After This Course

This is a list of things you can learn after completing this course.

Remember that you don't need to learn everything right now. Just pick the things that you find the most interesting / useful.

A good guideline to keep in mind is to focus on "things that don't change". With that I mean focus on the fundamentals instead of the hot new framework / language.

Things like data structures & algorithms, SQL databases & Linux skills are always a safe bet because they are going to be useful for many years to come :)

Testing / TDD

Having tests is a great way to reduce the amount of errors in your code. As an additional benefit practicing TDD will help you drive the design of your code.

There aren't that many up-to-date testing materials out there, but I would recommend checking out the official site and documentation for RSpec to get you started.

- Official Rspec Site
- RSpec Documentation
- RSpec Tutorial
- Discover TDD
- Minitest

Refactoring & Coding Style

We read code more often than we write, so it's important to write code for humans first and computers last.

You should learn good coding practices and refactoring techniques to make your code easier to work with.

- Ruby Style Guide
- Rubocop
- Ruby Critic
- Refactoring techniques
- Clean Coders

Object-Oriented Design

Good coding is not just style & refactoring, you also need to keep in mind that we are working in an Object-Oriented language.

There are many good practices & patterns developed around OOP which generally come under the name of "Object-Oriented Design".

Here are some resources for you to study:

- Tell don't ask
- Command/Query separation
- Design patterns

Computer Science / Algorithms

Learning about computer science can be a great way to improve your programming skills. In particular you should learn about common sorting and searching algorithms and data structures.

- Princeton has a great book and a free online course that you should check out if you are interested in this topic.
- Here is another online course from Standford: Algorithms: Design and Analysis
- Steven Skiena has a good set of algorithm videos here: Analysis of Algorithms
- If you want to practice your algorithm skills you could try the exercises in this site: https://leetcode.com/problemset/algorithms/

Javascript

If you do any web development at all you will have to deal with Javascript at some point. Here are two free resources you can use to get started or improve your current skills:

- Mozilla Developer Network
- Eloquent Javascript

If you are interested in making games, javascript has plenty of libraries to help you out. In particular, you should check out Phaser.

Databases

Working with data is an important part of many applications, so you should familiarize yourself with the different database solutions that exist out there. The most common type of database is based on the concept of relational data and SQL (Structured Query Language).

The two most popular databases in the Ruby world are **PostgreSQL** and **MySQL**. I would recommend going with PostgreSQL since it has way more features than MySQL.

- The PostgreSQL documentation is pretty decent
- I wrote an ebook titled "PostgreSQL for Ruby developers" that covers the basics

NoSQL Databases

What about NoSQL?

You may have used or heard about the so called NoSQL databases. This term encompasses pretty much every other database that does not use the SQL language as an interface. These include things like key-value stores (redis) or document databases (mongodb).

Here are some popular ones for you to investigate:

- Redis
- Mongodb
- Cassandra

I wouldn't recommend using these to replace a regular SQL database, but they can still be useful in certain situations.

For example, Redis is really good for caching.

Linux

Linux is pretty popular in the Ruby world, specially for running production Ruby apps, so it would be a good idea for your to become more familiar with this operating system and the different tools available.

- The Linux Command Line: A Complete Introduction
- Bash Scripting
- Interactive Linux Lessons

Get familiar with the different status & debugging commands you can use like: htop, dstat, lsof, strace, etc.

Web Security

All your efforts would be wasted if an attacker could just destroy your servers by breaching your security.

It's a good idea to learn what are some of the typical mistakes that lead into security issues and how to avoid them.

Here are two resources you can use to get started:

- Rails Security Guide
- Owasp Top 10

Version Control Systems

A version control system (VCS) lets you keep track of changes over time to your code base.

This can be an invaluable tool when evaluating progress & checking for previous version of the code.

Here are my recommended resources:

- Official Site
- Git Cheatsheet
- Resources to learn Git

Soft Skills

Not everything is technical stuff, you also have to learn how to communicate & deal with people.

For that I can recommend the following book:

- https://www.amazon.com/How-Win-Friends-Influence-People/dp/0671027034