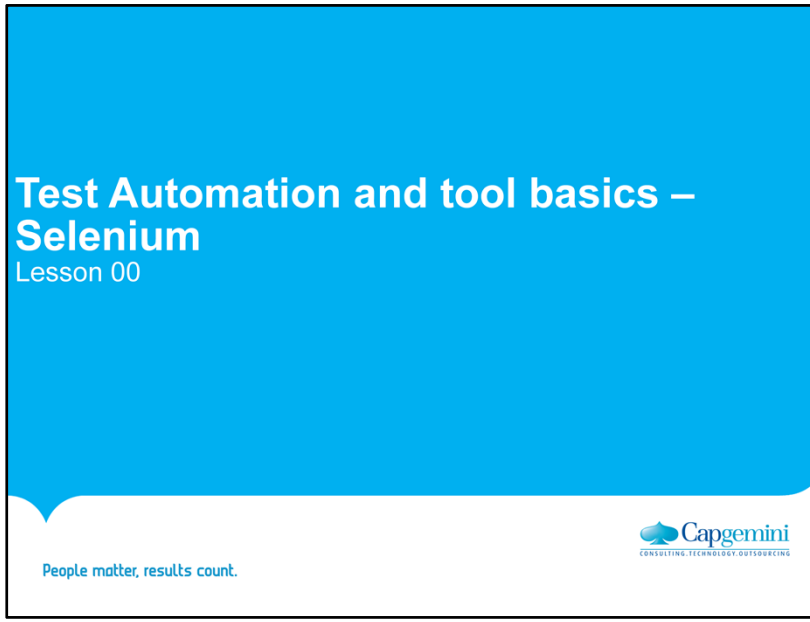


Document History

Date	Course Version No.	Software Version No.	Developer / SME	Reviewer(s)	Approver	Change Record Remarks
	1.0	Selenium RC & Web Driver	Prashanth J.			Content Creation
August 2013	2.0	Selenium IDE Selenium 2 Web Driver Latest Browsers	Shilpa Bhosle			Courseware enhancements for new version as well as to be used for ELTP batches
May, 2016	2.1	Selenium IDE Selenium 2 Web Driver Latest Browsers	Dayanand P. & Shilpa Bhosle	Shilpa Bhosle Amit Ghag	Mahima Sharma	Post-integration material alignment

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 2

Course Goals and Non Goals

- Course Goals
  - At the end of this program, participants gain an understanding of how to work with Selenium IDE for creating Test Scripts
  - To learn how to make use of Web Driver to create cross-browser based and effective Test Scripts
- Course Non Goals
  - This course does not cover other Selenium framework components like Selenium RC, Selenium Core & Selenium Grid

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 3

Pre-requisites

- Good knowledge of Software Testing
- Fair knowledge of Programming Concepts
- OOPs Concepts
- Familiarity with Automation Testing

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 4

Intended Audience

- Test engineers
- Test Leads

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 5

Day Wise Schedule

- Day 1
  - Lesson 1: Introduction to Selenium
  - Lesson 2: Working With Selenium IDE
  - Lesson 3: Overview of Object-Oriented Programming (OOP)
- Day 2
  - Lesson 4: Introduction To Java Language and Language Fundamentals
  - Lesson 5: Selenium 2 – Web Driver
  - Lesson 6: Testing Web Applications Using Web Driver API

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 6

Table of Contents

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 7

Table of Contents

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 8

Table of Contents

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 9

Table of Contents

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Table of Contents

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 11

References

- Book Reference
  - Selenium 2 Testing Tools – David Burns
- Web Reference
  - http://www.seleniumhq.org

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 12

Test Automation and tool basics -Selenium

Lesson 1: Introduction to Selenium

Lesson Objectives

- To understand the following topics:
  - What is Automation Testing?
  - Automation Testing – WHY and WHEN?
  - Manual Testing Vs Automation Testing
  - What should be automated?
  - Manual To Automated Testing
  - Disadvantages of Automation Testing
  - Introduction to Selenium
  - Features of Selenium
  - Flavors of Selenium
  - Why Selenium?
  - Summary

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 2

What is Automation Testing?

- The “Automation Testing” automates the job of testing a software
- In Automation Testing, a separate software is used to test the existing functional production software to be rolled out, based on the test cases identified
- Automation Testing reduces the overall efforts and time required in regression testing and speeds up testing life cycle

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 3

Automation Testing – WHY and WHEN?

- Frequent regression testing
- Virtually unlimited execution of test cases is required
- Rapid feedback to developers
- Reduction in human efforts
- Test same application in multiple environment
- Finding defects missed in manual testing

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

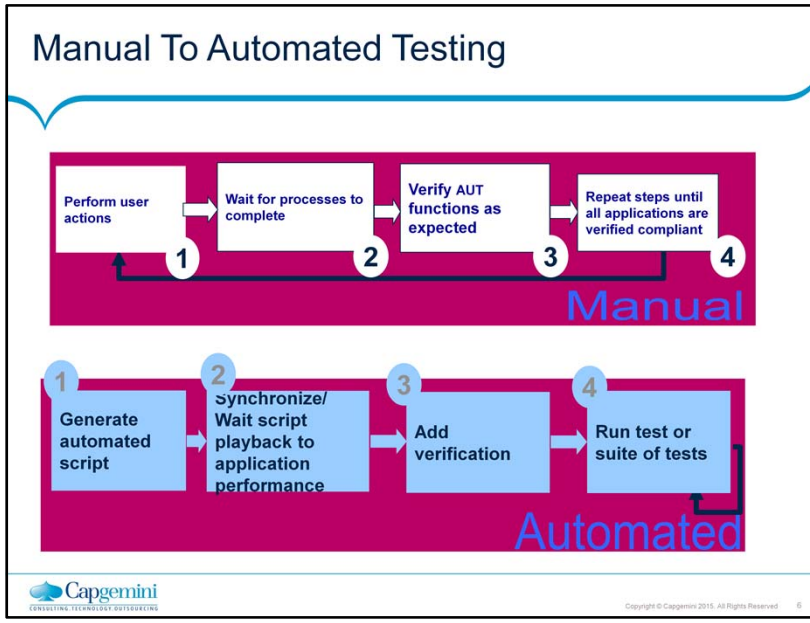
 4

Manual Testing Vs Automation Testing

- Manual Testing
  - Time consuming
  - Low reliability
  - Human resources
  - Inconsistent
- Automated Testing
  - Speed
  - Repeatability
  - Programming capabilities
  - Coverage
  - Reliability
  - Reusability

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 5


Manual To Automated Testing
Perform user actions
1
Wait for processes to complete
2
Verify AUT functions as expected
3
Repeat steps until all applications are verified compliant
4
Manual
1
Generate automated script
2
Synchronize/ Wait script playback to application performance
3
Add verification
4
Run test or suite of tests
Automated
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved 6

What Should Be Automated?

- Good candidates
  - Tests executed for each build
  - Business critical tests
  - Tests that are difficult/tedious to perform manually
- Bad candidates
  - Tests without predictable results
  - Test that require variable input/responses from the tester
  - Tests that perform operations in multiple environments

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 7

Automation Testing - Disadvantages

- High Initial Investment
- High Maintenance Cost
- Skill requirement
- Higher Timelines before use
- Long Payback Period
- Test Scripts Quality
- How to derive long term value

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 8

Introduction To Selenium

- Selenium is one of the most well known testing frameworks in the world that is in use
- It is an open source project that allows testers and developers alike to develop functional tests to drive the browser
- A functional testing tool for web applications
- It runs tests via a real browser that is driven by a JavaScript engine which is called "the BrowserBot"
- Works with any JavaScript-enabled browser “, since Selenium has been built using JavaScript
- It can be used to easily record and play tests

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 9

Features of Selenium

- Allow Cross browser testing (Record in Firefox, Execute in IE)
- No dedicated machine required for test execution(user can work in parallel)
- Selenium uses JavaScript and IFrames to embed the BrowserBot in your browser
- The engine is tweaked to support wide range of browsers on Windows, Mac OS X and Linux

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 10

Features of Selenium

- Languages Supported by Selenium – By Seleniumhq
  - Java
  - C#
  - Ruby
  - Python
  - JavaScript
- Third Party Language Bindings – NOT DEVELOPED by Seleniumhq
  - Perl
  - PHP
  - Haskell
  - Objective-C
- One should know at least one of these programming languages to dig deeper into Selenium

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

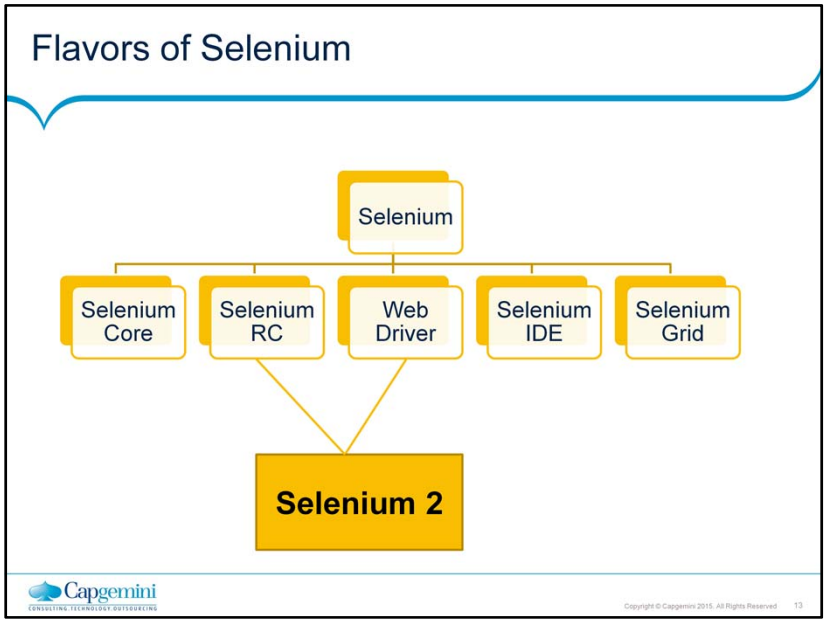
 11

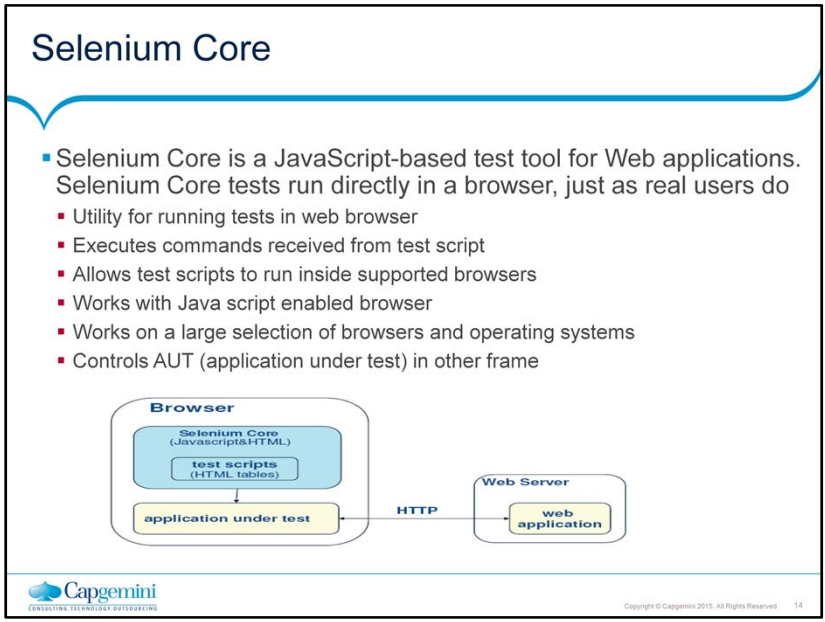
Features of Selenium

- Browsers Supported by Selenium
  - Mozilla Firefox
  - IE
  - Google Chrome
  - Opera
- Operating Systems supported by Selenium
  - Windows
  - Mac
  - Linux
  - Unix
  - Many more…..

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 12


Flavors of Selenium
Selenium
Selenium Core
Selenium RC
Web Driver
Selenium IDE
Selenium Grid
Selenium 2
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved 13



Selenium Core

Selenium was created by Jason Huggins in 2004. He created a JavaScript program that would automatically control the browser's actions. He named this program as the "JavaScriptTestRunner." Looking at the potential in this idea to help automate other web applications, he made JavaScriptRunner open-source which was later re-named as Selenium Core.

The Same Origin Policy Issue

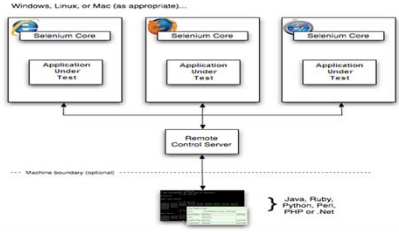
The **Same Origin Policy** restricts any JavaScript code from accessing elements from a domain that is different from where it was launched.

Example :

The HTML code in www.google.com uses a JavaScript program “HelloScript.js". According to the same origin policy HelloScript.js can access pages within google.com such as google.com/mail, google.com/login, or google.com/signup. However, it cannot access pages from different sites such as bing.com/search or yahoo.com/search because they belong to different domains.

Selenium RC (Remote Control)

- Selenium Remote Control (RC) is a test tool that allows you to write automated web application UI tests against HTTP website using any mainstream JavaScript-enabled browser
- Selenium RC consists of two parts:
- Selenium Server: works as an http proxy for web request
- Client Libraries: Client library for selected language for automation



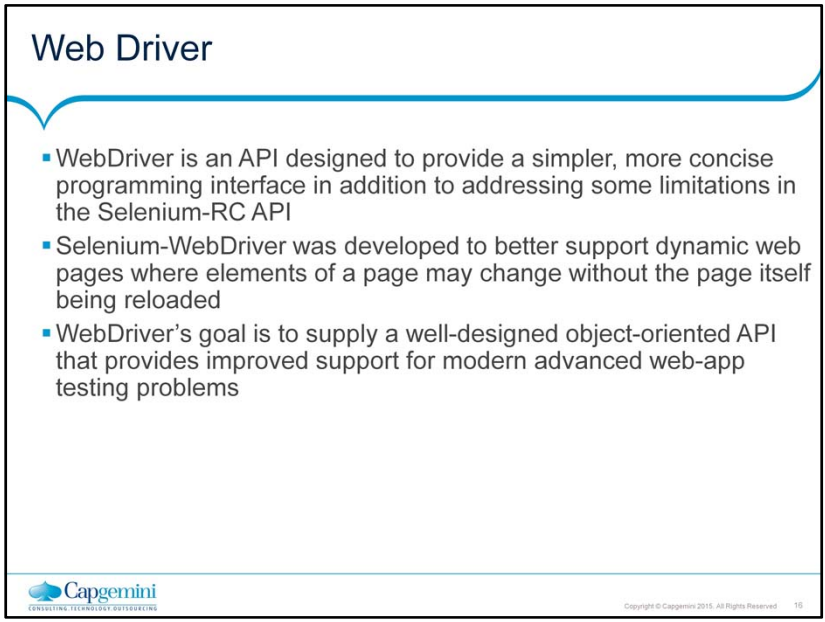
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 15

Selenium RC

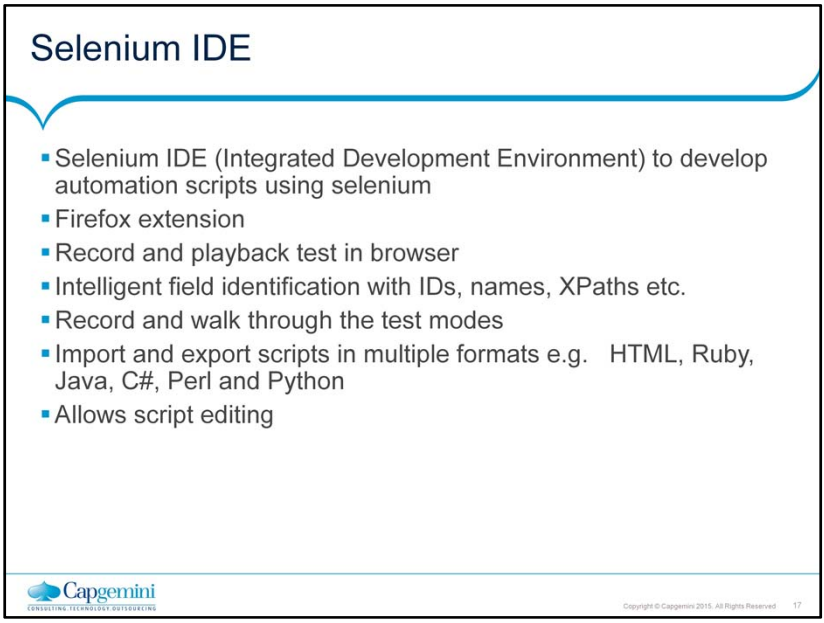
This same origin policy is the reason why prior to Selenium RC, testers needed to install local copies of both Selenium Core (a JavaScript program) and the web server containing the web application being tested so they would belong to the same domain.

Paul Hammant, ThoughtWork's engineer, created a server that will act as an HTTP proxy to "trick" the browser into believing that Selenium Core and the web application being tested come from the same domain. This system became known as the Selenium Remote Control or Selenium 1.



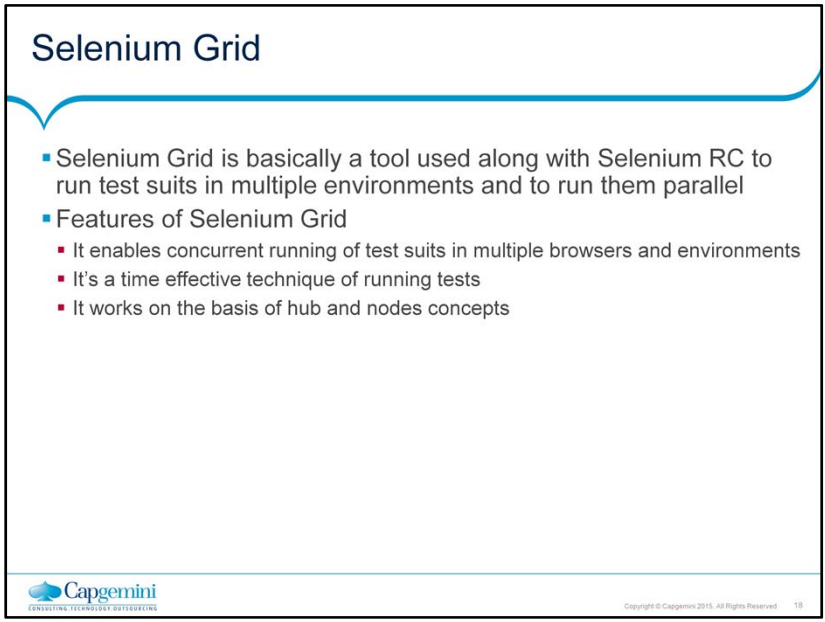
Web Driver

Simon Stewart created WebDriver circa 2006 when browsers and web applications were becoming more powerful and more restrictive with JavaScript programs like Selenium Core. It was the first cross-platform testing framework that could control the browser from the OS level.



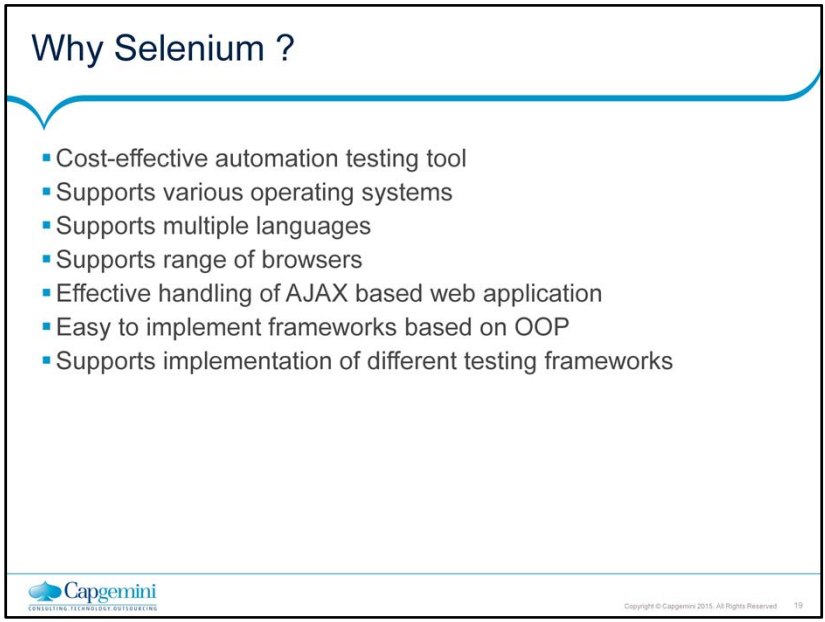
Selenium IDE

Shinya Kasatani of Japan developed Selenium IDE, a Firefox plug-in that can automate the browser through a record-and-playback feature.



Selenium Grid
Selenium Grid was developed by Patrick Lightbody to address the need of minimizing test execution times as much as possible. He initially called the system "Hosted QA."

Selenium 2
In 2008, the whole Selenium Team decided to merge WebDriver and Selenium RC to form a more powerful tool called Selenium 2, with WebDriver being the core. Currently, Selenium RC is still being developed but only in maintenance mode. Most of the Selenium Project's efforts are now focused on Selenium 2.

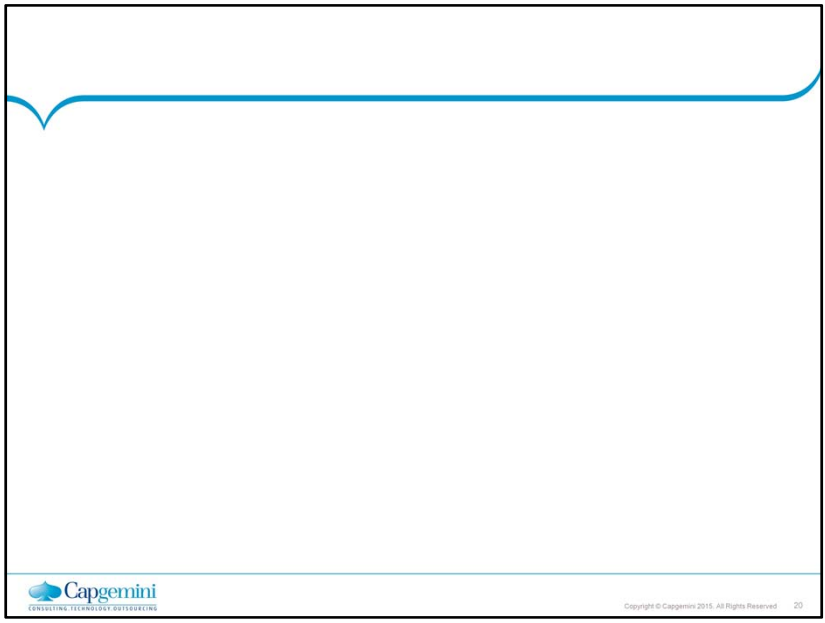


Why Selenium ?

Selenium is probably the best option for automated testing of Websites today. It is becoming increasingly popular and it is the first choice of automation testers as well as organizations for automating the testing of Web-based applications.

Following are some of the important reasons behind Selenium's growing popularity:

1. Cost-effective automation tool - Selenium is an open source testing tool and hence it serves for cost-effective automation testing. It is a free download and support is free too.
2. Supports various operating systems – In comparison with other automation testing tools, Selenium has the capability to operate on almost every operating system.
3. Supports multiple languages - Selenium supports multiple languages such as Python, Pearl, Ruby, PHP, .NET-C# and Java. You are required to be comfortable in just a single language in order to operate Selenium.
4. Supports range of browsers – Test scripts written using Selenium can be executed on range of browsers like Opera, Safari, Chrome, IE and Mozilla Firefox etc.



5. Effective handling of AJAX based web application - Selenium supports testing of web applications that implement part of their functionality within the browser using JavaScript and AJAX technologies.
6. Easy to implement frameworks based on OOP - With Selenium, it is convenient to implement frameworks that revolve around Object oriented programming like Keyword Driven, Data driven and Hybrid.
7. Supports implementation of different testing frameworks - Selenium provides support for integration of open source frameworks like TestNG, JUnit, NUnit and so on.

Review Question

- Question 1: _____ is the record and playback tool
- Question 2: ---------- is the language which is used to write test scripts in Selenium IDE
- Question 3: Selenium Grid is used to distribute your test execution on multiple platforms and environments
  - Option: True / False
- Question 4: Selenium supports testing of only web based applications
  - Option: True / False

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Review Question

- Question 5: Which of the following is not possible to automate?
  - Testing User friendly ness of UI
  - Regression testing test cases
  - Tests without expected result
  - Tests with expected result

- Question 6: Selenium IDE stands for _____
  - Integrated Development Environment
  - Information Development Environment
  - Initialization Development Environment

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Summary

- In this lesson, you have learnt:
  - Difference between Manual Testing & Automation Testing
  - Introduction to Selenium
  - Features of Selenium
  - Various flavours of Selenium

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 23

Test Automation and tool basics -Selenium

Lesson 2: Working With Selenium IDE

Lesson Objectives

- To understand the following topics:
  - Selenium IDE – An Introduction
  - Components of Selenium IDE
  - Introduction to Selenium Commands – "Selenese"
  - Understanding Element Locators in Selenium IDE
  - Matching Text Patterns
  - Storing information from the page in the test
  - Working with Alerts, Confirmation
  - Introduction to Debugging in Selenium IDE
  - Object Identification using Firebug
  - Creating Test Script using Selenium IDE
  - Creating & Executing Test Suits
  - Working with Test Scripts in Selenium IDE
  - Exporting scripts to multiple languages

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 2

Selenium IDE – An Introduction

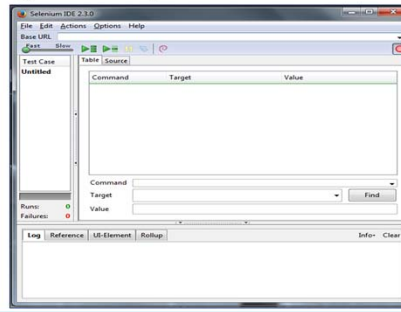
- Selenium IDE is an integrated development environment for Selenium tests
- It is implemented as a Firefox extension, and allows you to record, edit, and replay the web test in Firefox
- Using Selenium IDE is a great option available to testers to get started with writing test and group them together to build the Test Suit
- The recorded tests can be exported to many programming languages so that we can tweak them and put them in the testing framework
- The test cases and test suites can be replayed back to check the verifications and validations

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

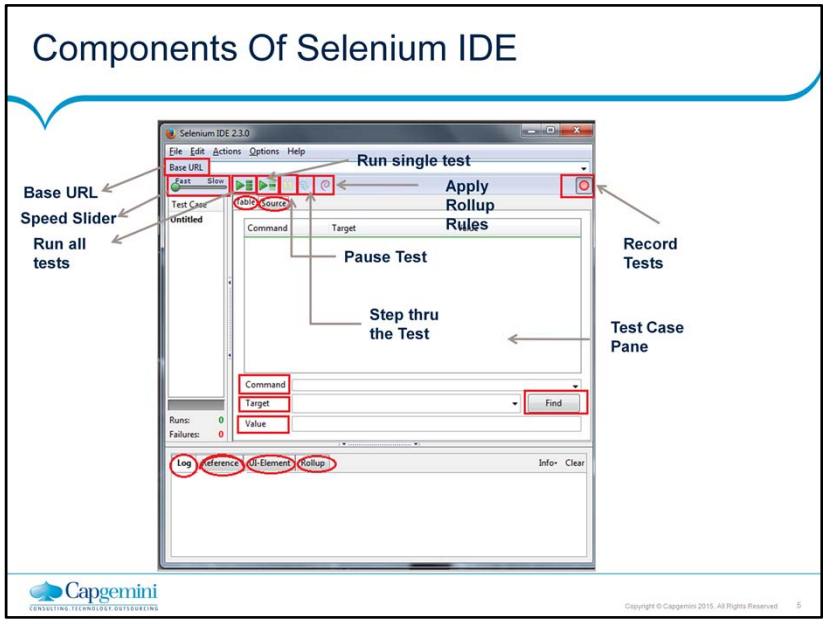
 3

Opening the Selenium IDE

- To run the Selenium-IDE, simply select it from the Firefox Web Developer option
- It opens as follows with an empty script-editing window and a menu for loading, or creating new test cases

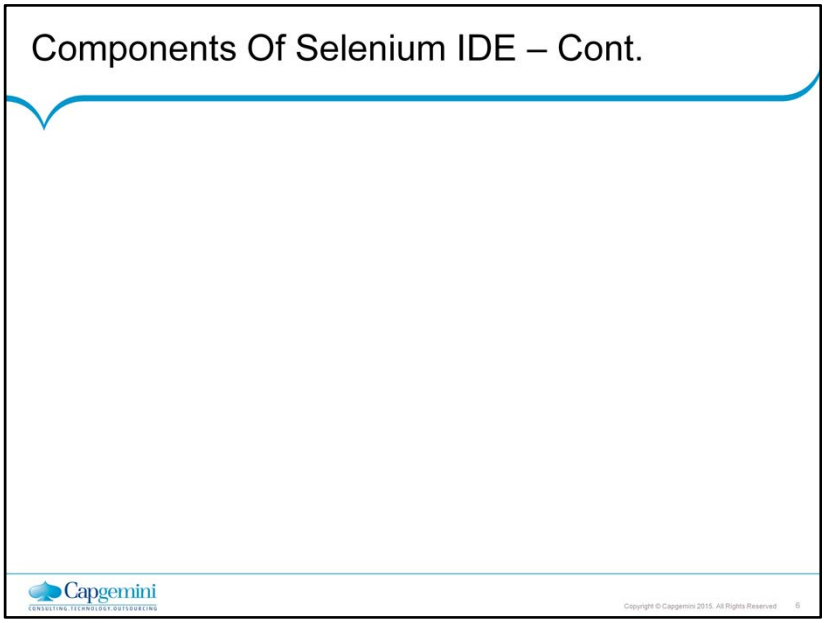


Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING



Components of Selenium IDE:

1. **Base URL:** This is the URL that the test will start at. All open commands will be relative to the Base URL unless a full path is inserted in the open command.
2. **Speed Slider:** This is the slider under the Fast and Slow labels on the screen.
3. **Run all tests:** Run all the tests in the IDE.
4. **Run single test:** Run a single test in the IDE.
5. **Pause Test:** Pause a test that is currently running.
6. **Step thru the test:** Step through the test once it has paused.
7. **Apply Rollup Rules:** This advanced feature allows repetitive sequences of Selenium commands to be grouped into a single action.
8. **Record:** Records the user's browser actions.
9. **Test Case Pane:** Your script is displayed in the test case pane. It has two tabs, one for displaying the command and their parameters in a readable "table" format.
10. **Source:** Source displays the test case in the native format in which the file will be stored. By default, this is HTML although it can be changed to a programming language such as Java or C#, or a scripting language like Python.



Components of Selenium IDE – Cont.

11. **Command:** The Command selectbox has a list of all the commands that are needed to create a test. You can type into it to use the auto complete functionality or use it as a dropdown.
12. **Target:** The Target textbox allows you to input the location of the element that you want to work against.
13. **Find:** The Find button, once the target box is populated, can be clicked to highlight the element on the page.
14. **Value:** The Value textbox is where you place the value that needs to change. For example, if you want your test to type in an input box on the web page, you would put what you want it to type in the value box.
15. **Log:** When you run your test case, error messages and information messages showing the progress are displayed in this pane automatically, even if you do not first select the Log tab. These messages are often useful for test case debugging.
16. **Reference:** The Reference pane will display documentation on the current command.
17. **UI-Element:** The UI-Element is for advanced Selenium users. It uses JavaScript Object Notation (JSON) to define element mappings.
18. **Rollup:** Rollup allows you to execute a group of commands in one step.

Demo

- Recording our first Test
- Running the Test
- Inserting commands

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 7

Introduction to Selenium Commands – "Selenese"

- Selenium commands, often called as "Selenese", are the set of commands that run your tests
- A sequence of these commands is a test script
- Selenium provides a rich set of commands for fully testing your web-app in virtually any way you can imagine
- These commands essentially create a testing language
- Selenese is essentially just a language which is nothing but the syntax and not dependent upon any language like C#, Java etc
- Selenese commands can have up to a maximum of two parameters: target and value

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 8

Capabilities of “Selenese”

- With help of Selenese one can :
  - Test the existence of UI elements based on their HTML tags
  - Test for specific content
  - Test for broken links, input fields, selection list options, submitting forms, and table data among other things
- In addition Selenese supports testing of:
  - Window size
  - Mouse position
  - Alerts
  - Ajax functionality
  - Pop up windows
  - Event handling
  - And many other web-application features

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 9

Types of Selenium Commands

Type	Description
Actions	These are the commands that changes the state of the application by directly interacting with the page elements. **Example**: Click the link, Select the option, Type text If an Action fails, or has an error, the execution of the current test is stopped. The **"AndWait"** suffix is used while calling the action. For example **"clickAndWait"**, this suffix instructs Selenium that it should wait for a new page to load.
Accessors	These are commands that allow you to examine the state of the application and store results in variables, e.g. "**storeTitle**".

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 10

Types of Selenium Commands

Type	Description
Assertions	They are like Accessors, but they verify that the state of the application conforms to what is expected. **Examples:** "make sure the page title is something" and "verify that this radiobutton is selected". **Three types of asertions** **Assert:** When an "assert" fails, the test is aborted. For example **"assertText"** **Verify:** When a "verify" fails, the test will continue execution, logging the failure. For example **"verifyText"** **WaitFor:** Before proceeding to the next command, "waitFor" commands will first wait for a certain condition to become true. **Step passes -** If the condition becomes true within the waiting period (30 Seconds). **Step fails -** If the condition does not become true. Failure is logged, and test execution proceeds to the next command.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 11

Demo

- Using Assert and Verify in the test
- Using waitFor in the test

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 12

Selenium Commands

Command	Description
open	It opens up the page using given URL
click/clickAndWait	It performs click operation and optionally waits for a new page to load
verifyTitle/assertTitle	It verifies an expected page title
verifyTextPresent	It verifies that the expected text is present on the page
verifyElementPresent	It verifies an expected UI element, as defined by its HTML tag, is present on the page
verifyText	It verifies that the expected text along with its HTML tag are present on the page
verifyTable	It can be used to verify the expected content on the table
waitForPageToLoad	It pauses execution until an expected new page loads. Called automatically when clickAndWait is used
waitForElementPresent	It pauses the execution until the expected UI is present on the web page

Demo

- Demo on common Selenium commands

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

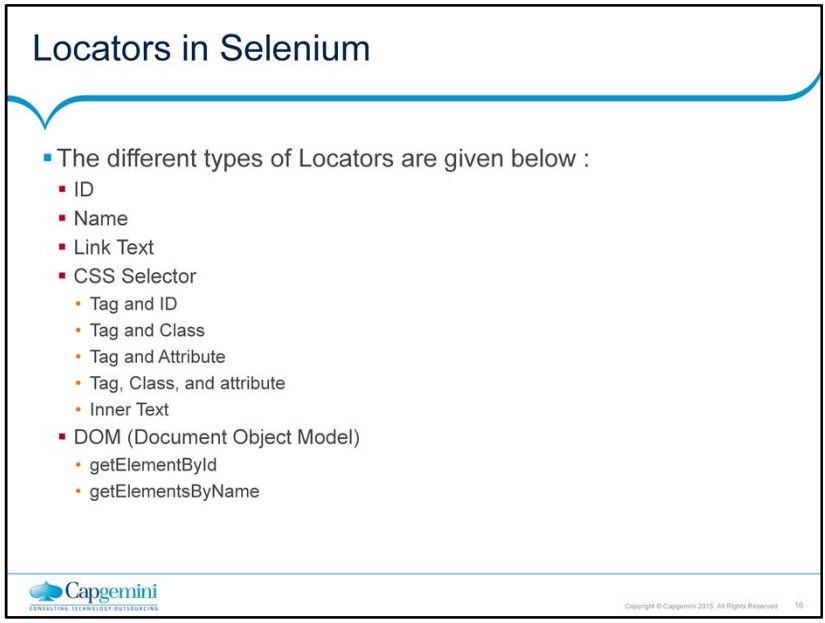
 14

Understanding Element Locators in Selenium IDE

- The “Locators” informs Selenium IDE about which GUI elements it is supposed to operate on
- Identification of correct GUI elements is a prerequisite to create an automation script
- Identifying the GUI element on a web page accurately is more difficult it sounds
- Sometimes we end up working on wrong GUI element or no elements at all
- Therefore, Selenium facilitates us with number of locators to precisely locate a GUI element on the web page

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 15


Locators in Selenium
The different types of Locators are given below :
ID
Name
Link Text
CSS Selector
Tag and ID
Tag and Class
Tag and Attribute
Tag, Class, and attribute
Inner Text
DOM (Document Object Model)
getElementById
getElementsByName
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
16

Locators in Selenium

- ID - This is the most common technique of locating elements on the web page as ID's are supposed to be unique for each element
- Name – Locating elements by their Name is very much similar to locating an element by its ID, except that we use "name=" instead
- Link Text – This type of locator is only used with the hyperlink element. We access the link by prefixing our target with "link=" and then followed by the hyperlink text

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 17

Finding elements by CSS

- CSS (Cascading Style Sheets) is a language for describing the rendering of HTML and XML documents
- CSS uses selectors for binding style properties to elements in the document
- Selenium is compatible with CSS 1.0, CSS 2.0, and CSS 3.0 selectors
- CSS Selectors are string patterns used to identify an element based on a combination of HTML tag, id, class, and attributes

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 18

Finding elements by CSS - Examples

CSS Selector	Description	Syntax & Example
Tag and ID	tag=HTML Tag id=The ID of the element being accessed	Syntax - css=tag#id Example – css=input#Uname
Tag and Class	tag=HTML Tag class=The class of the element being accessed	Syntax - css=tag.class Example – css=input.inputtext
Tag and Attribute	tag=HTML Tag [attribute=value]	Syntax – css=tag[attribute=value] Example – css=input[name=LName]
Tag, Class and Attribute	tag=HTML Tag class=The class of the element being accessed [attribute=value]	Syntax – css=tag.class[attribute=value] css=input.inputtext[name=LName]
Inner Text	tag=HTML Tag Inner text=The inner text of the element	Syntax – css=tag:contains("inner text" Example – css=input.contains("Helllo")

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 19

Locating elements by DOM - Examples

DOM	Description	Syntax & Example
getElementById	id of the element = this is the value of the ID attribute of the element to be accessed. This value should always be enclosed in a pair of parentheses (“”)	Syntax – document.getElementId(“id”) Example – document.getElementId(“txtName”)
getElementsByName	name = name of the element as defined by its ‘name’ attribute index = an integer that indicates which element within getElementsByName’s array will be used	Syntax - document.getElementsByName(“name”)[index] Example – document.getElementsByName(“rbGender”)[1]

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

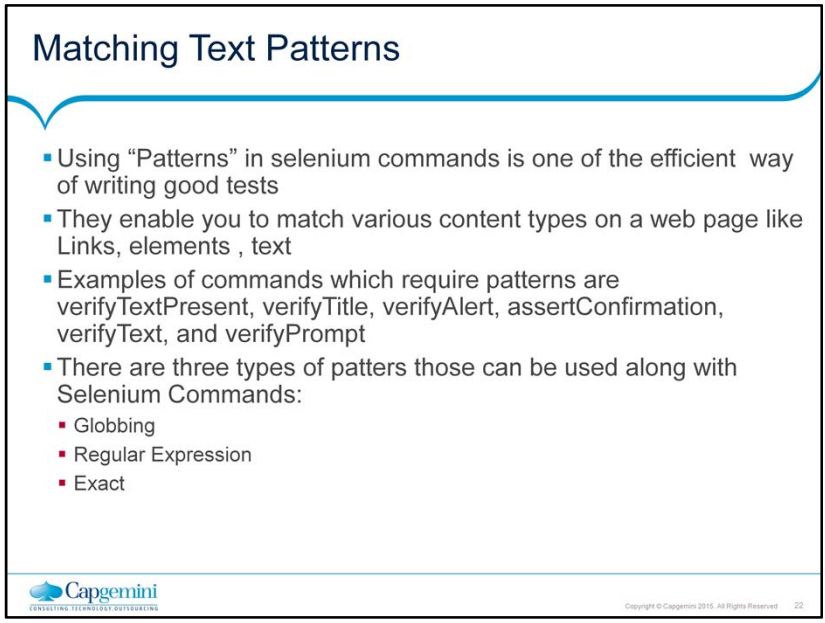
 20

Demo

- Locating Elements on the web page

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 21



Matching Text Patterns:
Just the way, we can use Locators along with selenium commands to locate a particular element on the page, similarly we can use “Patterns” to locate a particular text on the web page.

Examples of commands which require patterns are :

verifyTextPresent,verifyTitle, verifyAlert, assertConfirmation, verifyText, andverifyPrompt.

“Patterns allow you to describe, via the use of special characters, what text is expected rather than having to specify that text exactly.”

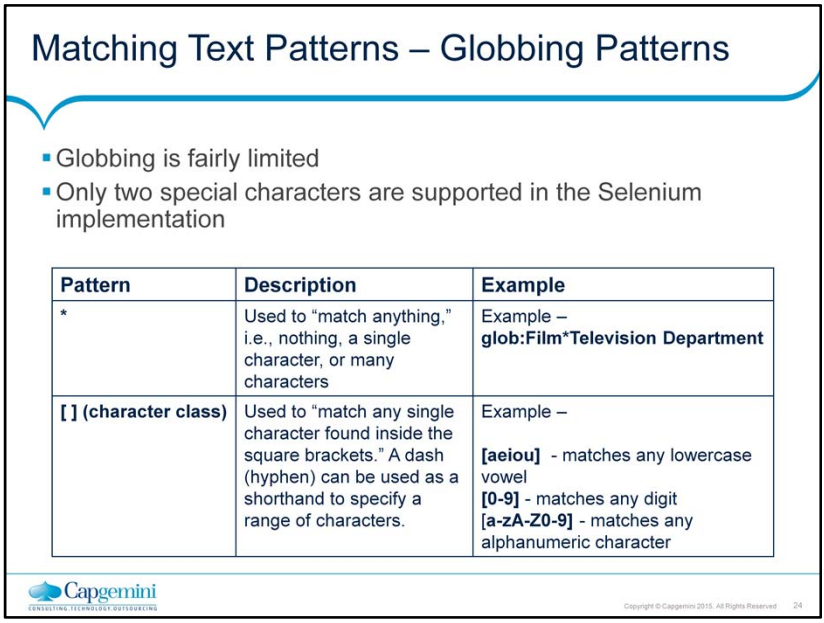
There are three types of patterns: globbing, regular expressions, and exact.

Matching Text Patterns – Globbing Patterns

- "Globbing Patterns" is the one of the matching text patterns in selenium
- You can describe expected text pattern with command's target column and can use it with verify and assert commands
- We can use globbing pattern when expected text string is dynamic and can use with commands like verifyTitle, assertText, verifyTextPresent, assertTextPresent etc

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 23



Matching Text Patterns – Globbing Patterns

While using a globbing pattern parameter along with a Selenese command, you can prefix the pattern with a glob: label. However, because globbing patterns are the default, you can also omit the label and specify just the pattern itself.
Example:

Command	Target
Click	link=glob:iGATE*Forbes take the iGATE CEO Cup Global
verifyTitle	glob:*iGATE*Forbes*

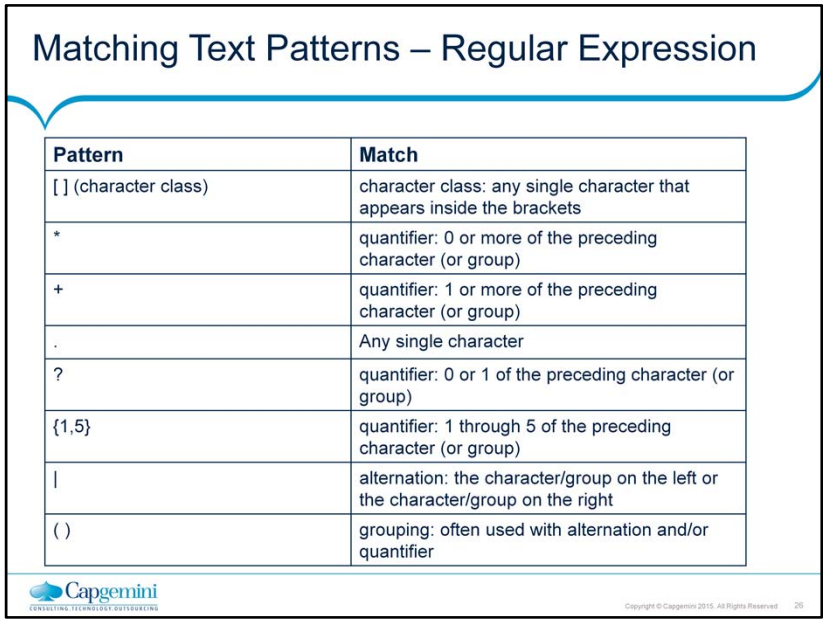
The above selenium commands uses globbing pattern.
The "click" command will work even if the link text is changed to "iGATE & Forbes take the iGATE CEO Cup Global" or "iGATE and Forbes take the iGATE CEO Cup Global".
The glob pattern's asterisk will match "anything or nothing"between the word "iGATE" and the word "Forbes".
By using a pattern rather than the exact text, the verifyTitle will pass as long as the two words "iGATE" and "Forbes" appear (in that order) anywhere in the page's title.

Matching Text Patterns – Regular Expression

- Regular Expression pattern is the most powerful of the three types of patterns that selenium command supports
- Regular expressions are also supported by most high-level programming languages
- In Selenese, regular expression patterns allow a user to perform many tasks that would be very difficult otherwise
- For example, if you need to create a test that needs to ensure that a textbox should contain nothing but a numeric value
- Selenese regular expression patterns offer the same wide array of special characters that exist in JavaScript

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 25



Matching Text Patterns – **Regular Expression**

The more complex example below tests that the Yahoo! Weather page for Anchorage, Alaska contains info on the sunrise time:

Command	Target
Open	http://weather.yahoo.com/forecast/USAK0012.html
verifyTextPresent	regexp:Sunrise: *[0-9]{1,2}:[0-9]{2} [ap]m

Let's examine the regular expression above one part at a time:

Pattern	Description
Sunrise: *	The string Sunrise: followed by 0 or more spaces
[0-9]{1,2}	1 or 2 digits (for the hour of the day)
:	The character : (no special characters involved)
[0-9]{2}	2 digits (for the minutes) followed by a space
[ap]m	"a" or "p" followed by "m" (am or pm)

Matching Text Patterns – Exact Pattern

- The exact type of Selenium pattern is of marginal usefulness
- It uses no special characters at all
- If you needed to look for an actual asterisk character which is special for both globbing and regular expression patterns, the exact pattern would be one way to do that
- For example, if you wanted to verify the text present on the web page like “* Conditions apply” then the code “glob:* Conditions apply” might not work
- In order to ensure that the “* Conditions apply” text is verified on the web page, the “exact” prefix can be used
- Valid pattern – exact: * Conditions apply

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 27

Demo

- Use of matching text patterns

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 28

Storing information from the page in the test

- Sometimes there is a need to store elements that are on the page to be used later in a test
- This could be that your test needs to pick a date that is on the page and use it later so that you do not need to hardcode values into your test
- You can also use Selenium variables to store constants at the beginning of a script
- Selenium variables can be used to store values passed to your test program from the command-line, from another program, or from a file
- Once the element has been stored you will be able to use it again by requesting it from a JavaScript dictionary that Selenium keeps track of
- To use the variable it will take one of the following two formats: it can look like ${variableName}

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 29

Store Commands

- Store
  - The plain store command is the most basic of the many store commands and can be used to simply store a constant value in a selenium variable
  - It takes two parameters, the text value to be stored and a selenium variable
  - Example:

Command	Target	Value
store	Selenium IDE Demo	myVariable
type	name=Textbox1	${myVariable}

  - The above test stores the value “Selenium IDE Demo” in the variable “myVariable”
  - You can read the value of the variable in the texbox on your web page named “Textbox1” by setting the value for the type command as ${myVariable}
  - Upon execution of above script will store the value “Selenium IDE Demo” in the textbox “Textbox1”

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 30

Store Commands

- storeElementPresent
  - This command stores either "true" or "false" depending on the presence of the specified element
  - Example:

Command	Target	Value
open		
storeElementPresent	name=loginName	flag1
storeElementPresent	name=Password	flag2

  - In the above test script, the variables falg1 & flag2 will store the values either true or false depending upon the presence of the two elements namely “loginName” & “Password”
  - We can verify the same by using “echo” command

Command	Target	Value
open		
storeElementPresent	name=loginName	flag1
storeElementPresent	name=Password	flag2
echo	${flag1}	
echo	${flag2}	

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 31

Store Commands

- storeText
- This command is used to store the inner text of an element onto a variable
  - For Example:

Command	Target	Value
open		
storeText	css=h2	textVar
echo	${textVar}	

- The above script will save the inner text in the variable “textVar” of the element having satisfied the condition i.e. “css=h2”

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

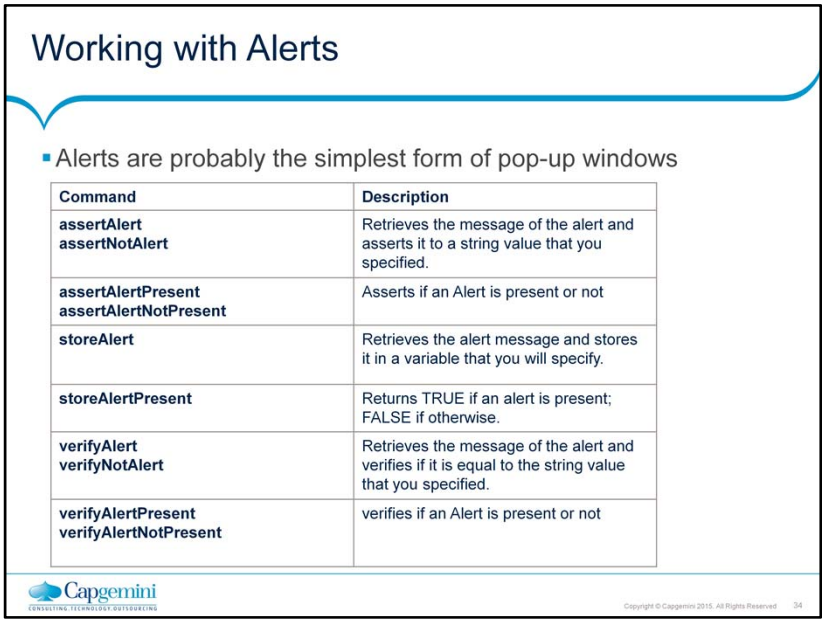
 32

Demo

- Using store commands

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 33



Working with Alerts, Confirmation:

Alerts are probably the simplest form of pop-up windows. The most common Selenium IDE commands used in handling alerts are given in the above slide:

Remember these two things when working with alerts:

1. Selenium IDE will automatically click on the OK button of the alert window and so you will not be able to see the actual alert.
2. Selenium IDE will not be able to handle alerts that are within the page's onload() function. It will only be able to handle alerts that are generated after the page has completely loaded.

Demo

- Demo on storing an alert message using storeAlert command

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 35

Working with Confirmation

- Confirmations are pop-ups that give you an OK and a CANCEL button, as opposed to alerts which give you only the OK button
- The commands you can use in handling confirmations are similar to those in handling alerts
  - assertConfirmation/assertNotConfirmation
  - assertConfirmationPresent/assertConfirmationNotPresent
  - storeConfirmation
  - storeConfirmationPresent
  - verifyConfirmation/verifyNotConfirmation
  - verifyConfirmationPresent/verifyConfirmationNotPresent
  - chooseOkOnNextConfirmation/chooseOkOnNextConfirmationAndWait
  - chooseCancelOnNextConfirmation

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 36

Demo

- Demo on usage of "chooseOkOnNextConfirmation" and "chooseCancelOnNextConfirmation"

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 37

Introduction to Debugging in Selenium IDE

- Debugging means finding and fixing errors in your test case
- This is a normal part of test case development
- Sometimes, as a test automator, you will need to debug your tests to see what is wrong
- There are various commonly used techniques are available in Selenium IDE which can be used to identify an error in the test case
- The tester can optionally break or start the execution of a test case to debug and figure out the existing error in the test case

Using Breakpoints in Test Case

- One can run up to a specific command in the middle of the test case and inspect how the test case behaves at that point
- To do this, set a breakpoint on the command just before the one to be examined
- Steps to be followed
  - Select a command
  - Right-click, and from the context menu select Toggle Breakpoint
  - Then click the Run button to run your test case from the beginning up to the breakpoint
  - Click on Step button to execute the test case which has halted as it has reached the breakpoint
  - Observer the test execution

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 39

Using Startpoint in Test Case

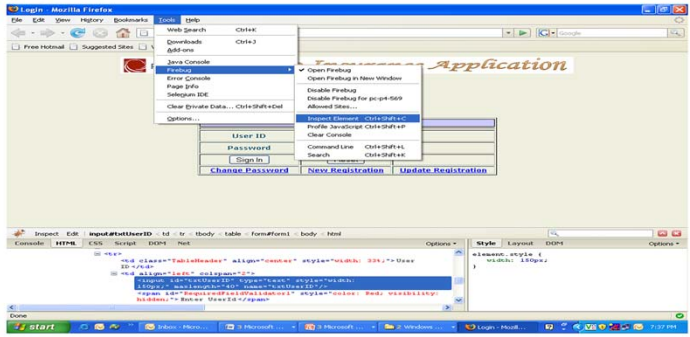
- If you have a really long test and it's failing towards the end, then you can set a custom start point so that you don't have to run the entire test when you're investigating the failure
- For example, your test might register a new user, log in, and then fail on the Home Page
- You could simply navigate to the home page yourself and set your test to start from there
- To set a start point simply right click on the first command you want Selenium IDE to execute and click 'Set / Clear Start Point'
- You will see a small play icon appear to the left of your command

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 40

Object identification using firebug

- Firebug is add on to Firefox
- It helps in getting object properties, DOM structure, CSS details and XPath



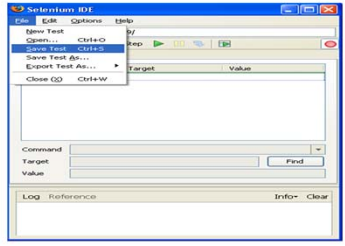
Create scripts using IDE

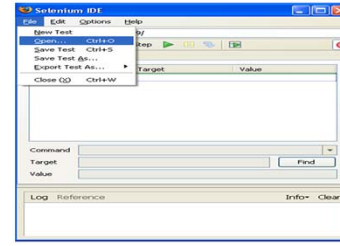
- Perform following steps to create script:
  - Perform following steps to create script:
  - Launch Mozilla Firefox
  - Open application in Firefox
  - Invoke Selenium Tools ->Selenium IDE
  - Invoke firebug Tools -> firebug -> Open Firebug
  - Enter command in Selenium IDE
  - Inspect element using firebug and specify element locator
  - Specify value if required
  - Repeat above steps as required.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Save and load scripts in IDE

- To save test go to
  - File->Save Test
  - Give the test name and save it in desired location
- To load a test go to
  - File->Open
  - And open a particular test

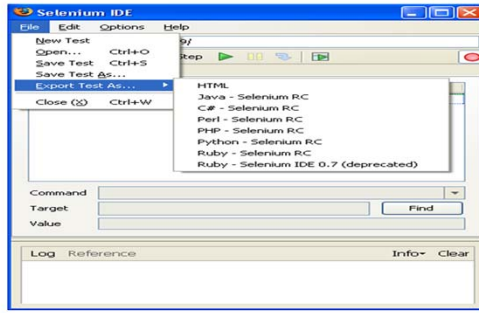




Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Export scripts to multiple language

- To export a test in particular language perform the following steps
  - File->Export Test As
  - Select the language in which you want to Export the test



Demo

- Object identification using firebug
- Create scripts using Selenium IDE
- Save and load scripts in IDE
- Export scripts to multiple language

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 45

Q.1. Actions
Q.2. verifyTextPresent
Q.3.True

Review Question

- Question 1: _____ Selenium IDE command verifies that the expected text is present on the page
- Question 2: _____ commands that changes the state of the application by directly interacting with the page elements.
- Question 3: storeText command is used to store the inner text of element in to some variable
  - Option: True / False

Q.4. storeElementPresent.
Q.5.verifyAlert
Q.6. Patterns

Review Question

- Question 4:Which command stores either "true" or "false" depending on the presence of the specified element?
  - store
  - storeText
  - storeElementPresent
  - storeElementsPresent

- Question 5:Which Selenium IDE command retrieves the message of the alert and verifies if it is equal to the string value that you specified.
- Question 6:Using _____ in selenium commands is one of the efficient way of writing good tests

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 47

Summary

- In this lesson, you have learnt:
  - Introduction & Installation of Selenium IDE
  - Working with Selenium Commands
  - Understanding Element Locators
  - Working Text Patterns
  - Storing information from the Web Page
  - Working With Alerts & Confirmation
  - Debugging in Selenium IDE
  - How to identify objects using Firebug

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

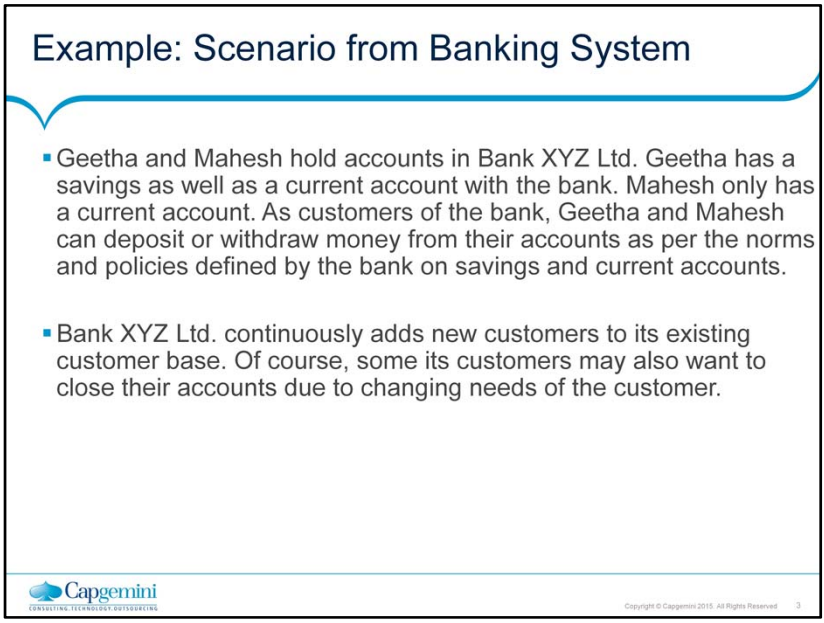
Test Automation and tool basics -Selenium

Lesson 3 : Overview of Object-Oriented Programming (OOP)

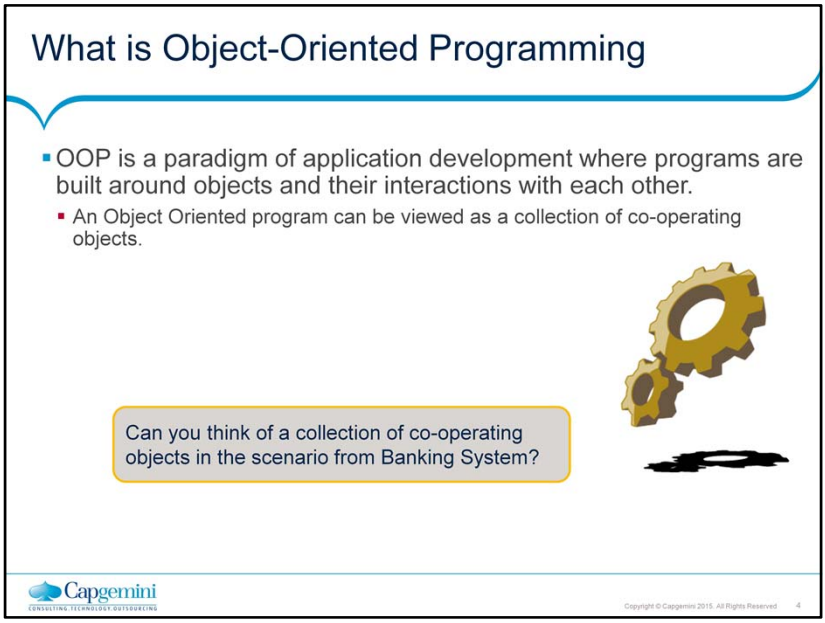
Lesson Objectives

- In this lesson, you will learn:
  - What is Object-Oriented Programming?
  - Why Object-Oriented Programming?
    - Object-Oriented Programming versus traditional software development methodologies
    - Benefits of Object-Oriented technology
  - What is an Object?
    - State, Behavior, and Identity of an Object
  - What is a Class?
    - Attributes and Operations of a Class
  - Object Oriented Principles

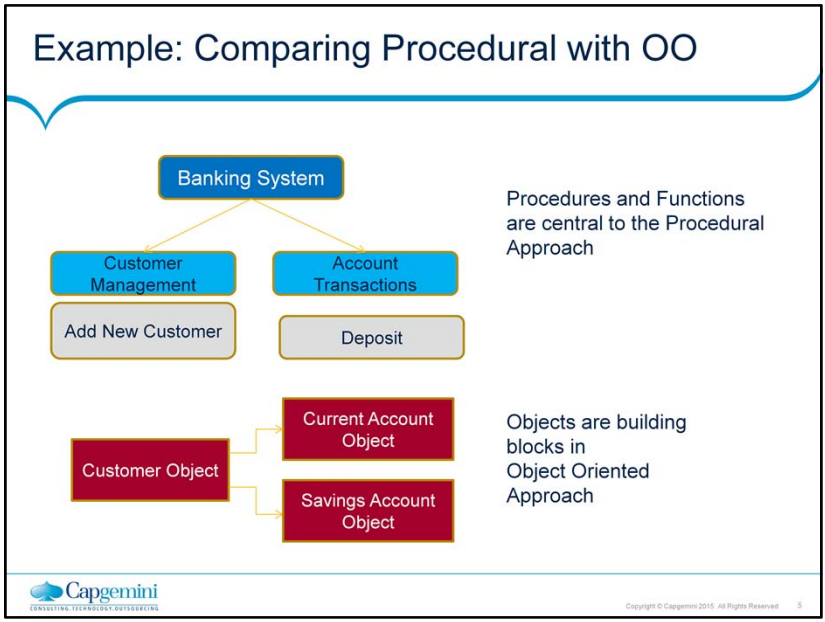
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING



If you consider this scenario for a Banking System, what services and features do you expect the bank to offer? Who are the customers mentioned for this bank? What operations can they perform on their accounts?



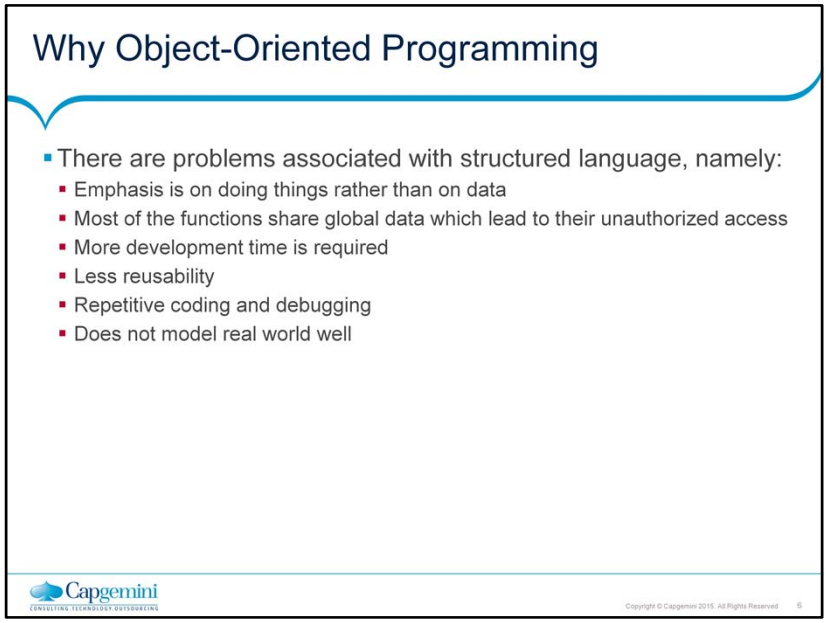
What is Object-Oriented Programming?
The object oriented approach is a fundamental shift from the procedural approach. Instead of functions and procedures being central to the program, in the OO world, we have objects that are the building blocks. An OO program is made up of several objects that interact with each other to make up the application.
For example: In a Banking System, there would be Customer objects pertaining to each customer. Each customer object would own its set of Account Objects, pertaining to the set of Savings and Current Accounts that the customer holds in the bank.
Today, most programming languages are object oriented.
For example: Java, C++, C#
Why do you think most of today's programming languages are object oriented? Are there any advantages of OO languages?



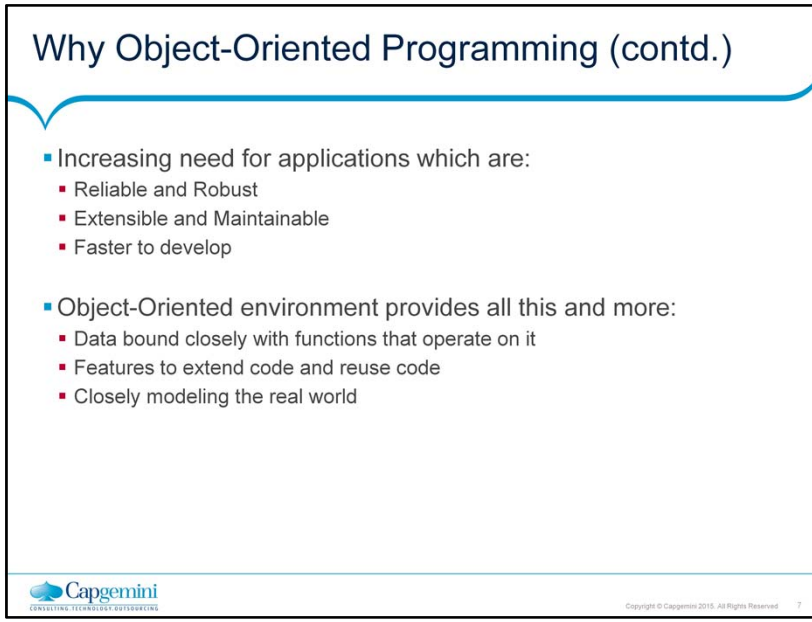
Consider the Banking System. In the procedural approach, we would try to find the top level modules of the application. Eg. A Module to maintain customers, another to maintain accounts and so on. In each of these modules, we would have procedures and functions to take care of different features. Eg. Procedure/Function to add customer or delete customer in the module for maintaining customer. Or procedures/ functions to deposit and withdraw in the module for maintaining accounts. So in the procedural approach, we identify modules, then identify procedures/functions – this is like a top down approach to system development.

Procedures and functions are the building blocks of the application in the procedural approach.

However, in the OO approach, it is the objects which are the building blocks. If we reconsider the same system, we would have objects for customer “Geetha” and customer “Mahesh”; we would also have objects for their respective savings and current accounts. These objects would interact with each other for us to achieve the desired features of the application.



Why Object-Oriented Programming?
Before the advent of Object-Oriented technology, the primary software engineering methodology was structured or procedural programming. Some drawbacks of this approach are as follows:Structured programming is based around data structures and subroutines. Data structures are simply containers for the information needed by subroutines. Thus, emphasis is almost entirely on algorithm required to solve a problem. Data is openly accessible to other parts of the program, which is risky. Structured programming tends to produce a design that is unique to that problem (thus non-reusable). Reusing code from another project usually involves a lot of effort and time. Moreover, since the emphasis is on functionality, functionality change might force entire code to be modified, thus increasing development time.In structured programming, while analysis starts with a consideration of real-world problems, the real-world focus is lost as requirements are transformed into a series of data flow diagrams.



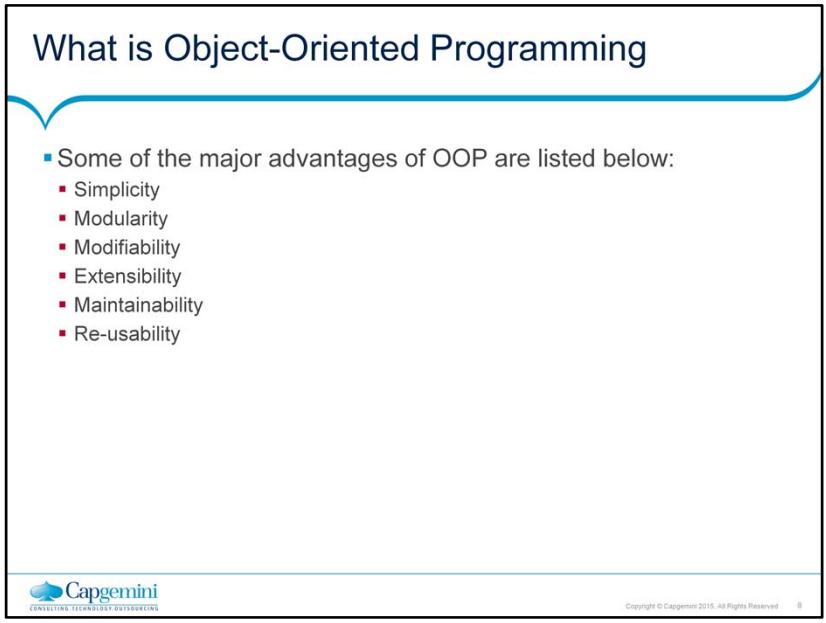
Why Object-Oriented Programming? (contd.)

With increasing complexity of software applications, some of the “must have” features are reliability, robustness, and maintainability. With increasing competition, high productivity which aids faster turn around times for application development and deployment is key.

Object-Oriented programs offer features which allow meeting the above goals. Binding of data and functions together means that data cannot be accessed unless designed for it. There is no possibility of mistakenly corrupting data. Decomposition in terms of objects allow for easily building new programs using existing objects and adding features to existing objects.

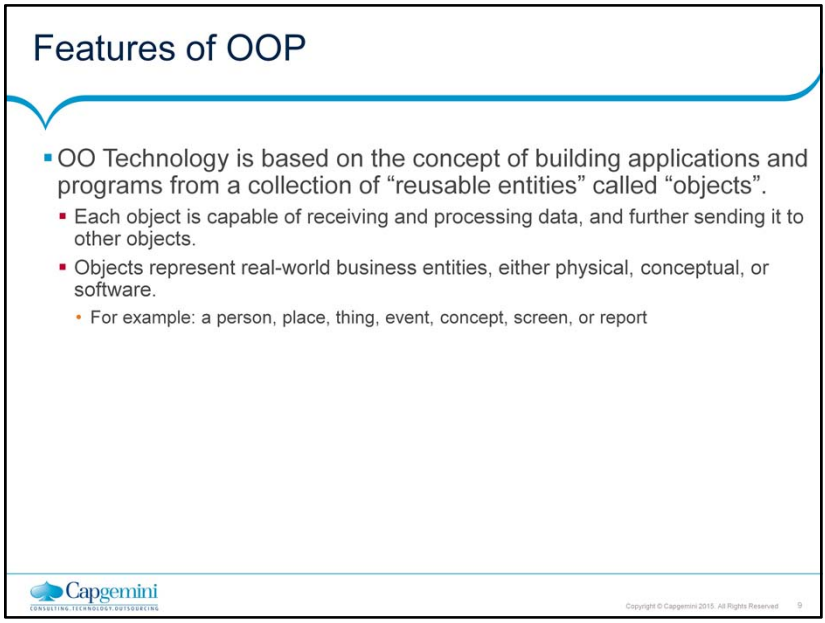
Moreover, the Object-Oriented world very closely models the real world, making it much more intuitive and faster to develop. The objects themselves often correspond to phenomena in the real world that the system is going to handle.

For example: An object can be an invoice in a business system or an employee in a payroll system. Thus, OO is natural way programming, i.e., “real life” objects are mapped as is in “programming” as classes / objects.

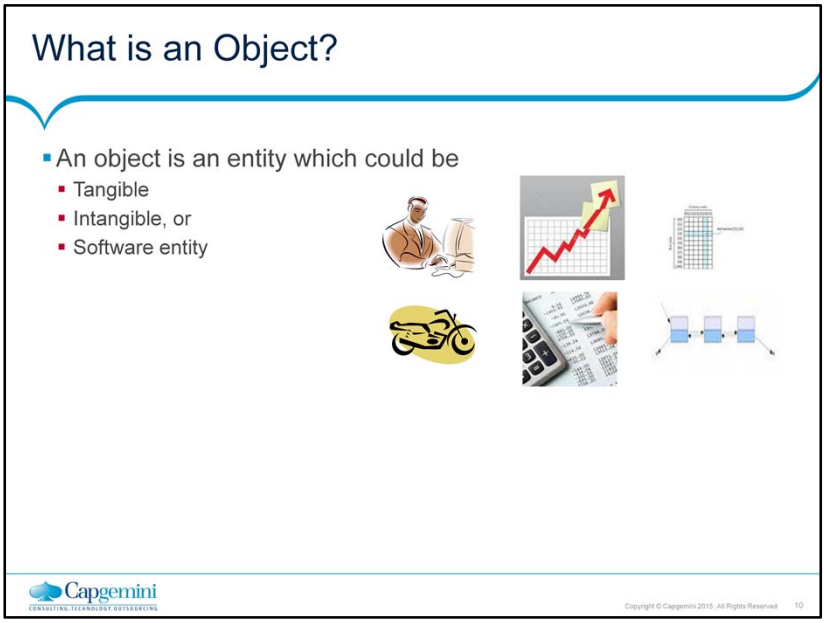


Advantages of OOP:Simplicity: Software objects model the real world objects. Hence the complexity is reduced and the program structure is very clear.Modularity: Each object forms a “separate entity” whose internal workings are decoupled from other parts of the system. Modifiability: It is easy to make minor changes in the data representation or the procedures in an OO program. Changes inside a class do not affect any other part of a program, since the only “public interface” that the external world has to a class is through the use of “methods”.Extensibility: Adding new features or responding to changing operating environments can be solved by introducing a few new objects and modifying some existing ones. Maintainability: Objects can be separately maintained, thus making locating and fixing problems easier.Re-usability: Objects can be reused in different programs.

OOP is more than just learning a new language. It requires “a new way of thinking”. The idea is primarily not to concentrate on the cornerstones of procedural languages - data structures and algorithms, instead think in terms of “objects”.



Features of OOP:
Models built by using Object-Oriented technology can be smoothly implemented in any software, by using “Object-Oriented modeling language”. These models also easily adjust to changing requirements.
It is based on best practices. As a result, the systems developed by using Object-Oriented technology are stable with a baselined architecture. The systems are more reliable, scalable, and succinct. They are more easily maintained and adaptable to change.

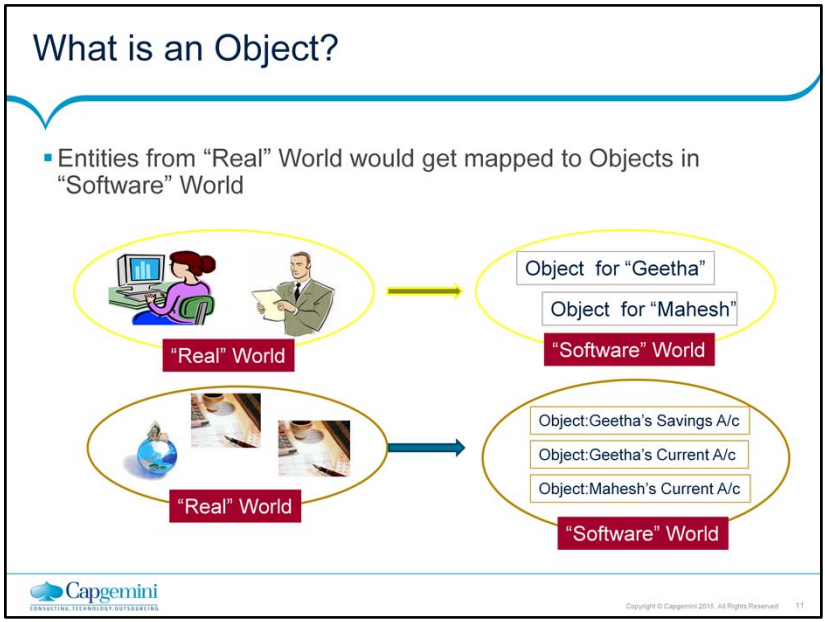


What is an Object?
An object in the real-world, can be physical, conceptual, or a software entity. We come across so many objects in the real world. In fact, everything can be considered as an object - a person, a pen, a vehicle, a book, etc. Essentially these are all tangible things that exist, can be felt, or can be destroyed.
However, there could be other entities which may not be considered as objects in the real world, like an account or a contract, or a set of business charts, or a linked list since they are “intangible” or “conceptual”. Nevertheless, they also have a well defined structure and behavior, and hence are treated as objects in the software domain.

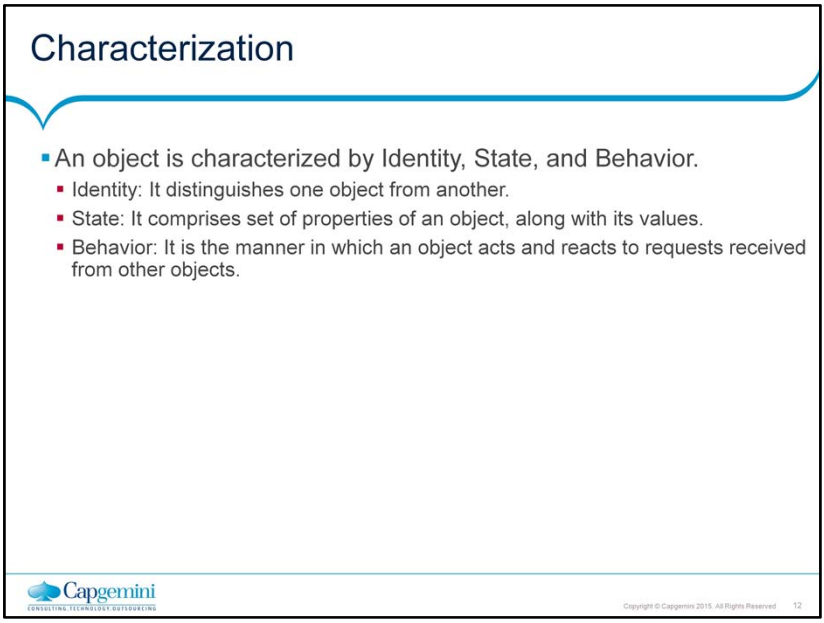
Examples of tangible entities
Person, Pen, Vehicle

Examples of intangible entities
Account, Contract, Business Charts

Examples of software entities
Database Management System



Remember the scenario from Banking System? "Geetha" and "Mahesh" from our example of Banking System are entities in the Real World. These would get mapped into corresponding objects in the software world. So in our OO application, we will have an object corresponding to "Geetha" and another object corresponding to "Mahesh". Similarly, we would have objects corresponding to Geetha's Savings Account, Geetha's Current Account as well as Mahesh's Current Account.
Typically objects could correspond to following
Roles played by people interacting with system (Like Geetha and Mahesh who are "Customer" objects)
Structures used for storing and processing data (Like Account objects for Geetha and Mahesh)
Other systems or devices interacting with system (Like Utility Payment System that can be accessed from Bank System)
Events and entities of the system (Like a transaction or a account status report)



Each object is characterized by identity, state, and behavior. Identity: Two books of same title are still two different books - they are two instances of a “book” which happen to have similar properties, just as there will be two copies if they existed in the library. The identity of one is to be distinguished from the other.State: It is one of the possible conditions that an object may be in. It is indicated by the set of values that each of its attributes possesses.
For example: An account object may be in an active or suspended state depending on the balance that it possesses.Behavior: It is what an object does when it receives instructions. For example: Deposit or withdrawal that occurs against an account object.

What is a Class?

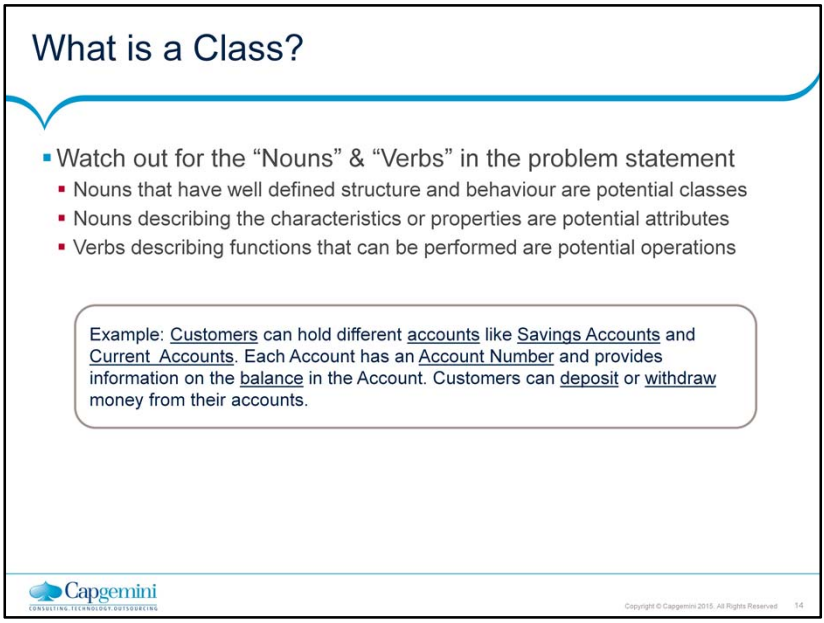
- A Class characterizes common structure and behavior of a set of objects.
- It constitutes of Attributes and Operations.
- It serves as a template from which objects are created in an application.

Class name	Customer
Class attributes	Name, Address, Email-ID, TelNumber
Class operations	displayCustomerDetails() changeContactDetails()

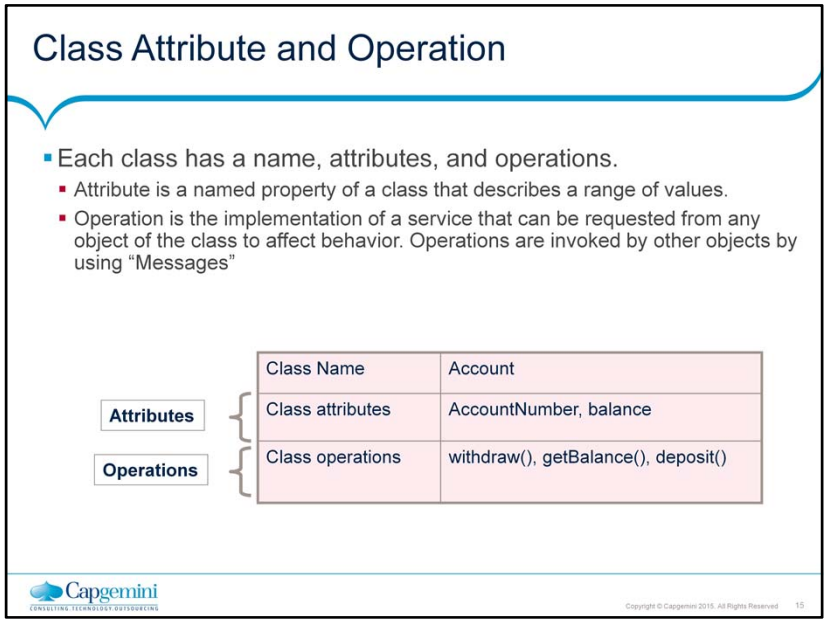
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 13

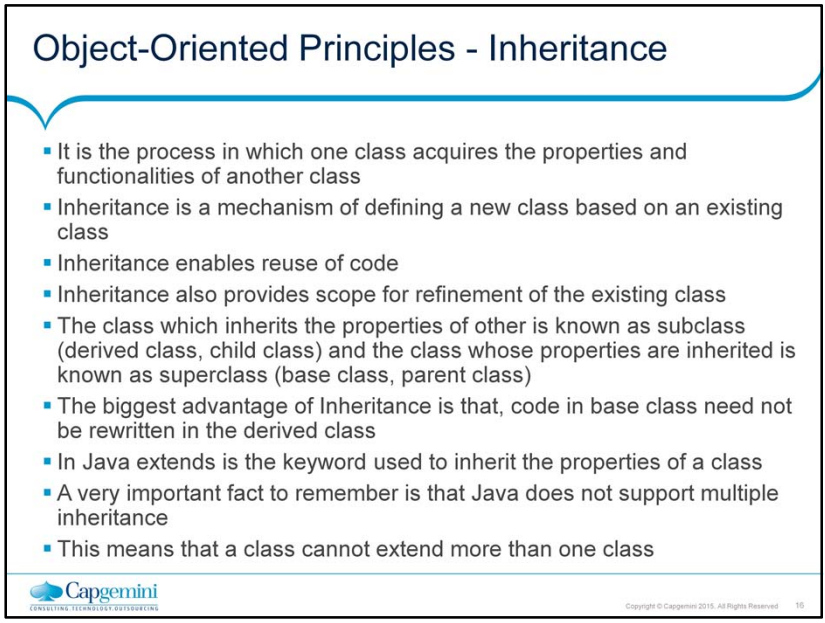
What is a Class?
Classes describe objects that share characteristics, methods, relationships, and semantics. Each class has a name, attributes (its values determine state of an object), and operations (which provides the behavior for the object).
What is the relationship between objects and classes? What exists in real world is objects. When we classify these objects on the basis of commonality of structure and behavior, what we get are classes. Classes are “logical”, they don’t really exist in real world. While writing software programs, it is the classes that get defined first. These classes serve as a blueprint from which objects are created.For example: In the example shown in the slide, there may be thousands of bank customers all having same set of attributes (i.e., Name, Address, Email-ID, TelNumber). Each customer is created from the same set of blueprints, and therefore contains the same attributes. Similarly, there can be thousands of Bank Accounts instantiated from the same “Account” class!In terms of Object-Oriented technology, we say that these customers are all “instances” of the “class of objects” known as Customer. A “class” is the blueprint from which individual “objects” are created.



To help identify potential classes, their attributes and their operations, watch out for the nouns and verbs of the problem statement. A Noun having a well defined structure and behaviour, which can be a standalone entity, is a potential class. Nouns which cannot be a stand alone entity but point to properties or characteristics of something could be potential attributes. And finally, verbs describing what could be “done” are potential operations of the class.
In the example here, note that all nouns and verbs are underlined. Potential Classes could be Customer, Account, Savings Account, Current Account. Potential Attributes (of Account Class) could be Account Number and Account Balance. Potential operations (of Account Class) could be deposit and withdraw.



Getting into Details – Class Attribute and Operation:
A class has named properties, which are attributes of the class. An attribute would be of a specific type. At runtime, an object will have associated values for each of its attributes.
A class can have several operations. An operation is an implementation of a service that can be requested from an object. When an operation of an object has to be invoked by another object, it passes a "message" to the object. Messages would correspond to the operation name.



Inheritance provides a powerful and natural mechanism for organizing and structuring your software.

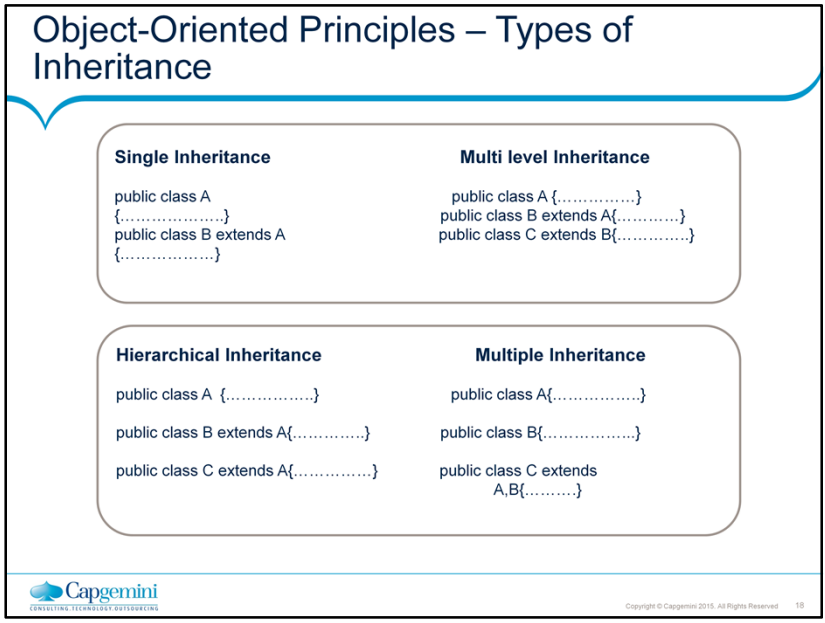
Object-Oriented Principles – Inheritance Example

```
class Calculation
{
    int Num3;

    public void Multiplication(int Num1,
    int Num2)
    {
          Num3 = Num2 * Num3 ;
          System.out.println("The addition
          of the given numbers:"+ Num3);
    }

    public void Division(int  Num1,int
    Num2)
    {
          Num3 = Num1/Num2;
          System.out.println("The division
          of the given numbers:"+Num3);
    }
}
```

```
public class MyCalculation extends
Calculation
{
      public void Substration(int Num1, int
      Num2)
      {
          Num3 = Num1-Num2;
          System.out.println("The
           difference between the given
           numbers:"+Num3);
      }
       public static void main(String
       args[])
       {
            int Num1 = 20, Num2 = 10;
            MyCalculation  cal = new
            MyCalculation();
            cal.Substration(Num1,Num2);
            cal.Multiplication(Num1,Num2);
            cal.division(Num1,Num2);  } }
```

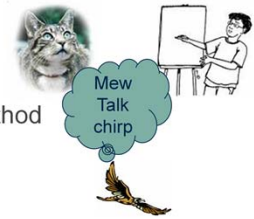
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 17


Object-Oriented Principles – Types of Inheritance
Single Inheritance
public class A
{……..}
public class B extends A
{……..}
Multi level Inheritance
public class A {……..}
public class B extends A{……..}
public class C extends B{……..}
Hierarchical Inheritance
public class A {……..}
public class B extends A{……..}
public class C extends A{……..}
Multiple Inheritance
public class A{……..}
public class B{……..}
public class C extends A,B{……..}
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
18

Object-Oriented Principles – Polymorphism

- It implies One Name, Many Forms
- It is the ability to hide multiple implementations behind a single interface
- There are two types of Polymorphism, namelv:
  - Static Polymorphism
  - Dynamic Polymorphism
- In java, we use method overloading and method overriding to achieve polymorphism



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 19

Polymorphism:
The word Polymorphism is derived from the Greek word "Polymorphous", which literally means "having many forms". Polymorphism allows different objects to respond to the same message in different ways!
There are two types of polymorphism, namely:

1. Static (or compile time) polymorphism, and
2. Dynamic (or run time) polymorphism

Polymorphism - Method Overloading Vs Overriding

- Overloading happens at compile-time while Overriding happens at runtime
- Argument list should be different while doing method overloading. Argument list should be same in method Overriding
- The most basic difference is that overloading is being done in the same class while for overriding base and child classes are required
- Overriding is all about giving a specific implementation to the inherited method of parent class
- Performance: Overloading gives better performance compared to overriding. The reason is that the binding of overridden methods is being done at runtime

Polymorphism - Method Overloading Example

```
class Sum
{
        int add(int Num1, int Num2)
       {
            return Num1+Num2;
       }
       int add(int Num1, int Num2,int Num3)
       {
            return Num1+Num2+Num3;
        }

        public static void main(String args[])
        {
             Sum obj = new Sum();
             System.out.println("Sum of two numbers: "+t.add(20, 21));
             System.out.println("Sum of three numbers: "+t.add(20, 21, 22));
        }
   }
```

Here we have 2 versions of same method add. We are overloading the method add().

Polymorphism - Method Overriding Example

```
class CarClass
{
        public int speedLimit()
        {return 80; }
}

class Alto extends CarClass
{
        public int speedLimit()
        { return 120; }

        public static void main(String args[])
        {
                CarClass c = new Alto();
                int num= c. speedLimit();
                System.out.println("Speed Limit is: "+num);
        }
}
```

Here speedLimit() method of class Alto is overriding the speedLimit() method of class CarClass.

Object-Oriented Principles - Abstraction

- Focus only on the essentials, and only on those aspects needed in the given context.
  - For example: Customer needs to know what is the interest he is earning; and may not need to know how the bank is calculating this interest
  - For example: Customer Height / Weight not needed for Banking System!

- Abstraction is a process of hiding the implementation
- details and showing only functionality to the user.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 23

Abstraction:

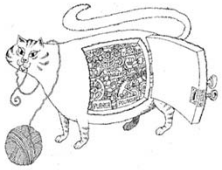
Abstraction is determining the essential qualities. By emphasizing on the important characteristics and ignoring the non-important ones, one can reduce and factor out those details that are not essential, resulting in less complex view of the system. Abstraction means that we look at the external behavior without bothering about internal details. We do not need to become car mechanics to drive a car!

Abstraction is domain and perspective specific. Characteristics that appear essential from one perspective may not appear so from another. Let us try to abstract “Person” as an object. A person has many attributes including height, weight, color of hair or eyes, etc. Now if the system under consideration is a Banking System where the person is a customer, we may not need these details. However, we may need these details for a system that deals with Identification of People.

Another way, it shows only important things to the user and hides the internal details for example sending sms, you just type the text and send the message. You don't know the internal processing about the message delivery.

Object-Oriented Principles -Encapsulation

- "To Hide" details of structure and implementation
  - For example: It does not matter what algorithm is implemented internally so that the customer gets to view Account status in Sorted Order of Account Number.
- In encapsulation the variables of a class will be classes, and can be accessed only through the current class, therefore it is also known as data



- To achieve encapsulation in Java
  - Declare the variables of a class as private.
  - Provide public setter and getter methods to modify and view the variables values.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 24

Encapsulation

Every object is encapsulated in such a way, that its data and implementations of behaviors are not visible to another object. Encapsulation allows restriction of access of internal data. Encapsulation is often referred to as information hiding. However, although the two terms are often used interchangeably, information hiding is really the result of encapsulation, not a synonym for it.

Review Question

- Question 1: Which of these keywords is used to make a class?
  - class
  - struct
  - myclass
  - int
- Question 2: Which of the following is a valid declaration of an object of class learn?
  - learn n=new learn()
  - learn n=new learn
  - n=new learn()
  - new learn n
- Question 3:In Java ‘extends’ is the keyword used to inherit the properties of a class
  - - Option: True / False

Review Question

- Question 4: Which of these is supported by method overriding in Java?
  - Abstraction
  - Encapsulation
  - Polymorphism
- Question 5: Overriding happens at compile-time while Overloading happens at runtime
  - Option: True / False
- Question 6: What are the major advantages of OOP?

Summary

- In this lesson, you have learnt:
  - The Object-Oriented Programming approach for software development
  - How Object-Oriented technology is used to design and develop stable and dynamic systems
  - Advantages of Object-Oriented Programming
  - Objects have an identity, state, and behavior
  - Class is a user-defined description of set of objects, sharing same structure and behavior
  - Object Oriented Principles

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 27

Test Automation and tool basics -Selenium

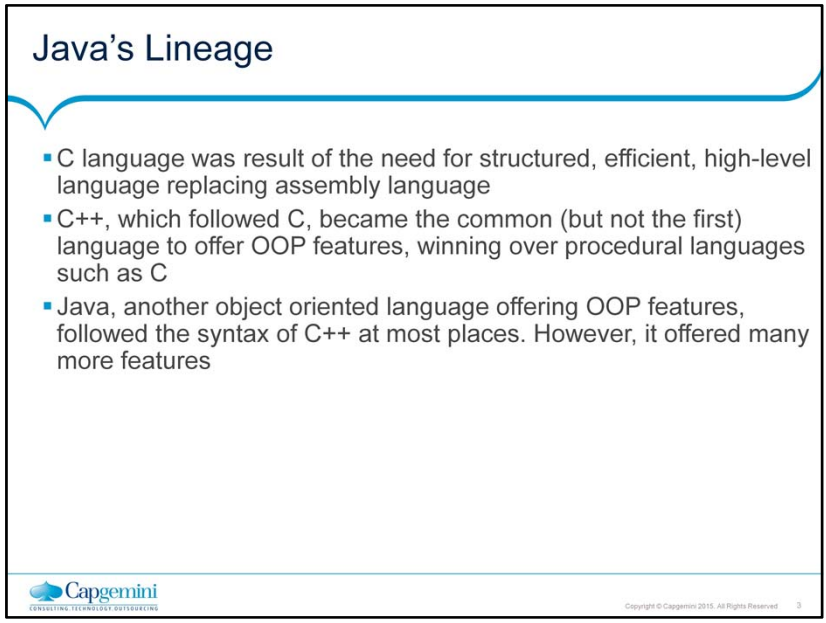
Lesson 4 : Introduction To Java Language and Language Fundamentals

Lesson Objectives

- To understand the following topics:
  - Introduction to Java
  - Features of Java
  - Java Development Process

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

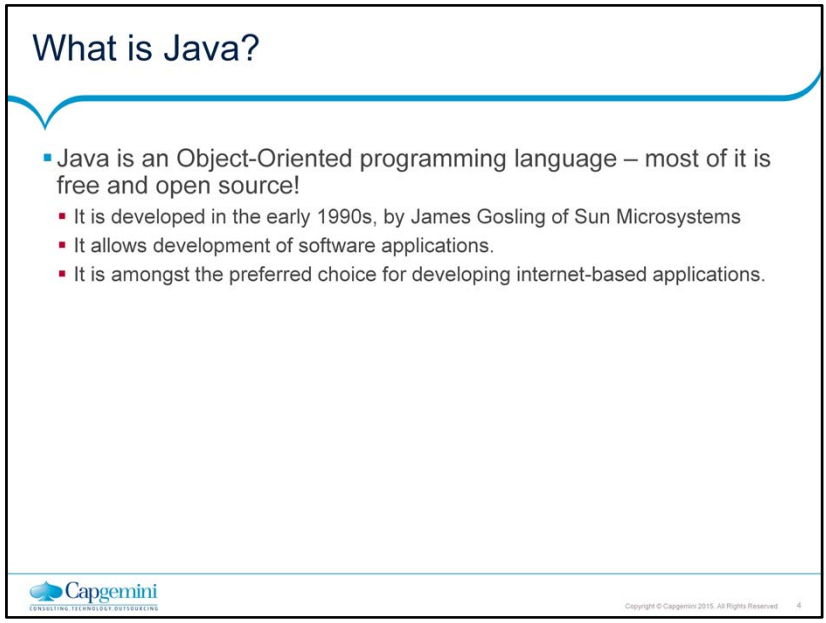
 2



Introduction to Java:

To understand Java, a new age Internet programming language, first it is essential to know the forces that drove to the invention of this kind of language. The history starts from a language called as B, which led to a famous one C and then to C++. However, the jump from C to C++ was a major development, as the whole approach of looking at an application changed from procedural way to object oriented way. The languages like, Smalltalk, were already using the Object Oriented features.

By the time C++ got the real acknowledgement from the industry, there was a strong need for a language which creates architecture neutral, platform independent, and portable software. The reason behind this particular need was the Embedded software market, which runs on variety of consumer electronic devices. Hence a project called as “OAK” was started at Sun Microsystems.

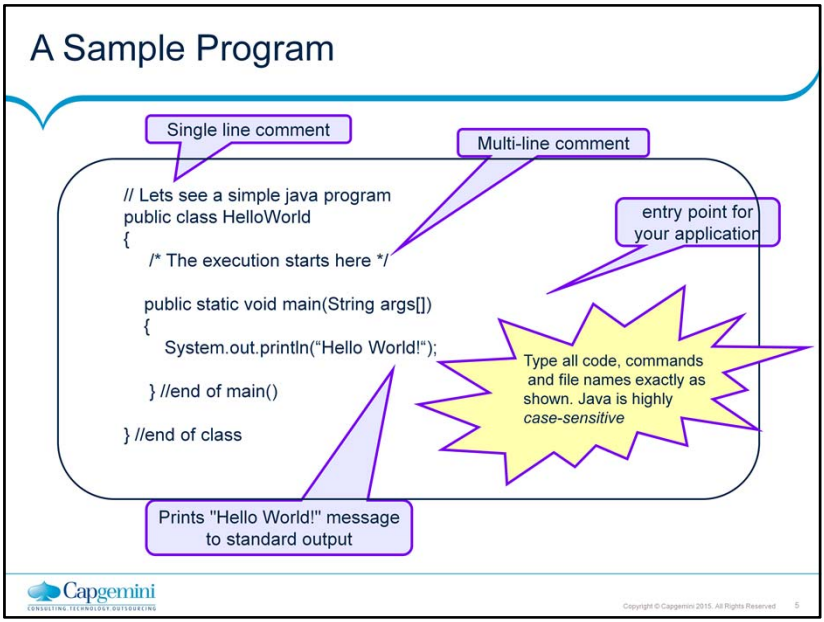


The second force behind this is WWW (World Wide Web). Now on WWW, most of computers over the Internet are divided into three major architectures namely Intel, Macintosh, and Unix. Thus, a program written for the Internet has to be a portable program, so that it runs on all the available platforms.

All this led to the development of a new language in 1995, when they renamed the “OAK” language to “JAVA”.

Introduction to Java Features – What is Java?

Java programming language is a high level programming language offering Object-Oriented features. James Gosling developed it at Sun Microsystems in the early 1990s. It is on the lines of C/C++ syntax, however, it is based on an object model. Sun offers much of Java as free and open source. Today we have the Java version 6 in the market. The Java programming language forms a core component of Sun’s Java Platform. By platform, we mean the mix of hardware and software environment in which a program executes – such as MS Windows and Linux. Sun’s Java is a “software-only” platform which runs on top of other hardware platforms. We will learn more about this on subsequent slides.

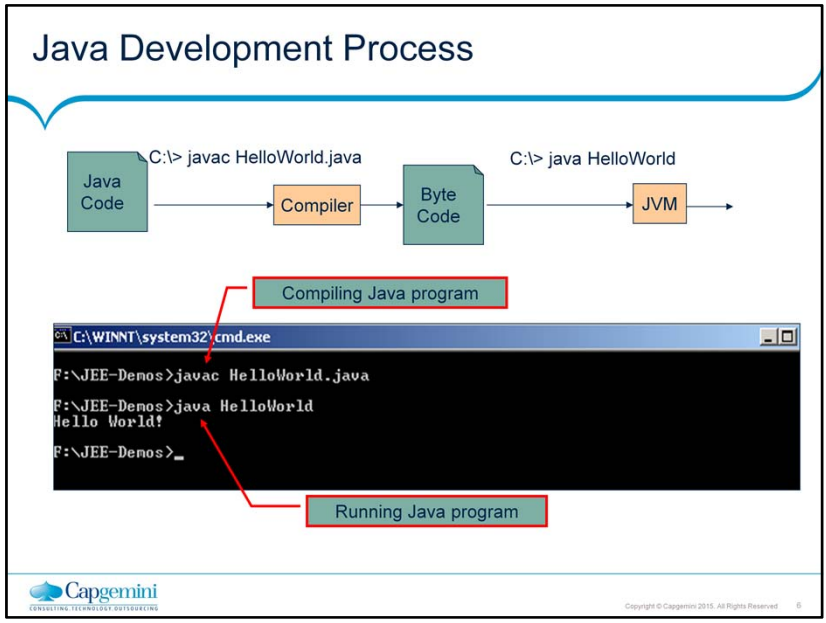


Introduction to Java Features – A Sample Program:

Here is our first Java program..

Let us have a closer Look at the "Hello World!" program:

1. class HelloWorld begins the class definition block for the "Hello World!" program.
2. Every Java program must be contained within a "class" block.
3. public static void main(String[] args) { …} is the entry point for your application.
4. Subsequently, it invokes all the other methods required by the program.
5. This program when executed will display "Hello World" on the screen. We shall see this in the next slide.



Introduction to Java Features – Java Development Process:

The Java Development Process involves the following steps:

Write the Java code in a text file with a .java extension, namely "HelloWorld.java". By convention, source file names are named by the public class name they contain.

At the command line, compile the code using the Java compiler (javac). The javac compiler converts the source into Byte Codes and stores it in a file having .class extension. What is special about byte codes? Unlike traditional compilers, javac does not produce processor specific native code. Instead it generates code which is in the language of JVM (Java Virtual Machine).

Note: To have access to the javac and the java commands, you must set your path first. To do so, you may type the following at the command prompt:

Set path=<your java-home directory>\bin
For example: Set path=C:\j2sdk1.4.1_07\bin

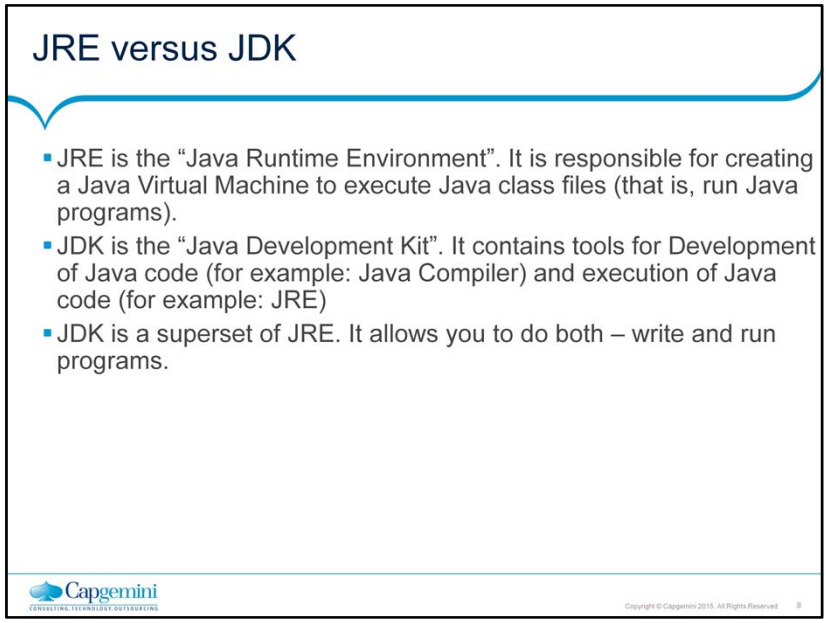
We can also set the environment variables (path and classpath) through the Control Panel.

Demo

- Creating and executing the First Java application

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

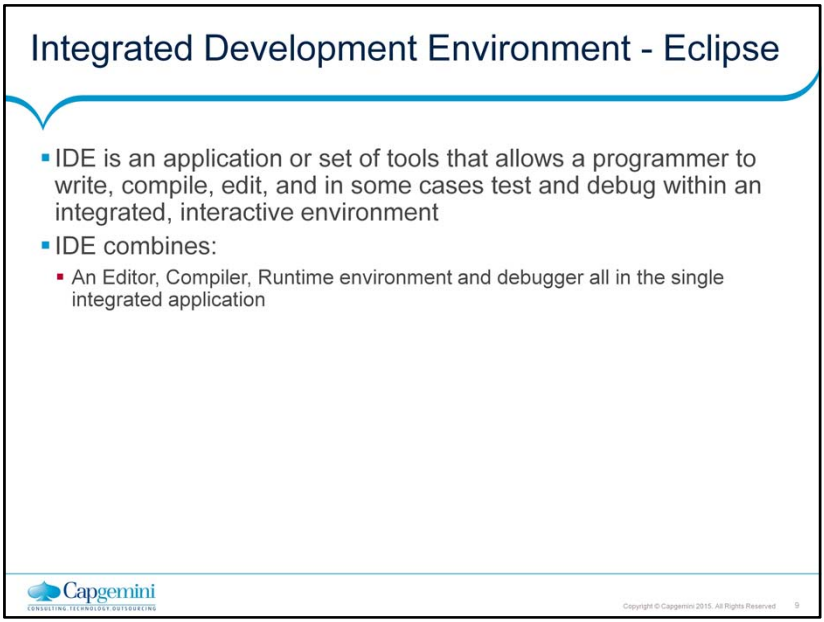
 7



Difference between JRE and JDK:

The Java Development Kit (JDK) is a superset which includes Java Compilers, Java Runtime Environments (JRE), Development Libraries, Debuggers, Deployment tools, and so on. One needs JDK to develop Java applications. We have different versions that include JDK 1.2, JDK 1.4, and so on.

The Java Runtime Environment (JRE) is an implementation of JVM that actually executes the Java program. It is a subset of JDK. One needs JRE to execute Java applications.



What is an IDE?

The Eclipse Project is an open source software development project dedicated to providing a robust, full-featured, commercial-quality, industry platform for the development of highly integrated tools and rich client applications. Eclipse runs on Windows, Linux, Mac OSX, Solaris, AIX and HP-UX.

Eclipse3.3 features include the following:

1. Creation and maintenance of the Java project
2. Developing Packages
3. Debugging a java program with variety of tools available
4. Running a Java program

Developing the Java program will be easier as Eclipse editor provides the following:

1. Syntax highlighting
2. Content/code assist
3. Code formatting
4. Import assistance
5. Quick fix

Demo

- Create Workspace
- Create a Java Project
- Select the JRE
- Adding external jar file to the project
- My first Java Program – Hello World
- Executing Hello World Program

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

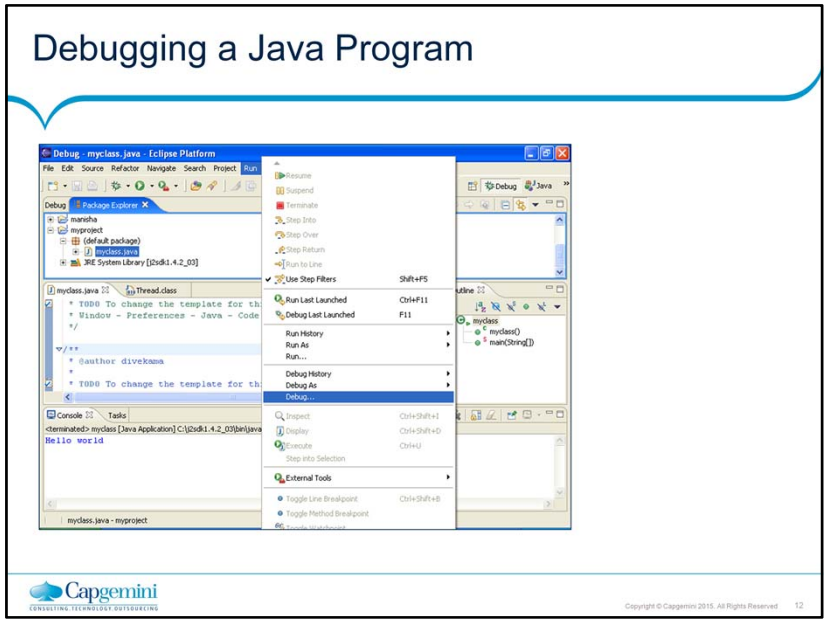
 10

Debugging your Java Program using Eclipse

- The Java Development Toolkit (JDT) includes a debugger that enables you to detect and diagnose errors in your programs running either locally or remotely
- The debugger allows you to control the execution of your program by employing the following:
  - Setting breakpoints, suspending launched programs, stepping through your code, and examining the contents of variables

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 11



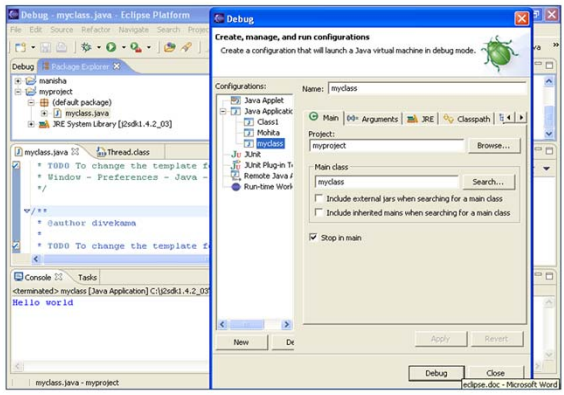
Creating and Managing Java Projects:

Debugging a Java Program:

Eclipse gives you auto-build facility, where recompilation of the necessary Java classes is done automatically.

To debug your Java program, select the Java source file which needs to be debugged, select Run → Debug. This will ask you to select an option to halt in public static void main() method, from where you may select to step into each and every function you come across or step over every function and only capture output of each function.

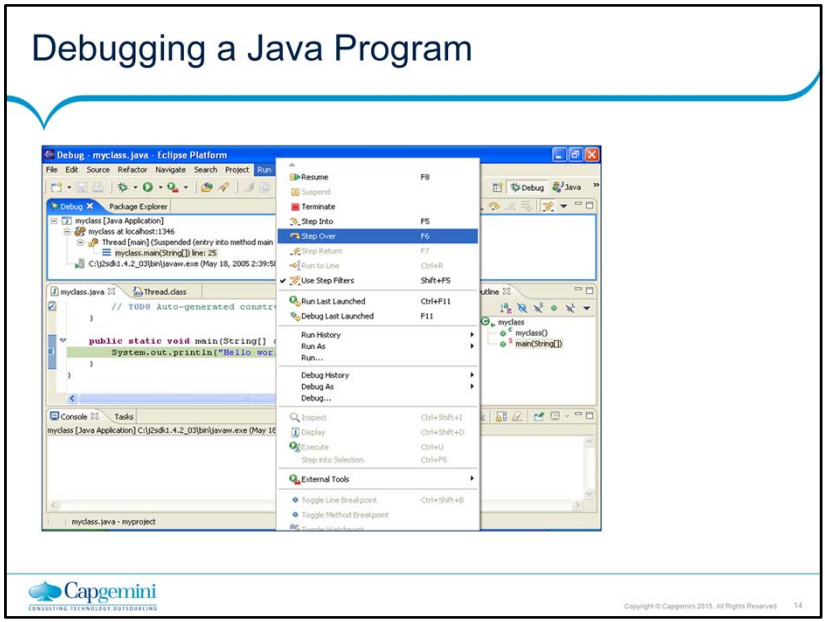
Specifying Debugging options



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 13

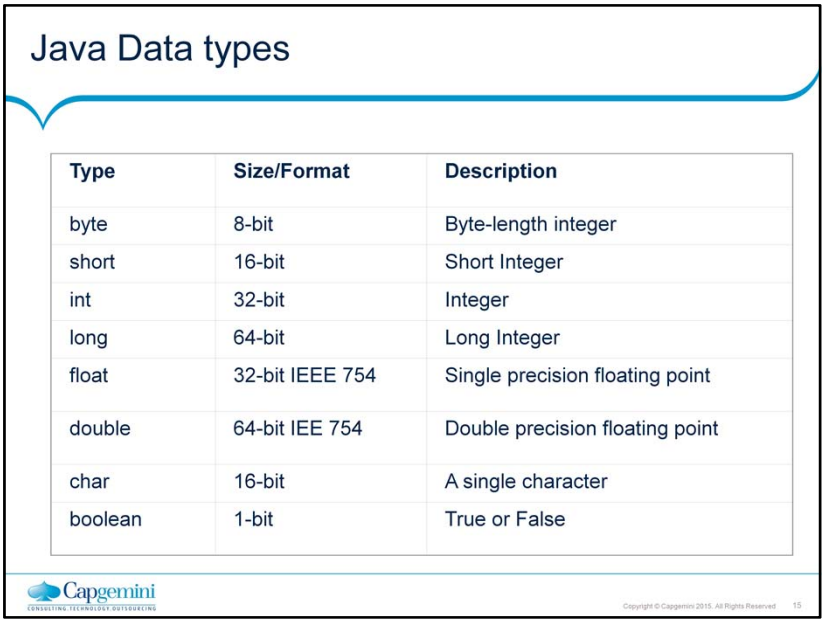
Note: Click Debug to start debugging process.



Creating and Managing Java Projects:

Debugging a Java Program:

Debugging can be attained by stepping-into or stepping-over the statements. Step-into will traverse through each and every statement in a function. Step-over will generate output after the function call is over. Tracing and watching the variable values is available as different debug views.



There are eight primitive data types supported by Java (see slide above). Primitive data types are predefined by the language and named by a key word.
The default character set used by Java language is Unicode character set and hence a character data type will consume two bytes of memory instead of a byte (a standard for ASCII character set).
Unicode is a character coding system designed to support text written in diverse human languages.
This allows you to use characters in your Java programs from various alphabets such as Japanese, Greek, Russian, Hebrew, and so on.
This feature supports a readymade support for internalization of java.

The default values for the various data types are as follows:

Integer : 0
Character : '\u0000'
Decimal : 0.0
Boolean : false
Object Reference: null

Keywords in Java

abstract	continue	for	new	switch
assert***	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum****	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp**	volatile
const*	float	native	super	while

* not used
** added in 1.2
*** added in 1.4
**** added in 5.0

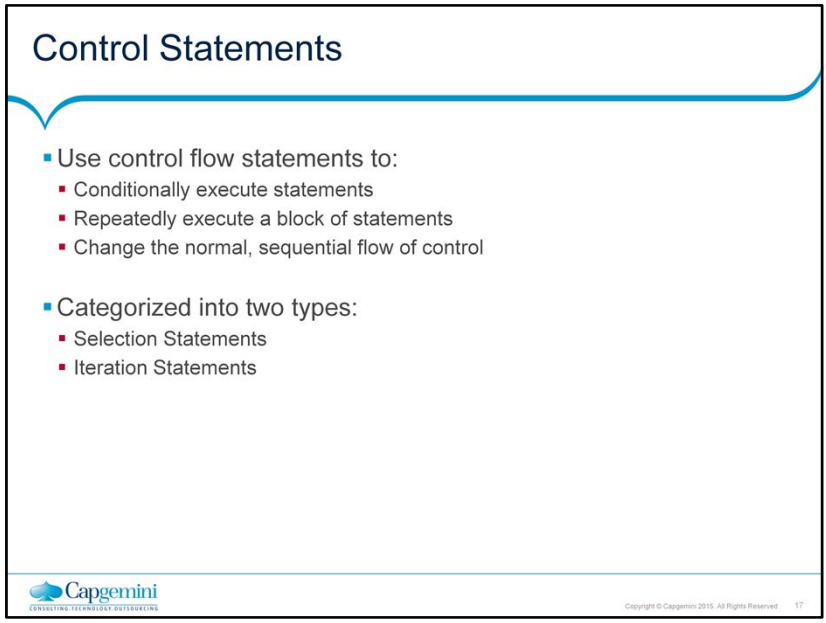
Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 16

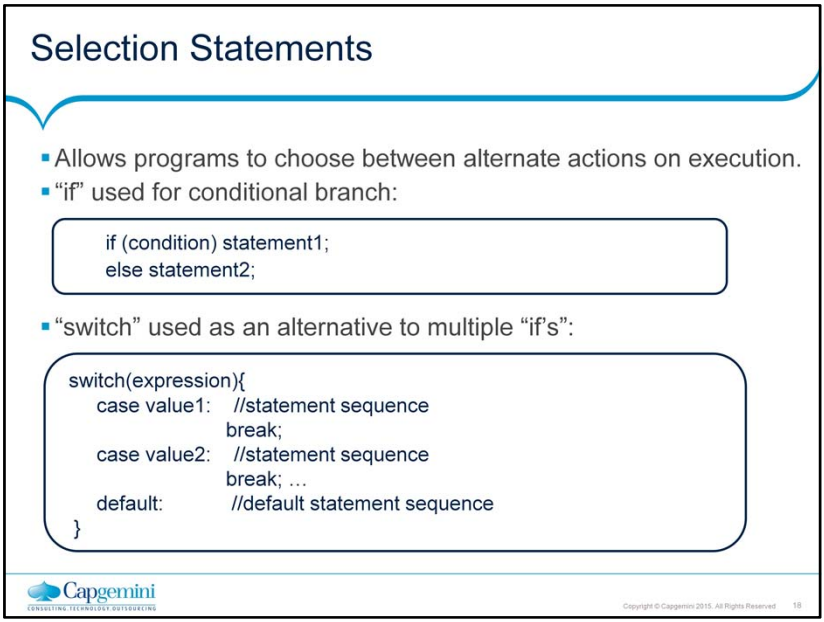
Keywords are reserved identifiers that are predefined in the language and cannot be used to denote other entities. E.g. class, boolean, abstract, do, try etc. Incorrect usage results in compilation errors.

In addition, three identifiers are reserved as predefined literals in the language: null, true and false.

The table above shows the keywords available in Java 5.



Java being a programming language, offers a number of programming constructs for decision making and looping.



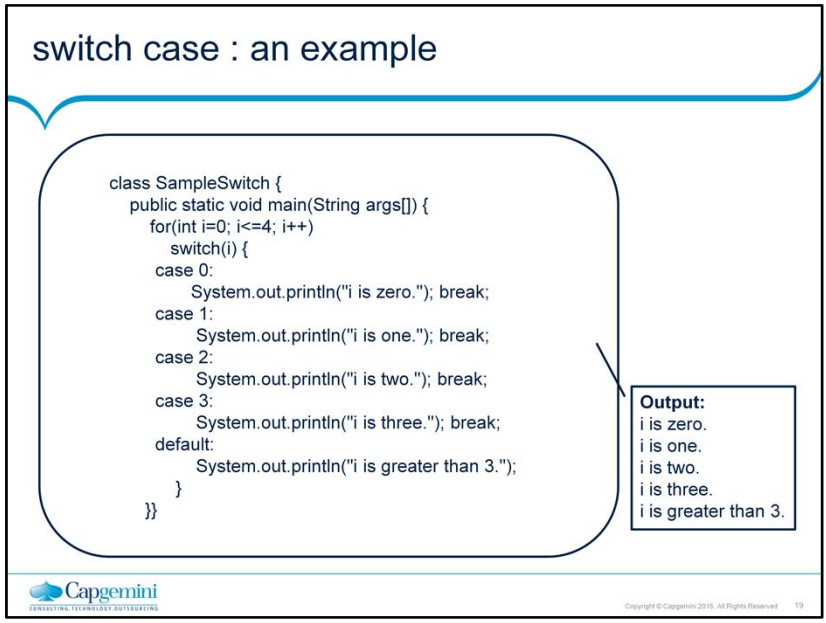
If Statement:

The if statement is Java's conditional branch statement. It can be used to route program execution through two different paths.

Each statement may be a single statement or a compound statement enclosed in curly braces. The condition is any expression that returns a boolean value. The else clause is optional.

The if works like this: If the condition is true, then statement1 is executed. Otherwise, statement2 (if it exists) is executed. In no case will both statements be executed. For example, consider the following:

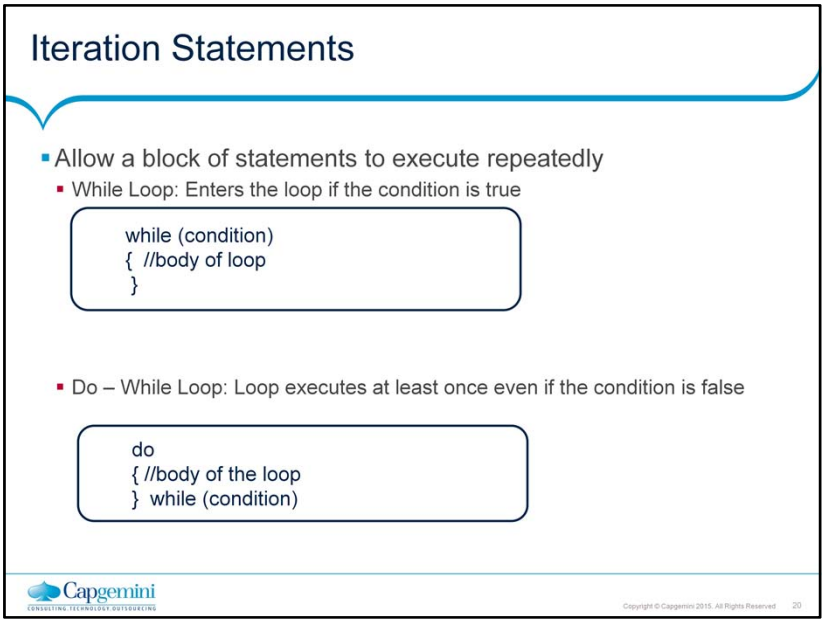
```
int a, b;
if(a < b) a = 0;
else b = 0;
```



Switch – Case:

The switch statement is Java's multi-way branch statement. It provides an easy way to dispatch execution to different parts of your code based on the value of an expression.

As such, it often provides a better alternative than a large series of if-else-if statements.



while statement:

The body of the loop is executed as long as the conditional expression is true. When condition becomes false, control passes to the next line of code immediately following the loop. Example:

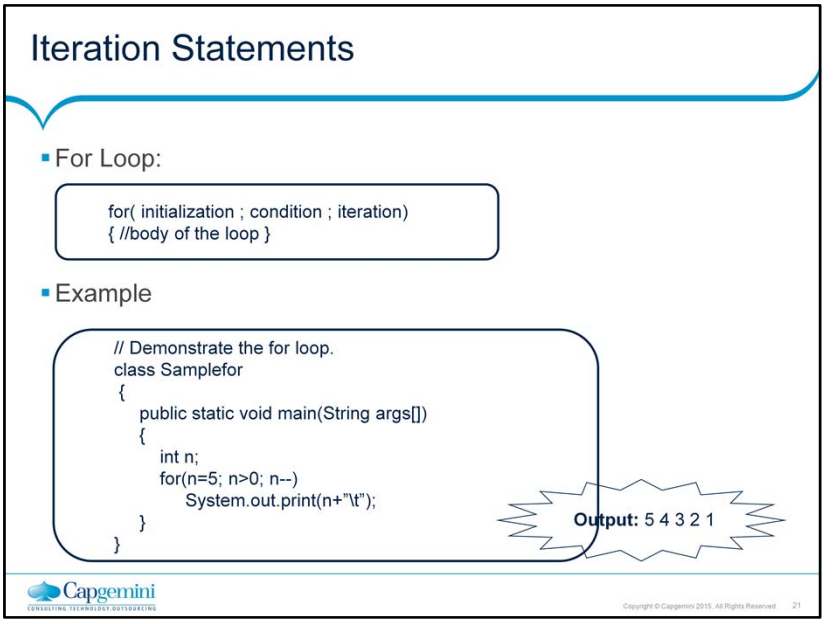
```
class Samplewhile {
        public static void main(String args[]) {
            int n = 5;
            while(n > 0) {
               System.out.print(n+"\t");
                n--;
    } } }
```

Output: 5 4 3 2 1

The while loop evaluates its conditional expression at the top of the loop. Hence, if the condition is false to begin with, the body of the loop will not execute even once.

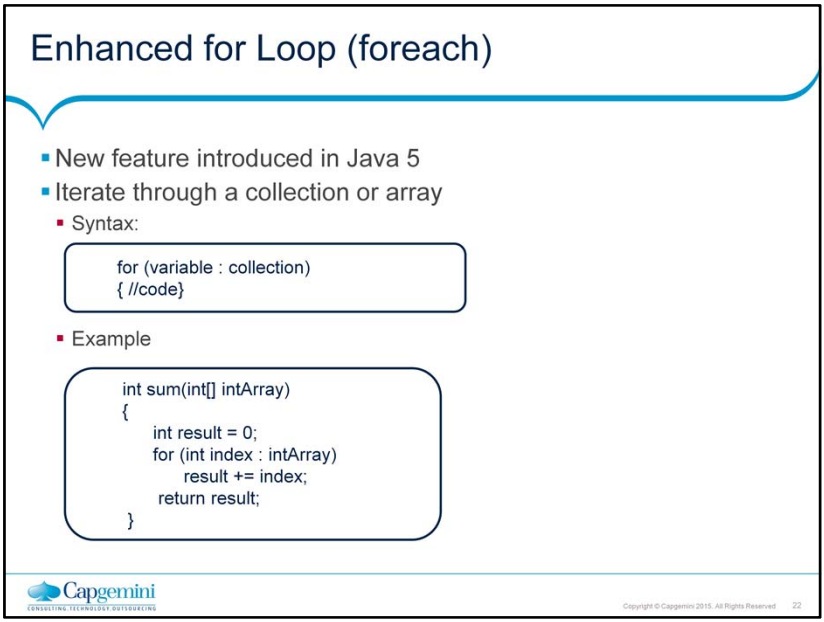
do – while loop:

This construct executes the body of a while loop at least once, even if the conditional expression is false to begin with. The termination expression is tested at the end of the loop rather than at the beginning.



for loop:

When the for loop first starts, the initialization portion of the loop is executed. Generally, this is an expression that sets the value of the loop control variable, which acts as a counter that controls the loop. The initialization expression is only executed once. Next, condition is evaluated. It usually tests the loop control variable against a target value. If this expression is true, then the body of the loop is executed, else the loop terminates. Next, the incremental portion of the loop is executed. This is usually an expression that increments or decrements the loop control variable. The loop then iterates, first evaluating the conditional expression, then executing the body of the loop, and then executing the iteration expression with each pass. This process repeats until the controlling expression is false.



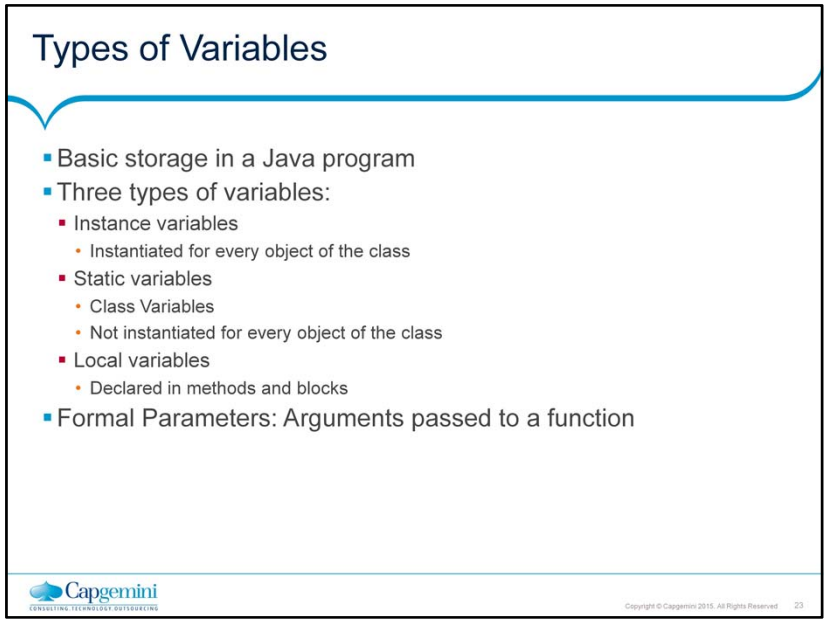
Enhanced for loop works with collections and arrays. Notice the difference between the old code where the three standard steps of initialization, conditional check and re-initialization are explicitly required to be mentioned.

Old code

```
public class OldForArray
{       public static void main(String[] args)
        {
                int[] squares = {0,1,4,9,16,25};
                for (int i=0; i< squares.length; i++)
                {
                System.out.printf("%d squared is %d.\n",i, squares[i]);
                }
}}
```

New Code

```
public class NewForArray
 {
       public static void main(String[] args)
      {
              int j = 0;
              int[] squares = {0, 1, 4, 9, 16, 25};
              for (int i : squares)
               {
                    System.out.printf("%d squared is %d.\n", j++, i);
                  }
     } }
```



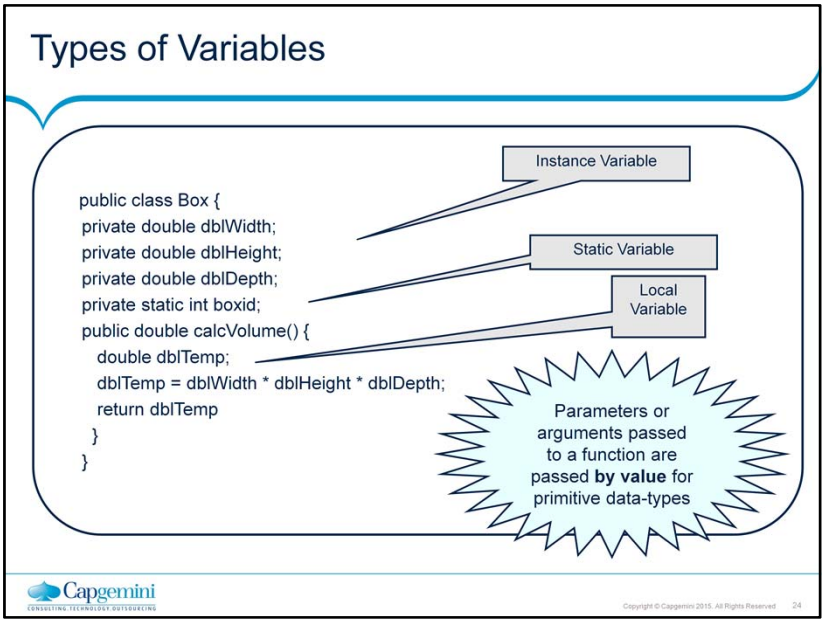
Instance variables: These are members of a class and are instantiated for every object of the class. The values of these variables at any instant constitute the state of the object.

Static variables: These are also members of a class, but these are not instantiated for any object of the class and therefore belong only to the class. We shall be covering the static modifier in later section.

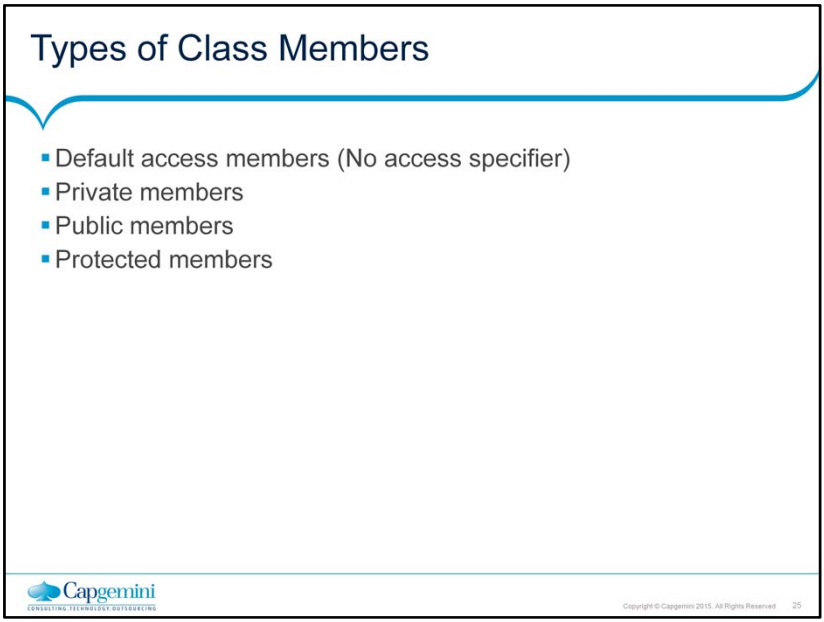
Local variables: These are declared in methods and in blocks. They are instantiated for every invocation of the method or block. In Java, local variables must be declared before they are used.

Life-cycle of the variable is controlled by the scope in which those are defined.

Formal parameters: These are the arguments passed to a function, which are similar to local variables.



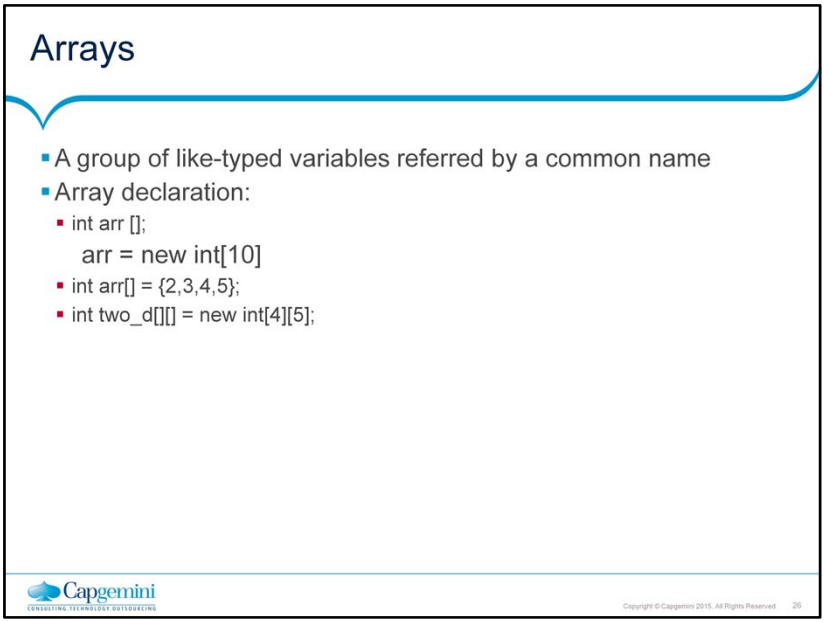
Methods and Parameter Passing:
Parameters or arguments passed to a function are passed by value for primitive data-types.
Parameters or arguments passed to a function are passed by reference for non-primitive data-types
Example: All Java objects.



public access modifier
Fields, methods and constructors declared public (least restrictive) within a public class are visible to any class in the Java program, whether these classes are in the same package or in another package.

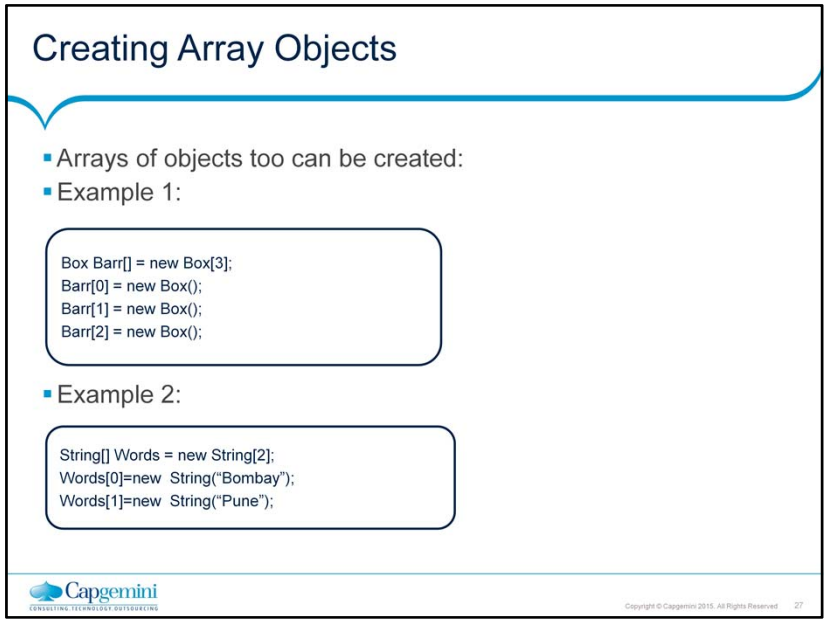
private access modifier
Fields, methods or constructors declared private (most restrictive) cannot be accessed outside an enclosing class. This modifier cannot be used for classes. It also cannot be used for fields and methods within an interface. A standard design strategy is to make all fields private and provide public getter methods for them.

protected access modifier
Fields, methods and constructors declared protected in a superclass can be accessed only by subclasses in other packages. Classes in the same package can also access protected fields, methods and constructors, even if they are not a subclass of the protected member's class. This modifier cannot be used for classes. It also cannot be used for fields and methods within an interface.
.



Syntax wherein the array is declared and initialized in the same statement:

```
strWords = { "quiet", "success", "joy", "sorrow", "java" };
```



Use new operator or directly initialize an array. When you create an array object using new, all its slots are initialized for you (0 for numeric arrays, false for boolean, '\0' for character arrays, and null for objects).
Like single dimensional arrays, we can form multidimensional arrays as well. Multidimensional arrays are considered as array of arrays in java and hence can have asymmetrical arrays. See an example next.

Demo

- Use of Arrays in Java Program

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 28

Review Question

- Question 1: In Java _____ variables are declared in methods and in blocks.
- Question 2: JRE is responsible for creating a JVM
  - Option: True / False
- Question 3: In Java fields, methods or constructors declared private cannot be accessed outside an enclosing class
  - Option: True / False

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 29

1. Local variables
2. True
3. True

Review Question

- Question 4: Java is developed by _____
- Question 5: java compiler converts the source into Byte Codes and stores it in a file having .class extension.
  - Option: True / False
- Question 6: Which variables in java are members of a class and are instantiated for every object of the class.
  - Instance variables
  - Static Variables
  - Local variables

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 30

4.James Gosling
5.True
6. Instance variables

Summary

- In this lesson, you have learnt:
  - Features of Java and its different versions
  - How Java is platform Independent
  - Difference between JRE and JDK
  - Writing, Compiling, and Executing a simple program
  - Installation and setting up Eclipse
  - Creating and managing Java projects
  - Debugging
  - The method to install Eclipse
  - Process to create a Java Project with Eclipse
  - Various useful features of Eclipse

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 31

Test Automation and tool basics -Selenium

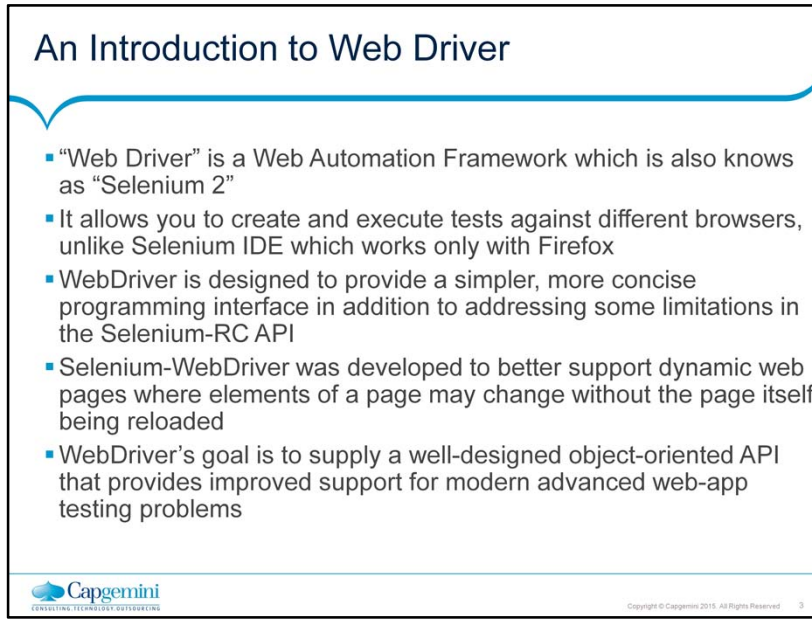
Lesson 5: Selenium 2 – Web Driver

Lesson Objectives

- To understand the following topics:
  - Introduction To Web Driver
  - Web Driver Vs Selenium RC
  - Benefits of Web Driver over Selenium RC
  - Limitations of Web Driver

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

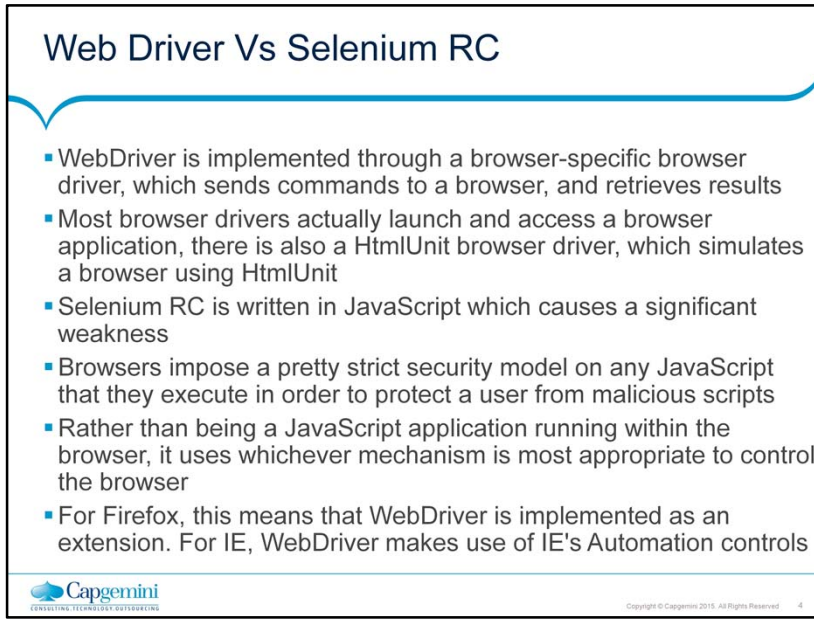
 2



An Introduction to Web Driver

Selenium 1.0 (RC) + WebDriver = Selenium 2.0

WebDriver is designed in a simpler and more concise programming interface along with addressing some limitations in the Selenium-RC API. WebDriver is a compact Object Oriented API when compared to Selenium1.0. It drives the browser much more effectively and over comes the limitations of Selenium 1.x which affected our functional test coverage, like the file upload or download, pop-ups and dialogs barrier WebDriver is the name of the key interface against which tests should be written in Java or any other programming languages supported by Selenium.

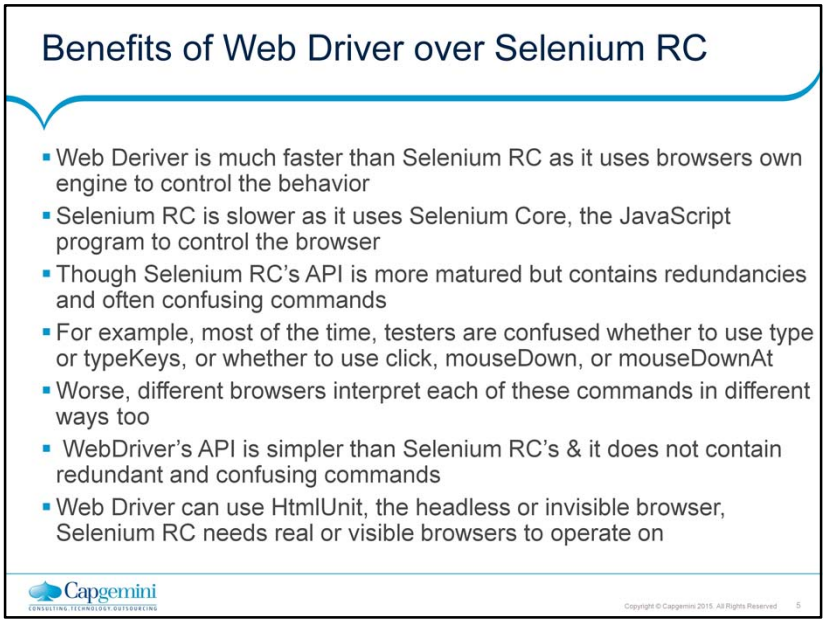


Web Driver Vs Selenium RC

Web Driver Architecture

It controls the browser from the OS level
All you need are your programming language's IDE (which contains your Selenium commands) and a browser.

Selenium Architecture
You first need to launch a separate application called Selenium Remote Control (RC) Server before you can start testing
The Selenium RC Server acts as a "middleman" between your Selenium commands and your browser
When the testing process starts, Selenium RC server injects a JavaScript program called Selenium Core into the browser
Once injected, Selenium Core will start receiving instructions relayed by the RC Server from your test program.
When the instructions are received, Selenium Core will execute them as Javascript commands.



Web Driver Vs Selenium RC

Selenium Architecture

The browser will obey the instructions of Selenium Core, and will relay its response to the RC Server.
The RC Server will receive the response of the browser and then display the results to you.
RC Server will fetch the next instruction from your test script to repeat the whole cycle

Limitation of Web Driver

- Web Driver cannot support new web browsers out of the box
  - Web Driver controls browser from OS level
  - Different web browsers communicate with the OS in a different way
  - New browsers may have different way of communicating with OS in a different way
- No built-in test result generator support
  - Selenium RC automatically generates the test result in an HTML format
  - Web Driver has no built-in provision that can help tester in generating Test Results File
  - The Tester would have to rely on your IDE's output window, or design the report yourself using the capabilities of your programming language and store it as text, html, etc

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 6

Review Question

- Question 1: What are the advantages of Web Driver?

- Question 2: What is the use of web driver?
  - To execute tests on HtmlUnit browser
  - To design test using Selenese
  - To quickly create test
  - To write and execute tests only using Firefox
- Question 3: Is Webdriver a component of the Selenium?
  - No
  - Yes

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 7

Review Question

- Question 4: Web Deriver is much faster than Selenium RC as it uses _____
- Question 5:Cross browser testing is possible using WebDriver.
  - Option: True / False
- Question 6: What is the Limitation of Web Driver?

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Summary

- In this lesson, you have learnt:
  - Introduction Web Driver
  - Advantages of Web Driver over Selenium RC
  - Limitations of Web Driver over Selenium RC

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

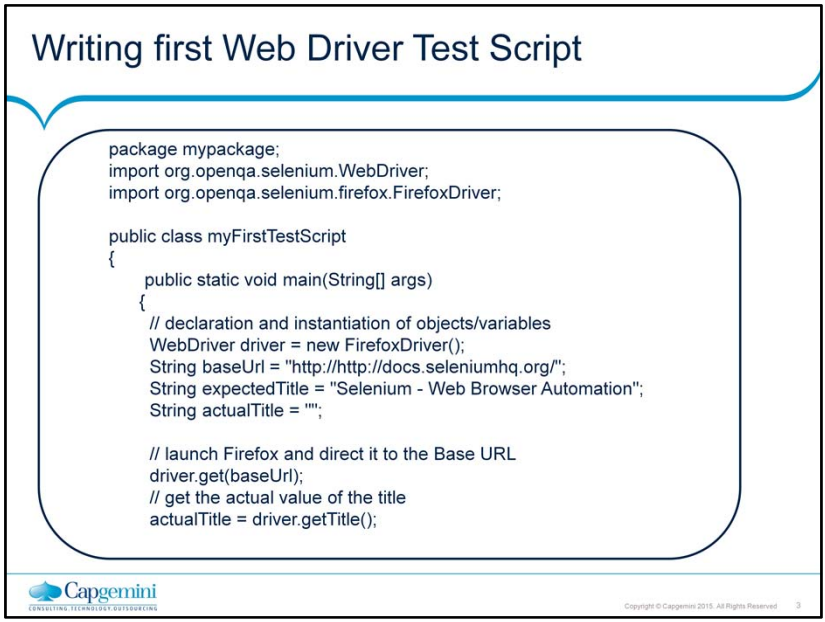
 9

Test Automation and tool basics -Selenium

Lesson 6: Testing Web Applications Using Web Driver API

Lesson Objectives

- To understand the following topics:
  - Writing first Web Driver Test
  - Locating UI Elements
  - Using sendKeys() and click()
  - Using Get Commands API
  - Using Navigate Commands API
  - Closing & Quitting Browser Window
  - Moving between Windows and Frames
  - Handling Popup Dialogs
  - Using Explicit & Implicit Wait
  - Using Explicit along with Expected Condition
  - Working with Forms using Web Driver

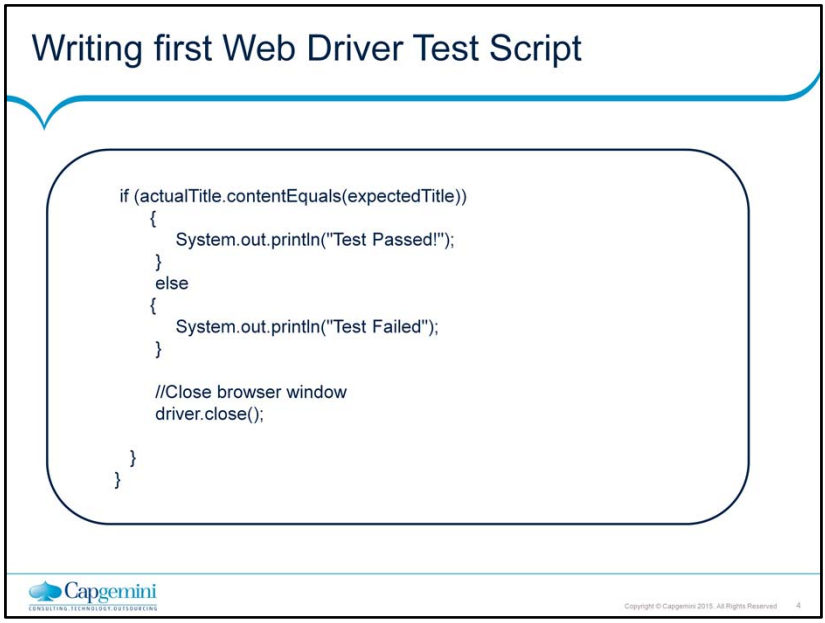


Code Explanation

Importing Packages:
Before you start writing the test, you need to import following two packages :
org.openqa.selenium.*- contains the WebDriver class needed to instantiate a new browser loaded with a specific driver.
org.openqa.selenium.firefox.FirefoxDriver – contains the FirefoxDriver class needed to instantiate a Firefox-specific driver onto the browser instantiated by the WebDriver class.

Instantiating objects and variables
A FirefoxDriver class with no parameters means that the default Firefox profile will be launched by our Java program.
WebDriver driver = new FirefoxDriver()

Starting browser session
driver.get(baseUrl)



Get the actual page title : The Web Driver class has a method, named getTitle(), this method can be used to read the title of the currently loaded web page.
actualTitle = driver.getTitle()

Comparison between actual and expected title:

```
if (actualTitle.contentEquals(expectedTitle))
    {
        System.out.println("Test Passed!");
    }
    else
    {
        System.out.println("Test Failed");
    }
```

Terminating the browser session:

```
driver.close()
```

Locating UI Elements

- Locating elements in WebDriver can be done on the WebDriver instance itself or on a WebElement
- Each of the language bindings expose a “Find Element” and “Find Elements” method
- The first returns a WebElement object otherwise it throws an exception
- The latter returns a list of WebElements, it can return an empty list if no DOM elements match the query.
- The “Find” methods take a locator or query object called “By”

Locating UI Elements

Locator	Description	Usage
ByID	Locates element using value of their "ID" attribute	HTML - ... Java - WebElement element = driver.findElement(By.id("div1"));
By.ClassName	Locates element using value of their "Class" attribute	HTML Cheddar Gouda Java - List<WebElement> cheeses = driver.findElements(By.className("cheese"));
By.Name	Locates element using value of their "Name" attribute	HTML <input name="txtUName" type="text"/> Java WebElement cheese = driver.findElement(By.name("txtUName"));

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Locating UI Elements

Locator	Description	Usage
ByLinkText	Finds a link element by the exact text it displays	HTML – cheese Java -WebElement cheese = driver.findElement(By.linkText("cheese"));
By.PartialLinkText	Find the link element with partial matching visible text.	HTML search for cheese Java - WebElement cheese = driver.findElement(By.partialLinkText("cheese"));
By.CSS	Finds elements based on the driver's underlying CSS Selector engine	findElement(By.cssSelector("input#email"))

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 7

Locating UI Elements

Locator	Description	Usage
By.tagName	locates elements by their tag name	HTML - ... Java - findElement(By.tagName("div"))
By.xpath	locates elements via Xpath	findElement(By.xpath("//html/body/div/table/tbody/tr/td[2]/table/tbody/tr[4]/td/table/tbody/tr/td[2]/table/tbody/tr[2]/td[3]/form/table/tbody/tr[5]"))

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Demo

- Demo on Locating UI elements using Web Driver API

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 9

Using sendKeys() and click()

- Example of sendKeys()

```
WebElement myElement = driver.findElement(By.id("Username"));
myElement.sendKeys("SeleniumUsers");
```

- Clicking on an Element

```
driver.findElement(By.name("Click Me")).click();
```

- It does not take any parameter/argument
- The method automatically waits for a new page to load if applicable
- The element to be clicked-on, must be visible

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 10

Using Get Commands API

Command	Description
Get()	1. It automatically opens a new browser window and fetches the page that you specify inside its parentheses 2. The parameter must be a string
getTitle()	1. Fetches the title of the current page 2. Return null if the current page has no title 3. Needs no parameters
getPageSource()	1. Return the source code of the page as a string value 2. Needs no parameters
getCurrentUrl()	1. Gets the url of the current page loaded in the browser 2. Needs no parameters
getText()	1. Fetches the inner text of the element that you specify

Demo

- Demo on Get Commands

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Using Navigate Commands API

Command	Description
navigate().to()	1. Behaves exactly same as get() method 2. It opens a new browser window and loads the page that you specify inside its parentheses
navigate().refresh()	1. Refreshes current loaded page in the browser 2. Needs no parameters
navigate().back()	1. Takes you back by one page on the browsers history 2. Needs no parameters
navigate().forward()	1. Takes you forward by one page on the browsers history 2. Needs no parameters

Demo

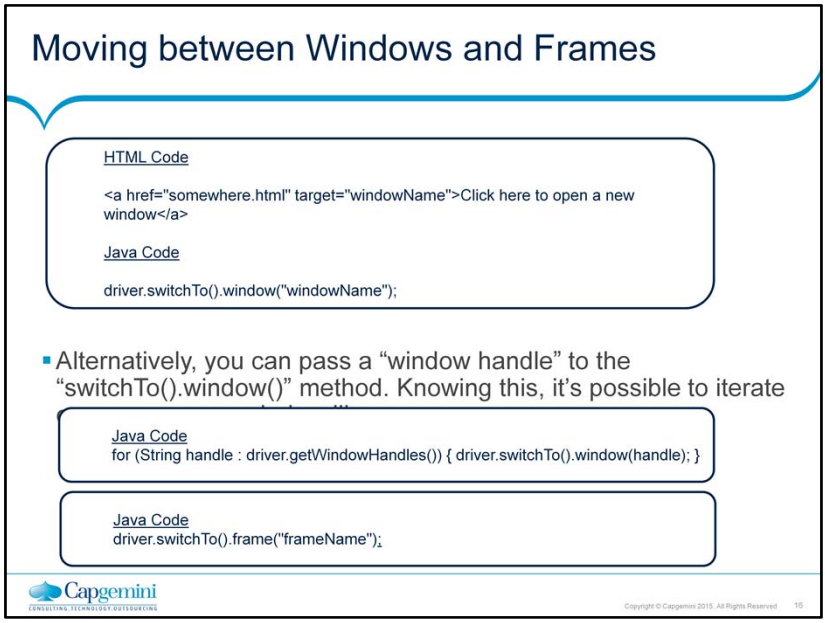
- Demo on Navigate Commands

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 14

Closing & Quitting Browser Window

Command	Description
close()	1. It closes the browser window which is being opened currently 2. Needs no parameters
quit()	1. It closes all windows that web drive has opened 2. Needs no parameters


Moving between Windows and Frames
HTML Code
Click here to open a new window
Java Code
driver.switchTo().window("windowName");
Alternatively, you can pass a “window handle” to the “switchTo().window()” method. Knowing this, it’s possible to iterate
Java Code
for (String handle : driver.getWindowHandles()) { driver.switchTo().window(handle); }
Java Code
driver.switchTo().frame("frameName");
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
16

Demo

- Demo on moving between Window or frames

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 17

Handling Popup Dialogs

- Starting with Selenium 2.0 beta 1, there is built in support for handling popup dialog boxes
- After you've triggered an action that opens a popup, you can access the alert with the following:

```
Java Code
Alert alert = driver.switchTo().alert();
```

- This will return the currently open alert object
- With this object you can now accept, dismiss, read its contents or even type into a prompt
- This interface works equally well on alerts, confirms, and prompts

Demo

- Demo on handling popup windows

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 19

Using Explicit & Implicit Wait

- Waiting is having the automated task execution elapse a certain amount of time before continuing with the next step
- Explicit Waits
  - An explicit waits is code you define to wait for a certain condition to occur before proceeding further in the code
  - There are some convenience methods provided that help you write code that will wait only as long as required
  - WebDriverWait in combination with ExpectedCondition is one way this can be accomplished
  - Import following two packages
    - import org.openqa.selenium.support.ui.ExpectedConditions;
    - import org.openqa.selenium.support.ui.WebDriverWait;

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 20

Using Explicit along with Expected Condition

- The ExpectedConditions class offers a wider set of conditions that you can use in conjunction with WebDriverWait's until() method

```
WebDriver driver = new FirefoxDriver();
WebDriverWait myWait = new WebDriverWait(driver,10);
```

```
myWait.until(ExpectedConditions.visibilityOfElementLocated(By.id("username")));
drive.findElement(By.id("username")).sendKeys("SeleniumUser");
```

- The above code will put an explicit wait on the "username" element before we type the text "SeleniumUser" into it

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 21

Using Explicit along with Expected Condition

- alertIsPresent – Waits until an alert box is visible

```
If(myWait.until(ExpectedConditions.alertIsPresent()) != null)
{
    System.out.println("Alert box is available");
}
```

- elementToBeClickable - Waits until an element is visible and, at the same time, enabled

```
WebElement txtQualification =
myWait.until(ExpectedConditions.elementToBeClickable(By.id("Qualificatio
n")));
```

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Using Explicit & Implicit Wait

- Implicit Waits
  - It is easy to code the Implicit wait compare to coding the explicit wait
  - The right place for declaring implicit wait for the test is in the instantiation part of the code
  - Import following package to declare implicit wait in the test
    - import java.util.concurrent.TimeUnit;
- Example : The below given code will set the 10 seconds as the default wait for the test

```
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
```

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 23

Demo

- Demo on using Explicit & Implicit Wait

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

Working with Forms using Web Driver

Element	Command	Example
InputBox	sendKeys() clear()	driver.findElement(By.name("username")).sendKeys("SeleniumUser");

RadioButton, CheckBox	click()	driver.findElement(By.cssSelector("input[value='Male']")).click();
		WebElement chkHobbies = driver.findElement(By.id("chkMusic"));
		chkHobbies.click();

Links	click()	Driver.findElement(By.linkText("Click Me")).click();
Drop-Down Box	Select	select drpCountry = new Select(driver.findElement(By.name("Country")));
Submit Form	submit()	The submit() method is used to submit a form. This is an alternative to clicking the form's submit button. You can use submit() on any element within the form, not just on the submit button itself. driver.findElement(By.name("password")).submit();

Working with Forms using Web Driver – DropDown Box

Command	Description	Example
selectByVisibleText() and deselectByVisibleText ()	Selects/deselects the option that displays the text matching the parameter.	drpFruit.selectByVisibleText("Mango");
selectByValue() and deselectByValue()	Selects/deselects the option whose "value" attribute matches the specified parameter.	drpFruit.selectByValue("123");
selectByIndex()	Selects/deselects the option at the given index.	drpFruit.selectByIndex(0);

isMultiple()	Returns TRUE if the drop-down element allows multiple selections at a time; FALSE if otherwise.	If(drpCountry.isMultiple())
		{
		//Do something
		}

deselectAll()	Clears all selected entries. This is only valid when the drop-down element supports multiple selections.	drpContry.deselectAll();

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 26

Demo

- Demo on Working with forms using Web Driver API

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 27

Review Question

- Question 1: _____ gets the url of the current page loaded in the browser
- Question 2: Webdriver close() method closes all windows that web drive has opened
  - Option: True / False
- Question 3: Select the command which is used to compare actual title Vs Expected title value
  - verifyTitle
  - assertTitle
  - VerifiedTitle
  - checktTitle

Review Question

- Question 4: _____ Webdriver method retrieves runtime title of web page

- Question 5: An explicit wait is code you define to wait for a certain condition to occur before proceeding further in the code
  - Option: True / False
- Question 6: Which command can be used to enter values in Textbox in Selenium Webdriver?
  - sendKeys()
  - sendKey()
  - sendKeys
  - type

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 29

Summary

- In this lesson, you have learnt:
  - Creating the Test Script using Web Driver and Java programming language
  - Understand how to locate UI Elements on web page
  - Using various Web Driver API commands to work with different aspects of Web Page testing
  - Working with Forms & submitting form

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 30

Test Automation and tool basics – Selenium

Lab BookVersion 1.1

Document Revision History

Date	Revision No.	Author	Summary of Changes
August 2013	1.0	Shilpa Bhosle	New course creation
May, 2016	1.1	Dayanand P.	Post-integration material alignment

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Table of Contents

Overview

This lab book is a guided tour for learning Automated Web Testing with Selenium tool. It comprises of lab assignments to be solved by the participants.

Setup Checklist for Selenium

Here is what is expected on your machine in order for the lab to work.

Minimum System Requirements

- Microsoft Windows 7 SP1
- Memory: 1 GB of RAM

Minimum Software Requirements

- Mozilla Firefox with Firebug – Latest version
- Internet Explorer 9.0 With browser driver
- Google Chrome latest version with browser driver
- Java Runtime Environment - 1.6 or above
- Java Development kit - 1.6 or Latest
- Selenium IDE – Latest version
- Eclipse editor
- Jar files required for selenium webdriver - v2.24 or above – Visit http://www.seleniumhq.org/download/

Instructions

- Create a directory by your name in drive <drive>. In this directory, create a subdirectory Selenium_Lab_Assign. For each lab exercise create a directory as lab <lab number>.

Lab 1. Creating Test Script using Selenium IDE

Goals	• Understand the process of creating recorded test using Selenium IDE • Working with Selenium commands
Time	45 minutes

1.1: Follow the given instructions and create the recorded TestScriptusing Selenium IDE.

1. Open Firefox web browser
2. Open Enquiry_page.html
3. Enter First Name
4. Enter Last Name
5. Select Gender
6. Enter Mobile Number
7. Enter Email
8. Enter Address
9. Select Education
10. Select Course Interested
11. Enter Query
12. click on Submit Button

1.2 Play the test case and verify it follows each and every step executedby you while creating a Test Script using Selenium IDE.

1.3 Save the Test Script

1.4 Reload the above Test Script in the Selenium IDE and insert useful comments in the script for the following steps.

1. Enter First Name
2. Enter Last Name
3. Select the computer course in which you are interested
4. Enter query if any

1.5 Do the given below changes in the script & observe the result of test execution

1. Modify the above test script to validate the title of the web page

   Hint: Use assertTitle Selenium command to perform the abovetask.

2. Try changing the value for assertTitle command in the Selenium IDE to “User Enquiry” and executethe same, you should see the error in test execution

3. Store the title of the web page in title variable and print the same

   Hint: Use echo and storeTitle command to perform the above task

4. Modify the above script to verify the test ‘Shree Computer Academy’ displayed onForm.

Hint: Use verifyTextPresent to accomplish the same

Lab 2. Working with Element Locators and Storecommandsin Selenium IDE

Goals	• Understanding the usage of various Element Locators available in Selenium IDE to identify the correct element on the web page to operate on. • Understand the usage of Store commands in Selenium IDE • Understanding & implementing pattern matching in web testing • Learn how to assert alert message in the script
Time	60 minutes

2.1: Execute EventRegistrationPage.html page in Firefox and tryto locatevarious UI elements of the web page using Element locatorsavailable in Selenium IDE.

Hint: Go through the source code for the above given page thoroughly to understand the page design .

You can use following locators to complete the above given task.

1. ID
2. Name
3. Link Text
4. CSS Selector
5. Tag and ID
6. Tag and class
7. Tag and attribute
8. Tag, class, and attribute
9. Inner text
10. getElementById
11. getElementsByName

2.3 Reopen the test script which you have created in Lab 1.1 &modify thesame so that it can take care of following requirements.

1. Verify that the title of the web page should begin with “User Enquiry Form”.

2. Verify that the Email id follows the general rule of being valid Email ID
3. Hint: Use verifyValue Command along with regular expression to validate the email id
4. Also, modify the script to assert the alert message by leaving the email field empty.

2.4 Load LoginPage.html web page in Firefox browser and complete thefollowing tasks.

1. Record the script in Selenium IDE for the following:
    a. Enter UserID – SeleniumUser
    b. Enter Password - selenium123
    c. Click on Login Button

2. Modify the recorded script in such a way that it should not only verify that the Username & Password textboxes are present on the form but also should print the status by displaying “True” or “False”, depending upon presence of these elements on the Selenium IDE.
3. Now, modify the recorded script in a such a way that, it should store and display the username & password entered by the user in a Selenium IDE
4. After successful login, user will be navigated to Welcome page. You need to modify the script to verify that the page has text “Welcome SeleniumUser” & store the same in the variable and output the value in the Selenium IDE.

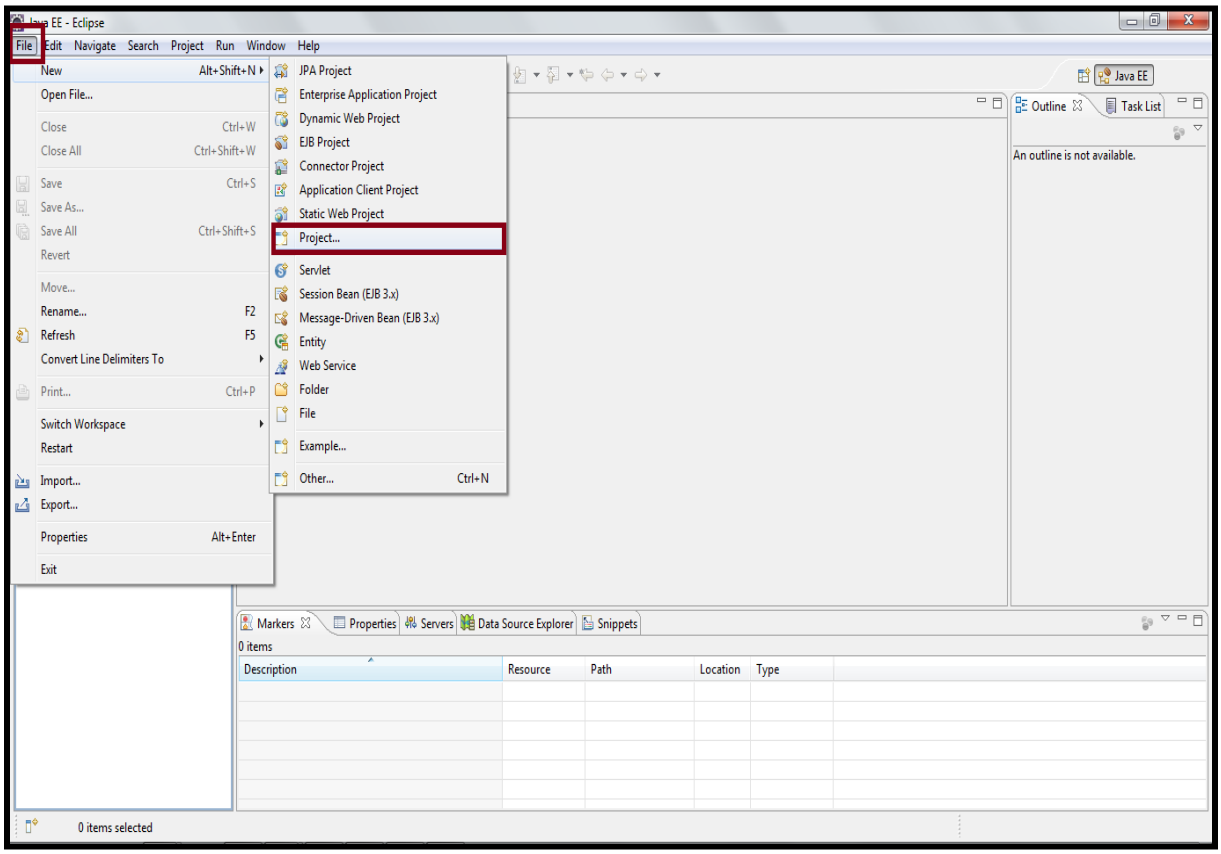
Lab 3. Working with Eclipse

Goals	• Creating your first Java project in Eclipse IDE • Create your first java program & execute the same
Time	30 minutes

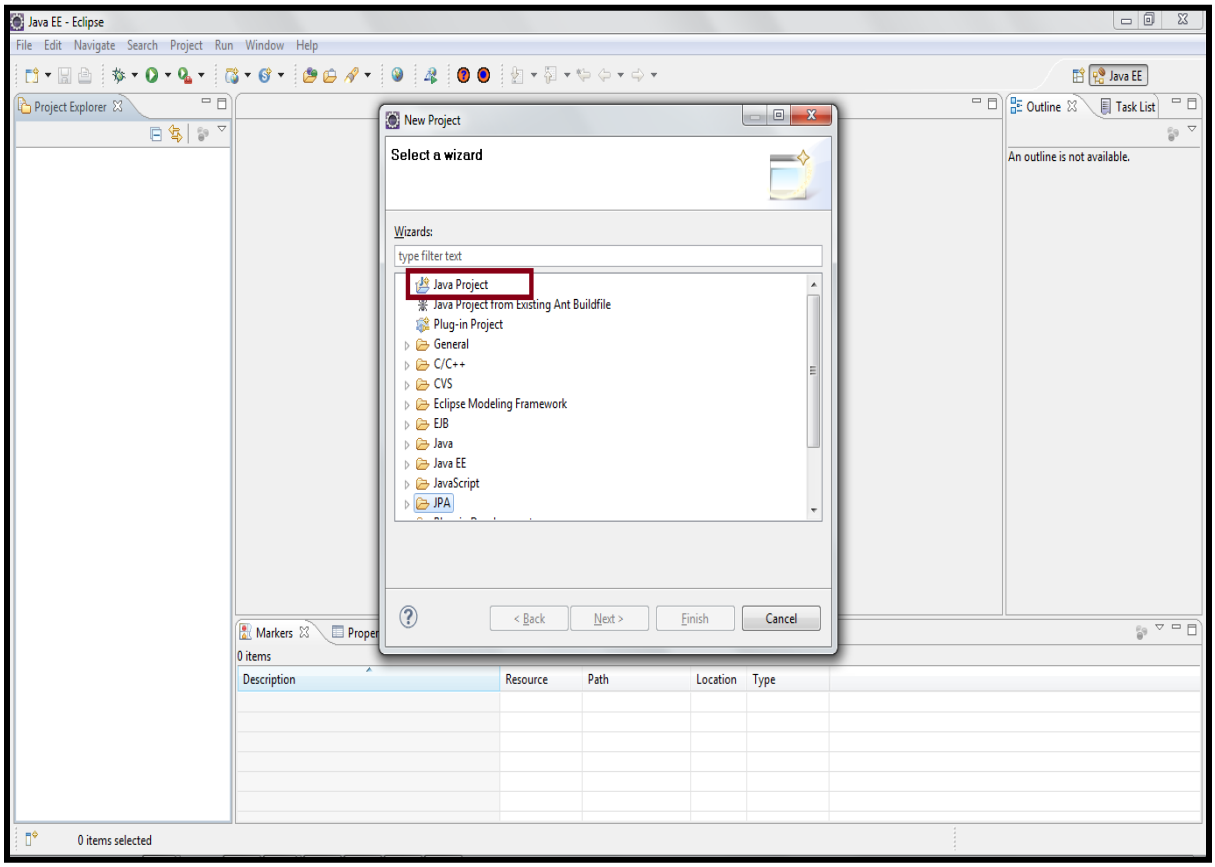
Follow the following steps to create your first Java application in Eclipse

Step 1: Start Eclipse IDE

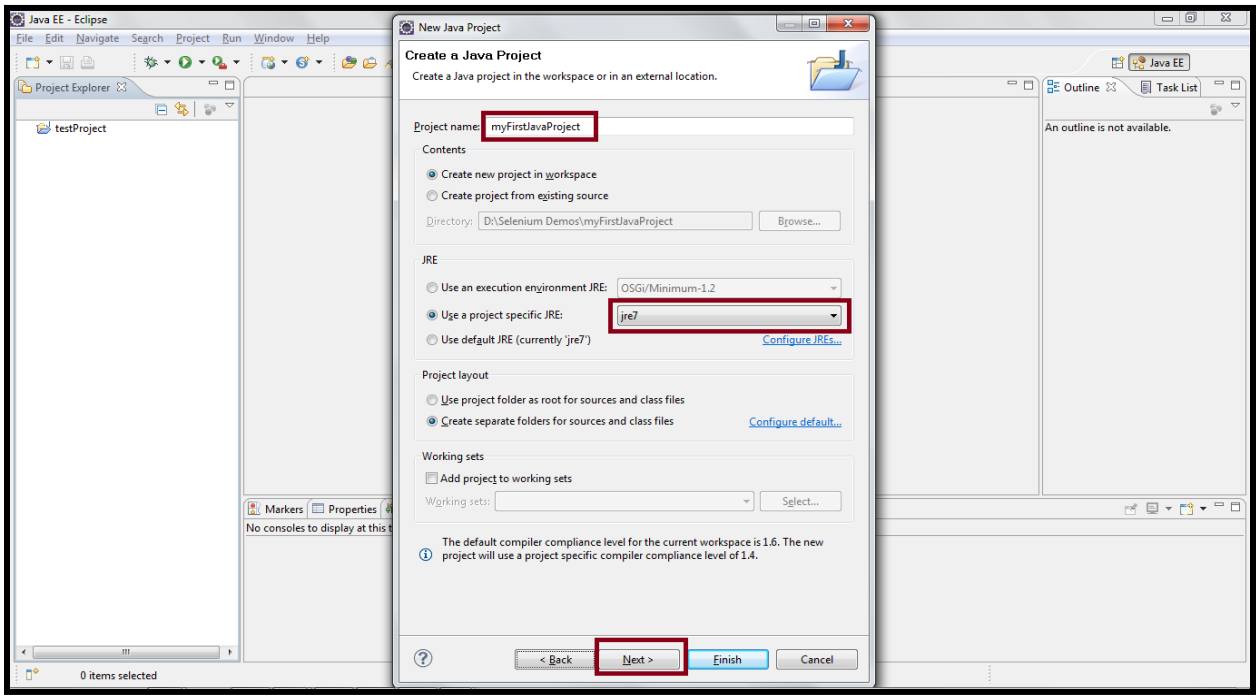
Step 2: Select **File→New→Project →Java project**.



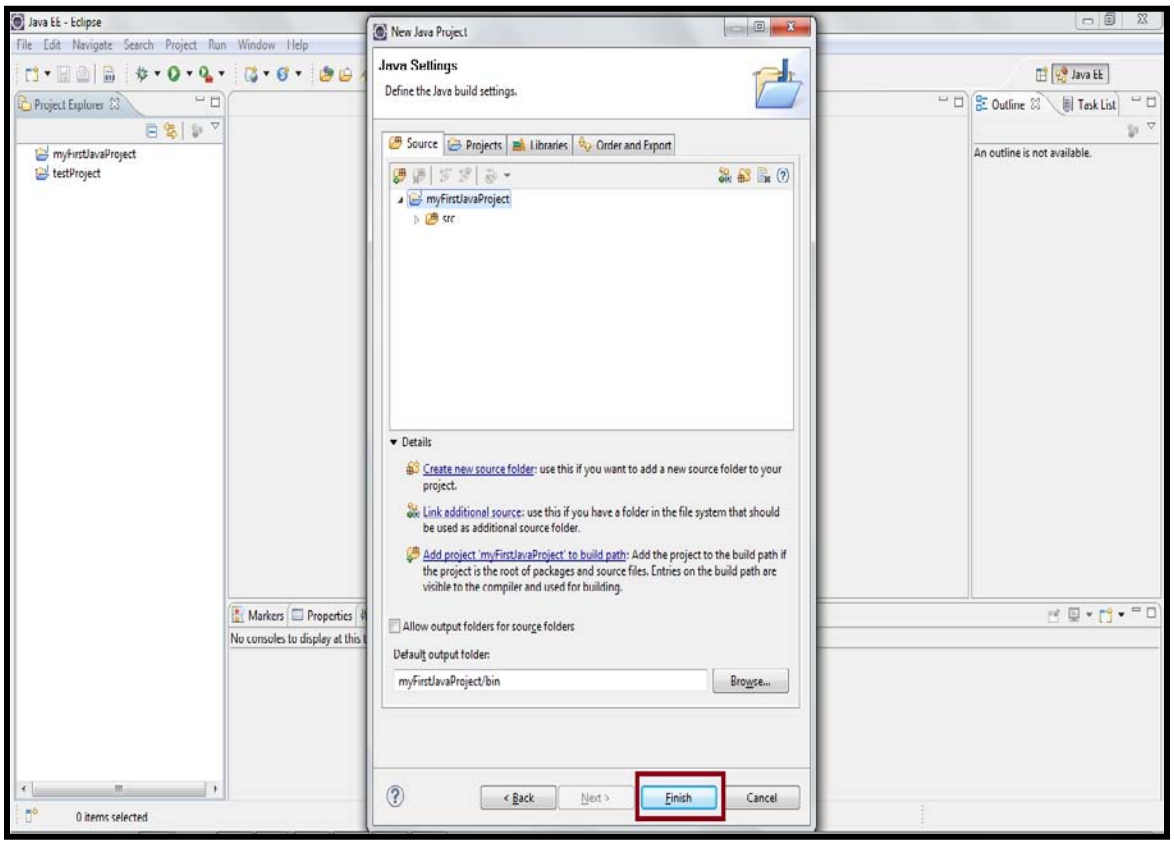
Step 3: Select **Java project**from the **New Project** window.



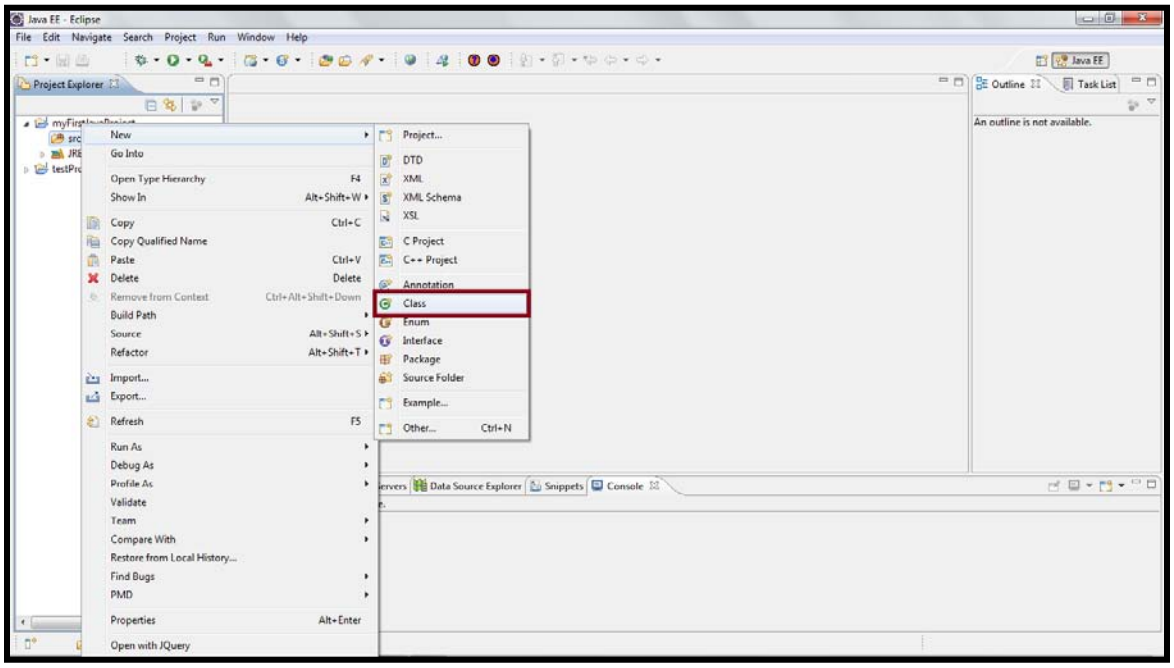
Step 4: Make sure your **"New Java Project"** windowlooks like the given screen shot by making necessary selections in the window and click on next.



Step 5: Click on **"Finish"**.

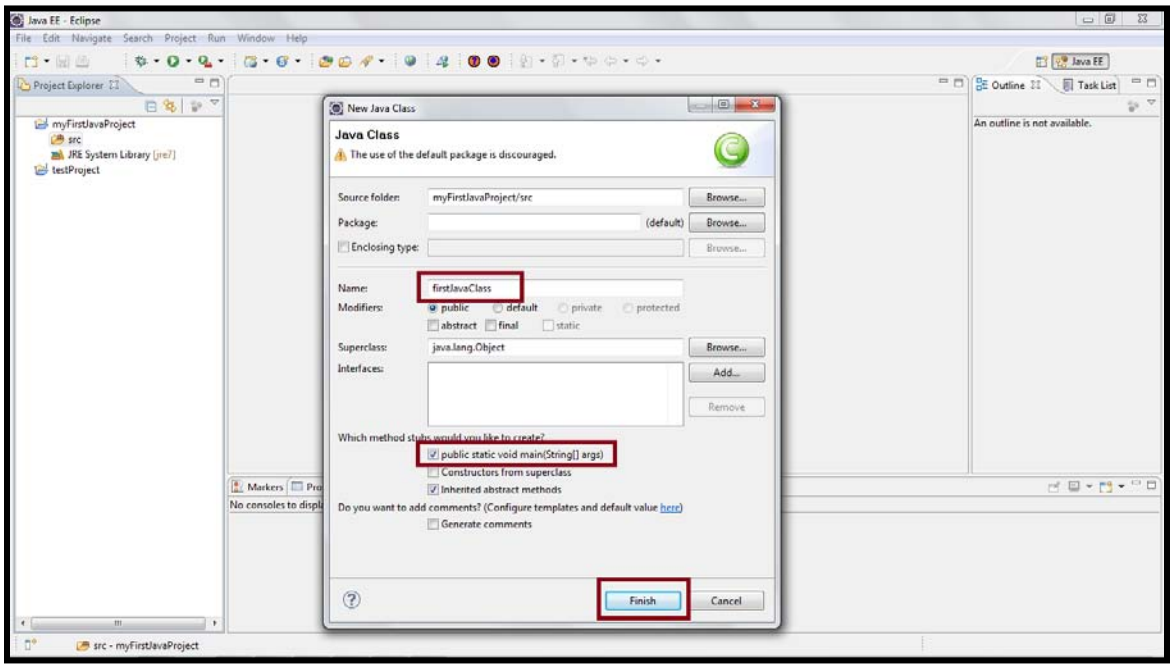


Step 6: Add **class** to **myFirstJavaProject.**



Step 7: Make sure that **New Java Class** window looks like as given in the following

screen shot and click on **Finish**.



Step 7: You are now ready to add functionality to your java project.

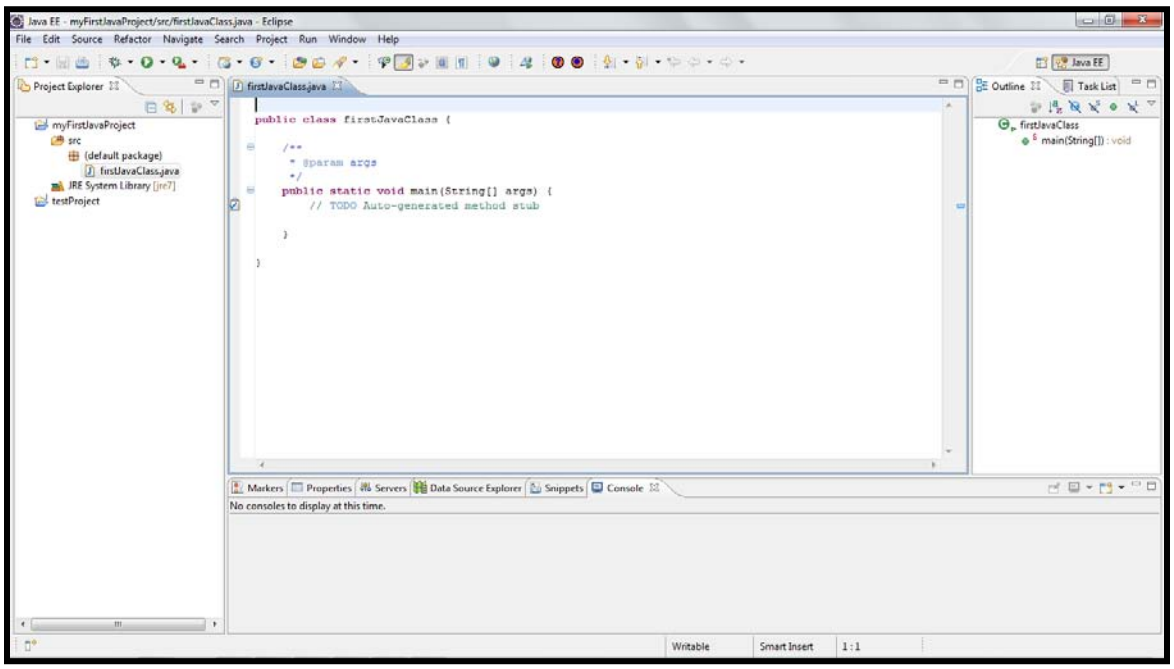
This will open firstJavaClass**.java** in the editor, with ready skeleton for the class, **main()** method, and necessary **javadoc** comments.

Step 8: Add following line of code in the class

System.out.println("Hello World");

Step 9: To run this class, select **Run** from toolbar, or select **Run As →Java Application**& check the output in the **Console Window.**

Output: Hello World



Lab 4. Working with Classes, Objects, Arrays in Java

Goals	• At the end of this lab session, you will be able to: o Create Classes and Objects o Working with Conditional Statements& loops o Working with Array
Time	120 minutes

4.1: Write a program to create a Date class and UseDate Classwhichinstantiates the Date class.

Solution:

Step 1: Type the following code in a text editor and editor and save the file with the name "**UseDate.java**".

```
class Date {
        intintDay, intMonth, intYear;

        Date(intintDay, intintMonth, intintYear)  // Constructor
        {
        this.intDay = intDay;
this.intMonth = intMonth;
this.intYear = intYear;
        }

        voidsetDay(intintDay)       // setter method for day
        {
                this.intDay = intDay;
        }

        intgetDay( )                // getter method for month
        {
                return  this.intDay;
        }

        voidsetMonth(intintMonth)
        {
```



```
                this.intMonth = intMonth;
        }

        intgetMonth( )
        {
                return  this.intMonth;
        }

        voidsetYear(intintYear)
        {
                this.intYear=intYear;
        }

        intgetYear( )
        {
                return  this.intYear;
        }


        public String toString() //converts date obj to string.
        {
                return    “Date is “+intDay+”/”+intMonth+”/”+intYear;
        }


} // Date class

classUseDate
{
        public static void main(String[] args)
        {
                Date d = new Date(8,5,2013);
                System.out.println(d);  //invokes toString() method
        }
} //class ends
```


Example 1: UseDate.java

Step 2: Execute the class**.** You should get the output as follows:

Output: Date is 8/5/2013

<<TO DO>>

Assignment 1: Implement the following in the above date class

a. Write methods to incorporate validation checks. (For example: day should be between 1 & 31, Year must be less than 1984).

b. Call the validate methods in main. If the validation fails, restore default date.

Assignment 2: Create a class **Staff** that has members as specified below. Implement the **setter** and **getter** methods for the data members. Make the changes as required and save the file as **Staff.java**.

```
public class Staff
{
        privateintstaffCode;       // Employee Code
        private String staffName;          // Name
        private String designation;        // Designation
        privateint age;              // Age

        // Date of Birth (Create an object of the Date  class given in the Example 4.1
        private Date dateOfBirth;

        public Staff()     // Constructor
        {
                // To Do: Initialize data members
        }

        public Staff(intstaffCode, String staffName,String designation, int age, Date
DOB)
        {
                // To Do: Initialize data members based on the parameters
```



```
}

// setter methods
void setStaffCode(int staffCode)
{                       // To Do: Setter method for staff code
}

public void setStaffName(String staffName)
{
        // To Do: Setter method for staffName
}

voidsetDesignation(String  designation)
{
        // To Do: Setter method for designation
}

voidsetAge(int age)
{
        // To Do: Setter method for age
}

voidsetDateOfBirth(Date birthdate)
{
        // To Do: Setter method for date of birth
}

// getter methods
intgetStaffCode()
{
        // To Do: getter method for staffCode
}

String getStaffName()
{
```



```
                // To Do: getter method for staffName
        }

        String getDesignation()
        {
                // To Do: getter method for designation
        }

        intgetAge()
        {
                // To Do: getter method for Age
        }

        Date  getDateOfBirth()
        {
                // To Do: getter method for Salary
        }

        // prints the staff details on the screen
        public void displayDetails()
        {
                System.out.println("The staff code  is "+ staffCode);
                // To Do: Print the other members in the same format
        }

} // end of class Employee

Class  StaffApplication
{
        public static void main(String[] args)
        {
                Staff staff = new Staff( );
                staff.displayDetails( );
        }
}
```


Example 2: Sample code

Assignment 3: Write a program in Java for calculating average value of array elements.

Assignment 4: Write a program in Java for generating Fibonacci series

Assignment 5: Create a class **ArrayDemo**, which behaves like a wrapper around an integer array. The class should have methods to create array, add elements into it, display contents of array, search for an element in the array. Minimum size of the array should be 10.

```
classArrayDemo{

int contents []     // declare array.
ArrayDemo (int){
        // Create array of size <int>
}

voidpopulateContents(){
        // Populate the array
}

voidshowContents(){
        // Display contents of array.
}

intsearchElement(intsearchElement){
        // search for element; if found, return its index location, else return –1.
}
}

Class  UseArray
{
        public static void main(String[] args)
        {
                ArrayDemoarrayDemo = new ArrayDemo( );
```



```
                // call to the methods in the class.
        }
}
```

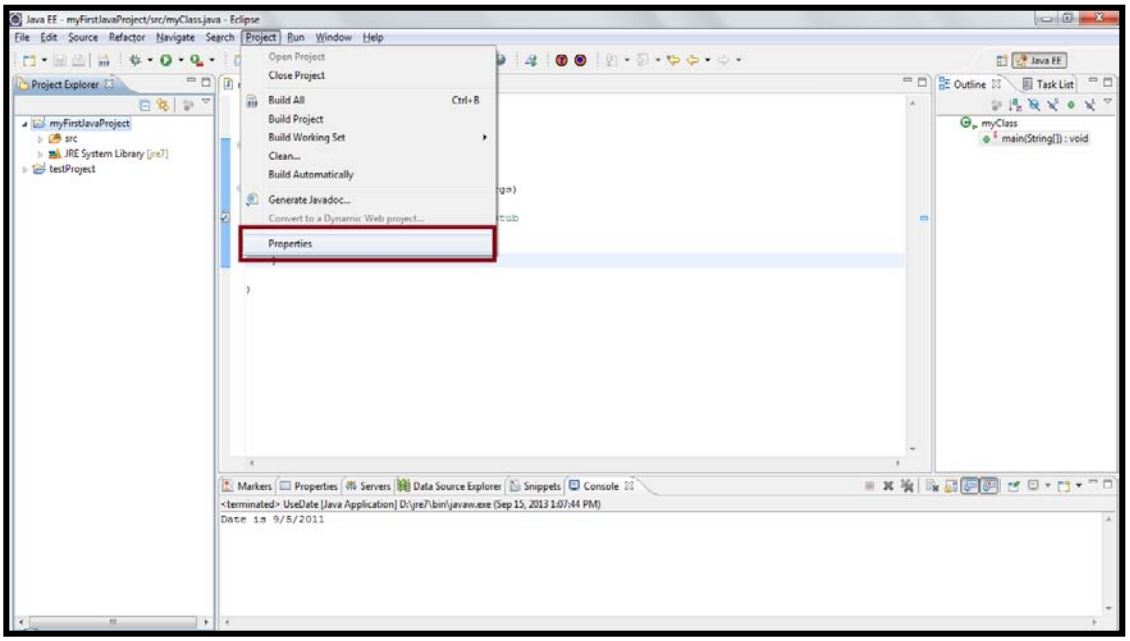

Lab 5. Using Web Driver for automating web testing

Goals	• At the end of this lab session, you will be able to: o Configuring Eclipse to work with Selenium Web Driver o Working with moving/switching between frames in Web Application o Create automated test script using Web Driver o Handling Forms using Web Driver
Time	120 minutes

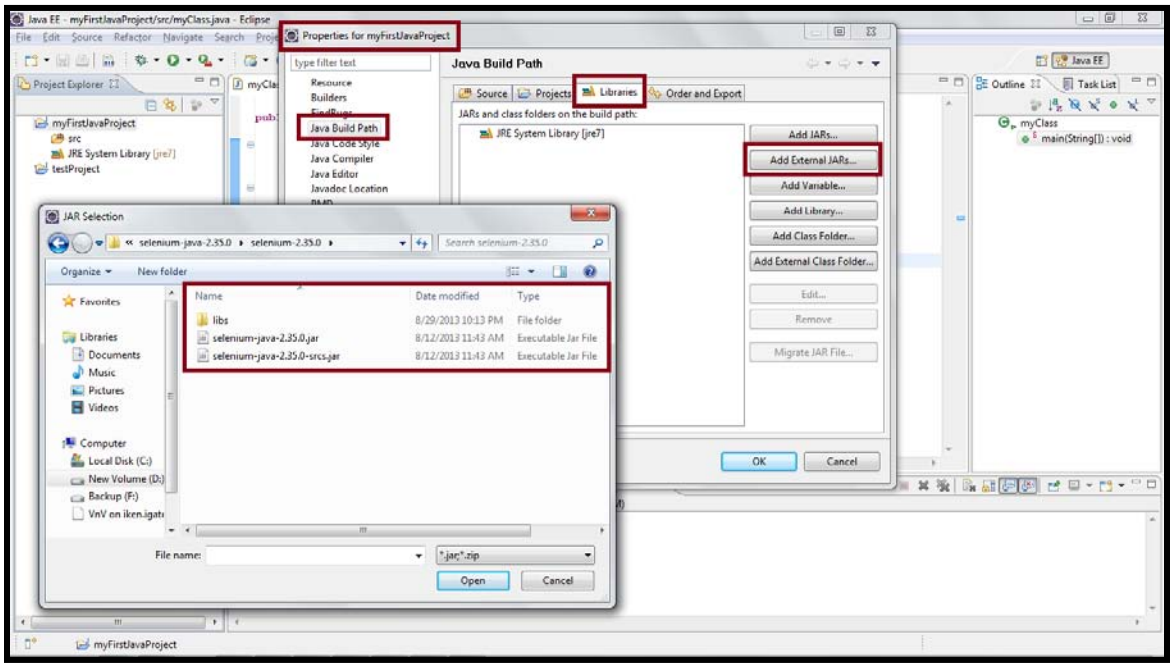
5.1: Configure Eclipse IDE to work with Selenium Web Driver.

Follow steps given below.

Step 1: Create the Java project & add java class in Eclipse IDE
Step 2: Select the **project**& click on **Project menu →Properties**



Step 3: Make the necessary selections as highlighted in the following given screenshot. This will add all necessary libraries to java project. Add all the binaries from **lib**folder as well. Once done, you are ready tocreate your first automated test script with Selenium Web Driver.



5.2 Create an automated test script using Selenium Web Driver for handing moving between different frames in your web application. Use frames.html & all other html files provided as Lab Files to complete this assignment.

Note : You can use any of the link to switch from main page to aparticular frame within web application. Links are “Computer” and“Electronic”

5.3 Load CreateAccount.html web page in the browser. Create an automated script using Web Driver to submit the page successfully.

