Introduction to Eclipse
Java Statements

• Declaration
• Assignment
• Method calls
Declaration

• A variable is like a container that can hold certain content/value.
• Variable declaration consists of type and name
• Variable name, by convention, begin with a lowercase letter, and every word after first word begins with a capital letter, e.g. firstName, accountNumber, courseName.
• The type shows what content the “container” can hold.
• The variable name will be used in other statements when such content is desired.
• int age;
• float weight;
• String name;
Assignment

- Assignment is to put content into a container.
- Assignment syntax is variable = value;
- age = 18;
- weight = 60.5;
- name = "Alex";
Method Call

• A method is a unit of work always done together.
• Call the method when you want to do such a unit of work.
• float total = sum(s1, s2, s3);
• getAccount(accountID);
• joe.buildHouse(location);
Comments

• End of line (single line) comment starts with //

//This is a comment line.

• Multiple line comments are enclosed by /*
  and */

/* This is
multiple line
comments */
Each statement ends in semicolon

- Declaration. int age;
- Assignment. age = 1;
- Method call. total = Sum(s1, s2, s3);
public class StatementExample {

    public static void main(String[] args) {

        int age;
        float weight;
        age = 5; //My dog is five years old.
        System.out.println("My dog is "+age+" years old.");
        weight = 10; //My dog weights 10 pounds.
        System.out.println("My dog weights "+weight+" pounds.");

    }

}
You can do in Java

• Split a statement in multiple lines to enhance readability, e.g.

```java
System.out.println (lastName +
    firstName +
    streetAddress +
    town +
    state +
    zip);
```
Good Programming Practice

• Write a comment before each class, documenting the purpose of the class.
• Write end of line (single line) comment, documenting the purpose of the statement.
• Use Eclipse to check your syntax error.
• Declare each variable in each line, allowing end of line comment. E.g.
  ```
  int age; //The age of my dog
  ```
• Choose meaningful variable names helps a program to be self-documenting. Easy to understand.
Which code is easier to understand?

- **Code Sample 1:**
  
  ```c
  float examOneGrade = 90.0;
  float examTwoGrade = 95.0;
  float examThreeGrade = 85.0;
  float myAverageExamGrade = (examOneGrade +
      examTwoGrade +
      examThreeGrade)/3;
  ```

- **Code sample 2**
  
  ```c
  float asdfghjk = 90.0;
  float rujflwe = 95.0;
  float asdfkJ = 85.0;
  float dfiefjek = (asdfghjk +
      rujflwe +
      asdfkJ)/3;
  ```
Console Output

• System.out.println(“Hello World!”);

• Display a single line of text with multiple statements.
  System.out.print(“Welcome to “);
  System.out.println(“ECE122!”);

• Display multiple line of text with a single statement
  System.out.println(“Welcome
  to
  ECE122!”);
Demo ConsoleOutput.java
Common Escape sequence

• \n  Newline. Position the screen cursor at the beginning of the next line.
• \t  Horizontal tab.
• \\  Backslash. Print a backslash character.
• ""  Double quote. Print a double quote character.
Demo ConsoleOutput.java
Console Input Demo

• Import package we will use. Let compiler and run time know where to find the class to load.

• Read from console is more “risky” than output. Things can go wrong. We want to prepare for it. Use “Exception Handling”. We will “try” to read from console. If something goes wrong, the system will throw something called “Exception”. We will then “catch” it and proceed.

```java
try {
    //read from console
}
catch(Exception e) {
    //do something to repair it
}
```
Demo ConsoleInput.java
Assignment

• Read “Head First Java” Chapter 1
• Read “Java 2, a beginner’s guide” Chapter 1
• Install JDK 1.4.2 on your home computer.
• Install Eclipse 3.0.1 on your home computer.
• Create a new java project within Eclipse workspace. Compile “HelloWorld.java”, run “HelloWorld”