Session 3: Data

Objectives

After completing this lesson, you should be able to

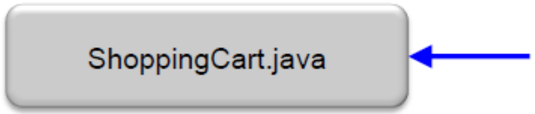
- Describe the purpose of a variable in the Java language
- List and describe four data types
- Declare and initialize String variables
- Concatenate String variables with the '+' operator
- Make variable assignments
- Declare and initialize int and double variables
- Modify variable values by using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

Topics

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

Java Classes

- A Java class is the building block of a Java application.

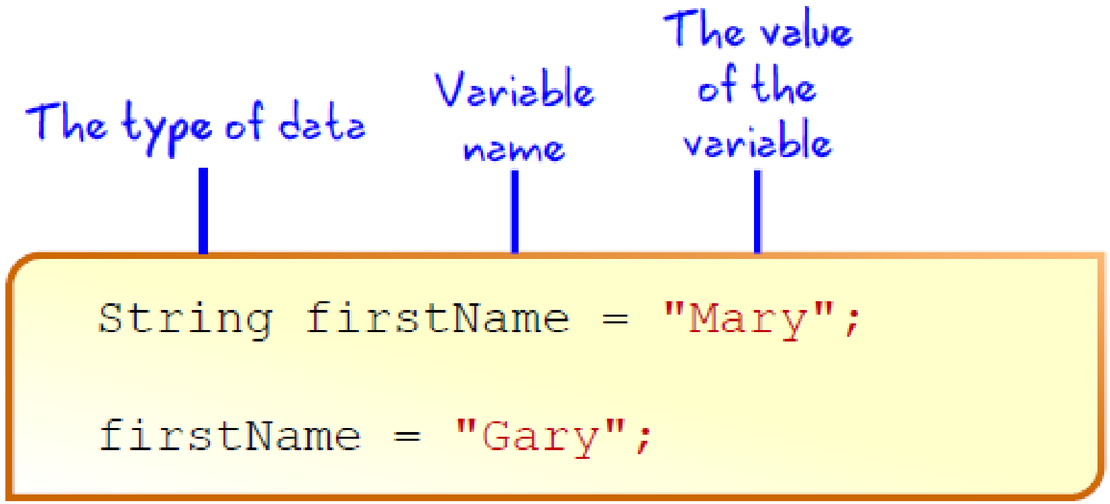


Includes code that

- Allows a customer to add items to the shopping cart
- Provides visual confirmation to the customer

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.



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)

Uses of Variables

- Holding data used in 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 ;
```

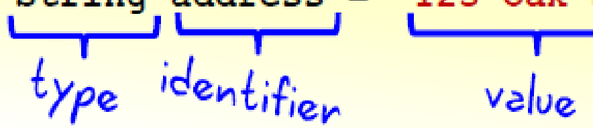
  - Printing the values to the screen:

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

Working with String Variables

Basic Example

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



Other Exmaples

```
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

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!
```

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!
```

Working with Numbers

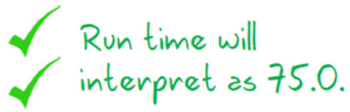
int and double Values

- 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 and double Values

- int variables:
  - int quantity = 10;
  - int quantity = 5.5;
- double variables:
  - double price = 25.99;



Standard Mathematical Operators

Purpose	Operator	Example	Comments
Addition	+	`sum = num1 + num2;`	If num1 is 10 and num2 is 2, sum is 12.
Subtraction	-	`diff = num1 - num2;`	If num1 is 10 and num2 is 2, diff is 8.
Multiplication	*	`prod = num1 * num2;`	If num1 is 10 and num2 is 2, prod is 20.
Division	/	`quot = num1 / num2;`	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

Rules of precedence:

- Operators within a pair of parentheses
- Increment and decrement operators (++ or --)
- Multiplication and division operators, evaluated from left to right
- 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;
```

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;
```