UDACITY

NANODEGREE PROGRAM SYLLABUS

Java Web Developer

Overview

The ultimate goal of the Java Web Developer Nanodegree program is to equip students with the unique skills they need to build enterprise-scale applications with Java. A graduate of this program will be able to:

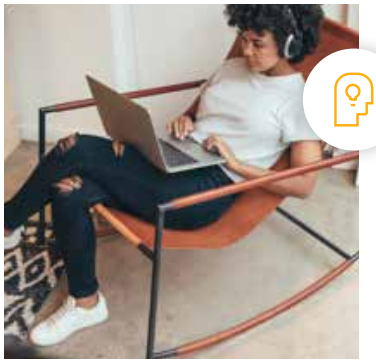
- Understand the fundamentals of the Spring Boot framework and associated integrations and plugins.
- Describe the differences between web services, APIs and microservices; develop REST and GraphQL APIs; and learn how to secure, consume, document and test those APIs and web services.
- Build applications that read and write to relational databases using both the Java Persistence API (JPA) and SQL. Use standard design patterns to make your persistence layer easy to test and integrate with a Spring Boot application.
- Learn about Git, version control and best practices for authorization and authentication. Use Jenkins to build CI/CD pipeline to deploy code to production.

This program is comprised of 4 courses and 4 projects. Each project you build will be an opportunity to demonstrate what you've learned in the lesson, and will show potential employers that you have skills in these areas.

Prerequisite Knowledge: Intermediate knowledge of Java. Some web development experience desirable, but not required. Students should also be able to: initialize and use primitive Java data types (i.e. float, int, etc), select and use Collections from java.util.Collections, design and create classes and class methods in Java, create interfaces and subclasses in Java, launch Java applications from an IDE, and write basic queries in SQL.

Estimated Time: 4 Months at 5-10hrs/week

Prerequisites: Intermediate Java Programming



Flexible Learning: Self-paced, so you can learn on the schedule that works best for you.

Need Help?
udacity.com/advisor
Discuss this program with an enrollment advisor.

Course 1: Spring Boots Basics

Learn the fundamentals of Java while being introduced to a Spring Boot framework and associated integrations and plugins.

Course Project :
Build a Web-based Personal Storage Application

In this project, students will use the skills acquired in the first course to build a web-based personal storage application: SuperDuperDrive! Students will implement user-facing features like file, note and secure credential storage with industry-standard, full-stack development tools. Building on Spring Boot as a base, students will use Spring MVC and Thymeleaf to build a Java-backed web app. Using H2 as an inmemory database, students will store user data with MyBatis, a deadsimple ORM library and secure that data from unauthorized access with Spring Security. Finally, to round out the development cycle and verify that the app is feature-complete, students will implement a series of automated user tests with JUnit and Selenium.

		LEARNING OUTCOMES
LESSON ONE	**Web Development in Java**	• Describe how the Java Application Server facilitates web development • Identify the role of a Servlet within a Java Application Server • Identify the role of Spring as a Servlet application tool • Add and update project dependencies within a Maven POMfile • Choose appropriate starter packs for Spring depending on the application's requirements
LESSON TWO	**Spring Boot Basics for Web Development**	• Set up and configure a working Spring Boot Environment for web development • Configure a Spring Boot application within Java using annotations and factory methods • Identify Spring registered components in Java applications based on Spring annotations • Annotate code to register custom components with a Spring App • Identify and customize essential properties for the Spring Server

		LEARNING OUTCOMES
LESSON THREE	**Spring MVC and Thymeleaf**	• Design HTML templates with Thymeleaf and populate HTML templates with Spring's MVC data model • Explain how a controller populates the data model for a given view • Identify the relationship between controller endpoint return values and the templates that are displayed • Identify the role of the model object passed to controller endpoint methods • Write variable resolution expressions within a ThymeLeaf template to access model data
LESSON FOUR	**Data Persistence and Security**	• Identify mappings between Java objects and SQL tables and leverage those mappings to connect an application with a data store • Explain how ORM leverages similarities between Java data types and SQL data types to reduce development time and programmer error • Write MYBatis SQL template queries using an application's data model • Explain how the @Mapper annotation functions in the Spring App context as a component annotation • Securely store user credentials in a database • Query user information and identify accessible pages based on that information • Use Spring security to automatically filter web traffic based on that information
LESSON FIVE	**Testing**	• Use Selenium/Webdriver to automatically perform user actions in order to test the functionality of web pages • Define JUnit test classes with the @Test annotation • Use the JUnit assertion class to test specific success or failure points • DRun a suite of JUnit tests from their IDE and interpret the results • Navigate to specific URLs with the Selenium web driver • Interact with queried elements from Selenium in the manner of a user to test that functionality exists as intended • Write JUnit tests using these techniques to test individual features of a web app • Organize tests into Page objects so that the application structure is mirrored by the test structure

Course 2: Web Services and APIs

Explore the differences between web services, APIs and microservices. Develop REST and GraphQL APIs, and learn how to secure, consume, document and test those APIs and web services.

Course Project:
Build the Backend System for a Car Website

In this course, the student will build a backend system for a website of cars. This backend will be composed of vehicles list services, pricing services, and location services as mentioned below: Vehicles API — a REST API to maintain vehicles data (CRUD), Pricing Service — a REST API to retrieve the price of a vehicle, and Location API — a HTTP client to retrieve the location of the vehicle. In the project, students will use Java APIs and frameworks to integrate different services using different communication styles. Students will write the CRUD operations to store and retrieve vehicle data and implement an HTTP client to retrieve the address of the vehicle given the latitude and longitude. Students will also integrate the clients (Vehicle API) with pricing services to retrieve the price. Lastly, students will learn to use Swagger to efficiently create documentation for their APIs. During the development of these steps, the student will be guided to write unit tests, error handling, logging and other best practices.

	LEARNING OUTCOMES	
LESSON ONE	**Web Services & APIs Overview**	• Describe web services and their advantages • Describe how web services communicate • Explore the differences between web services, APIs and microservices
LESSON TWO	**Develope REST APIs with Spring Boot**	• Describe the REST architectural style and the importance of data formats) • Develop a REST API using Spring Boot and incorporate exception handling. • Use proper HTTP response codes
LESSON THREE	**Develop GraphQL APIs with Spring Boot**	• Describe GraphQL and its advantages over REST • Create a GraphQL schema • Develop a GraphQL server and API using Spring Boot • Use GraphQL to execute queries and operations on data

	LEARNING OUTCOMES	
LESSON FOUR	**Develop Microservices with Spring Boot**	• Describe the Microservices Architecture (MSA) • Expose a microservice using Spring Boot • Register a microservice
LESSON FIVE	**Secure API Endpoints with Spring Security**	• Describe Spring Security • Explain the differences between authentication vs authorization • Incorporate Basic Authentication practices to secure an API
LESSON SIX	**Consume Web Services and APIs**	• Consume a REST API • Consume a SOAP-based web servicer • Fetch and process XML and JSON
LESSON SEVEN	**Document REST APIs**	• Describe Swagger, a n open-source software framework to design, build, document, and consume RESTful web services • Add Swagger annotations to model • Generate API documentation
LESSON EIGHT	**Test REST APIs**	• Describe and explain unit and integration testing • Incorporate unit and integration testing into a REST API

Course 3: Data Stores & Persistence

Build applications that read and write to relational databases using both the Java Persistence API (JPA) and SQL. Use standard design patterns to make your persistence layer easy to test and integrate with a Spring Boot application.

Course Project:
Design the Data Model for a SaaS Application

Students will design and implement the data model for Critter Chronologer, a Software as a Service application that provides a scheduling interface for small businesses that take care of animals. This enterprise project will allow users to create schedules that associate pets, owners and employees with calendar events. Students will configure their application to connect to an external database and use both JDBC and Hibernate to persist changes to it. Basic CRUD operations will be exposed via a REST controller layer so that students can test their application using Postman.

		LEARNING OUTCOMES
LESSON ONE	**Data in Multitier Architecture**	• Design Entities that map Java data types to database structures • Represent complex associations between Entities in persistence • Identify and select inheritance strategies • Isolate Entity scope through the use of annotations and DTOs
LESSON TWO	**Java Persistence API**	• Understand and utilize key concepts in Object Relational Mapping (ORM) such as Persistence Context and Entity Manager, and learn about the Repository design pattern • Propagate retrievals and persists with the help of Lazy Loading and Cascading • Write and execute object queries in Java using JPQL • Build implementations for your Repository methods automatically with Spring Data JPA • Control the execution of queries through Transactions

		LEARNING OUTCOMES
LESSON THREE	**Connecting to Data Sources**	• Connect Spring Boot to both internal and external data sources • Customize Spring DataSource construction and injection • Use Spring and Hibernate to automatically initialize your data sources • Configure unit tests to use different data sources
LESSON FOUR	**Persistence Without JPA**	• Learn about the differences in Data Object design when retrieving data with SQL • Initialize Data Sources with SQL scripts • Use the Data Access Object design pattern • Execute SQL queries with JdbcTemplate and automatically map the results to your Data Objects • Decide when to use SQL and when to use Hibernate, and learn how to combine them both in the same project

Course 4: Security and DevOps

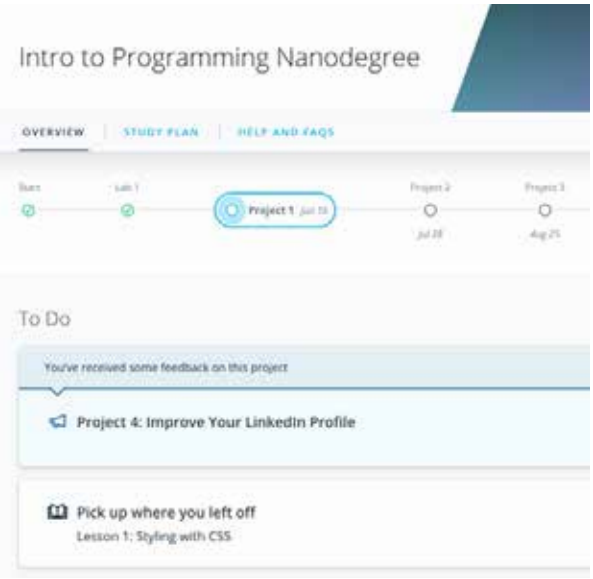
Learn about Git, version control and best practices for authorization and authentication. Use Jenkins to build a CI/CD pipeline to deploy code to production.

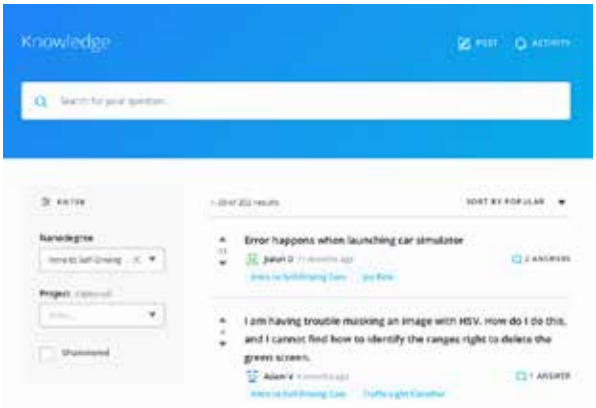
Course Project:
Implement Authorization for an eCommerce Application

In this project, students will add authorization using Spring Security with OAuth and username/password combinations to an eCommerce web application created in Spring Boot. Proper security and hashing will need to be implemented to store this data as well. Students will identify the right metrics for an effective analytics environment and use either Splunk or ELK to analyze the metrics. Students will also automate the configuration and deployment of these systems and the application. Students will use Jenkins to integrate with their version control and deploy their application to AWS.

		LEARNING OUTCOMES
LESSON ONE	**Git**	• Learn the basics of git such as branching, pull requests and merging • Describe what version control is and means
LESSON TWO	**Authorization and Authentication**	• Identify the need for security in modern day web applications • Describe best practices for authorization and authentication • Implement modern authorization and authentication technologies such as password hashing and JWT
LESSON THREE	**Testing**	• Learn and use testing frameworks such as junit • Describe the concept of code coverage and its importance • Implement negative testing as well as happy path testing
LESSON FOUR	**Loggin and Analytics**	• Identify important application metrics and log them • Send logs to Splunk • Create visualizations and dashboards in Splunk to display those metrics
LESSON FIVE	**Jenkins and CI/CD**	• Describe and explain CI/CD • Create a build pipeline using Jenkins • Build a Docker Image • Create a CI pipeline for a Docker Image • Deploy Docker container in production

Our Classroom Experience





REAL-WORLD PROJECTS
Build your skills through industry-relevant projects. Get personalized feedback from our network of 900+ project reviewers. Our simple interface makes it easy to submit your projects as often as you need and receive unlimited feedback on your work.

KNOWLEDGE
Find answers to your questions with Knowledge, our proprietary wiki. Search questions asked by other students and discover in real-time how to solve the challenges that you encounter.

STUDENT HUB
Leverage the power of community through a simple, yet powerful chat interface built within the classroom. Use Student Hub to connect with your technical mentor and fellow students in your Nanodegree program.

WORKSPACES
See your code in action. Check the output and quality of your code by running them on workspaces that are a part of our classroom.

QUIZZES
Check your understanding of concepts learned in the program by answering simple and auto-graded quizzes. Easily go back to the lessons to brush up on concepts anytime you get an answer wrong.

CUSTOM STUDY PLANS
Work with a mentor to create a custom study plan to suit your personal needs. Use this plan to keep track of your progress toward your goal.

PROGRESS TRACKER
Stay on track to complete your Nanodegree program with useful milestone reminders.

Learn with the Best

Peter Zastoupil

INSTRUCTOR

Peter Zastoupil is an enterprise developer and technical administrator. He has seven years of on-the-job experience building features for massive enterprise Java servers, and over four years of teaching those skills to new developers. He enjoys music production and long walks with his dog, Honeydew, in his free time.

Kesha Williams

INSTRUCTOR

Kesha has over 20 years experience in software development and is a software engineering manager at Chick-fil-A, routinely leading innovation teams in proving out the use of cloud services to solve complex business problems. She was recently named an Alexa Champion by Amazon.

Alex Pritchard

INSTRUCTOR

Alex is a Senior Software Engineer for CPA Global. He is excited to combine his background as a music educator with more than a decade of enterprise Java experience to help create this practical course on Data Stores and Persistence.

Sareeta Panda

INSTRUCTOR

Sareeta is a Java enthusiast and Senior Developer at Walmart e-Commerce. She specializes in Enterprise Application development with Java and Kafka, NoSQL, Spring security and CI/CD. Sareeta has over a decade of experience, spanning recently acquired startups to top Fortune 500 companies.

All Our Nanodegree Programs Include:

EXPERIENCED PROJECT REVIEWERS

REVIEWER SERVICES

- Personalized feedback & line by line code reviews
- 1600+ Reviewers with a 4.85/5 average rating
- 3 hour average project review turnaround time
- Unlimited submissions and feedback loops
- Practical tips and industry best practices
- Additional suggested resources to improve

TECHNICAL MENTOR SUPPORT

MENTORSHIP SERVICES

- Questions answered quickly by our team of technical mentors
- 1000+ Mentors with a 4.7/5 average rating
- Support for all your technical questions

PERSONAL CAREER SERVICES

CAREER SUPPORT

- Resume support
- Github portfolio review
- LinkedIn profile optimization

Frequently Asked Questions

PROGRAM OVERVIEW

WHY SHOULD I ENROLL?

Java is one of the most popular programming languages in the world, and a majority of large enterprises rely on Java for their back-end architecture. In this Nanodegree program, you'll learn to build and deploy back-end infrastructure(s) using Java, and graduates will have real-world projects to share with current or prospective employers to demonstrate mastery of these high-demand skills.

WHAT JOBS WILL THIS PROGRAM PREPARE ME FOR?

The addition of Java skills to your developer toolkit is an excellent move for any developer seeking a critical career advantage. This program emphasizes practical coding skills that demonstrate your ability to build, test and deploy back-end infrastructure using Java, and will prepare you for a variety of engineering roles that leverage the Java language.

It is designed for people with an existing background in programming who are looking to build a strong foundation in Java to either advance within their current field or position themselves to learn more advanced skills for a career transition.

HOW DO I KNOW IF THIS PROGRAM IS RIGHT FOR ME?

If you are interested in building out the infrastructure that powers and supports the many web, desktop, mobile and integrated applications in the business world, this program is a great fit for you.

Additionally, if you are a developer who doesn't have any back-end experience, or a back-end developer who doesn't know Java, this is a great place to build upon your existing skill set.

ENROLLMENT AND ADMISSION

DO I NEED TO APPLY? WHAT ARE THE ADMISSION CRITERIA?

No. This Nanodegree program accepts all applicants regardless of experience and specific background.

WHAT ARE THE PREREQUISITES FOR ENROLLMENT?

To enroll, you should have intermediate knowledge of Java. Some web development experience is desirable, but not required:

- Initialize and use primitive Java data types (i.e. float, int, etc)
- Select and use Collections from java.util.Collections
- Design and create classes and class methods in Java
- Create interfaces and subclasses in Java
- Launch Java applications from an IDE
- Write basic queries in SQL

FAQs Continued

IF I DO NOT MEET THE REQUIREMENTS TO ENROLL, WHAT SHOULD I DO?

If you believe you need more preparation, here are some additional resources you can use:
- Introduction to Programming Nanodegree program
- Full Stack Web Developer Nanodegree program

TUITION AND TERM OF PROGRAM

HOW IS THIS NANODEGREE PROGRAM STRUCTURED?

The Java Web Developer Nanodegree program is comprised of content and curriculum to support 4 (four) projects. We estimate that students can complete the program in four 4 months, working 5-10 hours per week.

Each project will be reviewed by the Udacity reviewer network. Feedback will be provided and if you do not pass the project, you will be asked to resubmit the project until it passes.

HOW LONG IS THIS NANODEGREE PROGRAM?

Access to this Nanodegree program runs for the length of time specified above. If you do not graduate within that time period, you will continue learning with month to month payments. See the **Terms of Use** and **FAQs** for other policies regarding the terms of access to our Nanodegree programs.

SOFTWARE AND HARDWARE

WHAT SOFTWARE AND VERSIONS WILL I NEED IN THIS PROGRAM?

There are no software and version requirements to complete this Nanodegree program. All coursework and projects can be completed via Student Workspaces in the Udacity online classroom.