

Introduction to Programming

with Java, for Beginners

- How does the computer work?
- Machine vs. Programming Language
- Java Compiler and VM
- Java Program Structure
- Sequential Programming
- User Input

What does the Computer process & store?

- An electronic device can represent uniquely only one of two things
 - Each "0" and Each "1" is referred to as a **Binary Digit** or **Bit**
 - Fundamental unit of information storage
- To represent more things we need more bits
 - E.g. 2 bits can represent four unique things: 00, 01, 10, 11
 - k bits can distinguish 2^k distinct items
- Combination binary bits together can represent some information or data. E.g. 01000001 can be
 1. Decimal value 65
 2. Alphabet (or character) 'A' in ASCII notation
 3. Command to be performed e.g. Performing Add operation

ESE112

2

What does the Computer Understand?

- At the lowest level, a computer has electronic "plumbing"
 - Operates by controlling the flow of electrons through very fast tiny electronic devices called transistors
- The devices react to presence or absence of voltage
 - Could react actual voltages but designing electronics then becomes complex
- Symbolically we represent
 1. Presence of voltage as "1"
 2. Absence of voltage as "0"

ESE112

1

Machine Language

- Computers understand only 0's and 1's
 - A.k.a **Machine (hardware) Language**
 - Each machine has it unique language
 - E.g. combination of 00101001 is not same meaning on different machine
 - Difficult for humans to program the computer in machine language



ESE112

3

Programming to Machine Language

- The compiler translates the programming language into a *specific machine* language
 - Specific Machine:
 - Electronic Hardware + Operating System
- Once translated (Programming -> Machine)
 - The same program **cannot run** on different machine
- Java avoids the above problem
 - Code is portable - Write one run anywhere!
 - One of the features for popularity of Java

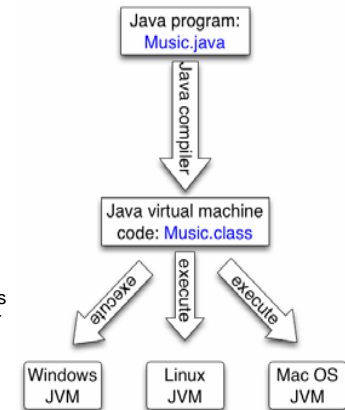


ESE112

4

Java Compiler and Virtual Machine

- The Java Compiler
 - Reads file with extension **.java**
 - Checks syntax / grammar
 - Creates a **.class** file which contains **byte**(or machine) code independent of any machine
- Java Byte Code
 - Is **portable**
- JVM(Java Virtual Machine)
 - Translates **byte code** in to instructions (actual machine code) for a particular processor
 - The actual machine code then is executed on the computer

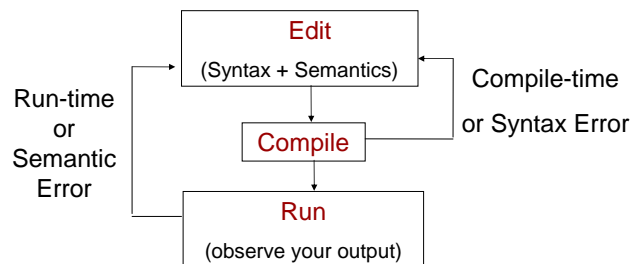


ESE112

5

Process of Computer Programming

- Programming Cycle



- Philosophy: program in increments

ESE112

6

Java Program Structure

- Consist of one or more files ending **.java**
- Each file has the following structure


```
class Classname {
    ...
}
```
- Filename and Classname **must exactly match**
- The curly braces { } define start and end of class description
 - Syntax error if the brace pair is missing
- Classname
 - Must start with alphabet – Java Rule
 - The first letter must be capital – Style Rule
 - Can be made up of alphanumeric characters and underscore

ESE112

7

Special Syntax

- Within any one class description if we have the special syntax

```
public static void main (String[] args) {  
    statement(s)  
}
```

- This known as a program's (computational solution) entry point i.e. where it starts getting executed
 - Called the *main* method
 - A method is a *named* group of statements
 - For now ignore keywords, public, static and void, String [] args

ESE112

8

Statements

- Describes a behavior
 - End with semicolon(;
- Examples:
 - Declaration and Initialization statement
 - int x = 560 * 9;
 - Printing to output screen
 - System.out.println("x = " + x);

ESE112

9

Example Hello.java

```
class Hello{  
    public static void main(String[] args){  
  
        // A statement that prints to output screen  
        System.out.println("Hello World");  
  
    }// end of main  
}
```

ESE112

10

Sequential Programming

- Computer executes statements in the order the statements are written
- Example:
Welcome to Dr. Java
 - > int num = 2;
 - > int sqNum;
 - > sqNum = num * num;
 - > System.out.println(sqNum);

ESE112

11

Sequential Programming (contd..)

```
class PrintSquare{
    public static void main (String [] args){
        int num = 2;
        int sqNum = 0;
        sqNum = num * num;
        System.out.println(sqNum);
    }
}
```

Note: For now all statements must be inside the main method

ESE112

12

User Input

- What if we make program ask the user to enter the number to be squared?
- Two things:
 - Ask the user so that he/she can respond – how?
 - System.out.println("Enter a number to be square");
 - Scan in and store the number into a variable
 - To store I need?
 - To scan – Use already implemented functionality
 - Provided in TextIO.jar

ESE112

13

Modified PrintSquare Program

```
class PrintSquare{
    public static void main (String [] args){
        //Declare & Initialize variables for computation
        int num = 0;
        int sqNum = 0;

        //Ask the user for input
        System.out.println("Enter a integer to be squared");

        /* To scan user input (integer value)
        * Must include TextIO.jar
        */
        num = TextIO.getInt(); //specific to reading an integer

        //Square the input
        sqNum = num * num;

        //Print the result
        System.out.println("Result = " + sqNum);
    }
}
```

ESE112

14

Files with extension .jar

- Jar stands for Java Archive
- Collection of precompiled java files (.class), and other files (images, sound etc.)
- Good way to port Java code written on one machine to another
- If you want to run a pre-compiled program, Dr Java needs to know where to find the file with extension .jar
 - See tutorial for how to add a jar

ESE112

15

Other Functionality in TextIO.jar

- Checks for errors:
 - E.g. Asking for an integer and user enters a non-numeric input
- To scan double:
 - E.g. `double d = TextIO.getInDouble();`
- To scan string:
 - E.g. `String name = TextIO.getInString();`