MONASH University

MONASH INFORMATION TECHNOLOGY

FIT5192 Lecture 1: Introduction to Module One - Enterprise Applications Development for the Web Using Java EE Technologies

GROUP OF EIGHT AUSTRALIA

Outline

- Topics Overview
- Brief overview of Enterprise Architecture
- Java Revision

Textbooks

- **The Java EE 7 Tutorial**
  - The official tutorial provided by Oracle
  - Provides a good break down of the features present in the Java EE 7 platform
- **Beginning Java EE 7**
  - Very easy to read and understand

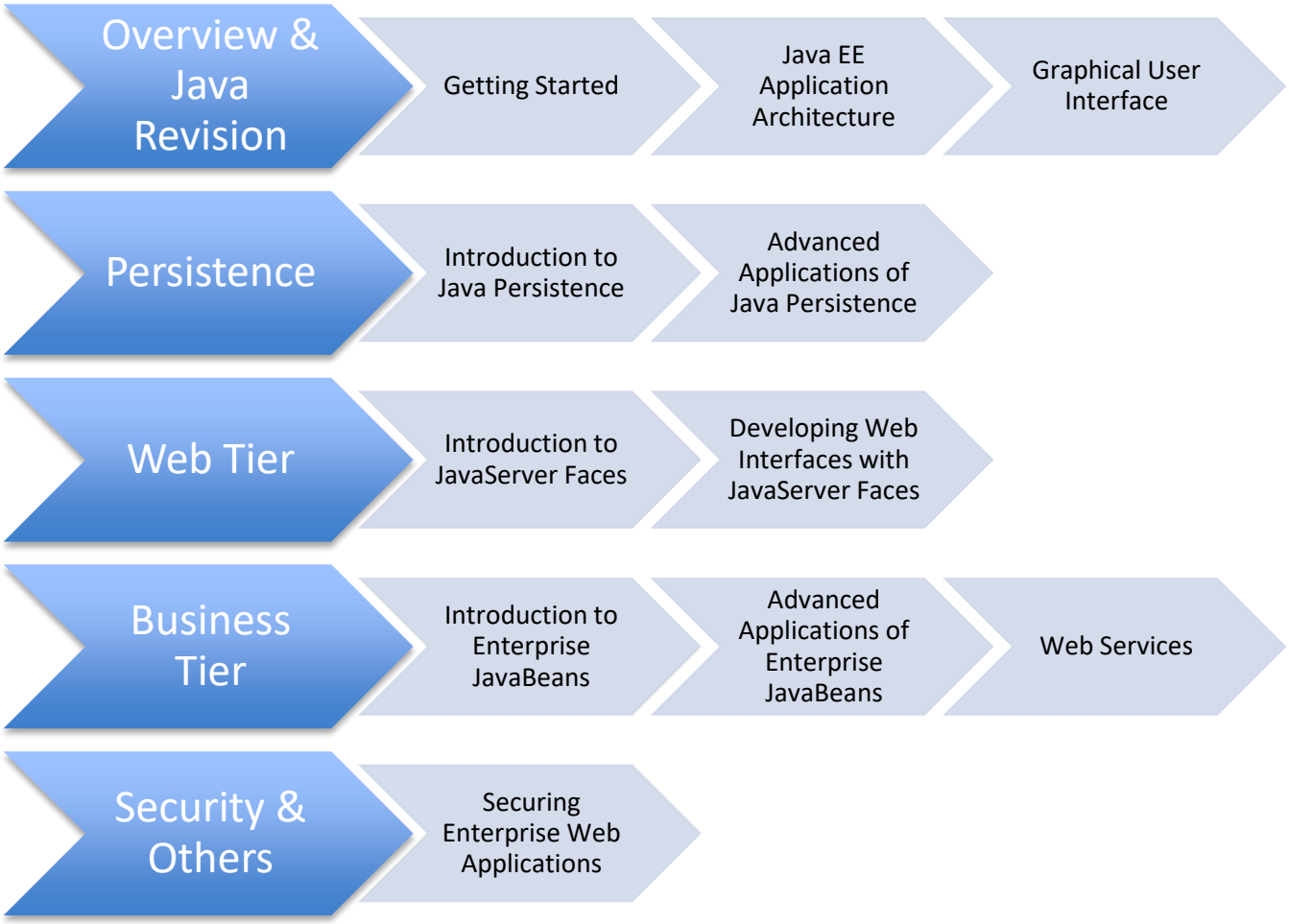
Development Kit and IDE

We will be using:

- Java EE 7
- JDK 8
- Netbeans IDE 8.1+
- GlassFish Server Open Source Edition 4.1+

Topics Overview

MONASH University



Enterprise Software

- **What is an Enterprise?**
  - An organization of individuals or entities that work together to achieve a set of common goals.
- **Often have a similar set of business activities:**
  - Information sharing
  - Information processing
  - Asset management
  - Resource planning
  - Payroll and so on…
- **Enterprise software relates to all the software involved in supporting such business activities.**

Enterprise Web Architecture

- Often involves many different distributed systems working together to satisfy the business requirements.
- Individual systems are often linked together via the web using web services
- Modern approach to Enterprise Architecture is a “tiered” approach
  - Separating the presentation (user interface), the business logic and data persistence of an application into individual layers.

Java is both a programming language and a platform

- Java programming language is a high-level Object Oriented language
- Java platform is an environment in which applications written in Java programming language run

- **There are 4 platforms of Java programming language:**
  - Java Platform, Standard Edition (Java SE)
  - Java Platform, Enterprise Edition (Java EE)
  - Java Platform, Micro Edition (Java ME)
  - Java FX
- **All Java platforms consist of:**
  - Java Virtual Machine (JVM): A program this is designed for a particular hardware and software platform that runs Java applications
  - API: A collection of software components that can be used to create other software components or applications.

Java SE and Java EE

- **Java SE**
  - Provides the core functionalities of the Java programming language (e.g. basic types, objects, high level classes that are used for networking, security, database access, GUI and etc.)
- **Java EE**
  - Extension of Java SE
  - Provides API and runtime environment for large-scale and multi-tiered enterprise application

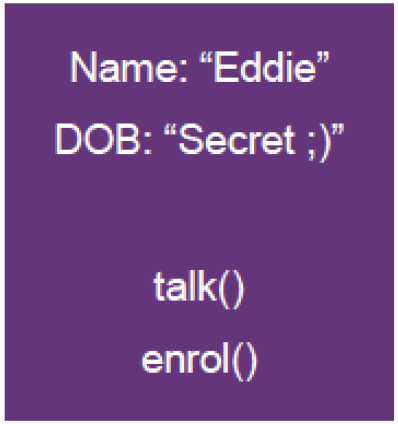
Java Revision

- Provides a quick revision of some object-oriented programming and Java concepts.
  - Objects
  - Classes
  - Inheritance
  - Interfaces
  - Abstract Classes
  - Exceptions
- Highlight the concepts and main constructs you need to know for this unit.
- It does not intend to cover everything!

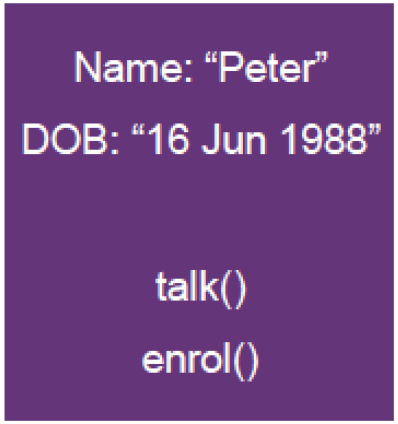
In a computer program, an object:

- is an abstraction of a 'thing' in the real world / a problem domain (e.g. a student)
- An object has:
  - identity
  - attributes (e.g. name)
  - behaviours (e.g. enrol)
- Each object has its own set of values for its attributes (the state of the object)

Car Object



Student Object



Student Object

Class

- A template that describes what its objects will be like
- It defines what attributes (e.g. student ID, DOB) and behaviors (e.g. withdraw, submit assignment) that the program wants to capture for a particular kind of objects
- Each class is defined in a source code that has an extension .java
- Basic elements of a Java class include:
  - Fields / instance variables (define the objects' attributes)
  - Constructors (special methods to create objects)
  - Methods (define the objects' behaviors)

Object Creation

- We use new operator to create/instantiate an object from a class. E.g.

  new Student();

- The statement allocates a block of memory big enough to hold a Student object, and call the constructor to initialise the values of its fields. The **new** operator returns the address of the newly created object.
- To allow the object being referenced later on, we can declare a variable to store the address of the object. E.g.

  Student student1 = new Student();

Basic Structure of a Java Program

- A program may consists of one or more classes
- A class can only be run if it has a main method defined
- A program may have more than one driver class

Example Java Program

```
public class Student
{
        private int studentID;
        private String name;
        public Student(int studentID, String name)
        {
                this.studentID = studentID;
                this.name = name;
        }
        Public String getName()
        {
        return name;
        }
}
Public class Application
{
        public static void main(String args[])
        {
                Student student = new Student(12345678, "John");
                System.out.println(student.getName());
        }
}
```

Object Interactions

- Objects communicate with each other by sending messages. There are three components of a message sent to an object:
  - The name of the object that is the receiver of the message
  - The action that the receiver is requested to take
  - Any information the receiver needs to know to carry out the action requested

  receiver.action(information);
- **Example:**

  student1.setFirstName(“Eddie”);

Data Collection

- Allows store multiple objects of the same type
- E.g. ArrayList, HashMap, HashSet etc.
- **Basic functions:**
  - Insert
  - Retrieve
  - Remove
- **To iterate through a data collection, we can use:**
  - for-each-loop
  - iterator (depending on the type of collection used)

Inheritance

- One of the major concepts in Object-Oriented programming
- **Allows functionalities to be extended from one class to many**
- Inheritance relationships are commonly described as:
  - Parent / Child
  - Base / Derived
  - Superclass / Subclass
- All public/protected attributes and methods are inherited by subclass
- Java only supports single inheritance

Interface

- Create a "**contract**" that spells out how objects of a class can be interacted with by defining
  - Signatures of the methods that need to be implemented
  - Any instance/class variables that will always have the same values (Constant)
- To use an interface, you write a class that implements the interface. This class must provides method body for each of the methods declared in the interface

Abstract Class

- It exists solely for inheritance
- A combination of superclass and interface
- **It is similar to a superclass in a way that:**
  - It can contain mutable fields
  - It can have fully implemented methods
- **It is similar to an interface in a way that:**
  - It can contain one or more abstract method(s) that have no method body
  - It cannot be instantiated

Exception Handling

- An exception is an indication that a problem has occurred during a program execution
- An exception object contains information about the problem
- Exception handler (try-catch)
  - Protects statements in which an exception may occur
  - Can be used for both checked and unchecked exceptions, but it is not a requirement for checked exception

Exception Example

```
try {
    if (/*condition*/) {
        throw new ExceptionClassName(optionalMessageString);
    }
} catch (ExceptionClassName param) {
   // Block of statements to be executed should the first Exception
   // class be thrown.
} catch (ExceptionClassName2 param) {
   // Block of statements to be executed should the second Exception
   // class be thrown.
} finally {
    // Optional block of statements which are executed regardless of
    // whether an exception was thrown or not.
}
```

Java Library

- A set of reusable classes
- Usually packed in a jar file
- Java SE has around 4000 classes/interface called Java API that comes with every JDK distribution
- Java EE also has a set of classes/interface that supports enterprise application development

Java API Docs

- No one can remember everything!
- The best reference is Java API documentations
- You can find the documentations for Java 8 at either:
  - http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html
  - Or Google Java 8 API Docs https://docs.oracle.com/javase/8/docs/api/

Summary

- Provided a quick revision of some object-oriented programming and Java concepts.
  - Objects
  - Classes
  - Inheritance
  - Interfaces
  - Abstract Classes
  - Exceptions
- Highlight the concepts and main constructs you need to know for this unit.