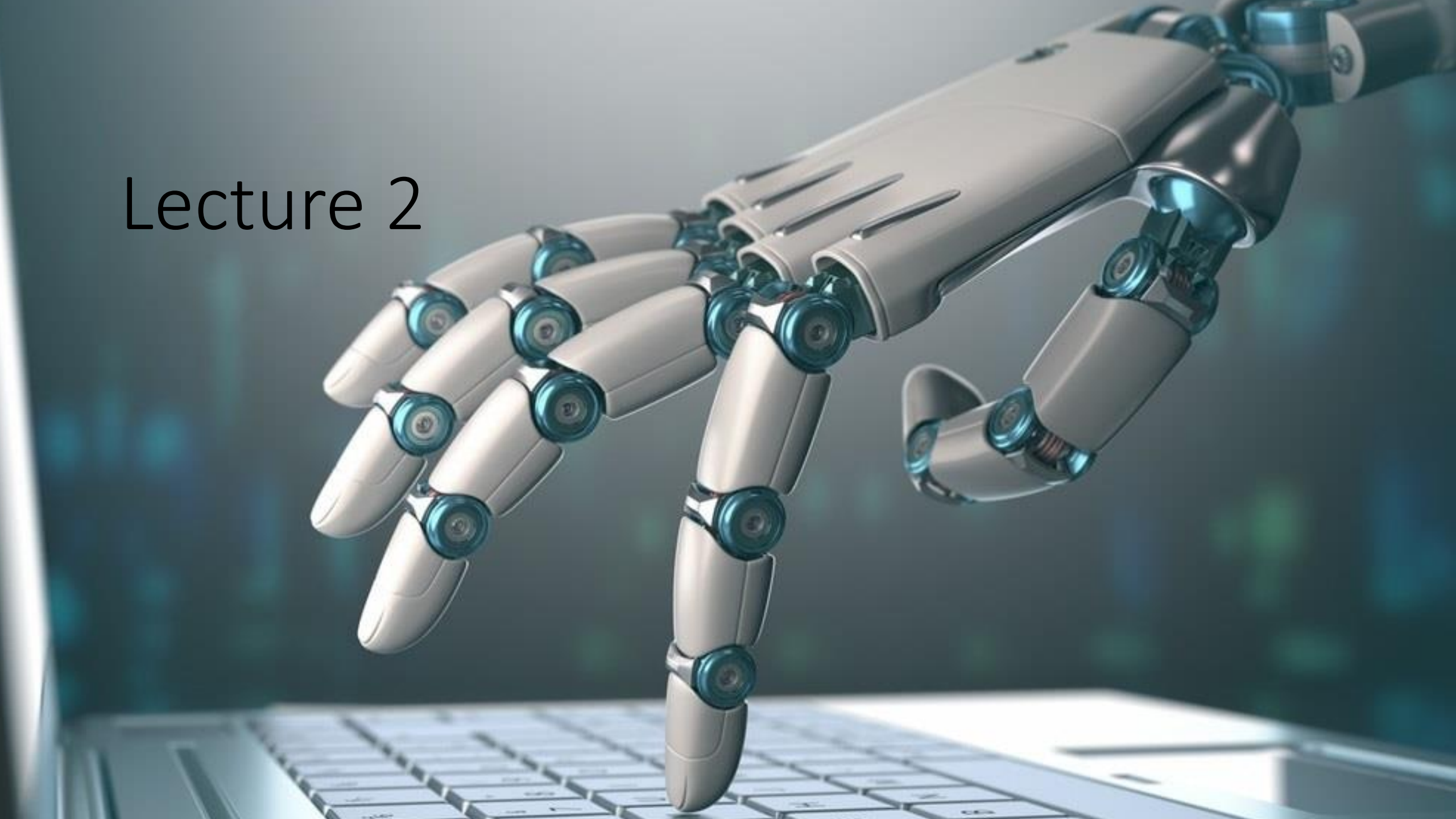
Lecture 2



Agenda

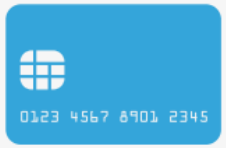
- **Classes**
- Methods
- Refactoring
- Practice

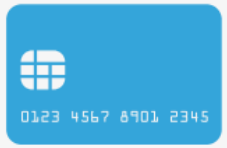
PRIMARY CONCEPTS: CLASS AND OBJECT

- Class describes template (blueprint) of something with state and behaviour
- Object is concrete instance of that class with set state

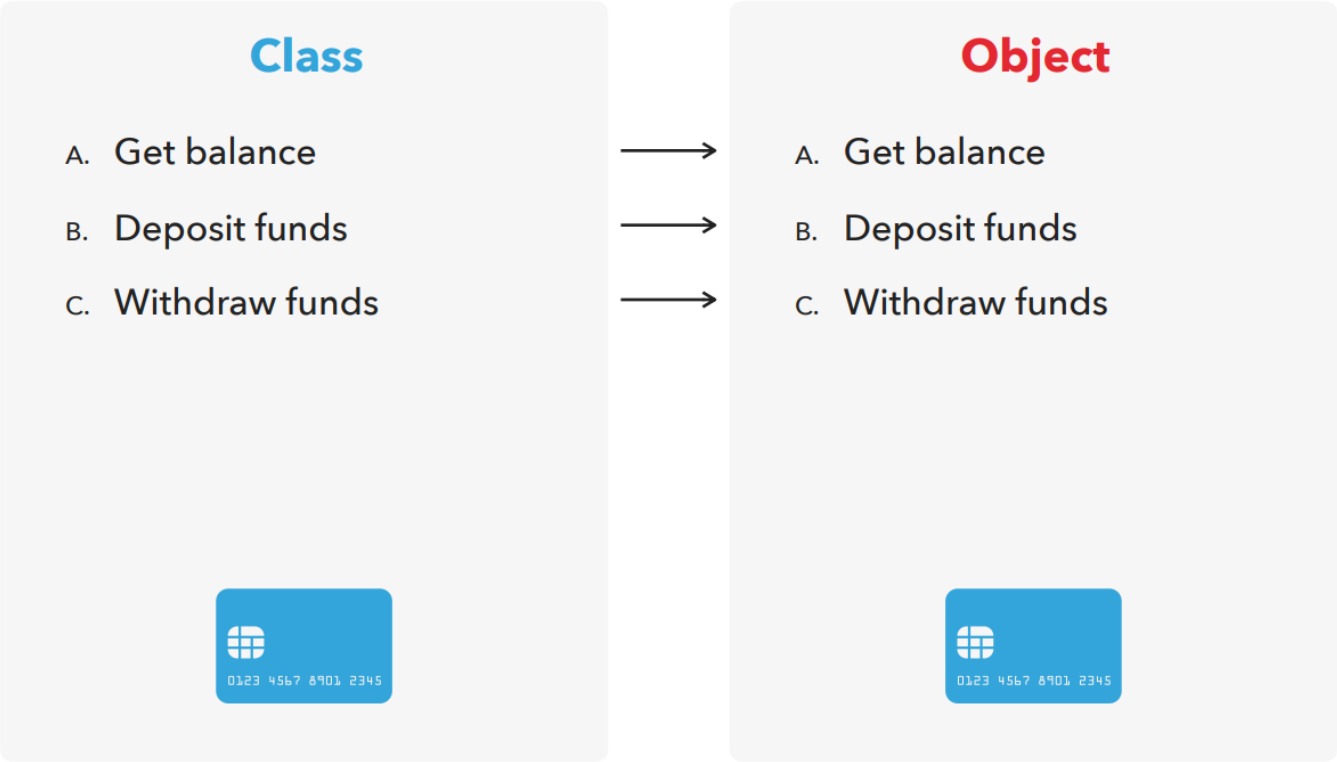
EXAMPLE: BANK CARD (STATE)

Class		Object
A. Bank Name	→	A. Citadele Banka
B. Payments Processor	→	B. Master Card
C. Name on Card	→	C. John Doe
D. Card Number	→	D. 5224 9989 7556 2871
E. Expiration Date	→	E. 12/2022
F. Security Code	→	F. 218





EXAMPLE: BANK CARD (BEHAVIOUR)



CLASS DECLARATION IN JAVA: SYNTAX

```
class ClassName {

    type variable1;
    type variable2;
    ...
    type variableN;

    method1() {}
    method2() {}
    ...
    methodN() {}

}
```

CLASS DECLARATION IN JAVA: EXAMPLE BREAKDOWN

Special keyword

Class name

```
class Window {

    String title;
    double height;

    void open() {

    }

    void close() {

    }

    void setHeight() {

    }

}
```

Class variables (fields) defining state

Class methods defining behaviour

OBJECT INSTANTIATION IN JAVA: SYNTAX

- Object instantiation without assignment

```
new Class();
```

- Object instantiation with assignment

```
Class var = new Class();
```

OBJECT INSTANTIATION IN JAVA: SYNTAX

- Object instantiation without assignment

```
new Window();
```

- Object instantiation with assignment

```
Window firstWindow = new Window();
```

THREE-STEP PROCESS OF OBJECT CREATION

1. Declaration - object variable declaration of a class type
2. Instantiation - the process of creating an object with new operator
3. Initialisation - the process of object construction by setting its initial state

CONSTRUCTORS

- Every class has a constructor
- If explicit constructor(s) is not specified in code, Java Compiler will generate default constructor implicitly
- Each time a new object is created, at least one constructor will be invoked
- Each defined constructor must have unique signature (i.e. ordered number and type of arguments)

CONSTRUCTOR DECLARATION IN JAVA: EXAMPLE BREAKDOWN

Explicit default constructor without arguments

```
public class Window {

    private String title;

    public Window() {

    }

    public Window(String title) {
        this.title = title;
    }

}
```

Explicit constructor with argument and initialisation

Agenda

- Classes
- **Methods**
- Refactoring
- Practice

METHOD DEFINITION

- Java method is a collection of statements that are grouped together to perform an operation
  - Invoking *System.out.println()* method actually executes several statements in order to display a message on the console
- Describes behaviour of class or actions that object can perform
- Method either produces output or not

METHOD DECLARATION IN JAVA: SYNTAX

Defines the access type of the method

Defines method name

```
modifier returnType methodName (arg1, arg2, ..., argN) {

    //body

}
```

Specifies return type for methods producing output

List of parameters, it is type, order and number

METHOD DECLARATION IN JAVA: WITH RETURN EXAMPLE

Can be accessed from everywhere

Result type

Method name

Method accepts two integer parameters

```
public int sum(int a, int b) {
    int result = a + b;
    return result;
}
```

Operation with parameters

Keyword with variable that returns value back to the invoker

METHOD DECLARATION IN JAVA: WITHOUT RETURN EXAMPLE

Method name

Keyword meaning no return value

Method accepts two integer parameters

```
public void printSum(int a, int b) {
    int result = a + b;
    System.out.println("Sum of numbers is" + result);
}
```

Operations with parameters

A BIT MORE ABOUT RETURNING RESULT

- After completion method returns to the code that invoked it
- Whether method returns value or not is declared in method signature
  - When type is void - return statement is unnecessary, however can be stated
  - Other type - return statement is necessary

ACCESSING AND CHANGING OBJECT STATE: GETTERS & SETTERS

- In OOP another party should not be able to access object state directly
- To keep things safe, one can
  - Retrieve object state via get methods (getters)
  - Change object state via set methods (setters)

GETTERS & SETTERS DECLARATION

```
public class Person {

    private String name;
    private int age;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }
}
```

Getters

Setters

GETTERS & SETTERS USAGE

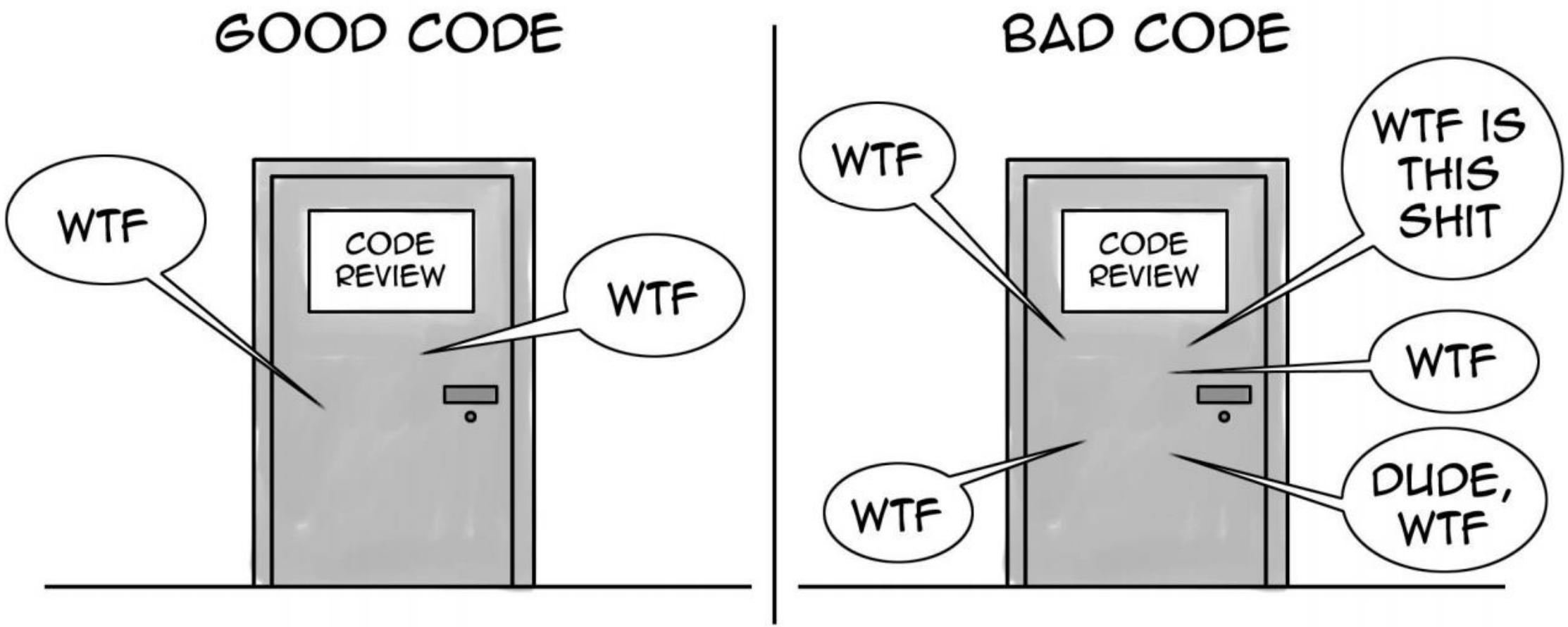
```java
public class PersonTest {

    public static void main(String[] args) {

        Person person = new Person();
        person.setName("John Doe");
        person.setAge(32);

        String personName = person.getName();
        int personAge = person.getAge();

        System.out.println("His name is " + personName);
        System.out.println("He is " + personAge + " years old");

    }

}
```

Agenda

- Classes
- Methods
- **Refactoring**
- Practice

ANY FOOL CAN WRITE CODE THAT COMPUTER UNDERSTAND. GOOD PROGRAMMERS WRITE CODE THAT HUMANS CAN UNDERSTAND.

Martin Fowler



THE ONLY VALID MEASUREMENT OF CODE QUALITY: WTFs/MINUTE

BAD CODE AND GOOD CODE

Bad

```java
public class Cat {

    private String n;

    public String getN() {
        return n;
    }

    public void setN(String n) {
        this.n = n;
    }

    public void v() {
        System.out.println("Meow");
    }

}
```

Good

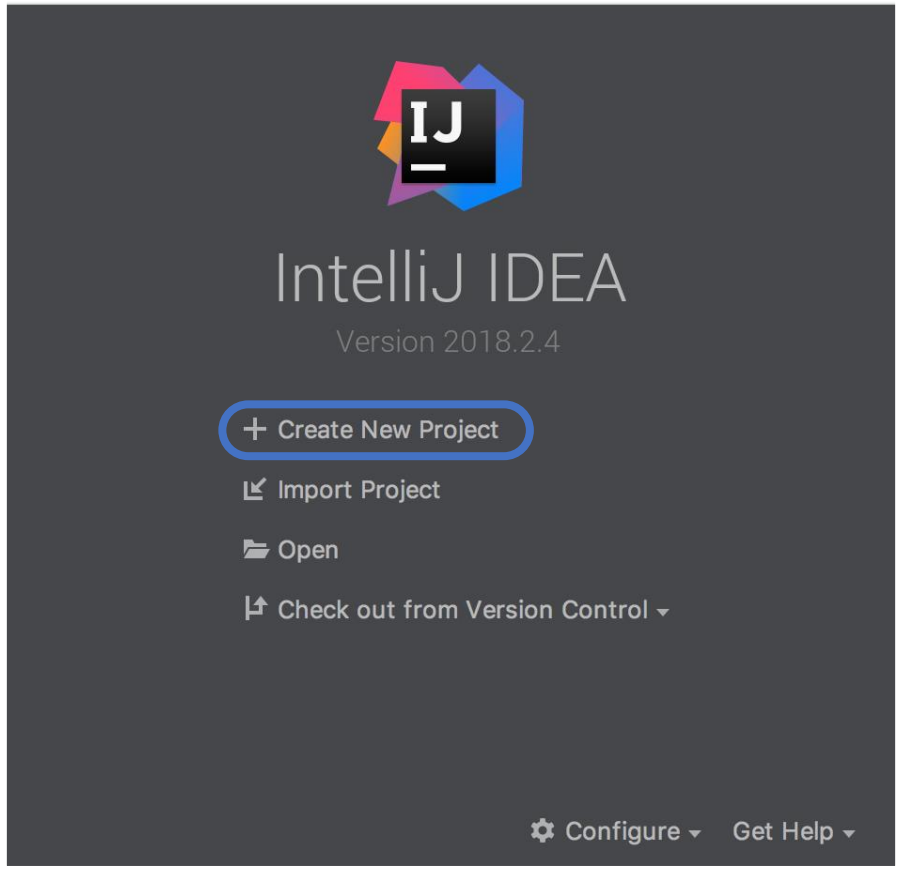
```java
public class Cat {

    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void voice() {
        System.out.println("Meow");
    }

}
```

Agenda

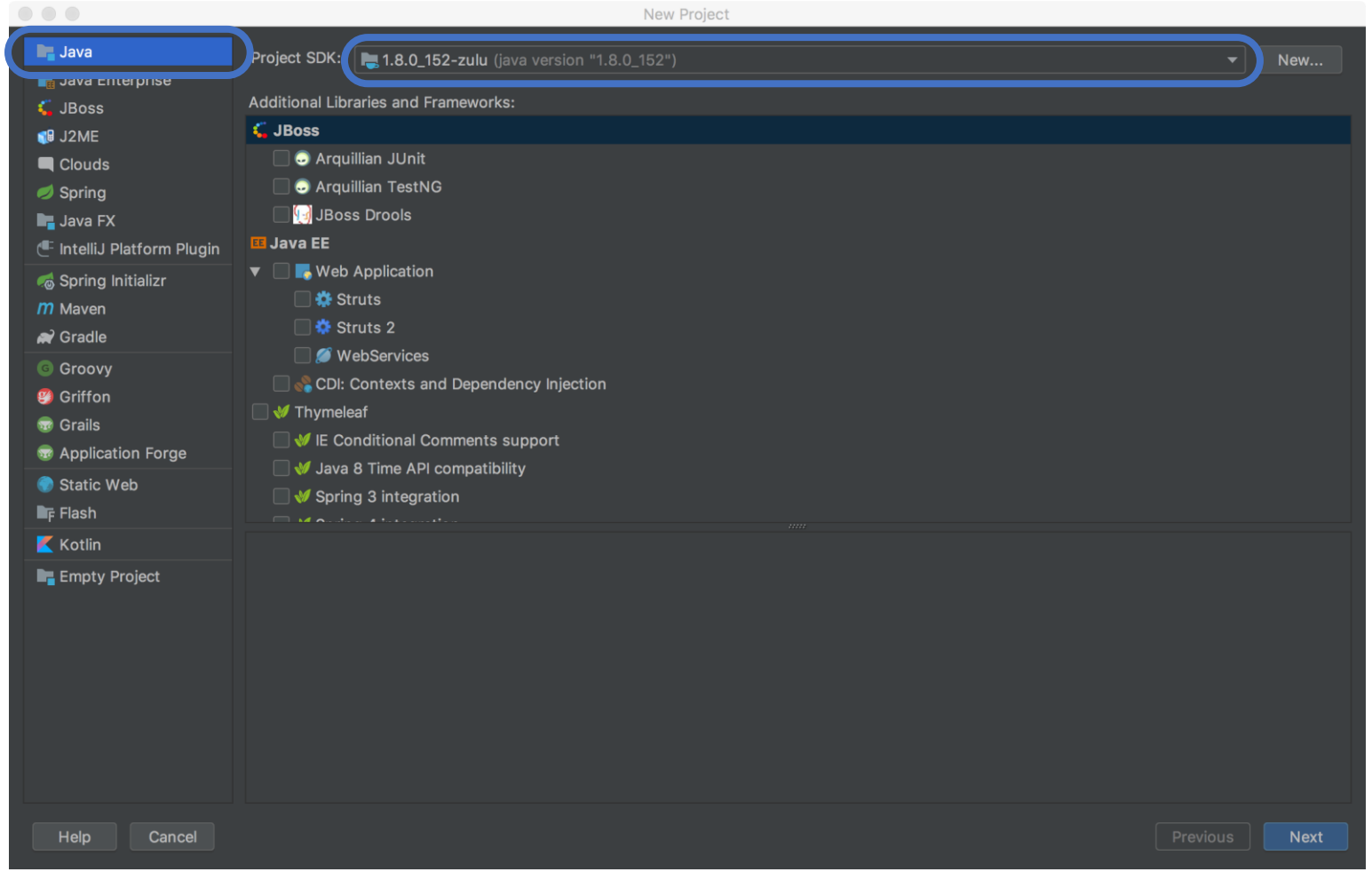
- Classes
- Methods
- Refactoring
- **Practice**

Creating project overview: run intellij idea

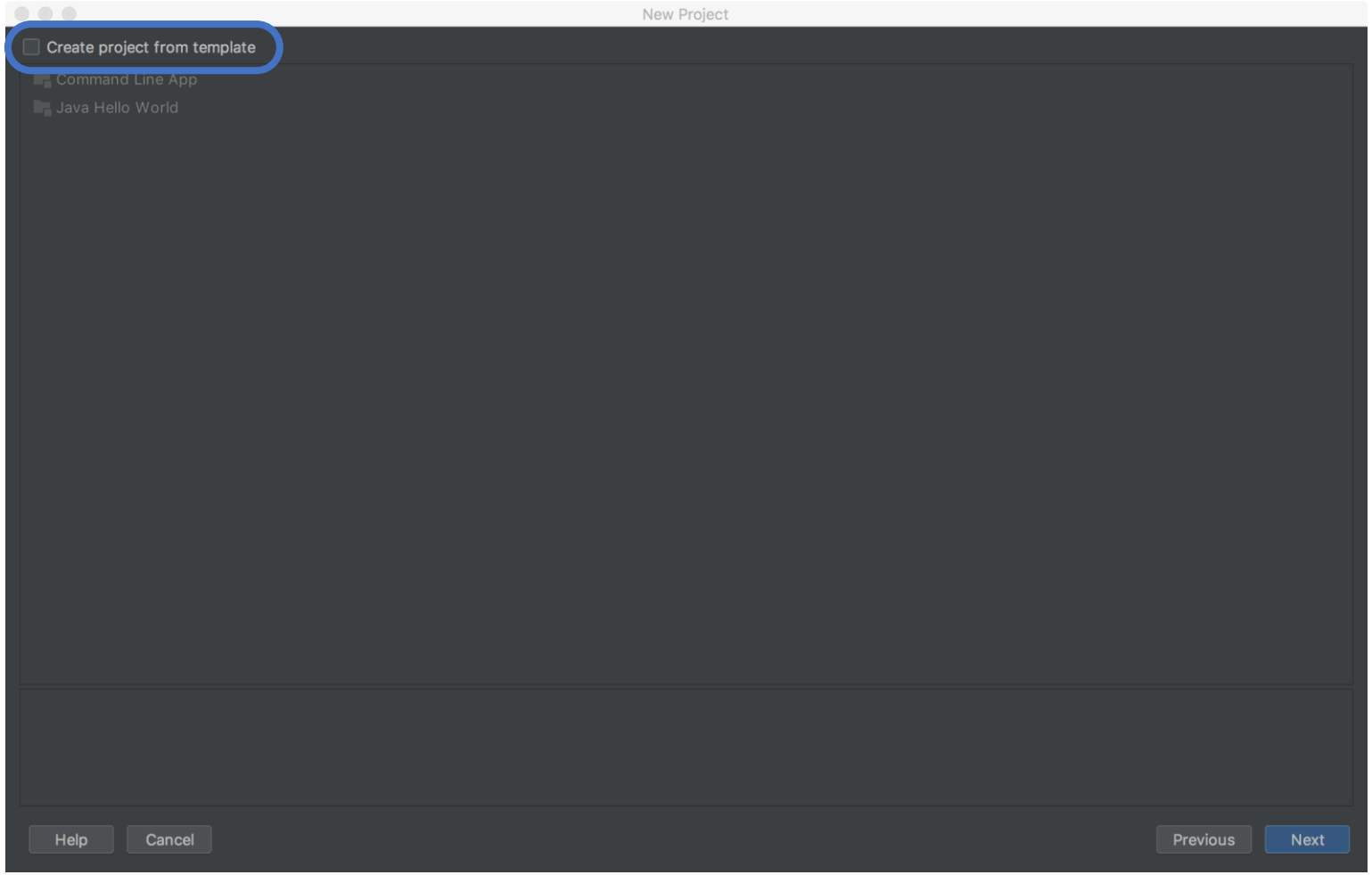
Creating project overview: welcome screen



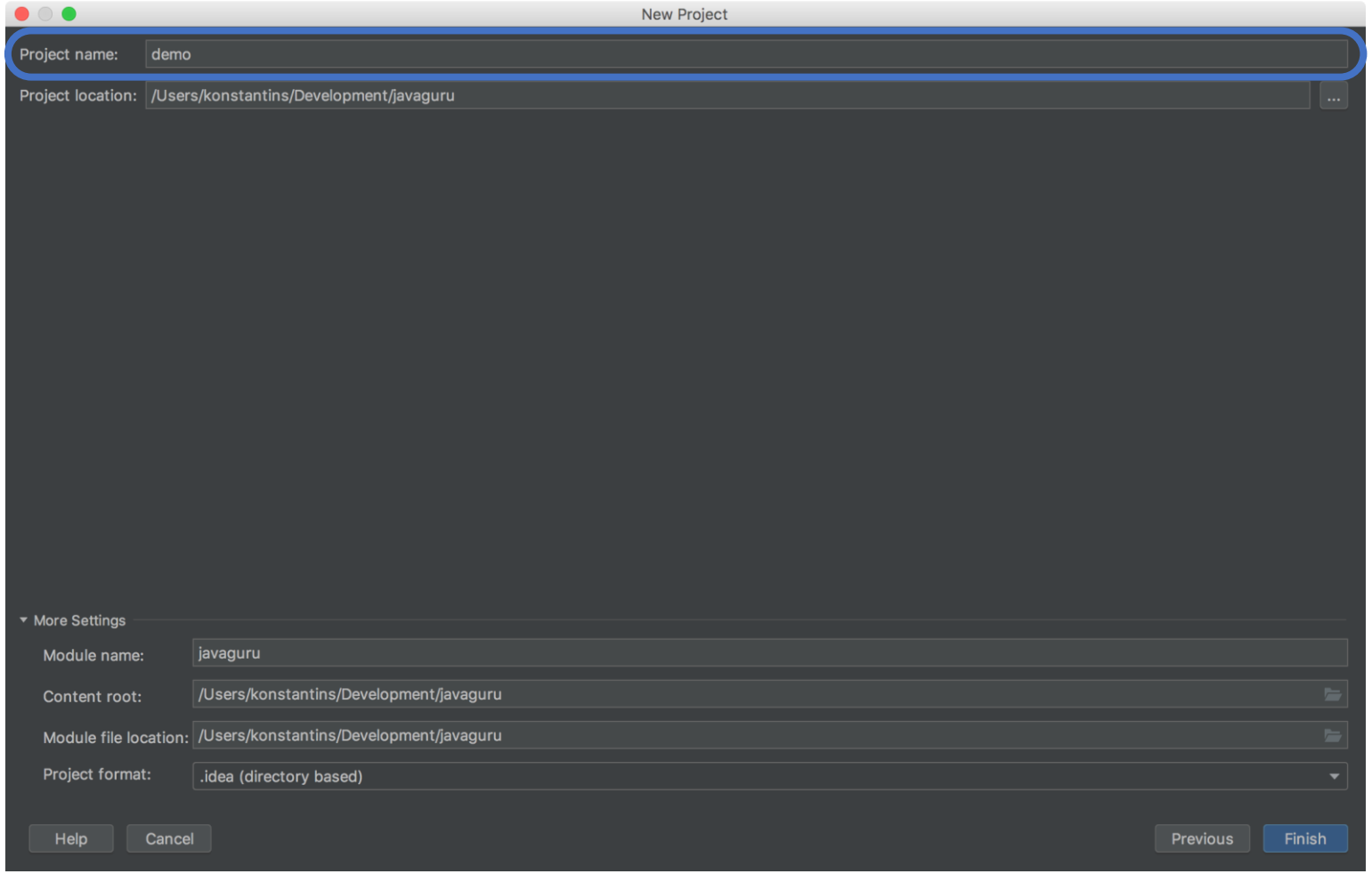
Creating project overview: platform selection



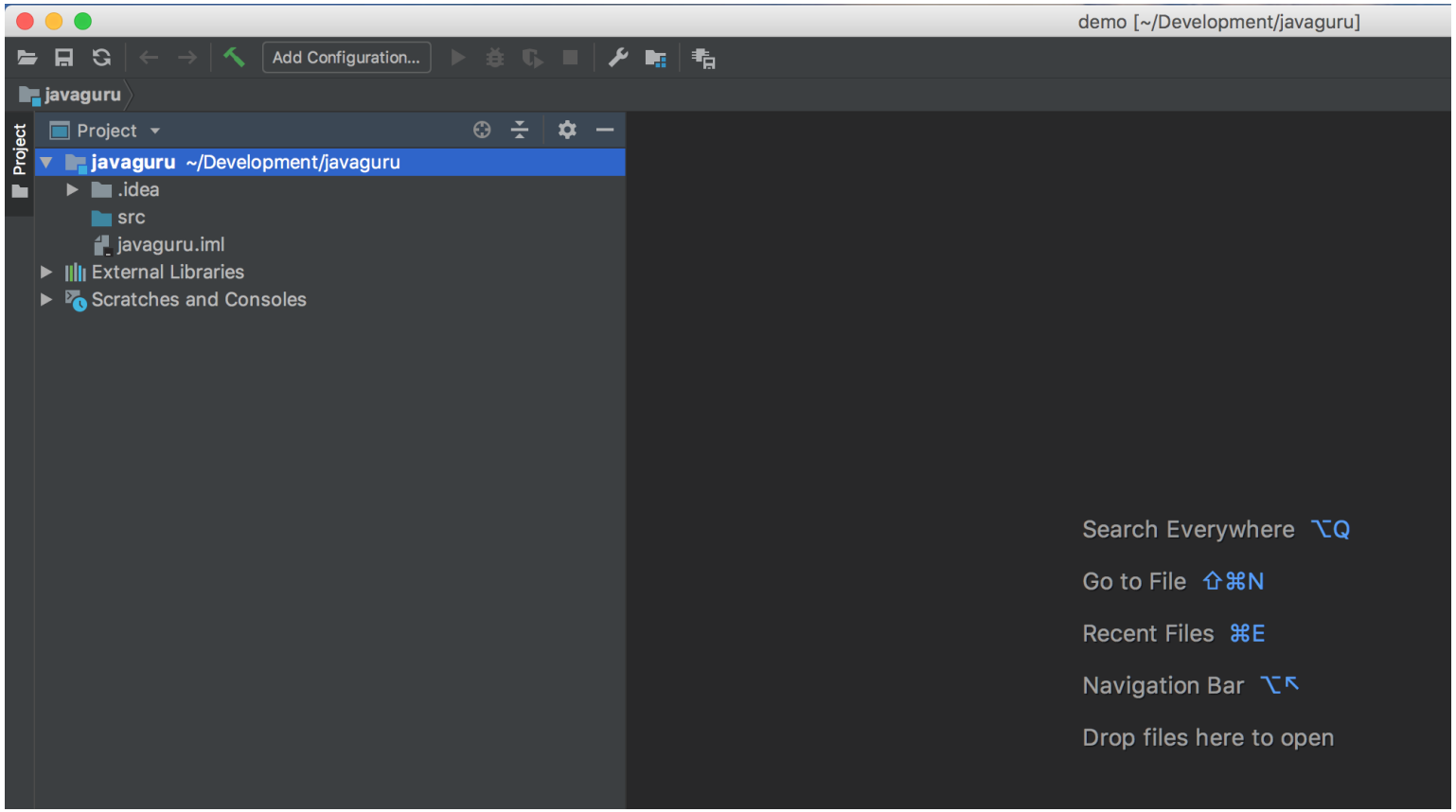
Creating project overview: template selection



Creating project overview: naming



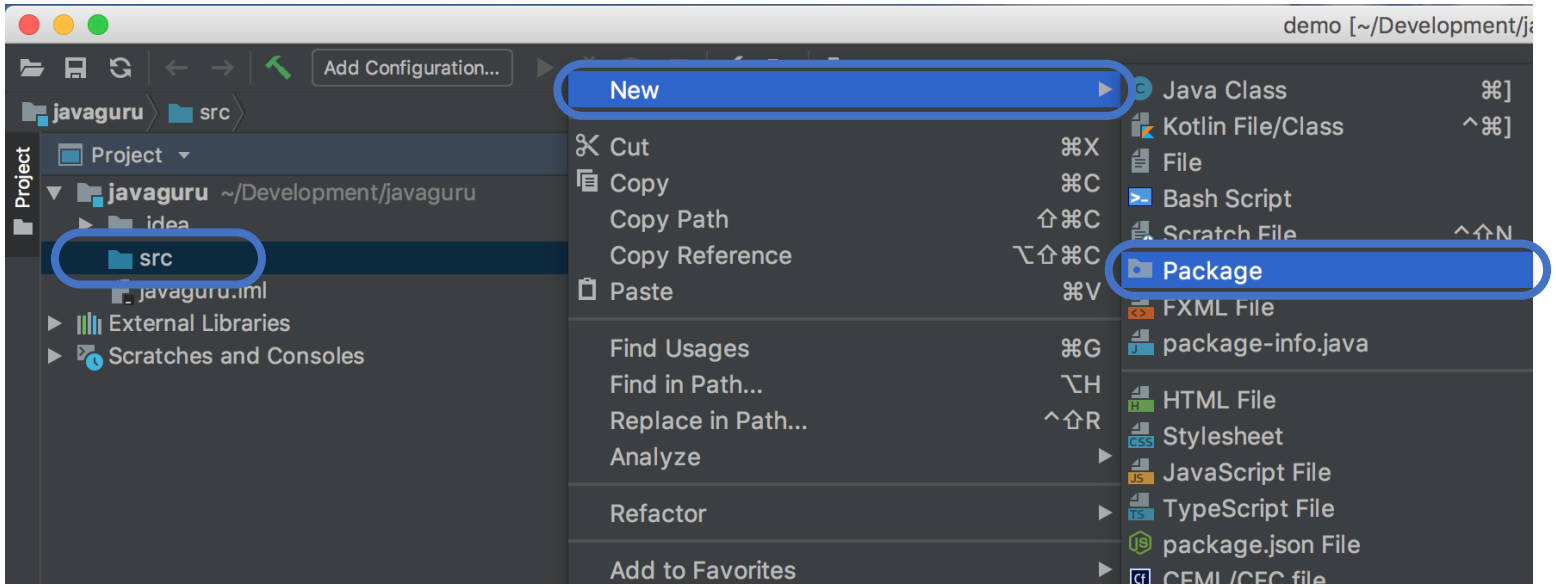
Creating project overview: great success



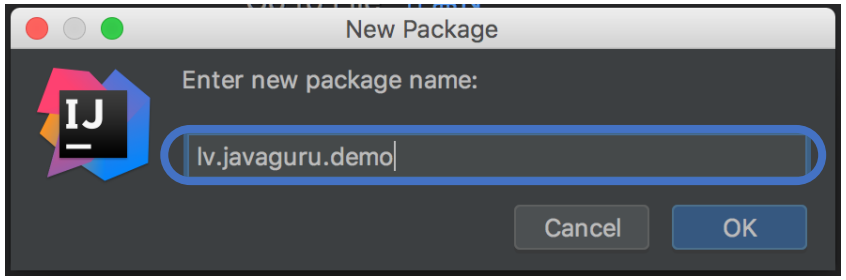
Task 0: Objectives

1. Print *"Hello, JavaGuru World"* text to the console

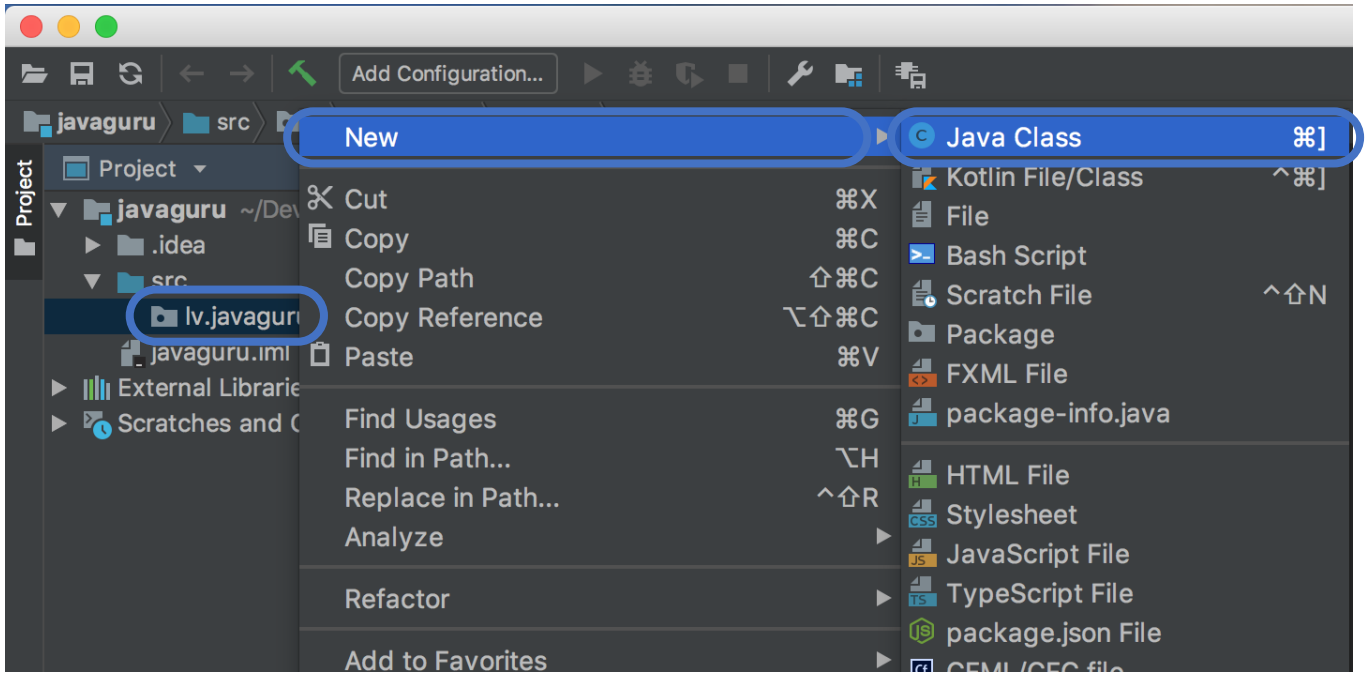
Task 0: create new package



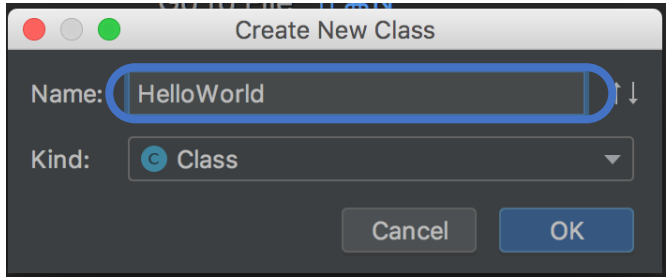
Task 0: name your package



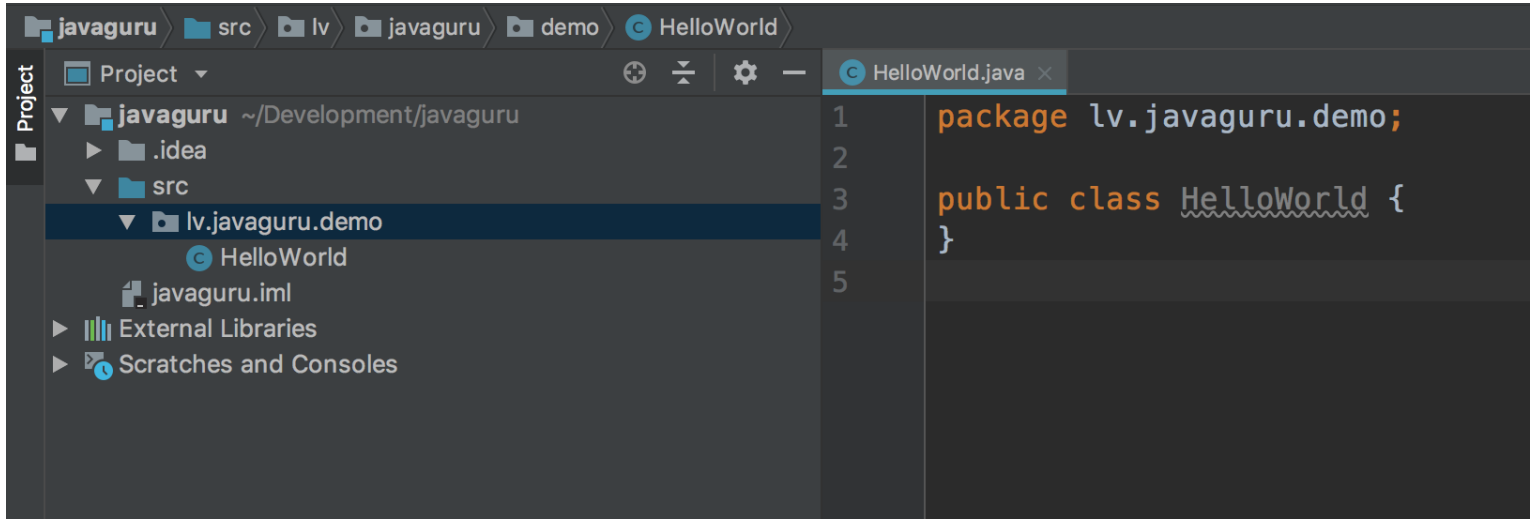
Task 0: create new java class



Task 0: name your java class



Task 0: expected result



Task 0: write main method - point of start

HelloWorld.java

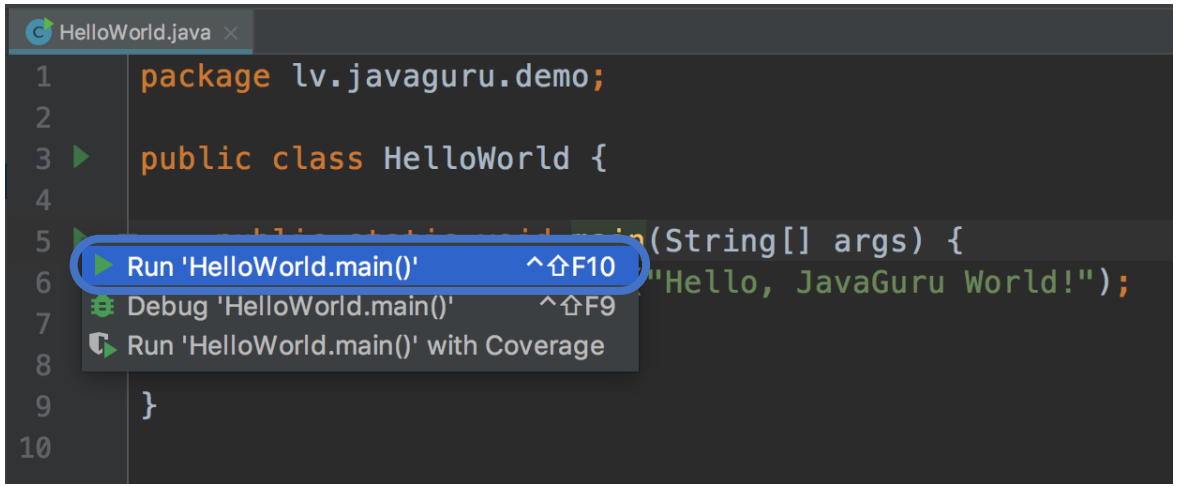
```
1   package lv.javaguru.demo;
2
3   public class HelloWorld {
4
5       public static void main(String[] args) {
6           |
7       }
8
9   }
10
```

Task 0: write code that greets the world

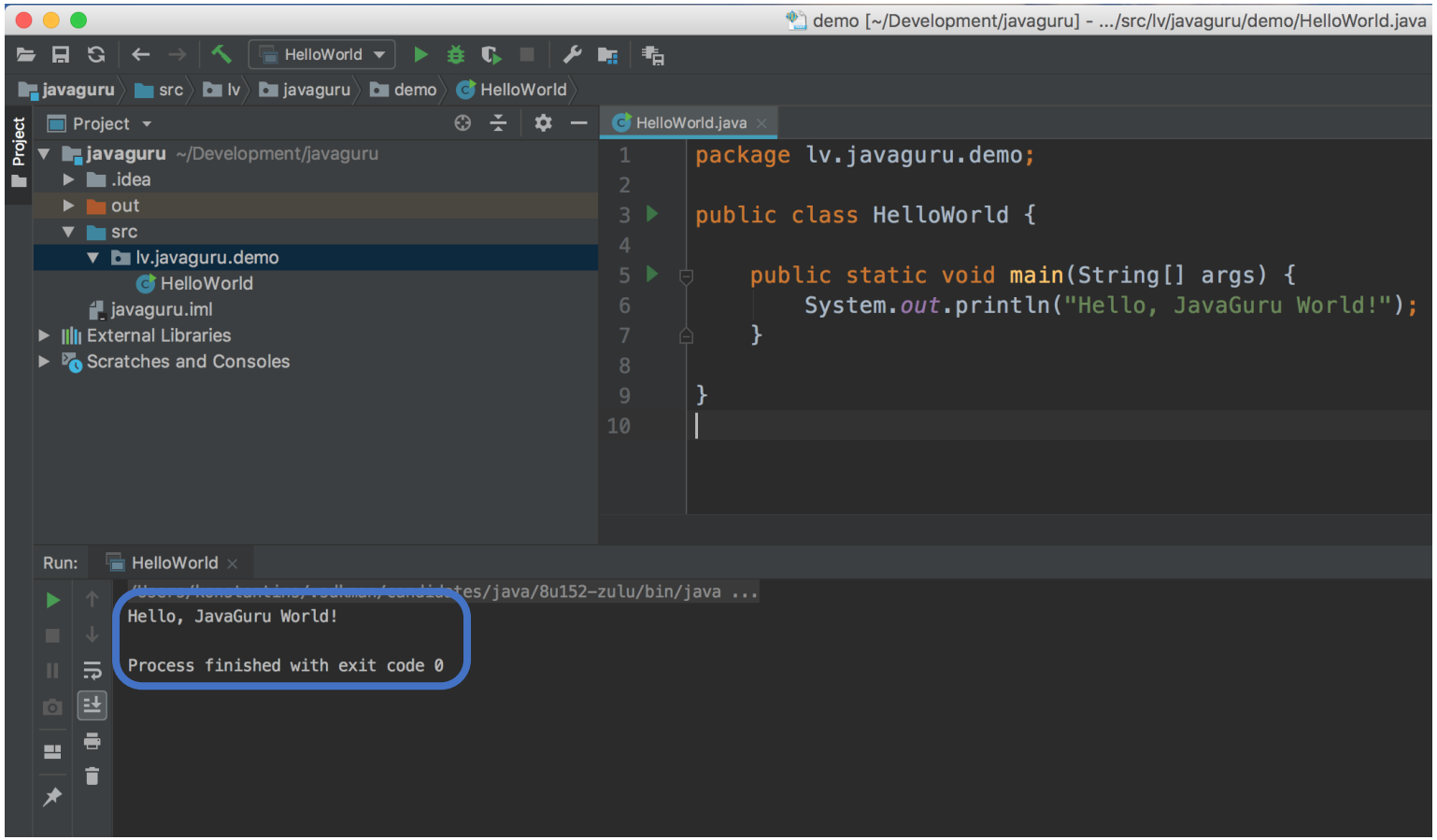
HelloWorld.java

```java
package lv.javaguru.demo;

public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello, JavaGuru World!");
    }

}
```

Task 0: run your code!



Task 0: final result



Task 1: Objectives

1. The program should define two integer variables
2. Calculate the sum of these variables
3. Print result to the console

TASK 1: CODE LISTING

1. Create 1st variable and assign it value of 10
2. Create 2nd variable and assign it value of 20
3. Assign 3rd variable result of computed sum
4. Print 3rd variable to the console

```
package lv.javaguru.demo;

public class CalculatorSumTest {

    public static void main(String[] args) {

      1 int firstNumber = 10;
      2 int secondNumber = 20;
      3 int sumResult = firstNumber + secondNumber;
      4 System.out.println(sumResult);

    }

}
```

Task 2: Objectives

1. The program should create random number generator
2. Generate random number within 0 - 100 range inclusive
3. Print result to the console

Task 2: code listing

1. Create random generator and assign it to the variable
2. Generate random number and assign result to the variable
3. Print *randomNumber* variable to the console

```
package lv.javaguru.demo;

import java.util.Random;

public class RandomNumberGeneratorTest {

  public static void main(String[] args) {
1   Random randomGenerator = new Random();
2   int randomNumber = randomGenerator.nextInt(101);
3   System.out.println(randomNumber);
  }

}
```

Task 2: Improved

1. The program should create random number generator
2. Generate random number within 0 - 100 range inclusive
3. Print result of two random number sum to the console

Task 3 - Bank

- Create a class where you have these methods:
- deposit(); - prints out – “You have deposited 1000 euros”
- withdraw(); - “You have withdrawn 1000 euros”
- checkBalance(); - “You have 1 000 000 euros on your bank account”

Task 3– Bank - improved

- Create a class where you have these methods:
- Create a global variable – for you bank balance
- deposit(int amount); - prints out – “You have deposited {amount} euros”
- withdraw(int amount); - “You have withdrawed {amount} euros”
- checkBalance(); - “You have {amount} euros on your bank account”

Task 4 - Calculator

- Create a class where you have main method and:
- Have two int paramametrs – int a, int b
- Prints result of sum of a and b
- Prints result of substract of a and b
- Prints result of multiplication of a and b
- Prints result of division of a and b

Task 4 – Calculator - improved

- Create a class where you have these methods:
- Sum method - Accepts 2 int parameters and sums them
- Substract method - Accepts 2 int parameters and substracts them
- Multiplication method - Accepts 2 int parameters and multiplies them
- Division method - Accepts 2 int parameters and divide them

Task 4 – Calculator - improved

- Create a class where you have these methods:
- Sum method - Accepts 2 int parameters and sums them
- Substract method - Accepts 2 int parameters and substracts them
- Multiplication method - Accepts 2 int parameters and multiplies them
- Division method - Accepts 2 int parameters and divide them

Task 5 – Create a person

- Create a class where you have these methods:
- Getters and Setters – Name, Age, Weight, isHungry

Task 5 – Create a person - Improved

- Create a class where you have these methods:
- Getters and Setters – Name, Age, Weight, isHungry
- Create and set all values
- Print them using get method

Task 6 – Create a book

- Create a class where you have these methods:
- Getters and Setters – Author, yearPublished, bookName, isNew, ISBN
- Create and set all values
- Print them using get method


THANK YOU FOR YOUR ATTENTION
YouTube