Programming

• For the next ten weeks you will learn basic programming principles
  – There is much more to programming than knowing a programming language

• When programming you need to use a tool, in this case the tool will be a language
  – In this course you will use java to explore programming
  – You will use other languages to program

Syntax and Semantics

• When using a programming language you must understand both the syntax and the semantics of the language.
  – Syntax refers to the rules you must follow to form valid statements in the language.
    • The syntax of a language is like English grammar.
  – Semantics describe the meaning of a statement in a language.
Language Translation

- The only language that a computer understands is machine language.
- A program, written in a programming language other than machine language, must be translated into machine language in order for it to run on a computer.
- Programs are typically either compiled or interpreted.

Compiling

- When a program is compiled it is converted directly into machine language.
  - Sort of like having a book translated from English to German.
- The program that does the translation is called a compiler.
Interpreted Languages

- Another approach is to convert the program to machine language while it is running
- An interpreter translates and immediately executes a program
  - Someone, who understands German, reads the book in German and translates while reading to English

Source Code ➔ Execute

What Is Java

- Java started as a programming language for embedded systems (toasters, microwave ovens, washers, etc.).
  - needed to be portable.
  - had to be reliable.
- The original language was called oak (rumor has it that Gosling has a large oak tree outside the window of his office). Marketing decided Java was a better name.
Sun’s Slant

• According to Sun:
  – Java is a simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, and dynamic language
• Java is a lot like C/C++ but there are a number of important differences

Program Structure

• A program in Java consists of one or more class definitions. One of these classes must define a method `main()`, which is where the program starts running

```java
// A Java Hello World Program

public class HelloWorld {
    public static void main( String args[] ) {
        System.out.println( "Hello World" );
    }
}
```
How Is Java Different

- Java differs from other **popular** languages:
  - It is interpreted
  - Architecture neutral
  - There are no C/C++ style pointers, only references
  - Garbage collected
  - Comes with a sophisticated class library
  - Includes support for concurrency, networking, and graphics
Java Environments

• There are lots of commercial Java programming environments.
  – IBM’s Visual Age.
  – Visual J++.
  – Semantic Café.
  – many others (most of which cost money).

• Sun provides the JDK (Java development Kit) for free.

The JDK

• The JDK consists of the following:
  – The Java development tools, including the compiler, debugger and the Java Interpreter.
  – The Java class libraries organized as a collection of packages.
  – A number of demonstration programs.
  – Various supporting tools and components, including the source code of the classes in the library.

Java Resources

- Java Home Page
- The Java Tutorial
  - http://www.java.sun.com/docs/books/tutorial
- Java Developer Connection
  - http://developer.java.sun.com
- The Swing Connection

Other Resources

- RIT Course Pages
  - http://www.cs.rit.edu/~cs1
  - http://www.cs.rit.edu/~cs2
  - http://www.cs.rit.edu/~cs3
- NT-EMACS
- JDE
  - http://sunsite.auc.dk/jde/
Applications and Applets

- Java programs come in two forms:
  - Applications.
  - Applets.
- Applets typically are downloaded into a browser and are run by the Java Virtual Machine that is part of the browser.
  - Usually are restricted as to what they can do.
- Applications are standalone programs that can do just about anything.

Basic Java Syntax

- The Java language will be described by working through its features:
  - variable types and expressions.
  - selection and iteration.
  - classes.
  - exceptions.
- Small sample programs will be provided to illustrate how each feature is used.
Applications and Applets

- Java programs come in two forms:
  - Applications.
  - Applets.
- Applets typically are downloaded into a browser and are run by the Java Virtual Machine that is part of the browser.
  - Usually are restricted as to what they can do.
- Applications are standalone programs.

Things & Stuff

- Any program that you will write will manipulate things
  - Numbers
  - Strings
  - Objects
  - ...  
- We need to be able to refer to these sorts of items in a program
Identifiers

- Identifiers are used in a programming language to refer to things
  - You can think of an identifier as a shortcut to a memory location somewhere in the computer
- Declarations in a program introduce identifiers and the type of thing they will refer to
- All identifiers must be declared before they may be used

Rules

- Identifiers
  - start with an alphabetic character
  - can contain letters, digits, or “_”
  - are unlimited in length
- Examples

  Answer, total, last_total, relativePosition, gridElement
  Person, Place, Stack, Queue
Declaring Variables

• The basic syntax for declaring variables is:
  – typename identifier;
• It is possible to declare two or more variables of the same type in a single declaration statement.

Categories of Variables

• There are two categories of variables:
  – Variables of primitive type which directly contain a representation of a value of a primitive type.
  – Variables of a reference type which hold a reference to an object or the value null (which is the null reference).
• All variables must be declared and initialized before being used.
Primitive Types

- The primitive types represent the basic, built-in types that are part of the Java language.
- Two basic categories:
  - Boolean - boolean.
  - Numeric.
    - Integral - byte, short, int, long, char.
    - Floating point - float, double.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>Has two values, true and false.</td>
</tr>
<tr>
<td>byte</td>
<td>8-bit signed 2’s complement integers, range: -128 to 127</td>
</tr>
<tr>
<td>short</td>
<td>16-bit signed 2’s complement integers, range: -32768 to 32767</td>
</tr>
<tr>
<td>int</td>
<td>32-bit signed 2’s complement integers, range: -2147483648 to 2147483647</td>
</tr>
<tr>
<td>long</td>
<td>64-bit signed 2’s complement integers, range: -9223372036854775808 to 9223372036854775807</td>
</tr>
<tr>
<td>char</td>
<td>16-bit unsigned values from 0 to 65535, representing Unicode characters</td>
</tr>
<tr>
<td>float</td>
<td>Single precision, 32-bit format IEEE 754 floating-point values, range: 1.40239846e-45 to 3.40282347e+38</td>
</tr>
<tr>
<td>double</td>
<td>Double precision, 64-bit format IEEE 754 floating-point values, range: 4.9406564581246544e-324 to 1.79769313486231570e+308</td>
</tr>
</tbody>
</table>

There are special floating point values: ‘positive infinity’, ‘negative infinity’, and ‘not a number’ (NaN).

Note: these types are platform independent.
Unicode

- An International Standard that defines the representation of characters from a wide range of alphabets.
- Unicode stores characters as 16-bit values providing 65,536 different characters.
- ASCII happens to be the first 127 characters in the Unicode standard.
- Java uses Unicode as opposed to ASCII.

Unicode Escapes

- Unicode escapes allow any character to be represented regardless of the editor being used.
- A Unicode escape stands for a character and is represented using the \u escape sequence followed by the hexadecimal digits of the character code.
- Examples:
  \u0343, \u2f4, \uabcd
Literals

<table>
<thead>
<tr>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>0, 123, -456, 55665, ...</td>
</tr>
<tr>
<td></td>
<td>00, 0123, 0777, -045323, ...</td>
</tr>
<tr>
<td></td>
<td>0x0, 0x125, -0xffed, 0xffff</td>
</tr>
<tr>
<td></td>
<td>Literals of type long (64-bit) are denoted by appending L or l to any integer literal.</td>
</tr>
<tr>
<td>Floating point</td>
<td>1.2345, 1234.423, 0.1, -1.23, ...</td>
</tr>
<tr>
<td></td>
<td>By default floating point literals are of type double. If the literal is suffixed with F or f it will be of type float.</td>
</tr>
<tr>
<td>Boolean</td>
<td>true, false</td>
</tr>
<tr>
<td>Characters</td>
<td>’a’, ’A’, ’!’, ...</td>
</tr>
<tr>
<td></td>
<td>’\b’, ’\f’, ’\n’, ’\r’, ’\t’, ’\’, ’''</td>
</tr>
<tr>
<td>Strings</td>
<td>“This is a string”, “Hello World\n”</td>
</tr>
<tr>
<td>Null</td>
<td>null</td>
</tr>
</tbody>
</table>

Assignment

- Declarations associates a type with an identifier
- Assignment associates a value with an identifier
  - Assignment is represented by an = sign
  - An identifier will always be on the left side of the equals sign
  - The computer will place a copy of the thing on the right into the area named by the identifier on the left
- Assignment is not the same as algebraic equality
Operators

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>unary postfix</td>
<td>[], (), ++, --</td>
</tr>
<tr>
<td>unary prefix</td>
<td>++, --, +, --</td>
</tr>
<tr>
<td>creation and cast</td>
<td>new(type)</td>
</tr>
<tr>
<td>multiplicative</td>
<td>*, /</td>
</tr>
<tr>
<td>additive</td>
<td>+, -</td>
</tr>
<tr>
<td>shift</td>
<td>&lt;&lt;, &gt;&gt;, &gt;&gt;&gt; (unsigned right shift)</td>
</tr>
<tr>
<td>relational</td>
<td>&lt;, &gt;, &gt;=, &lt;=, instanceof</td>
</tr>
<tr>
<td>equality</td>
<td>==, !=</td>
</tr>
<tr>
<td>and</td>
<td>&amp;</td>
</tr>
<tr>
<td>xor</td>
<td>^</td>
</tr>
<tr>
<td>or</td>
<td></td>
</tr>
<tr>
<td>boolean and</td>
<td>&amp;&amp;</td>
</tr>
<tr>
<td>boolean or</td>
<td></td>
</tr>
<tr>
<td>conditional</td>
<td>?:</td>
</tr>
<tr>
<td>assignment</td>
<td>+=, -=, *=, /=, %=, &gt;&gt;=, &lt;&lt;=, &gt;&gt;&gt;=, ^=,</td>
</tr>
</tbody>
</table>

Mixed Mode Expressions

- What happens if an expression contains two different types of numbers?
  - int a = 4 + 5 * .56;

- In most cases Java will automatically convert the values in the expression so that it may be evaluated
Automatic Type Conversion

- Java provides a variety of automatic type conversions.
- The following conversions are supported:
  - *Widening primitive conversions.*
    - byte to short, int, long, float, or double.
    - short to int, long, float, or double.
    - int to long, float, or double.
    - long to float or double.
    - float to double.

Manual Type Conversion

- In some situations Java will not perform automatic conversions
  - `int x = 3.1456;`
- In these cases you can force a conversion by specifying a cast
  - `int x = (int)3.1456;`
- Here information is lost, but the assignment will take place
Reference Types

- Reference types are used to declare variables that will \textit{refer} to objects
- The JDK provides a number of classes
  - The \texttt{String} class allows us to declare, create, and manipulate strings
- Declaring a string is no different from declaring a primitive type:
  - \texttt{String name;}

Creating Objects

- Before a reference to an object may be assigned to a variable, the object must be created
  - Operator \texttt{new} is used to create new objects
  - \texttt{String name = new String();}
  - \texttt{String name = new String( "Paul Tymann" );}
  - \texttt{String name = "Paul Tymann";}

References

- Variables refer to objects, they do not contain the object
- Several different variables may all refer to the same object
- If an object is not referred to by any variables, the object will eventually be destroyed

Methods

- A reference to an object can be used to invoke a method of the object
  - The dot (.) operator specifies method invocation
  - String name = “Paul Tymann”;
  - System.out.println( name.substring( 6 ) );
- An attempt to invoke a method using a null reference is an error
What Methods?

- How do you know what methods are available for a given object?
  - Look at the class definition
  - Look at the documentation for the class
- The JDK provides documentation for its classes using a tool called JavaDoc