Chapter 1: Creating Java Programs

What is Computer Programming?

Computer programming is a process that leads from an original formulation of a **computing problem to executable computer programs**.

A program is a set of **step-by-step instructions that directs the computer to do the tasks** you want it to do **and produce the results you want**.

Learning Programming Terminology

Machine (Low level programming language) VS High level programming language

- **Low-level programming language**
  - The most basic circuitry-level language such as machine code or assembly code.
- **High-level programming language**
  - Allows you to use a vocabulary of reasonable terms
- **Syntax**
  - A specific set of rules for the language
  - For example …is Java syntax case sensitive?

High level code:

```
Read num1
READ num2
Total= num1+num2
```

Low-level assembly code:

```
SED
LDA  NUM1
CLC
ADC NUM2L
STA RESL
LDA NUM1M
ADC NUM2M
STA RESM
CLD
RTS
```

Learning Programming Terminology

- **Compiler** or **interpreter**
  - Translates language statements into machine code
- **Syntax error**
  - Misuse of language rules
  - A misspelled programming language word
- **Debugging**
  - Freeing program of all errors
- **Logic errors**
  - Also called **semantic errors**
  - Incorrect order or procedure
  - The program may run but provide inaccurate output

Problems associated with programming

Features of the Java Programming Language

- **Java**
  - Developed by Sun Microsystems
  - An object-oriented language
  - General-purpose
  - Advantages
    - Security features
    - **Architecturally neutral**

Features of the Java Programming Language (cont'd.)

- **Java** (cont'd.)
  - Can be run on a wide variety of computers
  - Does not execute instructions on the computer directly
  - Runs on a hypothetical computer known as a **Java Virtual Machine (JVM)**
- **Source code**
  - Programming statements written in high-level programming language
- **Development environment (IDE)**
  - A set of tools used to write programs
  - Jgrasp, Netbeans

Analyzing a Java Application that Produces Console Output

- Even the simplest Java application involves a fair amount of confusing syntax
- Print “Hello World” on the screen

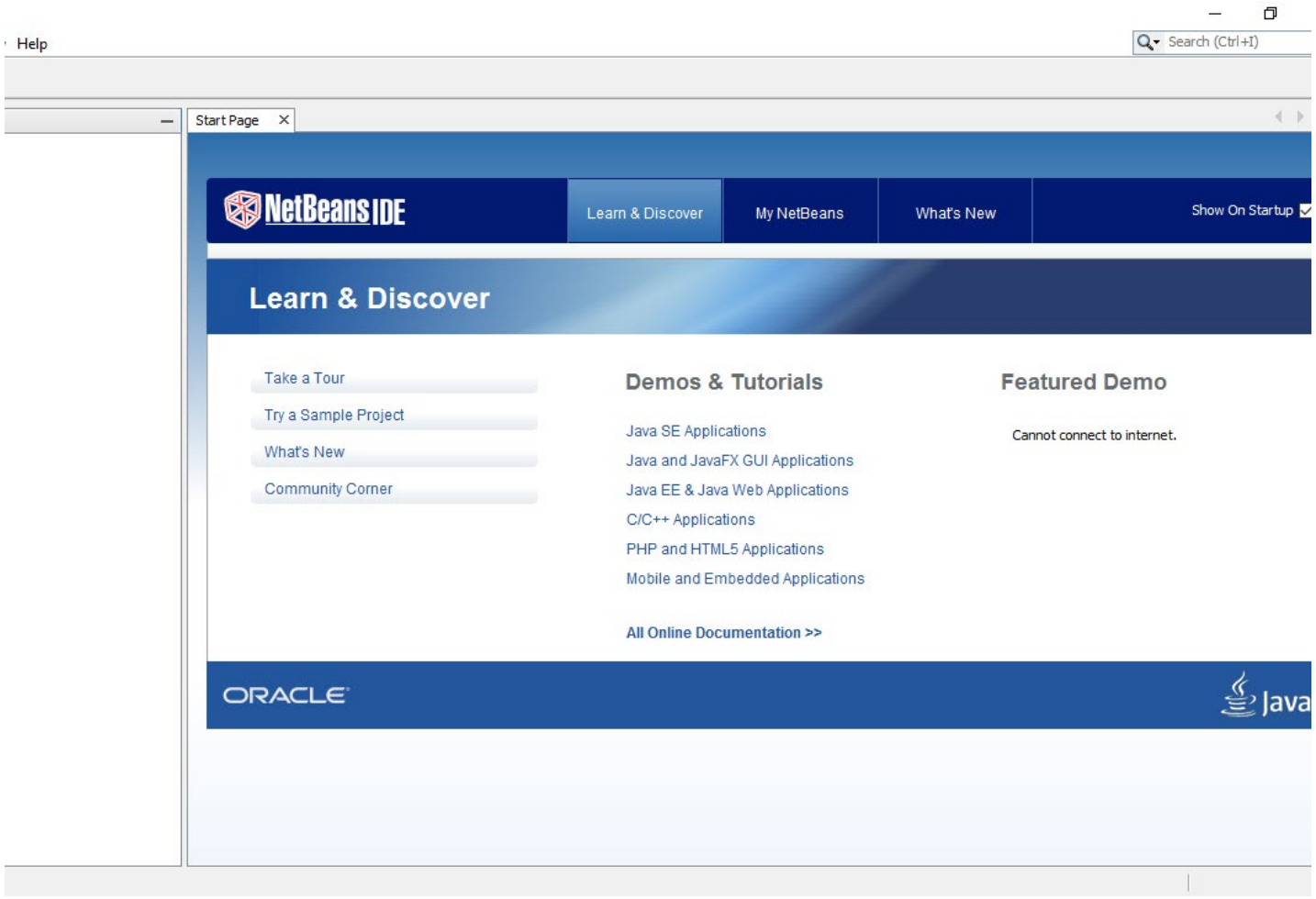
Demonstration

3 different ways of compiling your first Java program:

- **Notepad/command prompt (chapter 1 uses notepad) (popular editor: sublime)**
- **Jgrasp**
- **Netbeans (you will be using Netbeans in class)**

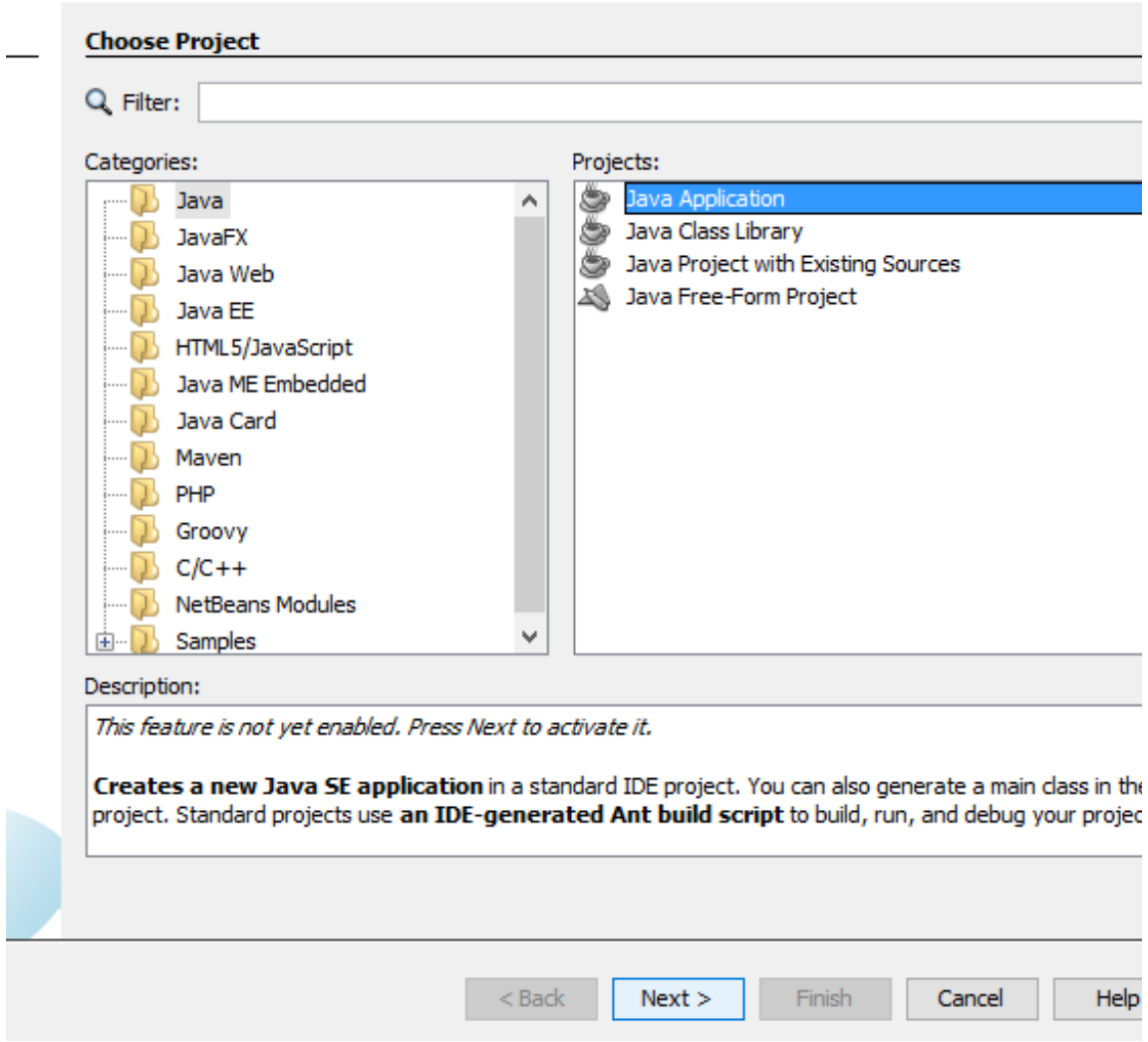
Activity 1: Load Netbeans

You should see a Start Page screen as below:



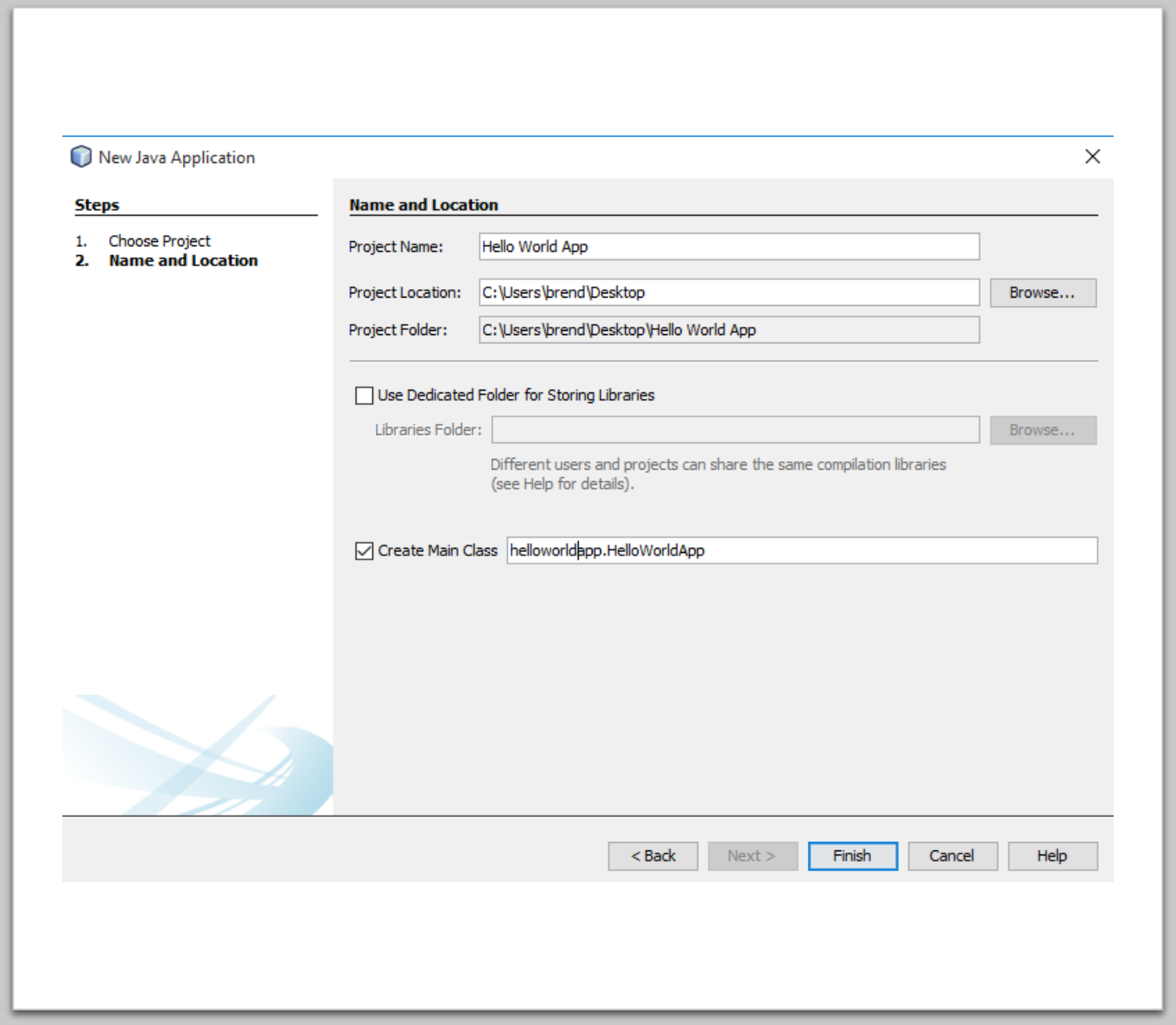
- **Activity 2: Creating a Project**

- - Choose **File** menu and select **New Project**
- In the new window that opens, click on **Java** in the **Categories pane** then click in **Java Application** in the Projects pane. Click **Next** to continue.



- **Activity 3: Creating a Project**

- -In the **Name and Location** page of the wizard, do the following (as shown in the figure below):
- In the **Project Name** field, type **Hello World App**.
- In the **Create Main Class** field, type **helloworldapp.HelloWorldApp**.
- **Click Finish**



- Activity 4: **Adding code to the Source Editor**

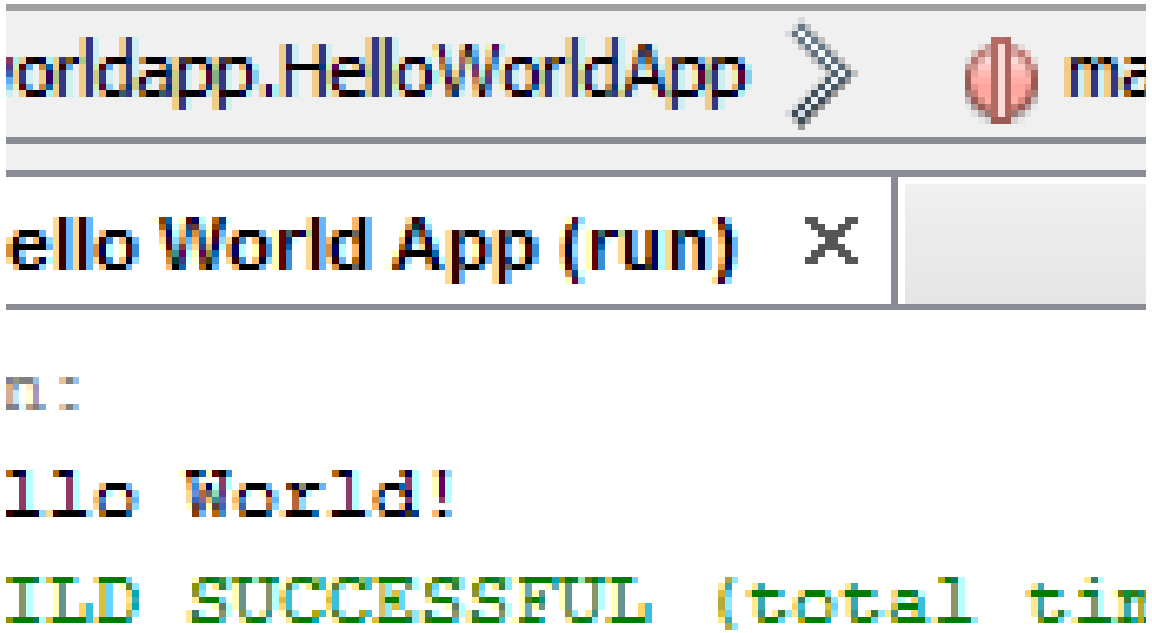
```
 * To change this template file, choose Tools
 * and open the template in the editor.
 */
package helloworldapp;

/**
 *
 * @author brend
 */
public class HelloWorldApp {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
    }

}
```

System.out.println("Hello World!"); // Display the string

- **Activity 5: Compile the Source File into a .class File**

- To compile your source file, choose **Run** | **Build Project (Hello World App)** from the IDE's main menu.
- From the IDE's menu bar, choose **Run** | **Run Main Project**

- The Output window opens and displays output similar to what you see in the following figure:



Adding Comments to a Java Class

Program comments

- Nonexecuting statements added to a program for documentation
- Use to leave notes for yourself or others
- Include the author, date, and class's name or function

Comment out a statement

- Turn it into a comment
- The compiler does not translate, and the JVM does not execute its command

Adding Comments to a Java Class (cont'd.)

- Types of Java comments
  - **Line comments**
    - Start with two forward slashes (//)
    - Continue to the end of the current line
    - Do not require an ending symbol
  - **Block comments**
    - Start with a forward slash and an asterisk (/*)
    - End with an asterisk and a forward slash (*/)

Adding Comments to a Java Class (cont'd.)

- Types of Java comments (cont'd.)
  - **Javadoc** comments
    - A special case of block comments
    - Begin with a slash and two asterisks (/**)
    - End with an asterisk and a forward slash (*/)
    - Use to generate documentation

Adding Comments to a Java Class (cont'd.)

```
// Demonstrating comments
/* This shows
   that these comments
   don't matter */
System.out.println("Hello"); // This line executes
   // up to where the comment started
/* Everything but the println()
     is a comment */
```

Figure 1-21 A program segment containing several comments