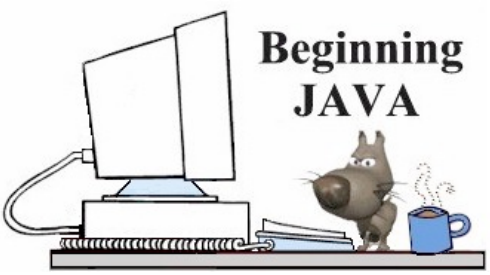
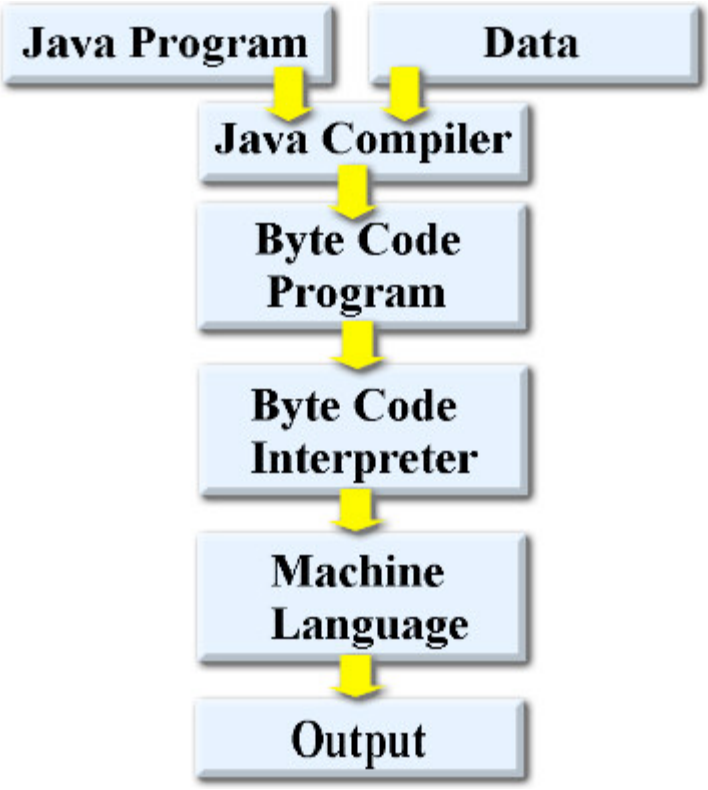


Unit 1 – Introduction

The Mechanics of Creating a Java Program



Java Program: a set of instructions for the computer to follow. The source code.

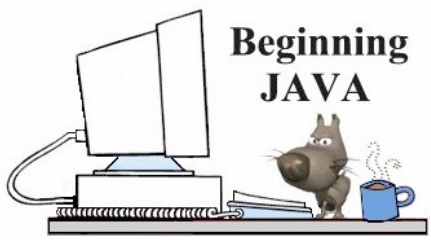
Data: data for the Java program

Java Compiler: translates the program from Java to a language that the computer can understand. Compilers differ by make of machine and operating systems.

Byte Code Program: the compiler translates Java into a language called byte-code. Byte code is the machine language for a hypothetical computer called the Java Virtual Machine.

Byte Code Interpreter: the interpreter translates each instruction of byte code into instructions that the computer in use can execute.

Machine Language: is the language the computer in use understands.



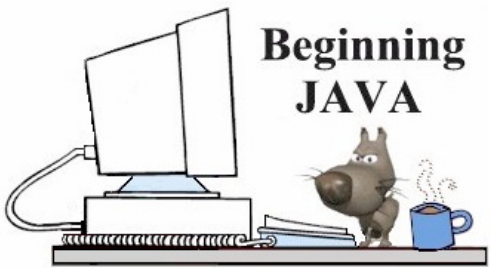
Unit 1

```
/**************************************
 *Project:  LabOneA
 *Programmer:  John Smith
 *Date:  September 23
 *Program Name:  Hello.java
**************************************/
import java.io.*;
public class Hello
{
    public static void main (String[ ] args)
    {
        System.out.println("This is a literal print.");
        System.out.print("Yea!");
        System.out.println("Go Java!");
    }
}
```

Parts of a Java Program

Remember that Java is case sensitive.

The CODE:	Information:
/************************************** *Project: LabOneA *Programmer: John Smith *Date: September 23 *Program Name: Hello.java **************************************/	All programs should begin with a **comment** identifying the purpose of the program and the programmer. /**documentation */ - documentation commenting. /* text*/ - is the format for block commenting. // text - is the format for single line commenting. Comment statements are ignored by the compiler. It is a message to the reader of the code.
import java.io.*;	The import command tells the computer to find **packages** that will be needed in the execution of the program. The java.lang package is the only package imported automatically without an explicit command; all other packages need an import command. This package is used for input and output. (Packages are collections of classes (libraries) which contain portable Java bytecode files.)
public class Hello	Every Java program is a class. **The program starts with the name of the class. This name must be the same name as the .java file in your folder.** In this case the file is saved as Hello.java. Class names must begin with a letter, an underscore or a dollar sign. Class names may contain only letters, digits, underscores and/or dollar signs. Class names may not use reserved words.
{	A set of French curly braces { } is needed for every class.
public static void main (String[ ] args)	A **main method** is needed for execution to have a place to start. This main method gets executed first. The idea of static will be discussed later. The array of strings parameter is a necessity.
{	A set of French curly braces { } is needed for main.
System.out.println("This is a literal print."); System.out.print("Yea!"); System.out.println("Go Java!");	**Output statements:** the basic output statement in Java is the System.out.println( ) statement. The **System.out.println( )** line will print what is between the double quotes" " (called a literal print) and move the printing cursor to the next line. The **System.out.println( )** line will what is between the double quotes and leave the printing cursor on the same line.
} }	Braces are closed to end the program.



Unit 1 - Introduction

Style Issues

Program Format:

```
import java.io.*;
//printing a message on the screen
//notice the format of the code
public class HelloClass
{
    public static void main (String[ ] args)
    {
        System.out.println ("Hello, Java world!");
        System.out.println ("I plan to be a Java expert!");
    }
}
```

Notice the format style that we will be using. The indentations keep the code clearly visible and easy to read.

It is possible, in Java, to write all of your code on one line -- this is called free form style. Free form style is extremely difficult to debug at a later date and is nearly impossible for a programming team to decipher. We will NOT be using free form style.

Case sensitivity:

```
if (netpay > grosspay)
If (NetPay > GrossPay)
IF (NETPAY > GROSSPAY)
```

Java is very picky about your caps lock key. The three lines of code at the left, at first glance, may appear to all say the same thing. The Java compiler, however, will only execute the first line of code. Most Java code is written in smaller case and ALL reserved words (such as "if") MUST be written in smaller case.

Comments:

```
import java.io.*;
//printing another message on the screen
//notice the commented lines
public class HelloAgainClass
{
    public static void main (String[ ] args)
    {
        System.out.println ("Hello!");  //first print
        System.out.println ("I just love this Java!");
    }
}
/*sometimes comments are longer statements that
wrap around to the next line on the screen*/
```

In Java, comments may be expressed in different forms.

The comments beginning with **// are single line comments.** They can appear on a line by themselves, or they may follow other lines of code.

The comments enclosed within **/* and */** are used for longer comments that wrap around a line.

Blank Space:

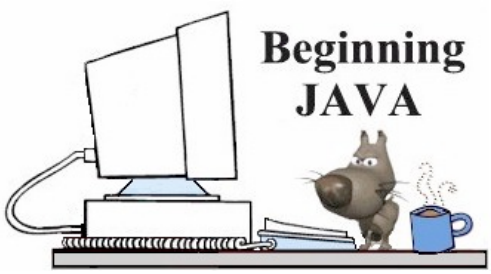
```
import java.io.*;
//notice the spacing in this code
public class HelloStillClass
{
    public static void main (String[ ] args)
    {
       System.out.println ("Java rocks!");

        System.out.println ("A real space cadet!");
    }
}
```

The compiler ignores extra blanks between words and symbols.

Blank lines between lines of code are also ignored. Notice the blank line between the two print statements.

You cannot, however, embed blanks in identifiers.

The use of blanks improves readability.

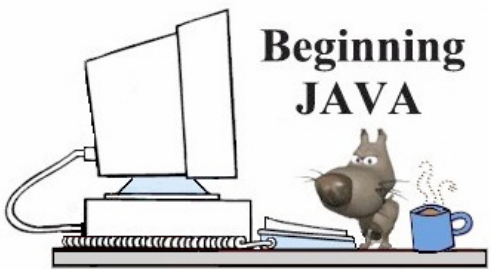


Unit 1 - Introduction

Reserved Words (Keywords) for Java

Here are the reserved words for Java (keywords). You may not redefine any of these reserved words. These words ALWAYS appear in lowercase. You should not give methods or variables the same name as any of these keywords.

abstract	boolean	break	byte
case	catch	char	class
const	continue	default	do
double	else	extends	final
finally	float	for	future
generic	goto	if	implements
import	inner	instanceof	int
interface	long	native	new
null	operator	outer	package
private	protected	public	rest
return	short	static	super
switch	synchronized	this	throw
throws	transient	try	var
void	volatile	while	



Unit 1 - Introduction

Examples of comments

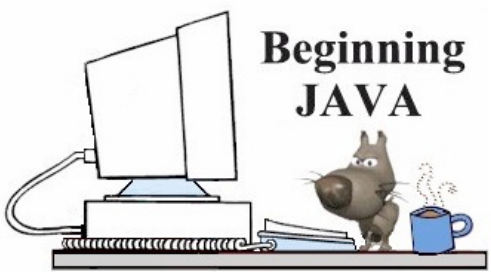
There are 3 styles of comments:	/********** *Project: Demo for Comments *Programmer: Mr. Data *Date: June, 2003 *Program Name: Comments.java **********/

/**documentation */ - documentation commenting.
/* text*/ - is the format for block commenting.
// text - is the format for single line commenting.
 	
Comment statements are ignored by the compiler. Comments are simply messages to the reader of the code.
	import java.io.*;
	public class Comments
	{
	public static void main (String[ ] args)
	{
	System.out.println("Hi"); //message to user
	System.out.println("LabOneA"); //assignment
	}
	}
	/* if you want to write a detailed comment, and it wraps around the screen, use this block style of commenting. */

The CODE:	Information about the code:
/********** *Project: Demo for Comments *Programmer: Mr. Data *Date: June, 2003 *Program Name: Comments.java **********/	All programs should begin with a comment identifying the purpose of the program and the programmer. For our course please use the style shown at the left.

import java.io.*;
public class Comments
{
public static void main (String[ ] args)
{
System.out.println("Hi"); //message to user
System.out.println("LabOneA"); //assignment
}
}
	Comments can also be placed within the body of a program to act as documentation. These comments should be brief and to the point.
	This body prints **Hi!** on the screen and **LabOneA** on the screen.

/* if you want to write a detailed comment, and it wraps around the screen, use this block style of commenting. */	The block form of commenting is used to display lengthy comments that wrap around the screen.



Unit 1 - Introduction

Escape Sequences

The following is a table of **escape sequences** to be used when printing in Java. These statements are embedded within a literal print remark (they go between the quotes):

Sequence	Name	Meaning
\n	New line	Moves to beginning of next line
\b	Backspace	Backs up one character
\t	Horizontal tab	Moves to next tab position Tab spacing is every 8 columns starting with 1. (Columns 9, 17, 25, 33, 41, 49, 57, 65, 73 ...)
\\	Backslash	Displays an actual backslash
\'	Single quote	Displays an actual single quote
\"	Double quote	Displays an actual double quote

System.out.println("\tGeorge\t\tPaul");

will tab, print George, tab twice more, and print Paul.