

What's code?

Wanna know how everything in the universe works?



Justin ✓
🔒 Apr 14, 2020



25



The TL;DR

Code is **step by step directions, but for computers.**

- Code is a way to **tell your computer what to do** under different circumstances
- All code eventually gets broken down into **ones and zeros**
- Developers write code in lots of **different languages** built for different purposes
- Most code that people write today is built on **packages of other people's code**

Code is probably the most fundamental building block of the difference between life today and life 20 years ago. So read this!

Code is just telling a computer what to do

The goal of any piece of software – as large as Gmail or small as a little calculator – is to **get something done**. The easiest way to understand code is to try and think through how you'd get that *thing* done, step by step. If we wanted to build Gmail, or any email client, we'd need to make sure it does a few things:

- Show me my emails
- Let me send and reply to emails
- Let me mark emails as junk or flagged

And if we wanted to dive deeper into one of these – let's choose *Let me send and reply to emails* – we could break it down into even *more* specific tasks:

- A text editor to write emails

- Address fields to put in recipients and cc / bcc
- A little “swoosh” sound when we send an email

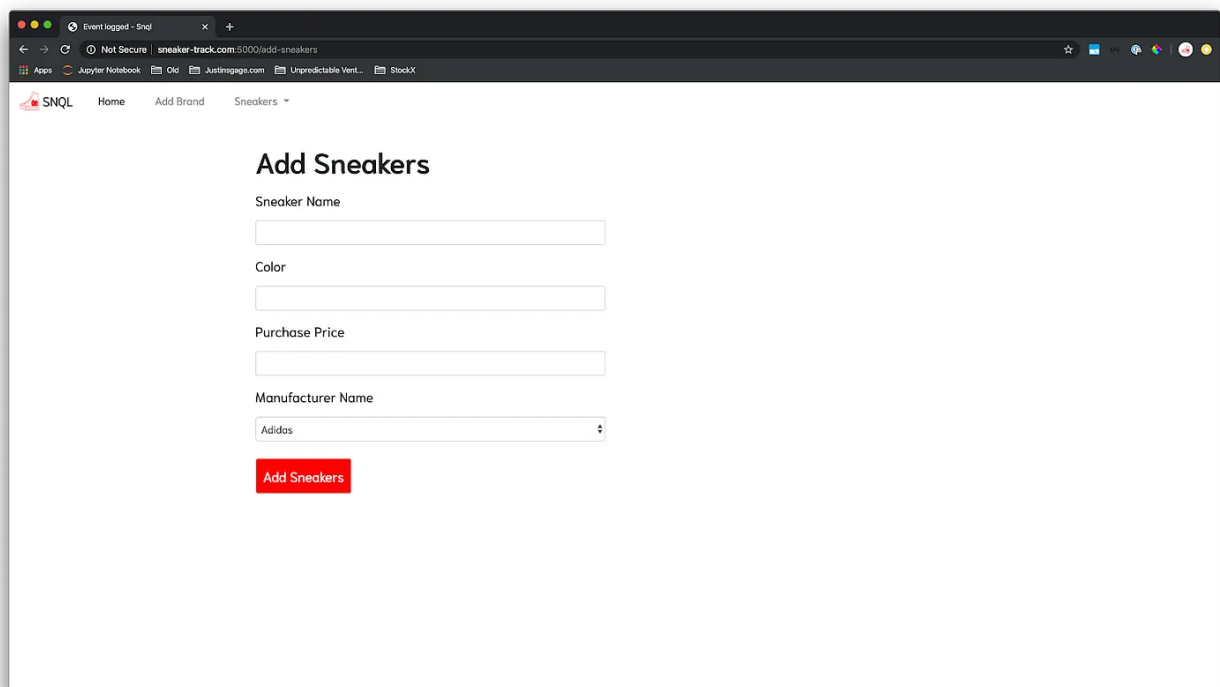
We can also break these into *even more* specific tasks – let's choose *Address fields to put in recipients and cc / bcc*:

- A cc field
- A bcc field
- Show cc in sent emails, but don't show bcc
- Auto-complete email addresses in these fields

You get the point by now. Every “feature” – or *thing* you want to build – eventually breaks down into “when your user does X, the program does Y.” Code is how you make that happen in practice.

To make this even more concrete, let's take a look at a little app I built last year called SNQL – it helps you track the sneakers you own and when you wear, trash, or sell them. I...need help.

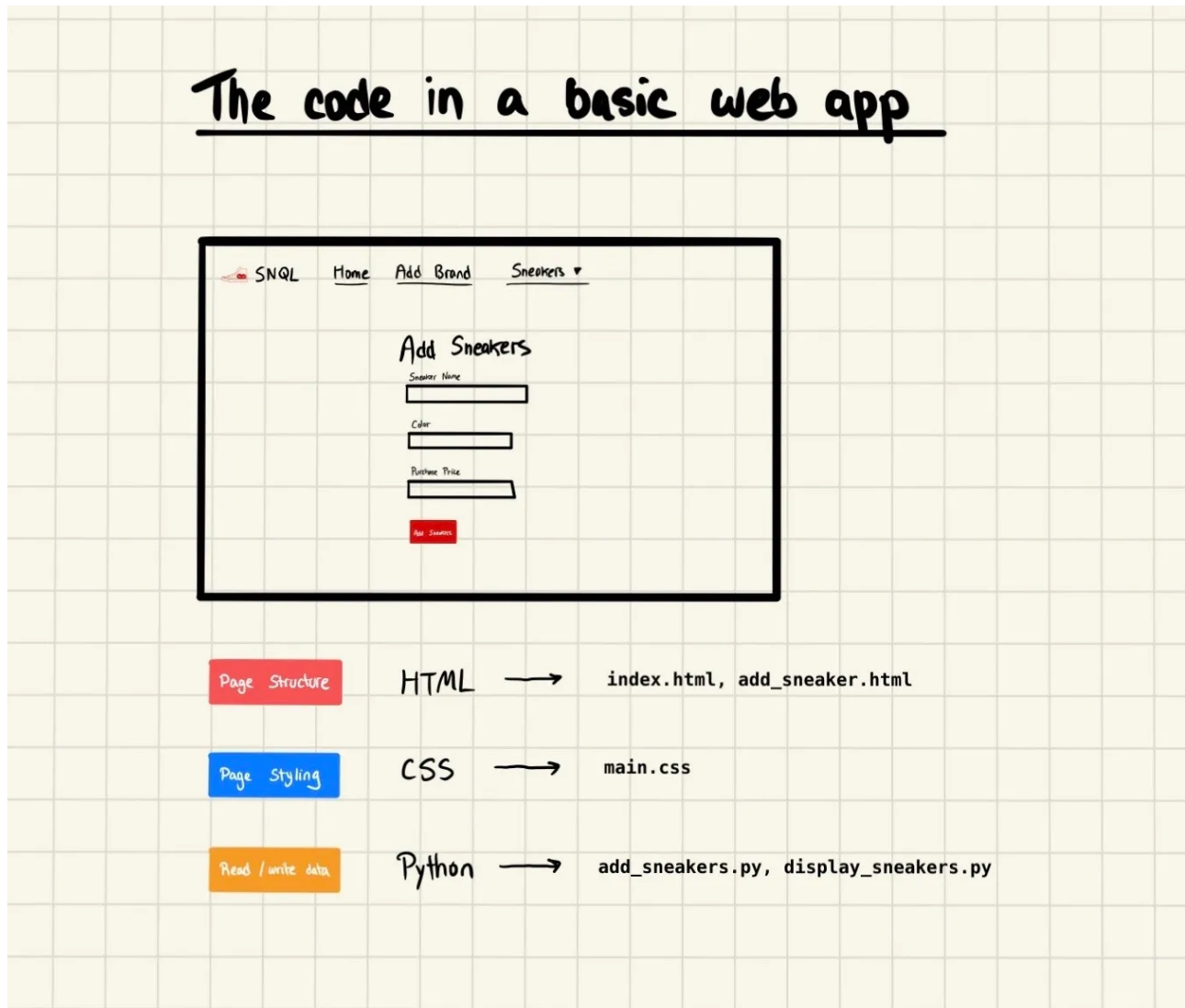
Here's what it looks like (*I know, it's not pretty*):



The screenshot shows a web browser window with the URL `sneaker-track.com:5000/add-sneakers`. The page has a navigation bar with 'SNQL', 'Home', 'Add Brand', and 'Sneakers'. The main content area is titled 'Add Sneakers' and contains a form with the following fields: 'Sneaker Name' (text input), 'Color' (text input), 'Purchase Price' (text input), and 'Manufacturer Name' (dropdown menu with 'Adidas' selected). A red 'Add Sneakers' button is at the bottom of the form.

When you add in a new pair, it gets stored in a database. So the *code* needed to build this program was pretty simple: HTML/CSS to build and style the web

page, and Python to add the data you're inputting into the form into a database on the backend. If you click on the *Manufacturer Name* dropdown, it shows you the available manufacturers; that's also the work of a Python script.



Where code “is”

Code “runs” on a computer, and gets written in a programming language. When you *run* a program, the computer executes those instructions. Most applications that you’re using – like Gmail – work by having their code constantly run in the background, displaying things and waiting for user instructions.

Writing code starts with a **programming language**. Languages are just different ways of expressing the same directions, but they’re all doing the same thing under the hood (give or take a few). The [most popular programming language](#) in the world is Javascript, followed by HTML/CSS and SQL. The nuances of how to accomplish a particular task in a language are usually referred to as **syntax**;

and because languages are so similar, getting basically proficient with a new one can be as simple as learning those details.

To get a better sense of what syntax means, let's take a look at one of the most common logic patterns in code: **a loop**. If you have a group of items (like a shopping list) and you want to look at all of the items in the group, you might loop through them:

```
for item in shopping_list:  
    print(item)
```

This would work in Python, but in Javascript you'd need to change it to:

```
for each (item in shopping_list):  
    console.log(item)
```

See? Pretty much the same pattern, but with slightly different wording.

Once you've written your code in your desired programming language, you've got to *run* that code. In practice, most apps you use will be a combination of different types of code in different languages that accomplish their own tasks: HTML/CSS for web page structure and styling, Javascript for interaction and animations in the browser, and another backend language (Python, Java, C++) for moving data around behind the scenes.

Deeper Look

Developers sometimes split languages into two broad types: **low level and high level**. Low level languages like Java and C++ are a bit harder to write, but offer more power and speed if you know what you're doing. These often require special infrastructure to run, and go through a translator called a *compiler*.

Deeper Look

Running a program can be really simple. If you've written a program in Python (`my_program.py`), you can run it by typing (`python my_program.py`) into the command line (Terminal) on your computer. In practice though, as apps have gotten bigger and more complicated, there are sophisticated ways to deploy and run your code that are a bit more involved than this.

Code frameworks and packages

Your app is unique, sure (except the guy who made *Yo – wtf man*), but most of them are actually pretty similar. Chances are that whatever you're building, you'll need to write directions for some of these:

- Formatting dates
- Making requests to a web server
- Generating cryptographic hashes
- Creating graphs and plots

As people built these, they started to share the code they wrote publicly for everyone else to benefit from (it's a really nice idea, in theory!). Those are called **frameworks** or **packages** or **libraries** or **modules** (there are more names but I'll spare you), and they're the basis of all modern code. There are tons of libraries out there for each of the 4 tasks we outlined above:

- *Formatting dates* – [Moment \(Javascript\)](#)
- *Making requests to a web server* – [Requests \(Python\)](#)
- *Generating cryptographic hashes* – [Cryptography \(Python\)](#)
- *Creating graphs and plots* – [D3 \(Javascript\)](#)

Using these modules means that instead of building the code to make a graph from scratch, you can use the code that someone else wrote to do it. Modern apps are *full* of these libraries, and that's most of what you should think about when you hear **open source**. Open source just means that the library code is public for everyone to see, and generally also implies some community around improving and maintaining that code.

Terms and concepts covered

Code

Programming language

Syntax

Framework, module, package, library

Open source

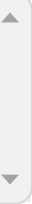
Further reading

- One of the best ways to understand what code is and how developers work is through data: the [2019 Stack Overflow Developer Survey](#) is a great place to start
- Some programming languages have very specific purposes: SQL, for example, is [exclusively for querying a database](#)
- Learning to code has been in vogue for the past few years, and there are more [online resources to get it done](#) than you could possibly ever cover in a lifetime

Comments



Write a comment...



© 2023 Justin · [Privacy](#) · [Terms](#) · [Collection notice](#)
[Substack](#) is the home for great writing