

Professional Skills
Agile Fundamentals
Jira
Git
Databases Introduction
Java Beginner
What is Java?
Installation
Hello World Example
Data Types
Packages
Naming Conventions Cheat Sheet
Flow of Control
Class Members
Operators
Conditionals
Iteration
Arrays
ArrayList
Enhanced For Loops
String Manipulation
Class Constructors
Access Modifiers
Installing Java & Maven To PATH
Object-Oriented Programming Principles
Encapsulation
Inheritance
Polymorphism
Abstraction
Interfaces
Type Casting
Static
Final
Garbage Collection
Input With Scanner
Pass by Value/Reference
JUnit

Class Constructors

Contents

- [Overview](#)
 - [Constructors](#)
 - [Instantiation](#)
 - [Parameters](#)
 - [Overloading](#)
- [Tutorial](#)
- [Exercises](#)

Overview

A constructor in Java is a method that is used to instantiate objects.

The constructor is called immediately when an object of a class is created. It can be used to set initial values for object attributes.

Constructors

A constructor must have the same name as the containing class, this lets the compiler know it is not like other methods in the class.

Constructors must not have a return type (not even void).

```
public class Trainee {
    private String name;
    private String pathway;

    public Trainee(){
        this.name = "Bob";
        this.pathway = "Software Development";
    }
}
```

Instantiation

The act of instantiating a class simply refers to creating a new object using that class as a template.

Parameters

Constructors can also take parameters, which are used to initialise attributes. There is no limit to the number of parameters you may include.

We'll change the above example to include parameters.

```
public class Trainee {
    private String name;
    private String pathway;

    public Trainee(String name, String pathway){
        this.name = name;
        this.pathway = pathway;
    }
}
```

To create an instance of this object we need to call the class, including any parameters of the constructor

<div><div></div><div>Test Driven Development</div></div> <div><div></div><div>UML Basics</div></div> <div><div></div><div>JavaDoc</div></div> <div><div></div><div>Peer Programming</div></div> <div><div></div><div>Code Reviews</div></div>
Maven
Testing (Foundation)
Java Intermediate
HTML
CSS
Javascript
Spring Boot
Selenium
Sonarqube
Advanced Testing (Theory)
Cucumber
MongoDB
Express
NodeJS
React
Express-Testing
Networking
Security
Cloud Fundamentals
AWS Foundations
AWS Intermediate
Linux
DevOps
Jenkins Introduction
Jenkins Pipeline
Markdown
IDE Cheatsheet

```
public static void main(String[] args) {
    Trainee bob = new Trainee("Bob", "Software Development");
}
```

Overloading

Constructors can also be 'overloaded' this means you can have multiple constructors for a class with different input parameters. Java will automatically use the constructor that matches the provided parameters.

```
public class Trainee {
    private String name;
    private String pathway;
    private boolean hired = false;

    public Trainee(String name, String pathway){
        this.name = name;
        this.pathway = pathway;
    }

    public Trainee(String name, String pathway, boolean hired){
        this.name = name;
        this.pathway = pathway;
        this.hired = hired;
    }
}
```

```
public static void main(String[] args) {
    Trainee bob = new Trainee("Bob", "Software Development");
    Trainee bill = new Trainee("Bill", "DevOps", true);
}
```

Tutorial

There is no tutorial for this module

Exercises

There is no exercise for this module

