UDACITY

NANODEGREE PROGRAM SYLLABUS

Java Developer

Overview

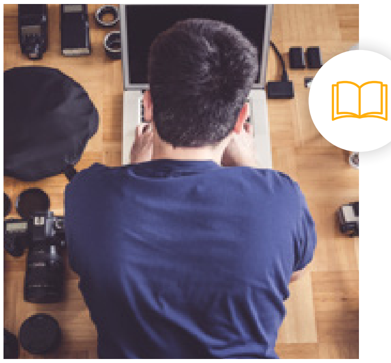
The ultimate goal of the Java Developer Nanodegree program is to equip students with the unique skills they need to build enterprise-scale applications with Java. A graduate of this program will be able to:

- Understand the fundamentals of Java, while being introduced to a 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.
- Work with relational and non-relational databases, use Java to read/write and build test cases for MySQL and MongoDB, and build persistence for Java applications
- Learn about Git, version control, and best practices for authorization and authentication. Use Jenkins to build CI/CD pipeline to deploy code to production.

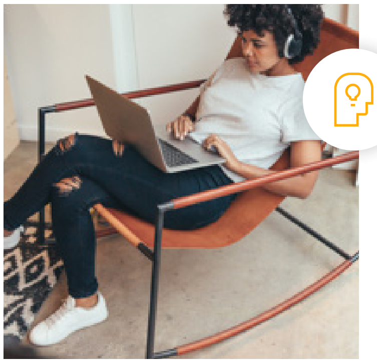
This program is comprised of 4 courses and 4 projects. Each project you build will be an opportunity to demonstrate what you've learned in the lesson, and will show potential employers that you have skills in these areas.

Prerequisite Knowledge: Intermediate knowledge of any programming language, preferably an object-oriented language (e.g, Python, C++). Some web development experience desirable, but not required.

Estimated Time:
4 Months at
10hrs/week



Prerequisites:
Intermediate
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: Java Basics

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

Course Project
Create a Chat Room App
with Spring Boot

In this project, students will use the skills acquired in the first course to build a web-based chat room application using spring boot and websocket components. Students will implement the message model and controller to build an in-memory application. Students will also use Thymeleaf to create web pages to the build user login flow and chat room interface. Once the application is ready, students will create web driver tests to make sure the web application has sufficient test coverage.

	LEARNING OUTCOMES	
LESSON ONE	**Introduction to Java**	• Use different data structures and algorithms in Java and calculate time complexity of the code • Implement exception handling, file IO, and wBufferReader
LESSON TWO	**Spring Boot Servlet, Filter and Listener**	• Describe how the model-view-controller (MVC) architecture works • Configure Spring Boot and create Spring Boot servlet, filter and listener, from scratch • Interact with static resources in a Spring Boot application
LESSON THREE	**Spring Boot with Thymeleaf and Mybatis**	• Manage spring boot application with Maven plugins • Create web pages with Thymeleaf • Integrate Spring Boot with Mybatis
LESSON FOUR	**Spring Boot Web Socket with Spring Boot tests**	• Implement Spring Boot websocket structure and components • Create the message model and implement chat room controller. • Create web driver tests to ensure test coverage.

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 web site of cars. This backend will be composed of vehicles list services, pricing services, and the following location services: 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.

	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	**Develop 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

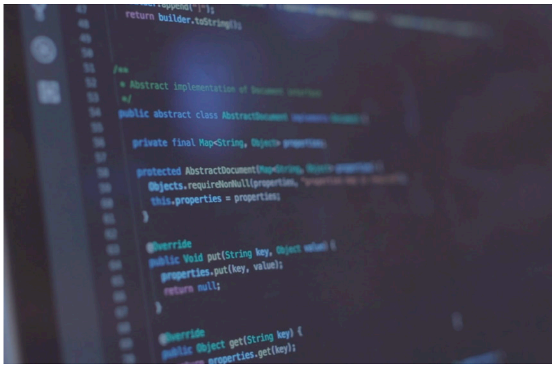
Learn about different data stores and how to build persistence for Java applications. Work with relational and non-relational databases, and use Java to read/write and build test cases for MySQL and MongoDB.

Course Project
Data Store for
Customer Reviews

Students will build the polyglot persistence layer for a REST API that will support the customer reviews section of a product page in an ecommerce application. Students will begin with setting up MySQL & MongoDB and define classes for MongoDB model. Students will then update the persistence service to read/write from both MySQL and MongoDB, and write code to calculate aggregate rating for a product. Students will also build corresponding test cases for JPA & MongoDB repositories and wire the persistence service to the already provided REST Controller. By the end of this project, students will have built a fully functioning REST API with persistence in Relational Database Management System (RDBMS) that can be inspected via a tool like Postman.

	LEARNING OUTCOMES	
LESSON ONE	**Web Services & APIs Overview**	• Describe the structure of relational databases (MySQL) and explain ACID compliance • Write SQL queries to operate on relational databases • Use Java Database Connectivity (JDBC) API to build persistence for a Java application • Use Flyway to perform database migrations
LESSON TWO	**Develop REST APIs with Spring Boot**	• Describe the benefits and uses for Object Relational Mapping and Java Persistence API (JPA) • Explain the concepts behind Spring Data project • Develop a Spring Boot app that defines a Repository • Use Spring and H2 for testing database code

	LEARNING OUTCOMES	
LESSON THREE	**NoSQL & MongoDB**	• Describe NoSQL, non relational databases and flexible schema • Describe the use cases for the document oriented database MongoDB • Use Mongo shell to perform different operations such as create/modify collections, select/create/update/delete documents and aggregations • Describe the pros & cons of nested documents and document references
LESSON FOUR	**MongoDB for Java**	• Connect to MongoDB from Java • Use Spring Data MongoDB to connect to MongoDB • Define document relationships • Define Repository and Repository methods that use filters, sort and pagination • Use Embedded MongoDB for testing Repositories



```
        return builder.toString();



/**
 * Abstract implementation of [illegible]
 */
public abstract class AbstractDocument [illegible]

  private final Map<String, Object> [illegible]

  protected AbstractDocument([illegible]
    Objects.requireNonNull(properties, [illegible]
    this.properties = properties;
  }

  @Override
  public Void put(String key, Object value) {
    properties.put(key, value);
    return null;
  }

  @Override
  public Object get(String key) {
    [illegible] properties.get(key);
```

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

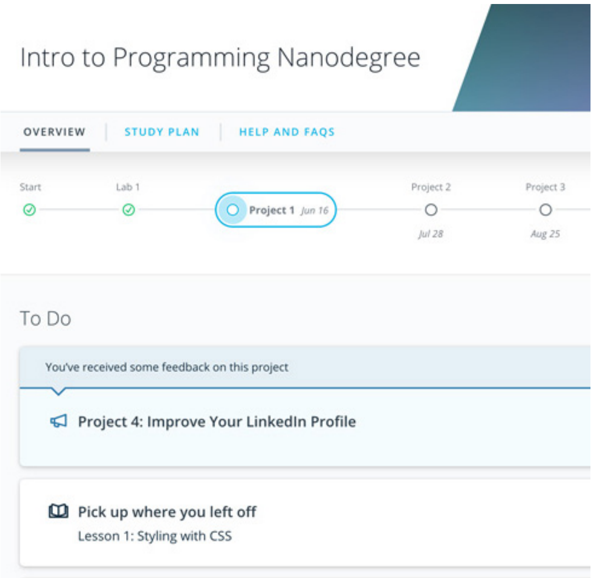
In this project, students will add authorization using Spring Security with OAuth and username/password combinations to an eCommerce web application created in Spring Boot. Proper security and hashing will need to be implemented to store this data as well. Students will identify the right metrics for 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	**Logging 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.



Intro to Programming Nanodegree

To Do

You've received some feedback on this project

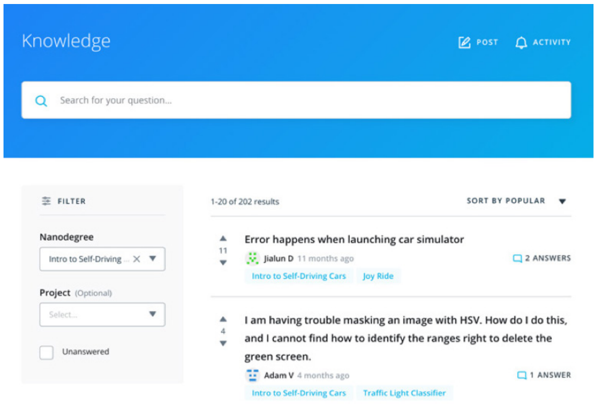
Project 4: Improve Your LinkedIn Profile

Pick up where you left off

Lesson 1: Styling with CSS

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.



1-20 of 202 results

Error happens when launching car simulator
11
Jialun D 11 months ago
2 ANSWERS
Intro to Self-Driving Cars
Joy Ride

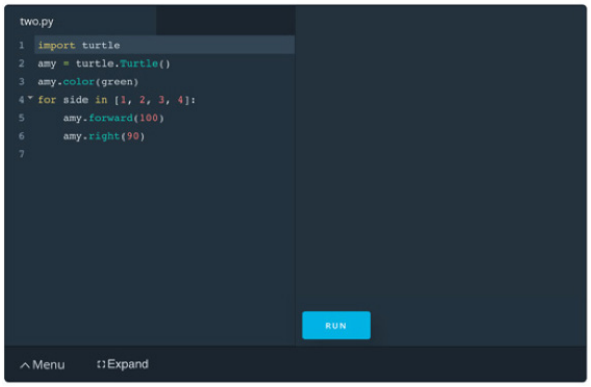
I am having trouble masking an image with HSV. How do I do this, and I cannot find how to identify the ranges right to delete the green screen.
4
Adam V 4 months ago
1 ANSWER
Intro to Self-Driving Cars
Traffic Light Classifier

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.



```
import turtle
amy = turtle.Turtle()
amy.color(green)
for side in [1, 2, 3, 4]:
    amy.forward(100)
    amy.right(90)
```

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

Stephen Chen

INSTRUCTOR

Stephen is a full stack software developer who's worked for many top-notch technology companies. Stephen has extensive experience with data structures and algorithms, and a great passion for teaching.

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.

Aravindan Ramkumar

INSTRUCTOR

Aravindan is a software engineer with over 15 years of experience building distributed software systems; he's currently working at Netflix on the infrastructure for their next generation media pipeline. Aravindan is a certified Java developer who is passionate about building utilitarian products.

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
- 900+ 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

- Chat with mentors in the Student Hub community
- Weekly learning plan focused on your progress, goals, and schedule
- 1000+ mentors with a 4.7/5 average rating
- Support for your questions when you need it

PERSONAL CAREER SERVICES

CAREER COACHING

- Personal assistance in your job search
- Monthly 1-on-1 calls
- Personalized feedback and career guidance
- Access to Udacity Talent Program used by our network of employers to source candidates
- Advice on negotiating job offers
- Interview preparation
- Resume services
- 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?

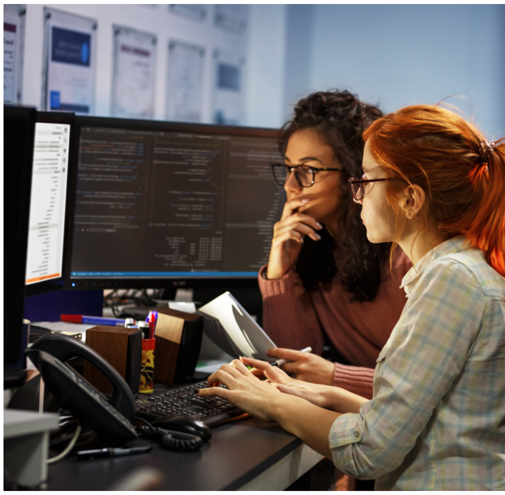
There is no application. This Nanodegree program accepts everyone, regardless of experience and specific background.

WHAT ARE THE PREREQUISITES FOR ENROLLMENT?

To enroll, you should have some experience in the following courses or skills:

- Programming with Python or another object-oriented programming language
- Data Structures including Lists, Arrays, Dictionaries
- Git/GitHub

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:

- Intro to Python Programming
- Introduction to Programming Nanodegree program
- Full Stack Web Developer Nanodegree program

TUITION AND TERM OF PROGRAM

HOW IS THIS NANODEGREE PROGRAM STRUCTURED?

The Java 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 in the payment card on the Nanodegree program overview page. If you do not graduate within that time period, you will continue learning with month to month payments. See the Terms of Use for other policies around the terms of access to our Nanodegree programs.

CAN I SWITCH MY START DATE? CAN I GET A REFUND?

Please see the Udacity Nanodegree program FAQs for policies on enrollment in our programs.

SOFTWARE AND HARDWARE

WHAT SOFTWARE AND VERSIONS WILL I NEED IN THIS PROGRAM?

There are no specific hardware or software requirements for this program, other than access to the internet and a 64-bit computer.