

A large, two-story, light-colored building with a red-tiled roof and a central tower, surrounded by green grass and trees under a clear blue sky.

# MAHARISHI UNIVERSITY of MANAGEMENT

*Engaging the Managing Intelligence of Nature*

## Computer Science Department

**CS390 Fundamental Programming  
Practices (FPP)  
Professor Anne McCollum**

# Lecture 1: Introduction to Java And the Eclipse Development Environment

*Pulling the Arrow Back to Hit the Target*

# Wholeness of the Lesson

Java is an object-oriented highly portable programming language that arose as an easy alternative to the once dominant C++ language. Eclipse is one of many open source, powerful but easy-to-use integrated development environments for use with Java and related technologies. Working from deeper levels of intelligence allows one to accomplish more with fewer mistakes and less effort.

# Goals of the Course

- Become a proficient Java developer by gaining the knowledge in the Unified Field Chart.
- Understand the object-oriented paradigm
- Understand the principles supporting optimal use and implementation of data structures - as general principles and in the Java language
- Become skillful in using the technique of recursion
- Enhance problem-solving skills
- By regularly practicing Transcendental Meditation, develop your potential more fully.
- Side Effect: Prepare for job interviews using FPP topics.
- Prepare yourself for classes in your Masters program.



# Overview of FPP Topics

Topic	Where It Shows Up Later
Java language (FPP and MPP)	Almost all CS courses are taught using Java
OO paradigm (FPP and MPP)	WAP, SWE, WAA, EA
Labs (FPP and MPP)	Core CS: problem-solving using a language, hands-on ability to write correct programs
Data Structures (FPP)	Algorithms, Data Science curriculum, Core CS topic
Swing (FPP)	Core CS. Related to problem-solving: Seeing solutions in the context of a UI is more interesting, prepares student for more advanced UI solutions
Exception-Handling (FPP)	Core CS
File I/O in Java and JDBC (FPP)	Core CS, framework solutions in WAP, WAA, EA are built on top of JDBC

# Enhance Problem-Solving Skills

There are many different approaches. Some examples:

- Get a Good Full Night's Sleep
- Define the problem clearly:
  - Outline your problem – use text &/or pictures. (Many techniques for doing this.)
  - Explain the problem to a friend
  - Work through the problem manually with sample data.
- Move – shake your hands and arms. And if you aren't taking a test: walk, shake your arms & legs, turn your head...
- Create a reminder document for repeated mistakes
- Isolate the Problem from other factors

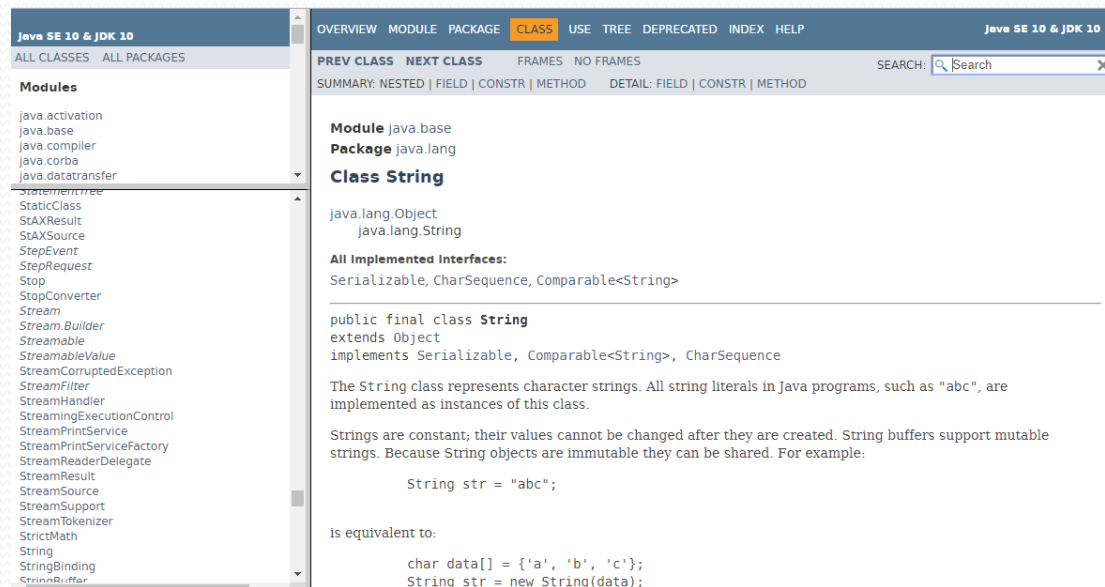
# About Java

- ***Brief History.*** The Java language began as a language for programming electronic devices, though the original project was never completed. Its creator was James Gosling, of Sun Microsystems. The language was developed privately starting in 1991, and was made publicly available in 1994. In 2009, Oracle bought the rights to Java from Sun Microsystems.
- ***Java Is an OOP Language.*** Java is an object oriented programming language. This means that Java programs are designed by defining software objects that interact with each other (mirroring the way things get done in the real world).
- ***Number 1 Language.*** For more than 20 years, Java has been the Number 1 programming language in the IT world.

# The Java 10 API Docs

- Oracle provides online documentation of all the Java library classes. Full documentation of each class in the Java libraries is provided. For Java 10, the link is

<https://docs.oracle.com/javase/10/docs/api/index.html?overview-summary.html>





# Integrated Development Environments

- A good IDE supports compiling, running, and debugging code with tools that are integrated and typically easy to use. For a large Java project, an IDE is indispensable.
- Good choices of IDE are NetBeans, IBM Rational Application Developer (formerly WebSphere Application Developer), Borland's JBuilder, JetBrains' IntelliJ
- Another excellent choice, which has become an industry standard, is the open-source IDE Eclipse, written entirely in Java. Among IDEs for Java, in recent years, Eclipse is the most widely used. We will use Eclipse in this course.

# The Eclipse IDE

- *Getting started.* To use Java 10, you need Eclipse Oxygen (or later); earlier versions of Eclipse do not support Java 10.
- Features of the IDE [demo]
  - Workspace > project > package > class
  - "Perspectives" in Eclipse: Java and Debug perspectives
  - Setting preferences
  - Save / compile / run / debug (example: HelloWorld)
  - Auto-formatting

# JDK and Eclipse Installation

1. The Java JDK 10 is available at the “Java SE 10 Archive Downloads”, this link:

<https://www.oracle.com/technetwork/java/javase/downloads/java-archive-javase10-4425482.html>

2. Eclipse Oxygen version IDE for Java Developers, is available at this link:

<https://www.eclipse.org/downloads/packages/release/oxygen/3a>

- Make sure you choose the correct operating system.
- If the zip file name is long, shorten it when you extract.

# "Hello World"

- Create the simplest Java program. This involves creating a package and then defining a class inside that package.

```
package lesson1.hello;
```

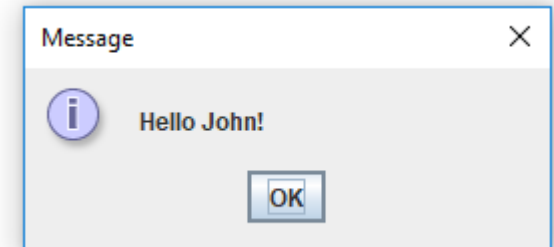
```
public class HelloWorld {
```

```
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }
```

```
}
```

# "Hello World" Dialog

- We create a simple Java program using a Java library. This involves creating a package and then defining a class inside that package.



```
package hello;

import javax.swing.JOptionPane;

public class HelloUI {
    public static void main(String[] args) {
        JOptionPane.showMessageDialog(null, "Hello John!");
    }
}
```



# Sakai Organization

<https://online.cs.mum.edu/portal> or  
<https://sakai.cs.mum.edu/portal>

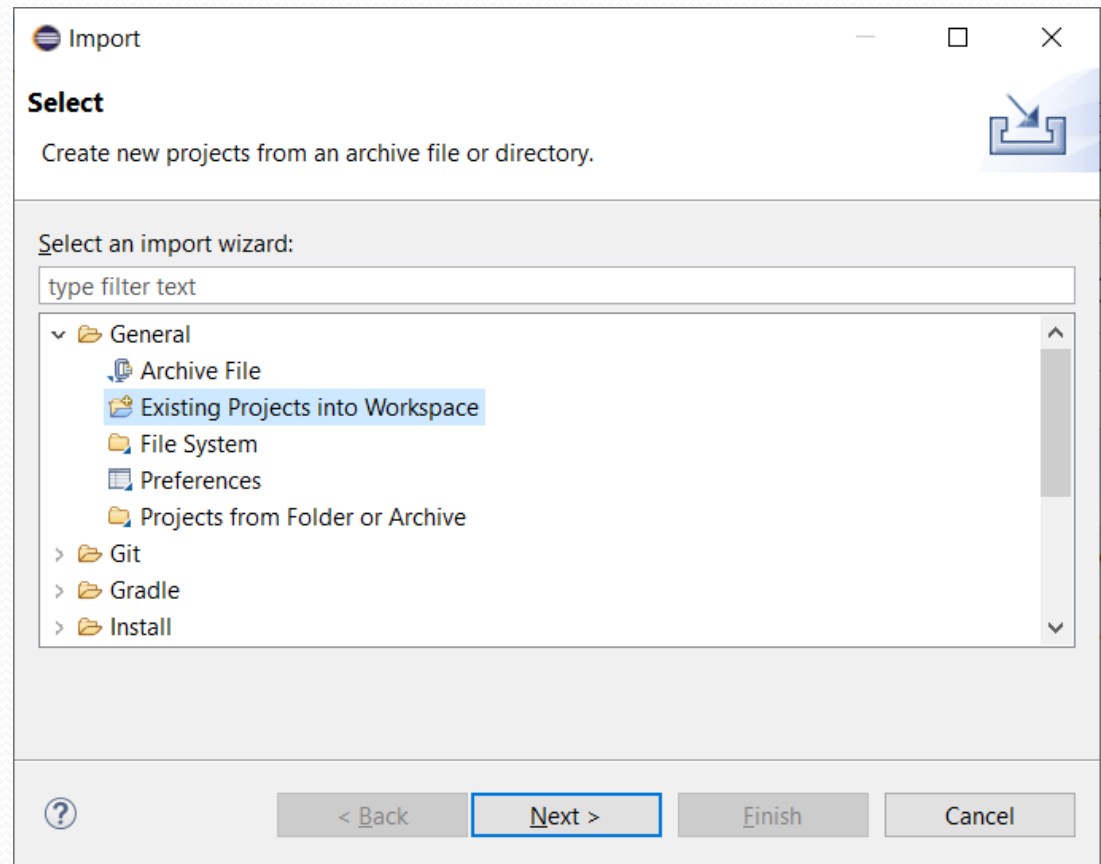
1. Resources folder contains:
  - Syllabus – overview of course and grading
  - InClassExercises
  - Lecture Slides
    - Interview Questions for Advanced Students
  - Lecture Code
  - Set-up
  - Readings
  - And other important information
2. Assignments folder
  - Labs – appear when you should start them, not before.
  - Labs are attached. See the Assignment Instructions for special instructions.

# Set Up InClassExercises

Two ways to import an existing project:

1. File, Import, then expand General and select Existing... then “Next”

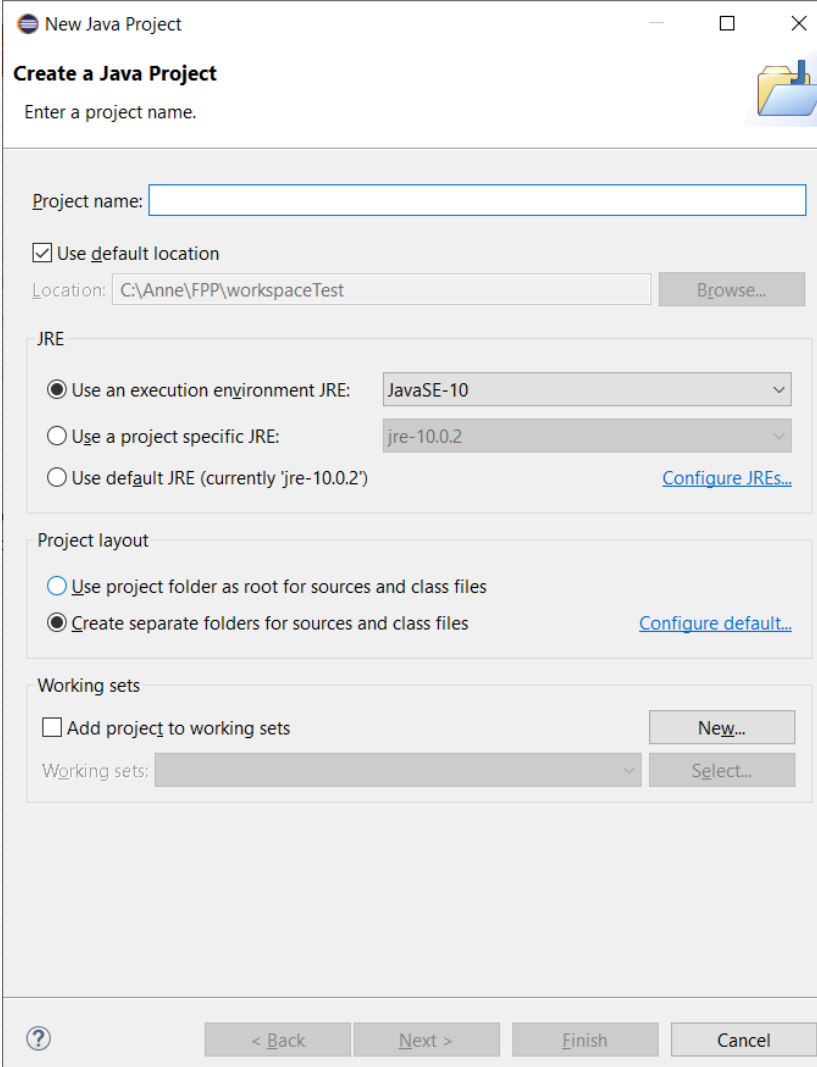
In next dialog,  
choose  
current  
Workspace for  
the root  
directory.



# Project Set Up for InClassExercises and LectureCode

2. Second method of  
importing an existing  
project:

File, New, Java Project  
to go to this dialog:



The screenshot shows the 'New Java Project' dialog box in the Eclipse IDE. The title bar reads 'New Java Project'. Below the title bar, it says 'Create a Java Project' and 'Enter a project name.'.

The dialog is divided into several sections:

- Project name:** A text input field.
- Location:** A text input field showing 'C:\Anne\FPP\workspaceTest' with a 'Browse...' button next to it.
- JRE:** A section with three radio buttons:
  - ☒ Use an execution environment JRE: A dropdown menu showing 'JavaSE-10'.
  - ☐ Use a project specific JRE: A dropdown menu showing 'jre-10.0.2'.
  - ☐ Use default JRE (currently 'jre-10.0.2') with a [Configure JREs...](#) link.
- Project layout:** A section with two radio buttons:
  - ☐ Use project folder as root for sources and class files.
  - ☒ Create separate folders for sources and class files with a [Configure default...](#) link.
- Working sets:** A section with a checkbox 'Add project to working sets' and a 'New...' button. Below it is a 'Working sets:' dropdown menu and a 'Select...' button.

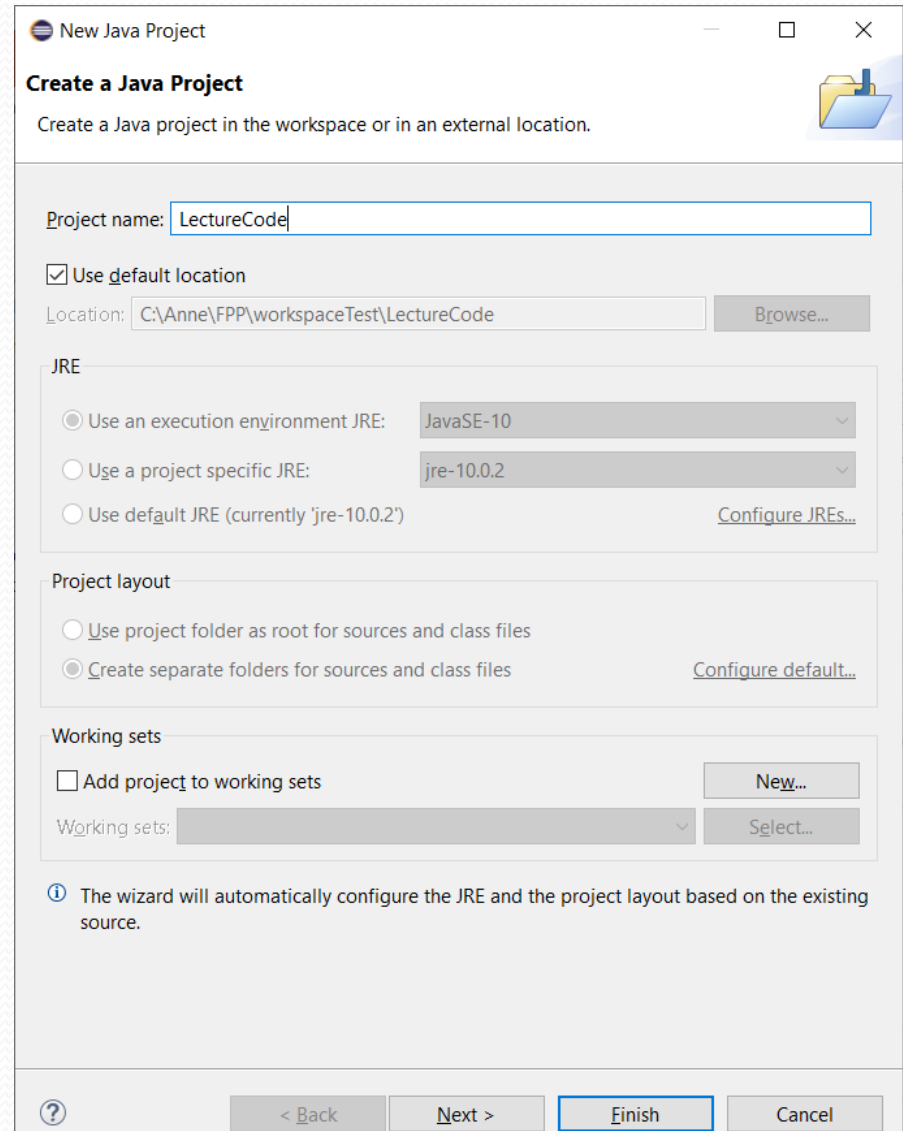
At the bottom of the dialog, there is a help icon (?) and four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

# Project Set Up for LectureCode

Second method of importing an existing project:

File, New, Java Project to go to this dialog. Enter the name of the existing project. Click Finish.

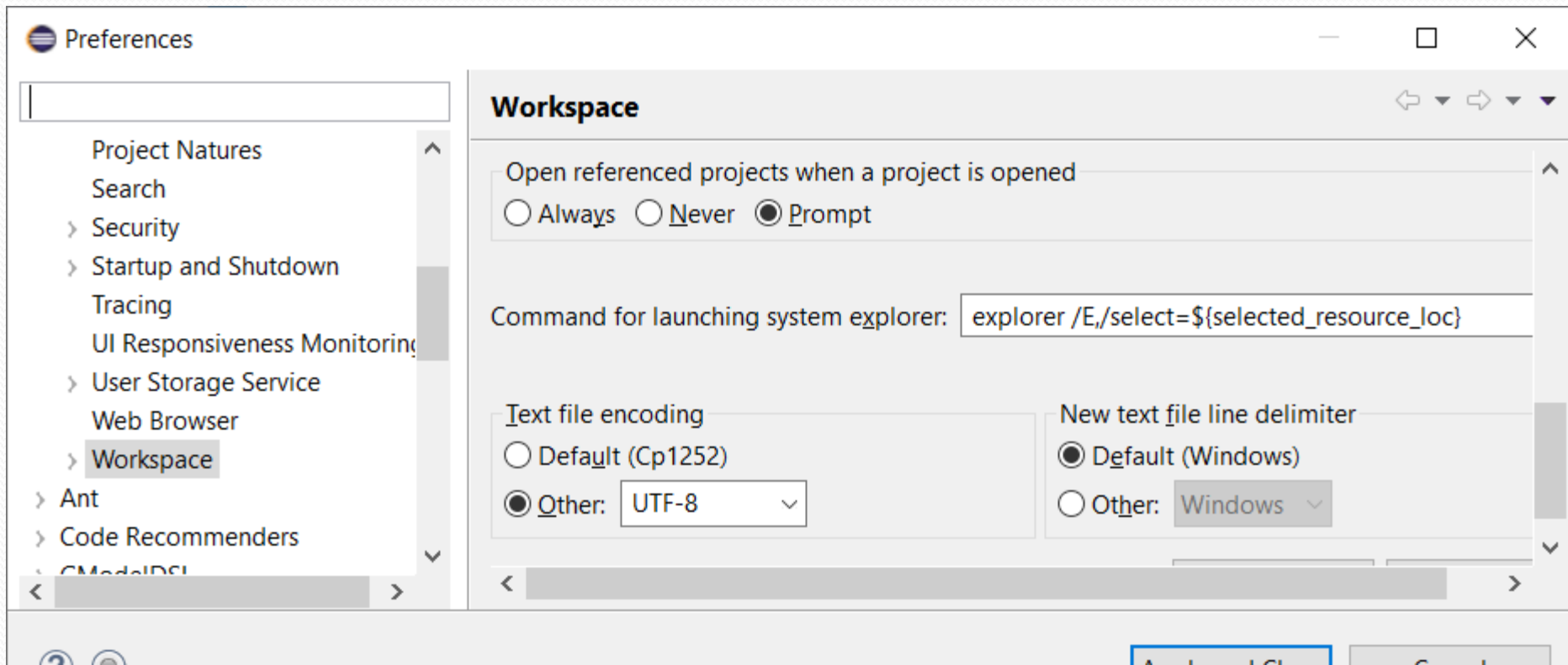
Note that the text at bottom changes when you name an existing project.



The screenshot shows the 'New Java Project' dialog box in the Eclipse IDE. The title bar reads 'New Java Project'. Below the title bar, the text 'Create a Java Project' is followed by the instruction 'Create a Java project in the workspace or in an external location.' The dialog is divided into several sections: 'Project name:' with a text field containing 'LectureCode'; 'Use default location' checked, with a 'Location:' text field showing 'C:\Anne\FPP\workspaceTest\LectureCode' and a 'Browse...' button; 'JRE' section with three radio buttons: 'Use an execution environment JRE:' (selected) with a dropdown showing 'JavaSE-10', 'Use a project specific JRE:' with a dropdown showing 'jre-10.0.2', and 'Use default JRE (currently 'jre-10.0.2')' with a 'Configure JREs...' link; 'Project layout' section with two radio buttons: 'Use project folder as root for sources and class files' and 'Create separate folders for sources and class files' (selected) with a 'Configure default...' link; 'Working sets' section with an 'Add project to working sets' checkbox, a 'New...' button, a 'Working sets:' dropdown, and a 'Select...' button. At the bottom, a message states: 'The wizard will automatically configure the JRE and the project layout based on the existing source.' The bottom navigation bar includes a help icon, '< Back', 'Next >', 'Finish' (highlighted), and 'Cancel'.

# Character Encoding in Eclipse

To change your project to an encoding that supports the Chinese characters in the lecture Code such as '终'. Go to Windows, Preferences, expand General, select Workspace, in the Text file encoding on right select Other and choose UTF-8 as shown below. After Eclipse recompiles, errors disappear...

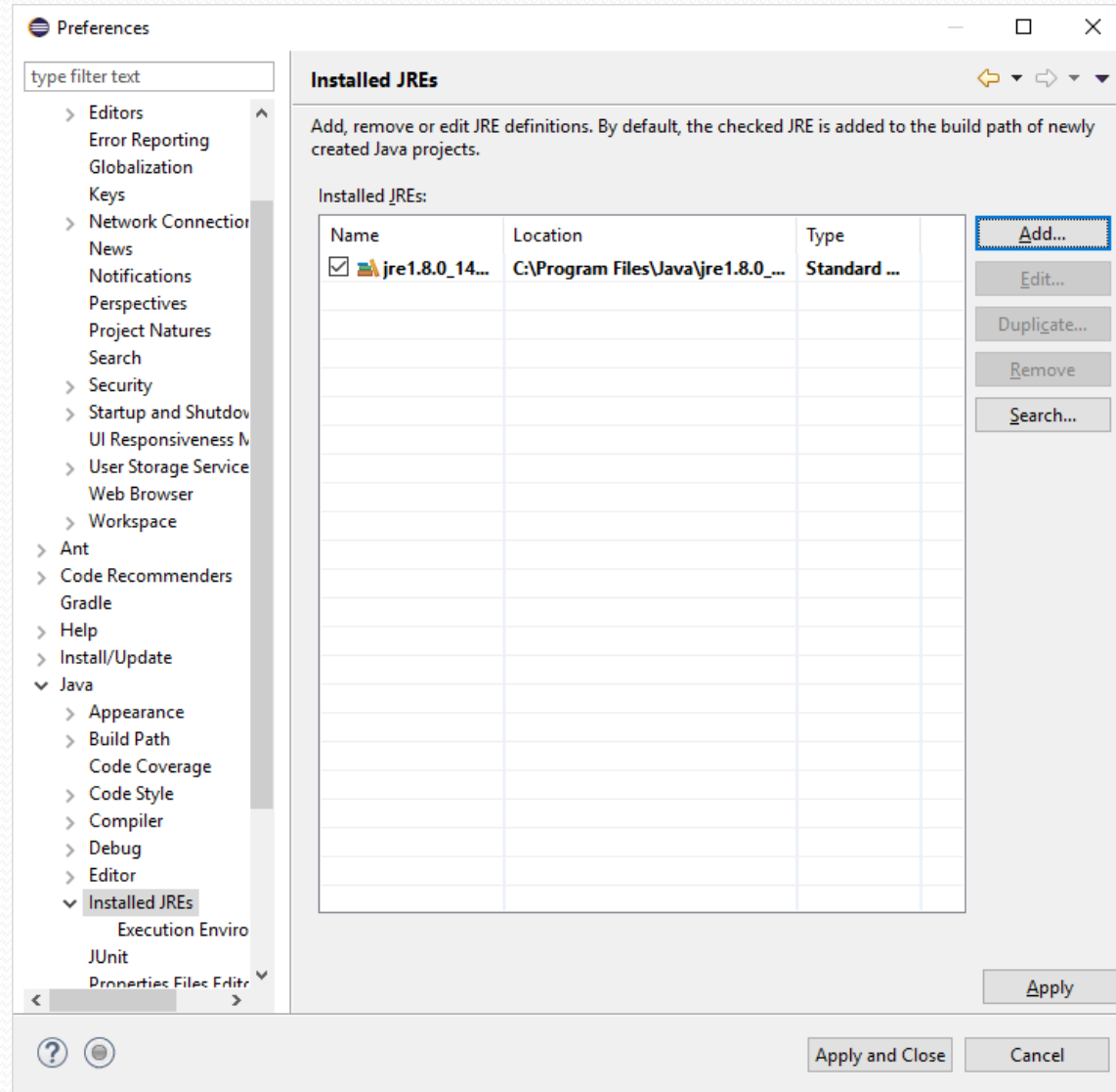




# Setting JDK in Eclipse

If you get errors about the JDK, make sure you have the right JDK for your workspace.

1. Go to Windows, Preferences, expand Java, select Installed JREs. The dialog shown here should appear.
2. Press the Add button in this dialog.



# Setting JDK in Eclipse (continued)

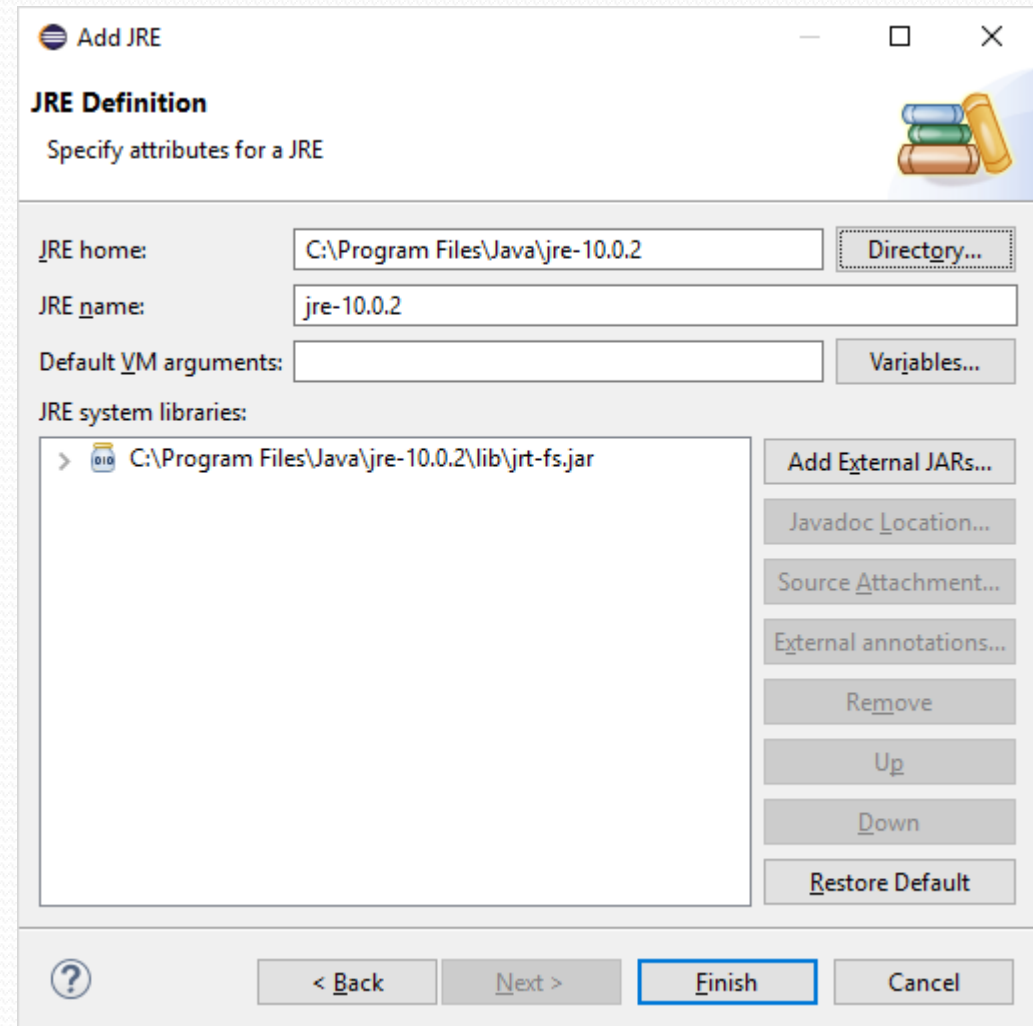
3. On the next dialog, select the default “Standard VM” then click next.

4. On the next dialog (shown here), click on the Directory button.

5. Navigate to the JRE for JDK 10 and select that directory and click open. You come back to this dialog as you see here.

6. Click Finish and it goes back to previous slide.

7. Click Apply and Close.



# In-Class Exercise 1.1

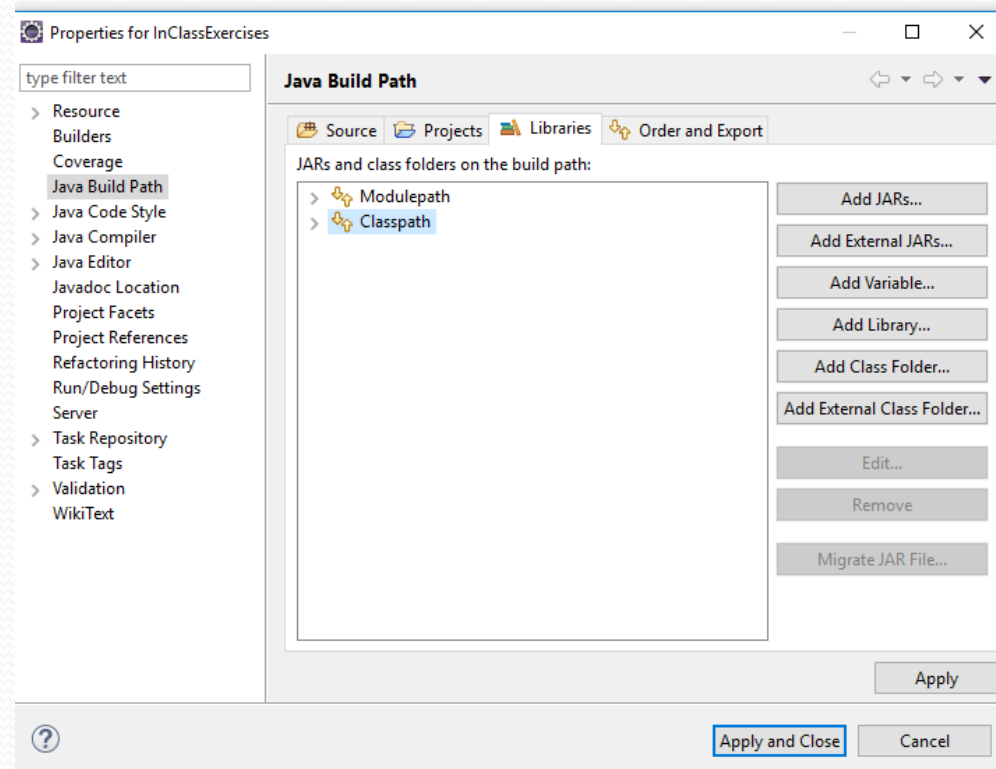
- In this exercise you will run a Java application `Sample` within your Eclipse workspace, available in the `InClassExercises` project.
- The comments written in the class `Sample` explain many details about basic Java code. The goal is to study the code and run it to get a feeling for how it works, and then try to use what you have learned to implement the requirements in a second Java class `MyClass`.

# In-Class Exercise 1.2

- In this exercise you will once again study sample code contained in a package `lesson1.exercise_2.program`, and try to use what you have learned to write some additional code.
- This program illustrates how in a Java program, objects interact with other objects to produce the desired program behavior.
- To make the sample code run properly, an extra library needs to be added in the form of a JAR file

# Adding JAR Files in Eclipse

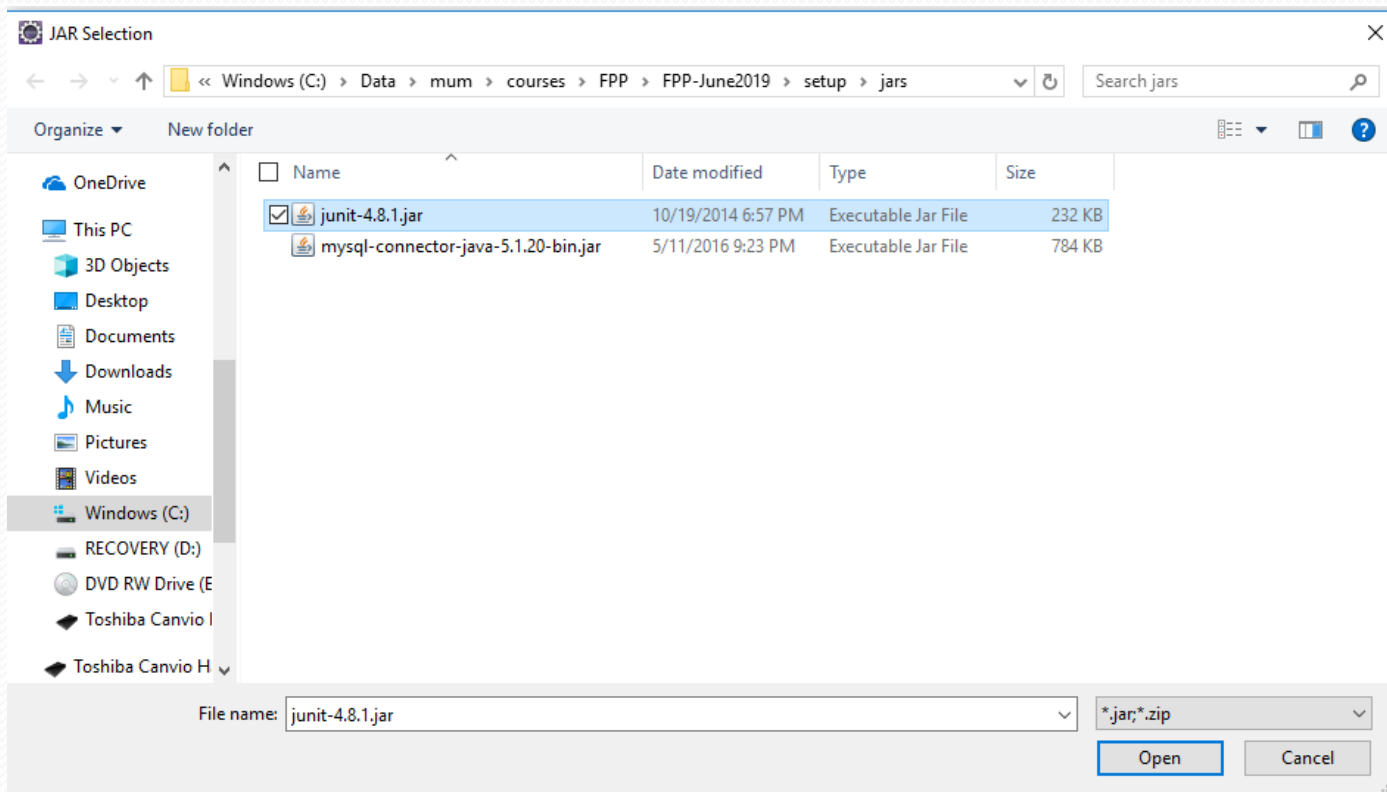
- You may need to add junit-4.8.1.jar to the "classpath" if you are using the Junit testing feature. You can find the junit-4.8.1.jar that is already on your computer or use the one in the Set-up folder under Resources. Note: This folder also has a fpp-setup.zip file, which contains the junit-4.8.1.jar and many other important files.
- Select then right click on the Project.
- Select Properties, Java Build Path, select Classpath, and click Add External Jars...





# Adding JAR Files in Eclipse

- Select the junit-4.8.1.jar that you downloaded
- Click Open.
- Click Apply and Close



# JShell - Isolating a Problem

To test a small piece of Java code, you may wish to compile and run it separately from your program or IDE. One of the ways that this can be done is by using Jshell, a commandline feature of Java.

There are two ways to use JShell:

1. Go to a command window, enter JShell, and hit return. In order for this to work properly, you *might* need to set up the JDK command line tools as shown on the next page.
2. Easiest: Go to the directory where JShell.exe resides and double click on it. The default location for JDK version 10 is C:\Program Files\Java\jdk-10.0.2\bin.

# Trying Out Code with JShell

- JShell is a Java feature where you run code interactively at the commandline.
- Use JShell to evaluate expressions (see below).

Reminder: Start Jshell by going to the directory where JShell.exe resides and double clicking.

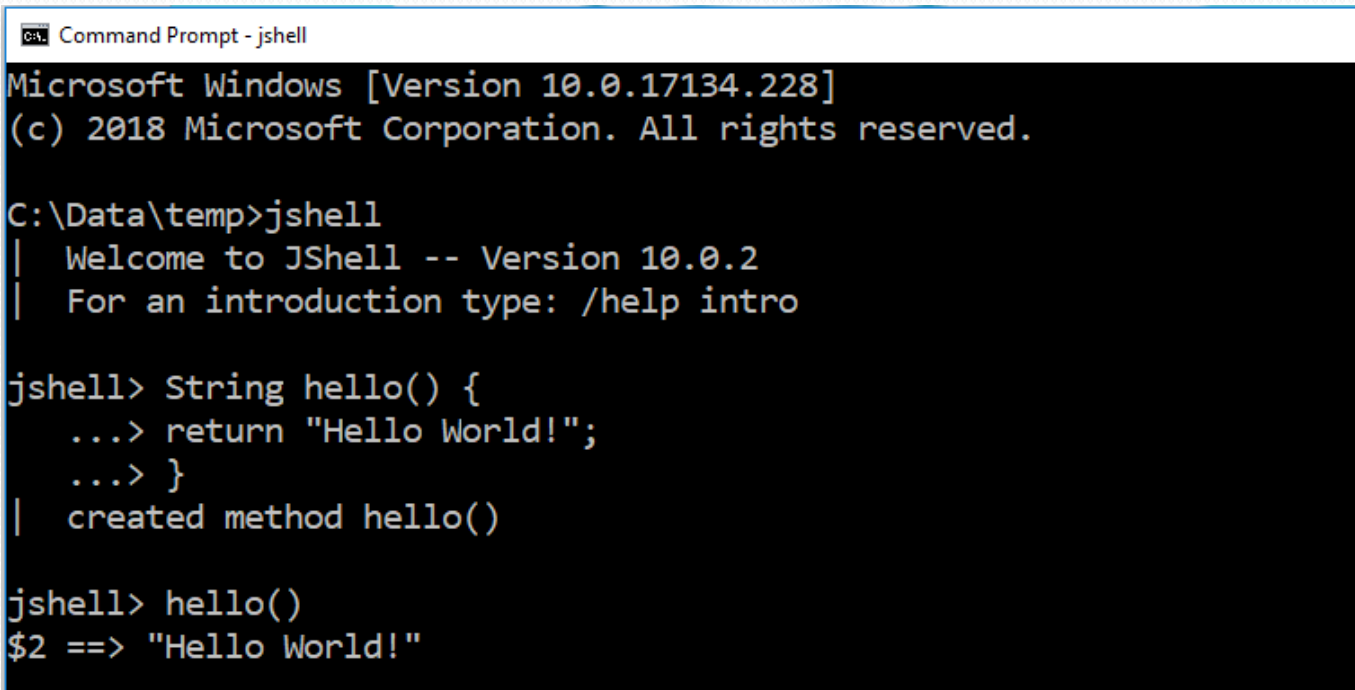
Command Prompt - jshell

```
C:\Data\temp>jshell
| Welcome to JShell -- Version 10.0.2
| For an introduction type: /help intro

jshell> 2 * 3 + 7
$1 ==> 13
```

# (continued)

- You can define a Java method in JShell and then run it



```
Command Prompt - jshell
Microsoft Windows [Version 10.0.17134.228]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Data\temp>jshell
| Welcome to JShell -- Version 10.0.2
| For an introduction type: /help intro

jshell> String hello() {
...> return "Hello World!";
...> }
| created method hello()

jshell> hello()
$2 ==> "Hello World!"
```

- Exit JShell by typing /exit at the jshell prompt or closing the command window.

# Main Point

Eclipse is a leading, open-source, integrated development environment, which provides excellent support for editing, compiling, running, and debugging Java applications. By analogy, we get the best possible advantages from diving deep to a state of great rest and silence in TM then engaging in activity that structures a life-supporting environment. The goal is to live 200% of life, enjoying 100% of our inner life and 100% of benefits from our outer environment.



# Connecting the Parts of Knowledge With the Wholeness of Knowledge

1. Using Java, highly functional applications can be built more quickly and with fewer mistakes than in some other languages.
  2. To optimize the use of Java's features, IDEs such as Eclipse ease the work of the developer by handling in the background many routine tasks.
- 
3. **Transcendental Consciousness:** To be successful, action must be based on the field of pure intelligence, which is located at the source of thought.
  4. **Wholeness moving within itself:** In Unity Consciousness, the pure intelligence located in TC is found pervading all of creation, from gross to subtle.