UDACITY

NANODEGREE PROGRAM SYLLABUS

Java Programming

UDACITY

Java Programming Nanodegree Program

BUILD AND DEPLOY JAVA APPLICATIONS AT SCALE

This Nanodegree program is intended to elevate your Java abilities and teach you the fundamental skills used by Java developers to design, code, test and deploy cutting-edge Java software. It is ideal for programming beginners with little or no Java experience who want to set themselves up for success as a professional Java developer and deploy functional Java-based applications of their own creation.

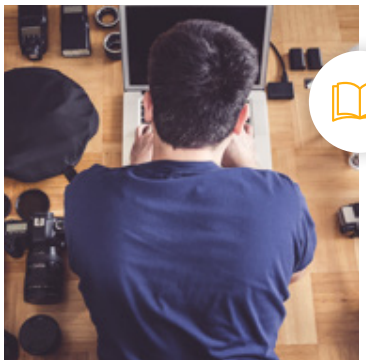
A graduate of this program will be able to:

- Apply basic features of Java programming language (data types, variables, conditional statements, loops, methods) to develop applications
- Use IntelliJ IDE to develop and troubleshoot Java applications
- Design and create classes and class methods in a Java application
- Apply interfaces, inheritance and polymorphism in Java applications
- Select and use Collections (lists, maps, sets, etc.) and basic features of functional programming (lambdas, streams, Optional) to organize and process data
- Apply advanced concepts (reflection, class loading, etc.) to implement advanced Java applications

Prerequisites:

No prior Java experience is required, but you should be familiar with basic programming concepts like variables, loops, conditionals, arrays and methods.

Length of Program*:
3 months



Textbooks required:
None

Frequency of Classes:
The program is flexible, self-paced with suggested project deadlines

Instructional Tools Available:
Video lectures, mentor-led student community, forums, project reviews

*This program includes 3 courses and 3 projects. Each project you complete will be an opportunity to grow your programming portfolio and will demonstrate to potential employers that you have skills in these areas.

Course 1: Java Fundamentals

In order to be a great Java developer you must first learn the fundamentals of Java. This course introduces the fundamental concepts and practices of Java programming. These include basic Object-oriented Programming (OOP) concepts such as classes, encapsulation and inheritance, as well as core Java concepts such as collections, exceptions and commonly used Java types.

Project 1
Hotel Reservation Application

Students will be designing and implementing a hotel reservation application. The hotel reservation application will allow customers to find and book a hotel room. Rooms will contain the price along with the dates that the room is reserved. All room bookings will be associated with a single customer account, the customer account will include the customer name (first and last) and email. The application will allow customers to retrieve a list of the hotel's free rooms. In addition, the hotel reservation application will allow customers to find and book rooms based on availability and price.

	SUPPORTING LESSON CONTENT	
LESSON ONE	**Java Programming Basics**	• Understand keywords and how they are used in Java. • Be able to declare variables in Java. • Describe the difference between primitive variables and objects. • Apply casting to change the type of a variable into another type. • Understand what truncation is and why it occurs. • Create a method in Java. • Describe the difference between Stack and Heap memory. • Know the four different access modifier types. • Create an array to store primitive variable types. • Create each of the three different loops (While, do While and For) to iterate over an array. • Create a Java program in IntelliJ. • Describe the purpose of Javadoc.
LESSON TWO	**Defining Classes**	• Explain what an object is and why we use them. • Describe the differences between a class and an object. • Know the parts and syntax for creating a class. • Create different objects from a specific object type. • Describe the purpose of Garbage Collection in Java.

	SUPPORTING LESSON CONTENT	
LESSON THREE	**Object-Oriented Programming**	• Describe what packages are in Java and why we use them. • Apply Inheritance to share behavior and state between related classes. • Apply polymorphism in Java to support decoupling. • Describe the difference between an Interface and Abstract class.
LESSON FOUR	**Common Type**	• Apply throwing an Exception from a method when an issue is discovered. • Create an Exception Handler to catch and handle an Exception. • Use Enums to provide a predefined selection of constants. • Use the Java Scanner class to read input from the console. • Apply Java Data and Calendar class to store and manipulate day and time. • Use the Java RegEx package to validate the input of a String. • Use some of the advanced methods of the String class to create substring, search for text and investigate different characters of a string.
LESSON FIVE	**Generics and Collections**	• Describe the purpose of using Generics in Java. • Create a generic method that processes different class types without casting. • Apply generics to Collections to create stronger type data sets. • Use data structures that implement the Collection interface. • Use the Collections utility class to sort a list of Strings.
LESSON SIX	**Advanced Java Types**	• Use Maps to store and retrieve data based on a key value. • Use Sets to remove duplicate values from a List. • Use a Queue to store values in a FIFO manner.

Course 2: Advanced Java Programming Techniques

The best Java programs combine excellent software designs with modern language features. This course will teach you some lesser known features of Java, such as functional programming, I/O, serialization and reflection. Strengthen your software design intuition by exploring the design ideas that underpin understandable, extensible and scalable software systems. Learn the basic concepts and techniques of concurrent programming in Java, so your programs can take advantage of modern, multi-core hardware.

Project 2
Legacy Web Crawler

Concurrency is a powerful tool to increase the performance of your Java programs. In this project, you'll use concurrent programming techniques to enhance a legacy web crawler so that it can take advantage of multi-core architectures. The crawler will read configuration from a JSON file, download and parse multiple HTML documents in parallel and record popular web terms in an output file. You'll also build a method profiling tool to measure the efficiency of the crawler and prove the benefits of the parallel crawler.

	SUPPORTING LESSON CONTENT	
LESSON ONE	**Functional Programming in Java**	• Understand the differences between imperative and functional programming. • Use functional interfaces, lambdas and method references to simplify and improve your Java code. • Apply Java's Stream API to perform quantitative analysis over large sets of data.
LESSON TWO	**Working with Files and I/O**	• Understand the differences between program memory and persistent storage. • Know the different uses of binary and encoded data and understand when to use each. • Use the Java Files API to read and write files. • Recognize different kinds of resource leaks and apply modern Java techniques to prevent them. • Serialize and deserialize between Java objects and common formats like JSON, XML and binary.
LESSON THREE	**Design Patterns**	• Identify the principles of good software design and build strong software design intuition. • Understand what design patterns are and why they are useful. • Use Creational, Behavioral and Structural design patterns to write flexible and easy-to-understand code. • Apply Dependency Injection to simplify object creation and promote testable designs.

	SUPPORTING LESSON CONTENT	
LESSON FOUR	**Reflection**	• Use reflection and annotations to introspect and add dynamic capabilities to your programs. • Implement Java interfaces at runtime using dynamic proxies. • Understand the fundamentals of Aspect Oriented Programming. • Use class loaders to customize how Java loads byte code.
LESSON FIVE	**Introduction to Concurrent Programming**	• Recognize when concurrency can help improve the performance of Java programs. • Use threads, thread pools and parallel streams to achieve parallelism in Java. • Apply Java synchronization tools to correctly share state between threads in a parallel program.

Course 3: Java Application Deployment

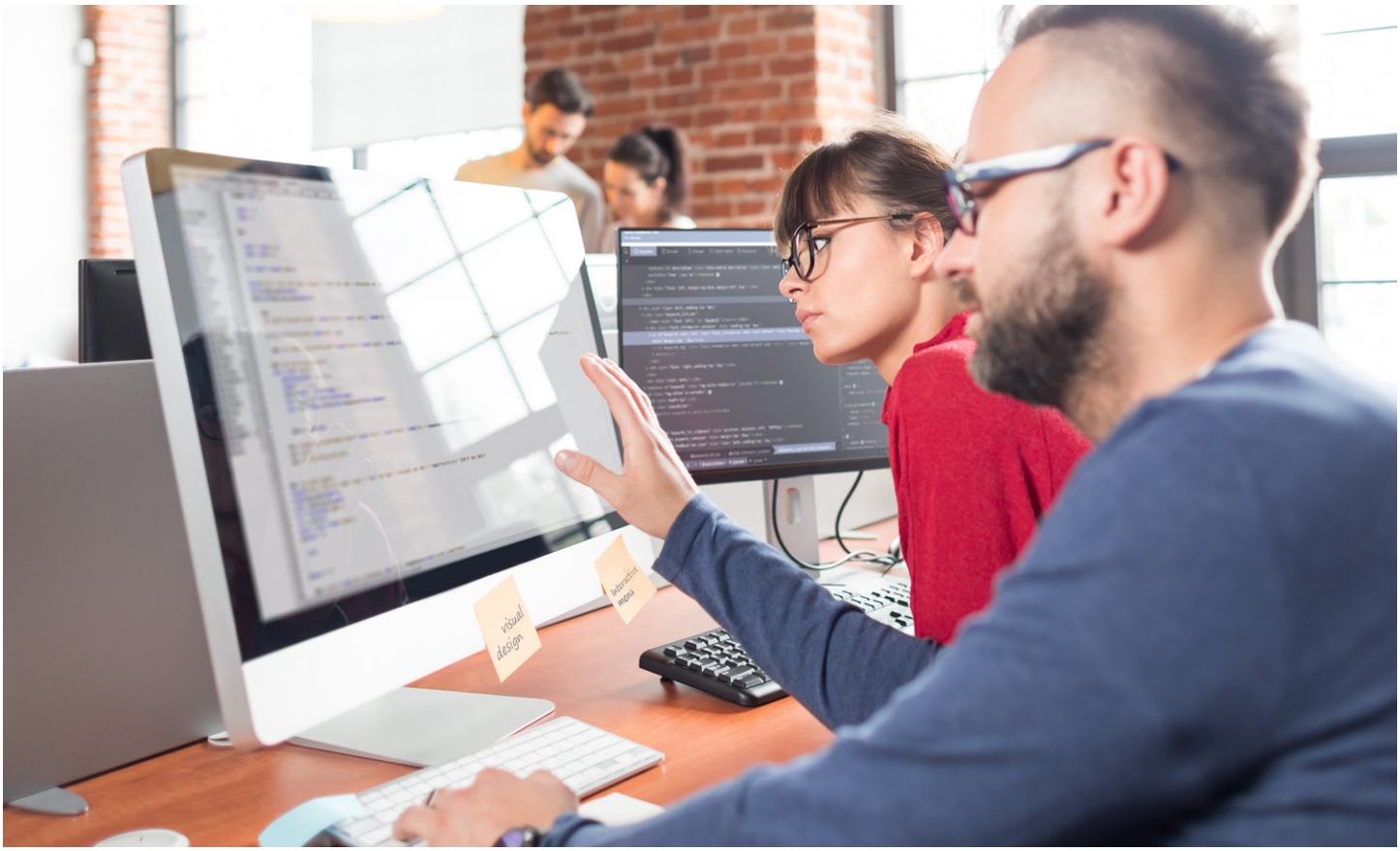
This course introduces Java ecosystem topics that are necessary to develop production-ready applications. It starts by covering the construction and makeup of Java program artifacts. You will learn how Java program code is compiled, packaged and executed. Next, you'll learn how to use Maven to automate and customize the build process, as well as manage external project dependencies. This course also covers the topic of Modules, introduced in Java 9. In addition to build topics, this course will also teach you to use the tools of JUnit 5 to write unit tests and evaluate code coverage. To expand our testing capabilities for complex applications, the Mockito library and test doubles will be covered as well.

Project 3
UdaSecurity

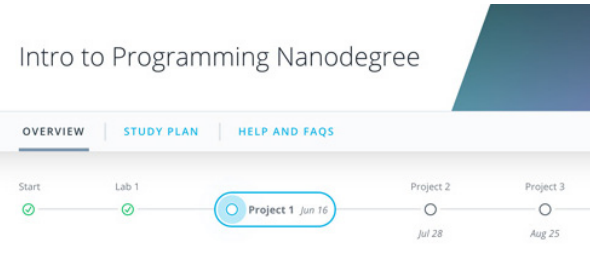
To practice all the skills covered in this course, you'll start with an existing project that needs help. The UdaSecurity program is a basic GUI application that allows users to perform various tasks related to managing their home security system. In order to prepare to scale the software, it's going to need some revisions. You'll need to refactor the program into a multi-module Maven project and you'll also be writing unit tests to verify that it actually does what it claims to do. You'll be using the JUnit 5 and Mockito libraries we cover in this course to write a full unit test suite for the project.

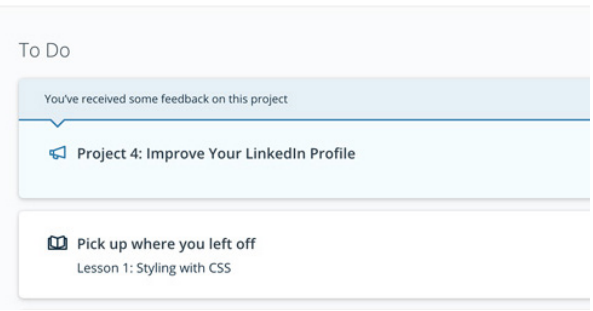
	SUPPORTING LESSON CONTENT	
LESSON ONE	**Running Java Applications**	• Learn to use the tools of the JDK to compile, package and run Java applications. • Recognize and evaluate bytecode. • Use the JShell application to execute arbitrary Java code.
LESSON TWO	**Dependency Management with Maven**	• Read and write well-formed XML, including Maven pom.xml files. • Use Maven to build Java projects and include external dependencies. • Use Maven plugins to modify the build process and perform additional tasks such as generating API documentation and executing static code analysis.
LESSON THREE	**Java Modules**	• Identify and create Java 9 modules using the Module Descriptor class. • Convert existing projects into module format. • Use the JLink tool to create a custom JRE for a specific Java module.

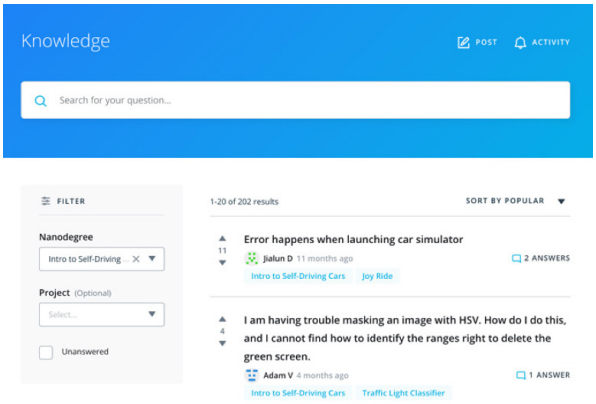
	SUPPORTING LESSON CONTENT	
LESSON FOUR	**Unit Testing with Java**	• Explain the benefits of automated testing and identify the steps in the unit testing lifecycle. • Write unit tests utilizing the features of JUnit 5 to cover all program requirements. • Run unit tests automatically with Maven. • Use tools in IntelliJ to identify any missing code coverage from your unit tests.
LESSON FIVE	**Mocking and Integration Testing**	• Identify and create test doubles to isolate testing requirements from their dependencies. • Use the Mockito library to create and modify the behavior of test doubles. • Understand the roles of Unit Testing, Integration Testing and Functional testing in the automated testing process. • Create mock endpoints using Wiremock to isolate testing requirements from external API calls.

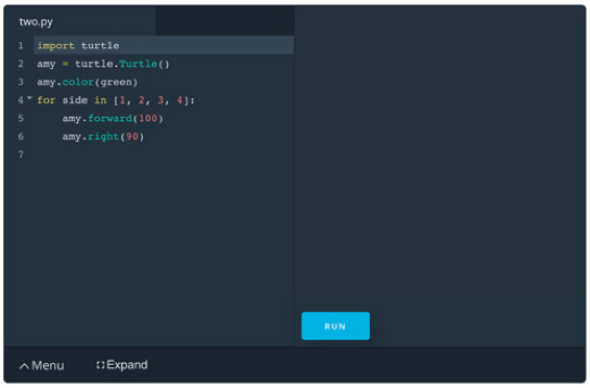


Our Classroom Experience









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

Jeff Phillips

SENIOR SOFTWARE ENGINEER

Jeff has a Master's in Computer Science and over 20 years of experience. He has worked on embedded avionics flight controls systems for both Honeywell and Boeing. Later in his career, he moved into Java Cloud based SAAS applications.

Dustin Hellstern

SOFTWARE ENGINEER

Dustin is a software engineer with 15 years of Java experience, including over 7 years designing and building large-scale systems for one of the top companies in the tech industry. He is excited to share his knowledge with you in this program.

Alex Pritchard

SENIOR SOFTWARE ENGINEER

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 hands-on course about testing and deploying Java applications.

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 computer programming languages today. Java developers are in serious demand. In fact, there are hundreds of thousands of open job opportunities posted on job boards currently. This program was designed to help you take advantage of the growing need for skilled programmers.

WHAT JOBS WILL THIS PROGRAM PREPARE ME FOR?

By the time you graduate, you'll be prepared to land a high-paying role producing web applications in a professional setting. You'll learn more than just writing code. For example, this Java class also teaches students to design and prototype apps using UI best practices to maximize engagement. Other Java tutorials are more piecemeal learning, but this will stand up a junior java developer so they can hit the ground running on a development team.

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

This is an entry level Nanodegree program that teaches the basics of Java programming to individuals who are already familiar with computer programming and are looking to advance their careers in programming.

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.

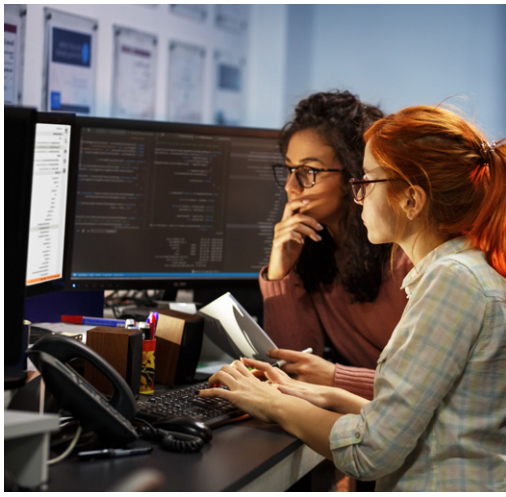
WHAT ARE THE PREREQUISITES FOR ENROLLMENT?

No prior Java experience is required, but you should be familiar with basic programming concepts like variables, loops, conditionals and methods. You should be comfortable running applications on Windows, MacOS or Linux.

FAQs Continued

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

Though no Java experience is required, you may want to prepare with our free What is Programming course or our Intro to Programming Nanodegree program.



TUITION AND TERM OF PROGRAM

HOW IS THIS NANODEGREE PROGRAM STRUCTURED?

The Java Programming Nanodegree program is comprised of content and curriculum to support three projects. We estimate that students can complete the program in three months, working five to ten 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.

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

Please see the Udacity Program Terms of Use and FAQs for policies on enrollment in our 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.