



# Building Application using IDE

# Objectives

After completing this lesson, you should be able to do the following:

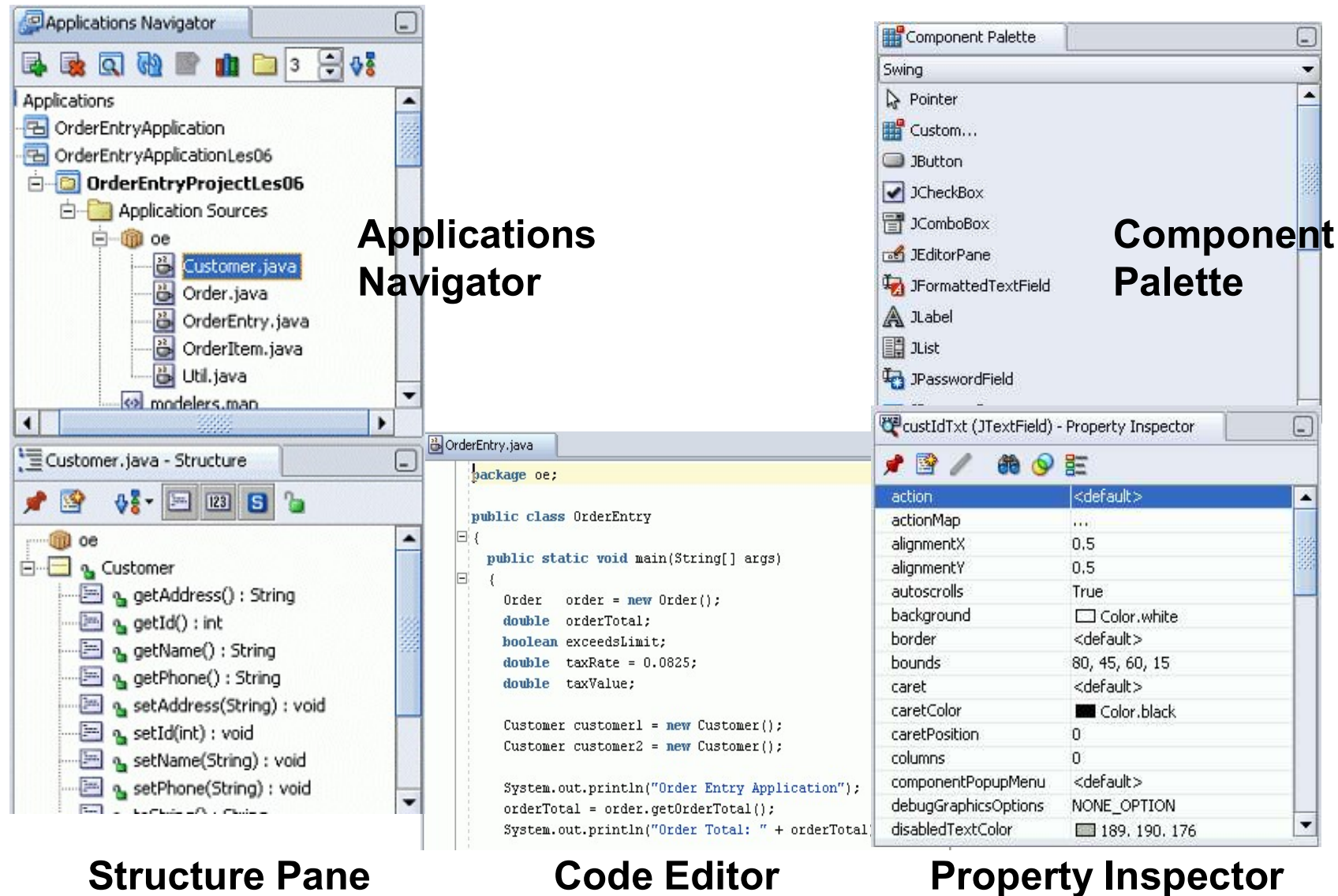
- Create applications and new projects
- Build Java applications in JDeveloper
- Enhance user interface frame design
- Debug applications with the JDeveloper debugger
- Define classes with JDeveloper
- Describe how JDeveloper can be used to build enterprise applications



# Oracle JDeveloper (11.1.3.2.0)

- Oracle JDeveloper (11.1.3.2.0) provides an integrated development environment (IDE).
- It enables you to:
  - Build, compile, and run Java applications
  - Use wizards to help build source code
  - View objects from many perspectives: code, structure, layout, and so on

# Oracle JDeveloper (11.1.3) Environment



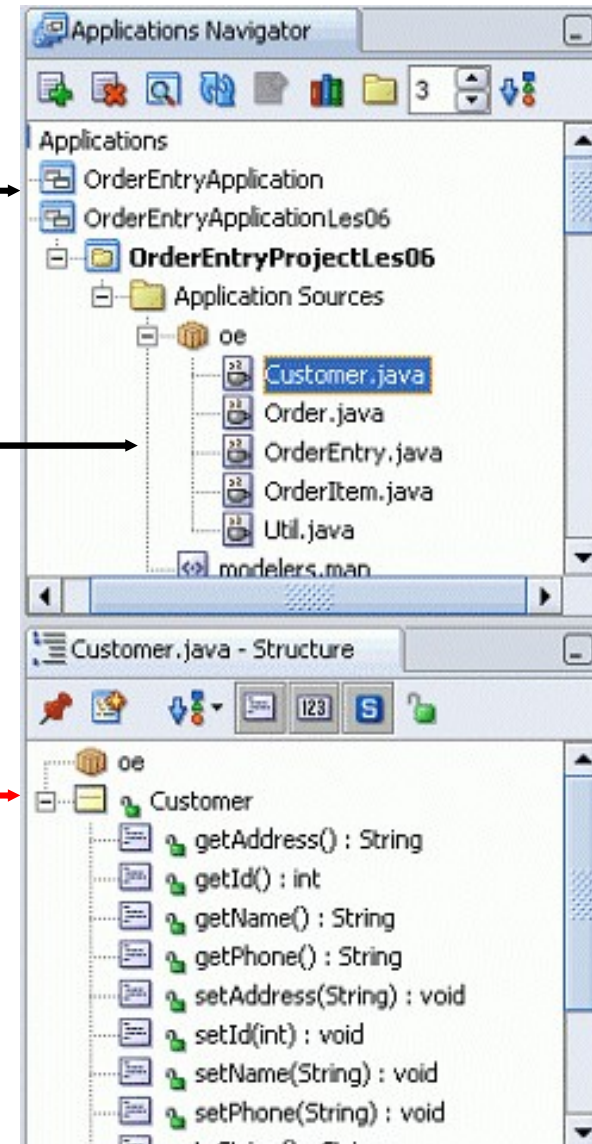
# Applications

- May contain multiple projects
- Enable you to view currently used objects

Application node

Applications  
Navigator  
pane

Structure  
pane

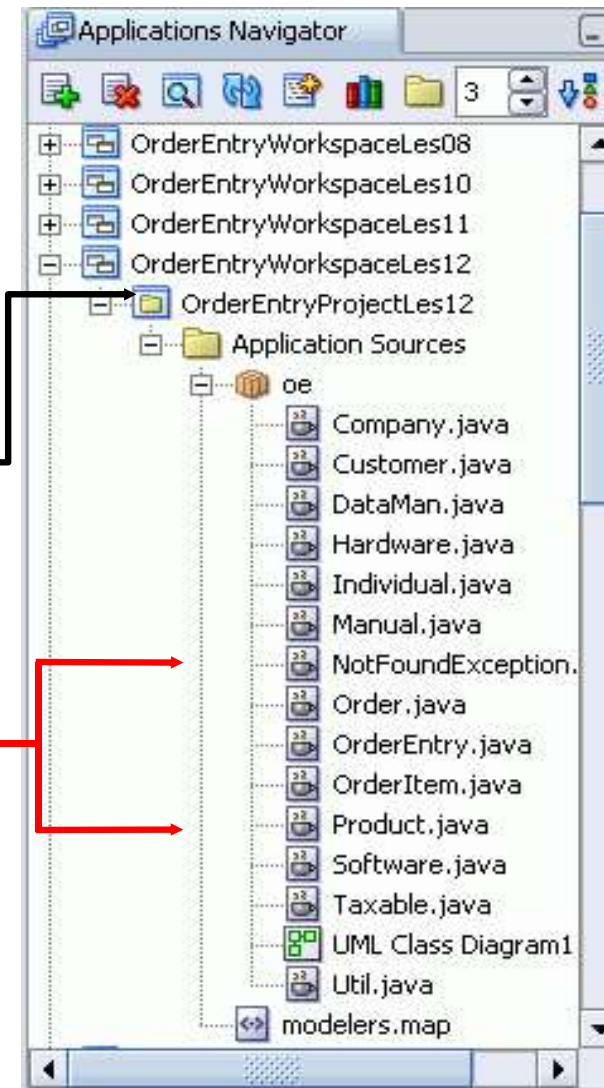


# Projects

- Contain related files
- Manage project and environment settings
- Manage compiler and debug options

Project  
node

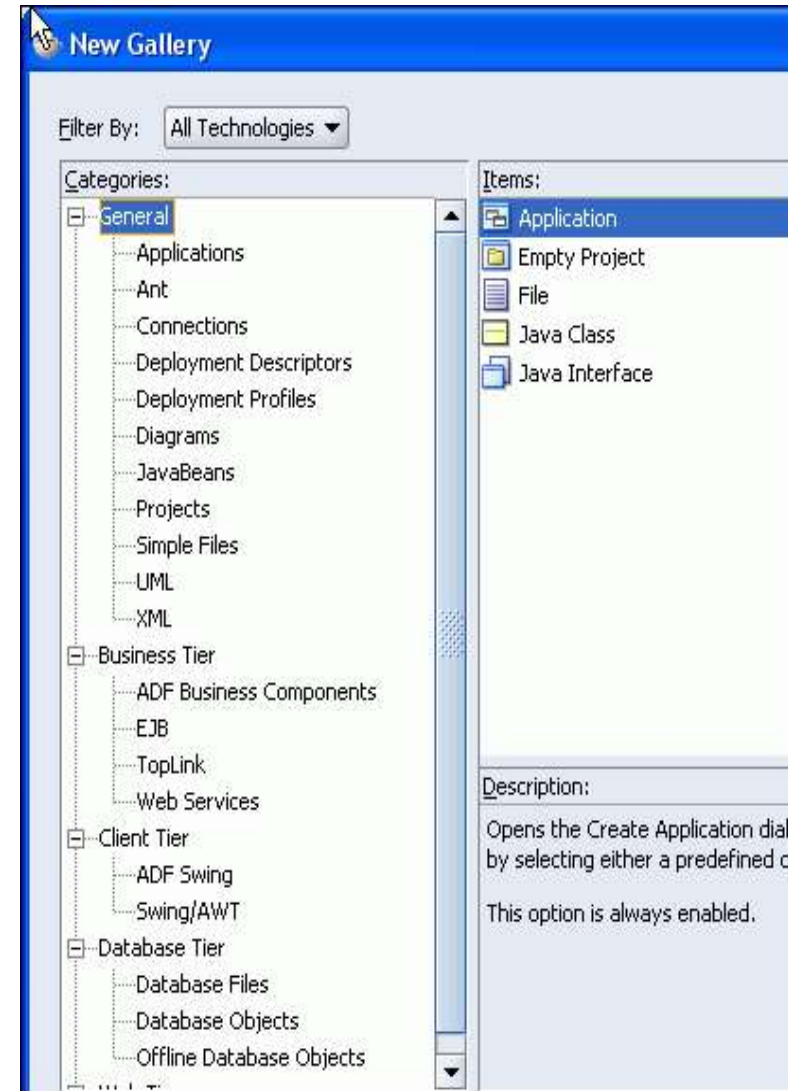
Project  
files





# Creating JDeveloper Items

- JDeveloper items are invoked by selecting File > New.
- They are categorized by type:
  - General
  - Business Tier
  - Client Tier
  - Database Tier
  - Integration Tier
  - Web Tier
- Create any JDeveloper element.



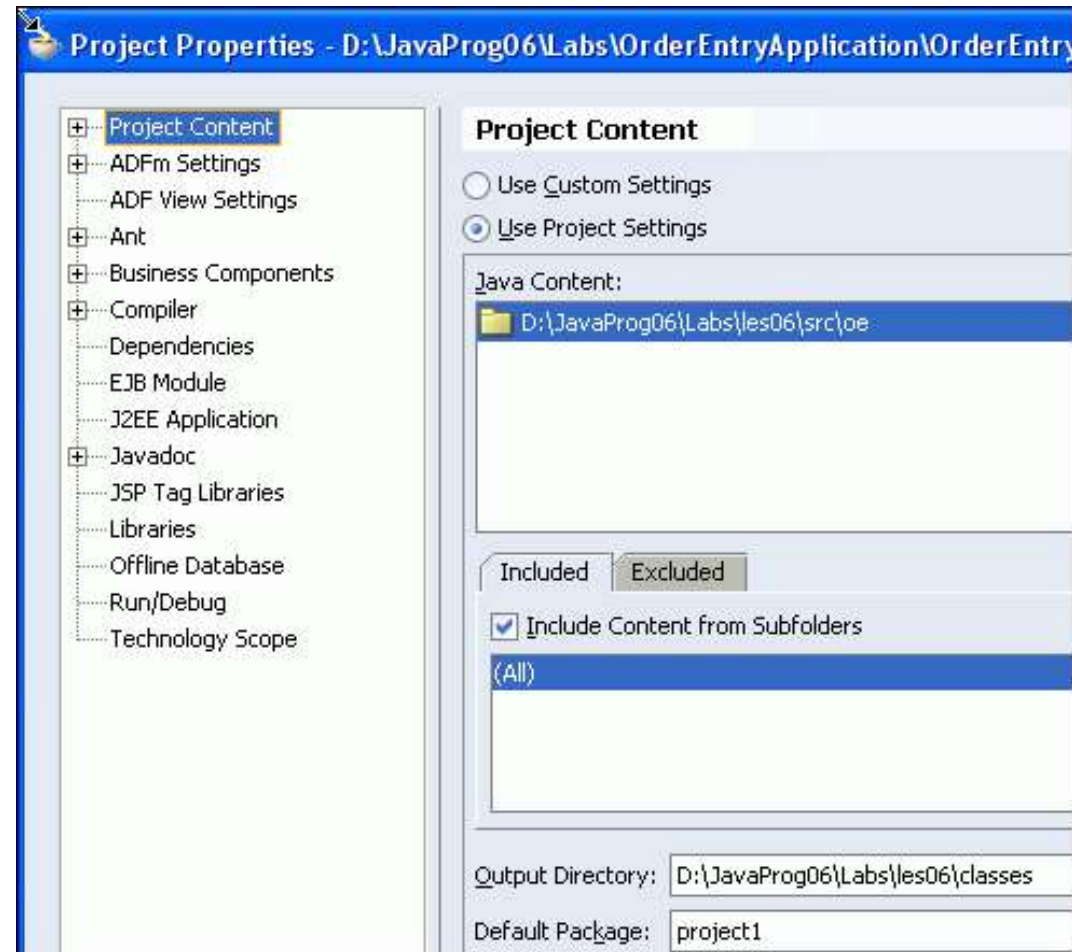
# Creating an Application

In the General category, select Application to invoke the Create Application dialog box.

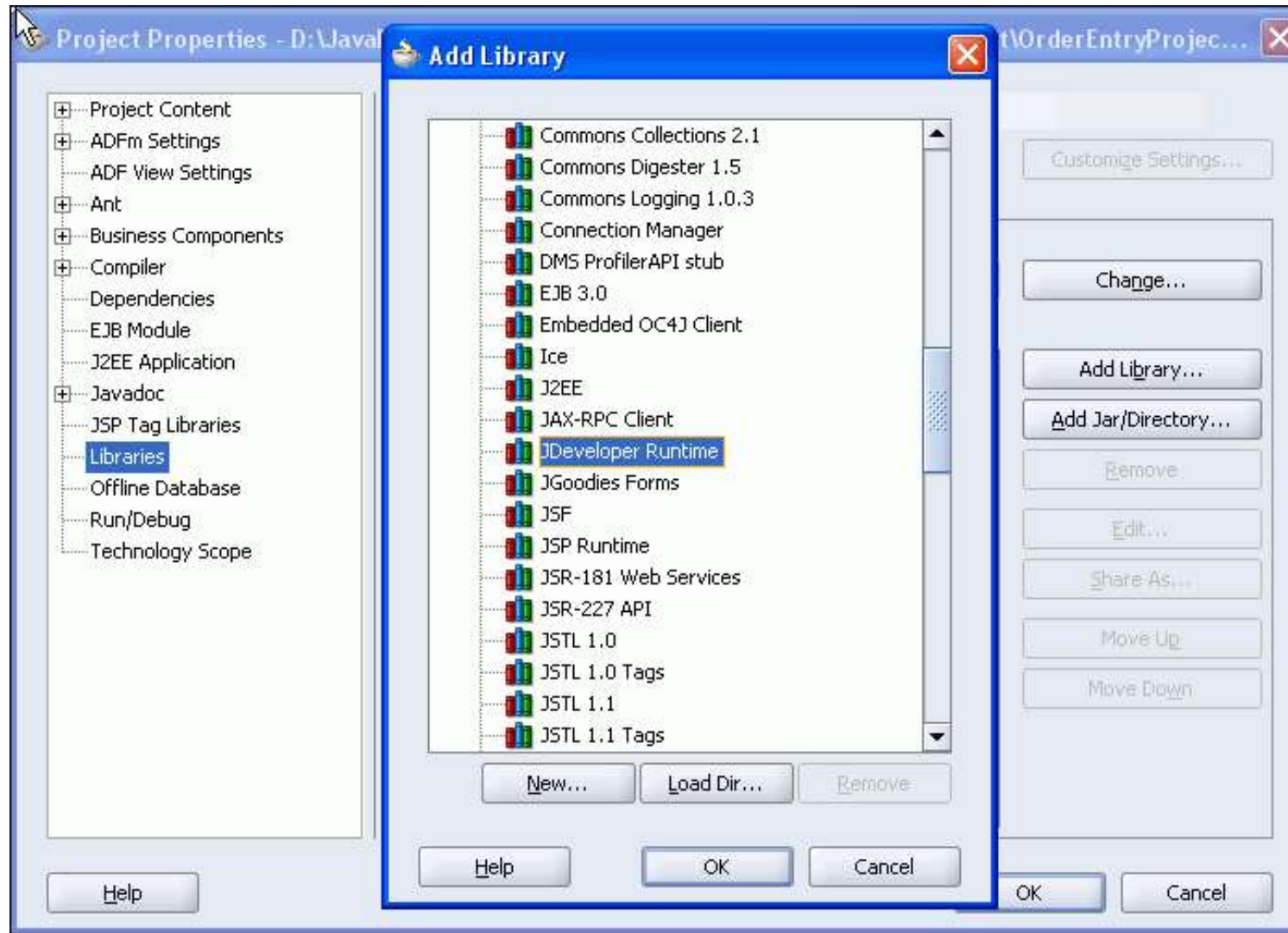




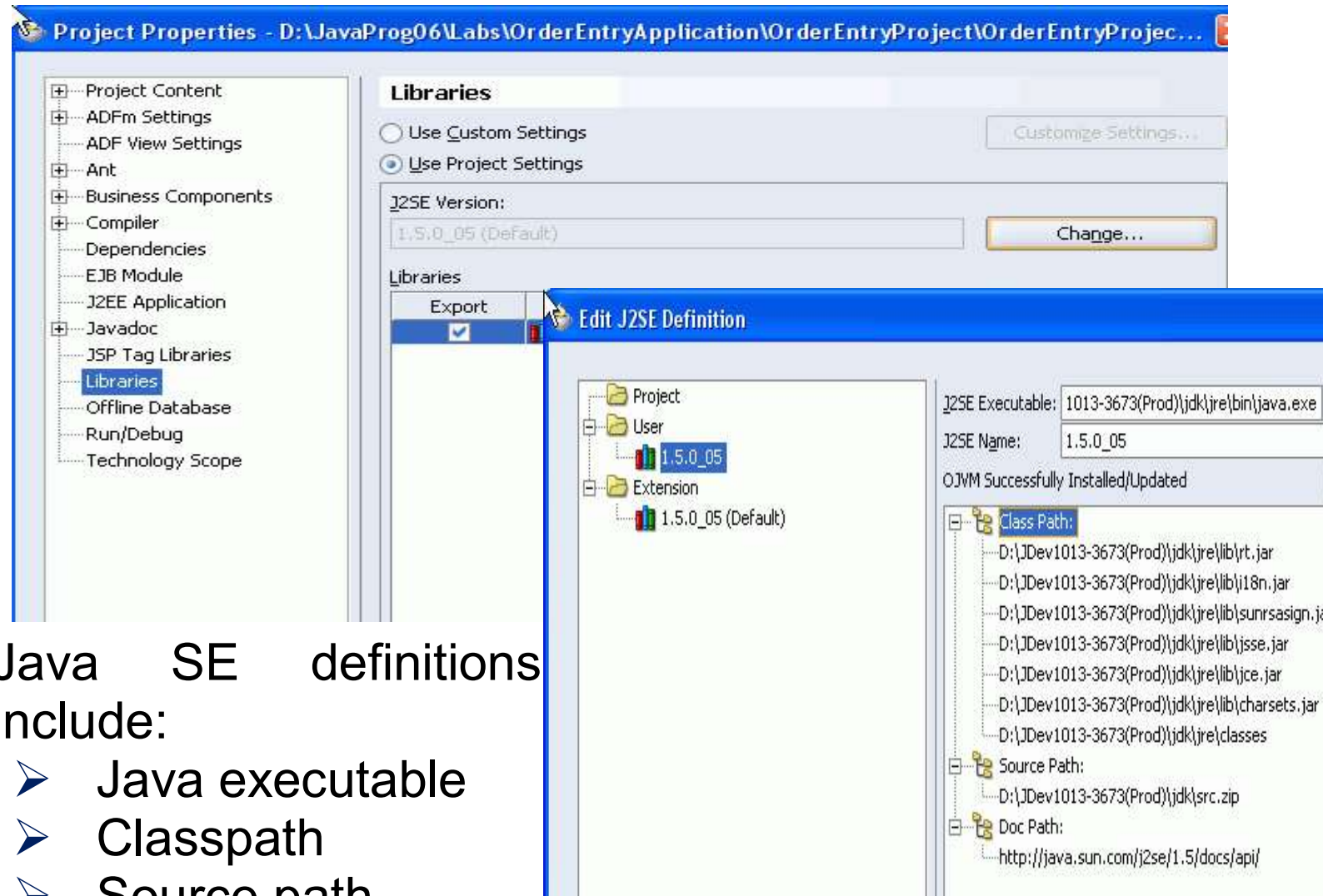
# Project Properties : Specifying Project Details



# Project Properties : Selecting Additional Libraries



# Adding a New Java SE



Java SE definitions include:

- Java executable
- Classpath
- Source path
- Doc path

# Directory Structure

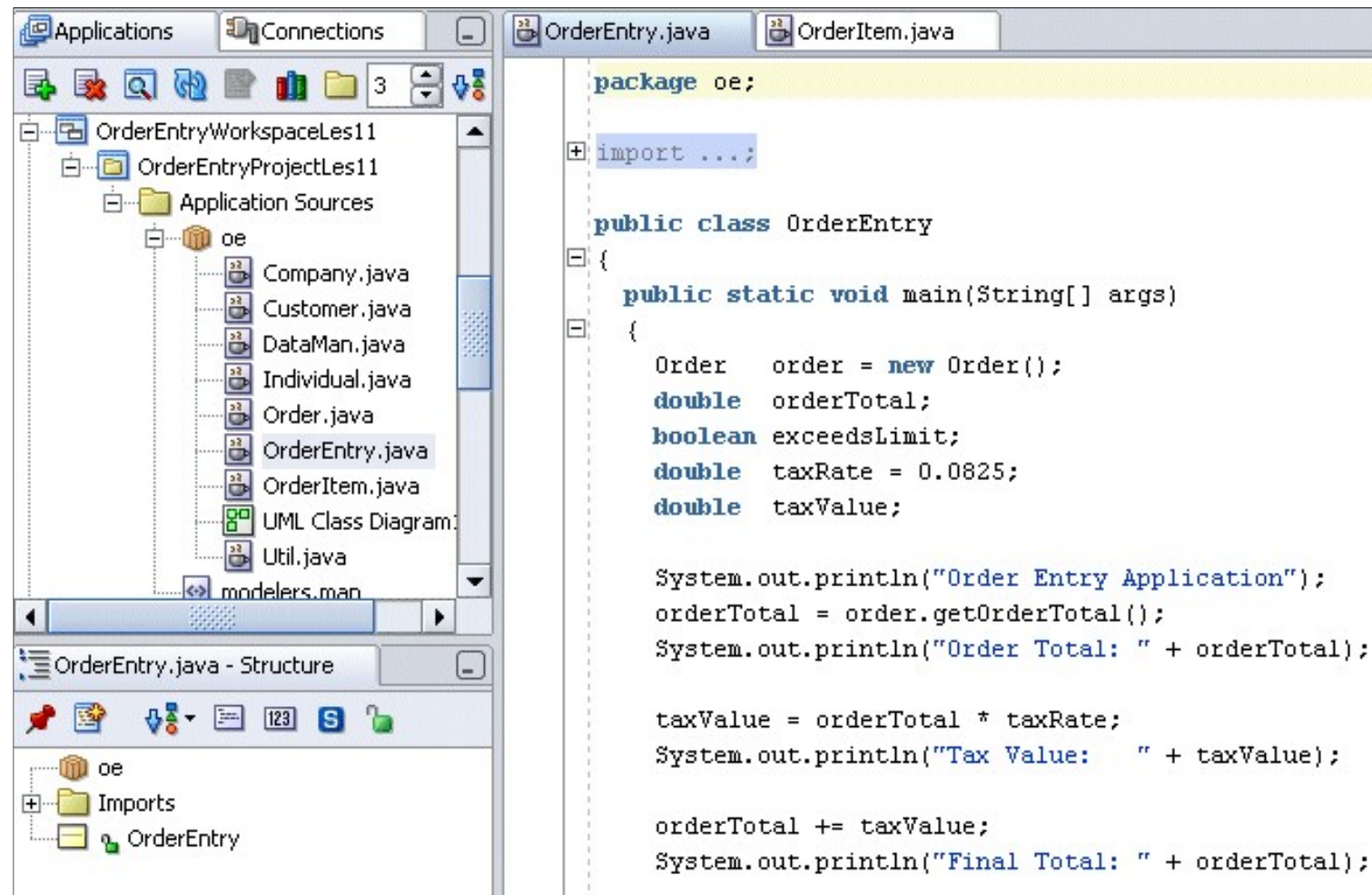
JDeveloper creates and stores `.java` and `.class` files by using the following conventions:

- `<ORACLE_HOME>\jdev\mywork`
- Followed by the application name
- Followed by the project name
  - `\classes\<package name>\`
  - `\src\<package_name>\`
- Followed by `class` and `src` files



# Exploring the Skeleton Java Application

Contains application and frame classes:



The screenshot displays an IDE interface with two main panels. The left panel shows a project tree for 'OrderEntryWorkspaceLes11' containing 'OrderEntryProjectLes11' and 'Application Sources'. Under 'Application Sources', there is a package 'oe' containing several Java files: 'Company.java', 'Customer.java', 'DataMan.java', 'Individual.java', 'Order.java', 'OrderEntry.java', 'OrderItem.java', 'UML Class Diagram', and 'Util.java'. The right panel shows the source code of 'OrderEntry.java'.

```
package oe;

import ...;

public class OrderEntry
{
    public static void main(String[] args)
    {
        Order    order = new Order();
        double   orderTotal;
        boolean  exceedsLimit;
        double   taxRate = 0.0825;
        double   taxValue;

        System.out.println("Order Entry Application");
        orderTotal = order.getOrderTotal();
        System.out.println("Order Total: " + orderTotal);

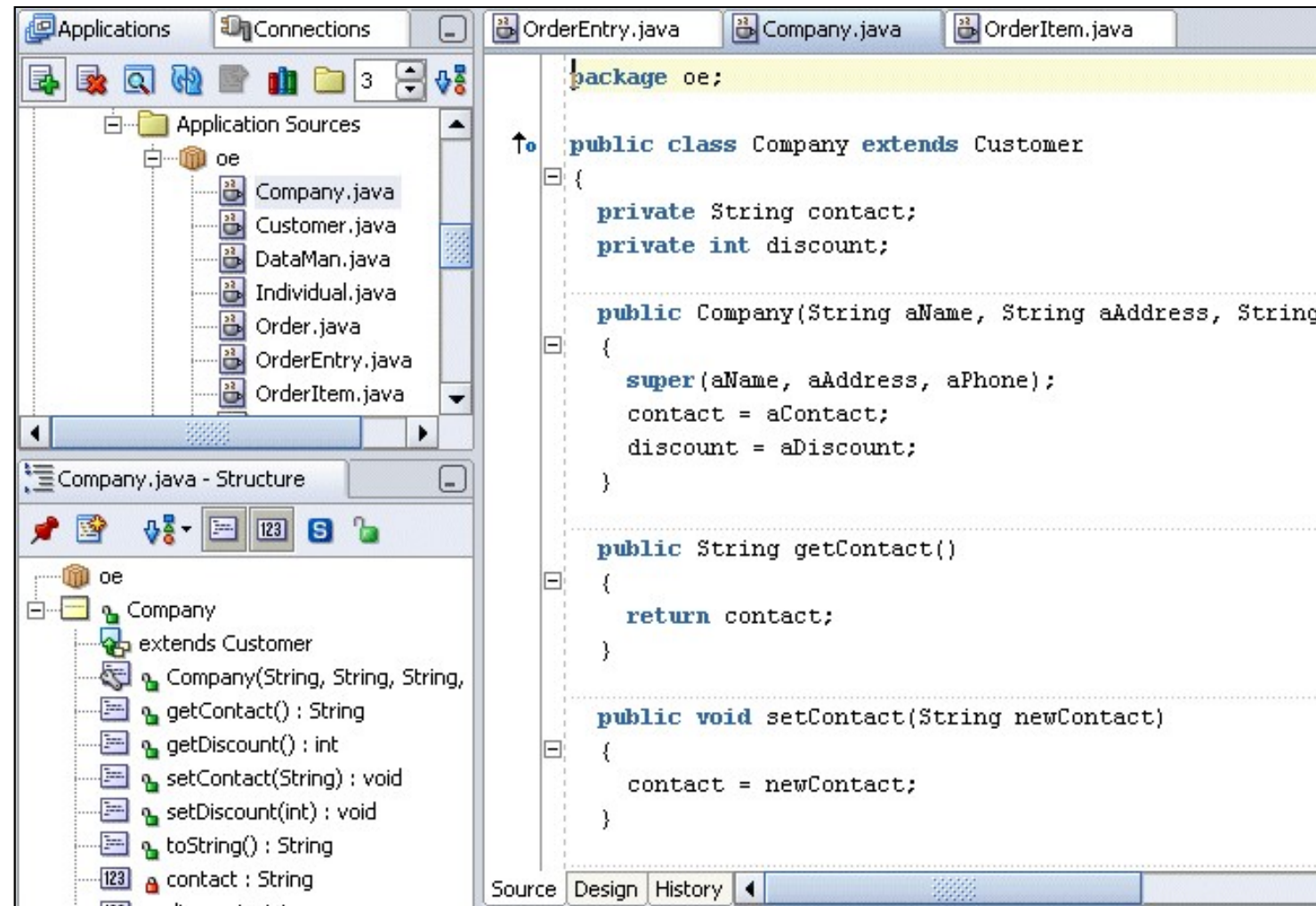
        taxValue = orderTotal * taxRate;
        System.out.println("Tax Value:   " + taxValue);

        orderTotal += taxValue;
        System.out.println("Final Total: " + orderTotal);
    }
}
```



# Finding Methods and Fields

Find methods and fields using the Structure pane:





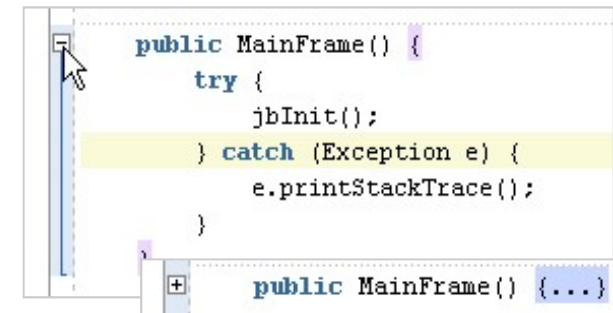
## Supporting Code Development

- Improve code quality with Code Coach.
- Evaluate execution stack with the Execution Sample profiler.
- Examine heap memory usage with the Memory profiler.
- Analyze event occurrence and duration with the Event profiler for:
  - JVM events
  - Business components for Java events
  - Custom events

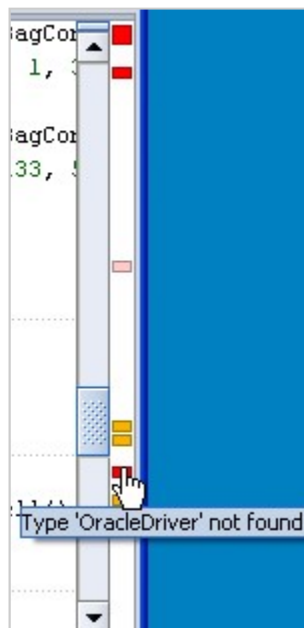
# New Code Editor Features



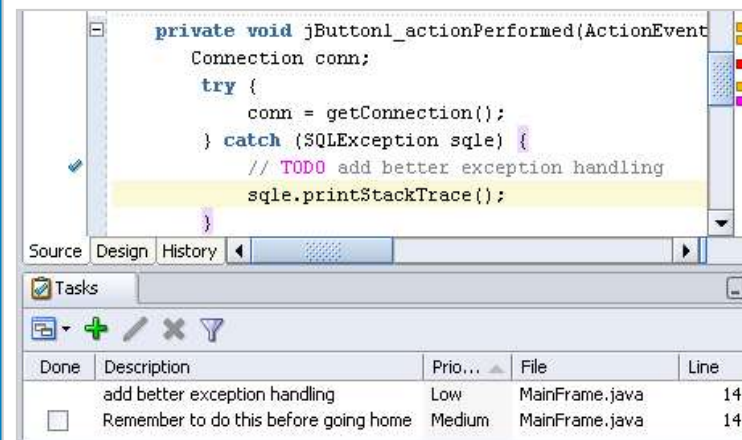
Code Assist



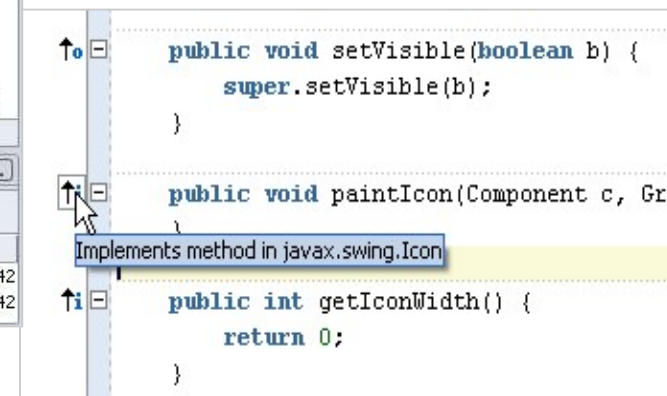
Scope and code folding



Overview margin



Tasks list



Implements and overrides navigation

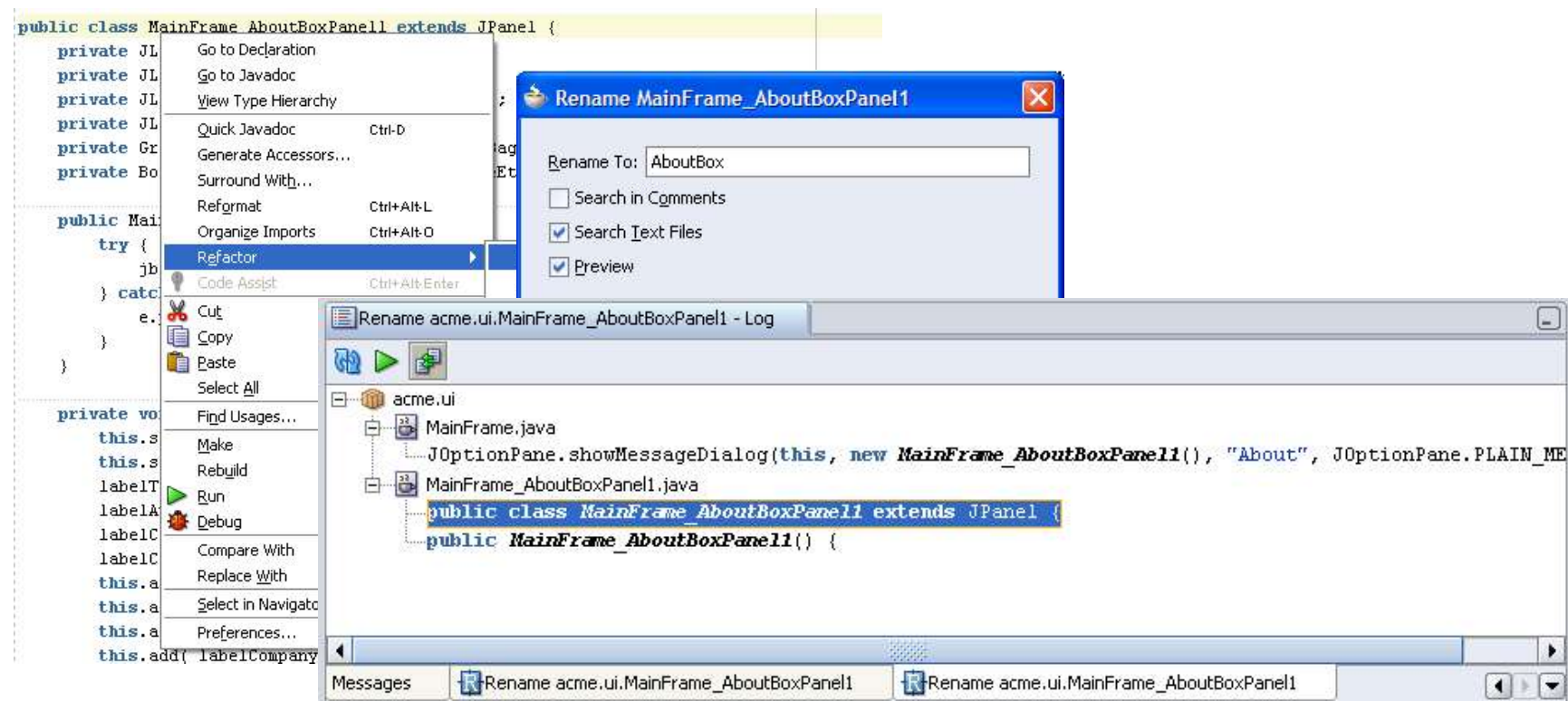
# Customizing JDeveloper (10.1.3.2.0)

Customize the IDE:

- Look and feel
- General environment
- Dockable windows
- Component Palette
- Preset keymaps

# Refactoring

Modify the structure of code without changing its behavior  
(or breaking it).



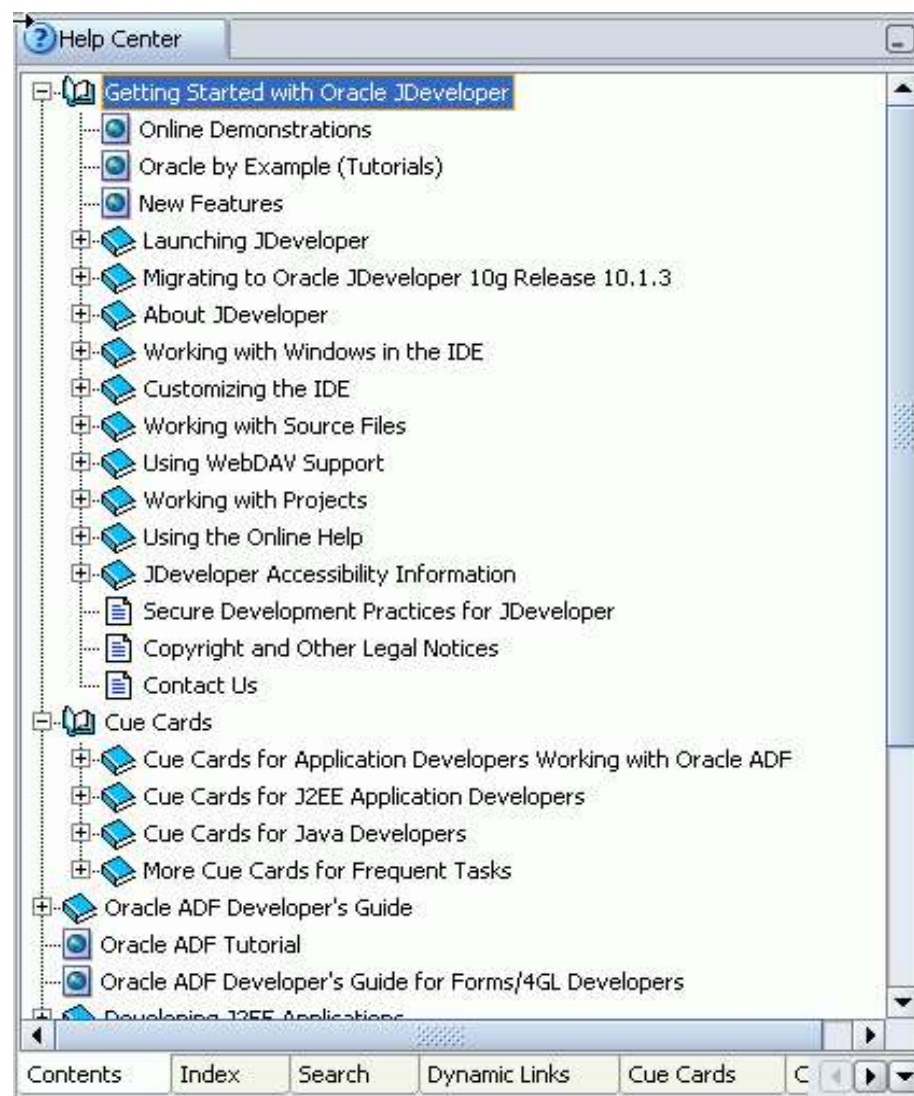
# Refactoring

- Drag-and-drop refactoring
- Refactor across entire application
- Refactor across source control
- More than 35 new refactoring operations, including:



- |                                  |                                       |
|----------------------------------|---------------------------------------|
| - <b>Rename Class</b>            | - <b>Introduce Field</b>              |
| - <b>Rename Field</b>            | - <b>Extract Interface</b>            |
| - <b>Rename Method</b>           | - <b>Use Supertype Where Possible</b> |
| - <b>Rename Package</b>          | - <b>Move Class</b>                   |
| - <b>Rename Parameter</b>        | - <b>Duplicate Class</b>              |
| - <b>Change Method Signature</b> | - <b>Pull Members Up</b>              |
| - <b>Introduce Variable</b>      | - <b>Safe Delete</b>                  |

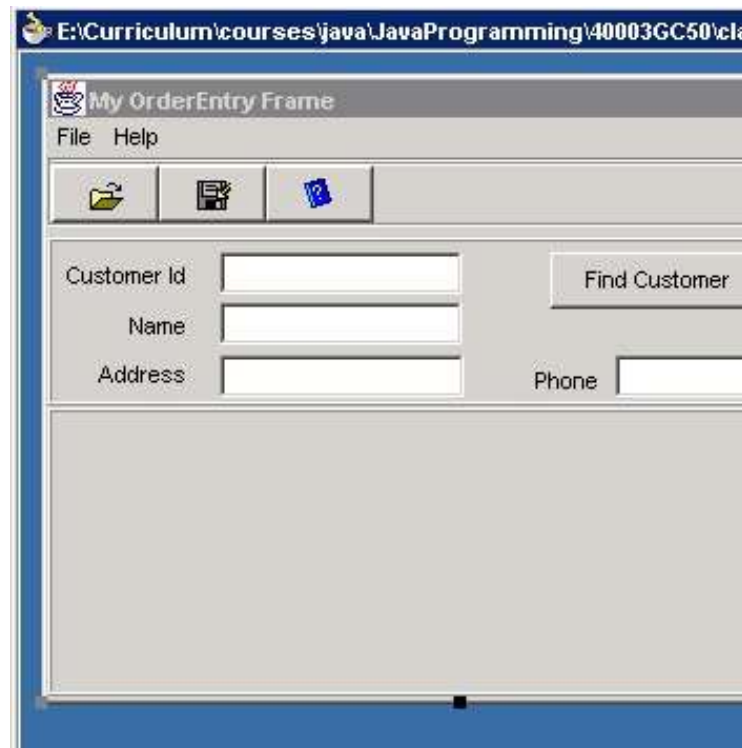
# JDeveloper Help System





# Obtaining Help on a Topic

Use [F1] to invoke context-specific help.



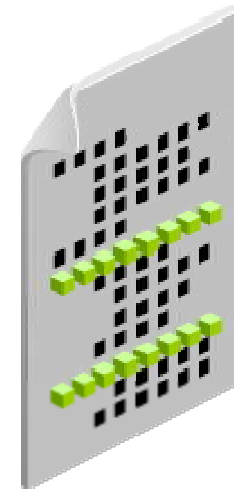
- Helps find and fix program errors:
  - Run-time errors
  - Logic errors
- Allows control of execution
- Allows examination of variables



# Breakpoints

Setting breakpoints:

- Manage multiple breakpoints
- Manage conditional breakpoints
- Define columns displayed in window
  - Description
  - Type
  - Status
- Control scope of action
  - Global > Application > Project



View debugging information:

- Classes: Displays list of loaded classes and status
- Watch: Evaluates and displays expressions
- Monitors: Displays information about active monitors
- Threads: Displays the names and statuses of all threads
- Smart Data: Analyzes source code near execution point
- ... and more

# Stepping Through a Program

Use the buttons on the debugger toolbar:

- Start the debugger.
- Resume the program.
- Step over a method call.
- Step into a method call.
- Step out of a method call.
- Step to the end of the method.
- Pause execution.
- Stop the debugger.



# Watching Data and Variables

- The Smart Data tab displays analyzed variables and fields.
- The Data tab displays arguments, local variables, and static fields from the current context.
- To watch other variables:
  1. Select a variable in the source window and right-click.
  2. Select Watch... at Cursor from the shortcut menu.
  3. View the variable in the Watch tab.
  4. Right-click a data item to modify it.



# Summary

In this lesson, you should have learned that:

- JDeveloper builds, debugs, and runs all types of Java applications
- JDeveloper can be used to develop:
  - Java applications
  - Java servlets
  - JSPs
  - EJBs
- JDeveloper can be used to build enterprise applications



## Practice : Overview

This practice covers the following topics:

- Exploring the JDeveloper IDE
- Creating an application and a project
- Populating the project with existing files

