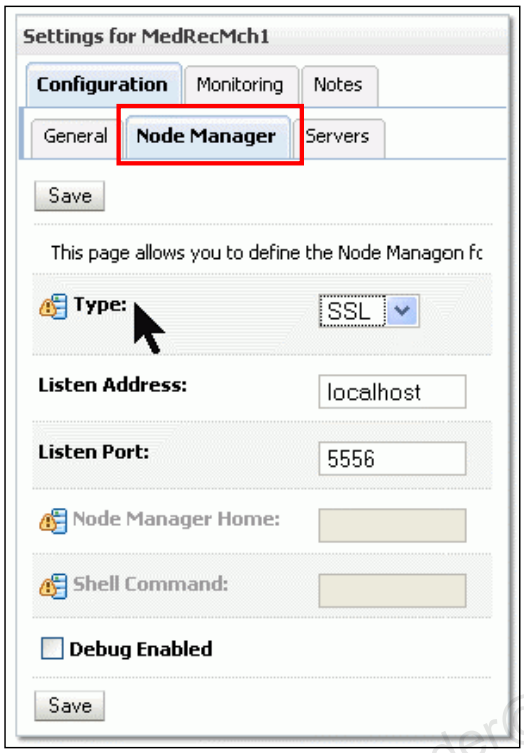
Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Monitoring Machines and Servers

This screenshot shows two different ways of viewing the same information. You may want to know which servers are on a particular machine, and which machine is associated with a server. The Administration Console provides two different ways to perform the same task. The top display is from the Machine list and the bottom display is from the Server list. Note that in this domain the Administration Server MedRecAdmSvr does not belong to a machine.

A Node Manager must run on each computer hosting the Oracle WebLogic Server instances that you want to control with a Node Manager. Configure each computer as a machine in Oracle WebLogic Server, and assign each server instance that you want to control with a Node Manager to the machine on which it runs.

Configuring a Machine to Use a Node Manager



A WLS machine resource maps a machine with the server instances that it hosts.

Choices for Type:

- Secure
  - SSH (`wlscontrol.sh`)
  - SSL (`startNodeManager`)
- Unsecure
  - Plain (`startNodeManager`)
  - RSH (`wlscontrol.sh`)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

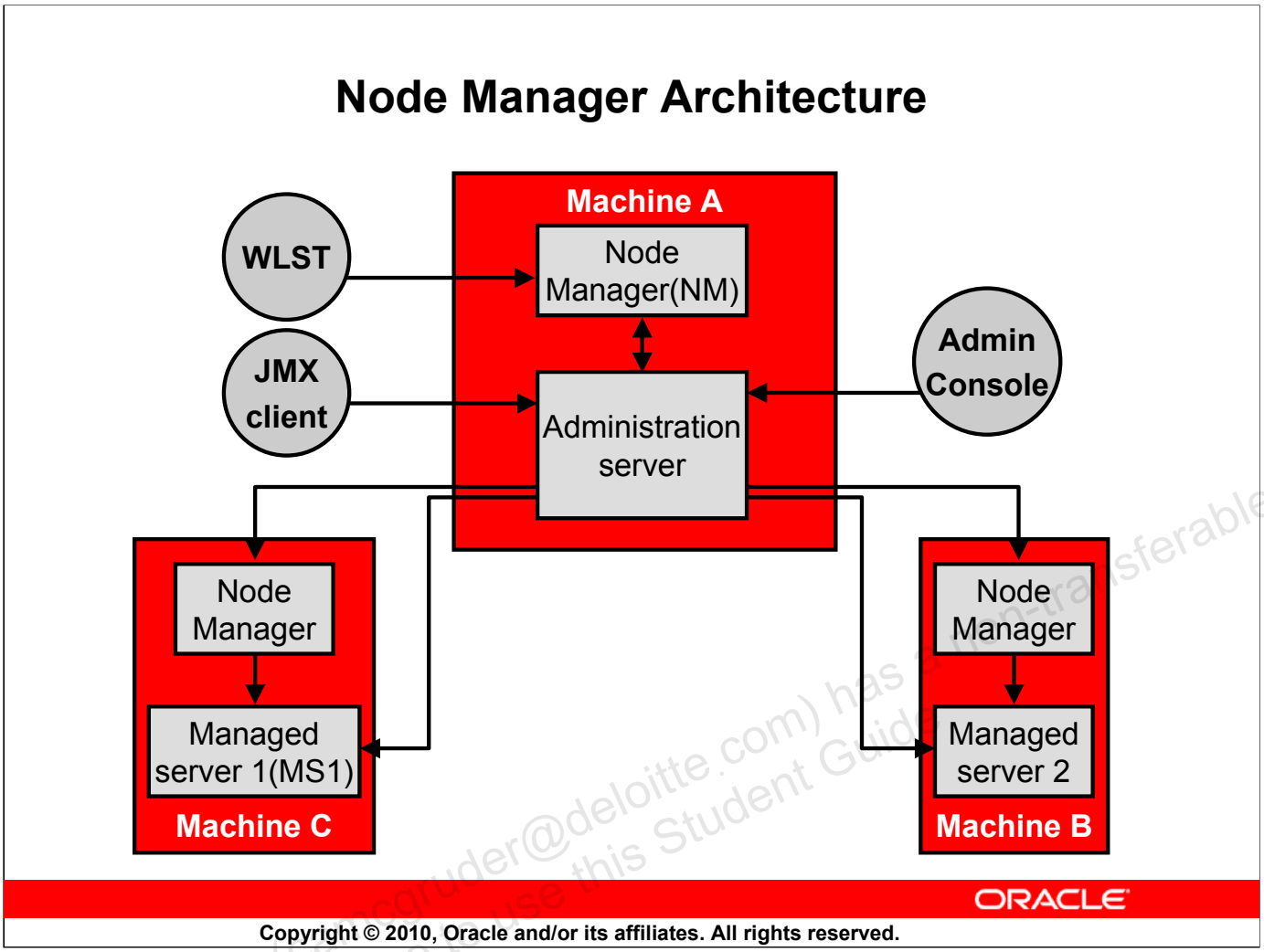
Configuring a Machine to Use a Node Manager

An Oracle WebLogic Server machine resource associates a particular machine with the server instances that it hosts and specifies the connection attributes for the Node Manager process on that system.

Configure a machine definition for each machine that runs a Node Manager process by using the Machines > Configuration > Node Manager page in the Administration Console. Enter the DNS name or IP address on which the Node Manager listens in the Listen Address field.

You enter `localhost` in the Listen Address field only if the administration and managed servers are on the same computer. You would most likely replace `localhost` with a real DNS name, such as `mywls.example.com` (in case the IP address changes), or a real address, such as 192.168.0.1. Listen Address must match the name in the `nodemanager.properties` file. Even if the host happens to be the local host, "localhost" does not work if the `nodemanager.properties` file has an actual host name in it. The best practice is to avoid using the default of `localhost`.

Plain and SSL communications uses Java, and therefore, can run on both Windows and UNIX. For UNIX, it calls the `startNodeManager.sh` script. For Windows, it calls the `startNodeManager.cmd` batch file. SSH and RSH are script-based, and therefore only run on UNIX platforms.



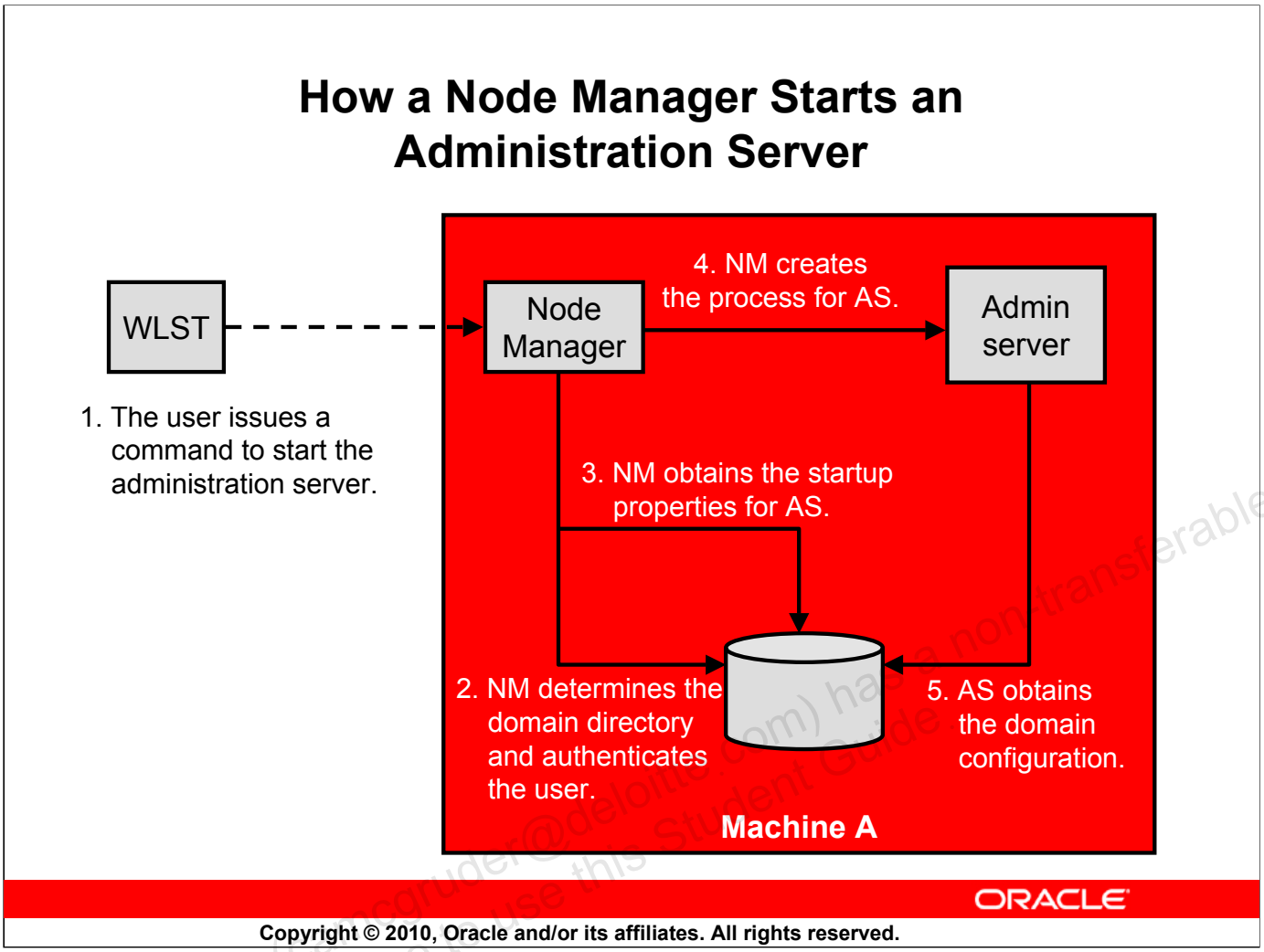
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Node Manager Architecture

The graphic illustrates the relationship between a Node Manager, its clients, and the server instances that it controls. A Node Manager client can be local or remote to the Node Managers with which it communicates. You can access either version of the Node Manager—the Java version or the script-based (SSH) version—from the following clients:

- **Administration Server:** Administration Console, from the Environments > Machines > Configuration > Node Manager page
- **JMX** utilities
- **WebLogic Scripting Tool (WLST) commands and scripts:** WLST offline serves as the Node Manager command-line interface that can run in the absence of a running administration server. You can use the WLST commands to start, stop, and monitor a server instance without connecting to an administration server. Starting the administration server is the main purpose of the stand-alone client. However, you can also use it to:
  - Stop a server instance that was started by a Node Manager
  - Start a managed server
  - Access the contents of a Node Manager log file
  - Obtain the server status for a server that was started with a Node Manager
  - Retrieve the contents of the server output log



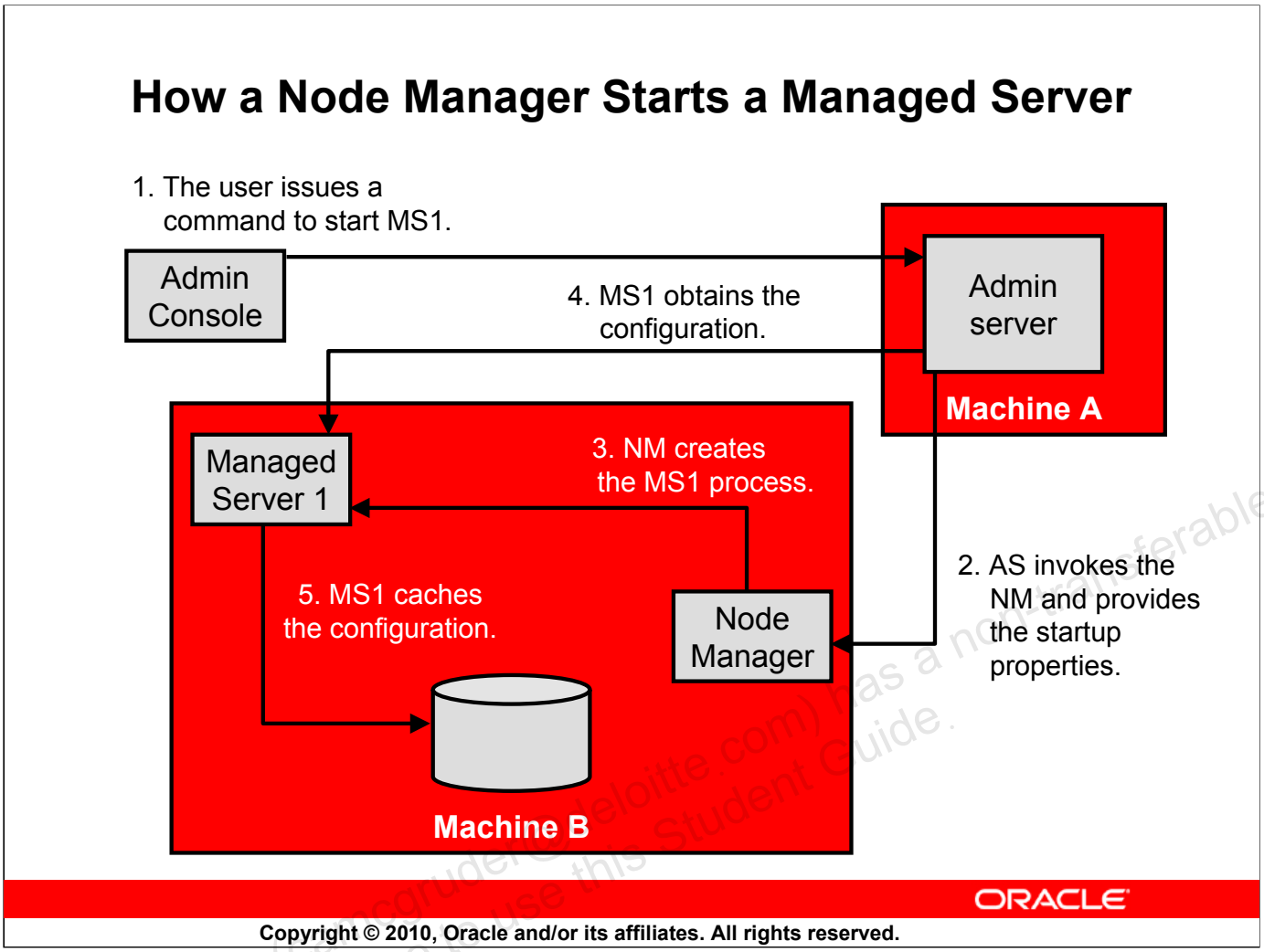
How a Node Manager Starts an Administration Server

The graphic in the slide illustrates the flow of a Node Manager starting an administration server.

1. An authorized user issues the WLST offline command, `nmConnect`, to connect to the Node Manager process on the machine that hosts the administration server. The authorized user then issues a command to start the administration server. (If the Node Manager instance is the SSH version, the user can connect using the SSH client.)
2. The `start` command identifies the domain and the server instance to start, and in the case of the Java Node Manager, provides the Node Manager username and password.
3. The Node Manager looks up the domain directory in `nodemanager.domains`, and authenticates the user credentials using a local file that contains the encrypted username and password.
4. The Node Manager creates the administration server process.
5. The administration server obtains the domain configuration from its `config` directory.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.



How a Node Manager Starts a Managed Server

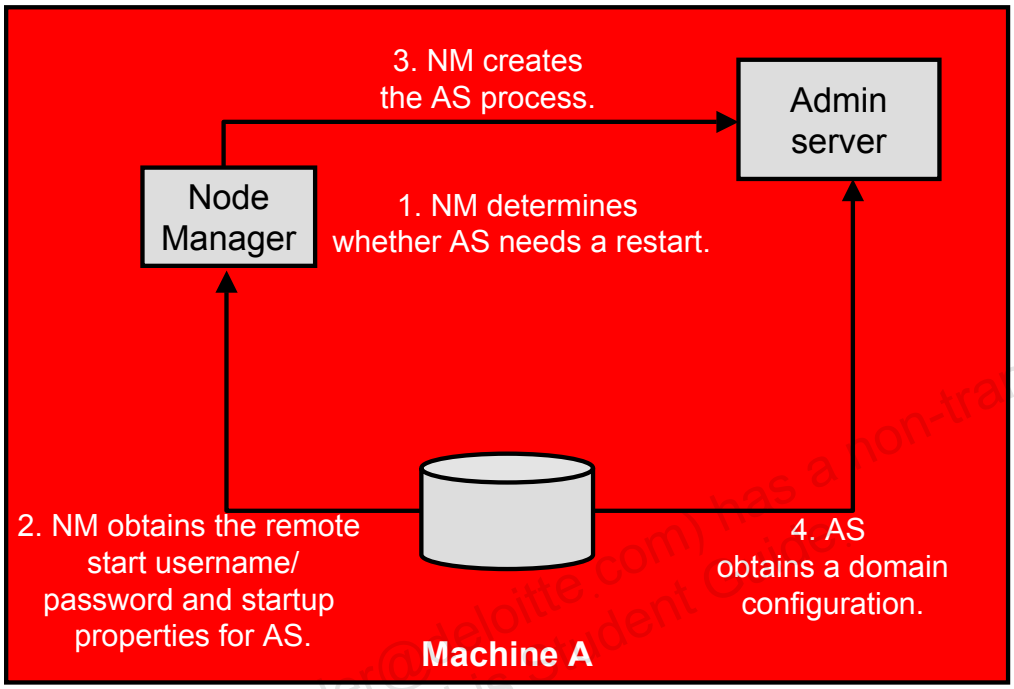
The graphic in the slide illustrates the flow of a Node Manager starting a managed server.

1. From the Administration Console, the user issues a `start` command for Managed Server 1.
2. The administration server issues a `start` command for Managed Server 1 to the Node Manager on Machine B, providing the remote start properties that are configured for Managed Server 1.
3. The Node Manager starts Managed Server 1. The Node Manager starts the managed server using the same root directory where the Node Manager process is running. To run the managed server in a different directory, set the Root Directory attribute on the Server > Configuration > Server Start Console page.
4. Managed Server 1 contacts the administration server to check for updates to its configuration information.
5. If there are outstanding changes to the domain configuration, Managed Server 1 updates its local cache of configuration data.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

How a Node Manager Restarts an Administration Server



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

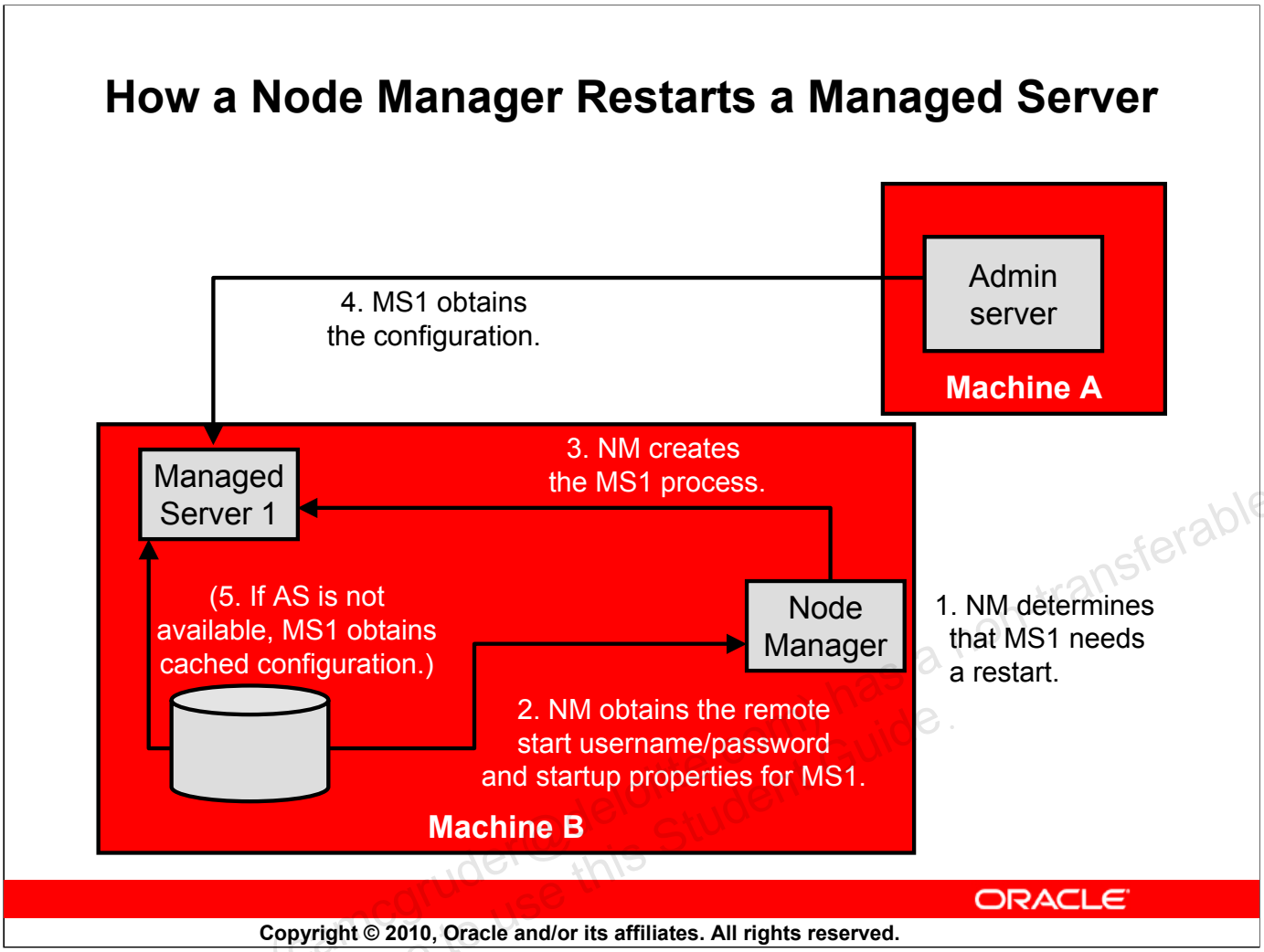
How a Node Manager Restarts an Administration Server

The graphic in the slide illustrates the flow of a Node Manager restarting an administration server.

1. The Node Manager determines from the administration server process exit code that it requires a restart. Note that no user is involved.
2. The Node Manager obtains the username and password for starting the administration server from the `boot.properties` file, and the server startup properties from the `server/security/startup.properties` file. These server-specific files are located in the `server` directory for the administration server.
3. The Node Manager starts the administration server.
4. The administration server reads its configuration data and starts up.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this student guide.



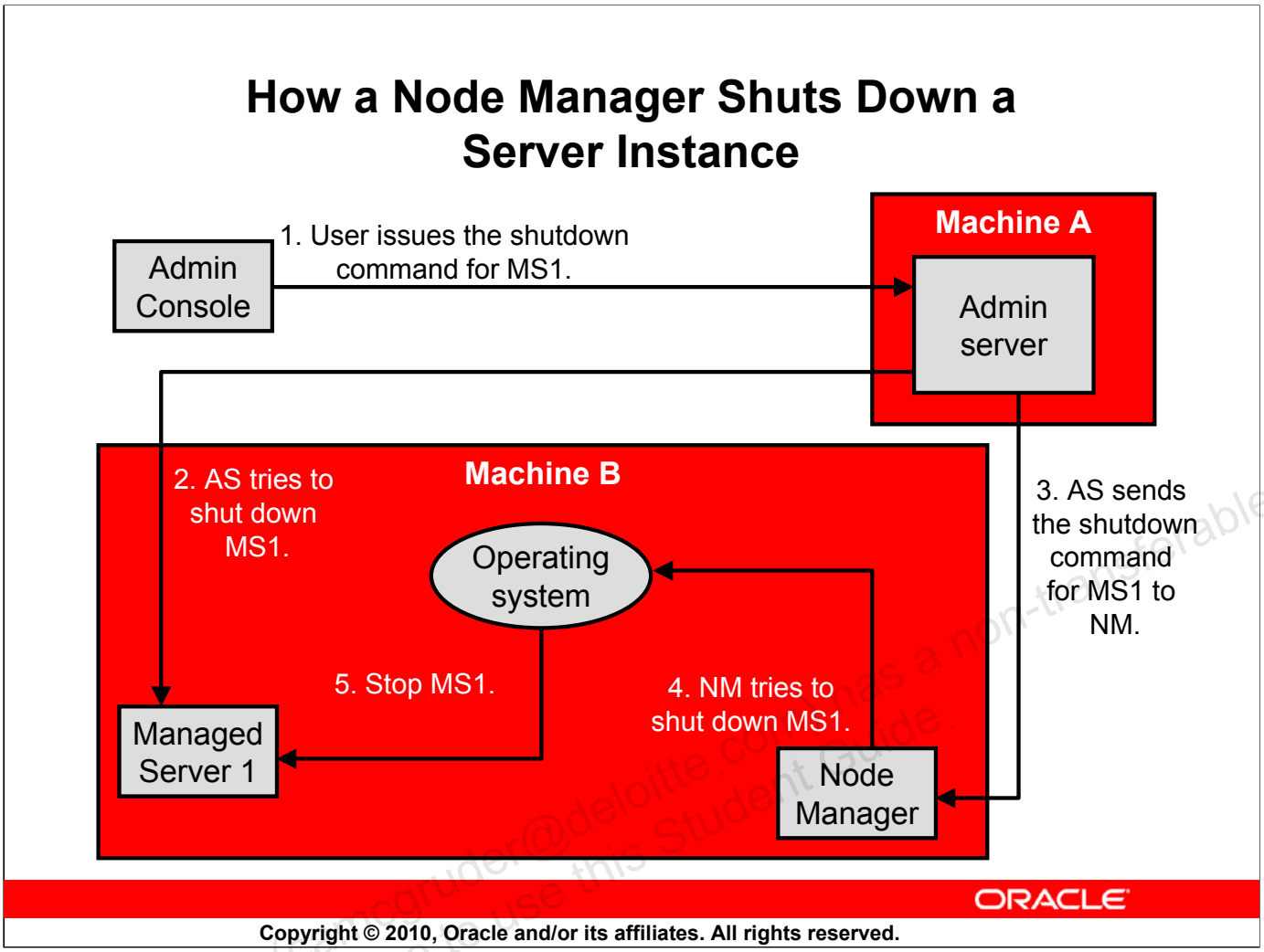
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

How a Node Manager Restarts a Managed Server

The graphic in the slide illustrates the flow of a Node Manager restarting a managed server.

1. The Node Manager determines from Managed Server 1's last known state that it requires a restart. Note that no user is involved.
2. The Node Manager obtains the username and password for starting Managed Server 1 from the `boot.properties` file, and the server startup properties from the `startup.properties` file. These server-specific files are located in the `server` directory for Managed Server 1.
3. The Node Manager starts the managed server.
   Note: The Node Manager waits `RestartDelaySeconds` after a server instance fails before attempting to restart it.
4. Managed Server 1 attempts to contact the administration server to check for updates to its configuration data. If it contacts the administration server and obtains updated configuration data, it updates its local cache in the `config` directory.
5. If Managed Server 1 fails to contact the administration server, and if MSI mode is enabled, Managed Server 1 uses its locally cached configuration data.



How a Node Manager Shuts Down a Server Instance

The graphic in the slide illustrates the flow of a Node Manager shutting down a server.

1. Through the Administration Console, an authorized user issues a shutdown command for Managed Server 1.
2. The administration server issues the shutdown command directly to Managed Server 1. If it successfully contacts Managed Server 1, Managed Server 1 performs the graceful shutdown sequence.
3. If in the previous step, the administration server fails to contact Managed Server 1, it issues a shutdown command for Managed Server 1 to the Node Manager on Machine B.
4. The Node Manager issues a request to the operating system to stop Managed Server 1.
5. The operating system ends the Managed Server 1 process.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Versions of Node Managers

- There are two versions of Node Managers:
  - Java-based Node Managers
  - Script-based Node Managers
- Java-based Node Managers run within a Java Virtual Machine (JVM) process.
- Script-based Node Managers (used only for UNIX systems) do not have as much security, but provide the ability to remotely manage servers over a network using Secure Shell (SSH).
- Node Managers are required for:
  - Whole server migration
  - Some configurations of automatic server migration

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Versions of Node Managers

Oracle WebLogic Server provides two versions of Node Managers: Java-based and script-based, with similar functionality. However, each version has a different configuration and security considerations.

The Java-based Node Managers run within JVM processes. It is recommended that you run Node Managers as Windows services on Windows platforms and as operating system services on UNIX platforms, allowing the Node Managers to restart automatically when the system is rebooted.

Oracle provides native Node Manager libraries for Windows, Solaris, HP-UX, Linux on Intel, Linux on Z-Series, and the AIX operating systems.

Note: Node Managers are not supported on Open VMS, OS/390, AS400, UnixWare, or Tru64 UNIX.

The Java-based version Node Manager determines its configuration from the `nodemanager.properties` file.

For UNIX systems, Oracle WebLogic Server provides a script-based version of the Node Manager. This script is based on UNIX shell scripts, but uses SSH for increased security. SSH uses user-ID based security.

Versions of Node Managers (continued)

The script-based version does not provide as much security as the Java-based version. However, the advantage of script-based Node Managers is that they can remotely manage servers over a network that has been configured to use SSH. No additional server installation is required. The scripts merely have to be copied to the remote machine. Although scripted, the Node Manager can be easy to configure in some cases, and its feature set does not contain some of the capabilities of a Java Node Manager. For those more familiar with configuring Java SSL than RSH or SSH, a Java-based Node Manager can be just as easy to configure.

Determining Which Node Manager Version to Use

- Automatic Server Migration is supported by both the scripted version of Node Managers and the Java version of Node Managers. In previous releases of WLS, only the scripted version supported automatic server migration.
- If you are installing Oracle WebLogic Server on a Windows system, you must use a Java version of Node Manager. The scripted version of a Node Manager is not supported on Windows.
- The script-based Node Manager requires a much simpler security configuration than the Java version. Remote Shell (RSH) and SSH are easier to configure than secure sockets layer (SSL), which is the method of security used by a Java version of Node Manager. The script version of a Node Manager also requires a smaller footprint than the Java version.
- The Java version of a Node Manager can be used with `inetd` on supported UNIX systems. `inetd` allows a Node Manager to be automatically restarted upon receiving a request on the configured port. The Java Node Manager is a running process that can also recover on its own in case of a full-machine shutdown. In the case where the Java Node Manager is restarted (automatically or manually), it will scan the domain directories for the servers it had previously managed or started and will automatically attempt to restart them. The scripted Node Manager is a passive process and is unable to do this.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGrudar (carl.mcgrudar@adelphia.com) has a non-transferable licence to use this Student Guide.

Road Map

- Node Managers
- Machines
- Configuring a Node Manager

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Node Manager Default Behaviors

- After Oracle WebLogic Server is installed, the Node Manager is “ready-to-run” if the Node Manager and the administration server are on the same machine.
- By default, the following behaviors are configured:
  - The Administration Console can use Node Manager to start the managed servers.
  - The Node Manager monitors the managed servers that it started.
  - The automatic restart of managed servers is enabled.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Node Manager Default Behaviors

The Node Manager is ready to run after Oracle WebLogic Server is installed if you run the Node Manager and the administration server on the same machine, and use the demonstration’s Secure Sockets Layer (SSL) configuration. By default, the following behaviors are configured:

- You can start a managed server using the Node Manager through the Administration Console.
- The Node Manager monitors the managed servers that it has started.
- Automatic restart of managed servers is enabled. The Node Manager restarts the server instances that it killed or that were killed by another method.

Configuring a Java-Based Node Manager

The configuration tasks for the Java-based Node Manager include:

- Reconfiguring the startup service for a Windows installation
- Daemonizing the Node Manager for UNIX systems
- Configuring the Java-based Node Manager security
- Reviewing `nodemanager.properties`
- Configuring the Node Manager on multiple machines

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring a Java-Based Node Manager

Except where noted, the configuration files apply to both Java-based and script-based Node Manager. The following files are located in `<WL_HOME>/common/nodemanager`.

- `nodemanager.properties`: This is the configuration file that is used by the Java-based version of the Node Manager.
- `nodemanager.domains`: This file contains mappings between the names of the domains managed by the Node Manager and their corresponding directories.
- `nm_data.properties`: This file stores the encryption data that the Node Manager uses as a symmetric encryption key. The data is stored in an encrypted form.

These files are located in other directories:

- `boot.properties`: The Node Manager uses this file to specify a boot identity when starting a server. This file is located in *domain_name*`/servers/`*server_name*`/data/nodemanager/`, not to be confused with the other `boot.properties` file under *domain_name*`/servers/`*server_name*`/security/`.
- `nm_password.properties`: This file stores a username/password pair that is specific to the Node Manager server that manages this domain. This is known as the Node Manager secret. The username and password are appended to a salt value (obtained from `SerializedSystemIni.dat` of the domain) and Secure Hash Algorithm (SHA) hashed. This file is located in *domain_name*`/config/nodemanager`.

Configuring a Java-Based Node Manager (continued)

If you start the WebLogic Administration Server with the `weblogic` user, then the `weblogic` user's password is not necessarily related to the Node Manager password. If you use the `weblogic` user to start the servers and are using a `boot.properties` file, changing the `weblogic` user's password will require re-creating all the `boot.properties` files. You can always create a different user and put that user in the Administrators group; then you can use that user to start the servers so changing the `weblogic` user's password would not require changing the `boot.properties` file.

The following file has been deprecated but you may still see it around:

- `nodemanager.hosts`: This file contained a list of all the trusted hosts that could issue commands to the Node Manager. It was located in `<WL_HOME>/common/nodemanager`.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Reconfiguring the Startup Service for a Windows Installation

1. Delete the Node Manager service using `uninstallNodeMgrSvc.cmd`.
2. Edit `installNodeMgrSvc.cmd` to specify the listen address and the listen port of the Node Manager.
3. Run `installNodeMgrSvc.cmd` to reinstall the Node Manager as a service, listening on the updated address and port.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Reconfiguring the Startup Service for a Windows Installation

The `<WL_HOME>/server/bin` directory (where `<WL_HOME>` is the top-level directory for the Oracle WebLogic Server installation) contains `uninstallNodeMgrSvc.cmd`, a script for uninstalling the Node Manager service, and `installNodeMgrSvc.cmd`, a script for installing the Node Manager as a service.

1. Delete the Node Manager service using `uninstallNodeMgrSvc.cmd`.
2. Edit `installNodeMgrSvc.cmd` to specify the listen address and the listen port of the Node Manager. Make the same edits to `uninstallNodeMgrSvc.cmd` as you make to `installNodeMgrSvc.cmd`, so that you can successfully uninstall the service in the future.
3. Run `installNodeMgrSvc.cmd` to reinstall the Node Manager as a service, listening on the updated address and port.

Node Manager as a UNIX Daemon

- Reinstall the Node Manager daemon.
- Configure the Node Manager using `nodemanager.properties`.
  - Reinstall the Node Manager daemon.

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Node Manager as a UNIX Daemon

Oracle WebLogic Server does not provide a command script for uninstalling and reinstalling the Node Manager daemon process. Refer to your operating system documentation for instructions about uninstalling existing daemons and setting up new ones.

1. Remove the Node Manager daemon process that the Oracle WebLogic Server installation process sets up.
2. At the command line, or in a script, reinstall the Node Manager daemon. You may want to view the contents of the `installNodeMgrSvc.cmd` file before setting up the new daemon. This cmd file is not installed on UNIX-based systems. Although this command file is Windows specific, it illustrates the following:
   - The key environment and local variables that must be defined
   - The validation steps that you might want to include in a script that installs the Node Manager as a daemon
   - The logic for setting default values for the listen address and port

Reviewing `nodemanager.properties`

- You can specify the properties for a Java-based Node Manager process either at the command line or in the `nodemanager.properties` file.
- Values supplied on the command line take precedence over those in the `nodemanager.properties` file.
- To configure a Node Manager to use the start/stop scripts, in the `nodemanager.properties` file:
  - Set the `StartScriptEnabled` property to `True` (default is `false`)
  - Set the `StopScriptEnabled` property to `True` (default is `false`)
  - Set the `StartScriptName` property to the name of your script (default is `startWebLogic.sh`)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@dbaints.com) has a non-transferable license to use this Student Guide.

Reviewing `nodemanager.properties`

The Node Manager properties define a variety of configuration settings for a Java-based Node Manager process. You can specify the Node Manager properties on the command line or define them in the `nodemanager.properties` file, which is created in the `<WL_HOME>/common/nodemanager` directory the first time that Node Manager starts up after the installation of Oracle WebLogic Server. Values supplied on the command line override the values in `nodemanager.properties`.

`nodemanager.properties` is created in the directory specified in `NodeManagerHome`, where `NodeManagerHome` is `<WL_HOME>/common/nodemanager`. If `NodeManagerHome` is not defined, `nodemanager.properties` is created in the current directory.

Each time you start a Node Manager, it looks for `nodemanager.properties` in the current directory, and creates the file if it does not exist in that directory. However, you cannot access the file until the Node Manager has started up once.

Reviewing `nodemanager.properties` (continued)

In many environments, the SSL-related properties in `nodemanager.properties` may be the only Node Manager properties that you must explicitly define. However, `nodemanager.properties` also contains non-SSL properties that you might need to specify, depending on your environment and preferences. For example:

- For a non-Windows installation, it might be appropriate to specify the `StartScriptEnabled` and `NativeVersionEnabled` properties.
- If a Node Manager runs on a multihomed system, and you want to control the address and port that it uses, define `ListenAddress` and `ListenPort`.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring a Script-Based Node Manager

- The SSH Node Manager is a shell script, `wlscontrol.sh`.
- An executable SSH client must reside on each machine where the Node Manager or the Node Manager client runs.
  - An SSH client is typically a standard part of a UNIX or Linux installation.
- The configuration tasks for a script-based Node Manager include:
  - Using SSH with the script-based Node Manager
  - Creating a Node Manager user
  - Configuring the script-based Node Manager security

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

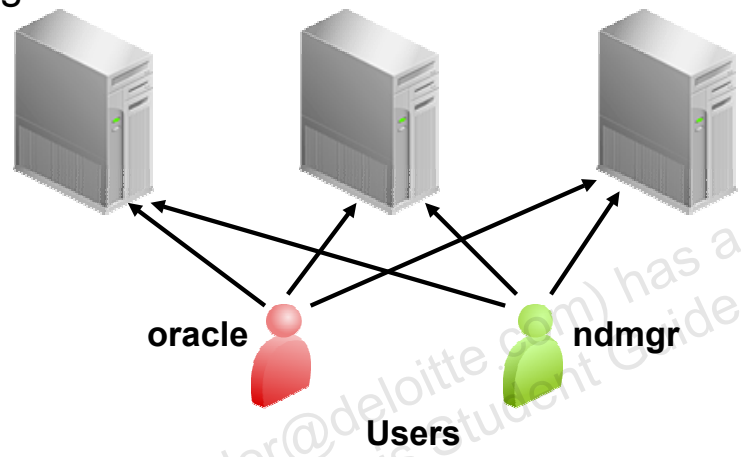
Configuring a Script-Based Node Manager

The SSH Node Manager is a shell script, `wlscontrol.sh`, located in `<WL_HOME>/common/bin/`. There are also copies of the script in each domain. The `wlscontrol.sh` file must exist on each machine that hosts the server instances that you want to control with the Node Manager. This script can be customized to meet site-specific requirements.

You must have an SSH client executable on each machine where the Node Manager or a Node Manager client runs. This script must also be in the path of the user ID that is running it. Typically, an SSH client is a standard part of a UNIX or Linux installation.

Creating Management OS Users

- Unless otherwise specified, the Node Manager runs as the user that started the domain.
- Before you run a Node Manager, you should create a dedicated UNIX user account for performing Node Manager functions.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating Management OS Users

Add the Node Manager dedicated user to all machines that will host the SSH Node Manager and to all machines that will host a Node Manager client, including the administration server.

For example:

- On each host machine, as the `root` user, create two new operating system (OS) users: **oracle** and **ndmgr**, both associated with a new group called **oracle**. There is nothing special about those two names, they could be any two dedicated names.
- Use **oracle** only for installing Oracle WebLogic Server.
- Use **ndmgr** to create an Oracle WebLogic Server domain, and start the administration server and remote managed servers using the Node Manager.
- Both OS users should have the same OS group (**oracle**) to ensure that the correct permissions are in place for **ndmgr** to run the WebLogic scripts and executables.

For example:

```
> groupadd oracle
> useradd -g oracle -m oracle
> passwd oracle
> useradd -g oracle -m ndmgr
> passwd ndmgr
```

Additional Configuration Information

Other Node Manager configuration tasks include:

- Configuring a machine to use a Node Manager
- Configuring the `nodemanager.domains` file
- Ensuring that the administration server address is defined
- Setting the Node Manager environment variables

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Additional Configuration Information

Each of these items is addressed in the following slides.

Configuring the `nodemanager.domains` File

- The `nodemanager.domains` file specifies the domains that a Node Manager instance controls.
- When a user issues a command for a domain, the Node Manager looks up the domain directory from this file.
- `nodemanager.domains` provides additional security by restricting the Node Manager client access to the domains listed in this file.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring the `nodemanager.domains` File

The `nodemanager.domains` file specifies the domains that a Node Manager instance controls. Thus, stand-alone clients do not need to specify the domain directory explicitly.

This file must contain an entry specifying the domain directory for each domain the Node Manager instance controls, in this form:

```
<domain-name>=<domain-directory>
```

Example:

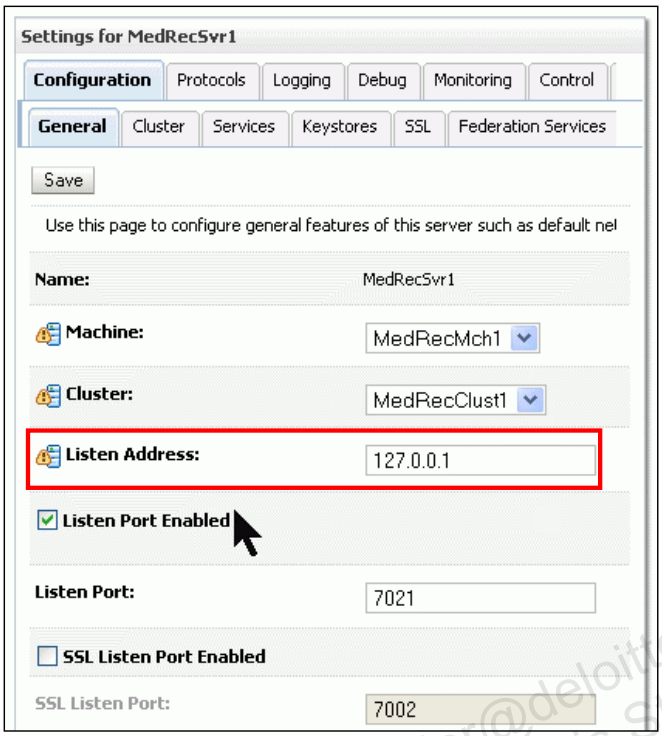
```
MedRecDomain=/u01/app/oracle/domains/MedRecDomain
```

When a user issues a command for a domain, the Node Manager looks up the domain directory from `nodemanager.domains`.

This file provides additional security by restricting the Node Manager client access to the domains listed in the file. The client can execute commands only for the domains listed in `nodemanager.domains`.

If you create your domain with the Configuration Wizard, the `nodemanager.domains` file is automatically created in `<WL_HOME>/common/nodemanager`. If necessary, you can manually edit `nodemanager.domains` to add a domain.

Defining the Administration Server Address



You must define a listen address for each administration server that connects to the Node Manager process.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Defining the Administration Server Address

If the listen address for an administration server is not defined, when a Node Manager starts a managed server, the managed server contacts the local host for its configuration information.

Set the listen address using the Servers > Configuration > General page in the Administration Console.

Either the listen port or the SSL listen port, or both must be enabled. This is necessary for Java-SSL Node Managers only to ensure the correct SSL handshake between the Node Manager and the administration server. If using SSL, remember that 127.0.0.1 is different from `localhost`, which is different from `wls-sysadm` when it comes to the certificates.

Setting Node Manager Environment Variables

Environment Variable	Description
JAVA_HOME	This is the root directory of the JDK that you are using for Node Managers. For example: `set JAVA_HOME=c:\oracle\jdk1.6.0_05` The Node Manager has the same JDK version requirements as Oracle WebLogic Server.
WL_HOME	This is the Oracle WebLogic Server installation directory. For example: `set WL_HOME=c:\oracle\wlserver_10.3`
PATH	You must include the Oracle WebLogic Server `bin` directory and path to your Java executable. For example: `set PATH=%WL_HOME%\server\bin;%JAVA_HOME%\bin;%PATH%`

LD_LIBRARY_PATH or SHLIB_PATH (UNIX only)	For UNIX systems, you must include the path to the native Node Manager libraries.
	Linux and Solaris example:
	`LD_LIBRARY_PATH:$WL_HOME/server/native/solaris:$WL_HOME/server/lib/solaris/oci816_8`
	AIX and HP-UX example:
	`SHLIB_PATH=$SHLIB_PATH:$WL_HOME/server/native/hpux11:$WL_HOME/server/lib/hpux11/oci816_8`

CLASSPATH	You can set the Node Manager `CLASSPATH` either as an option on the Java command line that is used to start the Node Manager or as an environment variable. Windows example: set `CLASSPATH=.;%WL_HOME%\server\lib\weblogic_sp.jar;%WL_HOME%\server\lib\weblogic.jar`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudden (cmccrudden@deloitte.com) has a non-transferable license to use this Student Guide.

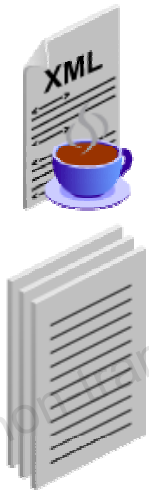
Setting Node Manager Environment Variables

You must set several environment variables before you start a Node Manager. You can set these variables manually on the command line or you can create a start script that sets them automatically. The two sample start scripts provided with Oracle WebLogic Server, `startNodeManager.cmd` and `startNodeManager.sh`, set the required variables.

Node Manager Configuration and Log Files

Two sets of files:

- The Node Manager `config` files, located in *DOMAIN_HOME*/`servers/server_name/data/nodemanager`
- The Node Manager `log` files, located in *DOMAIN_HOME*/`servers/server_name/logs` and `<WL_HOME>/common/nodemanager`



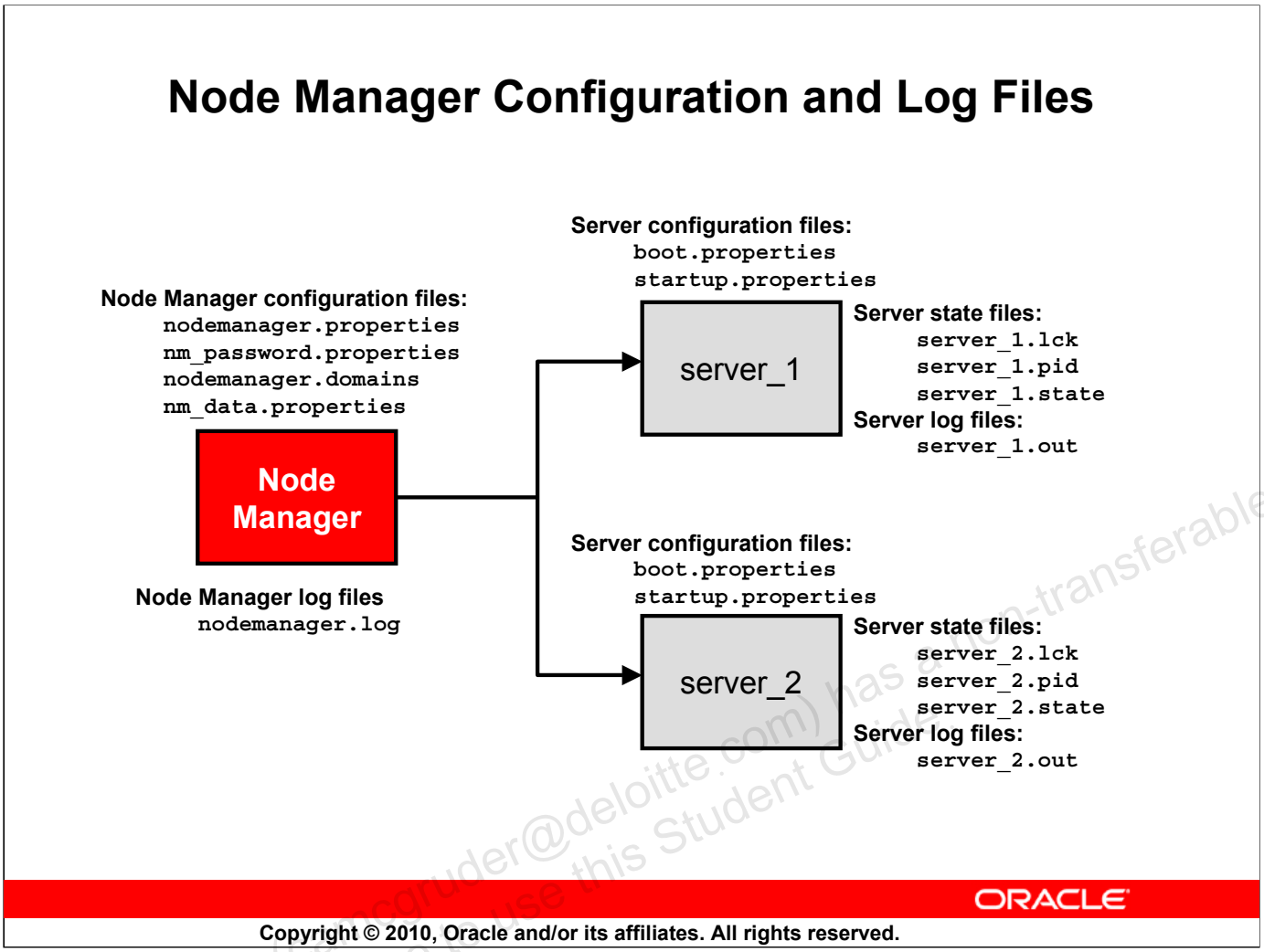
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Node Manager Configuration and Log Files

```
[root@wls-sysadm]# cd /u01/app/work/domains
[root@wls-sysadm]# cd MedRecDomain/servers/MedRecSvr1/data/nodemanager
[root@wls-sysadm]# ll
total 24
-rw-r--r-- 1 oracle oinstall  174 Feb  5 15:56 boot.properties
-rw-r--r-- 1 oracle oinstall    6 Feb  5 15:56 MedRecSvr1.lck
-rw-r--r-- 1 oracle oinstall    6 Feb  5 15:56 MedRecSvr1.pid
-rw-r--r-- 1 oracle oinstall   12 Feb  5 15:56 MedRecSvr1.state
-rw-r--r-- 1 oracle oinstall   22 Feb  5 15:56 MedRecSvr1.url
-rw-r--r-- 1 oracle oinstall 1052 Feb  5 15:56 startup.properties
[root@wls-sysadm]# cd
/u01/app/oracle/Middleware/11.1.1/wlserver_10.3/common/nodemanager/
[root@wls-sysadm]# ll
total 116
-rw-r--r-- 1 oracle oinstall   130 Feb  5 15:28 nm_data.properties
-rw-r----- 1 oracle oinstall   166 Feb  5 15:35 nodemanager.domains
-rw-r--r-- 1 oracle oinstall 69352 Feb  5 15:56 nodemanager.log
-rw-r--r-- 1 oracle oinstall     0 Feb  5 15:55 nodemanager.log.lck
-rw-r--r-- 1 oracle oinstall   900 Feb  5 15:55 nodemanager.properties
[root @wls-sysadm]#
```



Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Node Manager Configuration and Log Files (continued)

In managing multiple servers, a Node Manager uses multiple configuration files and outputs log files to multiple directories.

Configuration Files

Except where noted, the configuration files apply to both Java-based and script-based Node Managers.

- `nodemanager.properties`: This is the configuration file used by Java-based versions of the Node Manager. This file is located in `<WL_HOME>/common/nodemanager`, where `<WL_HOME>` is the location in which you installed Oracle WebLogic Server. This file needs to be updated manually for the lab.
- `nodemanager.domains`: This file contains mappings between the names of the domains managed by the Node Manager and their corresponding directories. This file is located in `<WL_HOME>/common/nodemanager`.
- `nm_data.properties`: This file stores the encryption data that the Node Manager uses as a symmetric encryption key. The data is stored in an encrypted form. This file is located in `<WL_HOME>/common/nodemanager`.
- `nm_password.properties`: This file stores the Node Manager username and password. This file is located in *`DOMAIN_HOME`*`/config/nodemanager`.

Node Manager Configuration and Log Files (continued)

- `boot.properties`: A Node Manager uses this file to specify the user credentials when starting a server. This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.
- `startup.properties`: Each managed server instance has its own `startup.properties` file with properties that control how the Node Manager starts up and controls the server.
  A Node Manager automatically creates this file by using the properties passed to it when the administration server was last used to start the server. This allows a Node Manager client or startup scripts to restart a managed server using the same properties that were last used by the administration server.
  These properties correspond to the server startup attributes that are contained in `ServerStartMBean` and the health-monitoring attributes in `ServerStartMBean`. This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.
- *server_name*.addr: *server_name*.addr stores the IP address that is added when a server starts or is migrated. This file is generated after the server IP address is successfully brought online during migration. *server_name*.addr is deleted when the IP address is brought offline.
  The server IP address is used to validate remove requests to prevent addresses from being erroneously removed while shutting down the server.
  This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.
- *server_name*.lck: *server_name*.lck is generated by each server and contains an internally used lock ID. This file is located in DOMAIN_HOME/servers/*server_name*/data/nodemanager.
- *server_name*.pid: server_name.pid is generated by each server and contains the process ID of the server. The Node Manager checks the process ID generated by the server during crash recovery. This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.
- *server_name*.state: server_name.state is generated by the server and contains the server's current state. The Node Manager monitors the contents of this file to determine the current state of the server.
  Note: Do not delete or alter this file. Without this file, the Node Manager cannot determine the current state of the server.
  This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCurdy (carl.mccurdy@oracle.com) has a non-transferable license to use this Student Guide.

Node Manager Configuration and Log Files (continued)

Log Files

Use the Node Manager and Oracle WebLogic Server log files to help troubleshoot problems in starting or stopping individual managed servers.

- **`nodemanager.log`:** A Node Manager creates a log file that is located in `NodeManagerHome/nodemanager.log`. This log file stores data about all the domains administered by Node Manager.
  This log file is generated by a Node Manager and contains data for all the domains that are controlled by a Node Manager on a given physical machine. This file is located in `<WL_HOME>/common/nodemanager`.
  The log output is appended to the current `nodemanager.log`. Log rotation is disabled by default, but can be enabled by setting `LogCount` in `nodemanager.properties`.
  You can view the Node Manager log file by:
  - Opening the Machines Monitoring Node Manager Log page in the Administration Console
  - Using the WLST `nmLog` command
- **`server_name.out`:** For each server instance that it controls, a Node Manager maintains a log file that contains the `stdout` and `stderr` messages generated by the server instance. If the remote start debug property is enabled as a remote start property for the server instance, or if the Node Manager `debug` property is enabled, the Node Manager includes additional debug information in the server output log information.
  Note: You cannot limit the size of the log files that a Node Manager creates. Logging to `stdout` is disabled by default. This file is located in *`domain_name`*`/servers/<`*`server_name`*`>/logs`.
  A Node Manager creates the server output log for a server instance in the server instance's `logs` directory, with the name, `server_name.out`, where *`server_name`* is the name of the server instance.
  You can view the Node Manager log file for a particular server instance by:
  - Selecting Diagnostics Log Files
  - Using the WLST `nmServerLog` command

  There is no limit to the number of server output logs that a Node Manager can create.

Oracle WebLogic Server Log Files

A server instance that is under the control of a Node Manager has its own log file, in addition to the log file created by the Node Manager.

You can view the log file for a server instance by selecting Diagnostics Log Files, selecting the server log file, and clicking View.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCollum (carcmccollum@gmail.com) has a non-transferable license to use this Student Guide.

Quiz

You can start a managed server using WLST and without using a Node Manager.

a. True

b. False

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

If you do not use a Node Manager, WLST cannot start managed servers. This method is available only to start the administration server.

Quiz

Which of the following statements is true?

a. There is one Node Manager for each machine.
b. There is one Node Manager for each domain.
e. There is one Node Manager for each cluster.

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

Although optional, there is no more than one Node Manager associated with a machine (host). The administration and managed servers on a single machine are managed by one Node Manager.

Quiz

To start a managed server using the Administration Console, a Node Manager must be configured on the machine where the managed server resides.

a. True

b. False

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

A managed server is assigned to a machine. A Node Manager must be configured on the machine and started in order to start the managed server using the Administration Console.

Summary

In this lesson, you should have learned how to:

- Configure machines
- Use a Node Manager
- Monitor domains and servers

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 9 Overview: Configuring Machines and Node Managers

This practice covers the following topics:

- Configuring and running Node Managers
- Creating a WebLogic machine using the Administration Console
- Configuring a machine
- Assigning managed servers to machines using the Administration Console
- Starting and stopping managed servers using the Administration Console and WLST

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 9 Overview: Configuring Machines and Node Managers

See Appendix A for the complete steps to do the practice.

10

Viewing and Managing Logs in Oracle WLS Environment

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Define and configure server and domain logs
- View and interpret the format of domain and server log files using the Administration Console
- Configure server standard output settings using the console
- Describe how applications can integrate with WLS logging infrastructure (Apache commons, log4j)
- Access online log message catalogs
- Create and apply a log filter using the console
- Configure log filter expressions

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

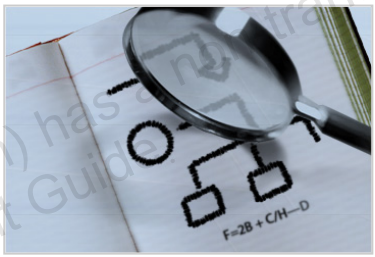
Objectives

Scenario

After the domain is running smoothly, you do not need to see all the informational messages, just the errors. Also, you want to save and rotate those logs on an automated basis. However, one server is acting in a peculiar fashion, so you do want to get an extra level of logging from just that one server. You need to configure logging to ensure that the logs have been managed as desired.

Road Map

- Logs and monitoring
  - Using log files
  - Integrating application logging
  - Logging file format
- Log filters



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle WebLogic Server Logs

Logs can aid in the discovery of:

- Any problems encountered while servicing requests
- Activity by day and time interval
- The IP addresses of users accessing an application
- Frequently accessed resources
- The amount of data sent and received

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Oracle WebLogic Server Logs

Oracle WebLogic Server logging services provide facilities for writing, viewing, filtering, and listening for log messages. These log messages are generated by the Oracle WebLogic Server instances, subsystems, and Java EE applications that run on Oracle WebLogic Server or on client JVMs.

Oracle WebLogic Server subsystems use logging services to provide information about events such as the deployment of new applications or the failure of one or more subsystems. A server instance uses them to communicate its status and respond to specific events. Debugging can also be enabled on individual subsystems to include additional life cycle.

The main subsystems that use logging services are:

- Server
- HTTP
- JTS
- JMS
- JDBC

Oracle WebLogic Server can keep a log of all HTTP transactions in a text file, named `access.log` by default. This file can also be used to determine which resources were accessed more often. This file can also track the amount of data being received by and sent back to HTTP clients.

Oracle WebLogic Server Logs (continued)

Oracle WebLogic Server provides a hierarchical Logger tree that lets you specify the Severity level for:

- Generated message catalog Logger classes from the XML I18N catalog using `weblogic.i18ngen` (Use the `weblogic.i18ngen` utility to parse message catalogs [XML files] to produce `Logger` and `TextFormatter` classes used for localizing the text in log messages.)
- Instances of the Commons Logging APIs when the Oracle WebLogic Server implementation of the commons `org.apache.commons.logging.LogFactory` interface is enabled

All Loggers inherit their Severity level from the nearest parent in the tree. You can, however, explicitly set the Severity level of a Logger, thereby overriding the level that is set for the nearest parent. You can set the Severity level for loggers from the Administration Console, WLST, or the command line.

The LogMBean interface offers two new attributes:

- `LoggerSeverity`
- `LoggerSeverityProperties`

Note

Logging in Oracle WebLogic Server 10.3 now provides finer control of Logging Severity, down to the level of the logging source that is generating the message. This is provided via a set of severities that are defined in the `weblogic.logging.Severities` class.

Log files have no performance indicators.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Server and Domain Logs

- A server log:
  - Logs all activity for a single server
  - Is stored in *SERVER_NAME*/logs/*SERVER_NAME*.log by default
- A domain log:
  - Logs key events for all servers in a domain
  - Is stored in *ADMIN_SERVER_NAME*/logs/*DOMAIN_NAME*.log by default
- Other logs:
  - HTTP
  - JMS
  - JDBC
- These logs are independently configured.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Can MCGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Server and Domain Logs

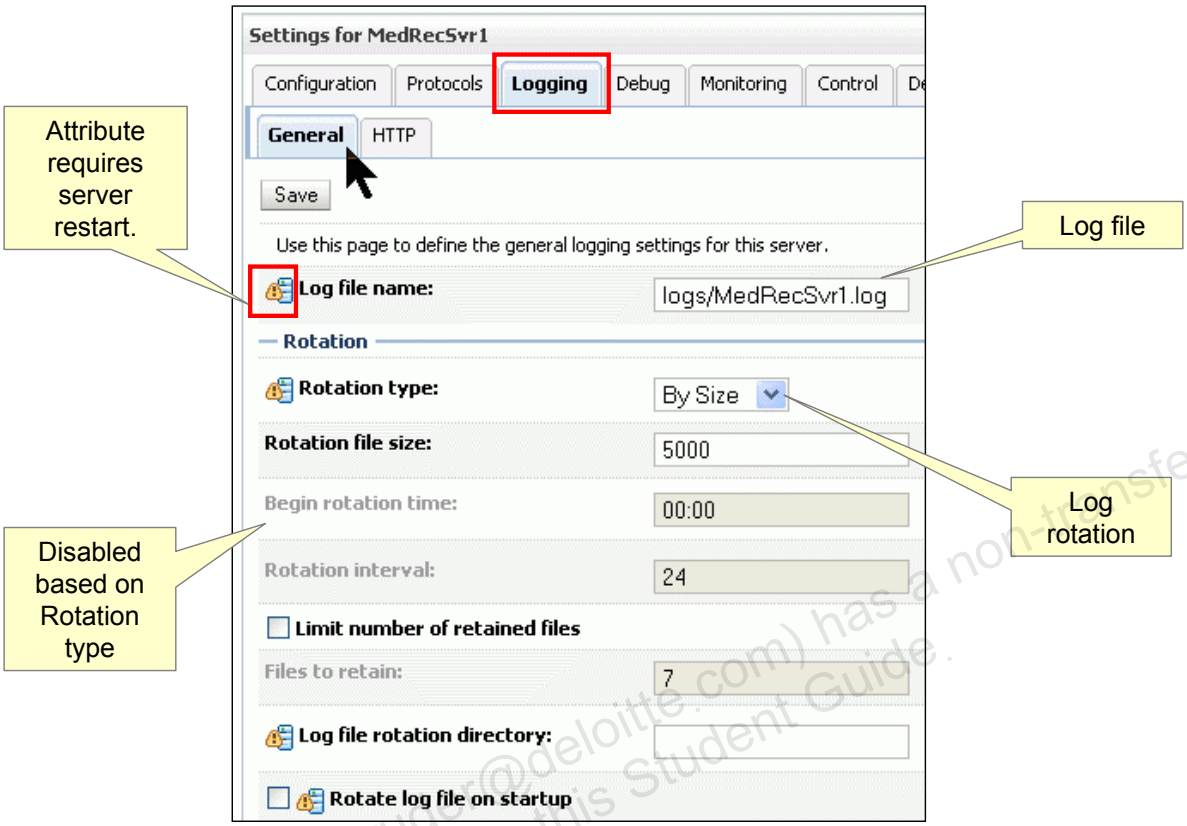
Each Oracle WebLogic Server instance writes all the messages from its subsystems and applications to a server log file that is located on the local host computer. By default, the server log file is located in the `logs` directory below the server instance root directory—for example, *DOMAIN_NAME*/servers/*SERVER_NAME*/logs/*SERVER_NAME*.log, where *DOMAIN_NAME* is the name of the directory in which you created the domain and *SERVER_NAME* is the name of the server.

In addition to writing messages to the server log file, each server instance forwards a subset of its messages to a domainwide log file. By default, servers forward only messages of the NOTICE severity level or higher. Although you can modify the set of messages that are forwarded, servers can never forward messages of the DEBUG severity level.

The domain log file provides a central location from which to view the overall status of the domain. The domain log resides in the administration server `logs` directory. The default name and location for the domain log file is *DOMAIN_NAME*/servers/*ADMIN_SERVER_NAME*/logs/*DOMAIN_NAME*.log, where *DOMAIN_NAME* is the name of the directory in which you created the domain and *ADMIN_SERVER_NAME* is the name of the administration server.

The server log messages and the log file communicate events and conditions that affect the operation of the server or the application. Some subsystems can also be configured to maintain additional log files to provide an audit of the subsystem's interactions under normal operating conditions.

Configuring Server Logging



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring Server Logging

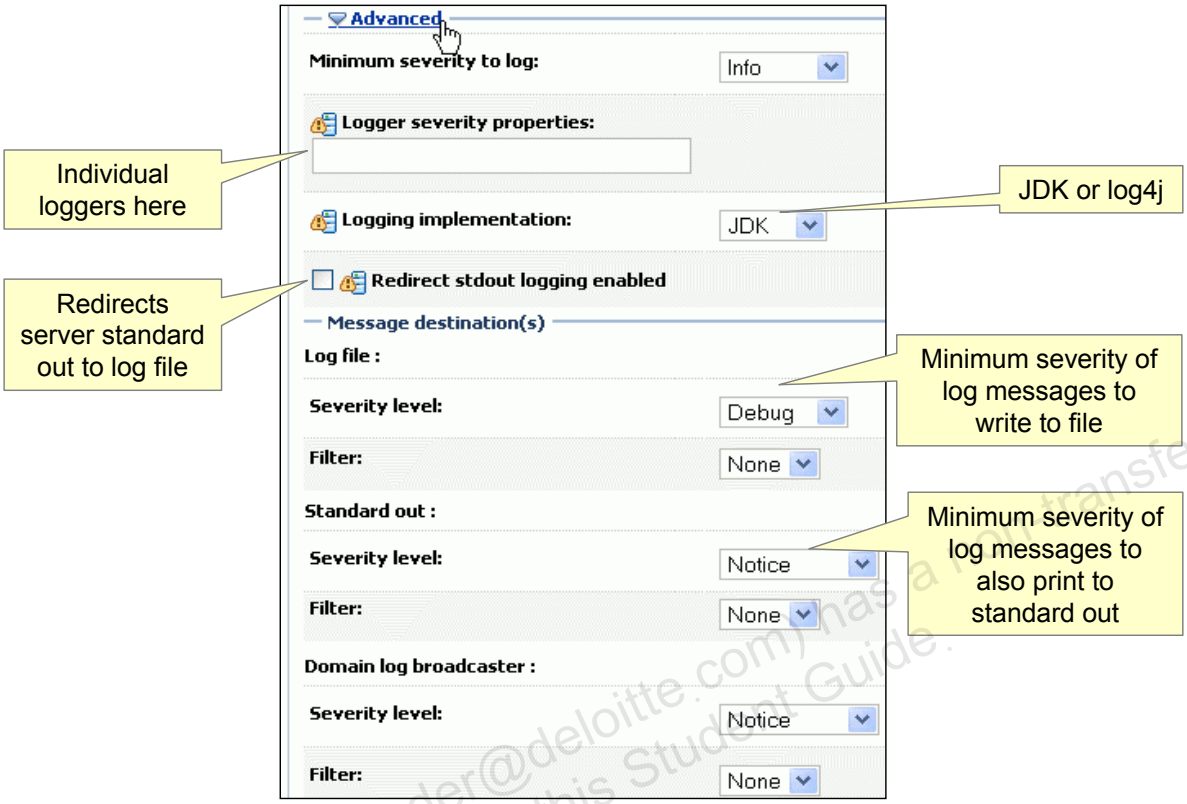
In the left pane of the console, expand **Environment** and select **Servers**.

In the Servers table, click the name of the server instance whose logging you want to configure.

Select Logging > General. The available options include:

- **Log file name:** The name of the file that stores the current log messages. Usually, it is a computed value based on the name of the parent of this MBean. For example, for a server log, it is `logs/`*`SERVER_NAME`*`.log`. However, if the name of the parent cannot be obtained, the file name is `weblogic.log`. If you specify a relative path name, it is interpreted as relative to the server's root directory.
- **Rotation type**
  - **None:** Messages accumulate in a single file. You must erase the contents of the file when the size is too big. Note that Oracle WebLogic Server sets a threshold size limit of 500 MB before it forces a hard rotation to prevent excessive log file growth.
  - **By Size:** When the log file reaches the size that you specify in "Rotation file size," the server renames the file `FileName.n`.
  - **By Time:** At each time interval that you specify in "Begin rotation time" and "Rotation interval," the server renames the file `FileName.n`.
- **Limit number of retained files:** After the server reaches this limit, it deletes the oldest log file and creates a new log file with the latest suffix.

Configuring Server Logging: Advanced



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Configuring Server Logging: Advanced

- **Logger severity properties:** Specify the severity level for any specific loggers for which you want to override the setting for the root Logger or its closest parent node in the logger tree. You can also specify severity levels for packages (if using the Commons Logging API) or for individual WebLogic Server subsystem Loggers (if using the Message Catalog Logger). For WebLogic Server components, the subsystem name is the logger name.
- **Logging implementation:** Specifies whether the server logging is based on a log4j implementation. By default, Oracle WebLogic Server logging uses an implementation based on the Java Logging APIs that are part of the Java Development Kit (JDK).
- **Severity level:** Specifies the minimum severity of log messages that are to be written to the server log file. By default, all messages go to the log file.
- **Filter:** Specifies the filter configuration for the server log file. A filter configuration defines simple filtering rules to limit the volume of log messages written to the log file.
- **Redirect stdout logging enabled:** When enabled, redirects the standard out of the JVM in which an Oracle WebLogic Server instance runs to the WebLogic logging system
- **Stdout Severity Level:** Specifies the minimum severity of log messages going to standard out. Messages with a severity lower than the specified value are not published to standard out.

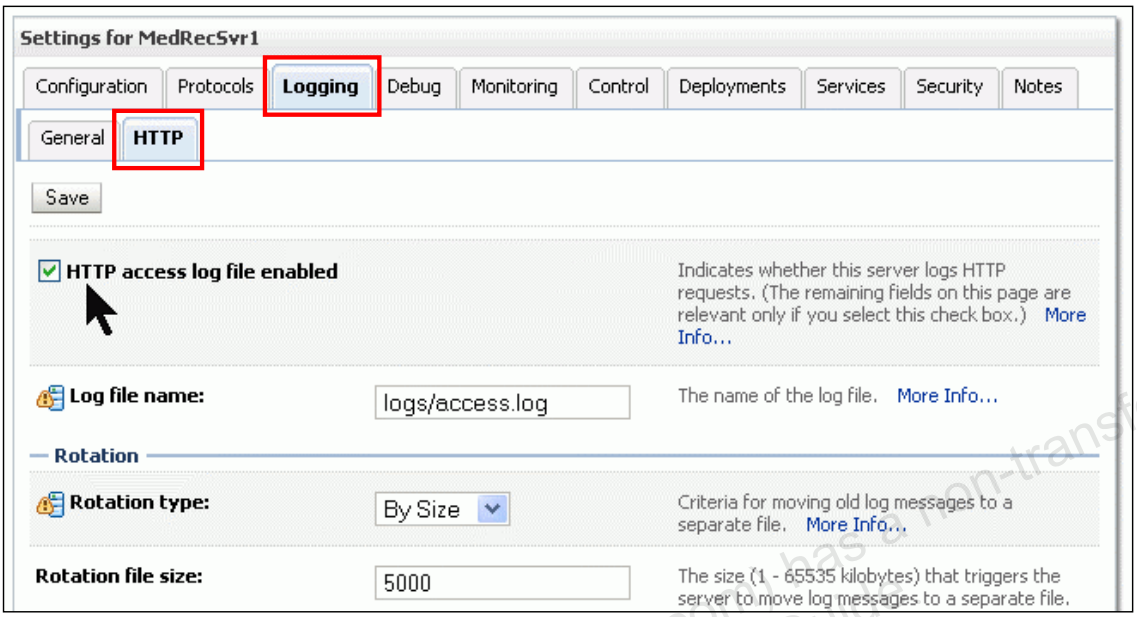
Configuring Server Logging: Advanced (continued)

- **Domain Log Severity Level:** Specifies the minimum severity of log messages going to the domain log from this server's log broadcaster. Messages with a severity lower than the specified value are not published to the domain log.
- **Buffer size:** Specifies the size of the buffer for the log messages that are sent to the domain log as a batch. The buffer is maintained on the managed server and is broadcast to the domain log when it gets full.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

HTTP Access Logs



The other options (rotation, number of retained files, and so on) are very similar to the other server logs.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

HTTP Access Logs

The screenshot shows the HTTP logging configuration options. Use this page to configure HTTP logging for the server. By default, HTTP logging is enabled and the server saves HTTP requests in a separate log file; it does not store HTTP requests in the server log file or the domain log file. If you use another HTTP server such as Oracle HTTP Server or Apache, disable the logging here.

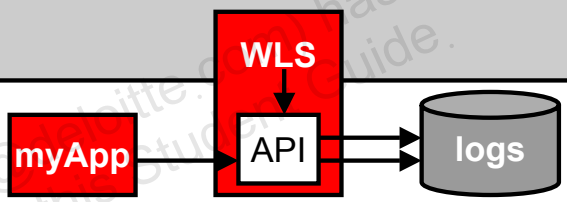
A sample of the HTTP access log file:

```
[user@wls-sysadm]$ cd /u01/app/oracle/domains/MedRecDomain/servers/MedRecSvr3/logs
[user@wls-sysadm]$ more access.log
  •   192.168.0.1 - - [12/May/2009:21:33:47] "GET /benefits HTTP/1.1" 302 259
  •   192.168.0.1 - - [12/May/2009:21:33:47] "GET /benefits/ HTTP/1.1" 200 5813
  •   192.168.0.1 - - [12/May/2009:21:33:51] "POST /benefits/servlet HTTP/1.1" 200 681
  •   192.168.0.1 - - [12/May/2009:21:33:53] "GET /benefits/welcome.html HTTP/1.1" 200 5813
  •   192.168.0.1 - - [12/May/2009:21:34:42] "GET /benefits HTTP/1.1" 302 259
  •   192.168.0.1 - - [12/May/2009:21:34:44] "POST /benefits/servlet HTTP/1.1" 200 265
```

Apache Commons Logging API

```
import org.apache.commons.logging.LogFactory;
import org.apache.commons.logging.Log;

public class MyCommonsTest {
  public void testWLSCommonsLogging() {
    System.setProperty(LogFactory.FACTORY_PROPERTY,
      "weblogic.logging.commons.LogFactoryImpl");
    Log clog =
    LogFactory.getFactory().getInstance("MyCommonsLogger");
    // Log String objects
    clog.debug("Hey this is common debug");
    clog.fatal("Hey this is common fatal", new Exception());
    clog.error("Hey this is common error", new Exception());
    clog.trace("Dont leave your footprints on the sands of
    time");
  }
}
```



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Apache Commons Logging API

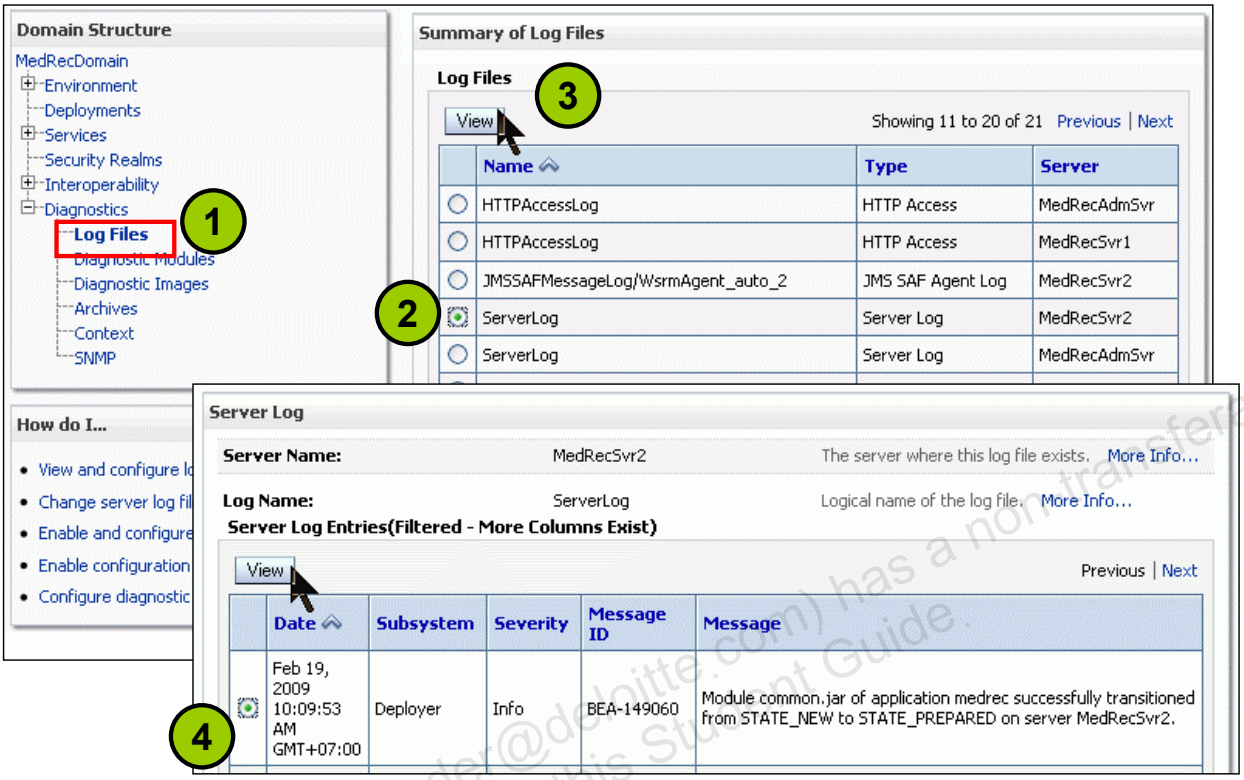
Application developers who want to use the WebLogic Server message catalogs and logging services as a way for their applications to produce log messages must know XML and the Java APIs. Many developers and system administrators use log4j, which is a predecessor to the Java Logging APIs. Log4j is an open source tool developed for putting log statements in your application. The log4j Java logging facility was developed by the Jakarta Project of the Apache Foundation.

The Jakarta Commons Logging APIs provide an abstraction layer that insulates users from the underlying logging implementation, which can be log4j or Java Logging APIs. WebLogic Server provides an implementation of the Commons LogFactory interface, letting you issue requests to the server Logger using this API. When developers use this, their applications can issue messages that are sent to wherever WebLogic is sending its messages (either JDK or log4j), so your application logs appear integrated with the WebLogic Server logs.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using the Console to View Logs



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Using the Console to View Logs

1. In the left pane of the console, expand Diagnostics and select Log Files.
2. In the Log Files table, select the option button next to the name of the log that you want to view.
3. Click View.
4. The page displays the latest contents of the log file—up to 500 messages in reverse chronological order. The messages at the top of the window are the most recent messages that the server has generated. Optionally, select the option button next to any log message and click View to see its full details.

The log viewer does not display messages that have been rotated into the archive log files.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using WLST to View Logs

```
wls:/offline> exportDiagnosticData (logicalName='ServerLog',
    logName='myserver.log', exportFileName='myExport.xml')

Input parameters: {logicalName='ServerLog',
    logName='myserver.log', logRotationDir='.',
    storeDir='../data/store/diagnostics', query='',
    exportFileName='myExport.xml', elfFields='',
    beginTimestamp=0L, endTimestamp=9223372036854775807L}

Exporting diagnostic data to myExport.xml ...
<Apr 3, 2010 11:23:56 AM EDT> <Info> <Store> <BEA-280050>
    <Persistent store "WLS_DIAGNOSTICS" opened:
    directory="/u01/app/oracle/Middleware/11.1.1/wlserver_10.3/
    server/data/store/diagnostics" writePolicy="Disabled"
    blockSize=512 directIO=false driver="wlfileio2">

Exported diagnostic data successfully.

wls:/offline>
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (ccmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Using WLST to View Logs

The example in the slide shows a client-side query against a local log file of a given name. The output is shown:

```
<?xml version='1.0' encoding='utf-8'?>
<DiagnosticData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" (…etc…) >
  <DataInfo>
    <ColumnInfo><Name>RECORDID</Name><Type>java.lang.Long</Type></ColumnInfo>
    <ColumnInfo><Name>DATE</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>SEVERITY</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>SUBSYSTEM</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>MACHINE</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>SERVER</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>THREAD</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>USERID</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>TXID</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>CONTEXTID</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>TIMESTAMP</Name><Type>java.lang.Long</Type></ColumnInfo>
    <ColumnInfo><Name>MSGID</Name><Type>java.lang.String</Type></ColumnInfo>
    <ColumnInfo><Name>MESSAGE</Name><Type>java.lang.String</Type></ColumnInfo>
  </DataInfo>
</DiagnosticData>
```

This can also be executed server side using the following syntax:

```
wls:/mydomain/serverRuntime> exportDiagnosticDataFromServer
(logicalName="HTTPAccessLog", exportFileName="myExport.xml")
```

Message Attributes

Attribute	Description	Standard Out?
Timestamp	The time and date when the message originated, in a format that is specific to the locale	✓
Subsystem	The particular WLS subsystem that was the source of the message (Management, Security, EJB, RMI, JMS, and so on)	✓
Severity	The degree of impact or seriousness of the event reported by the message	✓
Catalog ID	The unique ID assigned to this type of event, to reference in the online documentation	✓
Server Name	The WebLogic instance that generated the message	
Thread ID	The server thread that generated the message	
User ID	The current security context, if any	
Transaction ID	The current XA transaction context, if any	

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Message Attributes

When an Oracle WebLogic Server instance writes a message to the server log file, the first line of each message begins with #### followed by the message attributes. Each attribute is contained between angle brackets. The following is an example of a message in the server log file:

```
####<Sept 22, 2009 10:46:51 AM EST> <Notice> <WebLogicServer>
<MyComputer> <examplesServer> <main> <<WLS Kernel>> <> <null>
<1080575211904> <BEA-000360> <Server started in RUNNING mode>
```

In this example, the message attributes are Locale-formatted Timestamp, Severity, Subsystem, Machine Name, Server Name, Thread ID, User ID, Transaction ID, Diagnostic Context ID, Raw Time Value, Message ID, and Message Text.

If a message is not logged within the context of a transaction, the angle brackets for the Transaction ID are present even though no Transaction ID is present. If the message includes a stack trace, the stack trace is included in the message text.

When an Oracle WebLogic Server instance writes a message to standard out, the output does not include the #### prefix and does not include the Server Name, Machine Name, Thread ID, User ID, Transaction ID, Diagnostic Context ID, and the Raw Time Value fields:

```
<Sept 22, 2009 10:51:10 AM EST> <Notice> <WebLogicServer> <BEA-000360>
<Server started in RUNNING mode>
```

Message Severity

Severity	Description	Domain Log (by default)?
TRACE	Messages from the diagnostics framework	
DEBUG	Detailed internal messages (if debugging is enabled)	
INFO	Normal operations	
NOTICE	INFO message of greater importance	✓
WARNING	Suspicious operation or configuration	✓
ERROR	Error handling request, but no interruption in service	✓
CRITICAL	System or service error that may cause temporary loss or degradation of service	✓
ALERT	One or more services in an unusable state, requiring administrative attention	✓
EMERGENCY	Entire server in an unusable state	✓

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Message Severity

Each log message has an associated severity level. The level gives a rough guide to the importance and urgency of a log message. Oracle WebLogic Server has predefined severities, ranging from TRACE to EMERGENCY, which are converted to a log level when dispatching a log request to the logger. By default, servers forward only messages of the severity level NOTICE or higher. Although you can modify the set of messages that are forwarded, servers can never forward messages of the DEBUG severity level.

The Oracle WebLogic Server subsystems generate many messages of lower severity and fewer messages of higher severity. For example, under normal circumstances, they generate many INFO messages and no EMERGENCY messages.

Message Catalog Using the Web

Oracle WebLogic Server and WebLogic Express 10.3.3.0 Messa...

Oracle Fusion Middleware Oracle WebLogic Server Mess
11g Release 1 (10.3.3)
Part Number E14397-02

OTN Home > BEA WebLogic Server 10.3 Documentation > Index of Me

Index Of Messages By Message Range

Messages in the Message Catalog are part of the WebLogic S
Localization packages.

Range	Subsystem
WL-000001 - WL-009999	ConsensusLeasing
WL-000100 - WL-000199	Cluster
WL-000200 - WL-000399	WebLogicServer
WL-000400 - WL-000499	Socket
000500 - 000599	RJVM
WL-000600 - WL-000699	Common
WL-000700 - WL-000799	T3Misc
WL-000800 - WL-000899	Kernel
WL-000900 - WL-000999	Net
WL-001000 - WL-001999	JDBC
WL-002000 - WL-002499	IIOP
WL-002500 - WL-002600	DRS
WL-002601 - WL-002799	Server

BEA WebLogic Server and WebLogic Express 9.2 Mes

http://download.oracle.com/docs/cd/E13222_01/wl

Documentation Home > BEA WebLogic Server 10.0 Documentation > Index of Mes
(by Subsystem)

Index Of Messages By Message Range

Messages in the Message Catalog are part of the WebLogic Server Inter
Localization packages.

Range	Subsystem	Catalog
BEA-000100 - BEA-000199	Cluster	Cluster
BEA-000200 - BEA-000399	WebLogicServer	T3Srvr
BEA-000400 - BEA-000499	Socket	Socket
000500 - 000599	RJVM	RJVM
BEA-000600 - BEA-000699	Common	Common
BEA-000700 - BEA-000799	T3Misc	T3Misc
BEA-000800 - BEA-000899	Kernel	Kernel
BEA-000900 - BEA-000999	Net	Net
BEA-001000 - BEA-001999	JDBC	JDBC
BEA-002000 - BEA-002499	IIOP	IIOP
BEA-002500 - BEA-002600	DRS	DRS
BEA-002601 - BEA-002799	Server	Server

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Message Catalog Using the Web

The screenshot shows the first page of the message catalog index. Use the WLS online message catalogs to obtain more information about a specific log message ID.

For a detailed description of the log messages in the Oracle WebLogic Server message catalogs, see *Oracle WebLogic Server Message Catalogs* in the online documentation. This index of messages describes all the messages generated by the Oracle WebLogic Server subsystems and provides a detailed description of the error, a possible cause, and a recommended action to avoid or fix the error.

In an upcoming release of WebLogic Server, the current default prefix for catalog and noncatalog Message IDs will be changed from the current BEA prefix to WL.

Workaround

You should be prepared for this future change. In the interim, here are some guidelines to consider:

- Avoid depending on BEA for Message ID prefixes in scripts, filter expressions, and so on.
- For log messages, such as the following:
  `<Jan 30, 2009 12:51:49 AM CST> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to STARTING>`
  it is better for you to filter on '`000365`' and not on the BEA prefix itself.
- Your log parsing scripts should be updated to look for both BEA and WL, instead of filtering only on BEA.

Message Catalog Cross-Reference

```
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to ADMIN>
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to
      RESUMING>
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <Cluster> <BEA-000162> <Starting "async" replication service
      with remote cluster address "localhost">
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <Server> <BEA-002613> <Channel "Default" is now listening on
      139.185.35.128:7023 for protocols iiop, t3, CLUSTER-BROADCAST, ldap, snmp, http.>
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <Server> <BEA-002613> <Channel "Default[1]" is now listening on
      fe80:0:0:0:21a:a0ff:fec0:63fb:7023 for protocols iiop, t3, CLUSTER-BROADCAST, ldap, snmp, http.>
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <Server> <BEA-002613> <Channel "Default[2]" is now listening on
      0:0:0:0:0:0:0:1:7023 for protocols iiop, t3, CLUSTER-BROADCAST, ldap, snmp, http.>
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <Server> <BEA-002613> <Channel "Default[3]" is now listening on
      127.0.0.1:7023 for protocols iiop, t3, CLUSTER-BROADCAST, ldap, snmp, http.>
<Feb19,2009 10:10:27 AM GMT+07:00> <Notice> <WebLogicServer> <BEA-000330> <Started WebLogic Managed Server
      "MedRecSvr2" for domain "MedRecDomain" running in Production Mode>
<Feb19,2009 10:10:29 AM GMT+07:00> <Notice> <Cluster> <BEA-000102> <Joining cluster MedRecClust1 on
      239.192.0.0:7030>
<Feb19,2009 10:10:29 AM GMT+07:00> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to RUNNING>
<Feb19,2009 10:10:29 AM GMT+07:00> <Notice> <WebLogicServer> <BEA-000360> <Server started in RUNNING mode>
```

BEA-002613 *Notice:* Channel "*channel*" is now listening on *listenAddress:port* for protocols *protocols*.

Description The server successfully started the listen thread and server socket.

Cause None.

Action None.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide

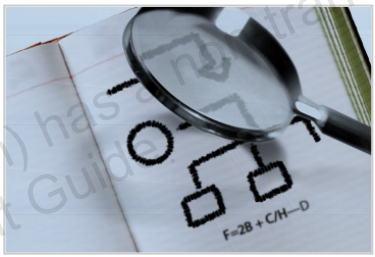
Message Catalog Cross-Reference

Server Subsystem Messages

The Server1.0 catalog contains messages in the range from BEA002601 to BEA002799. Messages in this catalog are part of the `weblogic.server` internationalization package and the `weblogic.i18n` localization package.

Road Map

- Logs and monitoring
- Log filters



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

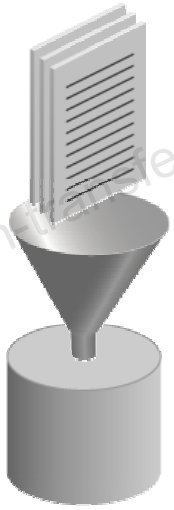
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Log Filters

Log filters:

- Control the log messages that get published
- Are based on the values of message attributes
- Can be applied to different message destinations:
  - Server log file
  - Server memory buffer
  - Server standard out
  - Domain log file



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

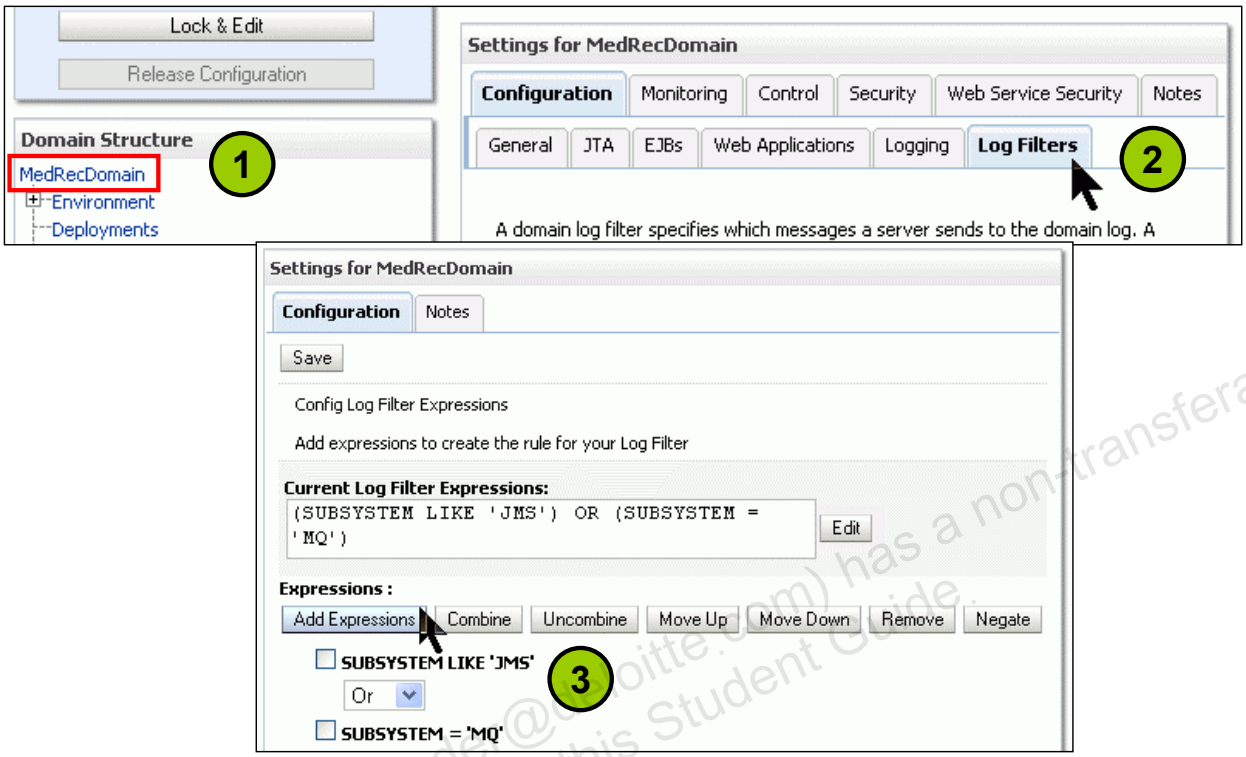
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Log Filters

Log filters provide control over the log messages that are published. A filter uses custom logic to evaluate the log message content, which you use to accept or reject a log message—for example, to filter out messages of a certain severity level from a particular subsystem (for example, JMS or MQ) or according to a specified criteria (for example, only during peak business hours). Only the log messages that satisfy the filter criteria are published. You can create separate filters for the messages that each server instance either writes to its server log file, standard out, or memory buffer, or broadcasts to the domainwide message log.

Creating a Log Filter



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

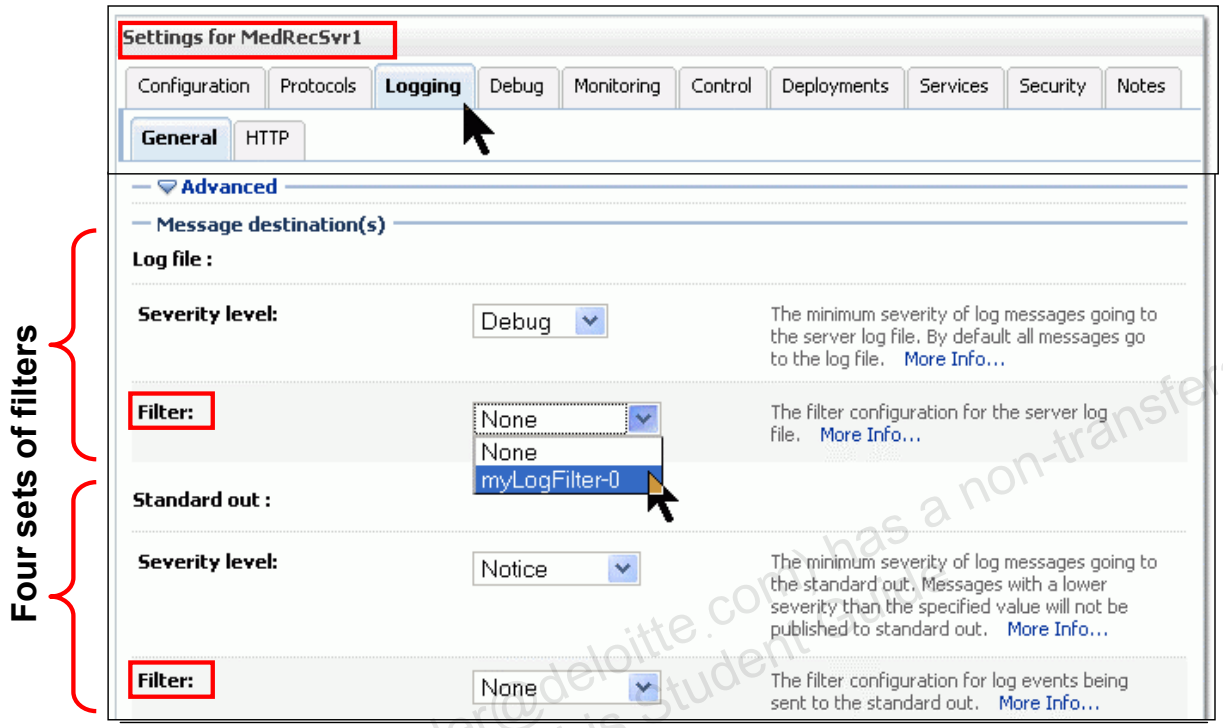
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Log Filter

1. In the left pane of the console, select the name of the active domain in the Domain Structure panel.
2. On the Configuration: Log Filters page, click New.
3. On the "Create a New Log Filter" page, enter a value to identify the filter in the Name field. Do not add single quotation marks around the value when you type it; the system will add the quotation marks automatically.
4. Click Finish.
5. The new log filter appears in the Log Filters table. To configure a filter expression, click the log filter name.
6. On the Configuration page, specify the criteria for qualifying messages. A filter expression defines simple filtering rules to limit the volume of log messages that are written to a particular log destination. Either enter the expression manually or construct one by using the supplied buttons.
7. Expression clauses can perform a comparison against any log message attribute, including severity and subsystem.

Applying a Log Filter



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Applying a Log Filter

There are four sets of logs that you can apply filters to. The Log file and Standard out are shown; the Domain log broadcaster and Memory buffer are not shown.

1. In the left pane of the console, expand Environment and select Servers.
2. Under Servers, click the name of the server instance whose logging you want to configure. In this example, it is MedRecSvr1.
3. Select Logging > General.
4. On the Logging: General page, click Advanced.
5. Under the Message destination(s) section, specify an existing filter for messages going to any combination of the four log message destinations (Log file, Standard out, Domain log broadcaster, and Memory buffer).

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using the Console to Monitor

The Administration Console offers some monitoring capabilities:

Attribute	Description
Monitoring	**Many of the Console's objects have a Monitoring tab that allows you to view monitoring information for that object.**
Customize this table	**You can customize the monitoring view by clicking the "Customize this table" link.**

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

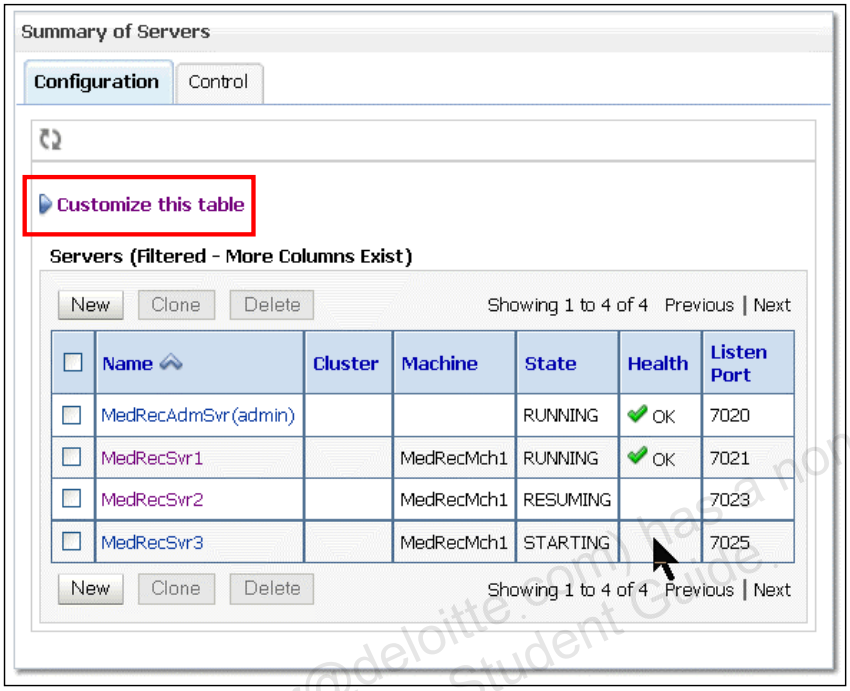
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using the Console to Monitor

Whenever a service or an application object can be monitored, a Monitoring tab is available in the Console for that object. Clicking it shows you the available monitoring information for the selected object.

Moreover, when the Monitoring page shows information in tabular format, you can change the way the information is displayed. To do this, click the "Customize this table" link and choose which columns to display and on what columns to sort the table.

Monitoring Running Servers



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl MCGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Monitoring Running Servers

You cannot monitor the activity of one domain from another domain. For example, you cannot open the Administration Console from the default domain and try to monitor servers within another domain.

Suppose you are never interested in monitoring information about clusters. You can suppress the Cluster column from the view.

Customizing Views

Customize this table

Filter

Filter by Column: Name Criteria:

View

Column Display:

Available:

- Accept Backlog
- Cluster Weight
- Current Machine
- Currently Open Sockets
- Default IIOP Username
- Expected To Run
- HTTPD Enabled

Chosen:

- Name
- Cluster
- Machine
- State
- Health
- Listen Port

Number of rows displayed per page: 1

Apply Reset

Notice the Cluster column. It can be suppressed in the view.

Columns can be reordered.

Servers (Filtered - More Columns Exist)

New Clone Delete Showing 1 to 4 of 4 Previous | Next

Name	Machine	State	Health	Listen Port
MedRecAdmSvr(admin)		RUNNING	OK	7020
MedRecSvr1	MedRecMch1	SHUTDOWN		7021
MedRecSvr2	MedRecMch1	SHUTDOWN		7023
MedRecSvr3	MedRecMch1	SHUTDOWN		7025

New Clone Delete Showing 1 to 4 of 4 Previous | Next

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

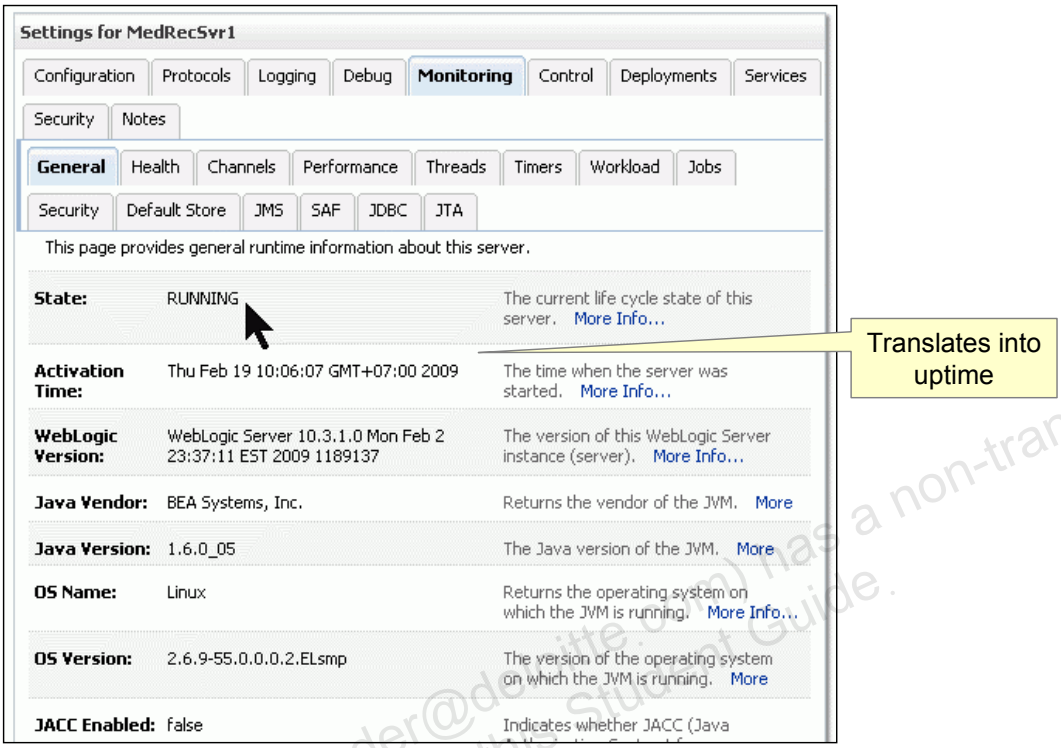
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Customizing Views

The screenshot shows that the Cluster column is no longer being displayed in the view. The "Customize this table" link allows you to change the order and the number of columns that are displayed.

Monitoring Individual Servers



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Monitoring Individual Servers

When you select a specific server from the *domain*/`Servers` element of the left-most console pane, a server-specific monitoring page is displayed. A large amount of detailed monitoring information can be obtained about any running instance of Oracle WebLogic Server. The Performance tab shows a running numerical output of request throughput, waiting request, and memory in use, and gives a fast indication of server performance. For a graphical view, use the WLDF Console Extension tab.

Additionally, in-depth information is available about a server's associated cluster, such as multicast packet loss, packet fragments, and so on. The Security tab provides statistics specific to security such as invalid logins and locked-out users. The remaining tabs provide in-depth information about their associated service.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmccrudder@dollarfee.com) has a non-transferable license to use this Student Guide.

Network-Addressing Environment

Production environments often have complex networking requirements. Using Network Channels, you can configure:

- Multiple NICs for a single WLS server
- Specific NICs or multiple port numbers on an NIC for specific WLS servers
- Multiple IP addresses with each server
- A single IP address with multiple ports
- Multiple SSL configurations on one server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Network-Addressing Environment

For many development environments, configuring the Oracle WebLogic Server network resources is simply a matter of identifying a managed server's listen address and listen port. However, in most production environments, administrators must balance finite network resources against the demands placed on the network. The task of keeping applications available and responsive can be complicated by specific application requirements, security considerations, and maintenance tasks, both planned and unplanned.

Oracle WebLogic Server allows you to control the network traffic associated with your applications in a variety of ways, and configure your environment to meet the varied requirements of your applications and end users. You can:

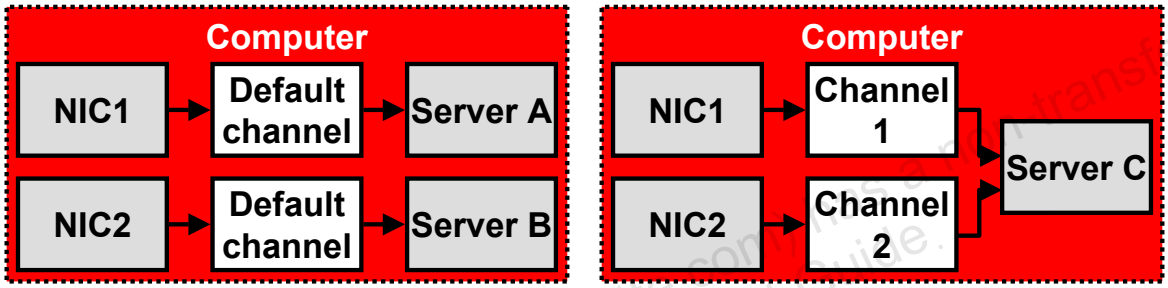
- Designate the Network Interface Cards (NICs) and ports used by managed servers for different types of network traffic
- Support multiple protocols and security requirements
- Specify connection and message time-out periods
- Impose message size limits

You specify these and other connection characteristics by defining a **network channel**, the primary configurable Oracle WebLogic Server resource for managing network connections. You configure a network channel either in the Administration Console (**Servers > Protocols > Channels**) or by using `NetworkAccessPointMBean`.

Network-Addressing Features

Add flexibility to networking configuration by defining and using channels for:

- Administration-traffic-only port
- Separating inbound versus outbound ports
- Defining Quality of Service



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Network-Addressing Features

The amount of logging traffic coming into the administration server from the other managed servers by server log broadcasting can be substantial. If the logging traffic starts to interfere with the application traffic, you might use network channels to guarantee that both administrative logging traffic and application traffic have separate virtual “pipes” on the same network facility.

A channel is a way that WebLogic Server can refer to an IP address:port combination by assigning a name to it for purposes of segregating traffic. This can be useful to guarantee that one kind of traffic does not monopolize specific network hardware at the expense of another kind of traffic (for example, user traffic versus administrative traffic). If the standard Oracle WebLogic Server administrative channel does not satisfy your requirements, you can configure a custom channel for administrative traffic. For example, a custom administrative channel allows you to segregate administrative traffic on a separate NIC. The *WebLogic Server 11g: Advanced Administration* course covers network channels in more depth.

Quiz

Which is *NOT* a standard severity level for Oracle WebLogic Server log messages?

a. Debug
b. Transaction
c. Info
d. Notice
e. Error

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

`Info`, `Debug`, `Notice`, and `Error` are all valid log message severities. Though log messages can have an optional Transaction ID attribute, it is not a severity level.

Summary

In this lesson, you should have learned how to:

- Configure server and domain logging
- Locate and interpret the format of domain and server log files
- View logs using the Administration Console
- Describe message attributes
- Configure server standard output settings using the console
- Integrate applications with the WLS logging infrastructure
- Access online log message catalogs
- Create and apply log filters and expressions

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 10 Overview: Viewing and Managing WLS Logs

This practice covers the following topics:

- Configuring a logging filter
- Viewing logs

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 10 Overview: Viewing and Managing WLS Logs

See Appendix A for the complete steps to do the practice.

11

Deployment Concepts

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Contrast autodeploy with manual deployment
- Configure and deploy Web applications via the Administration Console, command line, and WLST
- Configure deployment descriptors
- Test deployed applications
- Describe the role of Web servers
- Trace a typical Web interaction flow
- Contrast static and dynamic content and deployment
- Front-end deployed applications with a Web server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

You want to install (deploy) two applications: applA and applB. After they are deployed, there is a change to applA (version 1.1), so it needs to be updated (redeployed), and applB is redundant, so it will be deleted (undeployed). You have not decided which server will host the applications, so you would like to make a form of redirection such that the URL for the application will not change even if the location of the server changes. In other words, if applA goes on Server1, its URL would be http://wls-sysadm:7021/applA, but if it were located on Server2, its URL would be http://wls-sysadm:7023/applA. You do not have to tell the users that you are changing the URLs, so you pick a different URL: http://wls-sysadm:7777/applA, and within that there will be a redirection to either *host:port* (*host*= wls-sysadm or edvmr1p0, *port*= 7021 or 7023) or whatever URL you choose. The user only has to remember the one URL that never changes even though the Web administrator can change the infrastructure without the user knowing it.

Road Map

- Deployment concepts
  - Console deployment
  - Command-line deployment
  - Autodeployment
- Developer deployment
- Front-end with a Web server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Overview of Deployment

Two views of deployment:

- Developers
  - Development environment
  - Single stand-alone WLS server (both admin and applications together)
  - Deploy over and over again at will during the testing phase

- Administrators
  - Production environment
  - Multiple WebLogic Server instances or clusters
  - Deploy infrequently during maintenance schedules

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Overview of Deployment

To deploy is to validate a collection of resources and to prepare them for execution by the container. Different groups think of deployment in different ways.

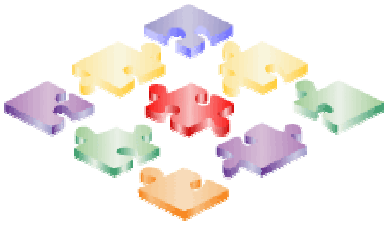
Developers may need to deploy an application in a development environment, package an application for delivery to an administrator or deployer, or create and export the configuration of an application for deployment to a testing, staging, or production environment. Generally, they deploy the same application dozens of times until the application is completely tested and debugged. Also, often their development domain is only one computer, they are the "administration" server except that they are the only server, so there is nothing else to administer. Deploying on to the administration server would never be done in a production environment, but it is fine for a development environment.

Administrators want to deploy Java Platform Enterprise Edition (Java EE) applications or application modules to WebLogic Server instances or clusters. It is assumed that you, as an administrator, are working in a production environment, which is generally characterized by multiple WebLogic Server instances or clusters running on multiple machines. It is also assumed that you have one or more application module archive files that have been tested and are ready to deploy on a production server.

What Is Deployed?

Deploy Java EE application in:

- Exploded form
  - Directory structure very important
  - Easier to update individual pieces
  - Harder to keep track of the whole collection



- Archive form
  - Similar to `.zip` files, can be maintained with ZIP or IDE tools
  - Type: `.jar`, `.war`, `.ear`, `.rar`, and so on
  - Contains code, metacode, descriptors (xml), directories

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrurder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Is Deployed?

Using Archived Files

An archive file is a single file that contains all of an application's or module's classes, static files, directories, and deployment descriptor files. In most production environments, the applications an Administrator receives for deployment are stored as archive files. Use an IDE tool such as JDeveloper, Eclipse, or command-line `jar` utility to create and maintain the archive files.

Deployment units packaged using `jar` have a specific file extension depending on the type:

- EJBs and client archives are packaged as `.jar` files.
- Web applications are packaged as `.war` files.
- Resource adapters are packaged as `.rar` files.
- Enterprise applications are packaged as `.ear` files, and can contain other Java EE modules, such as EJBs, JDBC, JMS, Web applications, and Resource Adapters.
- Web Services can be packaged either as `.war` files or as `.jar` files, depending on whether they are implemented using Java classes or EJBs. Typically, the `.war` or `.jar` files are then packaged in an `.ear` file.
- Java EE libraries are packaged either as an `.ear` file or as a standard Java EE module.
- Client applications and optional packages are packaged as `.jar` files.

In addition to an archive file, you may also receive a deployment plan, which is a separate file that configures the application for a specific environment.

What Is Deployed? (continued)

Using Exploded Archive Directories

An exploded archive directory contains the same files and directories as a JAR archive. If you choose to use an exploded archive directory, you may be required to manually unpack a previously archived deployment. However, the files and directories reside directly in your file system and are not packaged into a single archive file with the `jar` utility.

You may choose to deploy from an exploded archive directory under the following circumstances:

- You want to perform partial updates of an Enterprise application after deployment. Deploying Enterprise Applications from an exploded archive directory makes it easier to update individual modules of the application without having to re-create the archive file.
- You are deploying a Web application or Enterprise application that contains static files that you will periodically update. In this case, it is more convenient to deploy the application as an exploded directory because you can update and refresh the static files without re-creating the archive.
- You are deploying a Web application that performs direct file system input/output (I/O) through the application context (for example, a Web application that tries to dynamically edit or update parts of the Web application itself). In this case, the modules that perform the I/O operations should have a physical file system directory in which to work; you cannot obtain a file when the application is deployed as an archive, as per the specification.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deployment Process

Deploying an application involves the following tasks:

- **Preparing:** Choosing whether to package the application as an archived file or keep it in an exploded directory
- **Configuring:** Creating a deployment plan to maintain the configuration changes without changing the deployment descriptors
- **Deploying:** Targeting and distributing the application to servers in an Oracle WebLogic Server domain
  - Install or deploy
  - Update or redeploy
  - Delete or undeploy

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deployment Process

Preparing applications and modules for deployment: You can deploy applications either as archived files or as exploded archive directories. Oracle WebLogic Server also introduces the concept of an application installation directory, which helps you to organize the deployment files and the deployment configuration files for easy deployment using Oracle WebLogic Server tools. Before deploying an application, an administrator typically prepares the deployable modules by creating an application installation directory and copying the application archive file into the appropriate subdirectory.

Configuring the application or module for deployment to the Oracle WebLogic Server environment: Administrators typically receive a new application (or a new version of an application) from their development team and must deploy the application to a staging or production environment that differs from the environments used during development and testing. Oracle WebLogic Server helps you to easily configure an application for a new target domain *without* having to manually edit the deployment descriptor files provided by the development team. The configuration changes for a specific deployment environment are stored in a new configuration artifact—a *deployment plan*—that can be stored and maintained independently of the deployment files provided by the development team.

Deploying the application to Oracle WebLogic Server: After preparing both the deployment and configuration files, applications are distributed to the target servers in an Oracle WebLogic Server domain and made active for processing client requests.

Deployment Methods

- WLS supports three deployment methods:
  - Console deployment
  - Command-line deployment
  - Autodeployment
- Applications and EJBs can be deployed in an:
  - Archived file (`.ear`, `.war`, `.jar`)
  - Exploded (open) directory format

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deployment Methods

Oracle WebLogic Server supports three distinct deployment methods for your applications. Different deployment methods are provided because different scenarios exist in which the applications execute. For instance, an Oracle WebLogic Server system can be used for development, testing, staging, or production. The method of deployment can be dependent upon the domain that it is deployed to.

Console deployment uses the Administration Console. The Administration Console screens prompt you for values. Both programmers and administrators use the Administration Console.

Command-line deployment uses the WLST scripting tool. This is well suited for repetitive automated deployments. Both programmers and administrators use WLST.

Autodeployment is usually used in development mode by programmers, not in Production Mode.

Deployment Tools

Several methods are available to deploy applications and shared libraries to the Oracle WebLogic Server, including:

- Administration Console
  - Install
  - Update
  - Delete
- Command-line
  - WebLogic Scripting Tool (WLST)
  - `weblogic.Deployer` Java class
  - `wldeploy` Ant task
- Autodeployment folder

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deployment Tools

- The Administration Console provides a series of Web-based Deployment Assistants that guide you through the deployment process. The Administration Console also provides controls for changing and monitoring the deployment status, and changing selected deployment descriptor values while the deployment unit is up and running. Use the Administration Console when you need to perform basic deployment functions interactively and you have access to a supported browser.
- The WebLogic Scripting Tool (WLST) is a scripting interface that you can use to automate domain configuration tasks, including application deployment configuration and deployment operations.
- Similarly, `weblogic.Deployer` provides a command line–based interface for performing both basic and advanced deployment tasks. Use `weblogic.Deployer` when you want command-line access to the Oracle WebLogic Server deployment functionality, or when you need to perform a deployment task that is not supported using the Administration Console.
- The `wldeploy` task is an Ant-based version of the `weblogic.Deployer` utility. You can automate deployment tasks by placing the `wldeploy` commands in an Ant `build.xml` file and running Ant to execute the commands.
- The *`domain`*`/autodeploy` directory allows you to deploy an application quickly for evaluation or testing in a development environment. Autodeployment is covered in more detail later in this lesson.

Console Deployment Method

Deploying with the console allows full administrator control:

- Installation of an application from a location of your choice
- Manual configuration of the application name
- Targeting the application to individual servers or clusters, or both
- Configuring the application without targeting it
- Activating deployment when desired

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Console Deployment Method

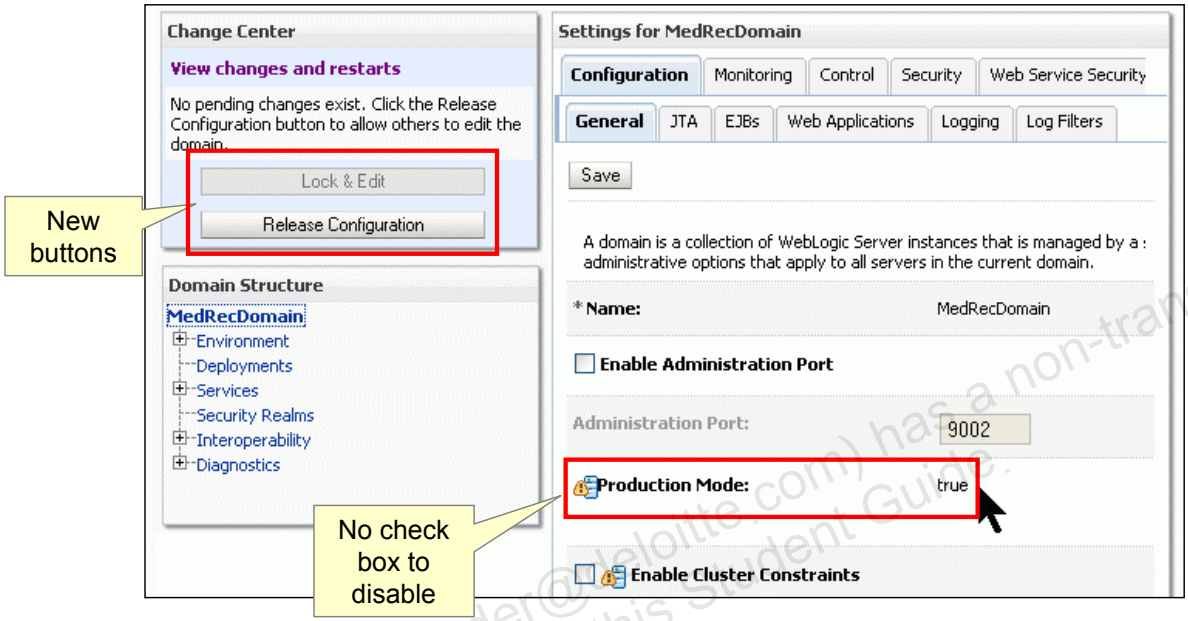
The console deployment method allows manual control over the deployment process. You can install applications directly from a network location of choice (configured). You do not have to move application files to deploy them.

You can also override the deployment unit name of the application at this point. This can be useful if, for example, application versioning changed the application archive file name.

Third, you can target each application separately to individual servers or clusters in the domain.

Console Deployment Production Mode

Best used with Production Mode:



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Console Deployment Production Mode

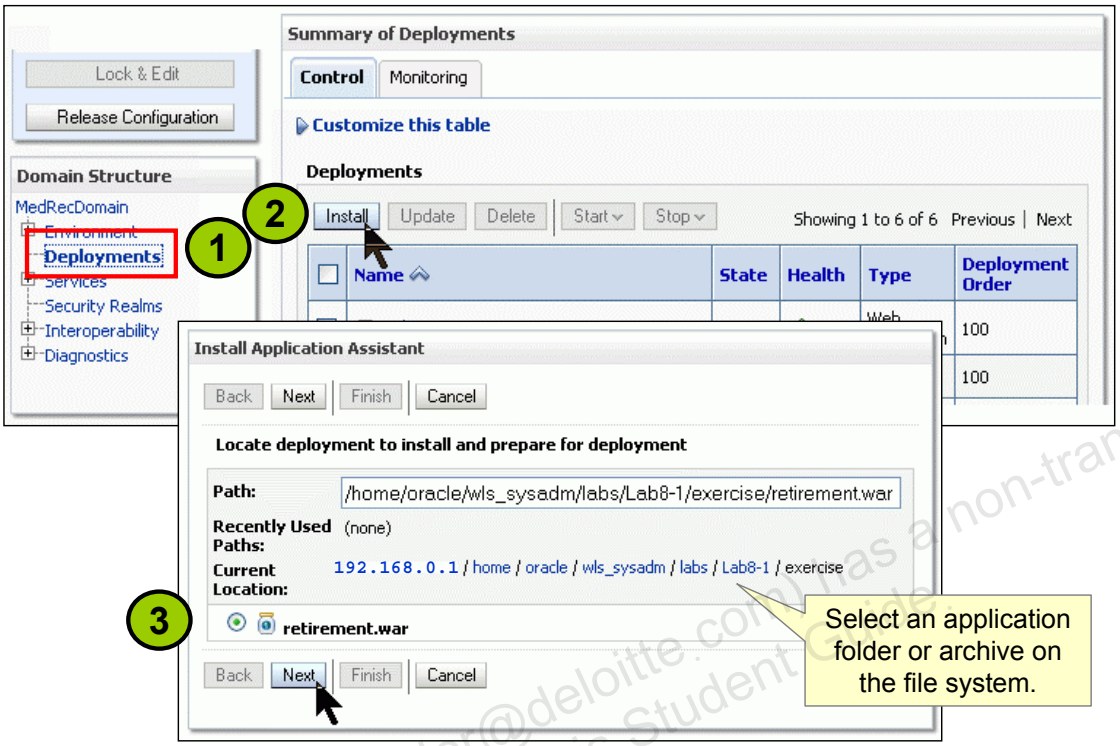
The screenshot shows that Production Mode for the domain is true, which means that it is in Production Mode. The console deployment methods are best used with autodeployment disabled. In this way, if you decide to install your applications to the *domain*/autodeploy directory, you still retain full manual control over the deployment process. Note that the autodeploy directory is not in your domain by default; it is neither created nor required for Production Mode. You do not have to copy your applications to this directory; you may choose any directory as long as the server has access to it. Autodeployment is covered later in this lesson.

The next ten slides show how to do console deployment.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Preparing a New Application



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

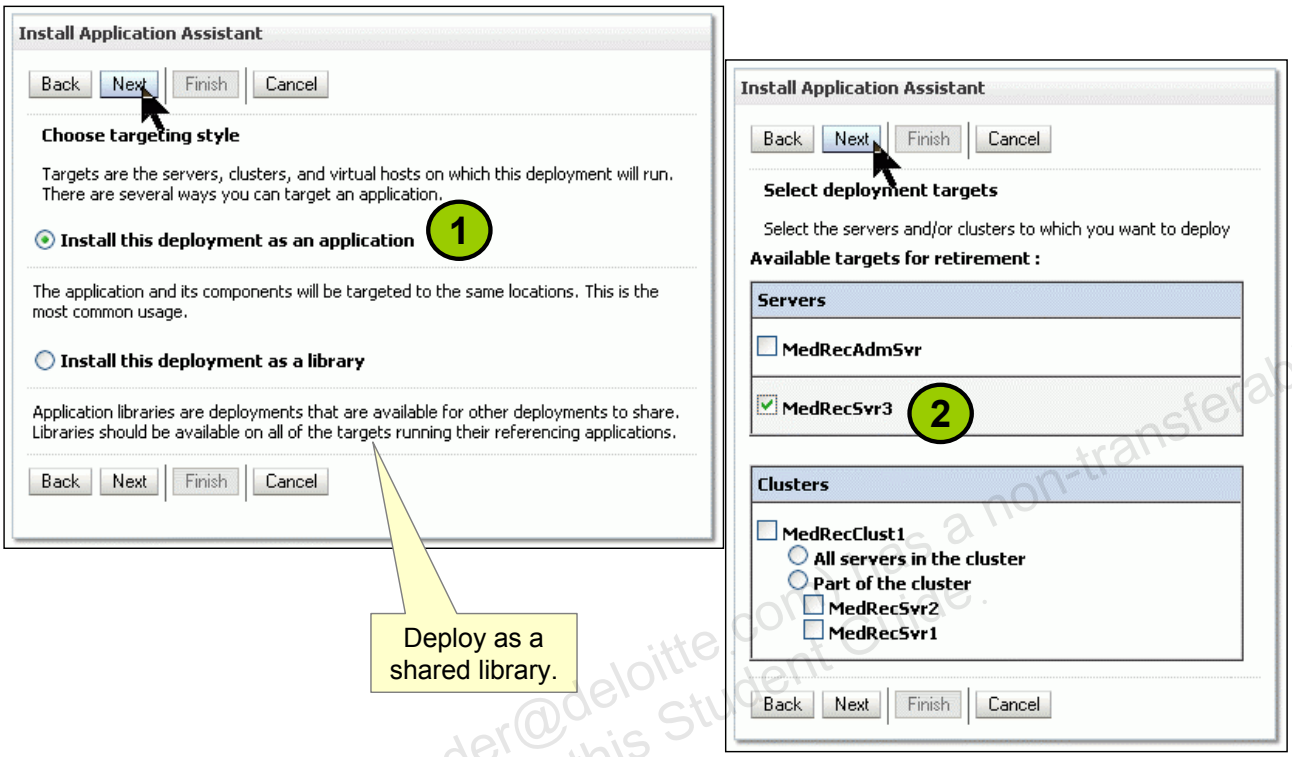
Preparing a New Application

The screenshot shows the deployment of the retirement application. All types of Java EE applications are deployed in the same way in Oracle WebLogic Server. Installing an application refers to making its physical file or directory known to Oracle WebLogic Server. An application can be installed as an archived file or as an exploded directory. After you have installed the application, you can start it so that users can use it.

1. In the left pane of the Console, click Deployments.
2. In the right pane, click Install.
3. Locate the file or exploded directory that corresponds to the application that you want to install. Either enter the path on the local file system manually, or use the Current Location or Recently Used Paths feature. If using these features, select the path or file using the supplied option buttons. Click Next.

If the application archive is not accessible from the Administration Server's file system, you can use the Upload link to upload the application from your browser's file system to the path indicated by Current Location.

Preparing a New Application: Targeting



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Preparing a New Application: Targeting

The screenshot shows some of the application deployment choices.

1. Select “Install this deployment as an application” to register it as a client-accessible application. Select “Install this deployment as a library” to instead register it as a shared library that can be referenced by other applications. For version 10.3.0, there was an “Install this deployment as an application, but target the modules individually” option. That option is no longer available. Click Next.
2. Select the managed servers or clusters to which you want to deploy the application. Click Next.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Preparing a New Application: Settings

Install Application Assistant

Back | Next | Finish | Cancel

3

Optional Settings

You can modify these settings or accept the defaults

General

What do you want to name this deployment?

Name: retirement

Security

What security model do you want to use with this application?

- **DD Only: Use only roles and policies that are defined in the deployment**
- **Custom Roles: Use roles that are defined in the Administration Console**
- **Custom Roles and Policies: Use only roles and policies that are defined**
- **Advanced: Use a custom model that you have configured on the realm**

Source accessibility

How should the source files be made accessible?

- **Use the defaults defined by the deployment's targets**

Recommended selection.

- **Copy this application onto every target for me**

During deployment, the files will be copied automatically to the managed servers to which the application is targeted.

- **I will make the deployment accessible from the following location**

Location: /home/oracle/wls_sysadm/labs/Lab8-1/exercise/retire

Provide the location from where all targets will access this application's files. This is often a shared directory. You must ensure the application files exist in this the location.

Install Application Assistant

Back | Next | Finish | Cancel

Review your choices and click Finish

Click Finish to complete the deployment. This may take a few moments to complete

Additional configuration

In order to work successfully, this application may require additional configuration. D

- **Yes, take me to the deployment's configuration screen.**
- **No, I will review the configuration later.**

4

Summary

Deployment: /home/oracle/wls_sysadm/labs/Lab8-1/exercise/retire
Name: retirement
Staging mode: Copy this application to every target for me
Security Model: DDOnly: Use only roles and policies that are defined in

Target Summary

Components	Targets
retirement	MedRecSvr3

Back | Next | Finish | Cancel

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

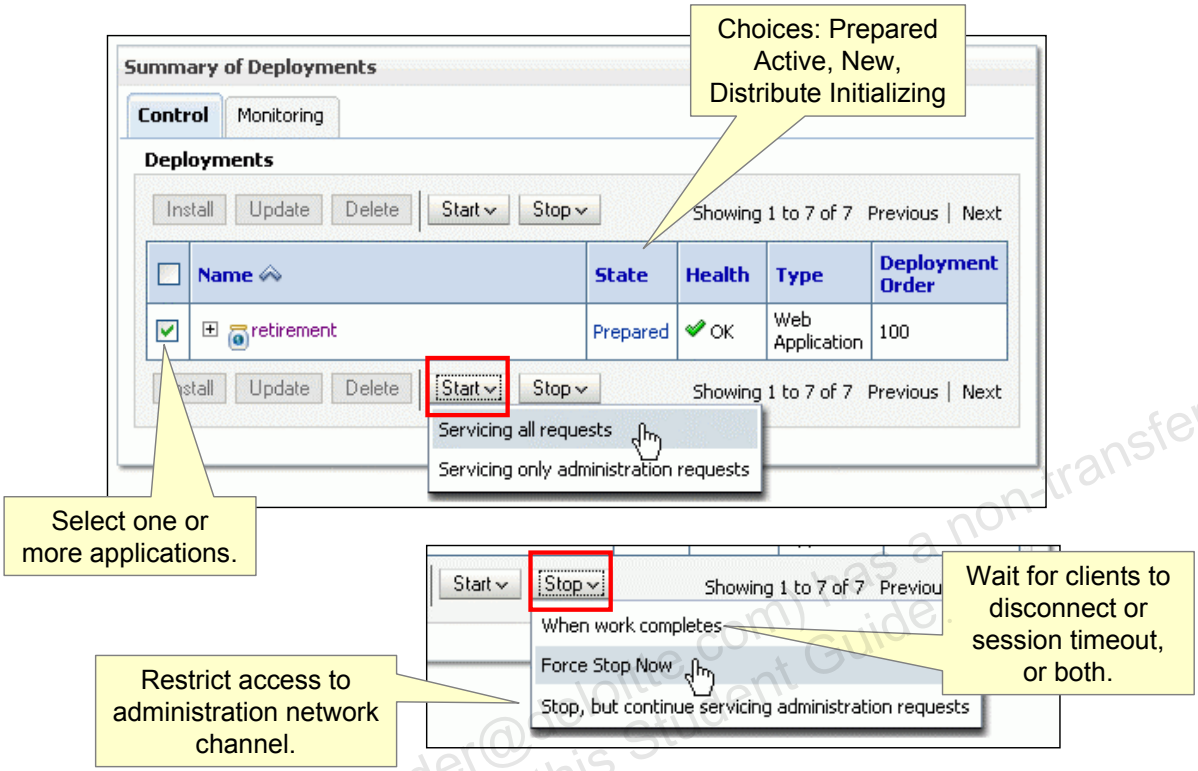
Preparing a New Application: Settings

The screenshot shows some of the application deployment choices.

3. Optionally, update the settings about the deployment, including the configuration name and the security model to use (deployment descriptor and/or WebLogic roles and policies). Click Next.
4. Review the configuration settings that you have chosen, specify whether you want to immediately update the application's configuration after you install it, and then click Finish to complete the installation.

The dotted lines make the description appear to go with the wrong choices. For example, the recommended selection is "Use the defaults defined by the development's targets," not "Copy this application onto every target for me." The description is below the option; ignore the dotted lines.

Deploying or Undeploying Applications



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Deploying or Undeploying Applications

The screenshots show the deployment status after deploying, but before starting. The State will change from "Distribute initializing" to Prepared.

When you start an application, you can make it immediately available to clients, or you can start it in Administration mode to first ensure that it is working. Starting in Administration mode allows you to perform the final ("sanity") check of the distributed application directly in the production environment without disrupting clients. Administration mode restricts access to an application to a configured Administration channel. Similarly, you can stop an application so that no clients use it, or you can stop, but continue servicing administration requests so that only the administrative tasks can be performed.

Stopping an application does not remove its source files from the server; you can later redeploy (also called update) a stopped application to make it available to the Oracle WebLogic Server clients once again.

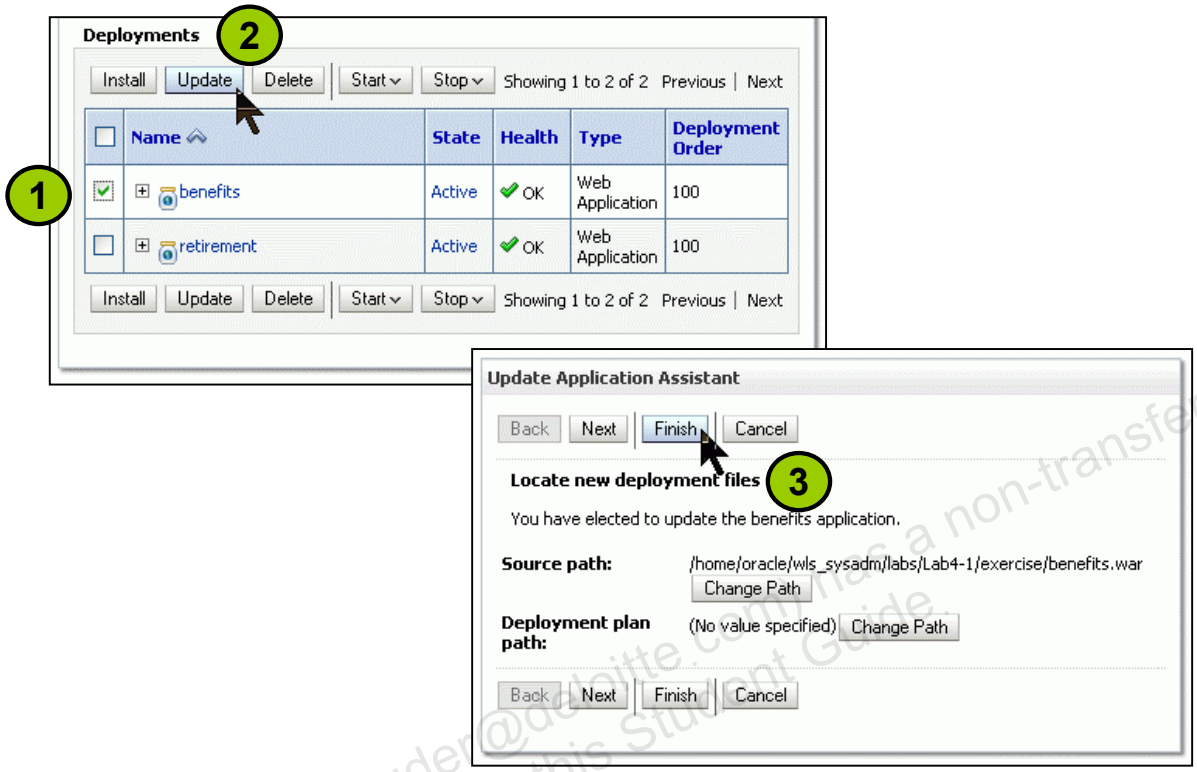
To deploy or undeploy applications, perform the following steps:

1. In the left pane of the console, select Deployments. A table in the right pane displays all the deployed applications and modules.
2. Select the check boxes for the applications that you want to start or stop.
3. Click Start or Stop, and select from the available options.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@ecollege.com) has a non-transferable license to use this Student Guide.

Redeploying an Application



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

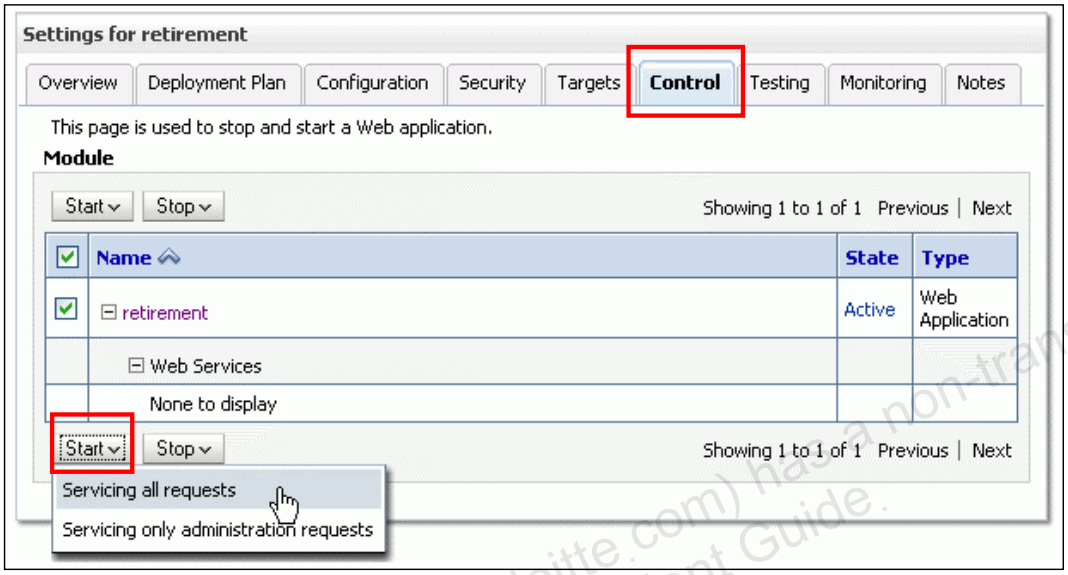
Redeploying an Application

The screenshot shows the application update or redeployment choices. When you update an application, you can specify that Oracle WebLogic Server should redeploy the original archive file or exploded directory, or you can specify that Oracle WebLogic Server should deploy a new archive file in place of the original one. You can also change the directory that contains the deployment plan that is associated with the application. Update an application if you have made changes to the application and want to make the changes available to the Oracle WebLogic Server clients, or if you want to redeploy an entirely new archive file in a new location.

1. In the left pane of the console, select Deployments. Select the check boxes for the applications that you want to redeploy.
2. Click the Update button. The application does not have to be stopped to be redeployed.
3. If the path of the application has not changed, click Finish. Otherwise, use the Change Path button to select a new location. If your application uses a deployment plan, you also have the option of changing its location as well using the Change Plan button. If the application is a versioned application, it will change the version number.
   However, changing paths is only supported for versioned applications. Attempts to change the path for a nonversioned application will give the following error:
   ```
   Changing the source location is not allowed for a previously
   attempted deployment.
   ```

Versioning is covered in a later lesson titled “Advanced Deployment.”

Starting and Stopping an Application



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

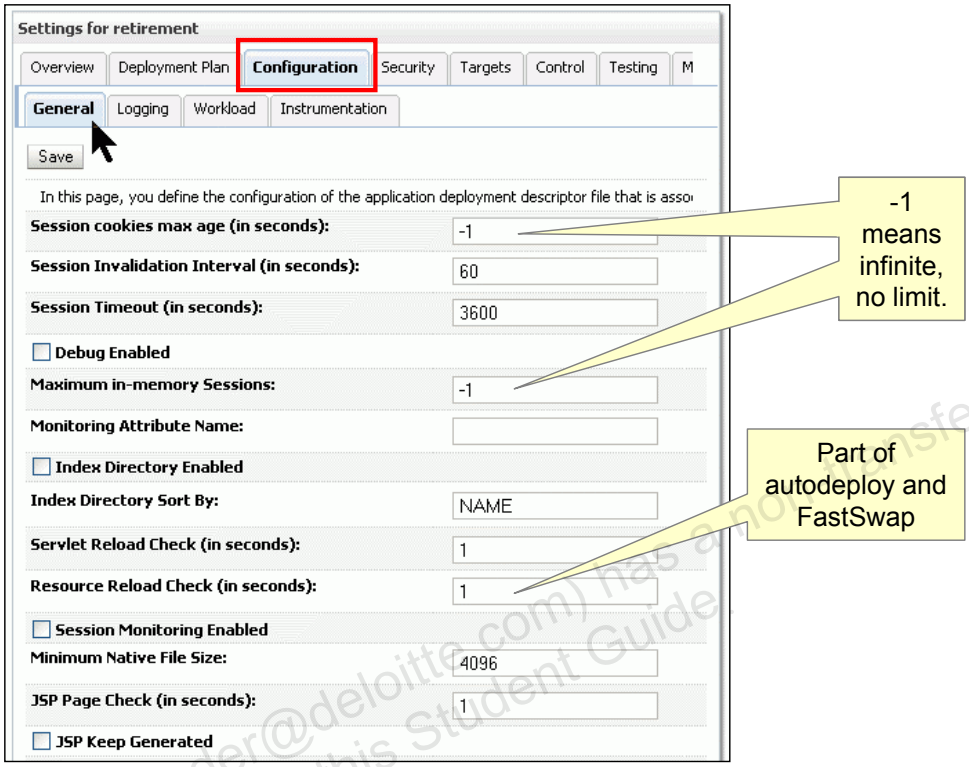
Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Starting and Stopping an Application

The screenshot shows starting and stopping an application. After you have reviewed your configuration and activated your changes, you need to start the application. You can choose to open it up to all requests (that is, including client requests) or only administration requests (that is, access to the application is restricted to a configured Administration channel). The latter is good for testing an application before releasing it to users.

Depending on the type of application, there may be other expandable items under Name. For example, in addition to Web Services, there can be EJBs and modules with their respective states and types.

Editing Deployment Descriptors



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Editing Deployment Descriptors

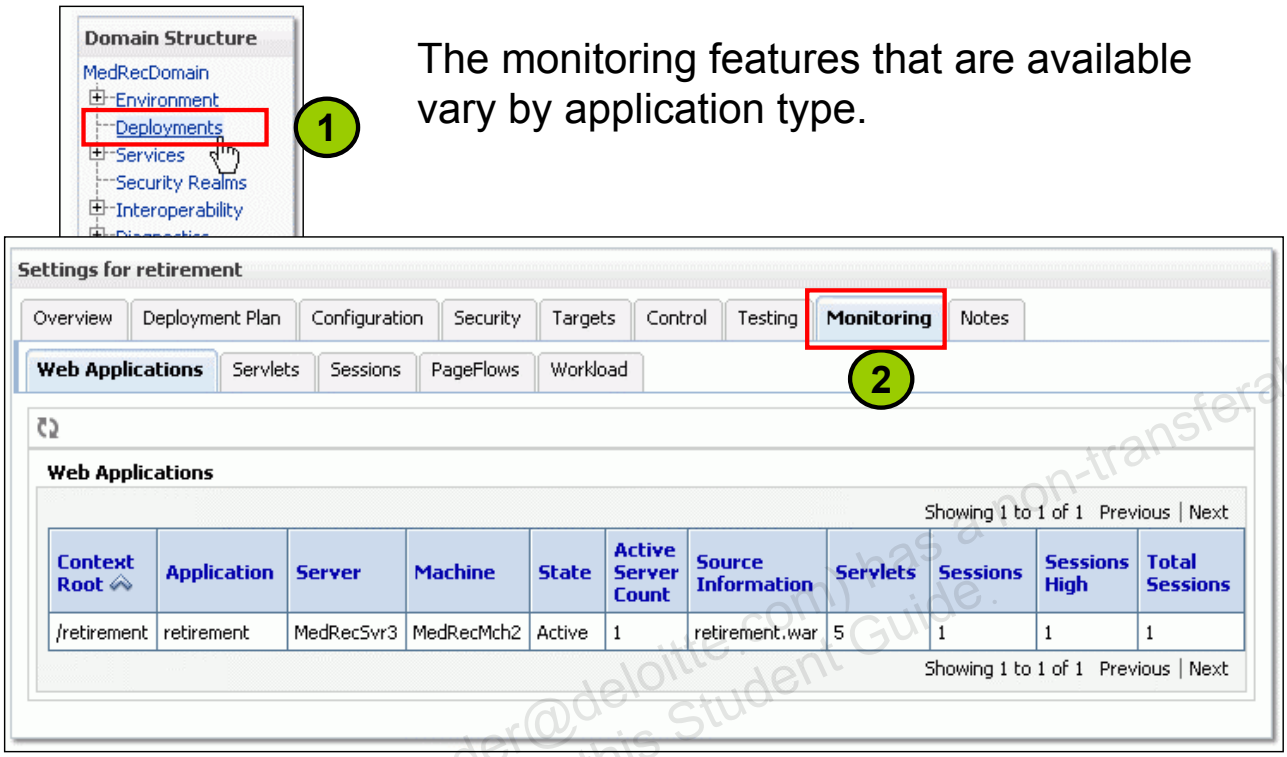
Some deployment descriptor elements are editable via the console.

The Oracle WebLogic Server Administration Console interface has been streamlined to make production-level deployment descriptors available for editing by administration users. Although full deployment descriptor editing is no longer available in the Administration Console, many WebLogic-specific descriptor elements are directly editable via the Administration Console fields. You can edit these descriptors without repackaging and redeploying the associated module.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (ccmccrudder@deloitte.com) has a non-transferable license to use this Student Guide

Monitoring an Application



The monitoring features that are available vary by application type.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Monitoring an Application

The screenshot shows some of the application-monitoring columns. There are more columns that are not shown.

1. In the left pane of the Administration Console, click Deployments. In the right pane, click the application that you want to monitor.
2. Click the **Monitoring** tab. The available subtabs will vary depending on the type of the application. For example, a Web application or module has tabs named Web Application, Servlets, Sessions, and Workload.

For Web applications, the available columns include:

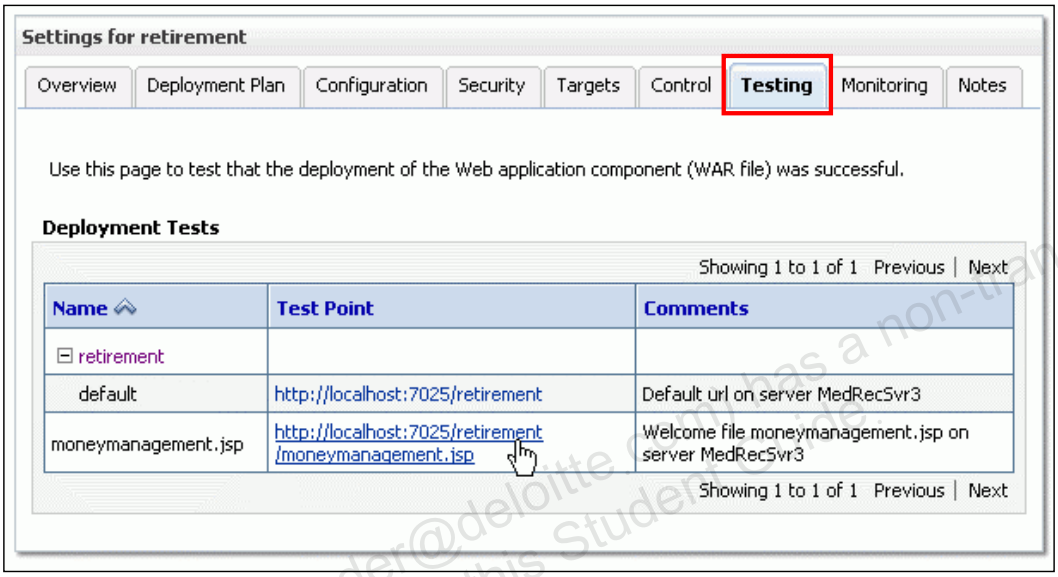
- **Context Root:** Returns the context root (context path) for the Web application
- **Servlets:** Is the number of Java servlets that are deployed within this application, including the internal WebLogic servlets. If required, the Servlets subtab displays statistics on a per-servlet basis.
- **Sessions:** Provides a count of the current number of open sessions in this module
- **Sessions High:** Provides the highest number of active sessions ever managed by this application. The count starts at zero each time the application is activated.
- **Total Sessions:** Provides a count of the total number of sessions opened since the application was deployed

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Application Testing

You can test a deployed application using the Administration Console.



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Application Testing

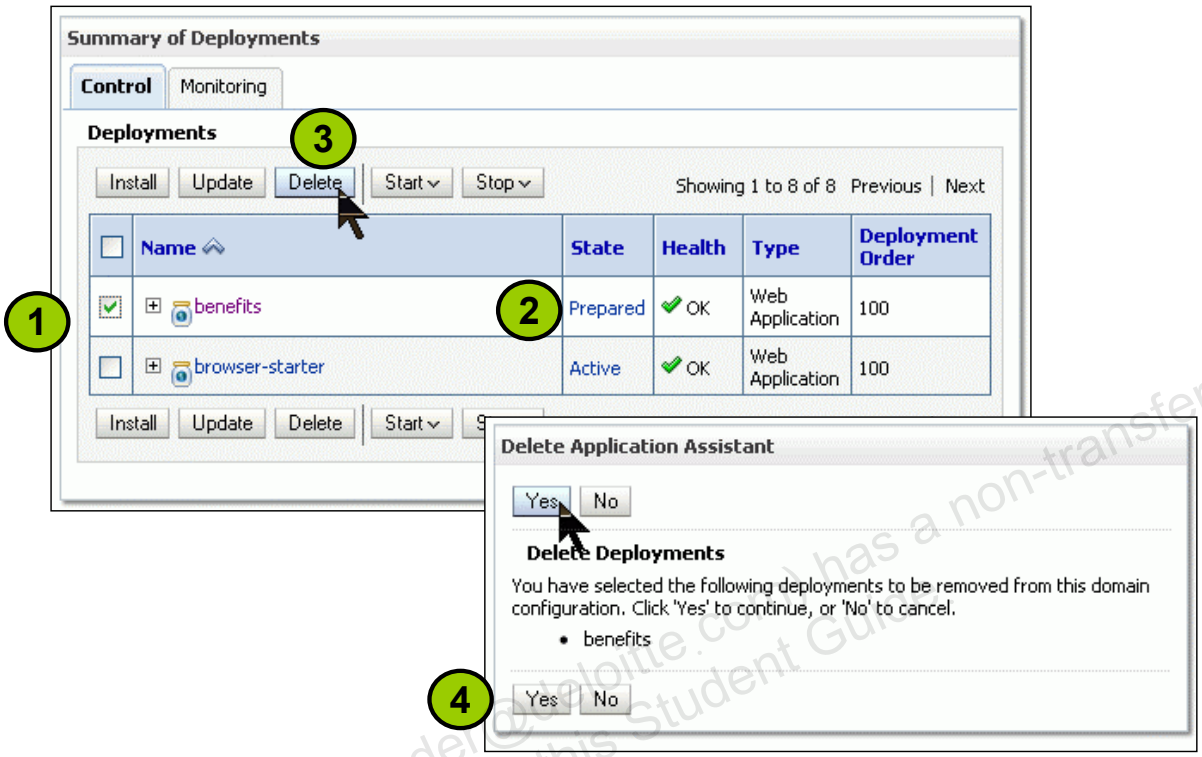
The screenshot shows two ways to test an application. Clicking either of the test links opens a Web browser tab to check the application. The welcome file tag points to the page that starts the application, be it `index.html`, or `welcome.jsp`, or in this case, `moneymanagement.jsp`.

From `retirement/WEB-INF/web.xml`:

```
<welcome-file-list>
  <welcome-file>moneymanagement.jsp</welcome-file>
</welcome-file-list>
```

It knows the port number because 7025 is the listen port on MedRecSvr3.

Deleting Applications



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deleting Applications

Using the console, you can either update or redeploy applications after configuration or component changes, or delete or undeploy them. All concurrent deployment activities are tracked by the administration server in a series of tasks:

- Task progress and outcome can be queried for each application.
- Reasons for failure are logged.

To delete an application, perform the following steps:

1. Select the application check box in the Deployments window.
2. Make sure that the application is not Active. Stop the application if necessary. The Prepared state as shown in the slide is okay for deleting.
3. Click **Delete**.
4. Confirm by clicking **Yes**.

Command-Line Deployment

- The `weblogic.Deployer` utility enables you to perform deployment operations similar to those available in the console.
- `weblogic.Deployer` actions can also be scripted with the Ant task `wldeploy`.

`weblogic.Deployer` Syntax:

```
% java weblogic.Deployer [options]
        [-deploy|-undeploy|-redeploy|-start|-stop|-listapps]
                [file(s)]
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Command-Line Deployment

The `weblogic.Deployer` utility is a Java-based deployment tool that provides a command-line interface to deployment tasks. It is an advanced tool that provides you with greater configuration flexibility through the command-line deployment. It enables you to deploy a new application, update an existing application, or undeploy an application. You can also deploy the JMS, JDBC, and WebLogic Diagnostic Framework (WLDF) modules.

If Autodeploy is ON in Development mode, using `weblogic.Deployer` will also target the application to the administration server or to the managed servers if it is a redeployment.

The following slide shows a few examples of using this utility. Note that it is not possible to target only an application. The application must be undeployed and redeployed in order to change its target.

Deployment with `weblogic.Deployer`

- Prepare and deploy a new application:

```
java weblogic.Deployer -adminurl t3://adminserver:7001
   -username myuser -password welcome1 -name HRServices
   -source /usr/HRServices.ear -targets serverA -deploy
```

- Redeploy an application:

```
java weblogic.Deployer -adminurl t3://adminserver:7001
   -username myuser –password welcome1 –name HRServices
   -redeploy
```

- Undeploy an application:

```
java weblogic.Deployer -adminurl t3://adminserver:7001
   -username myuser –password welcome1 –name HRServices
   -undeploy
```

- To list all deployed applications:

```
java weblogic.Deployer -adminurl t3://localhost:7001
   -username myuser -password welcome1 -listapps
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deployment with `weblogic.Deployer`

You can use the `setWLSEnv.sh` or `setWLSEnv.cmd` script that is located in the `server/bin` subdirectory of the Oracle WebLogic Server installation directory to set the required environment to run `weblogic.Deployer`. The general syntax for invoking `weblogic.Deployer` is the following:

`java [`*`SSL Arguments`*`] weblogic.Deployer [`*`Connection Arguments`*`]`

`[`*`User Credentials Arguments`*`] `*`COMMAND-NAME command-options`*` [`*`Common Arguments`*`]`

The available commands include:

- `deploy:` Deploys or redeploys an application, library, or module. Specify the application archive or path along with a comma-separated list of targets. Many options are supported, including the staging mode, version, deployment plan, and various timeouts. You can also use this command to upload the application remotely to the administration server's file system.
- `distribute:` Prepares deployment files for deployment by copying deployment files to target servers and validating them. A distributed application can be started quickly with the `start` command.
- `redeploy:` Redeploys a running application or part of a running application
- `start:` Makes a stopped (inactive) application available to clients on target servers. The `start` command does not redistribute deployment files to target servers.

More `weblogic.Deployer` Examples

- To list all deployment tasks:

```
java weblogic.Deployer -adminurl t3://localhost:7001
   -username system -password welcome1 -listtask
```

- To cancel a deployment task:

```
java weblogic.Deployer -adminurl t3://localhost:7001
   -username system -password welcome1 -cancel -id tag
```

```
[oracle@wls-sysadm]$ java weblogic.Deployer
    -adminurl t3://localhost:7020
    -username weblogic -password Welcome1 -listapps
weblogic.Developer invoked with options:
-adminurl t3://localhost:7020 -username weblogic -listapps
 jsf [LibSpecVersion=1.2,LibImplVersion=1.2.9.0] <ACTIVE VERSION>
 jstl [LibSpecVersion=1.2,LibImplVersion=1.2.0.1] <ACTIVE VERSION>
 medrec
Number of Applications Found : 3
[oracle@wls-sysadm]$
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

More `weblogic.Deployer` Examples

Option	Description
`-nowait`	Allows multiple tasks
`-timeout`	Specifies the time to wait for a task
`-source`	Specifies the location of the file to deploy
`-name`	Names the application to deploy
`-targets`	Specifies the list of servers as the targets for application deployment
`-id`	Names the task
`-activate`	Activates the application or component
`-deactivate`	Deactivates the application or component
`-unprepare`	Deactivates and unloads classes for the application
`-remove`	Removes the application from the server
`-cancel`	Cancels the task `<id>`
`-list <id>`	Specifies the status of the task `<id>`
`-deploy`	Is a convenient alias for `-activate`
`-undeploy`	Is a convenient alias for `-unprepare`

Deploying Applications with WLST

WLST provides a number of deployment commands. You can use these commands to:

- Perform life-cycle operations on applications and stand-alone modules in an Oracle WebLogic Server instance
  - Deploy
  - Undeploy
  - Redeploy
- Update an existing deployment plan
- Start and stop a deployed application

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deploying Applications with WLST

Syntax for WLST deployment:

```
deploy(appName, path, [targets], [stageMode], [planPath], [options])
```

- `appName`: Name of the application or stand-alone Java EE module to be deployed
- **`path`:** Name of the application directory, archive file, or root of the exploded archive directory to be deployed
- `targets` (optional)**:** Comma-separated list of the target. Each target may be qualified with a Java EE module name (for example, *`module1@server1`*) thus enabling you to deploy different modules of the application archive on different servers. This argument defaults to the server to which WLST is currently connected.
- `stageMode` (optional)**:** Staging mode for the application that you are deploying. Valid values are `stage`, `nostage`, and `external_stage`. This argument defaults to null.
- `planPath` (optional)**:** Name of the deployment plan file. The file name can be absolute or relative to the application directory. This argument defaults to the `plan/plan.xml` file in the application directory, if one exists.
- `Options` (optional)**:** Comma-separated list of deployment options, specified as name/value pairs

The `listApplications()` command of WLST lists all the applications that are currently deployed to the domain.

Deploying an Application with WLST

Deploy an application (`deployapp.py`):

```
##
# WLST script for Deploying Java EE Application #
##

# Connect to the server
print 'Connecting to server  .... '
connect('weblogic','welcome1','t3://localhost:7001')

appname = "mbeanlister"
applocation = "c:/domains/MedRecDomain/apps/mbeanlister"

# Start deploy
print 'Deploying application ' + appname
deploy(appname, applocation, targets='myserver',
       planPath='c:/myapps/plan/plan.xml')
print 'Done Deploying the application '+ appname
exit()
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@dell.com) has a non-transferable license to use this Student Guide.

Deploying an Application with WLST

The example in the slide shows how to deploy an application using the WLST scripts. You can invoke the Deploy method by passing the application name, type, and target server as arguments.

The `deploy()` command must be either on one line or must be indented to indicate to Jython that it is a continuation of the previous line.

Deployment with WLST

Prepare and deploy a new application, or redeploy an existing one:

```
connect('myuser','mypass1','t3://adminserver:7001')
name = "HRServices"
location = "/usr/myapplications/HRServices.ear"

deploy(name, location, targets='serverA')
```

Other WLST deployment commands:

```
distributeApplication(location, targets='serverA')
startApplication(name)
redeploy(name)
stopApplication(name)
listApplications()
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Deployment with WLST

The WLST deployment commands that are available include:

- `Deploy`: Deploys an application or library to an Oracle WebLogic Server instance, and returns a `WLSTProgress` object that you can access to check the status of the command. Many options are supported, including the staging mode, version, deployment plan, and various timeouts. You can also use this command to upload the application remotely to the administration server's file system.
- `distributeApplication`: Copies the deployment bundle to the specified targets. The deployment bundle includes modules, configuration data, and any additional generated code. The `distributeApplication` command does not start deployment.
- `startApplication`: Starts an application, making it available to users. The application must be fully configured and available in the domain.
- `Redeploy`: Reloads classes and redeploys a previously deployed application
- `stopApplication`: Stops an application, making it unavailable to users. The application must be fully configured and available in the domain.
- `listApplications`: Lists all applications that are currently deployed in the domain
- `Undeploy`: Undeploys an application from the specified servers

Road Map

- Deployment concepts
- Development deployment
  - Autodeployment
  - FastSwap
- Front-end with a Web server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

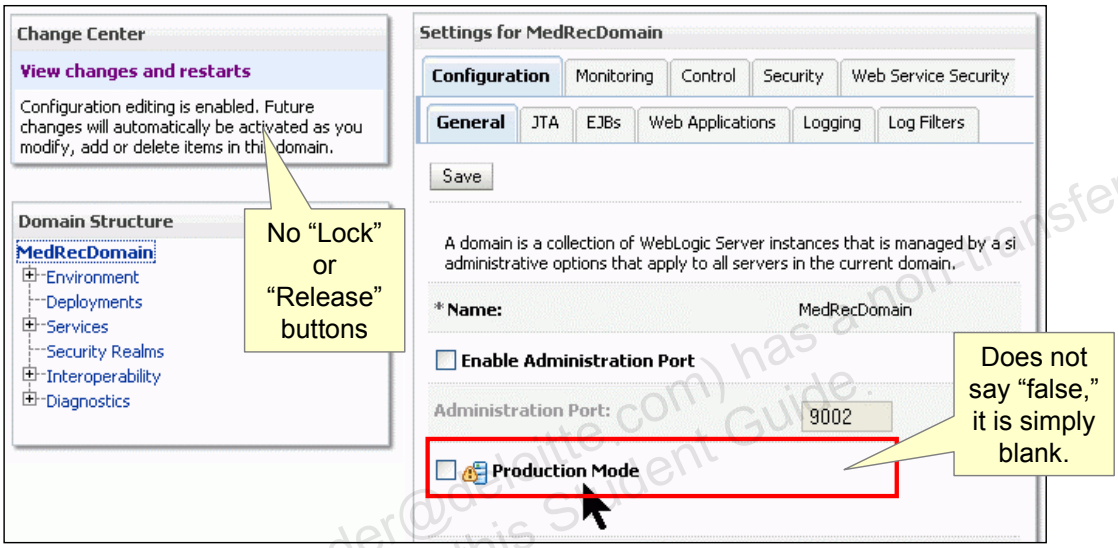
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Production Mode Flag

When Production Mode is disabled, applications can be dynamically deployed.

- An application poller is enabled in development mode.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Production Mode Flag

The screenshot shows that the Production Mode for the domain is blank, which means that it is in development mode. The mode is set for all Oracle WebLogic Servers in a given domain.

The autodeployment feature of development mode checks the applications folder every three seconds to determine whether there are any new applications or changes to the existing applications, and then dynamically deploys these changes.

The autodeployment feature is enabled for servers that run in development mode.

Production Mode applies to all Oracle WebLogic Server instances in a given domain. It specifies whether all servers in this domain run in Production Mode. When Production Mode is enabled, this can be disabled only in the administration server startup command line. If you start the administration server with the `-Dweblogic.ProductionModeEnabled=false` option, it will start as development mode, but the console will still reflect the value of `<production-mode-enabled>` in `config.xml` and say that it is enabled in Production Mode when in fact it is not.

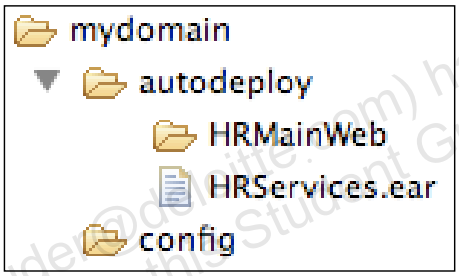
Note that the command-line option `-Dweblogic.ProductionModeEnabled=`**`true`** is deprecated in Oracle WebLogic Server 10.3, but `false` is still supported.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Autodeployment

- By default, the autodeployment feature is enabled only if the domain is *not* running in Production Mode.
- When enabled:
  - The administration server monitors its "autodeploy" folder for new, updated, or removed applications
  - Applications are targeted only to the administration server
  - Developers can quickly test or experiment with an application
- **`<WL_HOME>/user_projects/domains/`*`domain`*`/autodeploy`**



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carl.mccruder@dellwitte.com) has a non-transferable license to use this Student Guide.

Autodeployment

Autodeployment is a method for quickly deploying an entire application (exploded or jarred) to a stand-alone server (administration server) for evaluation or testing. You should use this method only in a single-server development environment—for example, developing on the administration server. You can run an Oracle WebLogic Server domain in two different modes: development and production. You can use the autodeployment feature only in development mode.

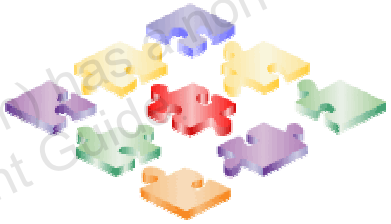
If autodeployment is enabled, when an application is copied into the *`domain`*`/autodeploy` folder of the administration server, the administration server detects the presence of the new application and deploys it automatically (if the administration server is running). If Oracle WebLogic Server is not running when you copy the application to the *`domain`*`/autodeploy` directory, the application is deployed the next time the Oracle WebLogic Server administration server is started. Autodeployment deploys only to the administration server.

A deployment unit that was autodeployed can be dynamically redeployed while the server is running. To dynamically redeploy, copy the new version of the archive file over the existing file in the *`domain`*`/autodeploy` directory. When an application is autodeployed in the exploded archive format, the administration server periodically looks for a file named `REDEPLOY` in the exploded application directory. If the time stamp on this file changes, the administration server redeploys the exploded directory. By default, it checks for new time stamps every three seconds.

Autodeploying Using an Expanded Directory

If the following conditions are true, you are a candidate for autodeploy. Consider autodeploy if the application is:

- In the development phase
- Being updated frequently
- Deploying to a single WLS server
  (for example, only the administration server)



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl MCGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Autodeploying Using an Expanded Directory

If you are working in a single server environment (deploying only to the administration server, probably in development mode), you can use directory-based autodeployment. That is, you can simply place the application in the `domain/autodeploy` directory to deploy it.

- You can change JSPs or any static data files in the application under the `domain/autodeploy` directory, and view your changes in the application.
- You can also directly compile updated servlet classes into a Web application under the `domain/autodeploy` directory (for instance, `domain/autodeploy/expanded_webapp_dir/WEB-INF/classes`). The new classes are incorporated into the Web application; no additional steps are required. Autodeploy is independent of the expanded directory, but they are often used together by developers.
- To incorporate the changes made to the Web application deployment descriptors, modify the `WEB-INF/REDEPLOY` file in the `domain/autodeploy` directory, so that the autodeployer detects the change. This redeploys the Web application.

If you are working in a multiple server environment (probably in Production Mode), you should use the `weblogic.Deployer` tool to deploy applications.

If you have made any changes to the application files, you must communicate these changes to the server using the `weblogic.Deployer` tool. This allows the changes to be incorporated into the deployed application.

FastSwap and On-Demand Deployment

- WebLogic's FastSwap feature is:
  - Enabled using the WebLogic deployment descriptors
  - Available only if the domain is *not* running in Production Mode
  - Applicable only to Web applications that are *not* archived
- When enabled:
  - WebLogic automatically reloads the modified Java class files within applications
  - Developers can perform iterative development without an explicit redeployment
- On-demand deployment:
  - **`weblogic.xml`:**
    `<fast-swap>true</fast-swap>`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

FastSwap and On-Demand Deployment

With FastSwap, the individual Java classes are redefined in place without reloading the classloader, thereby having the decided advantage of fast turnaround times. This means that you do not have to wait for an application to redeploy, and then navigate back to wherever you were in the Web page flow. Instead, you can make your changes, compile, and then see the effects immediately.

FastSwap is supported only when WebLogic Server is running in development mode. It is automatically disabled in Production Mode. Only changes to class files in exploded directories are supported. Modifications to class files in archived applications, as well as archived JAR files that appear in the application's classpath, are not supported. Within an exploded Web application, modifications to Java classes are supported only in the `WEB-INF/classes` directory.

To enable FastSwap in your application, add the `<fast-swap>` element to the `weblogic-application.xml` file for an enterprise application, or to the `weblogic.xml` file for a stand-alone Web application.

For headless applications (that is, applications that are not fronted by a Web application), class redefinition can be explicitly initiated using Oracle WebLogic Server's JMX interface. For convenience, the following Ant task that uses this JMX interface is also available: `com.bea.wls.redef.ant.FastSwapTask`.

FastSwap and On-Demand Deployment (continued)

On-Demand Deployment

There are many internal applications that are deployed during startup. These internal applications consume memory and require CPU time during deployment. This contributes to the Oracle WebLogic Server startup time and base memory footprint. Because many of these internal applications are not needed by every user, Oracle WebLogic Server has been modified to wait and deploy these applications on the first access (on demand) instead of always deploying them during server startup. This reduces the startup time and memory footprint.

There are two different types of internal applications. The first type displays a user interface, and includes the Administration Console, UDDI explorer, and WLS test client. The second type does not display a user interface, and includes UDDI and internal servlets for deployment and management file distribution.

For applications with a user interface, WLS displays a status page indicating that on-demand deployment is in progress. This page refreshes every two seconds. When the internal application completes deployment, you are redirected to the internal application. This status page is displayed on the first access of each application. Subsequent invocations do not deploy the application and go directly to the user interface for the internal application.

Configuring On-Demand Deployment

For a development domain, the default is for WLS to deploy internal applications on demand. For a production-mode domain, the default is for WLS to deploy internal applications as part of server startup. You can control the default behavior by configuring the `InternalAppsDeployOnDemandEnabled` attribute in the Domain MBean. You can change the configuration setting by using the Administration Console or by using the WebLogic Scripting Tool (WLST).

Configuring On-Demand Deployment Using the Administration Console

To change the `InternalAppsDeployOnDemandEnabled` attribute using the Administration Console, perform the following steps:

1. Start an edit session with the Lock & Edit button.
2. Select the domain to display the Configuration > General tab.
3. Change the setting of the “Enable on-demand deployment of internal applications” check box.
4. Click Save, and then click Activate Changes to activate the changes that will take effect when you restart WLS.

Configuring On-Demand Deployment Using WLST

To change the `InternalAppsDeployOnDemandEnabled` attribute using the WLST, execute the following commands:

- `connect()`
- `edit()`
- `startEdit()`
- `cmo.setInternalAppsDeployOnDemandEnabled(false)`
- `activate()`

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCulloch (camcg@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

- Deployment concepts
- Development deployment
- Front-end with a Web server
  - Web servers defined
  - HTTP
  - Static and dynamic content
  - Redirection

ORACLE

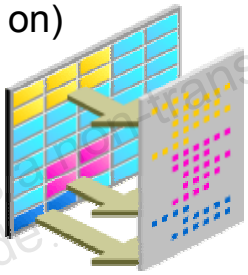
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Role of Web Servers

- Web servers are responsible for handling HTTP requests from clients.
- Web servers typically return:
  - Static content (HTML pages, graphics, media, and so on)
  - Dynamic content (generated by servlets, JSPs, CGIs, JSF, Struts, and so on)



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

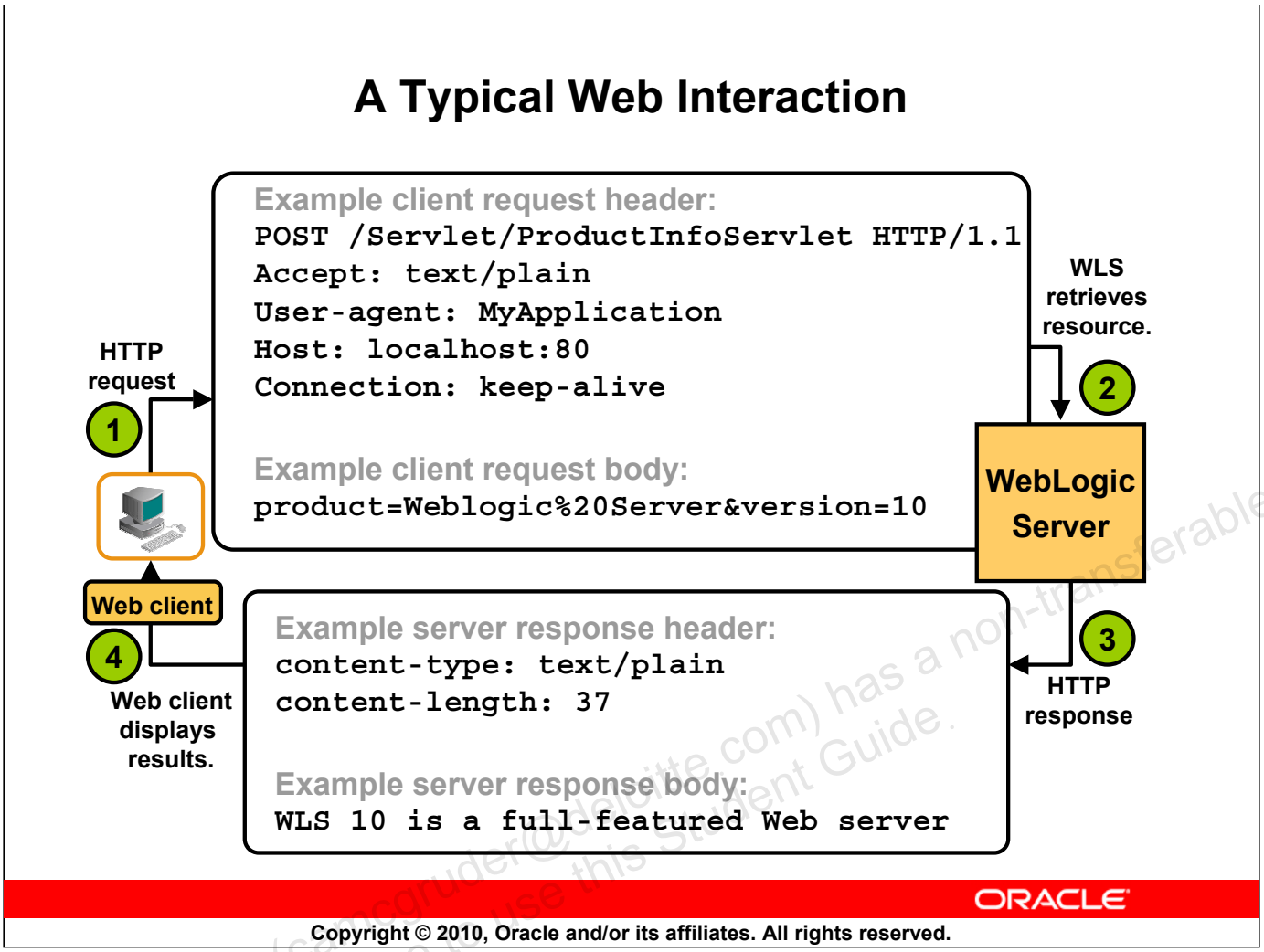
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Role of Web Servers

A Web server is responsible for handling HTTP requests from clients. HTTP is a stateless protocol. This means that for each request, a connection between the client and the server is established, the request is handled, and then the connection is closed. For example, there is nothing in HTTP to allow a server to treat a particular request differently depending on how many times that request has been made. Such behavior can, however, be simulated with the use of cookies, which are discussed later.

Oracle WebLogic Server is a fully functional Web server that can handle high-volume Web sites, serving static files such as HTML files and image files, as well as serving dynamic content using servlets, JavaServer Pages (JSP), and common gateway interfaces (CGIs). Oracle WebLogic Server supports the HTTP 1.1 specification. Nevertheless, you may want to add an Oracle HTTP Server as a Web tier to make a Demilitarized Zone (DMZ) to separate the exposure of sensitive data and applications to the public.



Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCluder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

A Typical Web Interaction

The HTTP protocol separates each transmission into a header and a body. Both the request from the client browser and the response from the Web server contain separate header and body sections. On the server, the header information is used to determine what resource is required as well as how best to respond to the request—for example, the content type and the protocol version to use—because different clients may use different browsers. On the client, the header is normally stripped off by the browser. The browser uses the header information to determine how to interpret the body.

- **Request header:** The first section contains the request method followed by its parameters. In the example in the slide, the `POST` method is being used to make a request for a servlet on the server with the path "`/Servlet/ProductInfoServlet`" using HTTP 1.1. The second section specifies the type of content that the client can accept—in this case, plain text. The format used for this is Multipurpose Internet Mail Extensions (MIME). The next two sections of the request header specify the client identity and server name, respectively. The final section specifies the connection type. HTTP 1.1 supports a keep-alive option to allow a single request and response for a page that may contain several other resources.

A Typical Web Interaction (continued)

- **Response header:** The content-type response header field is used to indicate to the client how to interpret the body of the message. In HTTP 1.1, it is necessary for the server to determine the length of the body in the response in order to send it back to the client. The content-length field is used for this purpose. The client requires this because it has no way of determining the end of the body. Instead, the client simply reads characters from the HTTP socket being used in the connection until it has read the number of characters specified by content length. Other fields can also be returned, such as a status code, which is useful for the client when a request cannot be serviced, and cookies, which can simulate sessions by having the client reuse the cookie that was sent back from the server the next time it makes a request to that server.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

MIME Types

- MIME is a protocol for identifying and encoding binary data.
- All HTTP response data is encoded with a MIME content type.
- Browsers interpret HTTP response data differently depending on the MIME type of the data:
  - HTML pages are parsed and displayed.
  - PDF documents can be sent to Adobe Acrobat.
  - Application code can be directly executed.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

MIME Types

Web browsers are capable of interpreting HTTP responses with data other than HTML. All HTTP responses that are sent out over the Web are encoded with a MIME type that specifies the "type" of data that is contained within the HTTP packet. Browsers are configured with embedded plug-ins that tell the browser how to react to the data contained within an HTTP response depending on its content type. When browsers receive packets with the text/`html` MIME encoding, they process the data in the response as HTML. If the MIME type is configured to be image/`gif`, the browser knows that the content contained within the HTTP packet is binary image data.

Oracle WebLogic Server automatically "tags" all outgoing HTTP packets with a special MIME type. Oracle WebLogic Server does this tagging by matching a file extension to the request made by a client application. If a client makes an HTTP request that has a file extension that maps to a registered Oracle WebLogic Server MIME entry, Oracle WebLogic Server sets the MIME content type of the outgoing response to be the type specified in the `weblogic.xml` file via the Administration Console.

HTTP Status Codes

- HTTP status codes indicate to the client whether or not the request was successful, and if not, provide the client a reason for a failed request.
- They are used by clients to provide alternate behavior.
- Some representative codes are:
  - Indicating success:
    - The default status code is 200, which indicates success.
  - Reason for failure:
    - A status code of 404 tells the client the requested resource was not found.
  - Providing alternate behavior:
    - If a browser receives a 401 status code, the browser prompts the user for an ID and password to log in.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

HTTP Status Codes

WebLogic Server provides standard HTTP status codes for Web requests. Status codes are useful in isolating problems with the server. For example, it is possible to search logs for failures in order to fix problems with resource access or with the resources themselves.

Some common status codes are:

- SC_OK 200
- SC_NO_CONTENT 204
- SC_MOVED_PERMANENTLY 301
- SC_MOVED_TEMPORARILY 302
- SC_UNAUTHORIZED 401
- Request for Resources Forbidden 403
- SC_NOT_FOUND 404
- SC_INTERNAL_SERVER_ERROR 500
- SC_NOT_IMPLEMENTED 501
- SC_SERVICE_UNAVAILABLE 503

Static Content

- Static content documents are predefined on the server and do not change.
- Oracle WebLogic Server can be used to serve static content such as:
  - HTML documents
  - Images
  - Media
  - PDF documents
- Oracle WebLogic Server can serve static documents:
  - Over standard HTTP
  - Through SSL using HTTPS

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

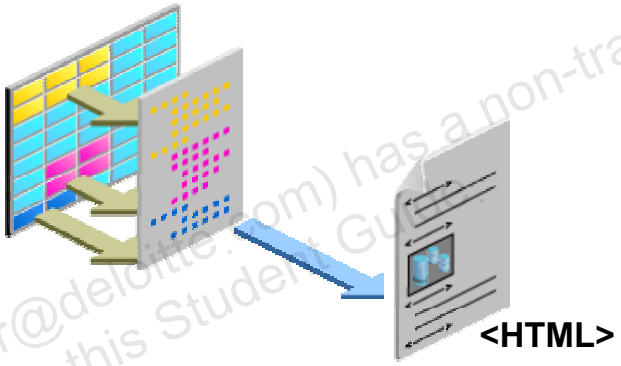
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Static Content

The mechanism behind how Oracle WebLogic Server serves static content is covered in the slides that follow. Currently, it is important to distinguish between static content and dynamic content. Static content involves predefined documents that are sent to the client based on request. Static documents, such as HTML and PDF documents, do not change based on the client's request; they are predefined documents that reside on the server. Secure sockets layer (SSL) uses certificates from a trusted authority to verify that the parties are who they assert that they are.

Dynamic Content

- Dynamic content documents may change based on the client's request.
- Dynamic content often involves a database query.
- HTML documents can be created using various means including:
  - Servlets
  - CGI programs
  - JSPs
  - JSF
  - Struts



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

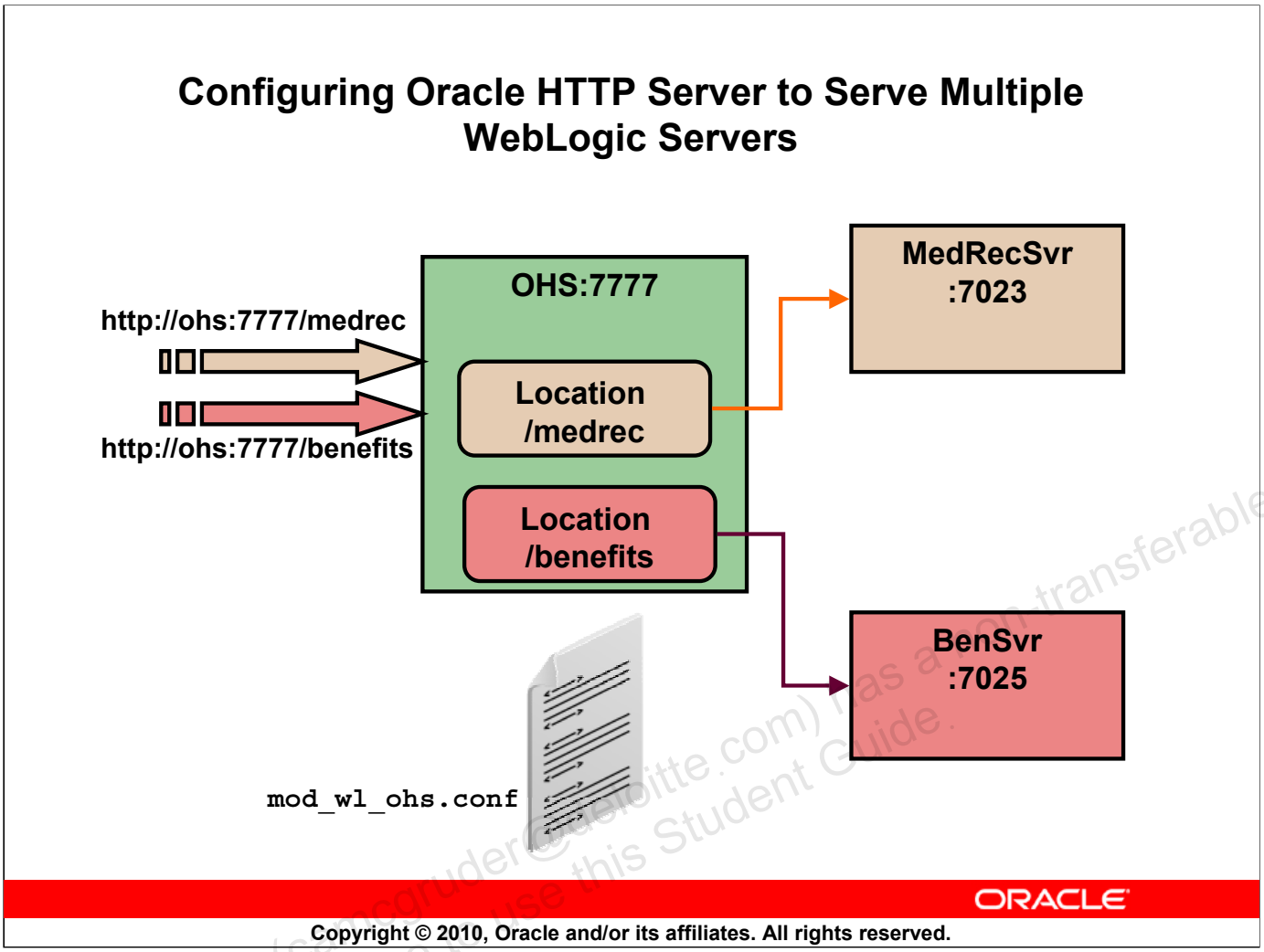
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Dynamic Content

Java servlets and JSPs are accessed using a request with a URL in the same way that static files are accessed.

With dynamic content generation, the resource requested in the URL actually refers to a "program" that is run. The output of this program is simply HTML that gets sent back from the server to the client. The client is unaware of whether the content received is from a static file or is generated. A simple example of a dynamically generated page might be a page that returns the name of the user that sent the request. The key distinguishing factor from static pages is that the dynamic page returned includes information based on input, which can be anything that the user entered or information about the environment of the user. Often, the dynamic page fetches rows from a database.



Configuring Oracle HTTP Server to Serve Multiple WebLogic Servers

You can configure one Oracle HTTP Server (OHS) to front-end multiple WebLogic servers that are serving different applications. The slide shows an OHS configured to service requests for `medrec` and `benefits` applications that are served by two different WebLogic servers.

`mod_wl_ohs.conf` for such a configuration sample is shown here:

```
# NOTE : This is a sample mod_wl_ohs.
LoadModule weblogic_module
"${ORACLE_HOME}/ohs/modules/mod_wl_ohs.so"
<IfModule mod_weblogic.c>
     MatchExpression *
</IfModule>
<Location /medrec>
     SetHandler weblogic-handler
     WebLogicHost wls-sys1
     WebLogicPort 7023
</Location>
<Location /benefits>
     SetHandler weblogic-handler
     WebLogicHost wls-sys2
     WebLogicPort 7025
</Location>
```

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

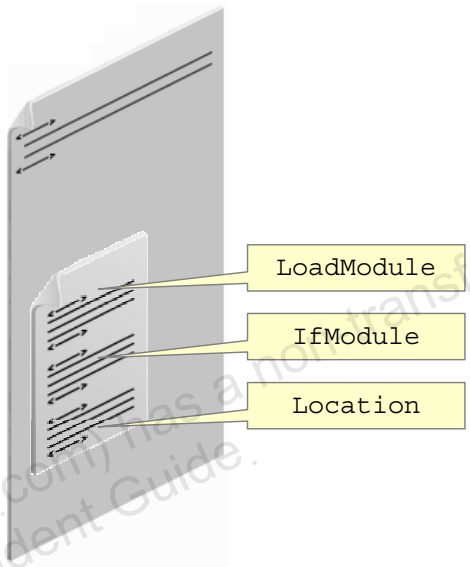
Can McCruder (cancrguder@deloitte.com) has a non-transferable license to use this Student Guide.

mod_wl_ohs.conf

The main sections of `mod_wl_ohs.conf` are:

- `LoadModule`: Enabled by default to load the weblogic_module when OHS starts
- `IfModule`: Specifies the host and port details of the WLS server or cluster
- `Location`: Specifies the root context of the application and advises OHS that WLS will handle requests for that application

Embedded inside of `httpd.conf`:



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

mod_wl_ohs

The `mod_wl_ohs.conf` file, located in `$INSTANCE_HOME` (for example, `/u01/app/work/instances/config/OHS/ohs1`), is used to configure the Web Server proxy plug-in of Oracle WebLogic Server. It is embedded at the end of Apache's `httpd.conf` in the same directory.

Here is a sample of `mod_wl_ohs.conf`:

```
# NOTE : This is a sample mod_wl_ohs.
LoadModule weblogic_module
"${WT_HOME}/ohs/modules/mod_wl_ohs.so"
<IfModule mod_weblogic.c>
      WebLogicHost wls-sysadm
      WebLogicPort 7023
      MatchExpression *
</IfModule>
<Location /medrec>
      SetHandler weblogic-handler
</Location>
```

If you configure multiple servers in a cluster, you should use the WebLogicCluster directive instead of the WebLogicHost and WebLogicPort directives, as a sample:

```
WebLogicCluster wls-sysadm:7025, wls-sysadm:7027
```

Verifying Ports Used by OHS

If OHS is running, you can verify ports using the `opmnctl status` command with the `-l` option:

```
[oracle@edvmr1p0 u01]$ cd /u01/app/work/instances/bin
[oracle@edvmr1p0 bin]$ ./opmnctl status -l

Processes in Instance: webtier
--------------+--------------+------+--------+-----------+---------+----------+----
ias-component | process-type |  pid | status |       uid | memused |   uptime |
     ports
--------------+--------------+------+--------+-----------+---------+----------+----
ohs1          | OHS          | 4253 | Alive  | 559158019 |  358996 | 18:11:13 |
     https:9999,https:4443,http:7777

[oracle@edvmr1p0 bin]$
```

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Verifying Ports Used by OHS

After you have configured OHS to front-end the applications, you can access the applications through OHS. To access OHS, you would need to get the port being used by OHS. You can get the list of ports used by OHS by using the `opmnctl status` command with the `-l` option (lowercase letter "L") as shown in the slide.

You can see the port configuration in the `ports.prop` file:

```
[oracle@wls-sysadm]$ cd /u01/app/work/instances
[oracle@wls-sysadm]$ cd config/OPMN/opmn/
[oracle@wls-sysadm]$ more ports.prop
#
#Thu Mar 26 11:38:10 EDT 2010
/opmn/remote_port=6701
/ohs1/ProxyPort=9999
/opmn/local_port=6700
/ohs1/ListenPort=7777
/ohs1/SSLPort=4443
[oracle@wls-sysadm]$
```

This `ports` file is automatically created as a result of configuring the `mod_wl_ohs.conf` file.

Quiz

Which environment supports FastSwap?

a. Production Mode, archived files
b. Production Mode, expanded files
c. Development mode, archived files
d. Development mode, expanded files
e. All of the above

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: d

You must also have `<fast-swap>true</fast-swap>` in the `weblogic.xml` file.

Quiz

What is the `web.xml` tag indicating the test point for testing applications?

a. `<welcome-file>`
b. `<test-point>`
c. `<deploy>`
d. `<monitor>`
e. `<debug>`

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: a

The `<welcome-file>` pair of tags is within a `<welcome-file-list>` pair of tags.

Quiz

It is possible to deploy an exploded directory that contains JAR files.

a. True
b. False

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

You must manually explode anything tarred, zipped, jarred, or otherwise archived in an exploded directory before it can be used.

Summary

In this lesson, you should have learned how to:

- Enable autodeploy with manual deployment
- Configure and deploy Web applications via the Administration Console, command line, and WLST
- Redeploy and remove applications
- Configure deployment descriptors
- Test deployed applications
- Front-end deployed applications with a Web server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Summary

There is no practice for this lesson.

12

Deploying Java EE Applications

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Describe Java EE Web applications
- Describe Enterprise Deployment Architectures
- Package Web applications in several forms
- Define Web application structure and Web application archive
- Explain why WebLogic augments standard Java EE deployment descriptors with `weblogic*.xml` files
- Look at deployment descriptors `web.xml` and `weblogic.xml`
- Describe URLs and Web applications

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

You plan to deploy the MedRec application to your application servers. It will be Web based, with clients accessing the medical records and other doctor/patient information via browsers. Some of the information is relatively static (office hours, directions, staff, and so on, changing every few months), whereas other bits of information are dynamic (billing, scheduling, test results, and so on, changing every few minutes). Some of the applications will be deployed in all the offices, but some of the applications need to be deployed only into the billing office location. This may affect which servers host which applications, though that may impact availability and failover. You need to understand the business environment in which your application will be deployed.

Road Map

- Web applications
  - Web applications
  - Directory structure and deployment descriptors
  - Using the Console to deploy Web applications
  - Monitoring Web applications
- EJB applications
- Enterprise applications

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java EE Web Applications

- Web application:
  - Responds to client requests using the HTTP protocol
  - Typically implements an interactive Web site
- The contents of a Web application can include:
  - Java servlets
  - JavaServer Pages (JSP) for dynamic content
  - Static content (HTML, CSS, images, and so on)
  - Java classes and libraries
  - Client-side libraries (JavaScript, Java Applets, and so on)
  - XML deployment descriptors:
    - Standard (`web.xml`)
    - WebLogic specific (`weblogic.xml`)

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java EE Web Applications

A Web application on Oracle WebLogic Server includes the following files:

- Servlets, JSP, and other Java classes
- Optionally, a `web.xml` deployment descriptor, which is a Java EE standard XML document that describes the contents of a Web Archive (WAR) file. The `web.xml` file can be quite short, as short as a half dozen lines (or zero lines!), or as long as hundreds of lines. Examples of this file are shown in a few slides from now.
- Optionally, a `weblogic.xml` deployment descriptor, which is an XML document containing Oracle WebLogic Server–specific elements for Web applications. Examples of this file are shown in a few slides from now.
- HTML and XML pages with supporting files, such as images and multimedia files

A servlet is a Java class that runs on a Java-enabled server, handles an HTTP request, and provides an HTTP response, usually in the form of an HTML page. The most common use of HTTP servlets is to create interactive applications using standard Web browsers for the client-side presentation. HTTP servlets can access databases, Enterprise JavaBeans (EJBs), messaging APIs, HTTP sessions, and other facilities of Oracle WebLogic Server.

Java EE Web Applications (continued)

JavaServer Pages are Web pages coded with an extended HTML that makes it possible to embed Java code on a Web page. JSP can call custom Java classes, known as tag libraries, using HTML-like tags. The `appc` compiler compiles JSP and translates them into servlets. Oracle WebLogic Server automatically compiles JSP if the servlet class file is not present or is older than the JSP source file. You can also precompile JSP and package the servlet class in a WAR file to avoid compiling in the server.

Note: The `appc` compiler generates and compiles the classes needed to deploy EJB and JSP to Oracle WebLogic Server. It also validates the deployment descriptors for compliance with the current specifications at both the individual module level and the application level. The application-level checks include checks between the application-level deployment descriptors and the individual modules, as well as validation checks across the modules.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Packaging Web Applications

You should package an application before it can be deployed to Oracle WebLogic Server. To appropriately package a Web application, perform the following steps:

1. Arrange the resources in a prescribed directory structure.
2. Develop or copy the `web.xml` deployment descriptor (optional).
3. Develop or copy the `weblogic.xml` deployment descriptor (optional and WLS specific).
4. Archive the Web application into a `.war` file using Java Archive (JAR).
5. Deploy the Web application onto Oracle WebLogic Server.
6. Configure the Web application with the Oracle WebLogic Server Administration Console.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

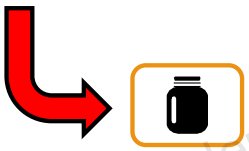
Carl McCrudder (carlmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Packaging Web Applications

Packaging and deploying a Web application is a relatively simple process. A more difficult aspect of the process is the configuration of the XML files. But after configuring these files once or twice, you or the developer should be able to create your own templates, which will streamline the deployment process for any later projects. This XML task sometimes falls to the WLS Administrator, but also is sometimes performed by the application programmer. The purpose and description of the two primary files are covered briefly in this lesson.

Web Application Structure

Directory or File	Description
MyWebApp	Public document root of Web application
WEB-INF	Private resources not served directly to clients
classes	Classes, such as servlets, filters, listeners
lib	Java libraries (JAR files)
web.xml	Optional Java EE deployment descriptor
weblogic.xml	Optional WebLogic deployment descriptor
index.html page1.jsp page2.jsp	Static and dynamic Web content



MyWebApp.war

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Web Application Structure

The structure of Web applications is defined by the servlet specification. You develop your Web application within a specified directory structure so that it can be archived and deployed on Oracle WebLogic Server, or another Java EE–compliant server. All servlets, classes, static files, and other resources that belong to a Web application are organized under a directory hierarchy. The root of this hierarchy defines the document root of your Web application. All files under this root directory can be served to the client, except for files under `WEB-INF`. All files under `WEB-INF` are private and are not served to a client, including XML deployment descriptors.

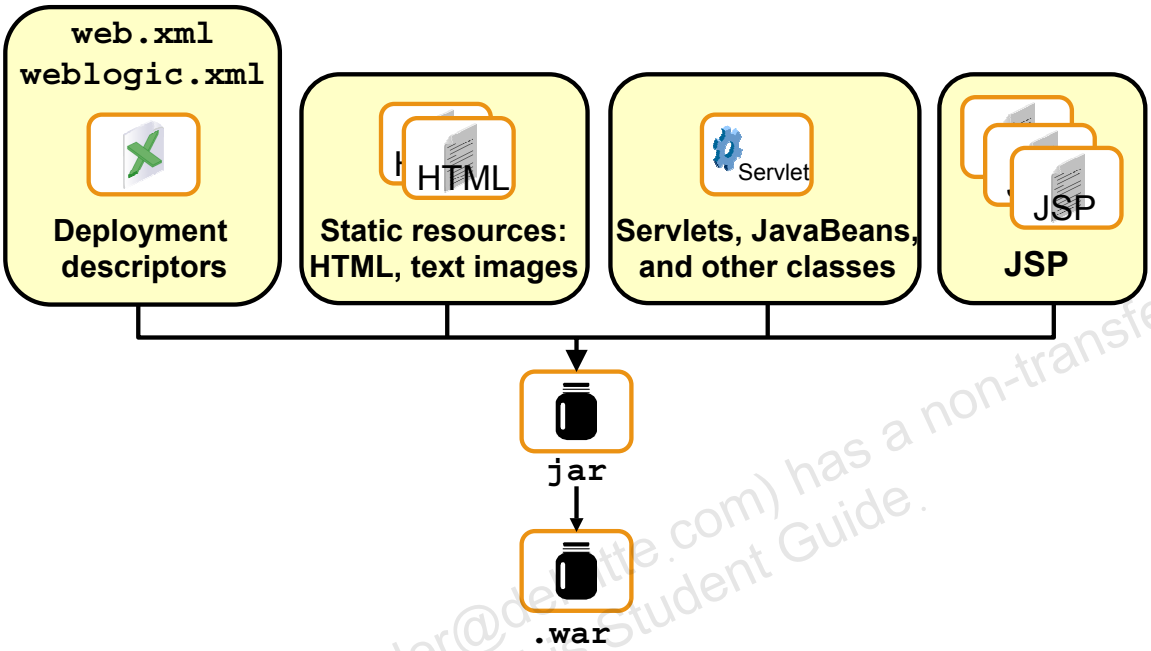
A WAR file contains the files that make up a Web application. It is deployed as a unit on one or more Oracle WebLogic Server instances. The WAR file can be deployed alone or packaged in an Enterprise Archive (EAR) file with other application components. If deployed alone, the archive must end with a `.war` extension. If deployed in an EAR file, the archive must end with an `.ear` extension. Alternatively, Oracle WebLogic Server enables you to deploy the Web application directory without archiving it. This technique is especially useful while the application is under development.

A Web application can be either:

- An archived file (`.war` file)
- An expanded (exploded, unzipped) directory structure

Web Application Archive

Web archives are created using the `jar` utility:



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Web Application Archive

The slide shows the application components that can be packaged into a WAR file. The JAR utility is used in many other places besides Web archives. Typically, JAR files are used as a normal means of packaging groups of classes together. The latest versions of the Java Development Kit (JDK) have most of the bootstrap classes stored in JAR files. Also, applets that are downloaded over the Internet are typically stored in JAR files. The JAR utility is modeled after the popular Tape Archive (TAR) utility on UNIX.

Optional Configuration of Web Applications

Web applications can be specified in `web.xml` and `weblogic.xml` deployment descriptors. The configurations include:

- Defining the run-time environment
- Mapping URLs to servlets and JSPs
- Defining application defaults such as `welcome` and `error` pages
- Specifying Java EE security constraints
- Defining work managers for applications
- Setting the context root for the application

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Optional Configuration of Web Applications

One of the pivotal components of a Web container and the Web applications that it contains is the ability to control deployment, which is a two-step process.

1. The first step in deployment is to define the normal everyday characteristics of a given Web application using the `web.xml` file. Characteristics such as which URLs should invoke which servlet or JSP page, and additional environmental information, such as where to store temporary files or who should receive error emails, can be defined within the deployment descriptor of a Web application. Other information, such as the first page to be displayed when an application is accessed or the name of an error page, is also part of the static definition of a Web application.
2. The second stage in deployment comes when an application is actually placed into service in a production environment. At such time, the security roles that were defined in the first stage must be mapped to the actual users and groups in the production system. Services such as database access, which are named in the first stage, must be assigned to actual resources in a production environment.

web.xml

The `web.xml` file is used to configure the following:

- Servlets and JSP registration
- Servlet initialization parameters
- JSP tag libraries
- MIME type mappings
- Welcome file list
- Error pages
- Security constraints and roles
- Resources
- EJB references

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

web.xml

The `web.xml` deployment descriptor file follows the Servlet 2.4 specification from Sun Microsystems. The `web.xml` file is packaged together with the Web application components in a `.war` file, which is then deployed in Oracle WebLogic Server. A sample `web.xml` follows:

```
<?xml version='1.0' encoding='UTF-8'?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <servlet>
  <servlet-name>Benefits</servlet-name>
  <servlet-class>com.dizzyworld.BenefitsServlet</servlet-class>
 </servlet>
 <servlet-mapping>
  <servlet-name>Benefits</servlet-name>
  <url-pattern>/servlet</url-pattern>
 </servlet-mapping>
 <welcome-file-list>
  <welcome-file>welcome.html</welcome-file>
 </welcome-file-list>
</web-app>
```

weblogic.xml

Using `weblogic.xml`, you can configure the following:

- The application's root context path
- Application logging
- Security role mappings
- Advanced session settings
- Session clustering
- References to shared libraries
- References to server resources (data sources, EJBs, and so on)
- Work managers and threading
- Virtual directories
- JSP compiler options

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

weblogic.xml

The `weblogic.xml` deployment descriptor follows a WebLogic-specific schema that is used only by Oracle WebLogic Server. It allows you to enable and configure Web application features that are not part of the Java EE specification. For example, you can:

- Change the default root URL path of the Web application.
- Send application log messages to a dedicated log file
- Change the default HTTP session timeout.
- Change the default cookie name that is used to track HTTP sessions.
- Enable clustering features, such as in-memory replication and persistence.
- Assign a WebLogic work manager to process requests to this application. Work managers are used to assign relative CPU time to high or low priority applications for tuning and fairness purposes.
- Tune the threading behavior that is used to process requests to this application.
- Map other file system locations to URLs for this Web application.
- Enable JSP precompilation.
- Enable directory index pages.
- Enable dynamic reloading of classes.
- Set the default MIME type.

weblogic.xml Deployment Descriptor

Example of the weblogic.xml deployment descriptor:

```
<?xml version='1.0' encoding='utf-8'?>
<weblogic-web-app
    xmlns="http://xmlns.oracle.com/weblogic/weblogic-web-app"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <session-descriptor>
       <timeout-secs>300</timeout-secs>
       <invalidation-interval-secs>60</invalidation-interval-secs>
       <persistent-store-type>jdbc</persistent-store-type>
       <persistent-store-pool>SessionDS</persistent-store-pool>
       <persistent-store-table>WL_SERVLET_SESSIONS</persistent-store-
    table>
   </session-descriptor>
</weblogic-web-app>
```

(You will see this file again in Lab 18.)

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

weblogic.xml Deployment Descriptor

To view the schema for weblogic.xml, go to:

http://xmlns.oracle.com/weblogic/weblogic-web-app/1.0/weblogic-web-app.xsd

Namespace descriptors pointing to bea.com will still work for the moment. Some of the descriptors available are: description, weblogic-version, security-role-assignment, run-as-role-assignment, resource-description, resource-env-description, ejb-reference-description, service-reference-description, session-descriptor (shown above), jsp-descriptor, auth-filter, and container-descriptor.

Sometimes, it is convenient to set a Web application as a default Web application for a server, so that in the request URL, you can omit the context name. You can select one default Web application per server. The only difference is that the default application has a stanza in weblogic.xml that says:

```
<weblogic-web-app>
    <context-root>/</context-root>
</weblogic-web-app>
```

To learn more about deployment descriptor elements, go to:

http://download.oracle.com/docs/cd/E12840_01/wls/docs103/webapp/index.html

URLs and Web Applications

The URL that is used to reference a resource in a Web application must include the name of the Web application.

Accessing a resource in a Web application:

`http://hostname:port/MyWebApplication/resource`

Where:

Hostname	Host name mapped to virtual host or *hostname:port*
MyWebApplication	Name of the Web application; not necessary if this is the default Web application
resource	Static page, servlet mapping, or JSP

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide

URLs and Web Applications

The protocol that is specified in a URL is typically HTTP for Web applications.

Server names can be any alphanumeric string that maps to a valid IP address. This string is usually registered and available from a domain name server (DNS) that does the name-to-IP mapping. UNIX and Windows workstations can also be configured to carry out this mapping, which may be useful for testing.

The port is typically not specified because it is assumed to be the default HTTP port, 80. It can, however, be any valid port for the server machine being used. Valid port numbers vary depending on the operating system because some port ranges are reserved for system use—for example, port numbers less than 1024 are reserved on UNIX-based machines.

Web Service Applications

A Web service application:

- Responds to HTTP client requests using the Simple Object Access Protocol (SOAP)
- Uses the same structure as a Java EE Web application
- Supports two additional deployment descriptors:
  - `webservices.xml`
  - `weblogic-webservices.xml`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Web Service Applications

Web Services can be shared by and used as modules of distributed Web-based applications. They commonly interface with existing back-end applications, such as customer relationship management systems, order-processing systems, and so on. Web services can reside on different computers and can be implemented by vastly different technologies, but they are packaged and transported using standard Web protocols, such as HTTP, thus making them easily accessible by any user on the Web. WebLogic Web services are typically packaged as Java EE Web applications.

The programming model for Java EE Web services allows you to create an annotated Java file, and then use Ant tasks to compile the file into a Java class and generate all the associated artifacts. The Java Web Service (JWS) annotated file is the core of your Web service. It contains the Java code that determines how your Web service behaves. The JWS annotations that you can use in a JWS file include the standard ones defined by the Web Services Metadata for the Java Platform specification as well as a set of other standard or WebLogic-specific annotations, depending on the type of Web service you create.

An application that configures one or more Web service endpoints is called a Web Service application. The standard Java EE deployment descriptor for Web services is called `webservices.xml`. This file specifies the set of Web services that are to be deployed to Oracle WebLogic Server and the dependencies they have on container resources and other services.

Virtual Directory Mappings

Virtual directories:

- Can be used to refer to physical directories
- Enable you to avoid the need to hard-code paths to physical directories
- Allow multiple Web applications to share common physical directories for specific requests such as images
- Decrease duplication of files across applications
- Are configured in `weblogic.xml`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Virtual Directory Mappings

Use the `virtual-directory-mapping` element to specify document roots other than the default document root of the Web application for certain kinds of requests, such as image requests. All the images for a set of Web applications can be stored in a single location and need not be copied to the document root of each Web application that uses them. For an incoming request, if a virtual directory is specified, the Servlet Container searches for the requested resource first in the virtual directory, and then in the Web application's original document root. This defines the precedence if the same document exists in both places.

Note: This has nothing to do with the Oracle Virtual Directories product, which is LDAP oriented. By contrast, these virtual directories are folder oriented.

Virtual Directory Mapping: Example

Snippet from `weblogic.xml`:

```
:
  <virtual-directory-mapping>
     <local-path>c:/usr/gifs</local-path>
     <url-pattern>/images/*</url-pattern>
     <url-pattern>*.jpg</url-pattern>
  </virtual-directory-mapping>
  <virtual-directory-mapping>
     <local-path>c:/usr/common_jsps.jar</local-path>
     <url-pattern>*.jsp</url-pattern>
  </virtual-directory-mapping>
:
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Virtual Directory Mapping: Example

The elements that you can define within the `virtual-directory-mapping` element are as follows:

- `<local-path>`: It is required and specifies a physical location on the disk.
- `<url-pattern>`: It is required and contains the URL pattern of the mapping.

The Oracle WebLogic Server implementation of `virtual-directory-mapping` requires that you have a directory that matches the URL pattern of the mapping. The example in the slide requires that you create a directory named `images` at `C:/usr/gifs/images`. This allows the servlet container to find images for multiple Web applications in the images directory.

Road Map

- Web applications
- EJB applications
  - Major EJB types and their purpose
  - EJB deployment descriptor files
- Enterprise applications

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide

EJB Applications

Enterprise JavaBeans (EJB):

- Standardize the development and deployment of server-side distributed components
- Are annotated Java classes
- Are packaged with XML deployment descriptors
- Support the following capabilities:
  - Remote access over a network
  - Object relational mapping via WLS or the Java Persistence API (JPA)
  - Transactions
  - Messaging integration
  - Dependency injection

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

EJB Applications

EJBs are server-side Java modules that implement a business task or entity and are written according to the EJB specification. There are three types of EJBs: session beans, entity beans, and message-driven beans.

One of the central goals of the EJB specification is to make it easier to program Java components, in particular by reducing the number of required programming artifacts and introducing a set of EJB-specific metadata annotations that make programming the bean file easier and more intuitive. Another goal of the EJB specification is to standardize the persistence framework and reduce the complexity of the entity bean programming model and object relational (O/R) mapping model.

Java EE cleanly separates the development and deployment roles to ensure that modules are portable between the EJB servers that support the EJB specification. Deploying an EJB in Oracle WebLogic Server requires running the Oracle WebLogic Server `appc` compiler to generate classes that enforce the EJB security, transaction, and life-cycle policies.

EJBs are a component architecture specification that defines the structure of the beans, the structure of the containers in which they operate, and the methods for interaction with their clients. Infrastructure services, application server implementation, and client implementation are defined by the developer and application server vendor.

Types of EJBs

EJB Type	Description	Example
Stateless Session	• Do not maintain state • Are synchronous • Are maintained in memory	• Check validity of stock symbol • Calculate billing of a phone call
Stateful Session	• Offer conversational interaction • Maintain state for client • Are synchronous	• Book a flight and car rental for travel • Manage a shopping cart
Entity	• Represent persisted data • Are synchronous	• Represent a player's statistics • Represent a stock's history
Message-Driven	• Are asynchronous and stateless • Consume JMS messages	• Store logging messages

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (carlmcgruder@yahoo.com) has a non-transferable license to use this Student Guide.

Types of EJBs

Each EJB has a particular design approach and requirements for its construction. There are many benefits that result from having a variety of EJB types. The differences in EJB types allow an application server to optimize performance by making assumptions about the state and persistence level of the component. The EJBs that have no state management can have a higher degree of pooling than EJBs that have state management. There are three levels of state behavior that a component can assume: no state, not persisted, and persisted.

Stateless session EJBs are components that implement a single-use service. That service can be invoked many times, but because the component does not maintain any state, the resulting effect is that the invocation provides a single use. Session beans provide a "reusable single-use service."

Stateful session beans are very similar to their stateless session bean counterparts. In fact, stateful session beans and stateless session beans are implemented in exactly the same manner. So, what is different about them? Stateful session beans are designed to maintain state across multiple client invocations on behalf of the client. A stateful session bean does this by storing stateful properties in the attributes of the component itself. An EJB container is then responsible for ensuring that subsequent client invocations for the same stateful bean are routed back to the object that hosts the stateful attributes.

Types of EJB (continued)

When an EJB is persistent, it means that the data in the EJB persists or exists whether the EJB is currently in memory or not. The persistence of an EJB can be implemented in a variety of ways. For instance, the object could be stored in a relational database, stored in a file, or placed in another form of media. Entity beans (of EJB 2.x) are replaced by JPA in EJB 3.0, and WLS 10.3 supports both.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

EJB Application Structure

Directory or File	Description
MyEJBApp	Application root folder
javapackage	Java classes organized into packages
MyEJB1.class MyEJB2.class	EJB and other Java class files
META-INF	Meta-information folder
ejb-jar.xml persistence.xml	Optional Java EE and JPA deployment descriptors
weblogic-cmp-jar.xml weblogic-ejb-jar.xml	Optional WebLogic deployment descriptors

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

EJB Application Structure

Unlike earlier versions of the EJB specification, you are no longer required to create the EJB deployment descriptor files. You can now use the metadata annotations in the Java bean file to configure metadata. You are still allowed, however, to use XML deployment descriptors; in the case of conflicts, the deployment descriptor value overrides the annotation value.

The optional Java EE-specified deployment descriptor, `ejb-jar.xml`, describes the enterprise beans that are packaged in an EJB application. It defines the beans' types, names, and the names of their home and remote interfaces, and implementation classes. The `ejb-jar.xml` deployment descriptor defines the security roles for the beans, and the transactional behaviors for the beans' methods.

Additional deployment descriptors provide WebLogic-specific deployment information. A `weblogic-cmp-rdbms-jar.xml` deployment descriptor, which is unique to the container-managed entity beans, maps a bean to the tables in a database. The `weblogic-ejb-jar.xml` deployment descriptor supplies additional information that is specific to the Oracle WebLogic Server environment, such as JNDI bind names, clustering, and cache configuration.

EJB modules are packaged as archive files having a `.jar` extension, but can also be deployed as exploded archive directories.

weblogic-ejb-jar.xml

Using `weblogic-ejb-jar.xml`, you can configure:

- Security role mappings
- Advanced security settings
- EJB clustering
- EJB pooling and caching
- Work managers and threading

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

weblogic-ejb-jar.xml

The `weblogic-ejb-jar.xml` deployment descriptor follows a proprietary schema that is used only by Oracle WebLogic Server. It allows you to enable and configure EJB features that are not part of the Java EE specification. Examples:

- Mapping the security role names that are used in EJB annotations to identities in the WebLogic security realm
- Running an EJB within the context of a specific WebLogic security identity
- Enabling load balancing and failover for remote EJB invocations
- Tuning EJB performance using pool and cache settings
- Assigning a WebLogic work manager to process requests to an EJB. Work managers can be created for high, medium, and low priority CPU cycles, and then an EJB can be assigned to a work manager (or not).
- Tuning the threading behavior that is used to process requests to an EJB. You can limit the maximum number of threads that a process can spawn for purposes of throttling resources and fairness.

Administrator Tasks with EJBs

The administrator tasks for EJBs include:

- Configuring and deploying
- Resolving JNDI and other infrastructure issues
- Monitoring EJB caches and pools

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Administrator Tasks with EJBs

An administrator should be aware of setting up the:

- Pool size of stateless session EJBs
- Cache size of the stateful session EJBs and other related information
- Pool and cache sizes of the entity EJBs and other related information

Other administrative tasks include clustering settings, which is covered in the lesson titled "Introduction to Clustering."

Road Map

- Web applications
- EJB applications
- Enterprise applications
  - Enterprise application concepts
  - Enterprise Archive (`.ear`) file structure
  - Enterprise application configuration

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide

What Is an Enterprise Application?

- An enterprise application is a grouping of several resources into one deployable unit that is packaged in an `.ear` file.
- These resources include:
  - Web applications (`.war`)
  - EJB applications (`.jar`)
  - Java applications (`.jar`)
  - Resource adapters (`.rar`)

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

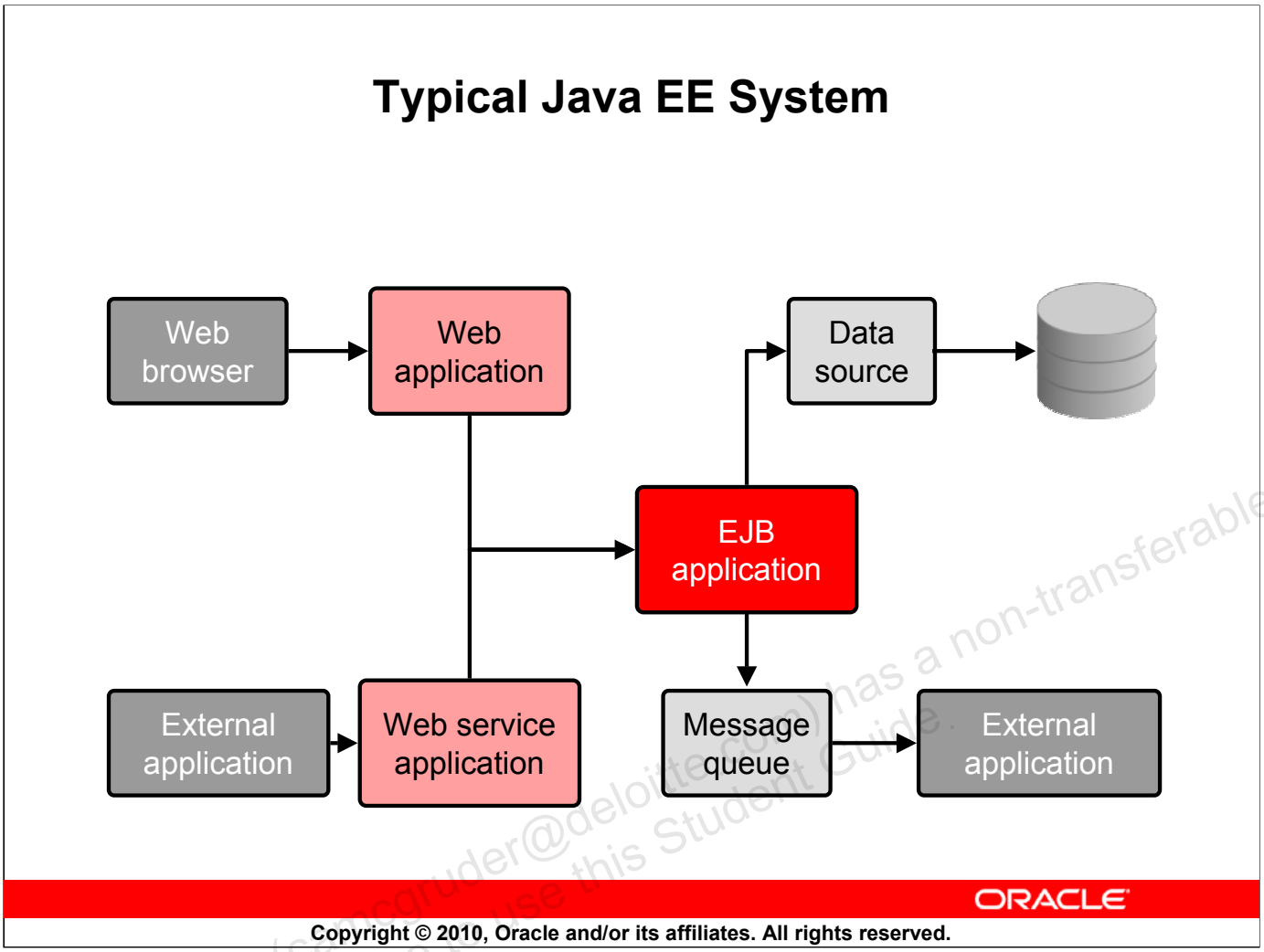
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Is an Enterprise Application?

An enterprise application is an archive file that packages a Web application along with any resources it might require. Typically, an enterprise application includes a `.war` file, one or more support `.jar` files, and any Enterprise JavaBeans that the application may require.

Note: Do not confuse this `.rar` extension with a zip-like compression format also called RAR (Roshal ARchive). This WebLogic RAR is a wrapper for non-Java legacy resources.



Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

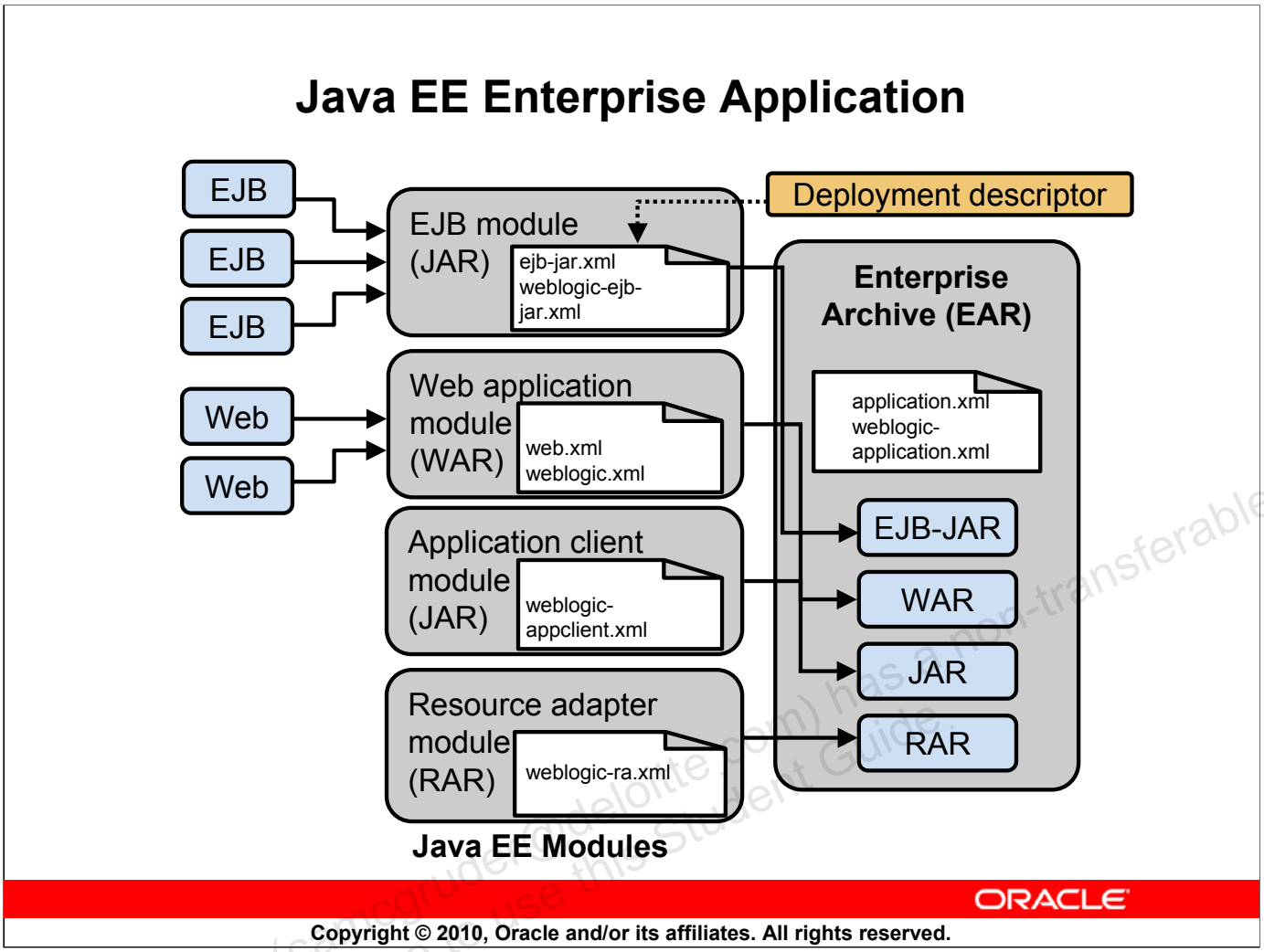
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Typical Java EE System

This diagram depicts a typical enterprise system built using Java EE and Oracle WebLogic Server:

- An EJB application contains distributed components to model the core business logic. Some EJBs use object relational mappings to retrieve and update business data.
- A JDBC data source manages the connections to the back-end database.
- EJBs use Java Message Service (JMS) to publish messages to other external Java EE and non-Java EE applications. EJBs may also subscribe to and process messages published from elsewhere in the enterprise.
- Users interact with the system through an HTML-based Web application, delivered via a Web browser. This Web application interacts with EJBs to perform business logic.
- To integrate other Java EE and non-Java EE systems within the enterprise, the business logic modeled in EJBs is also exposed as a collection of Web services.

Because of the distributed nature of Java EE and Oracle WebLogic Server, the components of this enterprise application can be deployed to a single server, to multiple servers, or even across multiple WebLogic domains. For example, the EJB application, Web application, and Web service application can all be distributed onto separate, dedicated server instances.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java EE Enterprise Application

The slide shows the components that can be found in an Enterprise Archive (EAR). A Java EE enterprise application is a group of Java EE application modules packaged into one deployable unit: an EAR file. As a general rule, if there is a `*.xml` deployment descriptor file, there is a corresponding `weblogic-*.xml` deployment descriptor file.

An EAR file has the `.ear` file extension. It contains one or more Java EE modules (EJBs, Web applications, application client modules, and resource adapters), and any resources that they require.

Typically, a Java EE EAR contains a Web application archive (`.war`) file and the resources on which the applications depend. The resources can be Java EE modules, such as EJB JARs, resource adapters (RARs), and others—for example, class libraries that are packaged as JARs. A library at this level would be available cross-module, whereas a stand-alone library would be available cross-application.

Java EE Enterprise Application

An enterprise application:

- Comprises one or more Java EE application modules:
  - Web applications
  - EJB applications
  - Other Java libraries (JARs)
- Allows related applications to be deployed as a unit
- Can include application-specific JDBC and JMS resources

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java EE Enterprise Application (continued)

A Java EE enterprise application consists of one or more Web application modules, EJB modules, connector modules, and other libraries, which can be managed and deployed as a single unit.

JMS and JDBC configurations are stored as modules as well, which are similar to standard Java EE modules. An administrator can create and manage JMS and JDBC modules as global system resources, as modules packaged with a Java EE application (as a packaged resource), or as stand-alone modules that can be made globally available.

For both production and development, you should package and deploy even stand-alone Web applications, EJBs, and resource adapters as part of an enterprise application. Doing so enables you to take advantage of the split development directory structure of Oracle WLS, which greatly facilitates application development.

Why Enterprise Applications?

Use enterprise applications to:

- Avoid namespace clashes
- Declare applicationwide security roles
- Deploy an application as one unit
- Share applicationwide EJB resources
- Configure local JDBC data sources
- Configure local JMS resources
- Configure local XML resources

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Why Enterprise Applications?

Enterprise applications are more than a packaging mechanism. You must consider the security, resources, and deployment issues when you deploy applications in a running system. You can package an application into a single coherent unit with enterprise applications. The application, its security constraints, and its resources are packaged and deployed together to simplify system management.

However, there are occasional reasons why you would *not* package an application into an `.ear` file. The most common reason is EJB use. If you have an EJB that is used by a large number of enterprise applications, it may be easier to separately deploy the EJB and the Web applications that use it.

Enterprise Application Structure

MyApp.ear

Directory or File	Description
MyApp	Application root folder
▼ APP-INF	WLS-specific
▶ lib	Contains `*.jar` files
▼ lib	Replaces `/APP-INF/lib`. May contain:
mylib1.jar	- Common Java class files
mylib2.jar	- Common Java libraries (JARs)
▼ META-INF	
application.xml	JEE deployment descriptor
mydatasource-jdbc.xml myqueue-jms.xml	JDBC and JMS modules
weblogic-application.xml	Optional WebLogic descriptor
EJBApp.jar WebApp1.war WebApp2.war	Web and EJB application modules

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide

Enterprise Application Structure

Enterprise applications can be deployed as a directory structure or packaged as an archive file with the `.ear` extension.

The `META-INF/application.xml` deployment descriptor contains an element for each Web application, EJB, and connector module, as well as the additional elements to describe security roles and application resources such as databases. The WebLogic deployer picks the list of modules from this descriptor. If this file is missing, you get the following error message:

```
If this is an EAR file, please ensure the
META-INF/application.xml exists.
```

By default, the Java classes present in the EJB modules are accessible by any Web application module found in the same enterprise application. However, the Java classes within one Web application module are not available to other Web application modules. To share class files and libraries among all modules, place them within the WebLogic-specific `APP-INF` directory of the enterprise application. These resources will then be made available to all application modules within the enterprise application.

weblogic-application.xml

Using `weblogic-application.xml`, you can configure:

- References to shared libraries
- Work managers and threading
- Default EJB and Web application parameter values

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

weblogic-application.xml

The `weblogic-application.xml` file is the Oracle WebLogic Server–specific deployment descriptor extension for the standard `Java EE application.xml` deployment descriptor. It allows you to enable and configure features such as:

- Referencing a shared library module that is deployed outside of this application
- Assigning a WebLogic work manager to process requests to the application modules
- Tuning the threading behavior used to process requests to the application modules
- Changing the default HTTP session timeout for all Web application modules
- Changing the default cookie name that is used to track HTTP sessions for all Web application modules
- Enabling clustering features such as in-memory replication and persistence for all Web application modules
- Configuring an applicationwide EJB cache (as opposed to an enterprisewide specification)

Application Scoping

Configure enterprisewide WLS-specific features with the `weblogic-application.xml` deployment descriptor:

- XML parsers
- XML entity mappings
- JDBC data sources
- JMS connection factories and destinations
- Security realms

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Application Scoping

Application scoping refers to configuring resources for a particular enterprise application rather than for an entire Oracle WebLogic Server configuration. In the case of XML, these resources include parser, transformer, external entity, and external entity cache configuration. The main advantage of application scoping is that it isolates the resources for a given application to the application itself. Using application scoping, you can configure different parsers for different applications, store the document type definitions (DTDs) for an application within the EAR file or exploded enterprise directory, and so on.

Another advantage of using application scoping is that by associating the resources with the EAR file, you can run this EAR file on another instance of Oracle WebLogic Server without having to configure the resources for that server.

EAR Class Libraries

- Extending the Java 2 EE 1.4 specification, Oracle added `APP-INF/lib` and `APP-INF/classes` to the standard Java EE EAR file structure. For Java EE 5, it is preferable to use the `/lib` directory.
- When an application is initialized, the paths extracted are prefixed to the application's classpath.
- Classes are added to the root classloader of the application.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

EAR Class Libraries

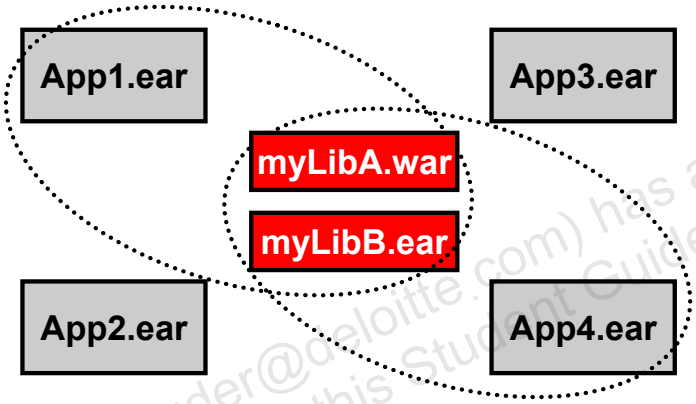
Oracle WebLogic Server provides a location within an EAR file where you can store the shared utility classes. Place the utility JAR files in the `APP-INF/lib` directory and the individual classes in the `APP-INF/classes` directory. These classes are loaded into the root classloader for the application.

This feature obviates the need to place the utility classes in the system classpath or place classes in an EJB JAR file (which depends on the standard Oracle WebLogic Server classloader hierarchy).

There is a new Java EE feature using a tag in `application.xml` that partially supersedes `APP-INF`. The `<library-directory>` tag is a Java EE standard that is equivalent to the `APP-INF/lib` directory and Oracle WebLogic Server will place JARs in this new directory on the classpath before those in `APP-INF/lib`. Using this feature would make the student's application more portable. The `/lib` directory is applicable only to EARs.

Java EE Library Support

- Create a library of Java EE modules, package the modules into an EAR, a WAR, or an EJB file, and then deploy and register the module with the application container.
- Other applications can later use the modules as if they were packaged in their own EAR, WAR, or EJB files.
- This allows for more reusability between the applications.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Java EE Library Support

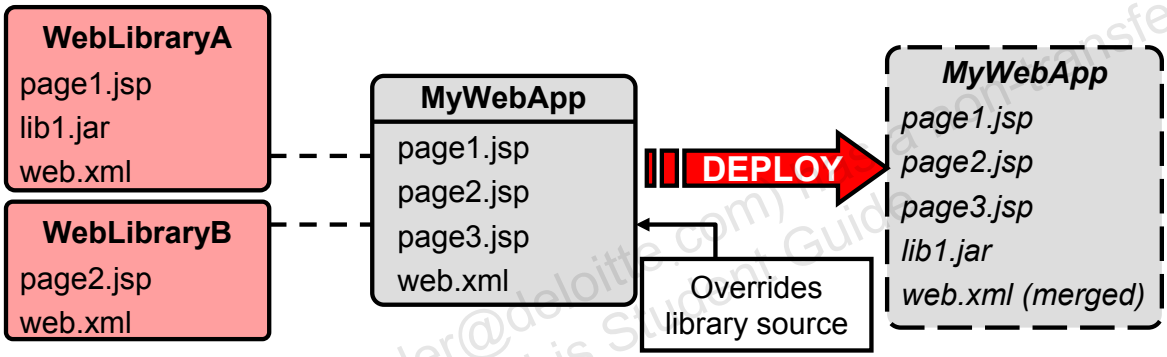
The Java EE library feature provides an easy way to share one or more types of Java EE modules among multiple enterprise applications. A Java EE library is a stand-alone EJB or Web application module, multiple EJB or Web application modules packaged in an Enterprise Archive (EAR), or a single plain JAR file that is registered with the Java EE application container upon deployment.

After the library has been registered, you can deploy enterprise applications that reference the library. Each referencing application receives a reference to the required library modules on deployment, and can use those modules as though they were packaged as part of the referencing application itself. The shared library classes are added to the classpath of the referencing application, and the referencing application's deployment descriptors are merged (in memory) with those of the Java EE library modules.

WebLogic Java EE Shared Libraries

A Java EE shared library:

- Is a reusable portion of a Web or enterprise application
- Is referenced by other deployed applications
- Avoids duplicating source files among Java EE projects
- Can contain deployment descriptors that are merged with the application's descriptors



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

WebLogic Java EE Shared Libraries

A Java EE shared library is a reusable portion of a Java EE enterprise application or Web application. At the enterprise application level, a shared library is an EAR file that can include Java classes, EJB deployments, and Web applications. At the Web application level, a shared library is a WAR file that can include servlets, JSPs, and tag libraries. Shared libraries can be included in an application by reference, and multiple applications can reference a single shared library.

You can deploy as many shared libraries to Oracle WebLogic Server as you require. In turn, libraries can reference other libraries, and so on. Because the shared library code and your own application code are assembled at run time, rules must exist to resolve potential conflicts. The following are the rules:

- Any file that is located in your application takes precedence over a file that is in a shared library.
- Conflicts arising between referenced libraries are resolved based on the order in which the libraries are specified in the `META-INF/weblogic-application.xml` file (for enterprise applications) or the `WEB-INF/weblogic.xml` file (for Web applications).

When a deployed enterprise application references one or more shared libraries, the server internally merges the information in the `weblogic-application.xml` file of the referencing enterprise application with the information in the deployment descriptors of the referenced libraries. Similar merging occurs for Web application deployment descriptors.

Shared Library References

- For Web applications, list the required shared libraries in `weblogic.xml`.
- For enterprise applications, list the required shared libraries in `weblogic-application.xml`.
- Excerpts from `weblogic.xml`:

Shared library name and version

```
:
<library-ref>
    <library-name>ajax-tools-lib</library-name>
    <specification-version>1.5.0</specification-version>
    <implementation-version>2.0.0</implementation-version>
</library-ref>

<library-ref>
    <library-name>help-web-lib</library-name>
    <specification-version>1.5.0</specification-version>
    <implementation-version>1.1.0</implementation-version>
</library-ref>
:
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Shared Library References

Oracle WebLogic Server supports versioning of the shared Java EE libraries, so that the referencing applications can specify a required minimum version of the library to use or an exact, required version. Oracle WebLogic Server supports two levels of versioning for the shared Java EE libraries:

- The specification version identifies the version number of the specification (for example, the Java EE specification version) to which a shared Java EE library or optional package conforms.
- The implementation version identifies the version number of the actual code implementation for the library or package. For example, this would correspond to the actual revision number or release number of your code. Note that you must also provide a specification version to specify an implementation version.

As a best practice, you should always include version information (an implementation version, or both an implementation and specification version) when creating shared Java EE libraries. Creating and updating version information as you develop shared components allows you to deploy multiple versions of those components simultaneously for testing.

Quiz

A _____ is a reusable Oracle WebLogic Server application that can be referenced by other deployed applications.

a. Java library
b. Shared library
c. Web library
d. Composite library
e. Reference library

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: b

Remember that applications can be deployed to WebLogic as shared libraries to facilitate code reuse.

Quiz

Which of the following is NOT a supported type of application in Oracle WebLogic Server?

a. Enterprise application
b. EJB application
c. Process application
d. Web service application
e. Web application

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: c

Currently, there is no Java EE or WebLogic application type named Process application.

Quiz

Identify the valid Oracle WebLogic Server deployment descriptor files for configuring applications.

a. `weblogic-webapp.xml`
b. `weblogic-ejb-jar.xml`
c. `weblogic.xml`
d. `weblogic-application.xml`
e. `weblogic-library.xml`

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Answer: b, c, d

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Summary

In this lesson, you should have learned how to:

- Package and deploy Web applications
- Describe deployment descriptors
- Explain Enterprise JavaBeans concepts
- Configure and deploy EJBs

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 12 Overview: Web Application Deployment Concepts

This practice covers the following topics:

- Deploying (installing) prerequisite libraries
- Deploying (installing) applications
- Starting and stopping applications
- Testing applications
- Redeploying (updating) applications
- Undeploying (deleting) applications
- Front-ending applications with a Web server—for example, Oracle HTTP Server

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 12 Overview: Web Application Deployment Concepts

See Appendix A for the complete steps to do the practice.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

13

Advanced Deployment

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Configure an application for multiple development environments
- Create a deployment plan
- Stage a deployment plan
- Use production redeployment

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Objectives

Scenario

You have a critical application that just got upgraded from version 1.2 to version 1.3 and now must be redeployed. However, you cannot afford to take it out of service even for a moment while it is being redeployed. What you would like to happen is for those clients in the middle of using 1.2 to complete what they are doing, and for the server not to start any new 1.2 clients. However, you do want new clients to have something to run; you want them to start with the new 1.3 version. What this means is that for an awkward period of time, there will be two versions running the old 1.2 version and the new 1.3 version, simultaneously. As soon as the last client finishes up with 1.2, undeploy that old version of the application.

Road Map

- Deployment plans
- Staged deployment
- Production redeployment

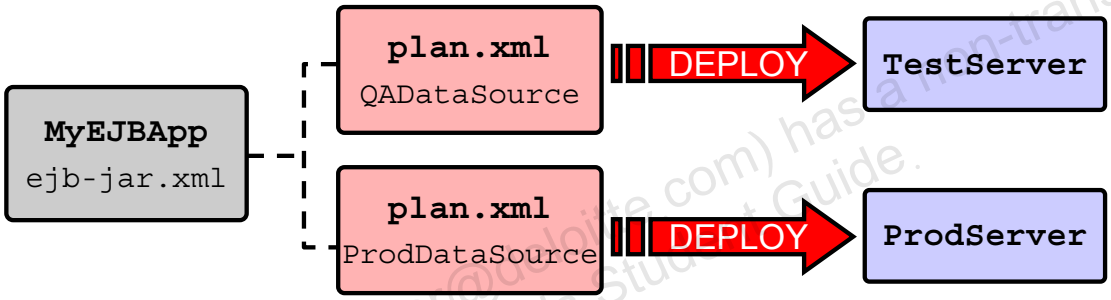
ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Is a Deployment Plan?

A Java EE deployment plan:

- Is an optional XML file associated with an application
- Resides outside an application archive
- Sets or overrides the values in the Java EE deployment descriptors
- Allows a single application to be easily customized to multiple deployment environments



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

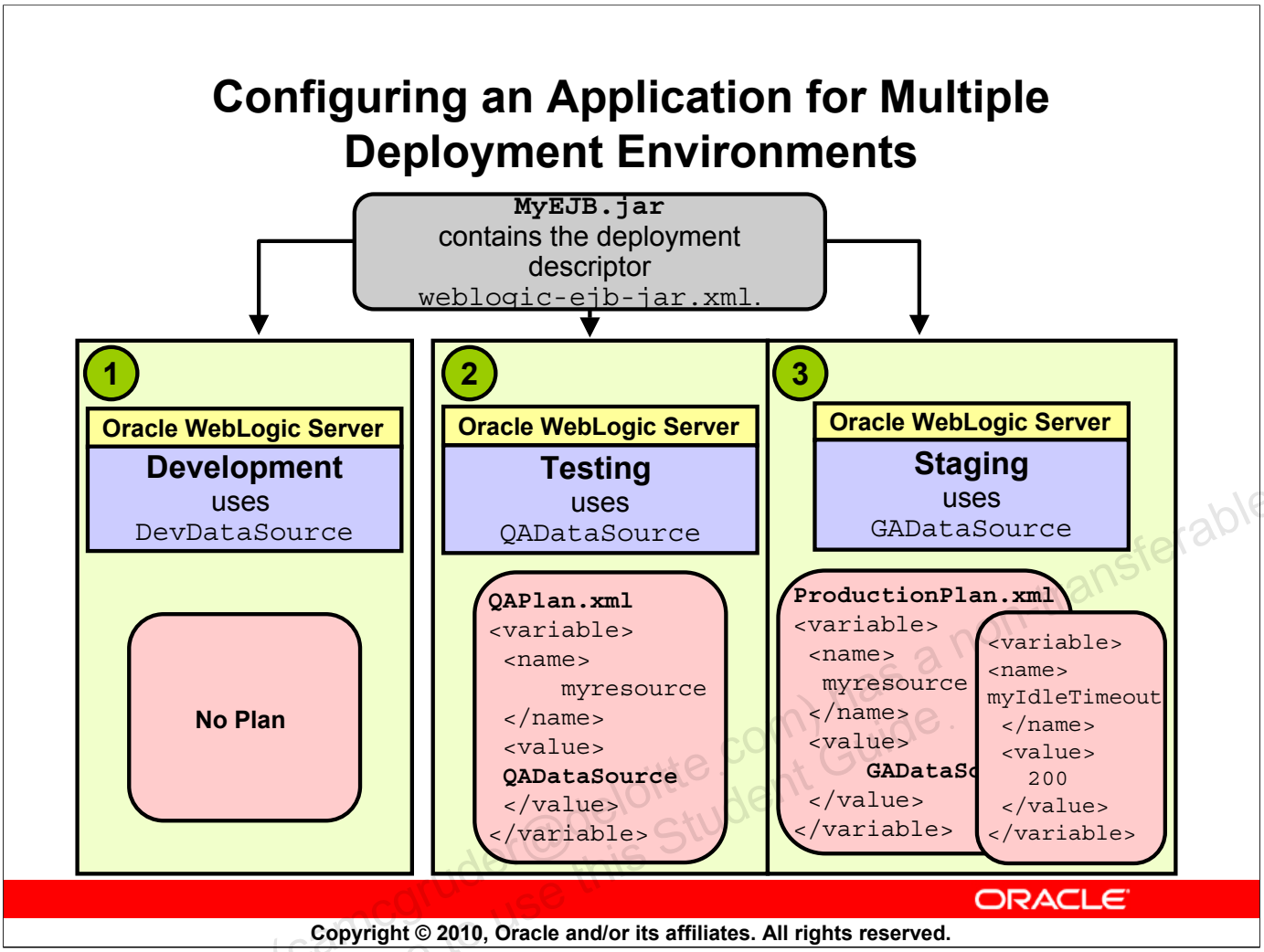
Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

What Is a Deployment Plan?

A deployment plan is an XML document that is used to define an application's deployment configuration for a specific Oracle WebLogic Server environment, such as development, test, or production. A deployment plan resides outside of an application's archive file and contains deployment properties that override an application's existing Java EE and Oracle WebLogic Server deployment descriptors. Use deployment plans to easily change an application's Oracle WebLogic Server configuration for a specific environment without modifying the existing deployment descriptors. Multiple deployment plans can be used to reconfigure a single application for deployment to multiple, differing Oracle WebLogic Server domains or servers.

Any external resources required by the application are subject to change when the application is deployed to a different environment. For example, the Java Naming and Directory Interface (JNDI) names of the data sources that are used in your development environment can be different from those used in testing or production. Exposing those JNDI names as variables makes it easy for deployers to use the available resources or create the required resources when deploying the application.

Certain tuning parameters that are acceptable in a development environment are unacceptable in a production environment. For example, it may suffice to accept default or minimal values for EJB caching on a development machine, whereas a production cluster would need higher levels of caching to maintain acceptable performance. To deploy the application to a new environment, an administrator simply creates or uses a new deployment plan as necessary.



Configuring an Application for Multiple Deployment Environments

1. **Development:** A developer develops and creates both Java EE and Oracle WebLogic Server deployment descriptors to configure the application for repeated deployments to the development environment. The development server uses a simple PointBase database for development, named "DevDataSource," and the `weblogic-ejb-jar.xml` descriptor identifies the resources for the application.
2. **Testing:** The developer packages the application into an archive file and delivers it to the administrator in the QA team. The testing environment uses a different data source named "QADataSource." At this point, the embedded deployment descriptors provide a configuration that is valid for the development environment used by the developer, but is not valid for the testing environment where the application must be deployed for testing. To deploy the application, the administrator of the testing environment generates a deployment plan `QAPlan.xml` to override the data source name configured in the application's embedded deployment descriptors.
3. **Staging/Production:** Similarly, when the application is released into production, the administrator of the staging or production environment creates or uses another deployment plan to configure the application. The production deployment plan `ProductionPlan.xml` again overrides the application deployment descriptors to identify a new JDBC data source "GADataSource" that is used by the production environment. For this environment, the deployment plan also defines tuning parameters to make better use of the additional resources that are available in the production domain.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@yahoo.com) has a non-transferable license to use this Student Guide.

Configuring an Application for Multiple Deployment Environments (continued)

Organizations that have numerous deployment environments that frequently change should use a configuration workflow with multiple deployment plans. In a multiple deployment plan workflow, each deployment plan is owned by the deployer of the application rather than the development team. You should store each deployment plan for a single application in its own plan subdirectory of the application's root directory. The multiple deployment plan configuration workflow works in the following way:

1. The development team releases a version of the packaged application deployment files (containing Java EE and Oracle WebLogic Server descriptors). The development team may or may not include a template deployment plan with exported variables for resource definitions or common tunable parameters.
2. Before deploying the application, each deployer generates a custom deployment plan to configure the application for the respective target environment.
   A custom deployment plan can be created by starting with a template deployment plan (or no deployment plan) and making changes to the application's deployment configuration using the Administration Console.
3. After defining the deployment configuration for the respective environment, each deployer retrieves the custom deployment plan and maintains it for future deployments of the application. It is recommended that you store the custom configuration plans in a source control system so that new versions can be tracked and reverted to if necessary.
4. For subsequent releases of the application, each deployer uses the respective customized deployment plan to configure the application for deployment. Using the customized plan allows deployers to perform deployments with `weblogic.Deployer` or automate deployments using WLST.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carl.mcgruder@esipts.com) has a non-transferable license to use this Student Guide.

Sample Deployment Plan

```
<?xml version="1.0" encoding="UTF-8"?>
<deployment-plan xmlns="http://xmlns.oracle.com/weblogic/deployment-plan" xmlns:xsi
    <!--weblogic-version:10.3.3-->
    <description>A deployment plan for project MyNewWSProjectEAR</description>
  <application-name>MyNewWSProjectEAR.ear</application-name>
    <variable-definition>
        <variable>
            <name>new_context_root</name>
            <value>new_root</value>
        </variable>
    </variable-definition>
  <module-override>
    <module-name>MyNewWSProjectEAR.ear</module-name>
    <module-type>ear</module-type>
  </module-override>
    <module-override>
        <module-name>MyNewWSProject.war</module-name>
        <module-type>war</module-type>
        <module-descriptor external="false">
            <root-element>weblogic-web-app</root-element>
            <uri>WEB-INF/weblogic.xml</uri>
            <variable-assignment>
                <name>new_context_root</name>
                <xpath>/weblogic-web-app/context-root</xpath>
                <operation>replace</operation>
            </variable-assignment>
        </module-descriptor>
    </module-override>
</deployment-plan>
```

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

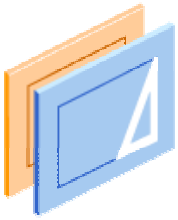
Sample Deployment Plan

The basic elements in a deployment plan serve the following functions:

- `deployment-plan`: Encapsulates the deployment plan's contents
- `application-name`: Corresponds to the deployment name for the application or module
- `variable-definition`: Defines one or more variable elements. Each variable element defines the name of a variable that is used in a plan and a value to assign, which can be null. In this example, the variable was `WeblogicWebApp_ContextRoots_11945442520421` with a value of `/beasys`.
- `module-override`: Are elements that define each module name, type, and deployment descriptor that the deployment plan overrides. A `module-descriptor` element can optionally contain a `variable-assignment` that identifies a variable name that is used to override a property in the descriptor and the exact location within the descriptor where the property is overridden.

Creating a Deployment Plan

- Tools for creating a deployment plan:
  - Development tool (for example, JDeveloper or Eclipse)
  - `weblogic.PlanGenerator`
  - Administration Console
- Goals for creating a deployment plan:
  - To expose the external resource requirements of the application as variables in the deployment plan
  - To expose additional configurable properties, such as tuning parameters as variables in the deployment plan



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (ccmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a Deployment Plan

- Exporting an application's deployment configuration is the process of creating a custom deployment plan that administrators can use for deploying the application into new Oracle WebLogic Server environments. You distribute both the application deployment files and the custom deployment plan to deployers—for example, testing, staging, or production administrators—who use the deployment plan as a blueprint for configuring the application for their environment.
- An administrator can install both the application and the custom deployment plan by using the Administration Console, which validates the deployment plan and indicates when specific configuration properties need to be filled in before deployment.
- `weblogic.PlanGenerator` creates a template deployment plan with null variables for selected categories of Oracle WebLogic Server deployment descriptors. This tool is recommended if you are beginning the export process and you want to create a template deployment plan with null variables for an entire class of deployment descriptors. You typically need to modify the deployment plan created by `weblogic.PlanGenerator` either manually or by using the Administration Console to delete extraneous variable definitions or add variables for individual properties.

Creating a Deployment Plan (continued)

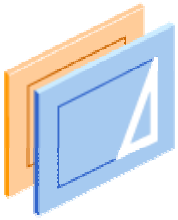
- The Administration Console updates or creates new deployment plans as necessary when you change the configuration properties for an installed application.
  You can use the Administration Console to generate a new deployment plan or to add or override variables in an existing plan. The Administration Console provides greater flexibility than `weblogic.PlanGenerator` because it allows you to interactively add or edit individual deployment descriptor properties in the plan rather than export entire categories of descriptor properties.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a New Deployment Plan

- The WebLogic Server includes tools to accelerate deployment plan creation.
- The Administration Console:
  - Generates a skeleton `plan.xml` if a plan folder is detected with a newly deployed application
  - Updates the `plan.xml` when you use the console to modify the deployment descriptor settings
- The `weblogic.PlanGenerator` Java class can also generate a skeleton `plan.xml` for an existing application.



ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Creating a New Deployment Plan

To create a deployment plan for a deployed application that does not already have a deployment plan, make a configuration change to the deployed application using the Administration Console. When you make a persisted configuration change to a deployed application that does not have an existing deployment plan, the console automatically creates a deployment plan for you and prompts you for the location in which to save it.

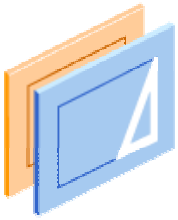
`weblogic.PlanGenerator` is a Java-based deployment configuration tool that is intended for developers who want to export portions of an Oracle WebLogic Server deployment configuration into a deployment plan. This utility can generate a brand new plan or can append to an existing one.

By default, `weblogic.PlanGenerator` writes an application's deployment plan to a file named `plan.xml` in the application's root directory. The syntax for invoking `weblogic.PlanGenerator` is the following:

```
java weblogic.PlanGenerator [options] [application]
```

weblogic.PlanGenerator

- Enables you to generate a basic Oracle WebLogic Server configuration for applications that have only Java EE deployment descriptors
- Enables you to:
  - Create an initial plan
  - Create a new plan based on an existing plan
  - Control components exported to a plan



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

weblogic.PlanGenerator

The following are examples of using `weblogic.PlanGenerator`:

- Creating an initial deployment plan in an application's root directory:
  `java weblogic.PlanGenerator -root /appRelease/MyApplication`
  In this example, the `plan.xml` file is automatically stored in `/appRelease/MyApplication/plan`.
- Creating a new deployment plan based on an existing plan:
  ```
  java weblogic.PlanGenerator -useplan
  /plans/MyApplication_template.xml
       -root /appRelease/MyApplication
  ```
- Controlling the components that are exported to a deployment plan
  The following command exports all configurable properties to null variables in a template deployment plan:
  `java weblogic.PlanGenerator -root /appRelease/MyApplication -all`

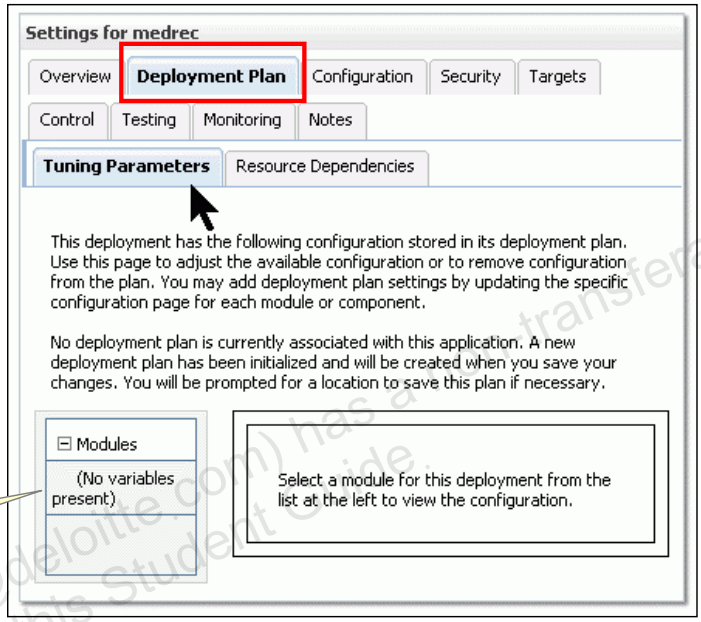
  You can use the `-all`, `-configurables`, `-dependencies`, `-declarations`, `-dynamics`, and `-none` options to specify the Oracle WebLogic Server deployment descriptor components that are exported to a template deployment plan.

For information about `weblogic.PlanGenerator`, see the Web site at:
http://download.oracle.com/docs/cd/E12840_01/wls/docs103/deployment/wlplangenerator.html

Using the Administration Console to Generate a Deployment Plan

You can generate a deployment plan with the Administration Console using the following steps:

1. Prepare the deployment files.
2. Install the application archive.
3. Save the configuration changes to a deployment plan.



Before: All empty

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Using the Administration Console to Generate a Deployment Plan

The Administration Console automatically generates or updates the deployment plan.

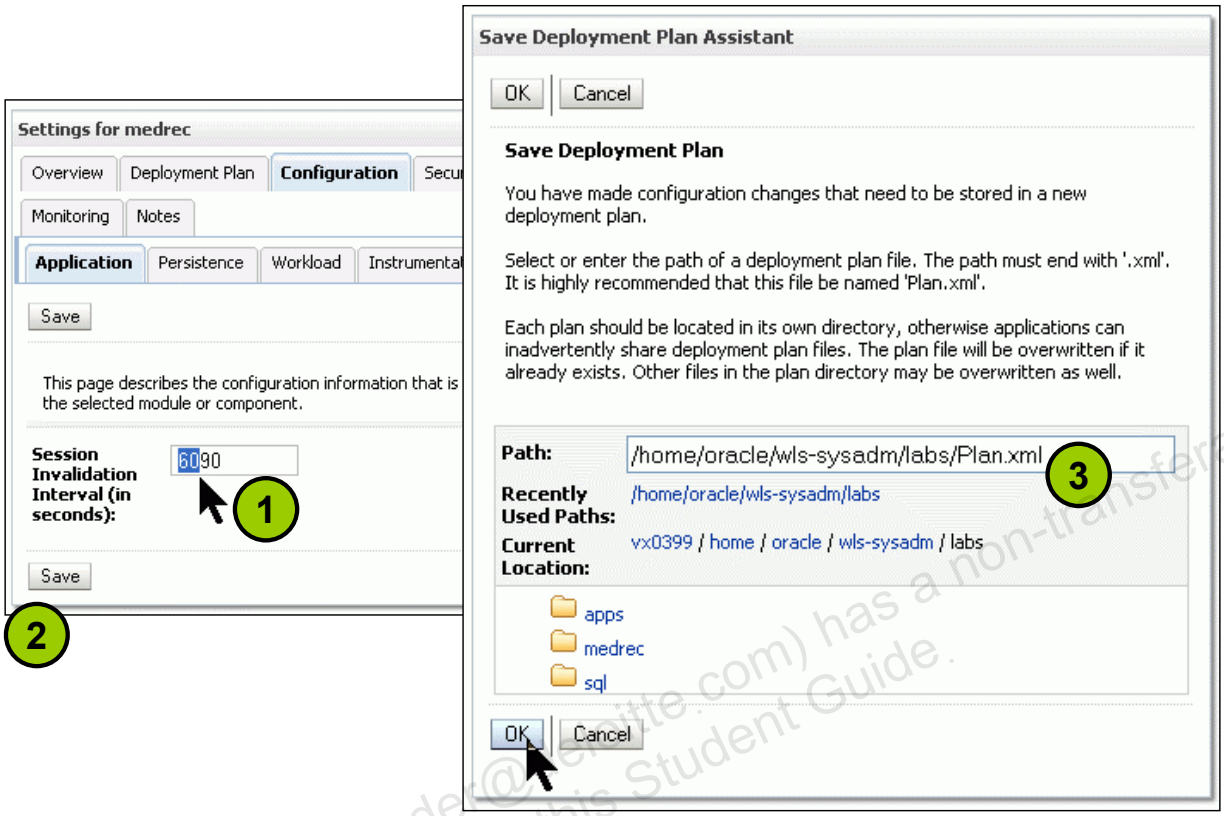
Note: You can use the generated deployment plan to configure the application in subsequent deployments, or you can generate new versions of the deployment plan by repeatedly editing and saving the deployment properties.

The Administration Console provides greater flexibility than `weblogic.PlanGenerator` because it allows you to interactively add or edit individual deployment descriptor properties in the plan, rather than export entire categories of descriptor properties.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCrudder (cmccrudder@deloitte.com) has a non-transferable license to use this Student Guide.

Modifying and Saving Data to Create a New Plan



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Modifying and Saving Data to Create a New Plan

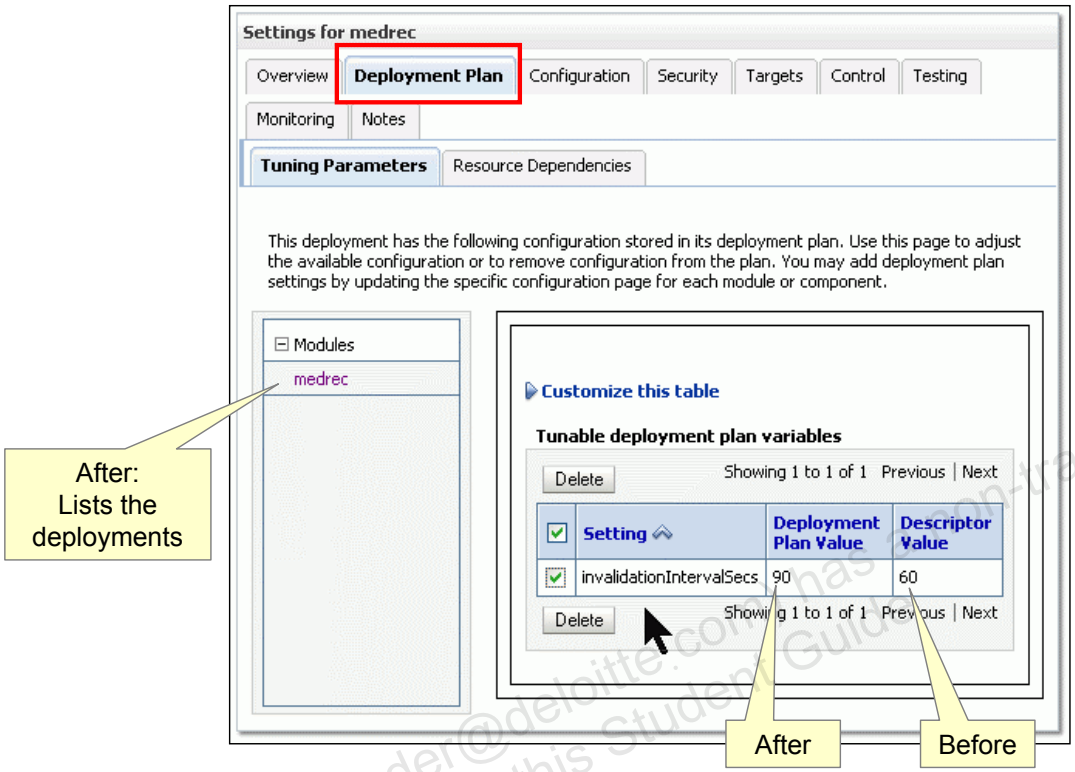
The application was originally deployed with various parameters (for example, a Session Invalidation Interval of 60 seconds).

1. A change was made from 60 to 90.
2. When you click Save, the system prompts you for a new or existing deployment plan into which to save this.
3. There was no deployment plan originally. So it is creating a new one called `Plan.xml`.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

New Deployment Plan Shows Changed Values



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

New Deployment Plan Shows Changed Values

Other modules and other variables can be added here as well.

This is a partial snippet of what was created. Notice the session invalidation in particular.

```
<?xml version='1.0' encoding='UTF-8'?>
<deployment-plan xmlns="http://xmlns.oracle.com/weblogic/deployment-plan"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.oracle.com/weblogic/deployment-plan
http://xmlns.oracle.com/weblogic/deployment-plan/1.0/deployment-plan.xsd">
  <application-name>labs</application-name>
  <variable-definition>
    <variable>
      <name>SessionDescriptor_invalidationIntervalSecs_12387843806400</name>
      <value>90</value>
    </variable>
  </variable-definition>
  <module-override>
    <module-name>medrec.ear</module-name>
    <module-type>ear</module-type>
    <module-descriptor external="false">
      <root-element>weblogic-application</root-element>
      <uri>META-INF/weblogic-application.xml</uri>
      <variable-assignment>

<name>SessionDescriptor_invalidationIntervalSecs_12387843806400</name>
        :
```

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using an Existing Deployment Plan to Configure an Application

1. Prepare the application.
2. Place the existing deployment plan in the plan subdirectory of the application root.
3. Install the application.
   - The Administration Console validates the deployment plan configuration against the target servers and clusters that are selected during the installation.
4. Use the Administration Console or the `weblogic.Deployer` utility to identify both the application and the plan to use for deployment.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using an Existing Deployment Plan to Configure an Application

The applications that you receive for deployment may come with varying levels of configuration information. If you have an existing deployment plan for an application, simply prepare the application and place the deployment plan in the `plan` subdirectory of the application root. Then install the application. The Administration Console automatically uses a deployment plan named `plan.xml` in the `\plan` subdirectory of an application root directory if one is available. If multiple plans are available for your application, they are placed in their own `\plan` subdirectories (for example, `\plan1` and `\plan2`), and the Administration Console cannot identify them. Therefore, `config.xml` must specify the plan that you want to use.

After you install a new application and an existing deployment plan, the Administration Console validates the deployment plan configuration against the target servers and clusters that were selected during installation. If the deployment plan contains empty (null) variables, or if any values configured in the deployment plan are not valid for the target server instances, you must override the deployment plan before you deploy the application. You can also configure tuning parameters to better suit the target environment in which you are deploying the application. The changes you make to the application's configuration are saved to a new deployment plan.

Using an Existing Deployment Plan to Configure an Application (continued)

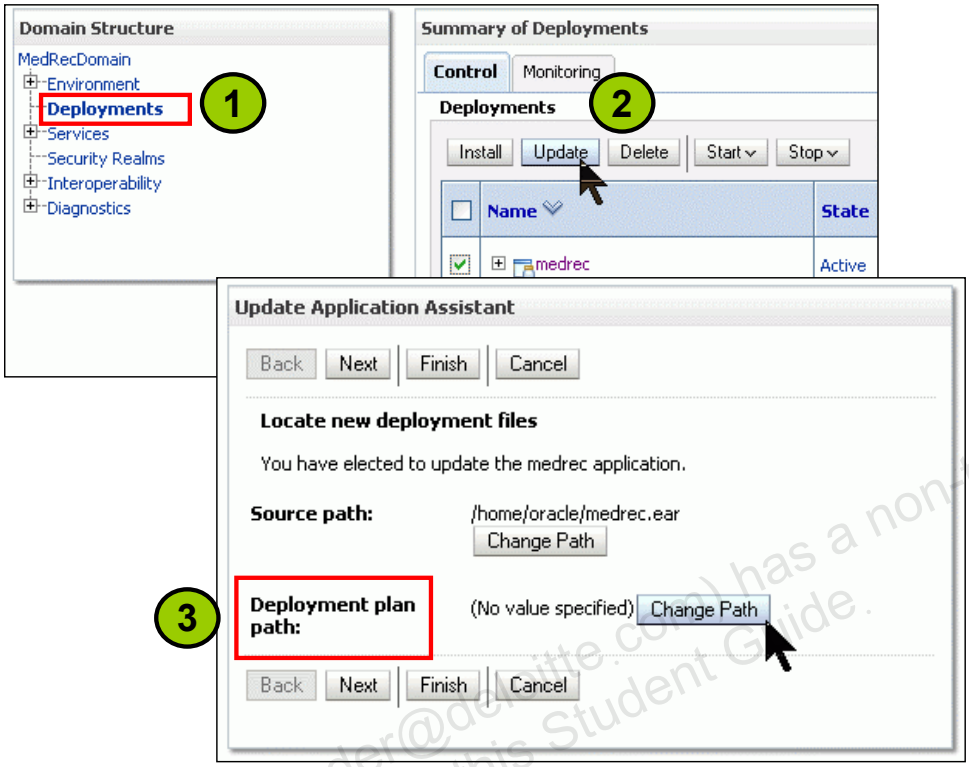
If you have a valid deployment plan that fully configures an application for the environment in which you are deploying, you can use either the Administration Console or the `weblogic.Deployer` utility to deploy an application with a deployment plan to use for deployment.

Note: A deployment plan that you use with the `weblogic.Deployer` utility must be complete and valid for your target servers. `weblogic.PlanGenerator` does not allow you to set or override individual deployment properties when it creates a plan.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Using an Existing Deployment Plan



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Using an Existing Deployment Plan

You can use the Administration Console to specify a deployment plan for your application.

1. In the left pane, click Deployments.
2. In the right pane, select the check box next to the application for which you want to specify a deployment plan. Click Update.
3. Click Change Path next to "Deployment plan path" to browse to the desired deployment plan. Click Next, and then click Finish.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (carlmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Directory Structure for Easier Production Deployment

Directory Tree	Description
my-app	Application root
app	Application deployment files (archive or exploded)
plan	plan.xml
AppFileOverrides	Optional application-specific overrides

This allows the deployment configuration files to be located in a well-known location.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

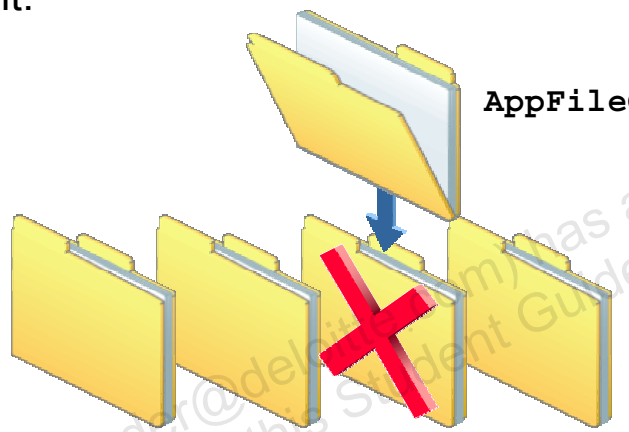
Directory Structure for Easier Production Deployment

The application directory structure separates the generated configuration files from the core application files, so that the configuration files can be easily changed or replaced without disturbing the application itself. The directory structure also helps you to organize and maintain multiple versions of the same application deployment files. The screenshot in the slide shows the directory hierarchy for storing a deployable application or module.

You should copy all new production deployments into an application installation directory before deploying to an Oracle WebLogic Server domain. Deploying from this directory structure helps you to easily identify all the files associated with a deployment unit; you simply deploy the installation root using the Administration Console, and the Console automatically locates the associated files such as deployment plans and WebLogic Server deployment descriptors that were generated during configuration. Applications can be deployed simply by specifying the installation root.

Generic File-Loading Overrides

- Place application-specific files to be overridden into a new optional subdirectory (named `AppFileOverrides`) in the existing plan directory structure.
- The presence or absence of this new optional subdirectory controls whether file overrides are enabled for the deployment.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Generic File-Loading Overrides

The files that are placed in the `/AppFileOverrides` subdirectory are staged and distributed along with the rest of the plan directory contents and are available on all the targets. To use this feature, you must:

- Specify a plan for the deployment
- Specify `config-root` within the plan
- Provide a `config-root/AppFileOverrides` subdirectory

Note: This mechanism is for overriding only resources and does not override classes.

These are application-specific files and the contents are opaque to WLS, so the entire file content is overridden when an override file is supplied.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCluder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Performing a Sanity Check in Production Without Disruption to the Clients

- Using Administration mode, administrators can deploy an application into a production environment without exposing the application to external clients.
- Access to the application is restricted to a configured administration channel.
- You can perform a final ("sanity") check on the application directly in the production environment without disruption to the clients.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Performing a Sanity Check in Production Without Disruption to the Clients

An administration channel would allow certain traffic to an application, but is not wide open to general traffic. This allows an administrator to test a "live" deployment without having potentially hundreds of clients in the channel at the same time (in case something needs fine-tuning).

Road Map

- Deployment plans
- Staged deployment
- Production redeployment

ORACLE

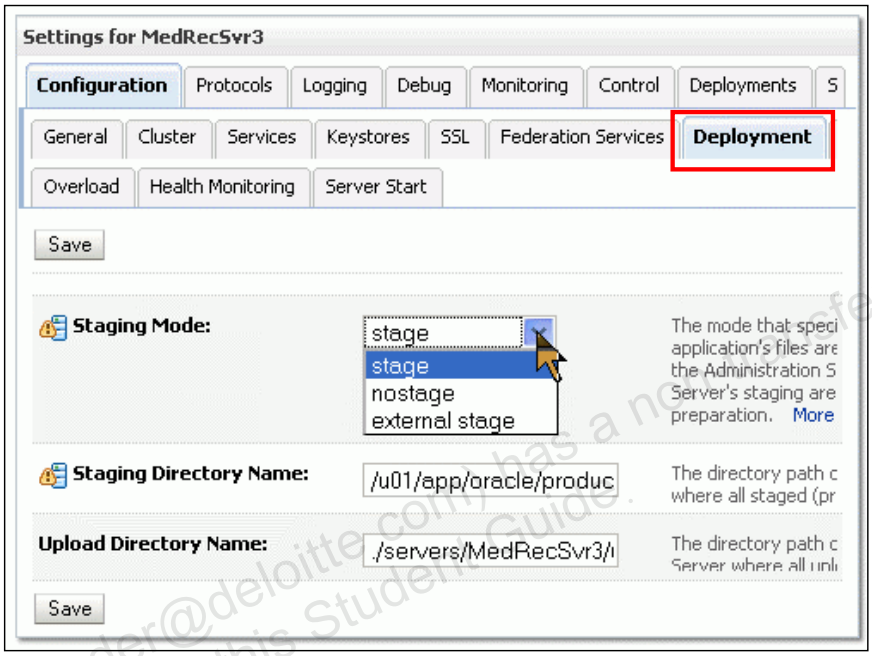
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (carlmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Staged Deployment

You can configure deployment per server or for each application.



Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Staged Deployment

With Oracle WebLogic Server, you have control over whether or not, where, and by whom the application files are copied before being deployed. All applications that are targeted to the managed servers can be copied by the administration server to the managed server before being prepared. This is called staging, and the files are staged to a configurable staging area for each server. There are three kinds of staging:

- **stage:** (default) Files are copied to the preconfigured staging directory for preparation and activation.
- **nostage:** Files are deployed from a static location.
- **external stage:** Files are copied by a user or a third-party tool (for example, JDeveloper or Eclipse) before deployment.

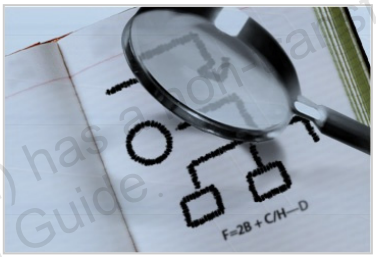
Applications can be deployed from the source location by changing the configuration (StagingMode). If the managed servers are running on a machine other than the administration server, it means that either the source location is on a file system that is accessible to the managed server (StagingMode = nostage) or the user or third-party tool performs the copy before the deployment request is issued (StagingMode = external stage).

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Road Map

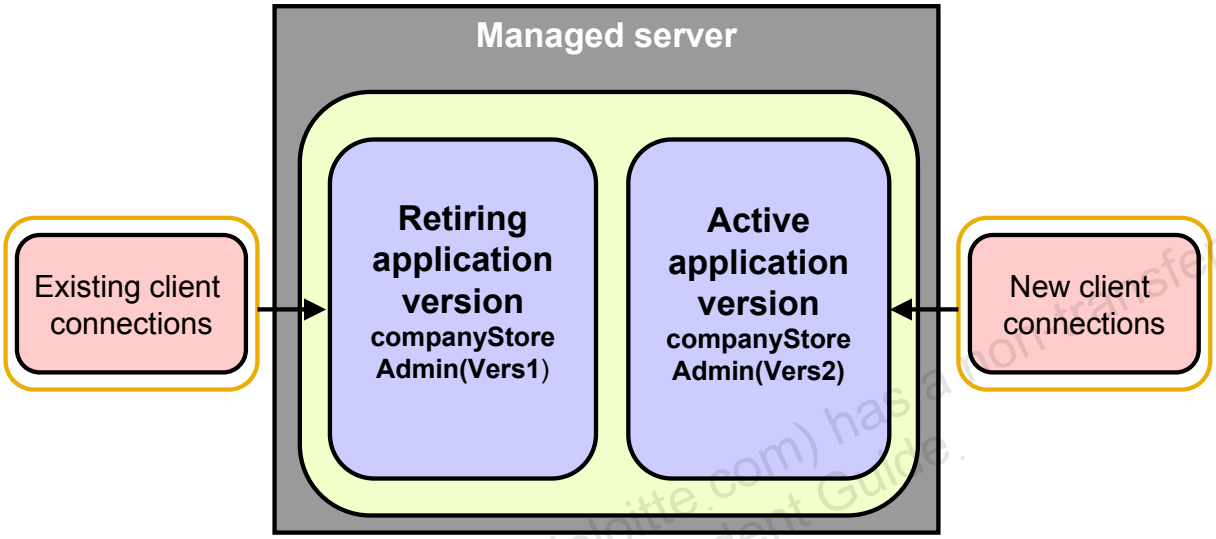
- Deployment plans
- Staged deployment
- Production redeployment



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McGruder (carmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Application Availability



ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Application Availability

By default, when an application is redeployed in place:

- It is unavailable to clients for a brief time
- Existing clients lose any conversational state

In a production environment, deployed applications frequently require 24×7 availability to provide uninterrupted services to customers and internal clients. In these cases, scheduling maintenance down time to replace applications is not desirable.

Oracle WebLogic Server uses an in-place redeployment scheme by default. In-place redeployment immediately replaces a running application's deployment files with the updated deployment files. In contrast to production redeployment, in-place redeployment of an application or stand-alone Java EE module does not guarantee uninterrupted service to the application's clients. This is because Oracle WebLogic Server immediately removes the running `classloader` for the application and replaces it with a new `classloader` that loads the updated application class files.

Production redeployment enables an administrator to redeploy a new version of an application in a production environment without stopping the deployed application or otherwise interrupting the application's availability to clients.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Production Redeployment and Application Versioning

- You can redeploy a revised version of a production application alongside the older version:
  - Without affecting the existing clients to the application
  - Without interrupting the availability of the application to the new client request
- Oracle WebLogic Server automatically manages client connections so that:
  - Existing clients continue to use the older application
  - New client requests are directed to the newer application
- The older version is retired and then undeployed after all current clients complete their work.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Production Redeployment and Application Versioning

Production redeployment works by deploying a new version of an updated application alongside an older version of the same application. Oracle WebLogic Server automatically manages client connections so that only new client requests are directed to the new version. Clients already connected to the application during the redeployment continue to use the older, retiring version of the application until they complete their work.

Production redeployment is supported primarily for applications with a Web application entry point (HTTP clients). Oracle WebLogic Server 10.3 can automatically manage the HTTP client entry points to isolate connections to the newer and older application versions. That is, production redeployment is supported for stand-alone Web application modules and for enterprise applications that are accessed via an embedded Web application module.

Production redeployment supports only HTTP clients and RMI clients. Your development and design team must ensure that applications using production redeployment are not accessed by an unsupported client. Oracle WebLogic Server does not detect when unsupported clients access the application and does not preserve unsupported client connections during production redeployment.

Production Redeployment and Application Versioning (continued)

Enterprise applications can contain any of the supported Java EE module types. Enterprise applications can also include application-scoped JMS and JDBC modules.

If an enterprise application includes a JCA resource adapter module, the module:

- Must be JCA 1.5 compliant
- Must implement the `weblogic.connector.extensions.Suspendable` interface
- Must be used in an application-scoped manner, having `enable-access-outside-app` set to `false` (the default value)

Before the resource adapters in the newer version of the Enterprise Archive (EAR) are deployed, the resource adapters in the older application version receive a callback. Oracle WebLogic Server then deploys the newer application version and retires the entire older version of the EAR.

When you are redeploying a new version of an application, the following features cannot change:

- Deployment targets
- Security model
- Persistent store settings

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

WebLogic Production Redeployment

- Production redeployment:
  - Enables two versions of a single Web application or module to run simultaneously
  - Requires you to include unique version information either:
    - Within the application's `META-INF/MANIFEST.MF` file
    - Or as part of the deployment process
- When a new version is redeployed, WLS automatically:
  - Routes existing clients to the earlier (retired) version
  - Routes new clients to the new version
  - Undeploys the earlier version when all existing clients finish their work or their conversations time out

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

WebLogic Production Redeployment

Production redeployment strategy involves deploying a new version of an updated application alongside an older version of the same application. Oracle WebLogic Server automatically manages client connections so that only new client requests are directed to the new version. Clients already connected to the application during the redeployment continue to use the older version of the application until they complete their work, at which point Oracle WebLogic Server automatically retires the older application. Production redeployment is currently not supported for stand-alone EJB applications.

When you redeploy a new version of an application, Oracle WebLogic Server treats the newly deployed application version as the active version and begins retiring the older version. During the retirement period, Oracle WebLogic Server automatically tracks the application's HTTP sessions and in-progress transactions. Oracle WebLogic Server tracks each HTTP session until the session completes or has timed out. In-progress transactions are tracked until the transaction completes, rolls back, or reaches the transaction timeout period.

To assign a version identifier to an application, it is recommended that you store a unique version string directly in the `MANIFEST.MF` file of the EAR or WAR that is being deployed. Alternatively, specify a version in the Administration Console by using the `appversion` argument of `weblogic.Deployer`.

Production Redeployment

- To support the production redeployment strategy, Oracle WebLogic Server now recognizes a unique version string entry in the Enterprise `MANIFEST` file.
- When a redeployment operation is requested, Oracle WebLogic Server checks the version string to determine whether to deploy a new version of the application.
- Production redeployment is performed automatically if:
  - An application supports production redeployment
  - Its deployment configuration is updated with changes to resource bindings
- This occurs even if no version string is specified in the application's manifest file.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Production Redeployment

For example, the following manifest file content describes an application with the version "v1":

```
Manifest-Version: 1.0
     Created-By: 1.5.0_04-b01 (Sun Microsystems Inc.)
     Weblogic-Application-Version: v1
```

Note: Production redeployment is also called side-by-side deployment.

It is recommended that you specify the version identifier in `MANIFEST.MF` of the application, and automatically increment the version each time a new application is released for deployment. This ensures that production redeployment is always performed when the administrator or deployer redeploys the application.

For testing purposes, a deployer can also assign a version identifier to an application during deployment and redeployment.

Oracle WebLogic Server obtains the application version from the value of the `Weblogic-Application-Version` property in the `MANIFEST.MF` file. The version string can be a maximum of 215 characters long and must consist of valid characters.

Advantages of Production Redeployment

Saves the trouble of:

- Scheduling application down time
- Setting up redundant servers to host new application versions
- Managing client access to multiple application versions manually
- Retiring older versions of an application manually

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Note

Production redeployment can be used with the `-distribute` command to prepare a new version of an application for deployment. For reference, see the manual titled *Distributing a New Version of a Production Application.*

Requirements and Restrictions for Production Redeployment

- Production redeployment strategy is supported for:
  - Stand-alone WAR modules and EARs whose clients access the application via HTTP
  - Enterprise applications that are accessed by inbound JMS messages from a global JMS destination or from inbound JCA requests
  - All types of Web services
- Production redeployment is *not* supported for:
  - Stand-alone EJB or RAR modules
  - Applications that use Java Transaction Service (JTS) drivers
  - Applications that obtain JDBC data sources via the DriverManager API
  - Applications that include EJB 1.1 container-managed persistence (CMP) EJBs

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmcgruder@gmail.com) has a non-transferable license to use this Student Guide.

Requirements and Restrictions for Production Redeployment

Note: Production redeployment supports only HTTP clients and RMI clients. Specific coding is required to handle the reconnection to the new version RMI client. Your development and design team must ensure that applications using production redeployment are not accessed by an unsupported client. Oracle WebLogic Server does not detect when unsupported clients access the application and does not preserve unsupported client connections during production redeployment.

There are ways to work around the limitations:

- **Stand-alone EJB or RAR modules:** If you attempt to use production redeployment with such modules, WebLogic Server rejects the redeployment request. To redeploy such modules, remove their version identifiers and explicitly redeploy the modules.
- **Applications that use JTS drivers:** For more information about JDBC application module limitations, see JDBC Application Module Limitations at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/jdbc_admin/packagedjdbc.html#wp1061858.
- **DriverManager API:** To use production redeployment, an application must use JNDI to look up data sources.
- **EJB 1.1 CMP EJBs:** To use production redeployment with applications that include CMP EJBs, use EJB 2.x CMP instead of EJB 1.1 CMP.

Redeploying a New Application Version

1. Verify that only one version of the application is currently deployed.
2. Verify the `MANIFEST.MF` files to ensure that both applications have different versions.
3. Copy the new version into a suitable directory.
4. Redeploy the new application version and specify the updated deployment files.
5. Verify that both versions are deployed and that new requests are being sent to the new version.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Redeploying a New Application Version

Following is an example of the `redeploy` command of `weblogic.Deployer`:

```
java weblogic.Deployer -adminurl http://localhost:7001
 -user weblogic
 -password Welcome1
 -redeploy
 -name retirement
 -source /myDeployments/myApplication/retirement
```

Place the new version of the application in a unique subdirectory similar to that of the original, before deployment.

The `weblogic.Deployer` tool enables you to specify a unique version string manually using the `-appversion` option when deploying or redeploying an application, which is helpful with an application that does not include a version string in the manifest file. An example of using the `-appversion` option is as follows:

```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic
    -password weblogic -deploy -name myTestDeployment
    -source /myDeployments/myApplication/10Beta
    -targets myCluster -stage -appversion .10Beta
```

Redeploying Versus Distributing

Distributing is an alternative to deploying an application.

Distributing a new version of the application makes it available for testing before being released for general consumption.	**Redeploying** a new version of an application places the application immediately into use and makes it available to new client requests.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

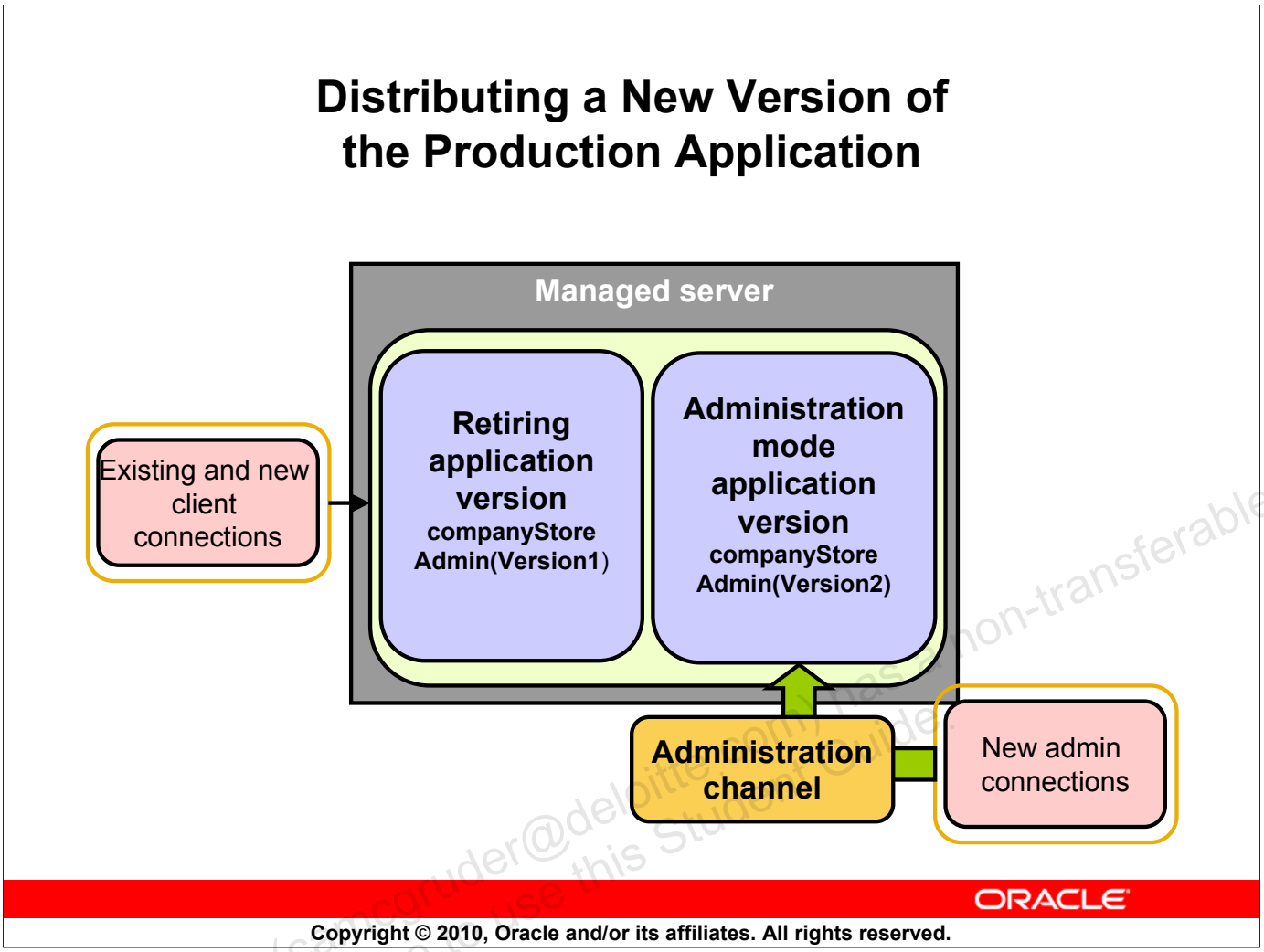
Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Redeploying Versus Distributing

Distributing an application prepares it for deployment by copying its files to all target servers and validating the files.

You can start a distributed application in Administration mode. Access to the application is then restricted to a configured administration channel.



Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Distributing a New Version of the Production Application

WLS prepares the new application version for deployment, which can be deployed in Administration mode and made available via a configured administration channel. However, the older version of the application is not automatically retired by WLS when the new version of the application is distributed and deployed in Administration mode. The older version of the application remains active to process both new and existing client requests.

The new application version can either be undeployed or started after it has been completely tested via an administration channel. WLS routes new client connections to the updated application upon starting the application and begins retiring the older application version.

Following are the steps for distributing a new version of an application:

1. To distribute a new version of an application in Administration mode, execute the `weblogic.Deployer -distribute` command:

```
java weblogic.Deployer -adminurl http://localhost:7001
     -user weblogic -password Welcome1
     -distribute -name companyStoreAdmin
     -source /myDeployments/myApplication/version2.0
     -appversion version2
```

Distributing a New Version of the Production Application (continued)

2. After the application is distributed, start it in Administration mode:
```
java weblogic.Deployer -adminurl http://localhost:7001 -user
weblogic
     -password Welcome1 -start -adminmode -name companyStoreAdmin
     -source /myDeployments/myApplication/version2
     -appversion version2
```
3. You can optionally specify the retirement policy or timeout period for distributed applications.
   Note: Starting the application in Administration mode makes it available only via a configured administration channel. See *Configuring Network Resources* at the following URL:
   http://download.oracle.com/docs/cd/E12840_01/wls/docs103/config_wls/network.html

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Distributing a New Application Version

1. Use the `weblogic.Deployer -distribute` command.
2. After the application is distributed, start the application in Administration mode.
3. Test the application.
4. When ready, start the application (without using `-adminmode`).
5. Optionally, set a retirement timeout for the older version of the application.

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Distributing a New Application Version

Following are `weblogic.Deployer` examples:

Distribute the application.

```
java weblogic.Deployer -adminurl http://localhost:7011 -user system
     -password weblogic -distribute -name retirement
     -source /myDeployments/myApplication/retirement
     -appversion 2.0
```

Start the application in Administration mode.

```
java weblogic.Deployer -adminurl http://localhost:7011 -user system
    -password weblogic -start -adminmode -name retirement
     -source /myDeployments/myApplication/retirement
     -appversion 2.0
```

Start the application in regular mode and set the retirement timeout for the older version.

```
java weblogic.Deployer -adminurl http://localhost:7011 -user system
     -password weblogic -start -name retirement
     -appversion 2.0 -retiretimeout 300
```

- `-appversion` is used when a version is not specified in the `MANIFEST.mf` file.
- `-retiretimeout` specifies the number of seconds after which the older version of the application is retired. This is optional. If this is not set, the older version of the application retires when existing clients have finished working with the application. In the meantime, all new requests are sent to the new version.

Production Redeployment

- Create `MANIFEST.mf` with at least the following contents:

```
Manifest-Version: 1.0
Weblogic-Application-Version: Version1.0Beta
```

- The value for WLS versioning has any text you choose, up to 215 characters, to indicate the version.

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Production Redeployment

The Sun documentation is available at:
http://java.sun.com/docs/books/tutorial/deployment/jar/packageman.html

Assume that you create the manifest in some directory, and below it are the application files. Go down one level and then create an EAR file named `someApp.ear` to be stored in the higher level (in the same directory as the manifest) by entering the following command:

```
jar cvfm ../someApp.ear ../Manifest.mf *
```

This syntax creates the EAR file with a manifest from the current directory contents but includes the manifest only once (otherwise, you would get the manifest stored in the EAR twice). The manifest now contains the versioning entry. The version text following the colon can be of any alphanumeric form you want. WebLogic Server requires only that the text be different from version to version. It need not be sequential; it need not be increasing. For example, the versions can be `red`, `blue`, or `green` instead of the traditional `1.0`, `2.0`, `2.5`, or `3.0`.

Quiz

Which is NOT true about the deployment plans in Oracle WebLogic Server?

a. Overrides values in application descriptors
b. Can be created by Oracle WebLogic Server during deployment
c. Is packaged within an application archive
d. Is an XML file
e. Can be created with `weblogic.PlanGenerator`

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Answer: c

Remember that deployment plans are not packaged within an application, so that the same application can be reused in multiple deployment environments.

Quiz

When an application is in the _____ state, it is distributed to the servers, but is not yet available to clients.

a. Activated
b. Staged
c. Targeted
d. Prepared
e. Loaded

ORACLE
Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Answer: d

Remember that before the Activated phase, applications first enter the Prepared phase.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCruder (cmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Quiz

Name four techniques or tools that can be used to deploy new applications to Oracle WebLogic Server.

a. Administration Console
b. WLST
c. `weblogic.PlanGenerator`
d. `weblogic.Deployer`
e. JMS
f. Autodeployment

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Answer: a, b, d, f

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.
Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Summary

In this lesson, you should have learned how to:

- Create deployment plans
- Stage deployments
- Perform production redeployment
- Configure an application for multiple development environments

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (cmcgruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 13 Overview: Deploying Production Applications

This practice covers the following topics:

- Version-enabling an application
- Deploying a versioned application
- Monitoring the status of a versioned application
- Deleting the expired version of an application

ORACLE

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McCruder (ccmccruder@deloitte.com) has a non-transferable license to use this Student Guide.

Practice 13 Overview: Deploying Production Applications

See Appendix A for the complete steps to do the practice.

Unauthorized reproduction or distribution prohibited. Copyright© 2011, Oracle and/or its affiliates.

Carl McGruder (camcgruder@deloitte.com) has a non-transferable license to use this Student Guide.