JAVA BASICS

2. Language Introduction

2. LANGUAGE INTRODUCTION

Agenda

- Variables, Data Types
- Arrays
- Control Flow
  - code block
  - condition
  - loops
  - switch
- Operators
- Methods
- Class Definition

2. LANGUAGE INTRODUCTION

Motivation

- What are your experiences using other programming languages? Which data types do you know
- Concerning data types - what's the difference between Java and Javascript?
- Making calculations with big whole numbers what should you consider closely?
- Making calculations decimal numbers (e.g. money), what should you be aware of?

Variables

- A variable:
  - holds a value
  - has a data type
  - is created in a declaration statement
- coding convention for naming:
  - starting with lower case
  - using camel case

Variables - Primitive Data Types

```
type variableName = value;
```

- **byte** whole numbers from -128 to 127
- **short** whole numbers from -32,768 to 32,767
- **int** whole numbers from -2,147,483,648 to 2,147,483,647
- **long** whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
- **float** floating point numbers. Sufficient for storing 6 to 7 decimal digits
- **double** floating point numbers. Sufficient for storing 15 to 16 decimal digits
- **boolean** true or false values
- **char** a single character/letter or ASCII values

Variables - Non-primitive pre-defined data types

- **String** stores text
- **BigDecimal** stores decimal numbers with a defined precision
- **BigInteger**

Examples

```
int myNum = 5;               // Integer (whole number)
float myFloatNum = 5.99f;    // Floating point number
char myLetter = 'D';         // Character
boolean myBool = true;       // Boolean
String myText = "Hello";     // String

String x = "10";
int y = 20;
String z = x + y;  // z will be 1020 (a String)
```

Arrays

- can have one or more dimensions

```
string[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
int[][] coordinates = {{1, 2}, {2,3}};
```

> more information can be find here

Operators

https://www.w3schools.com/java/java_operators.asp

Arithmetic Operators

- `+` Addition: Adds together two values x + y
- `-` Subtraction: Subtracts one value from another x - y
- `*` Multiplication: Multiplies two values x * y
- `/` Division: Divides one value by another x / y
- `%` Modulus: Returns the division remainder x % y
- `++` Increment: Increases the value of a variable by 1 ++x
- `--` Decrement: Decreases the value of a variable by 1 --x

Assignment Operators

Operator	Example	Same as
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
&=	x &= 3	x = x & 3
...	...	...

Comparision Operators

Operator	Example	Same as
==	Equal to	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical Operators

- **&&** Logical and Returns true if both statements are true `x < 5 &&  x < 10`
- **||** Logical or Returns true if one of the statements is true `x < 5 || x < 4`
- **!** Logical not Reverse the result, returns false if the result is true `!(x < 5 && x < 10)`

Control flow - condition

see https://www.w3schools.com/java/java_conditions.asp

```
if (condition1) {
  // block of code to be executed if condition1 is true
} else if (condition2) {
  // block of code to be executed if the condition1 is false and condition2 is true
} else {
  // block of code to be executed if the condition1 is false and condition2 is false
}
```

Control flow - switch

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

Control flow - loop

for each as example

```
String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
for (String i : cars) {
  System.out.println(i);
}
```

Methods, Parameters and structuring your code

- methods
  - are code block which only runs when they are called
  - use parameters to pass data into methods
  - can have a return value or void
  - are declared in a class or struct
- use methods in order to reuse code

Methods, Parameters and structuring your code

Example:

```
class SimpleMathExtension
{
    public int divideTwoNumbers(int number1, int number2)
    {
        return number1 / number2;
    }
}
```

Definition:

```
class Class_Name
{
    <access modifier> <return type> methodName(Parameters)
    {
        //method statements
        return Return_Value;
    }
}
```

Main Method

Entry method which is called after start of the app

```
class SimpleMathExtension
{
    public static void Main(String[] args)
    {
      int result = divideTwoNumbers(9, 3);
      System.out.println(result);
    }

    public int divideTwoNumbers(int number1, int number2)
    {
        return number1 / number2;
    }
}
```

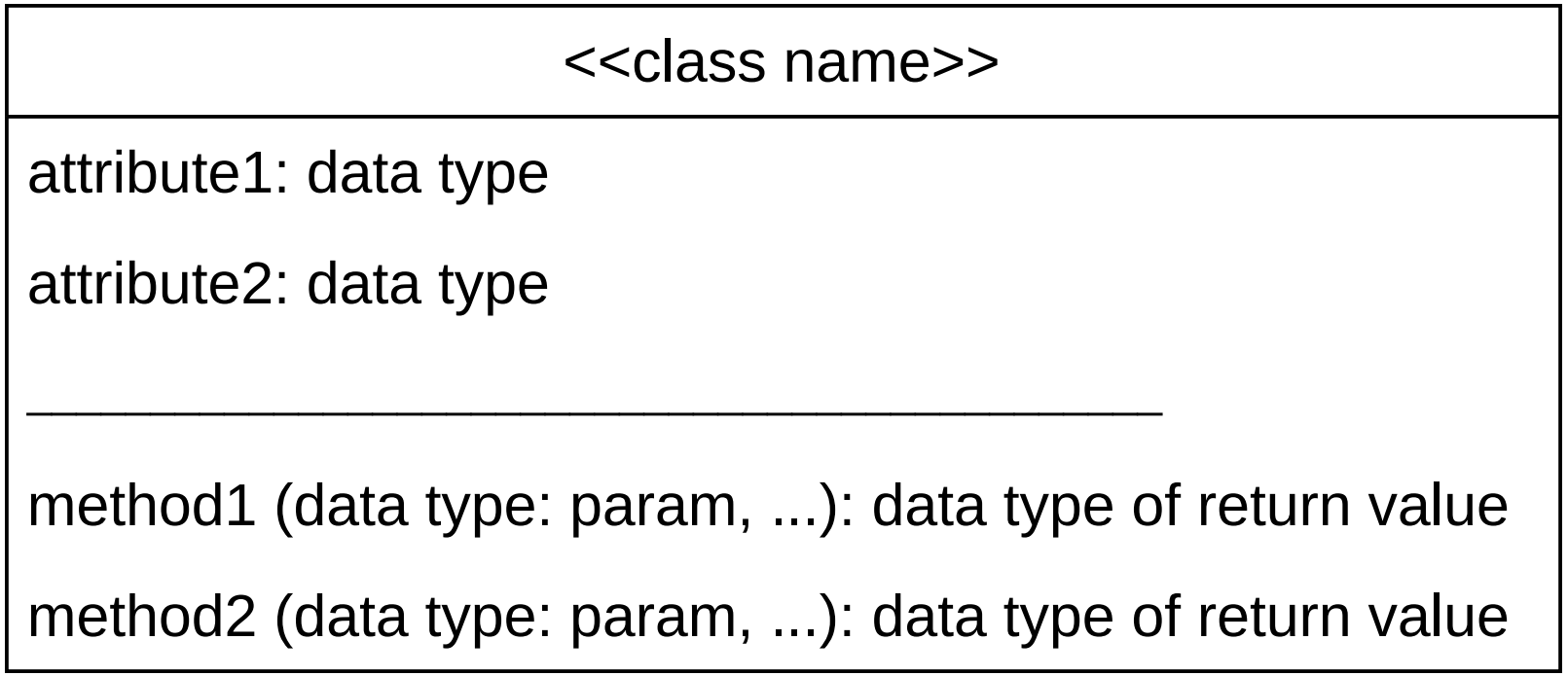
Method Overloading

```
/**
* Methods has the same name but different parameters
**/
class MethodOverloadExample
{
    public int addNumbers(int number1, int number2)
    {
        return number1 + number2;
    }

    public int addNumbers(int number1, int number2, int number3)
    {
        return number1 + number2 + number3;
    }

    public double addNumbers(double number1, double number2)
    {
        return number1 + number2;
    }
}
```

Local Scope Variables

```
public int divideTwoNumbers(int number1, int number2)
{
    //local scoped variable - only available in method
    int returnValue = number1 / number2;
    return returnValue;
}
```

Class Definition



Quiz

- Name a data type for: whole numbers, decimal numbers, text, and characters.
- Name arithmetic operators. What other types of operators exist?
- What is the name of the method that is executed when a project starts?
- What are local variables?
- What three elements make up the signature of a method?
- How can a class be used? What needs to be done for that?

2. LANGUAGE INTRODUCTION

see readme for tasks