FULL STACK WEB DEVELOPER

Udacity courses

Hankyu Jang

Index

Udacity

Full Stack Web Developer Nanodegree

Building Complex Web Applications (6–9 months, $200/mon.)

(co created by at&t, Google, GitHub, Amazon web services)

Nanodegree program Summary

This Nanodegree program is the most efficient curriculum to prepare you for a job as a Full Stack Web Developer. Participants in the program will learn to build complex server-side web applications that make use of powerful relational databases to persistently store data. You'll then learn how to secure and configure your very own Linux-based server to host your applications. Finally, you'll explore the challenges in horizontally scaling an application to support thousands of users with a cloud-based application hosting provider.

You'll also have the opportunity to prepare for your new career with reviews of your online presence (resume, LinkedIn, portfolio), prepare for interviews, take part in workshops covering topics like networking and salary negotiation as well as take part in a new program facilitating job placement.

What is a Nanodegree program Curriculum?

A Nanodegree program is both a curriculum and a credential, developed in partnership with leading technology companies. Our belief is the best way to establish the ultimate hireability of our students is to see them certified with credentials created - and endorsed - by the very companies where they want to work. We make this possible by:

- creating best-in-class courses taught by expert instructors
- deploying a responsive and rigorous review model
- enabling the creation of viable employer-ready work portfolios
- equipping students with the presentation skills necessary to show themselves in the best career light possible

Why Take This Nanodegree program?

The Full Stack Web Developer Nanodegree is designed to prepare you for a career in web development. As a Full Stack Web Developer, you are the "jack of all trades" who companies rely on to build, support and maintain their web applications. With our industry partners, we've carefully crafted the most efficient set of projects and skills training to guide you along the way.

In this Nanodegree you will hone your understanding of how the web works, you'll develop complex relational databases used to store your applications' data, you'll secure and configure your own Linux-based servers, and you'll build complete web applications using HTML, CSS, JavaScript and Python.

Prerequisites and Requirements

Prerequisites:

In order to determine if the Full Stack Web Developer Nanodegree is a good fit for you, please take the Readiness Assessment. This will help you determine if you are ready to start.

Minimum Requirements:

1. Experience using HTML, CSS and JavaScript to build complex client-side applications. If you do not have experience in these languages, we encourage you to enroll in our Front-End Web Developer Nanodegree.
2. Beginner-level experience in Python. If you do not have this experience, check out our Intro to Programming Nanodegree or the first three lessons of Intro to Computer Science.
3. Experience using Git for version control. If you do not have this experience, check out our How to Use Git and GitHub course.
4. You are self-driven and motivated to learn. Participation in this program requires consistently meeting the deadlines set for your cohort and devoting at least 10 hours per week to your work.
5. You can communicate fluently and professionally in written and spoken English.
6. You have access to a computer with a broadband connection, on which you'll install a professional code/text editor (ie. Sublime Text or Atom) as well as virtual machines (using VirtualBox and Vagrant).
7. You are willing to contribute to the success of the program, including collaborating with fellow students and giving us feedback on how we can improve.

Desirable Prior Experience:

1. You've completed an online programming course.
2. You've tried to build server-side applications in the past and want to learn how to do it like a pro.

See the Technology Requirements for using Udacity.

Nanodegree program Structure

These are the projects you'll build and the classes that will prepare you to build each of the projects. You'll have access to all these in the Nanodegree:

PROJECT Movie Trailer Website

You will write server-side code to store a list of your favorite movies, including box art imagery and a movie trailer URL. You will then serve this data as a web page allowing visitors to review their movies and watch the trailers.

Prepare for this project with: Programming Foundations with Python

PROJECT Tournament Results

You will develop a database schema to store the game matches between players. You will then write code to query this data and determine the winners of various games.

Prepare for this project with: Intro to Relational Databases

PROJECT Item Catalog

You will develop an application that provides a list of items within a variety of categories as well as provide a user registration and authentication system. Registered users will have the ability to post, edit and delete their own items.

Prepare for this project with:

- Full Stack Foundations
- Authentication and Authorization

PROJECT Conference Organization App

You will develop a cloud-based API server to support a provided conference organization application that exists on the web as well as a native Android application. The API supports the following functionality found within the app: user authentication, user profiles, conference information and various manners in which to query the data.

Prepare for this project with: Developing Scalable Apps with Python

PROJECT Linux-based Server Configuration

You will take a baseline installation of a Linux distribution on a virtual machine and prepare it to host your web applications, to include installing updates, securing it from a number of attack vectors and installing/configuring web and database servers.

Prepare for this project with:

- Configuring Linux Web Servers
- Linux Command Line Basics

Additional Information

We will be periodically reviewing the Full Stack Nanodegree curriculum - both classes and projects - and making adjustments based on student and industry feedback. We will notify active students working toward the Nanodegree about effects this may have on the degree requirements.

https://www.udacity.com/course/full-stack-web-developer-nanodegree--nd004

Front-End Web Developer Nanodegree

Creating Stunning User Experiences

(co created by at&t, Google, GitHub, Hack Reactor)

Nanodegree program Summary

Learn the fundamentals of how the web works and gain a working knowledge of the three foundational languages that power each and every website: HTML, CSS and JavaScript. This Nanodegree will provide a guided, efficient path for you to learn to build beautiful, responsive websites optimized for security and performance. You'll see the efforts of your work with each click of the browser's refresh button!

By the end of the Nanodegree you'll have built a diverse portfolio of projects to show employers. You'll also have the opportunity to prepare for your new career with reviews of your online presence (resume, LinkedIn, portfolio), prepare for interviews, take part in workshops covering topics like networking and salary negotiation as well as take part in a new program facilitating job placement.

What is a Nanodegree program Curriculum?

A Nanodegree program is both a curriculum and a credential, developed in partnership with leading technology companies. Our belief is the best way to establish the ultimate hireability of our students is to see them certified with credentials created - and endorsed - by the very companies where they want to work. We make this possible by:

- creating best-in-class courses taught by expert instructors
- deploying a responsive and rigorous review model
- enabling the creation of viable employer-ready work portfolios
- equipping students with the presentation skills necessary to show themselves in the best career light possible

Why Take This Nanodegree program?

This Nanodegree program will teach you the skills required to become a Front-End Web Developer. We've designed this curriculum with expert web developers and hiring managers, allowing you to demonstrate your skills by completing a series of projects approved by leading employers as the critical indicators of job-readiness. Specifically, you'll:

- create a professional portfolio using HTML and the Bootstrap CSS framework
- use the power of jQuery's DOM manipulation to dynamically populate a resume with your own information
- develop your very own arcade game in JavaScript with the HTML5 Canvas API
- discover how to optimize your application's perceived load time by taking the Critical Rendering Path into account

- learn that frames per second isn't just important for games and how to ensure a silky smooth experience in your applications by optimizing for 60 frames per second
- explore best practices in application architecture and design patterns
- build a complex mapping application using the Knockout framework and a variety of third-party API services
- develop and maintain applications with the confidence test-driven development promotes, using the Jasmine testing framework

Prerequisites and Requirements

In order to determine if the Front-End Web Developer Nanodegree is a good fit for you, please take theReadiness Assessment. It will ask you a series of technical and time management questions to help you determine if now is the right time to start the program. General prerequisites are summarized below.

General Requirements:

- You are self-driven and motivated to learn. Participation in this program requires consistently meeting the deadlines set for your cohort and devoting at least 10 hours per week to your work.
- You can communicate fluently and professionally in written and spoken English.
- You are willing to contribute to the success of the program, including collaborating with fellow students and giving us feedback on how we can improve.

Front-End Developer Nanodegree Specific Requirements:

- You have access to a computer with a broadband connection, on which you'll install a professional code/text editor (ie. Sublime Text or Atom).
- You can independently solve and describe your solution to a math or programming problem
- You are familiar with basic programming concepts such as variables, conditions and loops.

See the Technology Requirements for using Udacity.

https://www.udacity.com/course/front-end-web-developer-nanodegree--nd001

Developing Scalable Apps in Python
with Google App Engine (6 weeks, Free, Google)

Course Summary

You will learn about challenges of building applications that can serve hundreds of thousands of users, and how you can prepare for them as a developer. And more importantly - you will learn how to harness the power of **App Engine** - Platform as a Service, run by Google, so you can focus on your application's features, not on managing infrastructure that runs your app. Let Google run and scale the infrastructure and do what you do best - write code!

However, to use App Engine effectively, you have to learn how it works, and this is exactly what this course will teach! You will also learn the best practices of using Cloud Endpoints that allow you to easily create API services and make them accessible to iOS, Android and Javascript clients. They allow you to automatically generate client libraries to make wiring up the frontend easy. And there are some nifty built-in features, like denial-of-service protection and OAuth 2.0 support.

Why Take This Course?

Cloud computing is one of the fastest growing fields right now. And no wonder - it provides an easy and affordable way to run your applications. However, the traditional way of hosting and scaling applications on Virtual Machines in the Cloud comes with a cost - even if the infrastructure is virtual, you still have to manage it - do load balancing, bring instances up and down, take care of patching your software and in general spend a lot of your time and resources on just the infrastructure.

Google is one of the pioneers in the business of scaling, and now you can use their infrastructure, and let them do all the scaling work so that you can focus on the unique features your app provides.

Are you a Java developer? If so, maybe you'd like to check out our Developing Scalable Apps with Java course.

Project

There will be an overarching project throughout the course, starting from Lesson 2. You will develop an application for organizing conferences, similar to sites like meetup.com or eventbrite.com. We have provided the frontend of the app, and all your effort will go into making the backend.

And because you will be using Cloud Endpoints, it’s really easy to hook up your backend with a native mobile app. We have created an Android app that uses the same backend as the web frontend. You will able to recompile it against your backend, if you so wish!

At the end of the course you will do a final project, where you will have to expand the functionality of the same app.

Prerequisites and Requirements

You should be fairly comfortable programming in Python, preferably with some experience developing web applications and working with databases for at least a year. You will be developing the backend of a sample app, so you don’t have to worry about HTML or JavaScript.

You should have sufficient permissions to install new software on your computer, and comfortable configuring it, including setting up system variables.

We will be using Maven and Eclipse for this course, and we strongly suggest that you use them as well.

See the Technology Requirements for using Udacity.

What Will I Learn?

Syllabus

Overview

This course consists of 6 lessons. First one is an overview of cloud computing and the benefits of Platform as a Service. Lessons 2-5 will cover important theoretical concepts of Google App Engine and also plenty of hands-on exercises implementing what you have learned. Lesson 6 will be a short intro on an easy way to create native mobile apps that talk to your backend.

Lesson 1: Scalability Basics

Do you know how and why server racks were invented years ago? Have you realized all of the problems that you have to solve if you will try to scale your app by yourself? Learn about a better way.

Lesson 2: Getting Started

Set up your first App Engine project and learn how to define Cloud Endpoints.

Start the course project app - Conference Central and add authentication and user profiles.

Lesson 3: Storing and Retrieving Data

One of the most important things you want to do in your app is storing data. In this lesson you will learn what is Google App Engine Datastore, how is it different from RDBMS and how you can use it in your applications.

Add the ability to store and retrieve user profiles and conferences to the course project app.

Lesson 4: Advanced Datastore Concepts

Learn more about how Datastore works, including queries, filters, indexes and transactions.

Add different ways to query conferences.

Lesson 5: Advanced App Engine Topics

There are a lot of things you might want your application to do in background, without making the user wait, or even initiate the process. Learn about task queues and cron jobs and how you can use them to add advanced functionality to your app. You will also learn about Memcache, and get an overview of topics like Edge Caching, AppStats and other methods of optimizing your apps performance.

Add a push queue, cron job and a customized Memcache entry to your app.

https://www.udacity.com/course/developing-scalable-apps-in-python--ud858

Full Stack Foundations

Build a data-driven web app with Python (3 week, Free, AWS)

Course Summary

In this course you will learn the fundamentals of back-end web development! You will create your own web application that queries a database for items on restaurant menus and then dynamically generates complete menus in the form of web pages and API endpoints.

You'll start by learning how to interact with a database from a web application using an Object-Relational Mapping (ORM) layer. From there, you'll learn how GET and POST requests translate to CRUD operations. You'll then explore the Flask framework and the various ways in which it can speed up the development of your applications. Finally, you'll develop your very own web application from the ground up using the iterative development process.

Why Take This Course?

Interacting with a persistent datastore is what transforms static web pages into powerful and effective web applications. Giving users the ability to create, read, update and delete data is the backbone of the most popular services on the Internet today. But, you must do so safely and securely - the smallest mistake could bring your application down or leak your user's data.

In this course, you'll discover how all of the major components of web applications work and best practices in developing secure, data-driven web applications.

Prerequisites and Requirements

You should be comfortable with Python, including Object-Oriented Programming. If you'd like to brush up on your Python, try the first three lessons of our Intro to Computer Science course. For Object-Oriented Programming in Python, see Programming Foundations with Python.

You should also have a firm understanding of SQL and working with a relational database, particularly PostgreSQL. Our Intro to Relational Databases course can get you started if you feel you're not yet prepared.

Finally, you should be able to read and write HTML and CSS without any guidance. If you need a refresher on HTML and CSS, start with our Intro to HTML and CSS course.

To complete the final project for this course, it is ***critical*** you meet these prerequisites as you are given no assets to begin with and will truly build a complete web application from scratch.

See the Technology Requirements for using Udacity.

What Will I Learn?

Syllabus

Lesson 1 - Working with the CRUD

In the first lesson, you will learn about CRUD; Create, Read, Update, and Delete. You will learn why this acronym is important in web development and implement CRUD operations on a database. You will also learn to use an ORM (Object-Relational Mapping) as an alternative to SQL.

Lesson 2 - Making a Web Server

In the second lesson, you will build a web server from scratch using Python and some of the pre-installed libraries it includes. You will learn what GET and POST requests are and how we use them to retrieve and modify information on a web site. We will then use the concepts learned in Lesson 1 to add CRUD functionality to our website.

Lesson 3 - Developing with Frameworks

In the third lesson, we will discuss web frameworks like Django and Ruby on Rails. You will see how web frameworks simplify the development process and allow us to create web applications faster. We will use the Flask web framework to develop our own web application. We will also discuss API's (Application Programming Interfaces) and add JSON (JavaScript Object Notation) endpoints to our application to allow data to be sent in a format alternative to HTML.

Lesson 4 - Iterative Development

In the last lesson, you will build an entire web application on your own. You will learn about the iterative development process and how developing iteratively allows you to have a working prototype throughout all stages of the development process.

https://www.udacity.com/course/full-stack-foundations--ud088

Authentication & Authorization: OAuth

Implementing Web Security with OAuth 2.0 (2 weeks, Free)

Course Summary

As a Python programmer, leveraging Flask allows you to quickly and easily build your own web applications. But before you share your apps on the Internet you should protect your users' data, ensuring information stored on your site is safe from unwanted manipulation. You could implement web security and permissions on your own, but relying on trusted providers is a faster, safer, and easier way to allow users to login to your application - without having to create and maintain another account, profile, and password.

In this course, you will learn to implement the OAuth 2.0 framework to allow users to securely login to your web applications. You'll be provided a restaurant menu application created in Flask. By the end of this course, you will write the necessary code to implement Google+ Sign-In and Facebook Login in options so users can create restaurant menus that are viewable by everyone but only modifiable by the original creator.

Why Take This Course?

OAuth 2.0 is a popular framework that allows users to login to your web application by using third party sign ins, from providers they've already created and trust, with the click of a button. And because passwords and sensitive data are never sent, your web application does not have to deal with the complexities of secure password storage and security breaches. Your users can then control the level of access your application has to their data, and change or revoke this access at any point in time.

Prerequisites and Requirements

This course was built to expand upon the concepts introduced in Full Stack Foundations, specifically:

- performing CRUD operations
- making use of templates
- developing with the Flask framework

Additionally, HTML, JavaScript, AJAX are heavily used in this course. A basic understanding of these technologies is needed to get the most out of these lectures.

If you'd like to refresh your HTML knowledge start with our Intro to HTML and CSS course. You can check out the JavaScript Basics and Intro to AJAX courses to brush up on these topics as well.

See the Technology Requirements for using Udacity.

What Will I Learn?

Syllabus

Lesson 1 - Authentication vs. Authorization

Learn the difference between the concepts of authentication and authorization and address some major security concerns that developers must protect against when developing a web application. You will learn how OAuth 2.0 makes implementing security easier for developers and users alike by allowing your users to sign in to your applications while keeping all of the security on well-known and trusted OAuth providers. Finally, you will see OAuth 2.0 in action as you make API requests using Google's OAuth 2.0 Playground.

Lesson 2 - Creating a Google+ Sign-In

Learn about the different types of security flows your application can implement. You will see how security can be handled by your server, your user's browser, or both depending on the type of security your application needs. You will then add a Google+ Sign-In to an existing web application and implement a hybridized client/server flow.

Lesson 3 - Local Permission Systems

Add python code to create server-side rules that will constitute a permission system. This system will limit access of the database for each logged in user based on how the developer designs this code. You will add a User model model to your database to store the credentials, such as username, email, and profile picture, collected from the OAuth provider's API.

Lesson 4 - Adding Facebook & Other Providers

Learn to implement multiple OAuth providers on your web application. You will add Facebook Login as an alternative sign in option for your users and understand how to use OAuth provider documentation to add as many providers as you see fit for your application.

https://www.udacity.com/course/authentication-authorization-oauth--ud330

Configuring Linux Web Servers

Your First Ubuntu Server (1 week, Free)

Course Summary

In this course you'll learn the basic Linux fundamentals every web developer needs to know to share their web applications with the world! You'll get a basic Python WSGI application up and running within a Vagrant virtual machine that queries data from a PostgreSQL database.

You'll start by exploring various Linux distributions and learning the differences between a number of them. You'll then explore how the Linux operating system differs from other operating systems you may be more familiar with. With this base knowledge, you'll then move into Linux security - covering topics such as file permissions, user management, package management and configuring firewalls. Finally, you'll transform a safe and secure baseline server into a web application server by installing and configuring the Apache HTTP Server and PostgreSQL database server.

Why Take This Course?

A basic understanding of linux systems administration is required to not only get your web application up and running for the world to see, but also to ensure it continues operating efficiently. In this course, you'll explore a bare-bones linux system and how it differs from desktop environments you are currently familiar with. You'll then address a number of security concerns full stack developers must contend with and, finally, serve one of your applications from your very own piece of the Internet.

Prerequisites and Requirements

You should be comfortable with your terminal and working within a shell, if you need a refresher take a look at our Intro to the Shell.

To actually get a web application up and running you should be familiar with the Python programming language and PostgreSQL. Programming Foundations with Python and Intro to Relational Databases can help you get up to speed if you feel you need some additional practice.

See the Technology Requirements for using Udacity.

What Will I Learn?

Syllabus

Overview

This course consists of 3 lessons which will take you from absolutely zero knowledge about the Linux operating system all the way to hosting your very own data-driven application that is publically accessible to everyone in the world!

Lesson 1 - Intro to Linux

In this lesson you will be introduced to the Linux operating system and the various available distributions. You will then setup your very own Linux virtual machine on your own computer and explore that system, learning how the filesystem is organized and identifying key files.

Lesson 2- Linux Security

As you unleash your new server onto the Internet you'll need to be a good citizen - and this means, **security**! In this lesson, you'll learn how to execute administrative tasks, update the software on your system, install new software, and manage users. You'll then learn how to implement even stronger authentication mechanisms, how to interpret Linux file permission and, finally, how to configure a firewall to keep your system secure.

Lesson 3 - Web Application Servers

In this lesson you'll transform your secure and safe barebones server into a fully functional web application server by installing and configuring the Apache HTTP Server and PostgreSQL database server. You'll then write a very basic Python WSGI application that can query a database and present that data upon a web request.

https://www.udacity.com/course/configuring-linux-web-servers--ud299

Intro to Relational Databases

SQL, DB-API, and More! (4 weeks, Free)

Course Summary

This course is a quick, fun introduction to using a relational database from your code, using examples in Python. You'll learn the basics of SQL (the Structured Query Language) and database design, as well as the Python API for connecting Python code to a database. You'll also learn a bit about protecting your database-backed web apps from common security problems.

After taking this course, you'll be able to write code using a database as a backend to store application data reliably and safely.

Why Take This Course?

If you look under the hood of a lot of major web sites — from Wikipedia to Reddit — you'll find a relational database somewhere.

Database systems such as PostgreSQL and MySQL have been part of the web developer's toolkit for many years, and remain some of the most powerful tools available for storing and manipulating structured data.

If you're planning to continue on in full-stack development, knowing about databases is essential background. Even though many toolkits hide the details of the database from your application code, being able to interact with the database will serve you well in designing, debugging, and maintaining your applications.

Prerequisites and Requirements

You can read and write basic code in Python. This course uses programming exercises in Python. If you haven't worked with Python before, check out our course Programming Foundations with Python.

If you can understand this code (maybe with the help of the **random** module documentation), you know enough Python for this course:

```
import random


def ChooseTwice(items):

    a = random.choice(items)

    b = random.choice(items)

    return a, b


names = ["Alice", "Bob", "Charlie", "Debra"]

(one, two) = ChooseTwice(names)

if one == two:
    print "%s is happy!" % one
else:
    print "%s likes %s!" % (one, two)
```

You can use a command-line interface (terminal). Some of the exercises in this course involve using a Unix-style command-line interface to enter commands, run Python programs, and navigate directories.

If you have taken our course on Git and Github, the level of command-line use in this course is similar.

You don't need to be a Web programmer. This course does include a small Web application and some HTML and JavaScript in examples, but you will not need to make changes in these languages.

You don't need any previous database experience. This course is an entry-level introduction to relational databases.

You need a programming text editor (such as Sublime Text) installed on your computer. You should be able to use it to open and edit files of Python code.

See the Technology Requirements for using Udacity.

What Will I Learn?

Syllabus

Lesson 1: Data and Tables

In this lesson, you'll learn about how relational databases let you structure data into tables. You'll learn about the importance of unique keys and relationships between tables.

Lesson 2: Elements of SQL

In this lesson, you'll begin learning SQL, the Structured Query Language used by most relational databases. You'll learn about the **select** and **insert** statements, the basic operations for getting data out of a database and putting data into a database. You'll learn about the operators and syntax available to get the database to scan and join tables for you.

Lesson 3: Python DB-API

In this lesson, you'll learn how to access a relational database from Python code. You'll use a virtual machine (VM) running on your own computer to run a Python web application, and adapt that application to use a database backend. Then you'll learn about some of the most common security pitfalls of database-backed applications, including the famous Bobby Tables. This lesson also covers the SQL **update** and **delete**statements.

Lesson 4: Deeper Into SQL

In this lesson, you'll learn how to design and create new databases. You'll learn about normalized design, which makes it easier to write effective code using a database. You'll also learn how to use the SQL **join**operators to rapidly connect data from different tables.

Lesson 5: Final Project

In this project, you'll use your Python and SQL knowledge to build a database-backed Python module to run a game tournament. You'll design the database schema and write code to implement an API for the project.

https://www.udacity.com/course/intro-to-relational-databases--ud197

How to Use Git and GitHub

Version Control for Code (3 weeks, Free)

Course Summary

Effective use of version control is an important and useful skill for any developer working on long-lived (or even medium-lived) projects, especially if more than one developer is involved. This course, *built with input from GitHub*, will introduce the basics of using version control by focusing on a particular version control system called Git and a collaboration platform called GitHub.

This course is part of the Front End and Full Stack Nanodegrees..

Why Take This Course?

Git is used by many tech companies, and a public GitHub profile serves as a great portfolio for any developer. But more than that, you'll establish an efficient programming workflow that allows you to:

- Keep track of multiple versions of a file
- Track bugs by reverting to previous working versions of a file
- Seamlessly collaborate with other developers on a project

The use of tools like Git and GitHub is essential for collaborating with other developers in most professional environments.

Prerequisites and Requirements

While this course does not involve programming, students should have some experience with a programming or markup language. Additionally, students should be familiar with navigating the command line. If unfamiliar or if you want a refresher, check out these instructions or this introductory course.

See the Technology Requirements for using Udacity.

What Will I Learn?

Syllabus

Lesson 1: Navigating a Commit History

In this lesson, you'll learn about a few different types of version control systems and discover what makes Git a great version control system for programmers. You'll also get practice using Git to view the history of an existing project. You'll learn to see all the versions that have been saved, checkout a previous version, and compare two different versions.

Lesson 2: Creating and Modifying a Repository

In this lesson, you'll learn how to create a repository and save versions of your project. You'll learn about the staging area, committing your code, branching, and merging, and how you can use these to make you more efficient and effective.

Lesson 3: Using GitHub to Collaborate

In this lesson, you'll get practice using GitHub or other remote repositories to share your changes with others and collaborate on multi-developer projects. You'll learn how to make and review a pull request on GitHub. Finally, you'll get practice by collaborating with other Udacity students to write a create-your-own-adventure story.

Project: Contribute to a Live Project

Students will publish a repository containing their reflections from the course and submit a pull request to a collaborative Create-Your-Own-Adventure story.

https://www.udacity.com/course/how-to-use-git-and-github--ud775

Linux Command Line Basics

Getting Started with the Shell (1 week, Free)

Course Summary

We have built this course for beginners who have no experience with the Linux system and the command-line interface.

In this course, you'll learn the basics of the command line interface of a Linux server: the terminal and shell (GNU Bash). This course includes an introduction to files and directories in the Linux filesystem.

Why Take This Course?

Most servers on the Internet today run on Linux or other Unix-like systems. Installing, configuring, and troubleshooting often relies on the command line interface. This, accordingly, is foundational web knowledge, and in fact many of our intermediate and advanced courses rely on a familiarity with the command-line interface to run servers, work with version control systems and more.

Prerequisites and Requirements

To take this course, you should have beginner-level experience in a programming languages such as Python or JavaScript. While this course does not involve doing any programming, it does use concepts that are familiar to the beginning programmer such as "function", "expression", and "string".

Note: This course is intended for beginners to the shell environment. If you have done shell scripting or other extensive use of the shell before, this course will probably be too introductory for you. You might want to check out our Configuring Linux Web Servers course.

See the Technology Requirements for using Udacity.

What Will I Learn?

Syllabus

Lesson 1: Get Into the Shell

In this lesson, you'll learn about the terminal user interface and how you can interact with a Linux server using shell commands.

Lesson 2: Shell Commands

There are different kinds of shell commands that work with the terminal in different ways. This lesson also covers the use of the Linux manual (manpages) to expand your knowledge of shell commands.

Lesson 3: The Linux Filesystem

Working with files and directories is a big part of using the shell. In this lesson you'll learn basic commands for interacting with the filesystem.

https://www.udacity.com/course/linux-command-line-basics--ud595