

PUE DAY  
ABRIL 2016



10:15 - 11:20

## Workshop Oracle

“Curso oficial Java Foundations: implementación curricular”

Jordi Ariño, Responsable Desarrollo Software, PUE

ORACLE ACADEMY



# PUE DAY

---

## WORKSHOP ORACLE

“Curso oficial Java Foundations: implementación curricular”

Jordi Ariño, Responsable Desarrollo Software, PUE

MADRID - 13/04/2016

# Curso oficial Java Foundations: implementación curricular

Jordi Ariño Santos – *Agile Developer at PUE*  
jordi.arino@pue.es



# Acerca de...



## Jordi Ariño

Agile Developer at PUE  
Certified ScrumMaster (CSM)







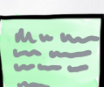

[jordi.arino@pue.es](mailto:jordi.arino@pue.es)  
[@jordiAS2K](https://twitter.com/jordiAS2K)





# Agenda

- 1.- Introducción
- 2.- Plan de formación y certificación en Java
- 3.- Curso Java Foundations
- 4.- Estructura del curso Java Foundations
- 5.- Demo: Curso Java Foundations
- 6.- Oracle iLearning: Informes y diplomas

	To do	Doing	Done
			
			
			
			

Kanban Board  
(Visual Management)

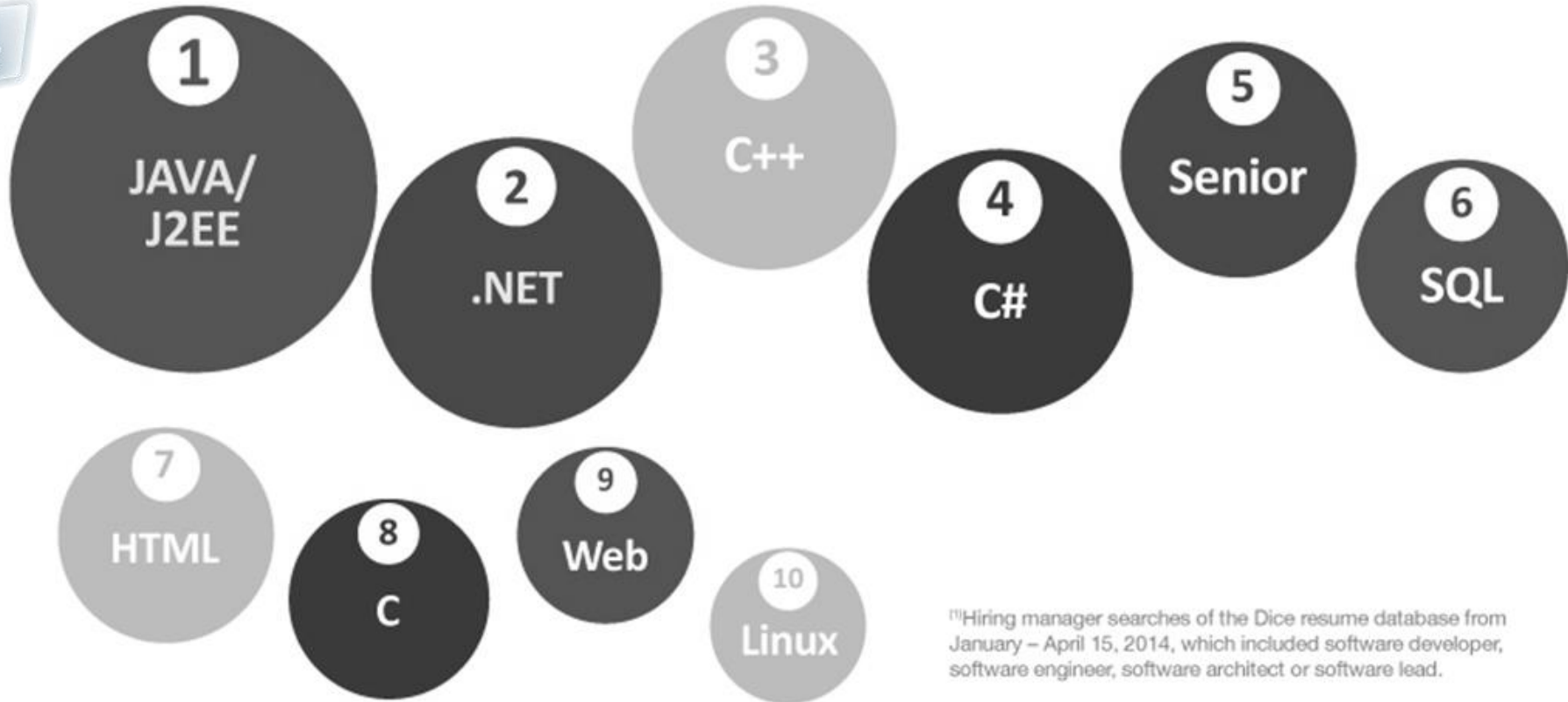


# Introducción



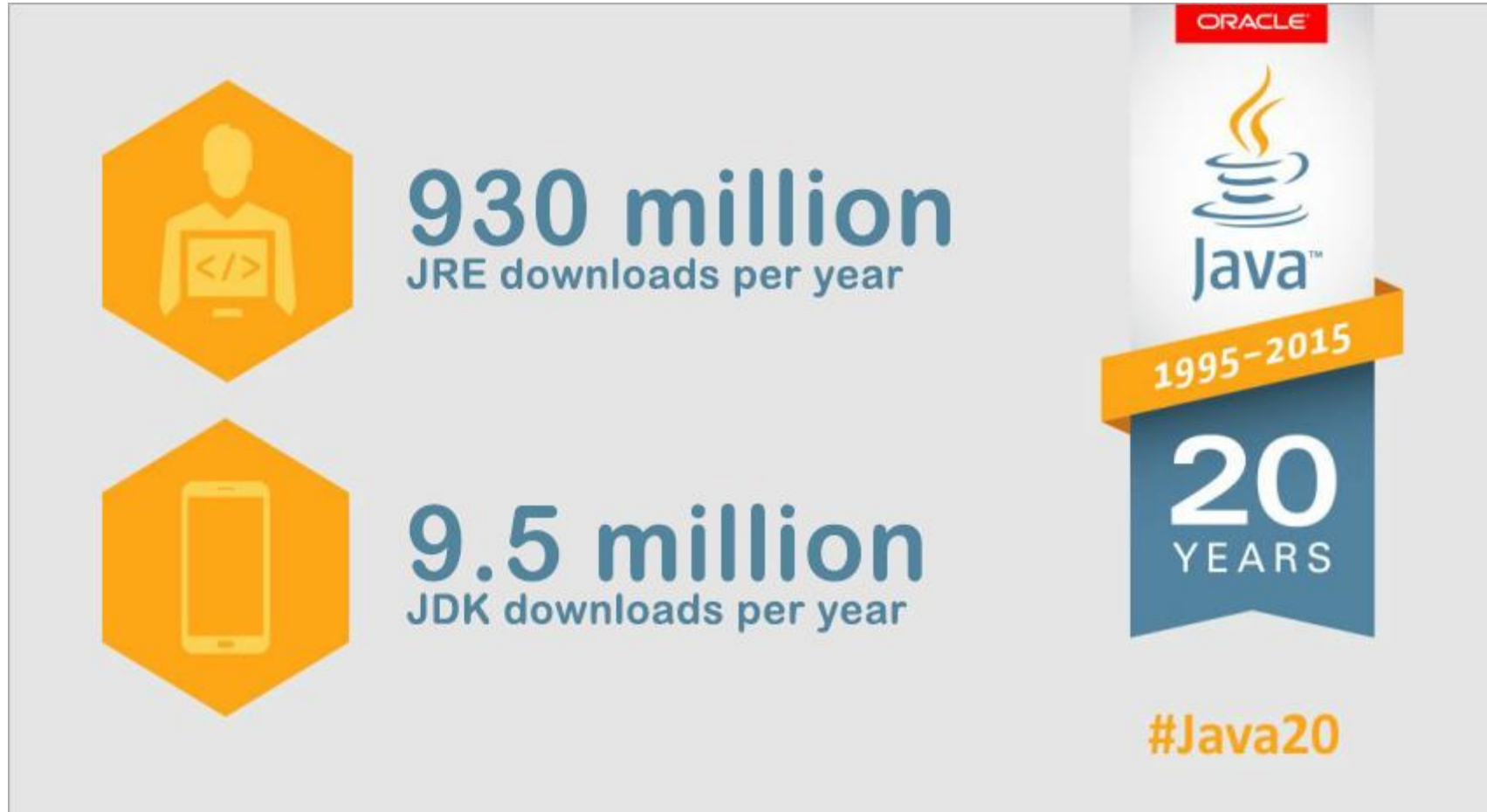
# Introducción

## Most Desired Skills: Software Developers



<sup>1)</sup>Hiring manager searches of the Dice resume database from January – April 15, 2014, which included software developer, software engineer, software architect or software lead.

# Introducción



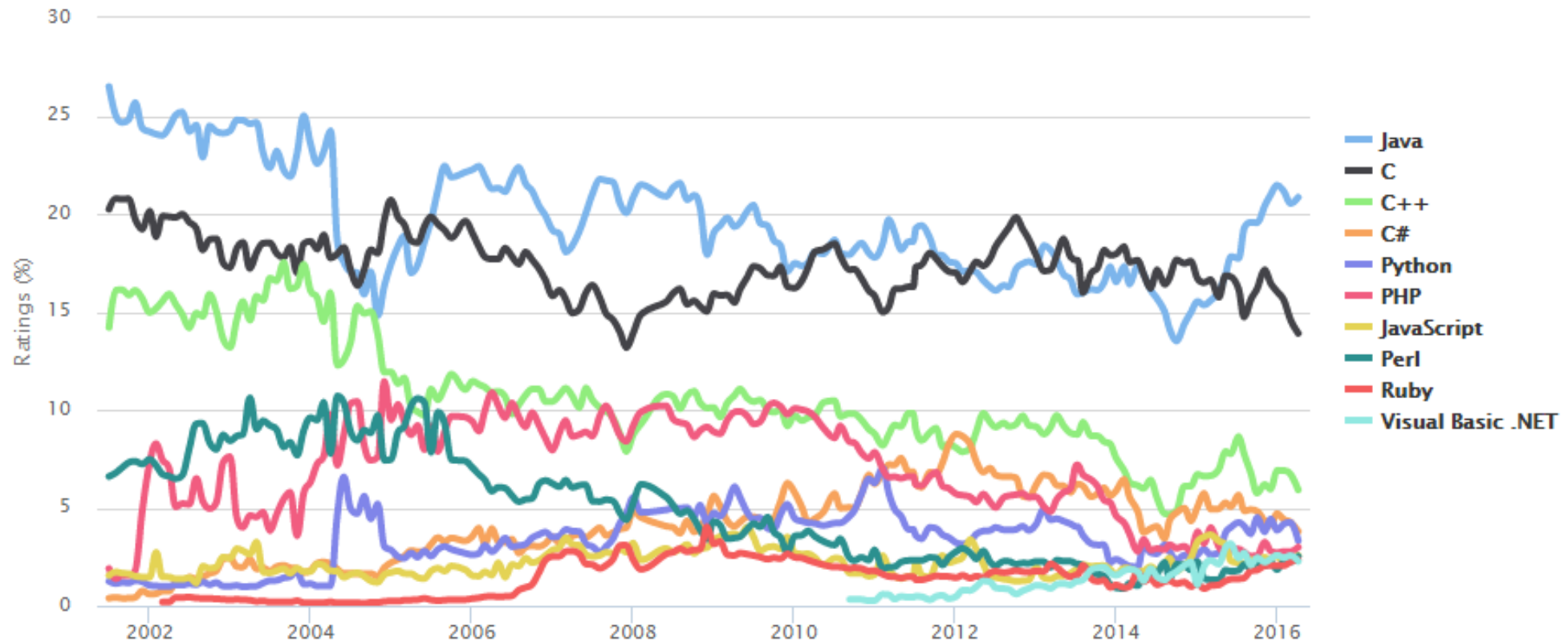


# Introducción



TIOBE Programming Community Index







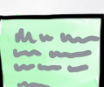

Source: [www.tiobe.com](http://www.tiobe.com)





# Agenda

- 1.- Introducción
- 2.- Plan de formación y certificación en Java
- 3.- Curso Java Foundations
- 4.- Estructura del curso Java Foundations
- 5.- Demo: Curso Java Foundations
- 6.- Oracle iLearning: Informes y diplomas

	To do	Doing	Done
			
			
			
			

Kanban Board  
(Visual Management)



# Plan de formación y certificación en Java

**ORACLE®**  
**Certified Associate**

La credencial *Oracle Certified Associate (OCA)* valida las **habilidades** y **conocimientos fundamentales** que forman una **base sólida de conocimiento** en los productos de Oracle. Convertirse en Oracle Certified Associate es el primer paso que se debe tomar para convertirse en un profesional certificado de Oracle.

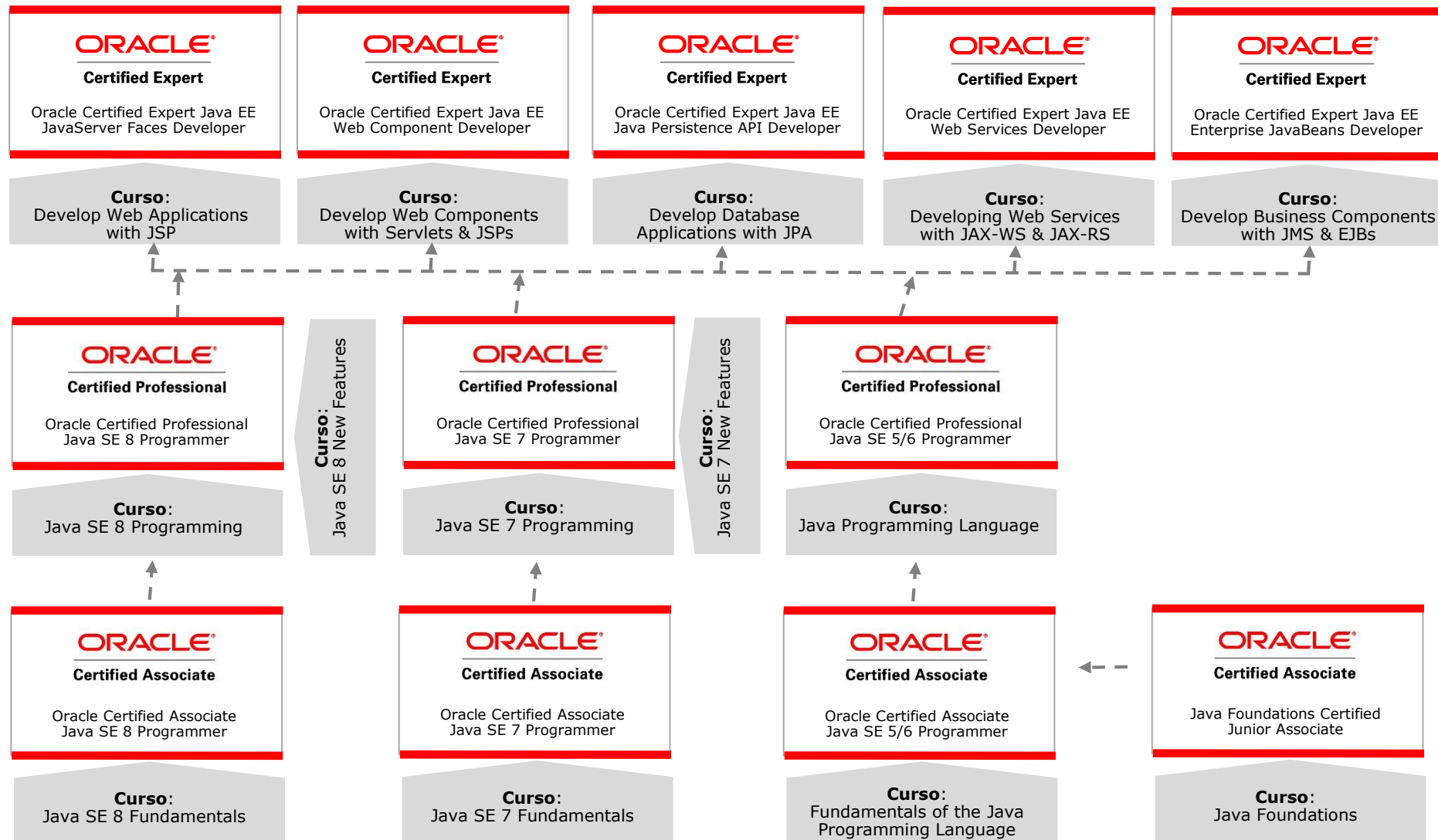
**ORACLE®**  
**Certified Professional**

La credencial *Oracle Certified Professional (OCP)* es el punto de referencia de la **cualificación profesional** y **experiencia técnica** en productos de Oracle. Valida la capacidad para administrar, desarrollar o implementar una amplia gama de productos de Oracle, incluyendo bases de datos, desarrollo de aplicaciones en Java y más. En las empresas a menudo se utiliza la credencial OCP para evaluar las habilidades y experiencia de los empleados y/o posibles candidatos.

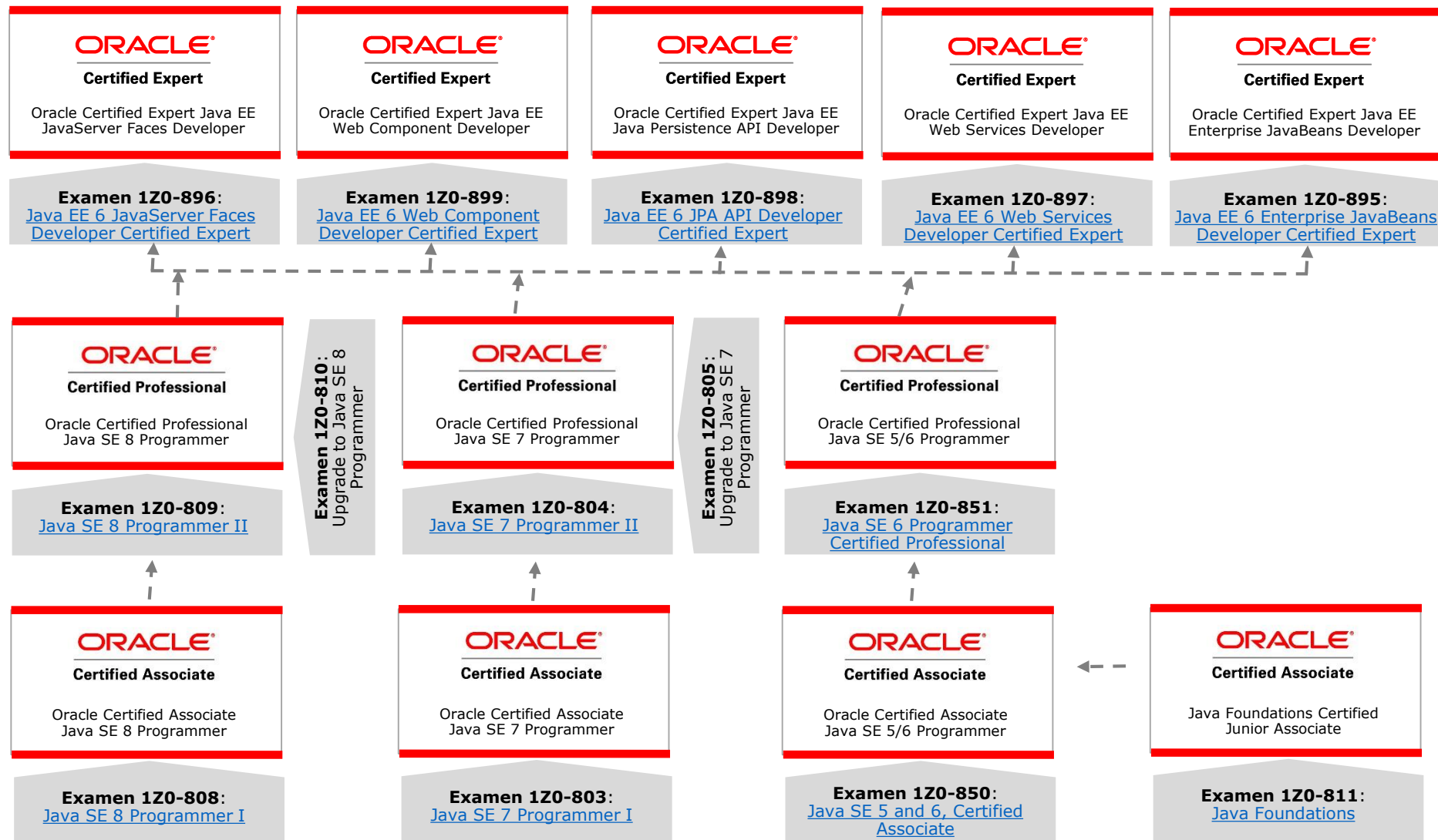
**ORACLE®**  
**Certified Expert**

La credencial *Oracle Certified Expert (OCE)* valida **experiencia** y **habilidades en tecnologías específicas**, arquitecturas o dominios que no están cubiertos específicamente dentro del path de certificaciones estándar o tradicional.

# Plan de formación y certificación en Java



# Plan de formación y certificación en Java



# Plan de formación y certificación en Java



**ORACLE®**  
**Certified Professional**  
Oracle Certified Professional  
Java SE Programming

**Curso:**  
Java Programming

**ORACLE®**  
**Certified Associate**  
Oracle Certified Associate  
Java SE Programming

**Curso:**  
Java Fundamentals

**ORACLE®**  
**Certified Associate**  
Java Foundations Certified  
Junior Associate

**Curso:**  
Java Foundations

Creating Java Programs  
with Greenfot

Getting started with Java  
using Alice

# Plan de formación y certificación en Java

## Workshop Descriptions

### Getting Started with Java Using Alice

This workshop is designed for students with little or no programming experience and teaches basic Java programming concepts through developing 3-D Animations in Alice 3.1. Students will have fun creating animated stories and games using objects and characters from a rich gallery of 3-D models.

**Duration 8 Hours**

### Creating Java Programs with Greenfoot

This workshop engages students who understand basic programming concepts to create 2-D games using Java. Students will learn detailed object-oriented programming terminology and concepts while creating 2-D games in a fun and interactive environment.

**Duration 16 Hours**

ORACLE

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. |

# Plan de formación y certificación en Java

ORACLE ACADEMY Alice Workshop - L1 - Get Started with Alice 3 Exit

Outline Notes

- Oracle Academy
- Getting Started with Java Using Alice**
- Objectives
- Understand the initial Scene
- Initial Scene Components
- Steps to Create a New Project
- Steps to Save a Project
- Navigating Between Editors
- Navigating Between Editors
- The Default Editor
- Add an Object to a Scene
- Add an Object to a Scene Display
- Naming the Object
- Scene Editor
- Scene Editor Display
- Gallery
- Gallery Tabs
- Select a Class
- Class Example
- Save New Project Version
- Steps to Save a Project Version
- Code Editor
- Methods Panel
- Instance Menu in Methods

## Getting Started with Java Using Alice

### Lesson 1 Get Started with Alice 3

Robot classes 39 Flyer classes 12 Pre classes 223 Customized classes 30 Swimmer classes 2 Transport classes 3

ORACLE ACADEMY

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

< PREVIOUS NEXT >



# Plan de formación y certificación en Java


ORACLE ACADEMY Alice Workshop - L1 - Get Started with Alice 3 Exit

Outline Notes

- Oracle Academy
- Getting Started with Java Using Alice
- Objectives
- Understand the initial Scene
- Initial Scene Components**
- Steps to Create a New Project
- Steps to Save a Project
- Navigating Between Editors
- Navigating Between Editors
- The Default Editor
- Add an Object to a Scene
- Add an Object to a Scene Display
- Naming the Object
- Scene Editor
- Scene Editor Display
- Gallery
- Gallery Tabs
- Select a Class
- Class Example
- Save New Project Version
- Steps to Save a Project Version
- Code Editor
- Methods Panel
- Instance Menu in Methods

## Initial Scene Components

- Below are components of a room scene.



Room is a template.

Furniture Items are scenery objects.

Bipeds are acting objects.

ORACLE ACADEMY AWL1 Get Started with Alice 3 Copyright © 2014, Oracle and/or its affiliates. All rights reserved. 5

< PREV NEXT >

# Plan de formación y certificación en Java

The screenshot displays the Oracle Academy Alice Workshop interface. The title bar reads "Alice Workshop - L1 - Get Started with Alice 3". The Oracle Academy logo is in the top left. On the left is a navigation sidebar with an "Outline" tab. The main content area is titled "Select a Class" and contains a bulleted list: "The Class Hierarchy tab groups objects by mobility type (biped, flyer, etc.)." Below this is a text box explaining that a class contains instructions for appearance and movement. At the bottom of the main area is a gallery titled "Browse Gallery By Class Hierarchy" with tabs for "all classes", "Browse Gallery By Theme", "Browse Gallery By Group", "Search Gallery", "Shapes/Text", and "My Classes". The gallery shows six categories: "Biped classes" (39 items), "Flyer classes" (12 items), "Prop classes" (223 items), "Quadruped classes" (30 items), "Swimmer classes" (2 items), and "Transport classes" (3 items). The footer includes the Oracle Academy logo, "AWL1 Get Started with Alice 3", copyright information, and page number 18. Navigation buttons for "PREV" and "NEXT" are at the bottom right.

ORACLE ACADEMY Alice Workshop - L1 - Get Started with Alice 3 Exit

**Outline** Notes

- Oracle Academy
- Getting Started with Java Using Alice
- Objectives
- Understand the initial Scene
- Initial Scene Components
- Steps to Create a New Project
- Steps to Save a Project
- Navigating Between Editors
- Navigating Between Editors
- The Default Editor
- Add an Object to a Scene
- Add an Object to a Scene Display
- Naming the Object
- Scene Editor
- Scene Editor Display
- Gallery
- Gallery Tabs
- Select a Class**
- Class Example
- Save New Project Version
- Steps to Save a Project Version
- Code Editor
- Methods Panel
- Instance Menu in Methods

## Select a Class

- The Class Hierarchy tab groups objects by mobility type (biped, flyer, etc.).

A class contains the instructions that define the appearance and movement of an object. All objects within a class have common properties. The class provides instructions to Alice 3 for creating and displaying the object when it is added to your scene.

Browse Gallery By Class Hierarchy Browse Gallery By Theme Browse Gallery By Group Search Gallery Shapes/Text My Classes

all classes

39	12	223	30	2	3
Biped classes	Flyer classes	Prop classes	Quadruped classes	Swimmer classes	Transport classes

ORACLE ACADEMY AWL1 Get Started with Alice 3 Copyright © 2014, Oracle and/or its affiliates. All rights reserved. 18

< PREV NEXT >

# Plan de formación y certificación en Java


ORACLE ACADEMY Alice Workshop - L3 - Use Procedures and Arguments

Outline Notes

- Oracle Academy
- Getting Started with Java Using Alice
- Objectives
- Objectives
- Display the Code Editor**
- Select Instance
- Methods Panel
- Procedures Tab
- Functions Tab
- Code Editor Tabs
- Control Statements
- Object Movement
- Examples of Movement Procedures
- Examples of Rotation Procedures
- Create a Programming Instruction
- Select and Set Argument Values
- Execute the Program
- Arguments
- Argument Menu
- Steps to Edit Arguments
- Arguments as Placeholders
- Steps to Reorder Programming Statements

## Display the Code Editor

- Click Edit Code (from the scene editor) to display the Code editor. The Code editor is where you program your animation.



ORACLE ACADEMY AWL3 Use Procedures and Arguments Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 5

< PREV NEXT >

# Plan de formación y certificación en Java

ORACLE ACADEMY

Greenfoot Workshop - L1 - Prepare for this course

Exit

Outline Notes

Oracle Academy

**Creating Java Programs with Greenfoot**

Overview

Download and install Greenfoot

Download and install Greenfoot

Download and install Greenfoot

Download and install Greenfoot

Download and install Greenfoot

Download and install Greenfoot

Launch Greenfoot

Greenfoot Textbook Scenarios

Steps to Download the Greenfoot Textbook Scenarios

Terminology

Try it

Summary

Oracle Academy

## Creating Java Programs with Greenfoot

### Lesson 1

#### Prepare for this course



ORACLE ACADEMY

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

< PREV NEXT >

# Plan de formación y certificación en Java



**ORACLE ACADEMY** Greenfoot Workshop - L1 - Prepare for this course Exit

**Outline** Notes

- Oracle Academy
- Creating Java Programs with Greenfoot
- Overview
- Download and install Greenfoot**
- Download and install Greenfoot
- Download and install Greenfoot
- Download and install Greenfoot
- Download and install Greenfoot
- Launch Greenfoot
- Greenfoot Textbook Scenarios
- Steps to Download the Greenfoot Textbook Scenarios
- Terminology
- Try it
- Summary
- Oracle Academy

## Download and install Greenfoot

- Visit this link to download and install Greenfoot:  
<http://www.greenfoot.org/download>
- Locate and select the correct version for your computer.

<b>Windows</b> For XP, Vista, 7 	<b>Mac OS X</b> Needs at least 10.5 ( Leopard) 	<b>Mac OS X</b> Includes JDK 7 Needs at least 10.7.3 ( Lion) 
<b>Ubuntu</b> Supports Debian 	<b>Pure Java</b> For any OS 	<b>Stand Alone</b> Includes BlueJ and JDK, runs on Windows from a USB stick 

**ORACLE ACADEMY** GFWL1 Prepare for this course Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 4

< PREV NEXT >

# Plan de formación y certificación en Java

ORACLE ACADEMY Greenfoot Workshop - L3 - Using Methods, Variables, and Parameters Exit

Outline Notes

- Oracle Academy
- Creating Java Programs with Greenfoot
- Overview
- Methods Example
- Methods
- Inheritance
- View Inherited Methods in Object Menu**
- Steps to View Inherited Methods in the Code Editor
- Method Summary
- Method Components
- Method Signature
- Return Types
- Methods with Void Return Types
- Invoking Methods with Void Return Types
- Ask Objects Questions
- Methods with Non-void Return Types
- Examples of Non-Void Return Types
- Method Parameters
- Examples of Method Parameters
- Method Parameter Lists
- Object Properties

## View Inherited Methods in Object Menu

- The object menu displays all of the methods that the instance inherits from its class and superclass.
  - Right click on the instance to display the menu.
  - Inherited From Actor displays a list of the methods that the class inherits from the Actor superclass.

ORACLE ACADEMY GFWL3 Using Methods, Variables, and Parameters Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 7

< PREV NEXT >

# Plan de formación y certificación en Java

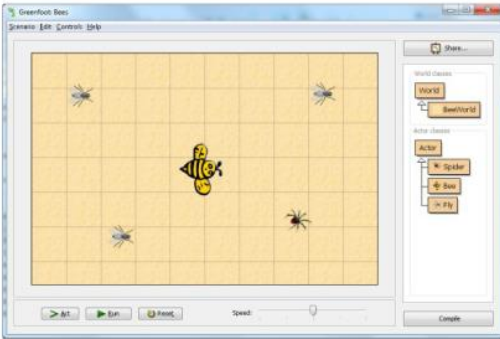
ORACLE ACADEMY Greenfoot Workshop - L5 - Developing and Testing an Application Exit

**Outline** Notes

- Oracle Academy
- Creating Java Programs with Greenfoot
- Objectives
- Program Testing Strategies**
- Compilation and Debugging
- Steps to Debug Your Program
- Keys to Recognizing Java Syntax Errors
- Auto-Layout
- Phases to Develop an Application
- Analysis Phase**
- Analysis Phase Tasks
- Analysis Example
- Analysis Pre and Post Conditions
- Design Phase
- Textual Storyboard Example
- Textual Storyboard Example
- Development Phase**
- Testing Phase
- Testing Numerical Representations and Limits Example 1
- Testing Numerical Representations and Limits Example 2
- Terminology

## Development Phase

- After you finalize your storyboard, develop your game in Greenfoot.
- Refer to your storyboard to determine the methods you need to program.



ORACLE ACADEMY JGFWL5 Developing and Testing an Application Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 17

< PREVIOUS NEXT >

# Plan de formación y certificación en Java



**ORACLE®**  
**Certified Professional**  
Oracle Certified Professional  
Java SE Programming

**Curso:**  
Java Programming

**ORACLE®**  
**Certified Associate**  
Oracle Certified Associate  
Java SE Programming

**Curso:**  
Java Fundamentals

**ORACLE®**  
**Certified Associate**  
Java Foundations Certified  
Junior Associate

**Curso:**  
Java Foundations

Creating Java Programs  
with Greenfot







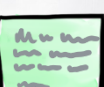

Getting started with Java  
using Alice





# Agenda

- 1.- Introducción
- 2.- Plan de formación y certificación en Java
- 3.- Curso Java Foundations**
- 4.- Estructura del curso Java Foundations
- 5.- Demo: Curso Java Foundations
- 6.- Oracle iLearning: Informes y diplomas

	To do	Doing	Done
			
			
			
			

Kanban Board  
(Visual Management)



# Curso Java Foundations

ORACLE ACADEMY [www.oracle.com/academy](http://www.oracle.com/academy)

## Java Foundations – Course Objectives

### Overview

This course of study engages students with little programming experience. Students are introduced to object-oriented concepts, terminology, and syntax, and the steps required to create basic Java programs using hands-on, engaging activities. Students will learn the concepts of Java programming, design object-oriented applications with Java and create Java programs using hands-on, engaging activities.

### Duration

- Recommended total course time: 90 hours\*
- Professional education credit hours for educators who complete Oracle Academy training: 30

\* total course time includes instruction, self-study/homework, practices, projects and assessment

### Target Audiences

#### Educators

- Technical, vocational, and 2- and 4-year college and university faculty members who teach computer programming, information communications technology (ICT), or a related subject at a foundational level.
- Secondary and vocational school teachers who teach computer programming.

#### Students

- Students who wish learn Java programming and build their Object Oriented Programming experience using Java.
- This course is a suitable foundational class for computer science majors, and when taught in sequence with Java Programming may be used to prepare students for the AP Computer Science A exam.

### Prerequisites

#### Required

- Oracle Academy Workshop - Getting Started with Java Using Alice
- Oracle Academy Workshop - Creating Java Programs with Greenfoot

#### Suggested

- Oracle Academy Course - Java Fundamentals

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## Descripción del curso

- Dirigido a los estudiantes con **poca o ninguna experiencia** en programación.
- Los estudiantes aprenderán la **sintaxis** y los **conceptos básicos de programación** en Java, así como a **diseñar sus primeras aplicaciones orientadas a objetos** dentro la plataforma Java.
- Ofrece gran variedad de **actividades, tests, juegos y prácticas** que fomentan la participación y aprendizaje de los estudiantes.
- Ofrece una **base de conocimiento sólida y robusta** en la plataforma Java y el paradigma OOP. Permite comenzar a coger experiencia en el uso del API nativo de Java.
- Enseña conceptos fundamentales en el desarrollo de software, además de buenas prácticas, como **clean code** y **refactorización**.

# Curso Java Foundations

ORACLE ACADEMY

[www.oracle.com/academy](http://www.oracle.com/academy)

## Java Foundations – Course Objectives

### Overview

This course of study engages students with little programming experience. Students are introduced to object-oriented concepts, terminology, and syntax, and the steps required to create basic Java programs using hands-on, engaging activities. Students will learn the concepts of Java programming, design object-oriented applications with Java and create Java programs using hands-on, engaging activities.

### Duration

- Recommended total course time: 90 hours\*
- Professional education credit hours for educators who complete Oracle Academy training: 30

\* total course time includes instruction, self-study/homework, practices, projects and assessment

### Target Audiences

#### Educators

- Technical, vocational, and 2- and 4-year college and university faculty members who teach computer programming, information communications technology (ICT), or a related subject at a foundational level.
- Secondary and vocational school teachers who teach computer programming.

#### Students

- Students who wish learn Java programming and build their Object Oriented Programming experience using Java.
- This course is a suitable foundational class for computer science majors, and when taught in sequence with Java Programming may be used to prepare students for the AP Computer Science A exam.

### Prerequisites

#### Required

- Oracle Academy Workshop - Getting Started with Java Using Alice
- Oracle Academy Workshop - Creating Java Programs with Greenfoot

#### Suggested

- Oracle Academy Course - Java Fundamentals

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## Contenido del curso

### Section 1 – Introduction

- 1.1 About the Course
- 1.2 A Brief History
- 1.3 Setting up Java

### Section 2 – Java Software Development

- 2.1 The Software Development Process
- 2.2 What is my Program Doing?
- 2.3 Introduction to Object-Oriented Programming Concepts

### Section 3 – Java Data Types

- 3.1 What is a Variable?
- 3.2 Numeric Data
- 3.3 Textual Data
- 3.4 Converting Between Data Types
- 3.5 Keyboard Input

# Curso Java Foundations

ORACLE ACADEMY

[www.oracle.com/academy](http://www.oracle.com/academy)

## Java Foundations – Course Objectives

### Overview

This course of study engages students with little programming experience. Students are introduced to object-oriented concepts, terminology, and syntax, and the steps required to create basic Java programs using hands-on, engaging activities. Students will learn the concepts of Java programming, design object-oriented applications with Java and create Java programs using hands-on, engaging activities.

### Duration

- Recommended total course time: 90 hours\*
- Professional education credit hours for educators who complete Oracle Academy training: 30

\* total course time includes instruction, self-study/homework, practices, projects and assessment

### Target Audiences

#### Educators

- Technical, vocational, and 2- and 4-year college and university faculty members who teach computer programming, information communications technology (ICT), or a related subject at a foundational level.
- Secondary and vocational school teachers who teach computer programming.

#### Students

- Students who wish learn Java programming and build their Object Oriented Programming experience using Java.
- This course is a suitable foundational class for computer science majors, and when taught in sequence with Java Programming may be used to prepare students for the AP Computer Science A exam.

### Prerequisites

#### Required

- Oracle Academy Workshop - Getting Started with Java Using Alice
- Oracle Academy Workshop - Creating Java Programs with Greenfoot

#### Suggested

- Oracle Academy Course - Java Fundamentals

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## Contenido del curso

### Section 4 – Java Methods and Library Classes

- 4.1 What is a Method?
- 4.2 The import Declaration and Packages
- 4.3 The String Class
- 4.4 The Random Class
- 4.5 The Math Class

### Section 5 – Decision Statements

- 5.1 Boolean Expressions and if/else Constructs
- 5.2 Understanding Conditional Execution
- 5.3 switch Statement

### Section 6 – Loop Constructs

- 6.1 for Loops
- 6.2 while and do-while Loops
- 6.3 Using break and continue Statements

# Curso Java Foundations

ORACLE ACADEMY

[www.oracle.com/academy](http://www.oracle.com/academy)

## Java Foundations – Course Objectives

### Overview

This course of study engages students with little programming experience. Students are introduced to object-oriented concepts, terminology, and syntax, and the steps required to create basic Java programs using hands-on, engaging activities. Students will learn the concepts of Java programming, design object-oriented applications with Java and create Java programs using hands-on, engaging activities.

### Duration

- Recommended total course time: 90 hours\*
- Professional education credit hours for educators who complete Oracle Academy training: 30

\* total course time includes instruction, self-study/homework, practices, projects and assessment

### Target Audiences

#### Educators

- Technical, vocational, and 2- and 4-year college and university faculty members who teach computer programming, information communications technology (ICT), or a related subject at a foundational level.
- Secondary and vocational school teachers who teach computer programming.

#### Students

- Students who wish learn Java programming and build their Object Oriented Programming experience using Java.
- This course is a suitable foundational class for computer science majors, and when taught in sequence with Java Programming may be used to prepare students for the AP Computer Science A exam.

### Prerequisites

#### Required

- Oracle Academy Workshop - Getting Started with Java Using Alice
- Oracle Academy Workshop - Creating Java Programs with Greenfoot

#### Suggested

- Oracle Academy Course - Java Fundamentals

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## Contenido del curso

### Section 7 – Creating Classes

- 7.1 Creating a Class
- 7.2 Instantiating Objects
- 7.3 Constructors
- 7.4 Overloading Methods
- 7.5 Object Interaction and Encapsulation
- 7.6 Static Variables and Methods

### Section 8 – Arrays and Exceptions

- 8.1 One-dimensional Arrays
- 8.2 ArrayLists
- 8.3 Exception Handling
- 8.4 Debugging Concepts and Techniques

### Section 9 – JavaFX

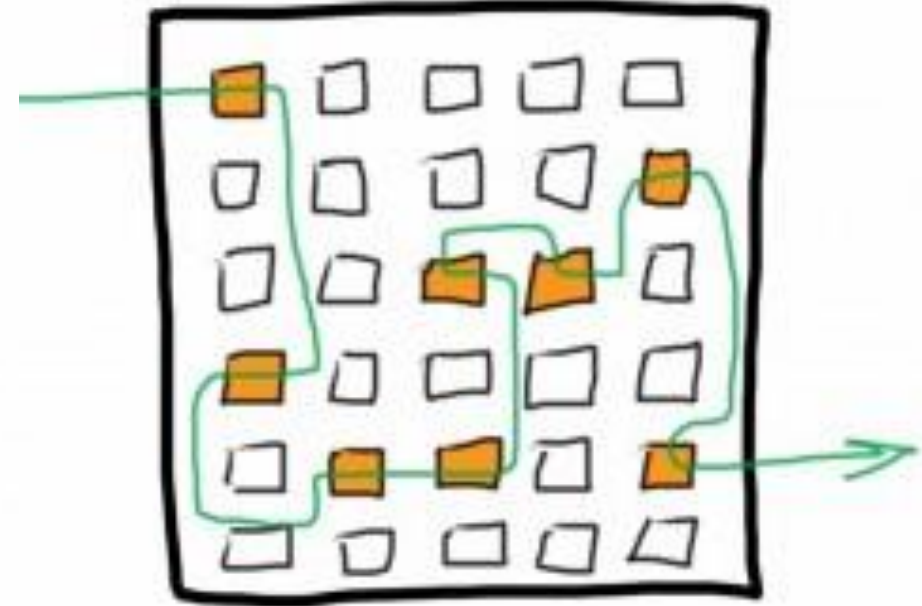
- 9.1 Introduction to Java FX
- 9.2 Colors and Shapes
- 9.3 Graphics, Audio and MouseEvents

# Curso Java Foundations

Finding our way  
through bad code



Finding our way  
through clean code

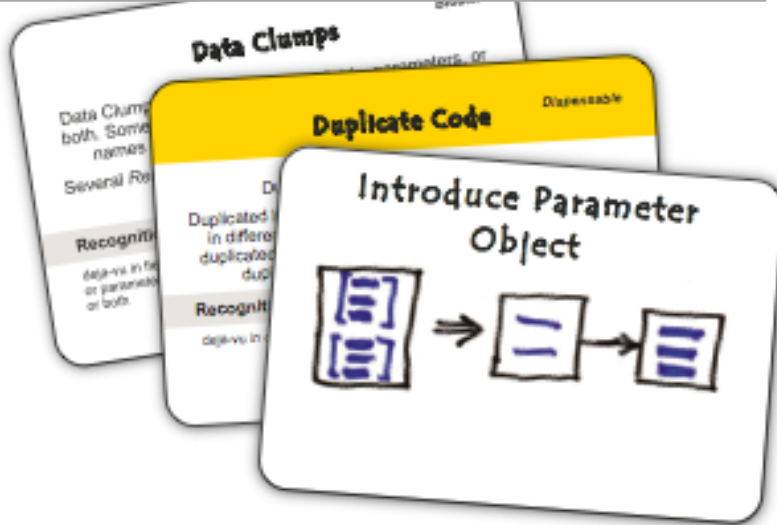


“Any fool can write code that a computer can understand.  
Good programmers write code that humans can understand.”

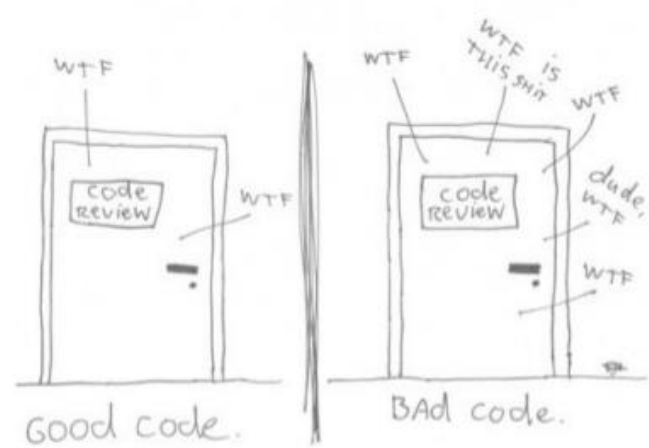
Martin Fowler, 2008

#PUEDAY16

# Curso Java Foundations



The ONLY valid measurement OF code QUALITY: WTFs/MINUTE



# Curso Java Foundations

## *“La Regla del Boy Scout”*

“Always leave the campground cleaner than you found it.”

“Try and leave this world a little better than you found it.”





# Curso Java Foundations

## OCA: Java Foundations

*(en exclusiva para centros Oracle Academy)*

**ORACLE®**

**Certified Associate**

Java Foundations Certified  
Junior Associate

Examen 1Z0-811:  
[Java Foundations](#)

### 1Z0-811: Java Foundations

- Plataforma:	Java SE 8
- Duración:	150 minutos
- Núm. Preguntas:	75 preguntas
- Nota aprobado:	65%
- Precio:	<del>8€</del> → 61,5 €

### Contenidos

What is Java?  
Java Basics  
Basic Java Elements  
Working with Java Data Types  
Working with Java Operator  
Working with the String Class  
Working with the Random and Match Classes  
Using Decision Statements  
Using Looping Statements  
Debugging and Exception Handling  
Arrays and ArrayLists  
Classes and Constructors  
Java Methods

**25%**

de descuento  
en certificaciones oficiales de Oracle

# Curso Java Foundations

Given the code fragment:

```
String a = "Java";  
String b = new String("Java");  
System.out.println(a.equals(b));  
System.out.println(a==b);
```

What is the result?

- A) false  
false
- B) true  
true
- C) false  
true
- D) true  
false

**SAMPLE**

# Curso Java Foundations

## Oracle CertView







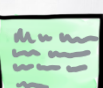

The screenshot shows the Oracle University CertView login page. At the top, there is a navigation bar with 'ORACLE UNIVERSITY' and links for 'Training Search', 'Training', 'Certification', and 'Community'. Below this is a 'Sign In' section with the Oracle University logo and the text 'Welcome to CertView, your web based portal for all Oracle Certification activity. Please sign in to view your account.' A 'Sign In To CertView' button is visible. Below the login section, there are links for 'Returning Users', 'First Time Users', and 'Create an Oracle Web Account'. The main content area is titled 'Returning CertView Users' and contains a list of instructions: 1. Click the Sign In To CertView button to access your account. 2. If you do not remember your Oracle Web Account user name, click here. If you do not remember your Oracle Web Account password, click here. 3. Common CertView questions and answers. A 'Sign In To CertView' button is also present at the bottom of this section.

The screenshot shows the Oracle University Certification History page for a user named Jordi Arino Santos (Oracle Testing ID: SR2889102). The page displays several certification logos for Java SE 7 Programmer and Java SE 6 Programmer. Below the logos is a table with the following data:

Credential/Certification	Date Achieved	Download Logo
Oracle Certified Associate, Java SE 7 Programmer	15-NOV-2012	<a href="#">BMD</a> <a href="#">QP</a> <a href="#">EES</a>
Sun Certified Web Component Developer for the Java Platform, EE 6	18-JUN-2010	<a href="#">BMD</a> <a href="#">QP</a> <a href="#">EES</a>
Sun Certified Programmer for the Java Platform, SE 6	10-DEC-2008	<a href="#">BMD</a> <a href="#">QP</a> <a href="#">EES</a>
Sun Certified Associate for Java Platform, SE	26-APR-2007	<a href="#">BMD</a> <a href="#">QP</a> <a href="#">EES</a>
Sun Certified Programmer for the Java Platform, SE 5.0	28-FEB-2007	<a href="#">BMD</a> <a href="#">QP</a> <a href="#">EES</a>

Below the table, there is a section titled 'Why does my certification not show up here?' with a list of reasons: 1. You have not allowed 48 hours for your certification to be processed. 2. You have not completed all of the requirements for the certification exam track. 3. Your exams were taken under different Oracle Testing IDs. 4. You have not received a printed copy of my certificate.

<https://certview.oracle.com>

	To do	Doing	Done
	 		
	 		
	 		
	 		

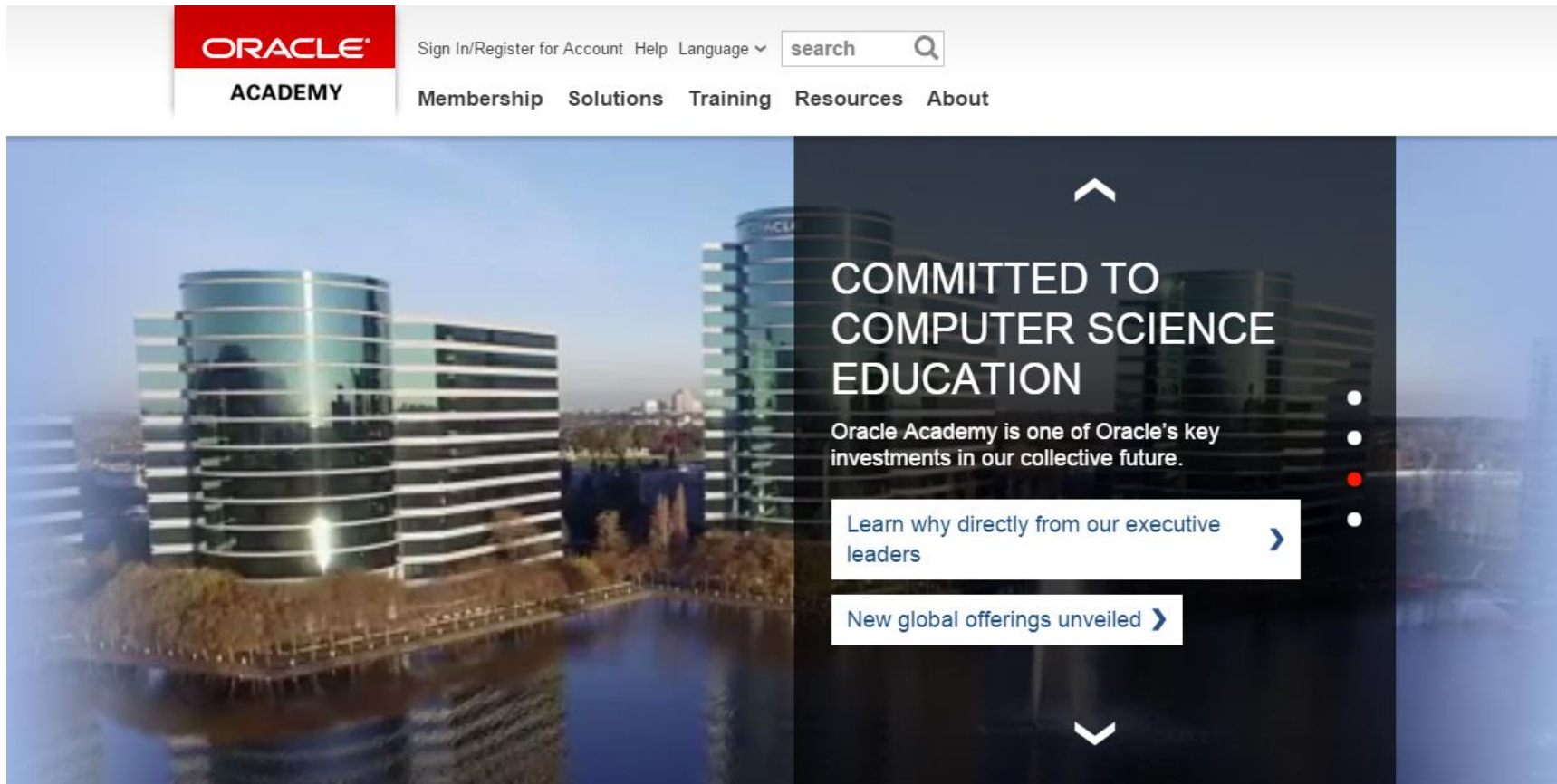
**Kanban Board  
(Visual Management)**

# Agenda

- 1.- Introducción
- 2.- Plan de formación y certificación en Java
- 3.- Curso Java Foundations
- 4.- Estructura del curso Java Foundations**
- 5.- Demo: Curso Java Foundations
- 6.- Oracle iLearning: Informes y diplomas

# Estructura del curso Java Foundations

Nueva plataforma Oracle Academy 3.0



<https://academy.oracle.com>

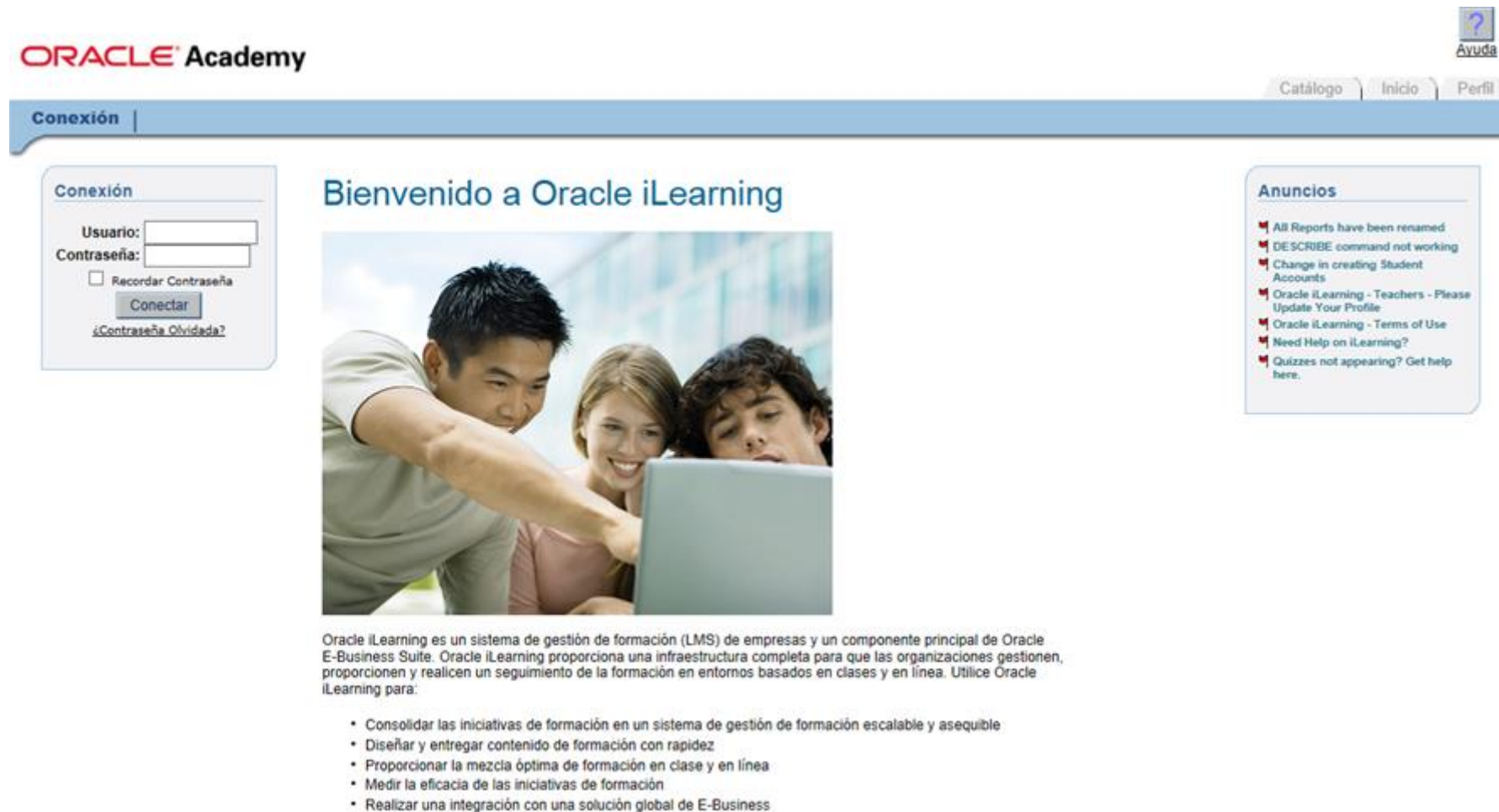
# Estructura del curso Java Foundations

Nueva plataforma Oracle Academy 3.0

The screenshot shows the Oracle Academy 3.0 homepage. At the top, there is a red navigation bar with the Oracle Academy logo on the left and user options (Welcome Diego, Account, Help, Log Out) and a search bar on the right. Below the navigation bar is a dark grey sidebar menu with options: Institution Member, Home, Welcome, Membership, Curriculum, Training, Resources, and Discounts. The main content area features a grid of seven tiles: Welcome, Membership, Survey, Curriculum, Training, Resources, and Discounts. At the bottom of the page, there is a footer with the text 'Integrated Cloud Applications & Platform Services', a list of links (Contact Us, Legal Notices, Terms of Use, Privacy, About Oracle), and social media icons for Facebook, Twitter, YouTube, and RSS.

<https://academy.oracle.com>

# Estructura del curso Java Foundations



**ORACLE Academy**

Conexión | [Catálogo](#) | [Inicio](#) | [Perfil](#)

**Conexión**


Usuario:

Contraseña:

Recordar Contraseña

[¿Contraseña Olvidada?](#)

## Bienvenido a Oracle iLearning



Oracle iLearning es un sistema de gestión de formación (LMS) de empresas y un componente principal de Oracle E-Business Suite. Oracle iLearning proporciona una infraestructura completa para que las organizaciones gestionen, proporcionen y realicen un seguimiento de la formación en entornos basados en clases y en línea. Utilice Oracle iLearning para:

- Consolidar las iniciativas de formación en un sistema de gestión de formación escalable y asequible
- Diseñar y entregar contenido de formación con rapidez
- Proporcionar la mezcla óptima de formación en clase y en línea
- Medir la eficacia de las iniciativas de formación
- Realizar una integración con una solución global de E-Business

**Anuncios**

- All Reports have been renamed
- DESCRIBE command not working
- Change in creating Student Accounts
- Oracle iLearning - Teachers - Please Update Your Profile
- Oracle iLearning - Terms of Use
- Need Help on iLearning?
- Quizzes not appearing? Get help here.

<http://ilearning.oracle.com/ilearn/en/learner/jsp/login.jsp?site=OracleAcad>

# Estructura del curso Java Foundations

Session Length 45 Minutes

	Session1	Session2	Session3	Session4	Session5
Week 1	Introduction				
Week 2	Java Software Development			Section 2 Quiz (15 preguntas)	
Week 3	Java Data Types			Section 3 Quiz A1-2 (15 preguntas)	
Week 4				Section 3 Quiz A3-5 (15 preguntas)	
Week 5	Java Methods and Library Classes			Section 4 Quiz A1-2 (15 preguntas)	
Week 6				Section 4 Quiz A3-5 (15 preguntas)	
Week 7	Decision Statements			Section 5 Quiz (15 preguntas)	
Week 8	Mid Term Exam Review			Mid Term Exam	
Week 9	Loop Constructs			Section 6 Quiz (15 preguntas)	
Week 10	Creating Classes			Section 7 Quiz A1-3 (15 preguntas)	
Week 11				Section 7 Quiz A4-6 (15 preguntas)	
Week 12	Arrays and Exceptions			Section 8 Quiz (15 preguntas)	
Week 13					
Week 14	Java FX			Section 9 Quiz (15 preguntas)	
Week 15					
Week 16	Final Exam Review			Final Exam	
Week 17	Final Project				
Week 18					

50 preguntas -> 3 intentos

50 preguntas -> 3 intentos



# Estructura del curso Java Foundations

**Estructura del curso**  
Panel de navegación  
por secciones y test de  
evaluación o quizzes

Java Foundations - Teacher - English

Estructura del Curso

- Java Foundations - Teacher - English
  - Curriculum
    - Section 0 - Course Resources - Teacher**
      - Section 1 - Introduction - Teacher
      - Section 2 - Java Software Development - Teacher
      - Section 3 - Java Data Types - Teacher
      - Section 4 - Java Methods and Classes - Teacher
      - Section 5 - Conditional Programming - Teacher
      - Section 6 - Loops - Teacher
      - Section 7 - Create a Java Class - Teacher
      - Section 8 - Arrays and Exceptions - Teacher
      - Section 9 - Java FX - Teacher
    - Quizzes and Exams
      - Section 2 Quiz
      - Section 3 Quizzes
      - Section 4 Quizzes
      - Section 5 Quiz
      - Java Foundations Midterm Exam
      - Section 6 Quiz
      - Section 7 Quizzes
      - Section 8 Quiz
      - Section 9 Quiz
      - Java Foundations Final Exam

Estructura del Curso

**Navegación entre secciones**  
Permiten avanzar o retroceder  
de unas secciones a otras  
disponibles en el curso

ENG\_JFo\_S0\_T

## Section 0 - Course Resources

Java Foundations Course Resources

Java Foundations - Course Programming Project

PDF Zip File Downloads

Java Zip File Downloads

Course Icon Map - Teacher Course

Course Icon Map - Student Course

Printable Icon Map

Oracle iLearning Resources

**Zona principal de contenido**  
Muestra el contenido de la  
sección del curso seleccionada

**Navegación en una sección**  
Permiten avanzar o retroceder  
entre los diferentes apartados o  
temas de una sección

< PREV NEXT >

# Estructura del curso Java Foundations

## Java Foundations Course Resources



- PDF Resource  
Java Foundations Course Objectives
- PDF Resource  
Java Foundations Course Map
- PDF Resource  
Java Software Requirements and Installation

## Java Foundations – Course Objectives

**Overview**

This course of study engages students with little programming experience. Students are introduced to object-oriented concepts, terminology, and syntax, and the steps required to create basic Java programs using hands-on, engaging activities. Students will learn the concepts of Java programming, design object-oriented applications with Java and create Java programs using handouts, engaging activities.

**Duration**

- Recommended total course time: 40 hours\*
- Professional education credit hours for educators who complete Oracle Academy training: 30
- \*Total course time includes instruction, self-paced learning, practice, projects and assessment

**Target Audience**

**Educators**

- Technical, vocational, and 2- and 4-year college and university faculty members who teach computer programming, information communications technology (ICT), or a related subject at a postsecondary level.
- Secondary and vocational school teachers who teach computer programming.

**Students**

- Students who wish learn Java programming and build their Object Oriented Programming experience using Java.
- This course is a suitable foundational class to computer science majors, and when taught in response with Java Programming may be used to prepare students for the OI Computer Science exam.

**Prerequisites**

**Required**

- Oracle Academy Vendors - Creating Started with Java Using Java
- Oracle Academy Vendors - Creating Java Programs with GUIs

**Recommended**

- Oracle Academy Course - Java Fundamentals

## Guidelines for Java Foundations

Session Length 45 Minutes

	Session1	Session2	Session3	Session4	Session5
Week 1	Introduction				
Week 2	Java Software Development				
Week 3	Java Data Types				
Week 4	Java Methods and Library Classes				
Week 5	Decision Statements				
Week 6	Mid Term Exam Review		Mid Term Exam		
Week 7	Loop Constructs				
Week 8	Creating Classes				
Week 9	Arrays and Exceptions				
Week 10	Java FX				
Week 11	Final Exam Review				
Week 12	Final Project				

Copyright © 2015. Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## Required Software for Java Courses

**Required Software**

The following software is required for the Java Foundations course:

- Java 8
- GUIs
- Editor
- Debugger

The following software is required for the Java Programming course:

- Debugger

The following software is required for the Java Foundations course:

- Debugger

**Software Download and Installation Instructions**

Follow the instructions at the sites below to download and install the software required for your course.

Software	Download and Installation Instructions
Java 8	<a href="http://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html">http://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html</a>
GUIs	<a href="http://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html">http://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html</a>
Editor	<p>Installation Guide: <a href="https://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html">https://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html</a></p> <p>Editor IDE Software: <a href="https://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html">https://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html</a></p>
Debugger	<a href="https://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html">https://www.oracle.com/technetwork/java/javase-downloads-javase8-2133161.html</a>

Copyright © 2015. Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## Oracle iLearning Resources



- PDF Resource  
How to Run Reports
- PDF Resource  
How to Create and Manage Student Accounts

## Running Reports in Oracle Learning

There are two types of reports: instructors can access for their students and access reports.

- Admin Reports
- Manager Reports

**How to Run Admin Reports**

- Click the Admin button located at the bottom of any non-admin page.
- Click the Reports tab.
  - There are dozens of reports in the list. However, there are only two reports available for each course.
  - Individual user access history for individual students.
  - You should enter an ID for the report that you wish to run in the "Search Term" field, then click "Go".
  - Go to the last page of the document to find a list of report IDs for each course.
- Click the Run icon (magnifying glass) to run and view the report of your choice.

Each report requires specific permissions to be entered in the system. Most reports require your teacher organization name.

- The teacher organization name must be entered exactly the same as it appears in Oracle Learning. Do not include underscores when typing the teacher organization name.
- You can enter your teacher organization name in your student access history or at the left side of the screen after pressing the Admin button and selecting the course ID.

- Save report selecting the Save icon instead of the Run icon.
  - Select the template "Course Separated List - For Save".
  - Open the saved file using Microsoft Excel or Open Office.
  - When a report is saved from Oracle Learning it is saved with a file name extension.

Please note that a report can take several minutes to run as it is reporting on multiple students, multiple quizzes, and multiple exams.

Copyright © 2015. Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## How to Create and Manage Student Accounts in Oracle Learning

**How to Create Student Accounts in Oracle Learning**


- Log in to Oracle Learning.
- Click the Admin button (top right) to access your Admin privileges.
- At the top right, you will see four tabs: Content (default), Users, Comments, and Reports. Click the Users tab.
- Your current organization name will appear on the left of the screen. Click your school organization name to display the properties in the right-hand pane of the window.
- Click the Manage Users link at the top right.
- Click the Add User button to create new accounts.
- For the Oracle Academy Terms and Conditions, you are prohibited from using real names for student accounts. Complete the information required on each form to create a student account. The table below provides a guide of what information should be entered in each field.

Field	Note
First Name	"User" (first name, "user" will be the same for all users)
Last Name	1 or a number (including the student number in the account, 014 - 015, etc.)
Username (Edx States only)	The format for student accounts is: "Country Abbreviation - State Abbreviation - School Abbreviation - 01" (e.g., "US - CA - 01" depends on the number of students you are building accounts for (01-09999)). For Java Foundations or JP Java Programming, and a number of students are 01-09999. (The number indicates the student number for accounts.) Examples are: 1. user_ja_us_01 (This is student 1 for Java Foundations for Java: High School in Virginia, United States) 2. user_ja_us_02 (This is student 2) 3. user_ja_us_03 (This is Student 1 of Java Programming)
Username (All other courses)	The format for non-Edx student accounts is: "Country abbreviation - schoolname - School Abbreviation - 01" (e.g., "US - CA - 01" depends on the number of students you are building accounts for (01-09999)). (The number indicates the student number for the accounts.) Examples are:

Copyright © 2015. Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

# Estructura del curso Java Foundations

PDF Zip File Downloads



- Zip - ALL Lesson Slides - PDF Files
- Zip - ALL Student Guides - PDF Files
- Zip - ALL Practices - PDF Files

Java Foundations


3-2  
Numeric Data



Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Representing Strings in Java

- In Java, strings are objects of the class named `java.lang.String`.
- Example:  
`String s1 = "Hello, World";`



6

Declaring and Creating a String


You can instantiate strings in two ways:

- String literals: Directly assign a string literal to a string reference.  
`String s1 = "Find a needle in a haystack";`
- `new` operator: Similar to any other class. Not commonly used and not recommended.  
`String s2 = new String("Aaaa Bbbbbb");`

Although you can use the `new` operator to create a string, don't use it. You'll learn why later in the course.

20

Java Zip File Downloads



- Zip - Section 1 Java Files
- Zip - Section 2 Java Files
- Zip - Section 3 Java Files
- Zip - Section 4 Java Files
- Zip - Section 5 Java Files
- Zip - Section 6 Java Files
- Zip - Section 7 Java Files
- Zip - Section 8 Java Files
- Zip - Section 9 Java Files

- AccountTest2\_Soln.zip
- PrisonTest\_7\_2\_Ex1\_Soln.zip
- PrisonTest\_7\_3\_Ex2\_Soln.zip
- PrisonTest\_7\_4\_Ex3\_Soln.zip
- PrisonTest\_7\_5\_Ex4\_Soln.zip
- PrisonTest\_Student\_7\_4.zip

- AccountTest3\_Soln.zip
- PrisonTest\_7\_2\_Ex2\_Soln.zip
- PrisonTest\_7\_3\_Ex3\_Soln.zip
- PrisonTest\_7\_4\_Ex4\_Soln.zip
- PrisonTest\_7\_6\_Ex2\_Soln.zip
- PrisonTest\_Student\_7\_5.zip

- AccountTest4\_Soln.zip
- PrisonTest\_7\_2\_Ex3\_Soln.zip
- PrisonTest\_7\_3\_Ex4\_Soln.zip
- PrisonTest\_7\_5\_Ex1\_Soln.zip
- PrisonTest\_7\_6\_Ex3\_Soln.zip
- PrisonTest\_Student\_7\_6.zip

- JavaPuzzleBall.jar  
Executable Jar File  
7,80 MB
- PrisonTest\_7\_3\_Ex1\_Soln.zip
- PrisonTest\_7\_4\_Ex2\_Soln.zip
- PrisonTest\_7\_5\_Ex3\_Soln.zip
- PrisonTest\_Student\_7\_3.zip
- ProblemSet7\_Soln.zip

# Estructura del curso Java Foundations

**Java Foundations - Teacher - English** Estructura del Curso

ENG\_JFo\_S3\_T **Sección del curso**

**Section 3 - Java Data Types**

3-2 Numeric Data **Apartado dentro de la sección**

**ORACLE ACADEMY**

**Teacher Resources**  
Student Resources  
Chickens01.zip  
Chickens02.zip

**Acceso y descarga del contenido**  
Permite bajarse las slides del instructor y del alumno, así como acceder al contenido online del curso

*\*After you complete this lesson, take the "Section 3 Quiz 1" found in the Quizzes and Exams section of the course outline!*

1 2 3 4 5 6

**Navegación en una sección**  
Permiten avanzar o retroceder entre los diferentes apartados o temas de una sección

**< PREV Lesson** **NEXT Lesson >**

# Estructura del curso Java Foundations

Section 7 - Create a Java Class

7-3 Constructors

**ORACLE** ACADEMY

Materia instructor: versión web o pdf

Materia alumno: versión web o pdf

- Teacher Resources
- Student Resources
- PrisonTest\_Student\_7\_3.zip
- PrisonTest\_7\_3\_Ex1\_Soln.zip
- PrisonTest\_7\_3\_Ex2\_Soln.zip
- PrisonTest\_7\_3\_Ex3\_Soln.zip
- PrisonTest\_7\_3\_Ex4\_Soln.zip

Proyecto de base para realizar las distintas prácticas guiadas

Soluciones de instructor para cada una de las prácticas guiadas del capítulo o apartado

1 2 3 4 5 6 7

# Estructura del curso Java Foundations

Menú de navegación de un apartado de una sección del curso

The screenshot shows the Oracle Academy interface for the course 'JFo - S4L4 - The Random Class'. On the left is a navigation menu with 29 items, where item 11, 'Methods Provided by the...', is highlighted. The main content area on the right is titled 'Methods Provided by the Random Class' and contains an introductory paragraph and a table of methods. The footer includes the Oracle Academy logo, course ID 'JFo 4-4', and copyright information.

Method	Produces
<code>boolean nextBoolean();</code>	A true or false value
<code>int nextInt();</code>	An integral value between <code>Integer.MIN_VALUE</code> and <code>Integer.MAX_VALUE</code>
<code>long nextLong();</code>	A long integral value between <code>Long.MIN_VALUE</code> and <code>Long.MAX_VALUE</code>
<code>float nextFloat();</code>	A decimal number between 0.0 (included) and 1.0 (excluded)
<code>double nextDouble();</code>	A decimal number between 0.0 (included) and 1.0 (excluded)

Zona principal de contenido

# Estructura del curso Java Foundations

Objetivos

Sumario/Índice

Contenido didáctico

Ejercicios / Prácticas

Resoluciones ejercicios

Interactive Quizzes

Resumen de contenido

Resumen de la lección

## Summary

In this lesson, you should have learned how to:

- Understand the memory consequences of instantiating objects
- Understand object references
- Understand the difference between stack and heap memory
- Understand how `Strings` are special objects



# Estructura del curso Java Foundations

Match each item with a definition.

The process of turning source code (.java) into bytecode (.class).

Compiling

Readable by humans

Source code (.java)

Readable by the JRE

Bytecode (.class)

Interactive Quizzes



# Estructura del curso Java Foundations

What would you type in line 5 to make this program print "Hello Java!" ?

```
1  
2 public class HelloWorld {  
3  
4     public static void main(String[] args) {  
5         System.out.println("Hello Java!");  
6     }  
7 }  
8
```

Interactive Quizzes

# Estructura del curso Java Foundations

Decide which bubbles represent properties, and which represent behaviors of a `Friend` Object. Drag the bubble into the corresponding box.

## Property

name

age

## Behavior

go shopping

laugh

watch a movie

where we met

Interactive Quizzes

# Estructura del curso Java Foundations

Drag and drop the term on the right to match the description on the left.

A blueprint for creating objects

Class

Can be physical or conceptual, and have properties and behaviors

Objects

Characteristics that can describe an object

Properties

Actions and interactions an object is capable of

Behaviors

An individual object created from a blueprint

Instance

Interactive Quizzes

# Estructura del curso Java Foundations

What should be the return type for this method, which returns a calculated value?

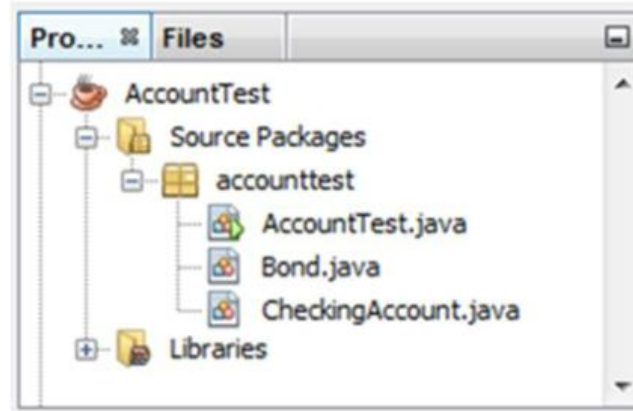
```
public ????? calcInterest() {  
    double interest = balance*interestRate/12;  
    return interest;  
}
```

- void
- int
- double
- String

Interactive Quizzes

# Estructura del curso Java Foundations

Which statements are true, based on the following project directory?



- It's impossible to add more classes to this project.
- Bond, and CheckingAccount are classes.
- AccountTest.java, Bond.java, and CheckingAccount.java are files that exist in the same folder.
- An instance of a `CheckingAccount` cannot be instantiated.

Interactive Quizzes

# Estructura del curso Java Foundations

Which two attempts at overloading the `printValues()` method cause a conflict?

```
public int printValues(String x, int y){  
    ...  
}
```

```
public void printValues(int i, String s){  
    ...  
}
```

```
public void printValues(int i){  
    ...  
}
```

```
public void printValues(String s, int i){  
    ...  
}
```

Interactive Quizzes

# Estructura del curso Java Foundations

Drag and drop each box to correctly comment the following code.

```
public class Prisoner{  
  
    private static int prisonerCount = 0;  
    private int bookingNumber;  
  
    public static void displayPrisonerCount () {  
        System.out.println(prisonerCount);  
    }  
    public void callAnotherMethod () {  
        displayPrisonerCount ();  
    }  
}
```

//Static Variable

//Static Method

//Instance Variable

//Instance Method


Interactive Quizzes


# Estructura del curso Java Foundations

ENG\_JFo\_S7\_T

## Section 7 - Create a Java Class

### Section 7 Practice



 Student Resources  
**ProblemSet7\_Soln.zip**

1 2 3 4 5 6 7

< PREV Lesson    NEXT Lesson >

ORACLE ACADEMY [www.oracle.com/academy](http://www.oracle.com/academy)

### Practices - Section 7: Date Night at the Arcade



**Overview**  
Tonight is date night at the arcade. After great evening of playing games and winning prizes, you and your date can't help wondering "How are these machines programmed?". You discuss possible designs on the subway back to campus. You enjoy the rest of the night romantically programming your ideas together.

You've made several observations about the arcade. A terminal is used to convert money into game credits. Credits are loaded onto plastic game cards. This data is stored in a card's magnetic strip. Cards may be swiped at any arcade game through the game's magnetic card reader. Games subtract credits from a card, but awards tickets. Tickets are also stored on a card's magnetic strip. Tickets may be exchanged for prizes at the terminal. The terminal is also used to check a card's credit balance and ticket count, and to transfer credits or tickets between cards.

**Tasks**  
Write a Java program that models the properties, behaviors, and interactions of objects at the arcade. You'll also need a test class that contains a main method. Use the main method to model actions that would drive the program such as object instantiations and card swipes. All fields must be `private`. Provide getter and any necessary setter methods.

**Cards**  
The magnetic strip on game cards offers limited storage space and zero computing power. Cards store information about their current credit balance, ticket balance, and card number. Neither balance should ever be negative. Individual cards are incapable of performing calculations, including simple addition, or realizing that their balances could go negative.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Práctica final de sección




# Estructura del curso Java Foundations

ENG\_JFo\_S9\_T

## Section 9 - Java FX

### Section 9 Practice



Student Resources  
CampusMap.jar  
ProblemSet9\_Soln.zip

1 2 3 4


< PREV Lesson    NEXT Lesson >

ORACLE ACADEMY [www.oracle.com/academy](http://www.oracle.com/academy)

## Practices - Section 9: Finding a Central Location

**Overview**  
Have you ever wondered where the most centrally located point on campus is? How about a centrally located point between you and a number of your friends? In this Problem Set, you'll write a JavaFX program that answers these questions visually.

**Tasks**  
Your goal is to create the CampusMap program that uses your map of campus, dorm names, dorm populations, and your group of friends. You're welcome to design your own campus map (this is your background graphic). You'll have to design your own campus map if your actual campus has fewer than 3 dorms, otherwise this Problem Set wouldn't be too interesting.



**The Dorms**  
Choose a way to visually represent dorms. The name and population of a dorm must also be visible. The population and location of each dorm must somehow be adjustable while the program is running.

**The Center Points**  
Your program must show two center points. The first point represents the central location of all students in all dorms. This is essentially a center of mass problem where dorms with a larger population are considered more "massive" and have a greater influence over the center point's location.

The second point represents the central location of your study group. Create a study group of at least 3 people, 1 of which must live in a different dorm.

Both center points must include a visual representation, a label, and display their location as numeric values. These points should automatically update as a dorm's location or population changes. You're welcome to leave these measurements as pixels or convert them into real-life units of distance.

However you choose to represent your dorms and points, remember to perform your distance calculations based on the geometric center of these visuals, and not the top-left corners.

**Hints:**  
There are certain concepts we didn't cover in Section 9, like how to work with a Text node. But we did discuss how to consult the JavaFX Ensemble. Part of the challenge of this Problem Set is understanding how to consult resources. If you have ideas about a feature you'd like to implement or a technique you'd like to explore, don't be afraid to consult the JavaFX Ensemble. It has a lot of fun things to show you.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Práctica final de sección

# Estructura del curso Java Foundations

**Java Foundations - Teacher - English** Estructura del Curso

**Estructura del Curso**

- Java Foundations - Teacher - English
  - Curriculum
    - Section 0 - Course Resources - Teacher
    - Section 1 - Introduction - Teacher
    - Section 2 - Java Software Development - Teacher
    - Section 3 - Java Data Types - Teacher
    - Section 4 - Java Methods and Classes - Teacher
    - Section 5 - Conditional Programming - Teacher
    - Section 6 - Loops - Teacher
    - Section 7 - Create a Java Class - Teacher
    - Section 8 - Arrays and Exceptions - Teacher
    - Section 9 - Java FX - Teacher
  - Quizzes and Exams
    - Section 2 Quiz
    - Section 3 Quizzes
    - Section 4 Quizzes
    - Section 5 Quiz
    - Java Foundations Midterm Exam
      - Java Foundations Midterm Exam**
      - Section 6 Quiz
      - Section 7 Quizzes
      - Section 8 Quiz
      - Section 9 Quiz
      - Java Foundations Final Exam

**Prueba: Java Foundations Midterm Exam**

Responda a las preguntas de esta página y haga clic en Siguiente para ir a la siguiente página de la prueba. Haga clic en Resumen para ver las preguntas que debe responder antes de enviar la prueba. Haga clic en Terminar Prueba si está preparado para enviarla.

**Section 3**  
(Responder todas las preguntas de esta sección)

16. The Scanner class accepts input in which form?

Marcar para Revisión (1) Puntos

- Future
- Tokens
- Callables
- Integer

17. You write a statement that assigns a value to a String variable as shown below.

```
String input = "This is Java Program";
```

This way of assigning values to variables is known as hard-coding.

Marcar para Revisión (1) Puntos

19. Which is valid syntax to declare and initialize a String variable?

Scanners cannot read text files.

A Scanner object opens a stream for collecting input.

A Scanner object doesn't have fields and methods.

Marcar para Revisión (1) Puntos

```
public class Welcome {  
    public static void main(String args[]) {  
        int a = 2;  
        System.out.println("a is" + a);  
    }  
}
```

2

int

a

Welcome

Marcar para Revisión (1) Puntos

## Java Foundations Midterm Exam

Examen tipo test de evaluación intermedio

Nº preguntas: 50

Nº intentos: 3

## Java Foundations Final Exam

Examen tipo test de evaluación final

Nº preguntas: 50

Nº intentos: 3

# Estructura del curso Java Foundations

## Prueba: Section 7 Quiz 1 - L1-L3

Revise las respuestas, los resultados y las puntuaciones de las preguntas que se muestran a continuación. Las respuestas correctas están marcadas con un asterisco (\*).

### Section 7 - Quiz 1 L1-L3

1. Class name should follow Camel casing rules.

- Verdadero (\*)
- Falso

Correct

2. The structure of a class consists of properties and behaviors.

- Verdadero (\*)
- Falso

Correct

3. First, you decide the radius of each circle in the logo. Then using the same radius you draw 5 circles of same size. All these circles will have properties like radius and color. All circles share behaviors to calculate circumference and area. Can you identify which of the following is an object?

- circle (\*)
- radius
- circumference
- fiveCircles

Correct

4. Which keyword is used to allocate memory for a newly created object?

- store
- new (\*)
- address
- memory

Correct

5. Which type of memory is allocated for the code below?

```
int x = 1;  
int y = 2;  
x=y;
```

- PileDriver memory
- Stack memory (\*)
- Heap memory
- No memory is allocated

Correct

Página 1 de 3







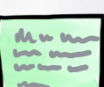

[Siguiete](#)

[Resumen](#)



# Agenda

- 1.- Introducción
- 2.- Plan de formación y certificación en Java
- 3.- Curso Java Foundations
- 4.- Estructura del curso Java Foundations
- 5.- Demo: Curso Java Foundations
- 6.- Oracle iLearning: Informes y diplomas

	To do	Doing	Done
			
			
			
			

Kanban Board  
(Visual Management)



# Demo: Curso Java Foundations

## Java Foundations

2-3

Introduction to Object-Oriented Programming Concepts



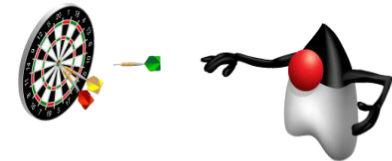
ORACLE ACADEMY

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

## Objectives

This lesson covers the following objectives:

- Differentiate between procedural and object-oriented programming
- Understand a **class** as a blueprint for an **object**
- Understand a class is used to create **instances** of an object
- Model objects as a combination of ...
  - **Properties** (data fields)
  - **Behaviors** (methods)



ORACLE ACADEMY

JFo 2-3  
Introduction to Object-Oriented Programming  
Concepts

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

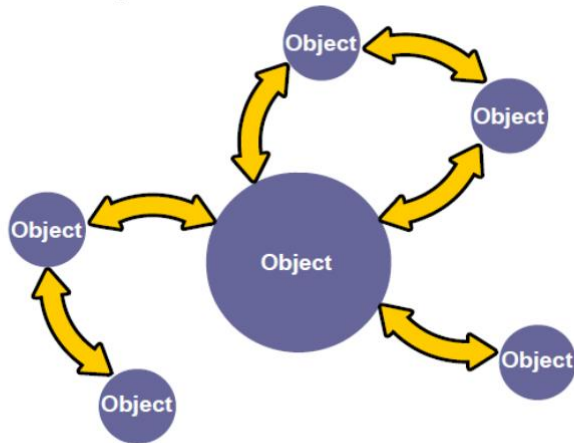
3

# Demo: Curso Java Foundations

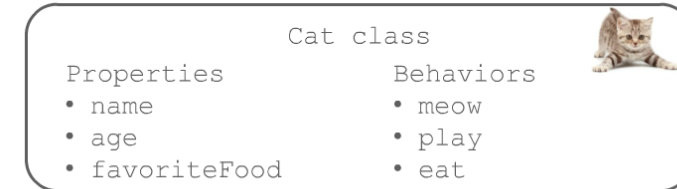


## Object-Oriented Programming

- Interaction of objects
- No prescribed sequence



## Creating New Instances from a Blueprint



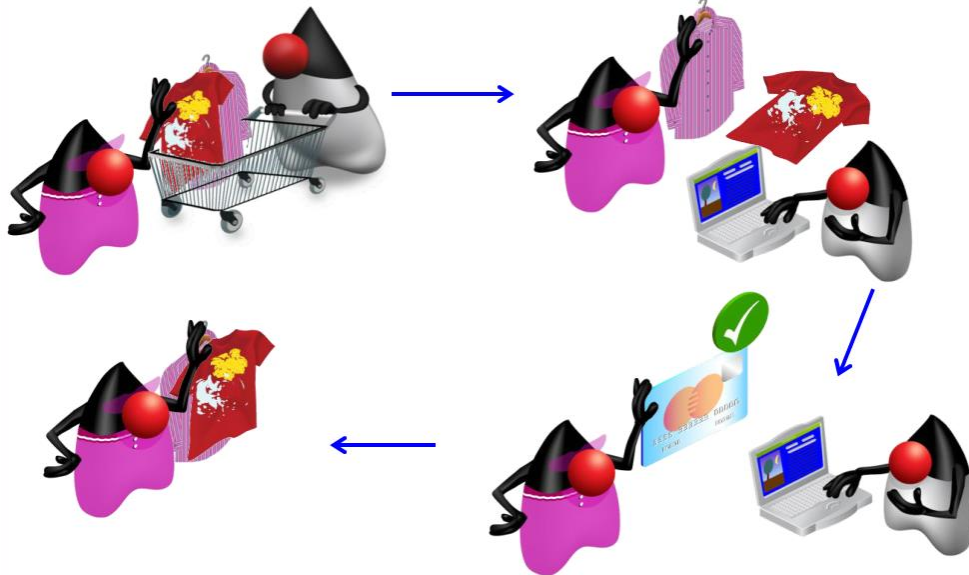
All cat instances share the ability to meow, play, and eat.



# Demo: Curso Java Foundations



## Duke's Choice Online Shopping

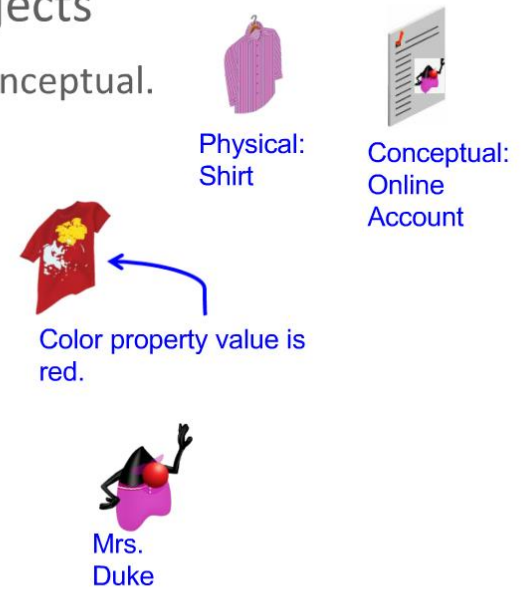


ORACLE ACADEMY

JFo 2-3 Introduction to Object-Oriented Programming Concepts Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 22

## Characteristics of Objects

- Objects are physical or conceptual.
- Objects have **properties**:
  - Size
  - Price
  - Color
- Objects have **behaviors**:
  - Shop
  - Put item in cart
  - Pay



ORACLE ACADEMY

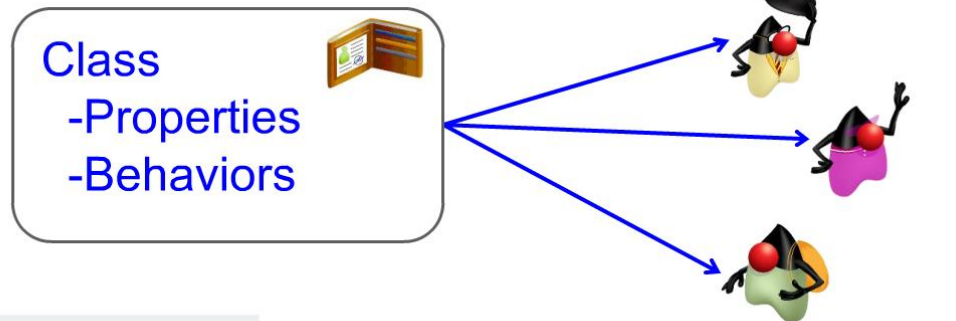
JFo 2-3 Introduction to Object-Oriented Programming Concepts Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 23



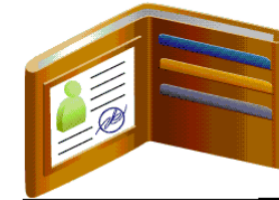
# Demo: Curso Java Foundations

## Classes and Instances

- Remember, a class ...
  - Is a blueprint or recipe for an object
  - Describes an object's properties and behaviors
  - Is used to create Object instances



## Customer Properties and Behaviors



- Properties:
  - Name
  - Address
  - Age
  - Order number
  - Customer number
- Behaviors:
  - Shop
  - Set address
  - Add item to cart
  - Ask for a discount
  - Display customer details



# Demo: Curso Java Foundations

## Translating into Java Syntax

```
1      Customer {  
2  
3  
4          Properties  
5  
6  
7  
8          Behaviors  
9  
10  
11 }
```

## Translating into Java Syntax

```
1 public class Customer {  
2     public String name = "Junior Duke";  
3     public int    custID = 1205;  
4     public String address;  
5     public int    orderNum;  
6     public int    age;  
7  
8     public void displayCustomer() {  
9         System.out.println("Customer: "+name);  
10    }  
11 }
```

# Demo: Curso Java Foundations

## Java Terminology

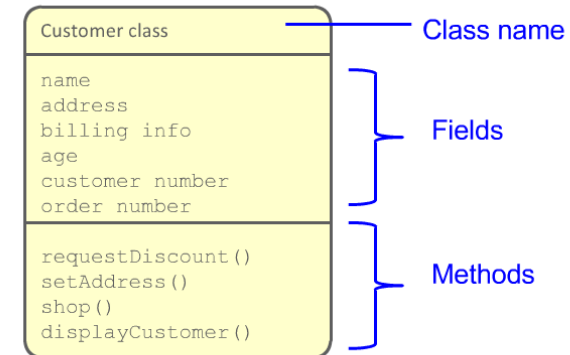
Class declaration

```
1 public class Customer {  
2     public String name = "Junior Duke";  
3     public int custID = 1205;  
4     public String address;  
5     public int orderNum;  
6     public int age;  
7  
8     public void displayCustomer() {  
9         System.out.println("Customer: "+name);  
10    }  
11 }
```

} Fields  
(Properties)  
(Attributes)

} Methods  
(Behaviors)

## Modeling Properties and Behaviors



# Demo: Curso Java Foundations

## Exercise 2, Part 1

Given the following scenario, what objects could you potentially model to complete your program?

Design a program for a coin-sorting machine. This machine should measure, count, and sort coins based on their size or value. It should also print a receipt.

- List at least 3 objects:



## Exercise 2, Part 2

- Chose an object from Part 1.
- What properties and behaviors of this object could you include in your program?
- Properties:
- Behaviors:

# Demo: Curso Java Foundations



Decide which bubbles represent properties, and which represent behaviors of a `Friend` Object. Drag the bubble into the corresponding box.

## Property

name

age

## Behavior

go shopping

laugh

watch a movie

where we met

Drag and drop the term on the right to match the description on the left.

A blueprint for creating objects

Class

Can be physical or conceptual, and have properties and behaviors

Objects

Characteristics that can describe an object

Properties

Actions and interactions an object is capable of

Behaviors

In individual object created from a blueprint

Instance

# Demo: Curso Java Foundations

## Java Foundations

7-1

Creating a Class



ORACLE ACADEMY

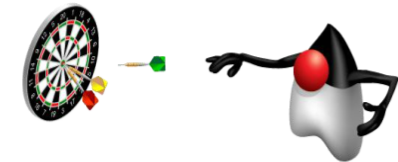
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

2

## Objectives

This lesson covers the following objectives:

- Create a Java test/main class
- Create a Java class in NetBeans
- Use conditionals in methods
- Translate specifications or a description into fields and behaviors



ORACLE ACADEMY

JFo 7-1  
Creating a Class

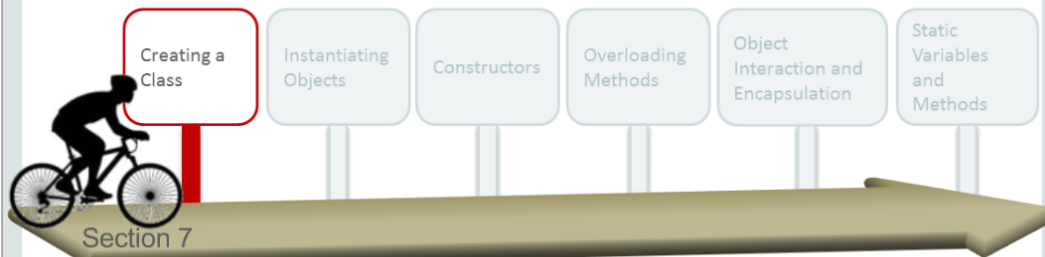
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

3

# Demo: Curso Java Foundations

## Topics

- Object-Oriented Thinking
- Object-Oriented Programming
- Scope of Variables
- Creating a Class from Specifications



ORACLE ACADEMY

JFo 7-1  
Creating a Class

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

4

## Section 7 - Create a Java Class

### 7-1 Creating a Class

ORACLE ACADEMY




Teacher Resources  
Student Resources  
JavaPuzzleBall.jar  
AccountTest2\_Soln.zip  
AccountTest3\_Soln.zip  
AccountTest4\_Soln.zip

1 2 3 4 5 6 7


# Demo: Curso Java Foundations

Section 7 - Create a Java Class

7-1 Creating a Class





Student Resources  
JavaPuzzleBall.jar



Exercise 1

- Play **Basic Puzzles 6 and 7**.
  - Your Goal: Design a solution that deflects the ball to Duke.
- Consider the following:
  - What happens when you put an icon on the blue wheel?



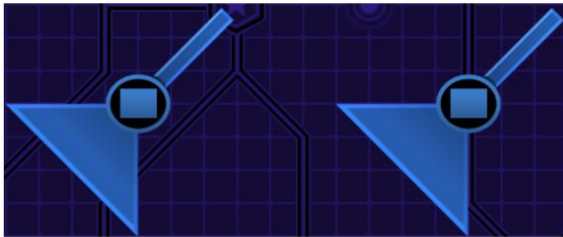
ORACLE ACADEMY

JFo 7-1  
Creating a Class

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 6

# Demo: Curso Java Foundations

## Describing a Blue Bumper



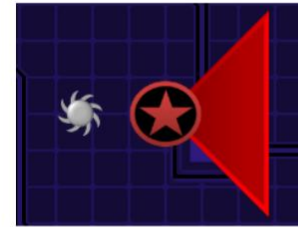
### (Fields)

- Properties:
  - Color
  - Shape
  - x-position
  - x-position

### (Methods)

- Behaviors:
  - Make ping sound
  - Flash
  - Deflect ball
  - Get destroyed

## What If the Ball Collides with a Bumper?



A method with the following logic is called:

```
public void onCollisionWithBall(Ball ball){  
    if(ball.isBlade == true){           //Ball is blade  
        getDestroyed();  
    }  
    else{                               //Ball is not blade  
        deflectBall();  
    }  
}
```



# Demo: Curso Java Foundations

## Modeling a Savings Account

- You could model one savings account like this:

```
public class SavingsAccount{  
  
    public static void main(String args[]){  
  
        int balance = 1000;  
        String name = "Damien";  
  
    }  
}
```

- And two accounts like this:

```
int balance1 = 1000;  
String name1 = "Damien";  
  
int balance2 = 2000;  
String name2 = "Bill";           //Copy, Paste, Rename
```

## How to Structure a Class

- Code should fit this format:

```
1 public class SavingsAccount {  
2  
3     Properties  
4  
5  
6     Behaviors  
7  
8  
9 }
```

# Demo: Curso Java Foundations

## How to Structure a Class

- Code should fit this format:

```
1 public class SavingsAccount {
2     public double balance;
3     public double interestRate = 0.01;
4     public String name;
5
6     public void displayCustomer() {
7         System.out.println("Customer: "+name);
8     }
9 }
```

- With one simple line of code(line 3), all 1000 accounts have an interest rate.
  - And we can change the rate at any time for any account.

## The Main Method as a Driver Class

- Place the main method in a test class.
  - The main method is often used for instantiation.

```
1 public class AccountTest {
2     public static void main(String[] args){
3
4         SavingsAccount sa0001 = new SavingsAccount();
5         sa0001.balance = 1000;
6         sa0001.name = "Damien";
7         sa0001.interestRate = 0.02;
8
9         SavingsAccount sa0002 = new SavingsAccount();
10        sa0001.balance = 2000;
11        sa0001.name = "Bill";
12 }
```

# Demo: Curso Java Foundations

## Variable Scope

- Fields are accessible anywhere in a class.
  - This includes within methods.

```
public class SavingsAccount {  
    public double balance;  
    public double interestRate;  
    public String name;  
  
    public void displayCustomer() {  
        System.out.println("Customer: "+name);  
        System.out.println("Balance: " +balance);  
        System.out.println("Rate: "    +rate);  
    }  
}
```

## Exercise 2



- Create a new Java project.
- Create an `AccountTest` class with a main method.
- Create a `CheckingAccount` class.
  - Include fields for `balance` and `name`.
- Instantiate a `CheckingAccount` object from the main method.
  - Assign values for this object's `balance` and `name` fields.

# Demo: Curso Java Foundations



## Variable Scope

- Variables created within a method cannot be accessed outside that method.
  - This includes methods parameters.

```
public class SavingsAccount {  
    public double balance;  
    public double interestRate;  
    public String name;  
}
```

```
public void deposit(int x){  
    balance += x;  
}
```

Scope of x

```
public void badMethod(){  
    System.out.println(x);  
}
```

Not scope of  
x

## Accessing Fields and Methods from Another Class

1. Create an instance.
2. Use the dot operator (.)

```
public class AccountTest {  
    public static void main(String[] args){  
  
        1) SavingsAccount sa0001 = new SavingsAccount();  
        2) { sa0001.name = "Damien";  
           sa0001.deposit(1000);  
        }  
}
```

```
public class SavingsAccount {  
    public String name;  
    public double balance;  
  
    public void deposit(int x){  
        balance += x;  
    }  
}
```



# Demo: Curso Java Foundations



## Passing Values to Methods

- 1000 is passed to the `deposit()` method.
- The value of `x` becomes 1000.

```
public class AccountTest {  
    public static void main(String[] args) {  
  
        SavingsAccount sa0001 = new SavingsAccount();  
        sa0001.name = "Damien";  
        sa0001.deposit(1000);  
    }  
}
```

```
public class SavingsAccount {  
    public String name;  
    public double balance;  
  
    public void deposit(int x) {  
        balance += x;  
    }  
}
```

*x = 1000*



## Exercise 3

- Continue editing the `AccountTest` project.
- Write a `withdraw()` method for checking accounts that ...
  - Accepts a double argument for the amount to be withdrawn.
  - Prints a warning if the balance is too low to make the withdrawal.
  - Prints a warning if the withdrawal argument is negative.
  - If there are no warnings, the withdrawal amount is subtracted from the balance. Print the new balance.
- Test this method with the instance from Exercise 2.



# Demo: Curso Java Foundations

## What if I Need a Value from a Method?

- Variables are restricted by their scope.
- But it's still possible to get the **value** of these variables out of a method.

```
public class SavingsAccount {
    public double balance;
    public double interestRate;
    public String name;

    public void calcInterest(){
        double interest = balance*interestRate/12; Scope of
                                                    interest
    }
}
```

## Returning Values from Methods

- If you want to get a value from a method ...
  - Write a return statement.
  - Change the method type from `void` to the type that you want returned.

```
public class SavingsAccount {
    public double balance;
    public double interestRate;
    public String name;

    //This method has a double return type
    public double calcInterest(){
        double interest = balance*interestRate/12;
        return interest;
    }
}
```

# Demo: Curso Java Foundations



## Returning Values: Example

- When `getInterest()` returns a value ...

```
public class AccountTest {  
    public static void main(String[] args) {  
        SavingsAccount sa0001 = new SavingsAccount();  
        sa0001.balance = 1000;  
        sa0001.balance += sa0001.calcInterest();  
    }  
}
```

- It's the equivalent of writing ...

```
public class AccountTest {  
    public static void main(String[] args) {  
        SavingsAccount sa0001 = new SavingsAccount();  
        sa0001.balance = 1000;  
        sa0001.balance += 0.83;  
    }  
}
```

– But it's better and more flexible because the value is calculated instead of hard-coded.

## Summary About Methods

```
public double calculate(int x, double y) {  
    double quotient = x/y;  
    return quotient;  
}
```

Method name: `calculate`  
Method return type: `double`  
Parameters: `int x, double y`  
Implementation: `{ ... }`



# Demo: Curso Java Foundations

## Limiting the Main Method

- The main method should be as small as possible.
- The example below isn't very good because ...
  - Increasing an account's balance based on interest is a typical behavior of accounts.
  - The code for this behavior should instead be written as a method within the `SavingsAccount` class.
  - It's also dangerous to have an account program where the balance field can be freely manipulated.

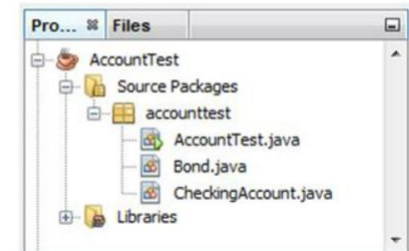
```
public class AccountTest {
    public static void main(String[] args) {
        SavingsAccount sa0001 = new SavingsAccount();
        sa0001.balance = 1000;
        sa0001.balance += sa0001.calcInterest();
    }
}
```

## Exercise 4

- Continue editing the `AccountTest` project.
- Create a new class according to the description. Be sure to instantiate this class and test its methods.

Create a Savings Bond. A person may purchase a bond for any term between 1 and 60 months. A bond earns interest every month until its term matures (0 months remaining). The term and interest rate are set at the same time. The bond's interest rate is based on its term according to the following tier system:

- 0–11 months: 0.5%
- 12–23 months: 1.0%
- 24–35 months: 1.5%
- 36–47 months: 2.0%
- 48–60 months: 2.5%





# Demo: Curso Java Foundations

## Describing a Savings Bond



- Properties:
  - Name
  - Balance
  - Term
  - Months Remaining
  - Interest Rate
- Behaviors:
  - Set Interest Rate Based on Term
  - Earn Interest
  - Mature (0 months remaining)

## Translating to Java Code: Part 1

- Your `Bond` class may have represented fields like this:

```
public class Bond{  
    public String name;  
    public double balance, rate;  
    public int term, monthsRemaining;  
    ...  
}
```

# Demo: Curso Java Foundations

## Translating to Java Code: Part 2

- And include the following methods:

```
public void setTermAndRate(int t){
    if(t>=0 && t<12)
        rate = 0.005;
    else if(t>=12 && t<24)
        rate = 0.010;
    else if(t>=24 && t<36)
        rate = 0.015;
    else if(t>=36 && t<48)
        rate = 0.020;
    else if(t>=48 && t<=60)
        rate = 0.025;
    else{
        System.out.println("Invalid Term");
        t = 0;
    }
    term = t;
    monthsRemaining = t;
}
```

## Translating to Java Code: Part 3

```
...
public void earnInterest(){
    if(monthsRemaining > 0){
        balance += balance*rate/12;
        monthsRemaining--;
        System.out.println("Balance: $" +balance);
        System.out.println("Rate: " +rate);
        System.out.println("Months Remaining:"
            +monthsRemaining);
    }
    else{
        System.out.println("Bond Matured");
    }
}
}
```

# Demo: Curso Java Foundations

Savings Bonds have several properties. How could these properties be translated and represented by Java code? Match each property with the best data type.

Owner's Name	Term	Rate
<b>String</b>	<b>int</b>	<b>double</b>
	Months Remaining	Balance

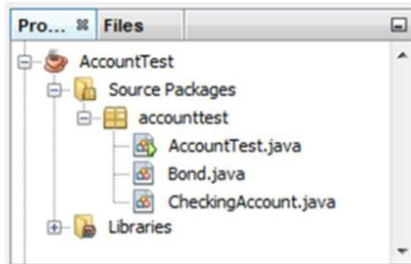
What should be the return type for this method, which returns a calculated value?

```
public ????? calcInterest(){  
    double interest = balance*interestRate/12;  
    return interest;  
}
```

- void
- int
- double
- String

# Demo: Curso Java Foundations

Which statements are true, based on the following project directory?



- It's impossible to add more classes to this project.
- Bond, and CheckingAccount are classes.
- AccountTest.java, Bond.java, and CheckingAccount.java are files that exist in the same folder.
- An instance of a `CheckingAccount` cannot be instantiated.

Which statements are true of the main method?

- The main method is stupid. Programs don't need a main method.
- The main method should be as short as possible.
- It's common to place the main method in a test class.
- The main method is one of many behaviors that an object may have.

# Demo: Curso Java Foundations

## Section7 Lesson1

Your Score: 100% (40 points)

Passing Score: 66% (26.4 points)

### Result:



Congratulations, you passed.

[Review Quiz](#)

# Demo: Curso Java Foundations



## Java Foundations

7-2  
Instantiating Objects



ORACLE ACADEMY

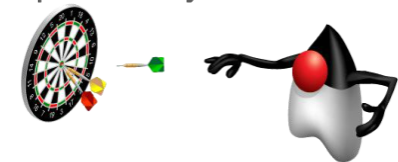
Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

2

## Objectives

This lesson covers the following objectives:

- Understand the memory consequences of instantiating objects
- Understand object references
- Understand the difference between stack and heap memory
- Understand how `Strings` are special objects



ORACLE ACADEMY

JFo 7-2  
Instantiating Objects

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

3



# Demo: Curso Java Foundations



Thank you for developing software for my bank! It would be an honor to shake your hand.

Thanks to you, our clients are opening more accounts than ever before.

And the children have never been happier!

Later that night ...

**CRASH! BANG! BANG!**

**ORACLE** ACADEMY JFo 7-2 Instantiating Objects Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 4

**BANG!**

Ha! Ha! Ha! Stealing is fun!

**ORACLE** ACADEMY JFo 7-2 Instantiating Objects Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 5

# Demo: Curso Java Foundations



**ORACLE** ACADEMY

JFo 7-2  
Instantiating Objects

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 6

## Describing a Prisoner

- Properties:
  - Name
  - Height
  - Years Sentenced
- Behaviors:
  - Think about what they've done

**ORACLE** ACADEMY

JFo 7-2  
Instantiating Objects

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. 8





# Demo: Curso Java Foundations



## Exercise 1, Part 1



- Create a new Java project.
- Create a `PrisonTest` class with a main method.
- Create a `Prisoner` class based on the description in the previous slide.
- Instantiate two prisoners and assign them the following properties:



Variable: bubba  
Name: Bubba  
Height: 6'10" (2.08m)  
Sentence: 4 years



Variable: twitch  
Name: Twitch  
Height: 5'8" (1.73m)  
Sentence: 3 years

## Exercise 1, Part 2



Can prisoners fool security by impersonating each other?

- Write a print statement with a boolean expression that tests if `bubba == twitch`.
- Change the properties of `twitch` so that they match `bubba`.
- Then test the equality of these objects again.



Variable: bubba  
Name: Bubba  
Height: 6'10" (2.08m)  
Sentence: 4 years



Variable: twitch  
Name: Bubba  
Height: 6'10" (2.08m)  
Sentence: 4 years



# Demo: Curso Java Foundations

## Programming the Prisoner Class

Your class may look something like this:

```
public class Prisoner {
    public String name;
    public double height;
    public int sentence;

    public void think(){
        System.out.println("I'll have my revenge.");
    }
}
```

## Prisoner Impersonation

- The boolean `bubba == twitch` is `false`.
  - Security wasn't fooled by prisoners who share the same properties.
  - Security understood that each prisoner was a unique object.
- How is this possible?

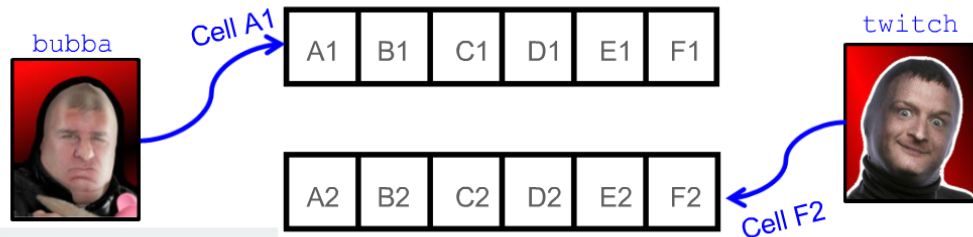
```
public class PrisonTest {
    public static void main(String[] args){
        Prisoner bubba = new Prisoner();
        Prisoner twitch = new Prisoner();
        ...
        System.out.println(bubba == twitch);    //false
    }
}
```

# Demo: Curso Java Foundations



## Prisoner Locations

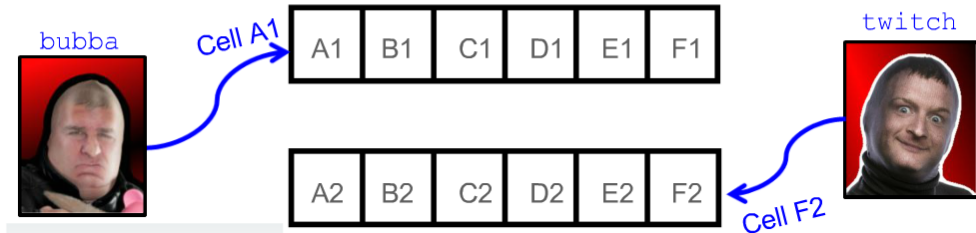
- Prisoners live in cells.
- New prisoners are assigned an available cell for living quarters.
- If a prisoner lives in a unique cell, he's a unique object.



## Prisoner Object Locations

- Cells are like locations in memory.
- Instantiating a `Prisoner` fills an available location in memory with the new `Prisoner` object.

```
public class PrisonTest {  
    public static void main(String[] args){  
        Prisoner bubba = new Prisoner();  
        Prisoner twitch = new Prisoner();  
    }  
}
```



# Demo: Curso Java Foundations



## The `new` Keyword


- The `new` keyword allocates available memory to store a newly created object.
- Java developers don't need to know an object's location in memory.
  - We only need to know the variable for the object.
  - But we can still print memory addresses.

```
public class PrisonTest {  
    public static void main(String[] args){  
        Prisoner bubba = new Prisoner();  
        Prisoner twitch = new Prisoner();  
        System.out.println(bubba); //prisontest.Prisoner@15db9742  
        System.out.println(twitch); //prisontest.Prisoner@6d06d69c  
    }  
}
```


Memory addresses

## Objects with the Same Properties

- Objects may share the same properties.
- But it doesn't mean that these objects are equal.
- As long as you use the `new` keyword during instantiation ...
  - You'll have unique objects.
  - Each object will have a different location in memory.



Variable: `bubba`  
Name: Bubba  
Height: 6'10" (2.08m)  
Sentence: 4 years  
Memory Address: `@15db9742`



Variable: `twitch`  
Name: Bubba  
Height: 6'10" (2.08m)  
Sentence: 4 years  
Memory Address: `@6d06d69c`



# Demo: Curso Java Foundations



## Comparing Objects

- If you compare two objects using the == operator ...
  - You're checking if their **memory addresses** are equal.
  - You're **not** checking if their fields are equal.
- The boolean `bubba == twitch` is **false** because ...
  - Memory addresses `@15db9742` and `@6d06d69c` are different.
  - It doesn't matter if `bubba` and `twitch` share the same properties.

```
public class PrisonTest {  
    public static void main(String[] args){  
        Prisoner bubba = new Prisoner();  
        Prisoner twitch = new Prisoner();  
        ...  
        System.out.println(bubba == twitch);    //false  
    }  
}
```

## Accessing Objects by Using a Reference



The camera is like the **object** that's accessed by using a reference.



The remote is like the **reference** that's used to access the camera.

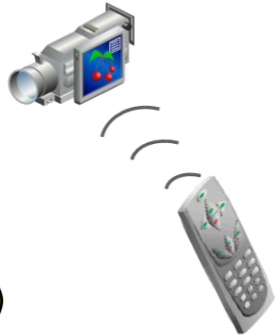


# Demo: Curso Java Foundations



## Working with Object References

- 1 Pick up remote to gain access to the camera.



- 2 Press remote controls to have the camera do something.

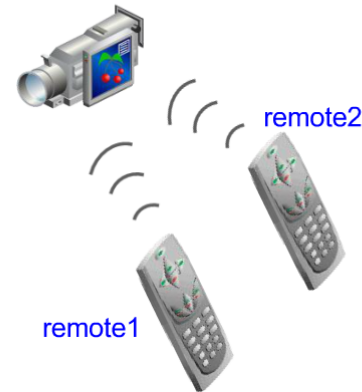
- 1 Create a Camera object and get a reference to it.

```
Camera remotel = new Camera();
```

- 2 Call a method to have the Camera object do something.

```
remotel.play();
```

## Working with Object References: Example 2



There's only one Camera object.

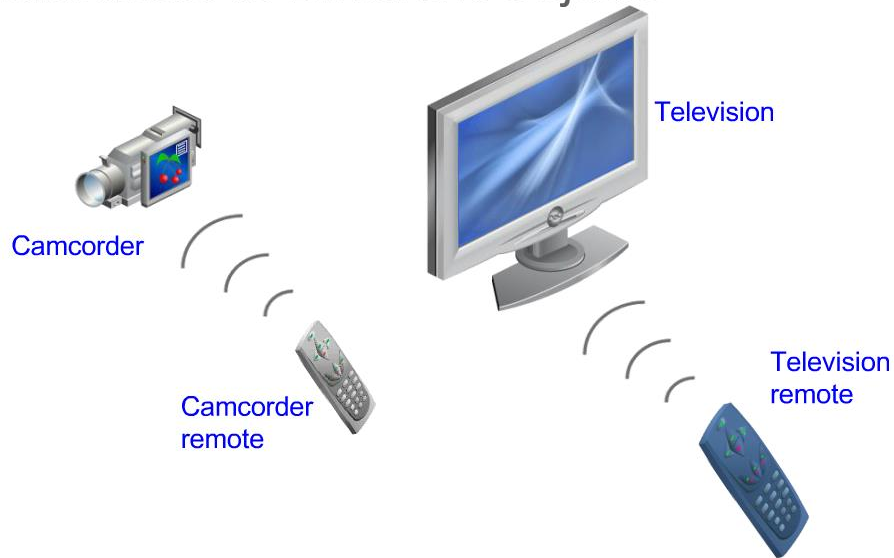
```
Camera remotel = new Camera();  
  
Camera remote2 = remotel;  
  
remotel.play();  
remote2.stop();
```



# Demo: Curso Java Foundations



## References to Different Objects



## References to Different Objects: Example

Reference type      Reference variable      Object type

```
Camera remote1 = new Camera ();  
remote1.menu ();  
  
TV remote2 = new TV ();  
remote2.menu ();  
  
Prisoner bubba = new Prisoner ();  
bubba.think ();
```



# Demo: Curso Java Foundations



## References to Different Objects: Example

- The following example isn't allowed because ...
  - The **Reference Type** doesn't match the **Object Type**.
  - A prisoner and a TV are completely different things.



```
Prisoner twitch = new TV();
```



## Exercise 2

- Continue experimenting with the `PrisonerTest` class.
- Is security fooled when reference variables change?
  - Instantiate two prisoners and assign them the properties below.
  - Test the equality of these objects.
  - Then set the reference variable for `bubba` equal to `twitch`.
  - Test the equality of these objects again.



Variable: bubba  
Name: Bubba  
Height: 6'10" (2.08m)  
Sentence: 4 years



Variable: twitch  
Name: Twitch  
Height: 5'8" (1.73m)  
Sentence: 3 years





# Demo: Curso Java Foundations



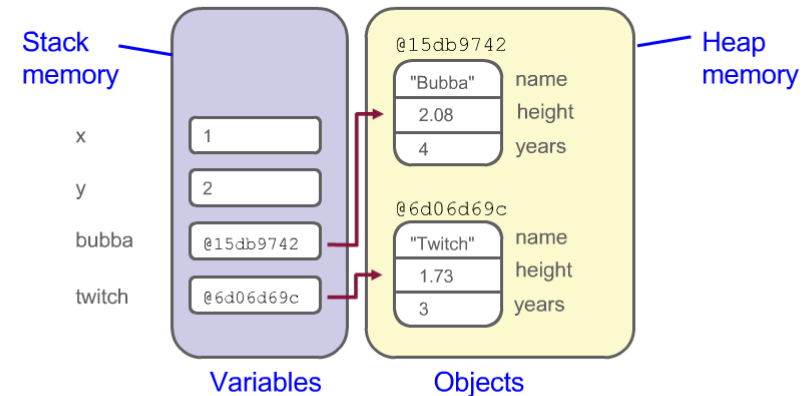
## Stack Memory and Heap Memory

Understanding the results of Exercise 2 requires an understanding of the types of memory that Java uses.

- **Stack memory** is used to store ...
  - Local variables
  - Primitives
  - References to locations in the heap memory
- **Heap memory** is used to store ...
  - Objects

## References and Objects in Memory

```
int x = 1;  
int y = 2;  
Prisoner bubba = new Prisoner();  
Prisoner twitch = new Prisoner();  
...
```

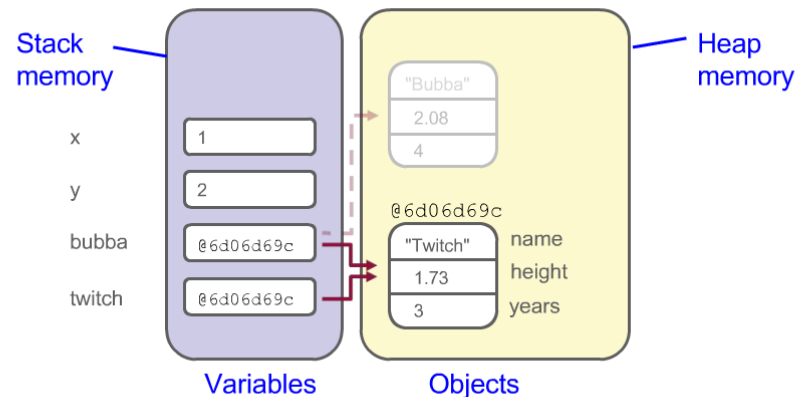


# Demo: Curso Java Foundations



## Assigning a Reference to Another Reference

```
bubba = twitch;
```



## Two References, One Object

- As of line 14, `bubba` and `twitch` reference the same object.
- Either reference variable could be used to access the same data.

```
11 Prisoner bubba = new Prisoner();
12 Prisoner twitch = new Prisoner();
13
14 bubba = twitch;
15
16 bubba.name = "Bubba";
17 twitch.name = "Twitch";
19
20 System.out.println(bubba.name); //Twitch
21 System.out.println(bubba == twitch); //true
```

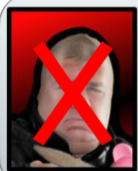


# Demo: Curso Java Foundations



## What Happened to Bubba?

- If no more reference variables point to an object ...
- Java **automatically** clears the memory once occupied by that object.
  - This is called **Garbage Collection**.
  - The data associated with this object is lost forever.



Variable:  
Name: Bubba  
Height: 6'10" (2.08m)  
Sentence: 4 years  
Memory Address:



Variables: twitch, bubba  
Name: Twitch  
Height: 5'8" (1.73m)  
Sentence: 3 years  
Memory Address: @6d06d69c

## Strings Are Special Objects

- Printing a `String` reference prints the actual `String` instead of the object's memory address.
- `Strings` can be instantiated with the `new` keyword.
  - But you shouldn't do this.
- `Strings` should be instantiated without `new`.
  - This is more memory-efficient.
  - We'll explore why in the next few slides.

```
String s1 = new String("Test");
```

```
String s2 = "Test";
```



# Demo: Curso Java Foundations

## Exercise 3



- Continue experimenting with the `PrisonTest` class.
- See the memory consequences of `Strings` for yourself.
  - Instantiate two prisoners with the names shown below.
  - Set their names by using the `new` keyword and test the equality of these `Strings` by using `==`.
  - Set their names without using the `new` keyword and test the equality of these `Strings` by using `==`.



Variable: bubba  
Name: **Bubba**  
Height: 6'10" (2.08m)  
Sentence: 4 years

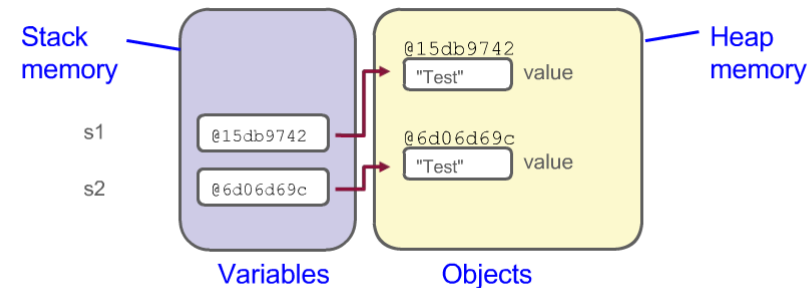


Variable: twitch  
Name: **Bubba**  
Height: 6'10" (2.08m)  
Sentence: 4 years

## Instantiating `Strings` with the `new` Keyword

Using the `new` keyword creates two different references to two different objects.

```
String s1 = new String("Test");  
String s2 = new String("Test");
```



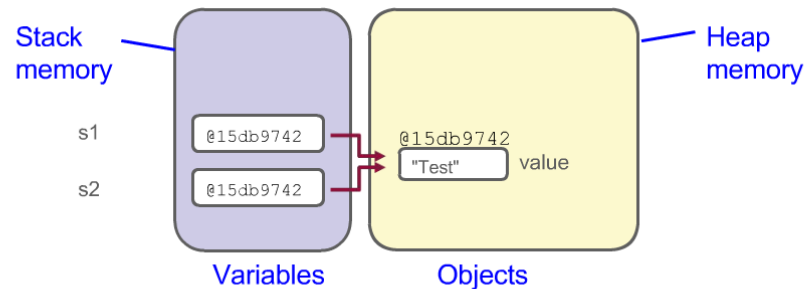
# Demo: Curso Java Foundations



## Instantiating Strings Without the `new` Keyword

- Java automatically recognizes identical `String`s and saves memory by storing the object only once.
- This creates two different references to one object.

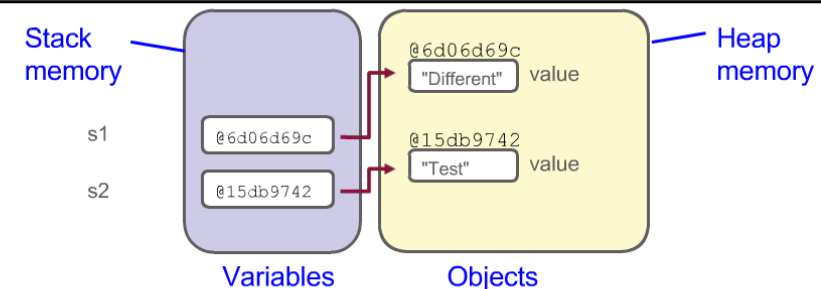
```
String s1 = "Test";  
String s2 = "Test";
```



## String References

- Altering a `String` using one reference won't affect other references.
- Java allocates new memory for a different `String`.

```
String s1 = "Test";  
String s2 = "Test";  
s1 = "Different";
```



# Demo: Curso Java Foundations



Where would each of the following be stored in memory? Drag each item to the correct box.

**Stack  
Memory**

**Heap  
Memory**

Objects

Local Variables

Strings

Object References

Primitives

What could potentially be the output of the following code?

```
12 Prisoner bubba = new Prisoner();
13 Prisoner twitch = bubba;
14
15 bubba.name = "Bubba";
16 twitch.name = "Twitch";
17
18 System.out.println(bubba.name);
```

- Bubba
- Twitch
- prisontest.Prisoner@6d06d69c
- name



# Demo: Curso Java Foundations



What could you write in line 18 to make **garbage collection** occur?

```
12 Prisoner bubba = new Prisoner();
13 Prisoner twitch = new Prisoner();
14
15 bubba.name = "Bubba";
16 twitch.name = "Twitch";
17
18
```







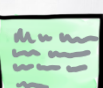

- bubba = twitch;
- bubba = new Prisoner();
- bubba.name = "Twitch";
- bubba == twitch

According to the following code, how many unique objects are created in the heap memory?

```
String s1 = "Test";
String s2 = "Test";
String s3 = new String("Test");
```

- 0
- 1
- 2
- 3



	To do	Doing	Done
			
			
			
			

Kanban Board  
(Visual Management)

# Agenda

- 1.- Introducción
- 2.- Plan de formación y certificación en Java
- 3.- Curso Java Foundations
- 4.- Estructura del curso Java Foundations
- 5.- Demo: Curso Java Foundations
- 6.- Oracle iLearning: Informes y diplomas



# Oracle iLearning: Informes y diplomas

**ORACLE** ACADEMY www.oracle.com/academy

---

### Running Reports in Oracle iLearning

There are two types of reports instructors can access to view student quiz and exam scores.

- Admin Reports
- Manager Reports

#### How to Run Admin Reports

- Click the **Admin** button located at the bottom or top of any non-admin page.
- Click the **Reports** tab.
  - There are dozens of reports in the list. However, there are only two reports available for each course.
    - Quiz and Exam Scores for the entire course
    - Individual quiz score history for individual students
  - You should enter an ID for the report that you wish to run in the "Search Text" textbox, then click "Go"
  - Go to the last page of this document to find a list of report IDs for each course
- Click the **Run** icon (eyeglasses) to run and view the report of your choice.
- Each report requires specific parameters to be entered before execution. Most reports request your teacher organization name.
  - The teacher organization name must be entered exactly the same as it appears in Oracle iLearning. Do not include underscores when typing the teacher organization name.
  - You can view your teacher organization name in your student account listing or at the left side of the screen after pressing the Admin button and selecting the Users tab.
- Save report by selecting the **Save** icon instead of the Run icon.
  - Select the template "Comma Separated List - For Save".
  - Open the saved file using Microsoft Excel or Open Office.
  - When a report is saved from Oracle iLearning it is saved with a txt filename extension.

Please note that a report can take several minutes to run as it is reporting on multiple students, multiple quizzes, and multiple exams.

Copyright © 2015, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

## ORACLE Academy

Cerrar

### 13A - JFo English

2016 Course - Java Foundations English - Quiz and Exam Scores - All Students. Enter the teacher organization name exactly as it appears in Oracle iLearning.

#### Plantilla

Seleccionar Plantilla

#### Parámetros

Introduzca los valores de los siguientes parámetros:

PLEASE\_ENTER\_THE\_TEACHER\_ORGANIZATION\_NAME

Mostrar SQL

Ejecutar

Cerrar

### 13A - JFo English

Username	S2	S3_1	S3_2	S4_1	S4_2	S5	S6	S7_1	S7_2	S8	S9	Completed_Quizzes	Avg_Quiz_Score	Midterm	Final
es_pue_jfo01_s008	100	-	-	-	-	-	-	-	-	-	-	1	100.00	-	-
es_pue_jfo01_s071	93.3	-	-	-	-	-	-	-	-	-	-	1	93.30	-	-
jordiA52K	93.3	100	93.3	-	-	-	-	100	100	-	-	5	97.32	24	-

Report Type

Quiz and Exam Scores  
All Students

Individual Student,  
Individual Quiz

Report Number

1717763372

1717766196

# Oracle iLearning: Informes y diplomas

ORACLE ACADEMY

Bienvenido Luis Cuenta Ayuda Cerrar sesión Español Buscar sitio de Oracle Acad

Página de inicio de Oracle Academy

Plan de estudios / Certificados para estudiantes

### Certificados de finalización para estudiantes

Tenemos el agrado de ofrecerle certificados de finalización de curso de Oracle Academy. Los certificados de finalización de curso de examen final se deben imprimir en dos copias.

Para crear los certificados:

- Descargue el archivo zip correspondiente.
- Lea las instrucciones de otorgamiento.
- Imprima los certificados en el papel recomendado.

Nombre	Tamaño	Comprimido	Tipo	Modificado	CRC32
Carpeta de archivos					
Oracle Academy Award Instructions.doc	188.928	169.468	Documento de Mi...	07/12/2012 15:00	6950693D
Oracle Academy Course Completion Award - Java Fundamentals.doc	41.472	22.599	Documento de Mi...	28/08/2012 17:09	5583BD03
source.xls	15.872	3.635	Hoja de cálculo de...	07/12/2012 15:03	D2C2A502

Total 246.272 bytes en 3 ficheros

Curso	Certificado de finalización de curso	Certificado de examen final
Database Design and Programming with PL/SQL	<a href="#">Descargar</a>	<a href="#">Descargar</a>
Database Design and Programming with SQL	<a href="#">Descargar</a>	<a href="#">Descargar</a>
Java Fundamentals	<a href="#">Descargar</a>	<a href="#">Descargar</a>
Java Programming	<a href="#">Descargar</a>	<a href="#">Descargar</a>

<https://academy.oracle.com>

# Oracle iLearning: Informes y diplomas

## Certificado de finalización del curso

Este equipo > OS (C:) > Usuarios > Jordi > Descargas > OA Course Complet

Nombre	Fec
Oracle Academy Award Instructions	07/
Oracle Academy Course Completion Award - Java Fundamentals	28/
source	07/

*Instrucciones* (pointing to Oracle Academy Award Instructions)  
*Listado de alumnos* (pointing to source)  
*Plantilla de diploma* (pointing to Oracle Academy Course Completion Award - Java Fundamentals)

## Certificado de superación del examen final

Este equipo > OS (C:) > Usuarios > Jordi > Descargas > OA Final Exam Awa

Nombre	Fecha
Oracle Academy Award Instructions	07/12/
Oracle Academy Final Exam Award - Java Fundamentals	28/08/
source	07/12/2012 14:03

*Instrucciones* (pointing to Oracle Academy Award Instructions)  
*Listado de alumnos* (pointing to source)  
*Plantilla de diploma* (pointing to Oracle Academy Final Exam Award - Java Fundamentals)

**ORACLE** ACADEMY

### Award Instructions

**To print your awards:**

1. Open the Excel file called source.xls.
2. Enter the student names in the Name column and the award date in the Date column.
3. Save the Excel document and close it.
4. Open the award Word document and select "mail merge" from the tools menu.
5. Click "get data" and select "open data source."
6. The "open data source" window will pop up. Locate your source Excel file containing the student names.
7. A window will pop up. Click OK for "entire spreadsheet."
8. Click the "merge" button.
9. A merge window pops up. Click "merge." This will create a new document with all of the awards.
10. Print this document.

Thank you for helping to make this program a success.

Sincerely,  
Oracle Academy

07/12/2012 14:03 Hoja de cálculo de Mi... 16 KB

# Oracle iLearning: Informes y diplomas

## Certificado de finalización del curso

**ORACLE** ACADEMY

AWARD *of* COURSE COMPLETION

Java Foundations

PRESENTED TO

«Name»

FOR SATISFACTORY COMPLETION OF ALL  
COURSEWORK AND TRAINING

«Date»

\_\_\_\_\_  
Oracle Academy Instructor

## Certificado de superación del examen final

**ORACLE** ACADEMY

AWARD *of* ACHIEVEMENT

PRESENTED TO

«Name»

FOR SUCCESSFULLY COMPLETING THE ORACLE ACADEMY

Java Foundations

FINAL EXAM

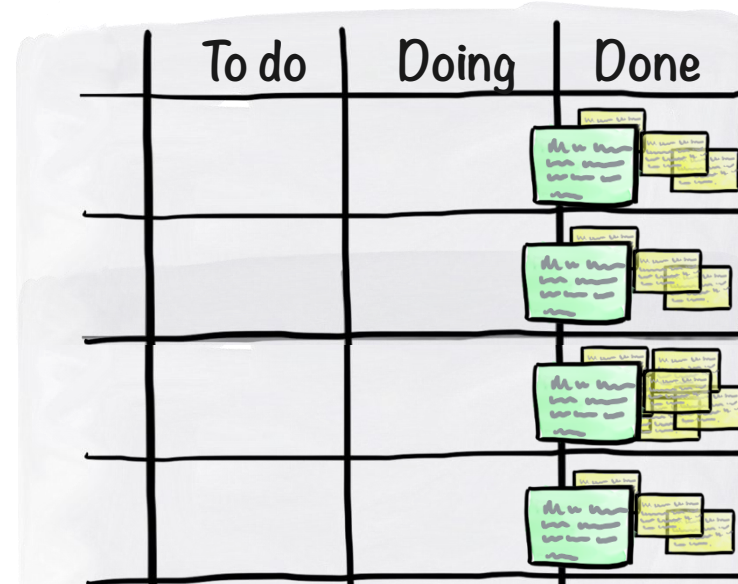
«Date»

\_\_\_\_\_  
Oracle Academy Instructor

# ¡Muchas gracias!

jordi.arino@pue.es  
@jordiAS2K

*#beagile*



Kanban Board  
(Visual Management)



pue

IMPULSANDO EL CONOCIMIENTO  
TIC CUALIFICADO