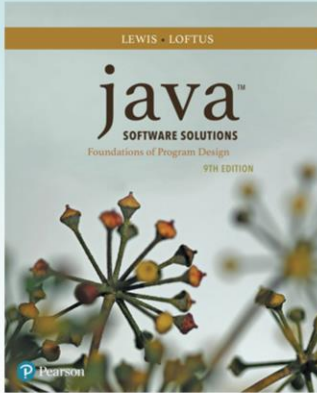


# Chapter 1

## Introduction



### Java Software Solutions Foundations of Program Design 9<sup>th</sup> Edition

John Lewis  
William Loftus

Copyright © 2017 Pearson Education, Inc.

# Outline

**Computer Processing**

**Hardware Components**

**Networks**



**The Java Programming Language**

**Program Development**

**Object-Oriented Programming**

Copyright © 2017 Pearson Education, Inc.

# Java

- The Java programming language was created by Sun Microsystems, Inc.
- It was introduced in 1995 and its popularity has grown quickly since
- A *programming language* specifies the words and symbols that we can use to write a program
- A programming language employs a set of rules that dictate how the words and symbols can be put together to form valid *program statements*

Copyright © 2017 Pearson Education, Inc.

-Previously we discussed how we can communicate with a machine using a binary language (0s and 1s to control tiny transistors)

-We call binary language a “lower-level” language because it is tightly connected and at the level of the actual hardware, binary speaks in “machine” language

-A “higher-level” language is one that resembles more of how we communicate (e.g. if, for, while)

-If we instruct the computer what to do using a “higher-level” language, there must be a mechanism that “translates” this language to the lower-level language used by the computer. We’ll look at such mechanisms called compilers and interpreters later.

-There are many higher-level languages including the one we are studying called the Java language

-Java was originally designed for small TV devices and devices with limited resources and gained popularity when used to write one of the first Web browsers

-Java is an Object-Oriented (OO) language meaning that programs are designed using the notion that all things are objects. We’ll be looking in-depth at these concepts throughout the course.

-Writing in Java involves writing our own “object” types (called classes) and using other classes from a large library called the Java standard class library (or Java API)

# Java Program Structure

- In the Java programming language:
  - A program is made up of one or more *classes*
  - A class contains one or more *methods*
  - A method contains program *statements*
- These terms will be explored in detail throughout the course
- A Java application always contains a method called `main`
- See `Lincoln.java`

Copyright © 2017 Pearson Education, Inc.

-Let's examine the textbook example Listing 1.1 and look at each of the statements closely as discussed in the textbook.

```

//*****
//  Lincoln.java      Author: Lewis/Loftus
//
//  Demonstrates the basic structure of a Java application.
//*****

public class Lincoln
{
    //-----
    //  Prints a presidential quote.
    //-----
    public static void main (String[] args)
    {
        System.out.println ("A quote by Abraham Lincoln:");

        System.out.println ("Whatever you are, be a good one.");
    }
}

```

Copyright © 2017 Pearson Education, Inc.

### Output

```

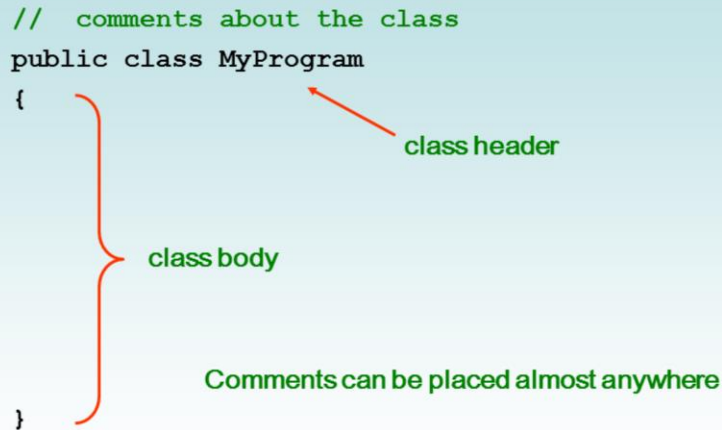
//*****
// Lincoln: A quote by Abraham Lincoln:
// Demons: Whatever you are, be a good one.
//*****

public class Lincoln
{
    //-----
    // Prints a presidential quote.
    //-----
    public static void main (String[] args)
    {
        System.out.println ("A quote by Abraham Lincoln:");
        System.out.println ("Whatever you are, be a good one.");
    }
}

```

# Java Program Structure

```
// comments about the class
public class MyProgram
{
    class body
}
Comments can be placed almost anywhere
```


A diagram illustrating the structure of a Java program. It shows a code snippet with annotations. An arrow points from the text "class header" to the line "public class MyProgram". A large red curly brace on the left side of the code, spanning from the opening curly brace "{" to the closing curly brace "}", is labeled "class body". Another line of text, "Comments can be placed almost anywhere", is placed below the class body.

Copyright © 2017 Pearson Education, Inc.

- First notice the overall structure and how the braces are used to enclose different areas and statements
- Braces are used to enclose statements (lines of words) that are associated together
- The larger braces enclose what we call a “class” defined by the first line before the first braces (public class Lincoln)

# Java Program Structure

```
// comments about the class
public class MyProgram
{
    // comments about the method
    public static void main (String[] args)
    {
    }
}
```



Copyright © 2017 Pearson Education, Inc.

- The smaller braces enclose what we call a “method” within the class defined by (public static void main(String[] arg))
- This method has a special name called “main” which indicates where our actual “main” program should begin. All Java programs have this method to designate where to start processing.
- Once in this main method, the computer reads each line one by one to execute your program
- Note the two lines in the main method
- These are statements that execute another method (println) to print out characters to the screen
- When the computer encounters this method, it executes its instructions just like it does when it encounters our main method
- This process is also termed to “call” a method or to “invoke” the method
- After processing the statements in println, the computer comes back to our main method and processes the next statement
- Recall that these instructions are what was loaded in memory and executed by the CPU one statement at a time!



# Comments

- Comments should be included to explain the purpose of the program and describe processing steps
- They do not affect how a program works
- Java comments can take three forms:

```
// this comment runs to the end of the line
```

```
/* this comment runs to the terminating  
   symbol, even across line breaks */
```

```
/** this is a javadoc comment */
```

Copyright © 2017 Pearson Education, Inc.

- Comments help you and others describe what you are trying to do
- This form of documentation is very useful and required for our assignments
- Habits writing good comments will pay dividends down the road!

# Identifiers

- Sometimes the programmer chooses the identifier (such as `Lincoln`)
- Sometimes we are using another programmer's code, so we use the identifiers that he or she chose (such as `println`)
- Often we use special identifiers called *reserved words* that already have a predefined meaning in the language
- A reserved word cannot be used in any other way

Copyright © 2017 Pearson Education, Inc.

-Identifiers: words used to write our programs

-Cannot use words that are already “reserved” words (e.g. `class`, `public`, ...)

# Identifiers

- *Identifiers* are the "words" in a program
- A Java identifier can be made up of letters, digits, the underscore character ( `_` ), and the dollar sign
- Identifiers cannot begin with a digit
- Java is *case sensitive*: `Total`, `total`, and `TOTAL` are different identifiers
- By convention, programmers use different case styles for different types of identifiers, such as
  - *title case* for class names - `Lincoln`
  - *upper case* for constants - `MAXIMUM`

Copyright © 2017 Pearson Education, Inc.

- Good to get in the habit of using naming conventions for our "words"
- Oftentimes software development companies enforce their own
- Get used to knowing and using conventions

# Reserved Words

- The Java reserved words:

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

Copyright © 2017 Pearson Education, Inc.

## Quick Check

Which of the following are valid Java identifiers?

grade  
quizGrade  
NetworkConnection  
frame2  
3rdTestScore  
MAXIMUM  
MIN\_CAPACITY  
student#  
Shelves1&2

Copyright © 2017 Pearson Education, Inc.

## Quick Check

Which of the following are valid Java identifiers?

<code>grade</code>	Valid
<code>quizGrade</code>	Valid
<code>NetworkConnection</code>	Valid
<code>frame2</code>	Valid
<code>3rdTestScore</code>	Invalid – cannot begin with a digit
<code>MAXIMUM</code>	Valid
<code>MIN_CAPACITY</code>	Valid
<code>student#</code>	Invalid – cannot contain the '#' character
<code>Shelves1&amp;2</code>	Invalid – cannot contain the '&' character

Copyright © 2017 Pearson Education, Inc.

## White Space

- Spaces, blank lines, and tabs are called *white space*
- White space is used to separate words and symbols in a program
- Extra white space is ignored
- A valid Java program can be formatted many ways
- Programs should be formatted to enhance readability, using consistent indentation
- See `Lincoln2.java` and `Lincoln3.java`

Copyright © 2017 Pearson Education, Inc.

-Like naming conventions, software development companies often have conventions for how to use white space.