_nology

Software Engineering

_nology + DB - 25th November 2020

Unit 0 - Precourse Work (approx 20 hours self-guided learning)

Our precourse work will guide learners through the fundamentals of HTML, CSS and JavaScript. This will allow us to spend less time on these things during the course material.

Unit 1 - Web Fundamentals (8 – 10 days)

We begin the journey by deploying the precourse work to production on the first day. Then, we begin the introduction to the world of software engineering, from the basics of the command line to the fundamentals of building a website.

- **Review of the Full Development Stack**
- **Client Server Model**
- **Tooling Setup**
- **Hardware Fundamentals**
- **Unix and the Command Line**
- **Version Control - Git and Github**
- **CI/CD Fundamentals with Github Actions**
- **HTML Fundamentals**
- **CSS Fundamentals**
- **Package Management - NPM**

Project One: Build your own simple website and deploy it live using Github pages.

Unit 2 - JavaScript Fundamentals (10 - 12 days)

Next, we'll learn how to write clean ES6 JavaScript that's easy to read and understand. We also applied TypeScript to introduce strong typing.

- **Programming fundamentals (Javascript)**
- **Variables and Types**
- **Control Flow**
- **Arrays and Iterators**
- **Functions**
- **Objects**
- **Classes**
- **Debugging**

Project Two: Build a simple JavaScript browser game using functional logic

Unit 4 - Front-end Frameworks with React (10 days)

Next we'll use React to teach all the major principles in modern Javascript front end programming.

- **React Components**
- **Functional React with Hooks**
- **Managing Data Flow with Props and State**
- **Separating concerns with Services**
- **Asynchronous code with Promises**
- **Consuming HTTP services / REST**
- **Persisting data to an API**
- **Deployment**
- **Unit and integration testing**

Project Five: Build a personal React project and deploy it live.

Unit 5 - Java Fundamentals (10 – 12 days)

The next step is to graduate from simple Javascript and Typescript to Java, a stricter Object Oriented language. Here we will focus in more detail on best practices in our coding, especially maintenance, testing, performance and design patterns.

- **Using an IDE - IntelliJ**
- **Installing Java**
- **Gradle and Maven**
- **The SOLID principles**
- **Java Fundamentals**
- **JUnit testing**
- **Streams, Threading and Lambdas**

Project Six: Build a command line trading system using Java.

Unit 6 - Database Fundamentals (2 – 3 days), more if time?

We'll spend 2 days ensuring the basics of SQL databases are understood by the group before we move into APIs. This content can essentially be optional if the candidates have a lot of experience of it. We'll also have a day on NoSQL databases so the difference is clearly understood.

- **SQL Database fundamentals**
- **SQL Query Fundamentals with MySQL**
- **SQL joins**
- **Complex and compound SQL queries**
- **SQL vs NoSQL**
- **NoSQL Fundamentals**
- **Fundamentals of Data Design and Normalisation**

Unit 7 - Building Servers with Spring/Spring Boot (5 - 7 days)

After we go deep into Java, it's time to build well tested APIs to give us the data we need.

- **HTTP request/response**
- **Status codes**
- **Spring and Spring Boot**
- **RESTful APIs**
- **The MVC design pattern**
- **Services**
- **Data Layer Abstraction**
- **API Integration Testing**
- **ORMs with Hibernate**
- **Entities and Database integration**
- **Interfaces in Server design**

Project Seven: Build a ForEx trading API in Spring, host it and test it with unit and integration tests.

2 weeks left

Unit 8 - Microservice Architecture Fundamentals (1.5 weeks)

Once we'd built the server, we focused more on data persistence and how to manipulate data using SQL, specifically MySQL.

- **Microservices vs Monoliths**
- **Cloud Fundamentals with GCP**
- **CI/CD with CloudRun**
- **Deploying Microservices**
- **Serverless Architecture**
- **Communicating in the Cloud**
- **Database Deployment**

- **Security in the Cloud**
- **Monitoring in GCP**

Project Eight: Create a self healing cluster of APIs and Databases deployed in GCP.