

# Data in a Cart



# Objectives

After completing this lesson, you should be able to:

- Describe the purpose of a variable in the Java language
- List and describe four data types
- Declare and initialize `String` variables
- Concatenate `String` variables with the '+' operator
- Make variable assignments
- Declare and initialize `int` and `double` variables
- Modify variable values by using numeric operators
- Override default operator precedence using ( )



# Topics

- Introducing variables
- Working with `String` variables
- Working with numbers
- Manipulating numeric data



# Variables

- A variable refers to something that can change.
  - Variables can be initiated with a value.
  - The value can be changed.
  - A variable holds a specific type of data.

The diagram illustrates the components of a variable declaration and assignment in Java. It features a yellow rounded rectangle containing two lines of code. Above the rectangle, three blue labels with vertical lines pointing to specific parts of the code are present: 'The **type** of data' points to 'String', 'Variable name' points to 'firstName', and 'The **value** of the variable' points to '"Mary"' in the first line. The code inside the rectangle is as follows:

```
String firstName = "Mary";  
  
firstName = "Gary";
```

# Variable Types

- Some of the types of values a variable can hold:
  - `String` (example: "Hello")
  - `int` (examples: -10, 0, 2, 10000)
  - `double` (examples: 2.00, 99.99, -2042.00009)
  - `boolean` (true or false)
- If uninitialized, variables have a default value:
  - `String: null`
  - `int: 0`
  - `double: 0.0`
  - `boolean: false`

# Naming a Variable

## Guidelines:

- Begin each variable with a lowercase letter. Subsequent words should be capitalized:
  - `myVariable`
- Names are case-sensitive.
- Names cannot include white space.
- Choose names that are mnemonic and that indicate to the casual observer the intent of the variable.
  - `outOfStock` (a boolean)
  - `itemDescription` (a String)

# Java SE 9: The Underscore Character Is Not a Legal Name

- If you use the underscore character ("\_") as a one-character identifier in source code, then your code won't compile in Java SE 9.
- For example:

```
9  as of release 9, '_' is a keyword, and may not be used as an identifier
10  ----
11  (Alt-Enter shows hints)
12  {
13      int _ = 10;
14      System.out.println("Value of underscore ( ) = " + _);
15  }
16
17
```



# Uses of Variables

- Holding data used within a method:

```
String name = "Sam" ;  
double price = 12.35;  
boolean outOfStock = true;
```

- Assigning the value of one variable to another:

```
String name = name1;
```

- Representing values within a mathematical expression:

```
total = quantity * price ;
```

- Printing the values to the screen:

```
System.out.println(name) ;
```

# Topics

- Introducing variables
- **Working with `String` variables**
- Working with numbers
- Manipulating numeric data



# Examples: Variable Declaration and Initialization

- Basic Example :

```
String address = "123 Oak St";           //one variable declared
// and initialized
```

*type*   *identifier*   *value*

- Other Examples:

```
String customer;                          //One variable declared

String name, city                          //Two variables declared

String country ="USA", state="CO" //Two variables declared
//and initialized

city=" USA";                               //One variable initialized after
//being declared earlier
```

# String Concatenation

- String **variables** can be combined using the '+' operator.
  - `stringVariable1 + stringVariable2`
  - `stringVariable1 + "String literal"`
  - `stringVariable1 + "String literal" + stringVariable2`

- **Example:**

```
String greet1 = "Hello";  
String greet2 = "World";  
String message = greet1 + " " + greet2 + "!";  
String message = greet1 + " " + greet2 + " " + 2014 + "!";
```

# String Concatenation

You can concatenate `String` variables outside or inside a method call:

```
String greet1 = "Hello";  
String greet2 = "World";  
String message = greet1 + " " +greet2  + "!";  
  
System.out.println(message);  
System.out.println(greet1 + " " + greet2 + "!");
```

Output:

```
Hello World!  
Hello World!
```

# Quiz



Which of the following variable declarations and/or initializations are correct?

- a. `int count = 5; quantity = 2;`
- b. `string name, label;`
- c. `boolean complete = "false";`
- d. `boolean complete = true;`



# Topics

- Introducing variables
- Working with `String` variables
- Working with numbers
- Manipulating numeric data



# int and double Values

- `int` variables hold whole number values between:
  - `-2,147,483,648`
  - `2,147,483,647`
  - Examples: 2, 1343387, 1\_343\_387
- `double` variables hold larger values containing decimal portions.
  - Use when greater accuracy is needed.
  - Examples: 987640059602230.7645 , -1111, 2.1E12

# Initializing and Assigning Numeric Values

- `int` variables:

- `int quantity = 10;`
- `int quantity = 5.5;`



Compilation fails!

- `double` variables:

- `double price = 25.99;`
- `double price = 75;`



Run time will  
interpret as 75.0.

# Topics

- Introducing variables
- Working with `String` variables
- Working with numbers
- Manipulating numeric data



# Standard Mathematical Operators

Purpose	Operator	Example	Comments
Addition	+	<code>sum = num1 + num2;</code>	If num1 is 10 and num2 is 2, sum is 12.
Subtraction	-	<code>diff = num1 - num2;</code>	If num1 is 10 and num2 is 2, diff is 8.
Multiplication	*	<code>prod = num1 * num2;</code>	If num1 is 10 and num2 is 2, prod is 20.
Division	/	<code>quot = num1 / num2;</code>	If num1 is 31 and num2 is 6, quot is 5. The remainder portion is discarded. Division by 0 throws an exception.

# Increment and Decrement Operators (++ and --)

The long way:

```
age = age + 1;
```

or

```
count = count - 1;
```

The short way:

```
age++;
```

or

```
count--;
```

# Operator Precedence

Here's an example of the need for rules of precedence.

Is the answer to the following problem 34 or 9?

```
int c = 25 - 5 * 4 / 2 - 10 + 4;
```

# Operator Precedence

Rules of precedence:

1. Operators within a pair of parentheses
2. Increment and decrement operators (++ or --)
3. Multiplication and division operators, evaluated from left to right
4. Addition and subtraction operators, evaluated from left to right

# Using Parentheses

## Examples:

```
int c = (((25 - 5) * 4) / (2 - 10)) + 4;  
int c = ((20 * 4) / (2 - 10)) + 4;  
int c = (80 / (2 - 10)) + 4;  
int c = (80 / -8) + 4;  
int c = -10 + 4;  
int c = -6;
```

# Quiz



Which of the following statements are correct Java code?

- a. `int count = 11.4;`
- b. `double amount = 11.05;`
- c. `int cost = 133_452_667;`
- d. `double total = 1.05 * amount;`



# Quiz



Given:

```
String name = "Bob";  
String msg;  
int num = 3;
```

Which of the following statements correctly assigns the value “Bob wrote 3 Java programs.” to the msg variable?

- a. `msg = name + " wrote " + num + " Java programs.";`
- b. `msg = name + " wrote " + 3 + " Java programs.";`
- c. `msg = "Bob wrote " + (2+1) + " Java programs.";`
- d. `msg = name + " wrote " + 2+1 + " Java programs.";`



# Summary

In this lesson, you should have learned how to:

- Describe the purpose of a variable in the Java language
- List and describe four data types
- Declare and initialize `String` variables
- Concatenate `String` variables with the '+' operator
- Make variable assignments
- Declare and initialize `int` and `double` variables
- Modify numeric values by using operators
- Override default operator precedence using ( )

