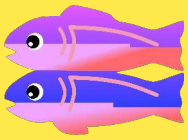
You Got This!

Zine

In this issue:

v1.0



Glitch is the friendly community where you'll build the app of your dreams.

With working example apps to remix, a code editor to modify them, instant hosting and deployment – anybody can build a web app on Glitch, for free.

Learn more at **glitch.com**

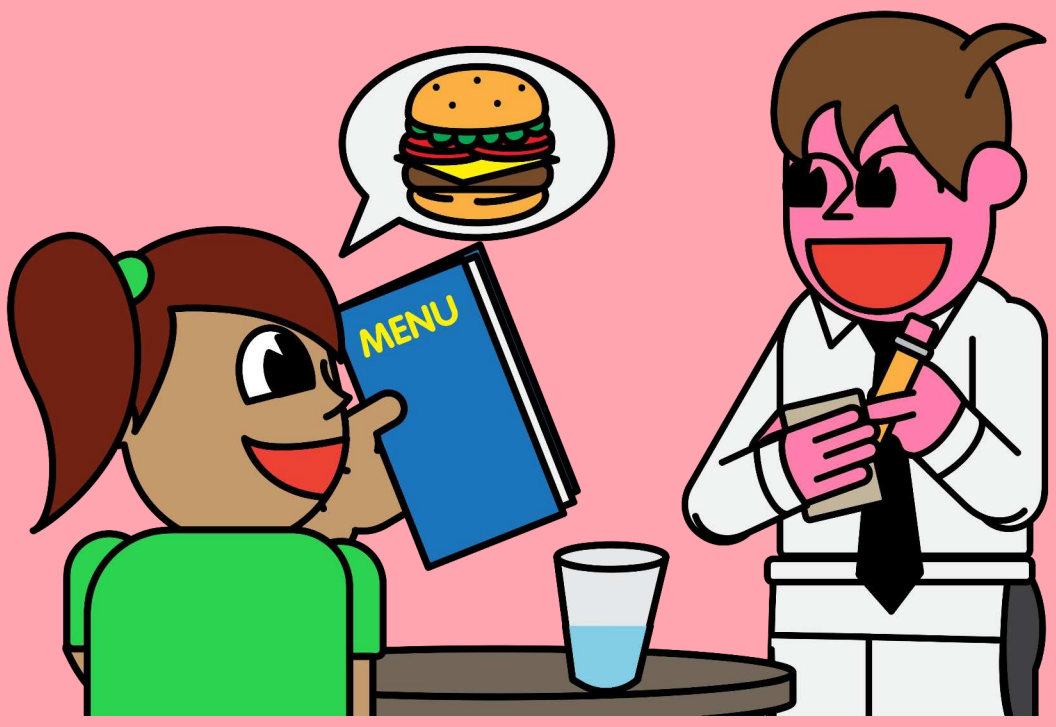
Contents

How Web Servers Work

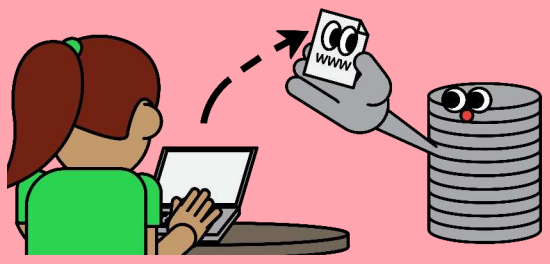
What happens when you enter a URL into your browser? It turns out, a whole bunch of things happen.

But one of them is you sending a request to a server (which is just a computer) somewhere, asking it to give you the app or site that relates to that URL.

You can think of a computer server as being a bit like a server in a restaurant - you request something and they should return with what you requested. At a diner, for example, if you ask them for food, they eventually return with that food.

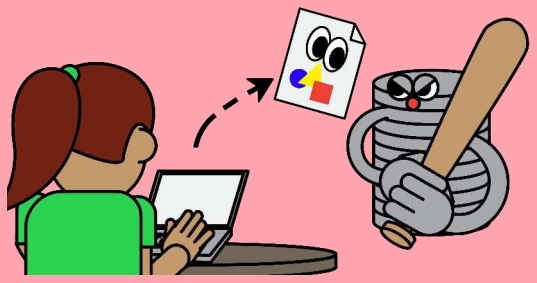


So when you're surfin' the web, and you go to example.com, or click a link, you're asking a server to send you a web page.

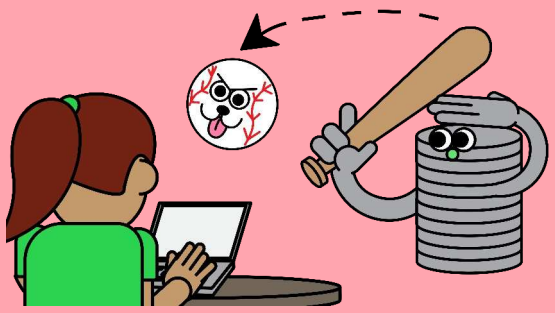


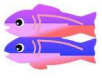
When someone enters the URL of your own app or site into their browser, or clicks on a link to it, a request to the server holding your app's files is returned.

If those files reference other files like images and .css files, then we'd need to ask the server to send us those too!



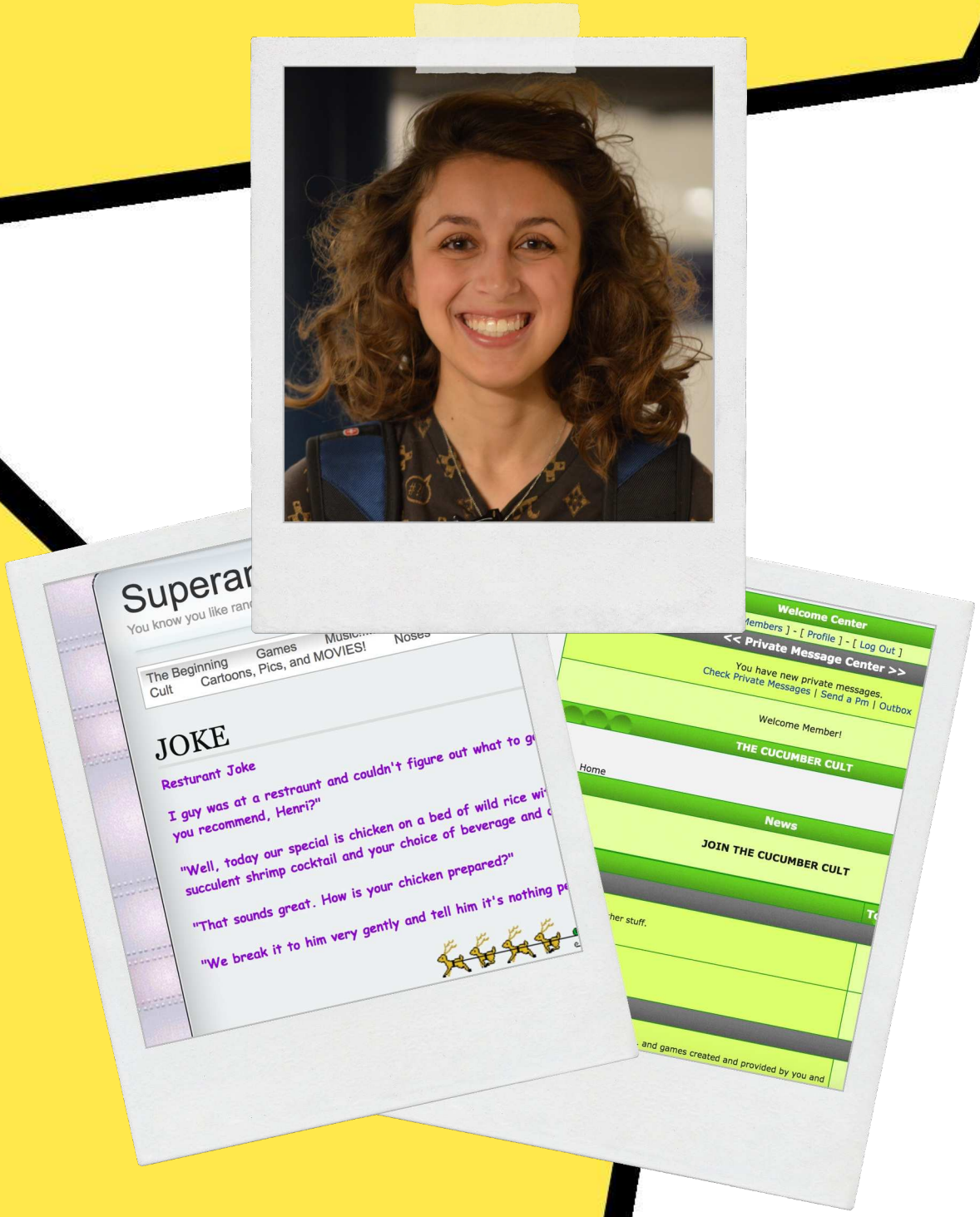
And the server will oblige. How does the server know which files to send? You have to tell it to by writing some server-side code.





glitch.com/~intro-to-node

The intro-to-node project explains more about servers and includes a website that uses a Node server.



Cassidy Williams

Cassidy is lead of Developer Voice Programs at Amazon!

She was featured in *Glamour* magazine's '35 Women Under 35 Who Are Changing the Tech Industry', and starred in the documentary *Big Dream*.

How did Cassidy get into coding?

When I was first learning, I didn't know what code was. Nobody in my family did, either.

> "I just tried things, and kept learning

But when I was in 8th grade, I heard a neighbor say, "check out my website!" while I was talking home from school. When I got home, I literally just started looking up how to make one.

I started with a WYSIWYG editor, and then I realized that if you knew how to code, you could do a *lot* more. So I started messing with it from there!

My first website ever was called *Superandomness*. It had animations all over the place, a different tiled background image on every page, and Comic Sans out the wazoo.

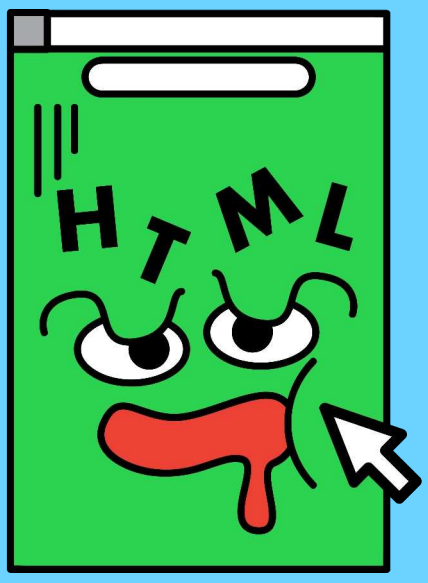
The next site I made was a forum called *The Cucumber Cult* (I don't remember why it was named that, but I think I liked the alliteration). This was how my friends and I communicated after school every single day.

My coding experience started to grow through my AP Computer Science class, to eventually majoring in Computer Science, to interning, to hackathoning, to working as a backend dev.

Most of the time (both then and now), I just tried things, and kept learning as I went. It hasn't been the smoothest learning path for sure, but I've absolutely loved it.

Anatomy of a Web App

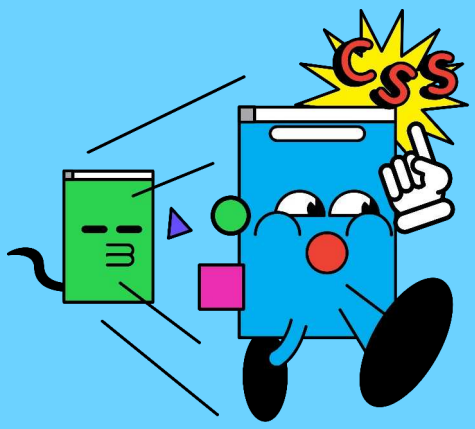
A typical web app consists of four main things: HTML, CSS, JavaScript and Server-side code, like Node.js.



1. HTML

HTML stands for HyperText Markup Language. It is a language for adding content to a web page or app. That content can be text, links, images, audio and video.

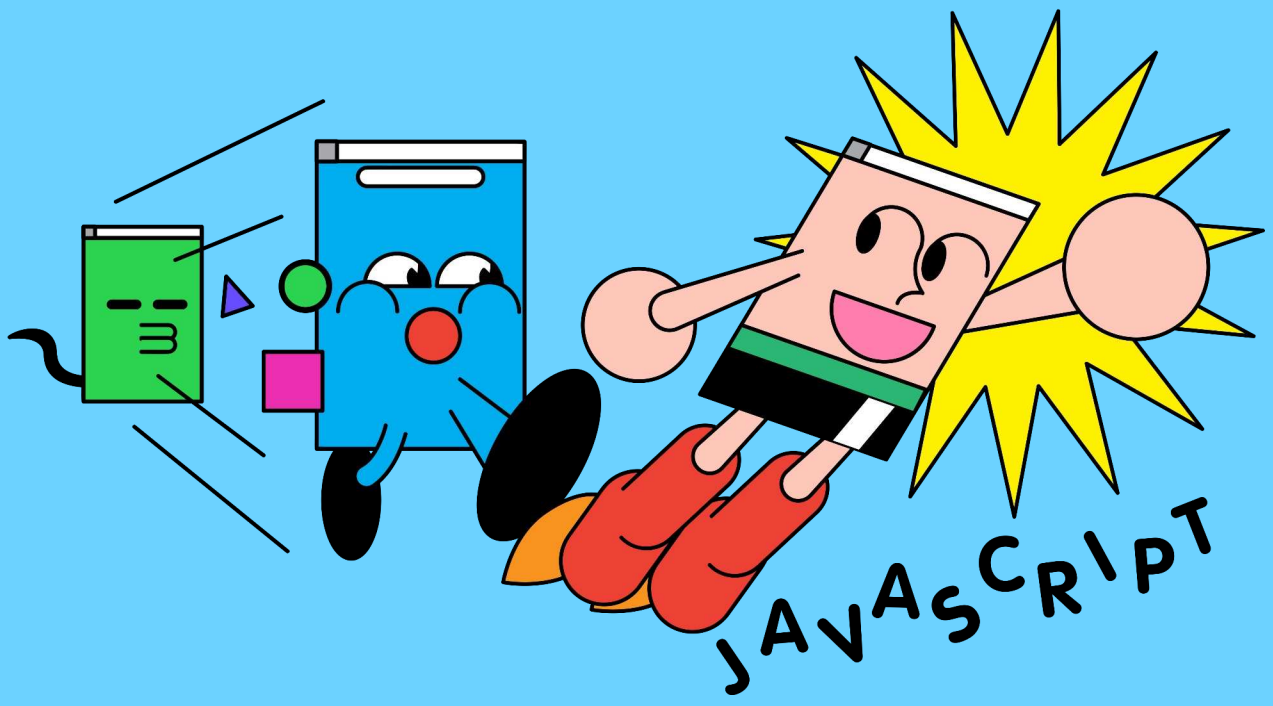
The ability to add content to a site that anyone on the Web can see is very powerful, but the result of HTML is not very pretty.



2. CSS

CSS stands for Cascading Style Sheets, which is a language for styling elements on a web page or app. It's what can make a website go from ugly to beautiful.

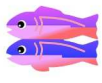
In CSS, we write sets of rules that can set the style of all the HTML elements on a web page.



3. JavaScript

With HTML and CSS your web pages are static, they just display the content you've given, styled however you've told it. To be able to update content or add some interactivity to your web page you need JavaScript.

The sky's the limit on what JavaScript can do, from updating the page when a user clicks something, to showing animations, playing videos, and other fun things.



glitch.com/website-starter-kit

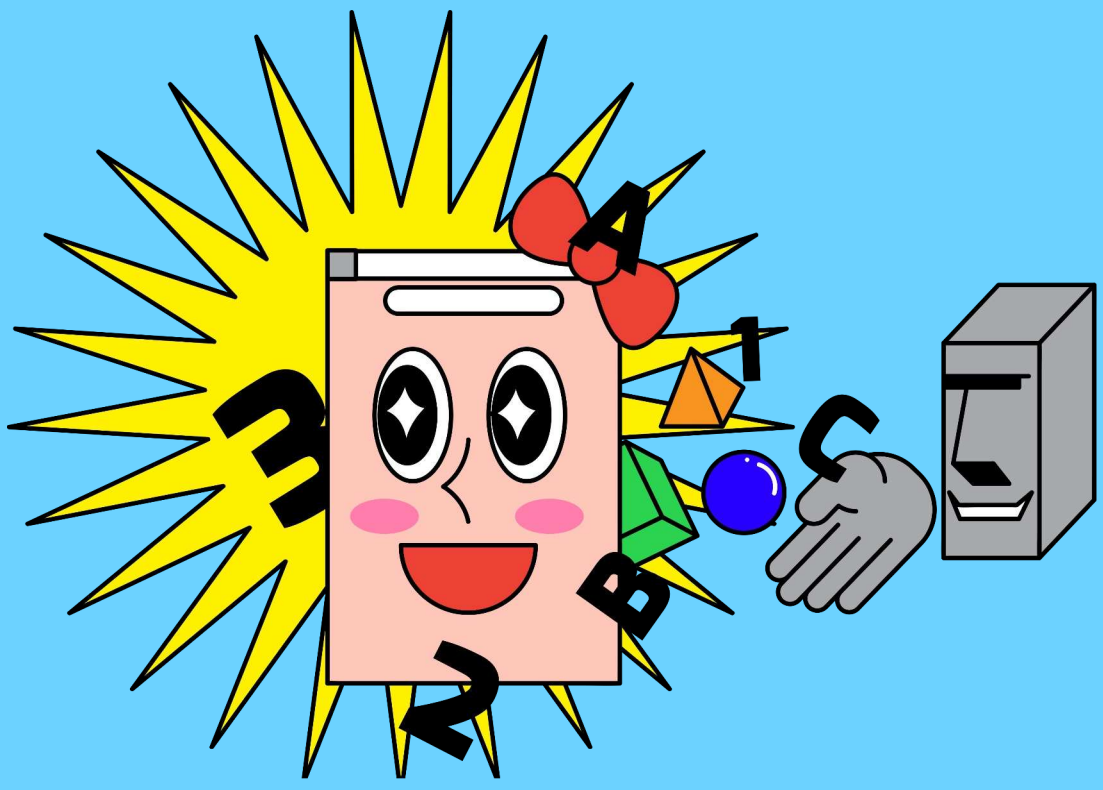
To learn more about HTML, CSS, JavaScript and Node, check out the Website Starter Kit. A free, 4-part video course with interactive code examples that guides you through building a website.

4. Server-side Code

The HTML, CSS and JavaScript we've discussed so far is what's known as 'client-side code'. This means it's code that is run on the end-user's computer. When they view your web page in a browser the client-side code is downloaded, run and displayed.

But for things like the handling of someone logging in to a website with a password, or getting information from a database to display on your web page, you need to use what's known as server-side code.

There are many different server-side programming languages, but you can just use JavaScript again as Node.js lets you use JavaScript on the server too.



Expert Advice for New Developers

"Don't try to learn without a purpose. The desire to learn a new skill is driven by curiosity. Try to build something, find a problem, and learn only to solve it."

Dayle Rees, Prev. Head of Engineering at JustPark, now at Crowdcube

"Be rigorous! When I was younger, I thought the only thing that mattered was having a working project. That attitude slowed me down because it kept me from understanding some things deeply."

Mary Rose Cook, Makers Academy

"Don't fragment your forces into too many pieces. Focus on the intersection of things that at the same time you find valuable, and many people find valuable."

Salvatore Sanfilippo, Creator of Redis

"Get involved with the developer community: contribute to an Open-Source project, answer questions on StackOverflow, join a local User Group, etc. Constantly engaging with other people will help push you to grow as a developer."

Jared Parsons, Principal Developer Lead on the C# Language Team at Microsoft

"Write code you don't know how to write, tackle problems you don't know how to solve. More importantly, learn things you don't want to learn. If it looks boring or if it looks too hard, study it."

Dusty Phillips, Software Engineer at Facebook

"Figure out what you're passionate about and do it. That might sound obvious, but focus and persistence are important for success, and both require passion."

Lindi Emoungu, Senior Software Engineer at Google

How did Saron get into coding?

> “It’s still a work in progress... but we’re making progress

"I got a Windows computer at around 9 years old. However, I didn’t really know what to do with it.

Later, I worked at a few tech startups but I felt frustrated at how little I was able to contribute as a non-technical person. So I quit my job and learned to code.

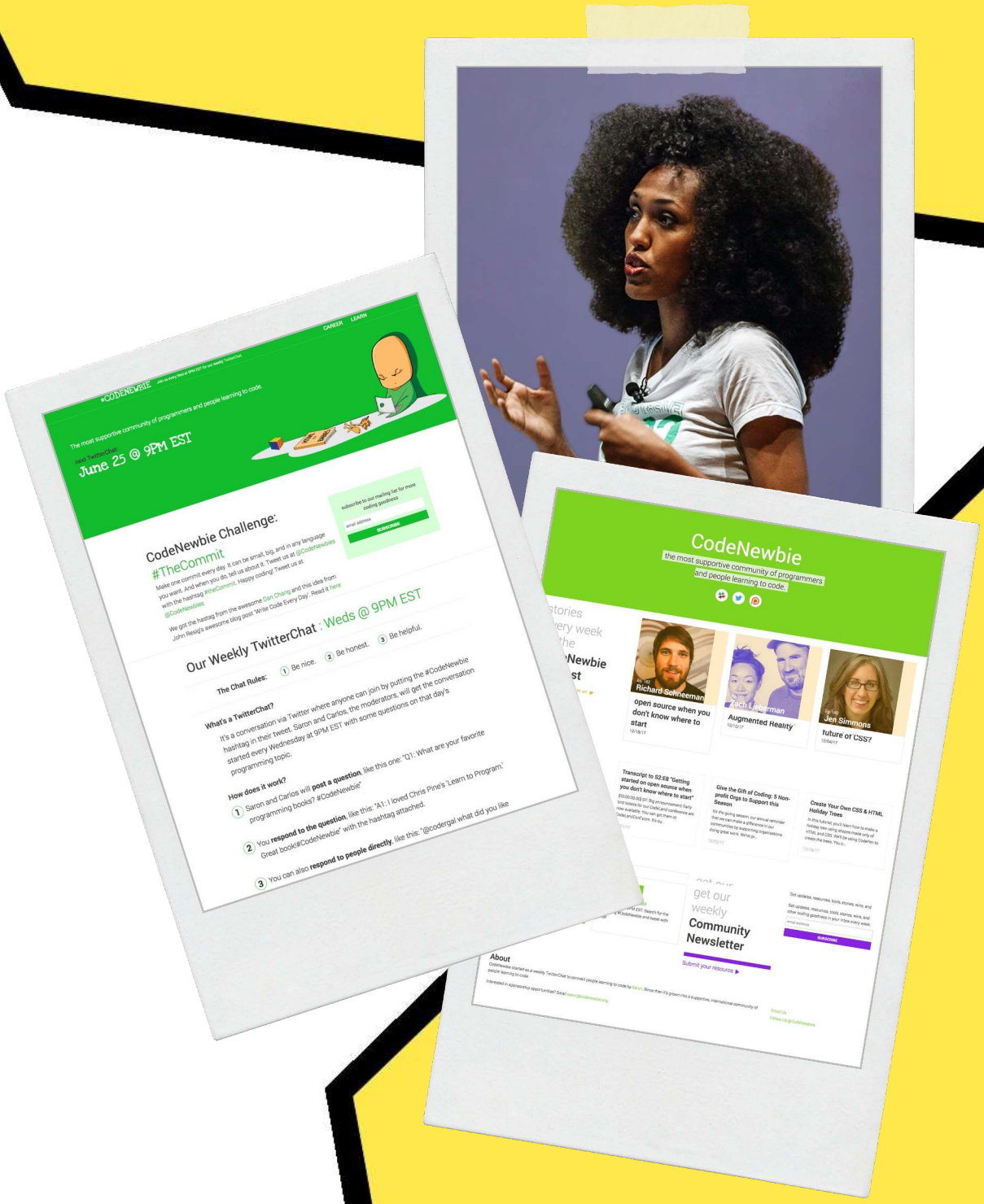
I was self-taught for three months, using different free and cheap online resources before doing a three-month bootcamp at The Flatiron School.

I went on to found *CodeNewbie* originally as a regular Twitter chat. We used the hashtag #CodeNewbie, and I would tweet out questions related to programming and learning to code. It grew from being a weekly chat to a weekly podcast, managing a team of bloggers, a Slack community, and an online forum."

Things are rarely perfect the first time. Keep working, and try things out.

Take Saron's CodeNewbie website for instance. At first, "it was all about the Twitter chat, because that’s all CodeNewbie was!" As they added new things, like a podcast and blog posts, "we had to find a way to show them, so the layout had to change. Adding a nav bar seemed like a good idea. I also wasn’t happy with the feed, so we reworked the layout."

The latest iteration, "let things breathe a bit more, and focused on the different projects we were doing. It’s still a work in progress - working on a better design now, but we’re making progress!"



Saron Yitbarek

Saron is the founder of the programming community, *CodeNewbie*!

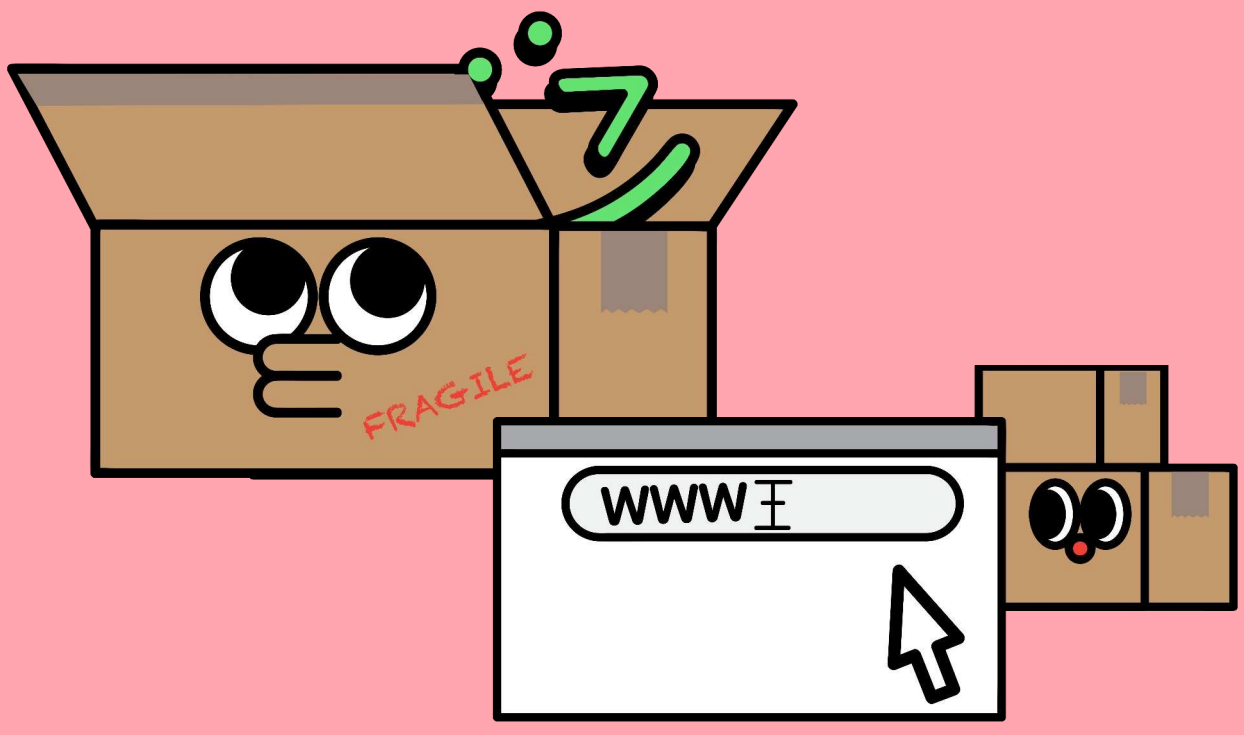
She previously worked as a developer at Thoughtbot and Microsoft.

What's npm exactly?

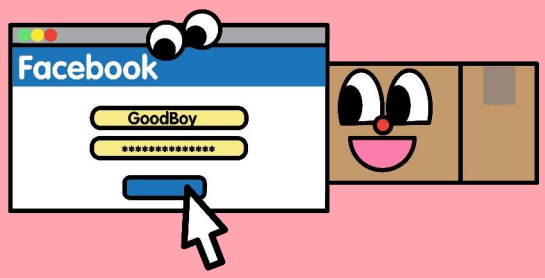
npm is a package manager for JavaScript. It makes it easy for JavaScript developers to reuse code other developers have shared.

Developers create small building blocks of code that solve one problem well and "package" it up.

You can then use these packages in your own applications so you don’t have to write the code for every part of an application yourself. A typical application or website will consist of dozens or hundreds of small packages of code.

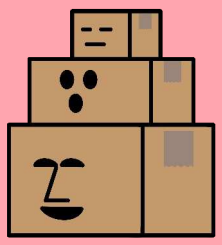


So if you want to add the ability to login to your website with Facebook, there's a package for that.



And if you want to grab data from a database to display on your website, there's a package for that too. In fact, npm consists of more than 600,000 different packages of code that you can use in your own applications.

Just like an application might consist of multiple packages of code, some packages themselves make use of other packages to implement the functionality they offer too.



When packages are updated with new features or fixes for bugs, npm makes it easy for you to update the packages you're using in your application, and for the packages you're using to update the packages they use too.

Steve Klabnik

Steve works at Mozilla where he leads the documentation team for the programming language, *Rust*!

He's a prolific open source contributor, previously working on projects such as Ruby and Ruby on Rails.

How did Steve get into coding?

"My family pooled some money together and got me one of those computers from the back of a Sears catalog that hooked up to your TV. It had a GW-BASIC interpreter on it, and I spent most of my time playing around with it. Eventually, I graduated to C, then C++, then Java.

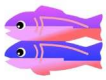
> "I turned to the 'Basic Error Messages' so many times the page ripped out

Later on, I went to college for CS. I dropped out to do a startup. Like most, it imploded, and I went back and finished."

Steve developed his skills by starting simple

"My uncle was a software developer, and when he brought a computer home to show his parents what he did, I happened to be present and was made aware of this wonderful machine and the possibilities it offered. After seeing *Adventure*, a text-based adventure game, I was hooked."

Steve went on to learn how to program by recreating apps created by others, typing them out from a computer programming manual. "I turned to the 'Basic Error Messages' so many times the page ripped out of the book."



glitch.com/games

There are games on Glitch, which you can remix and edit to understand how they work.

Understanding Async/Await in Node

Node.js processes requests in an asynchronous manner. It’s like ordering food in a fast food diner, rather than a fancy restaurant.

Say you want to be seated, place a food order and get some mayonnaise for your food as quickly as possible.

If you were to request those in a fancy restaurant, synchronously, you’d ask to be seated. They’d get you seated. Then you’d ask for the food you want and they’d bring it to you. And then you’d ask for mayo and they would get that for you too.

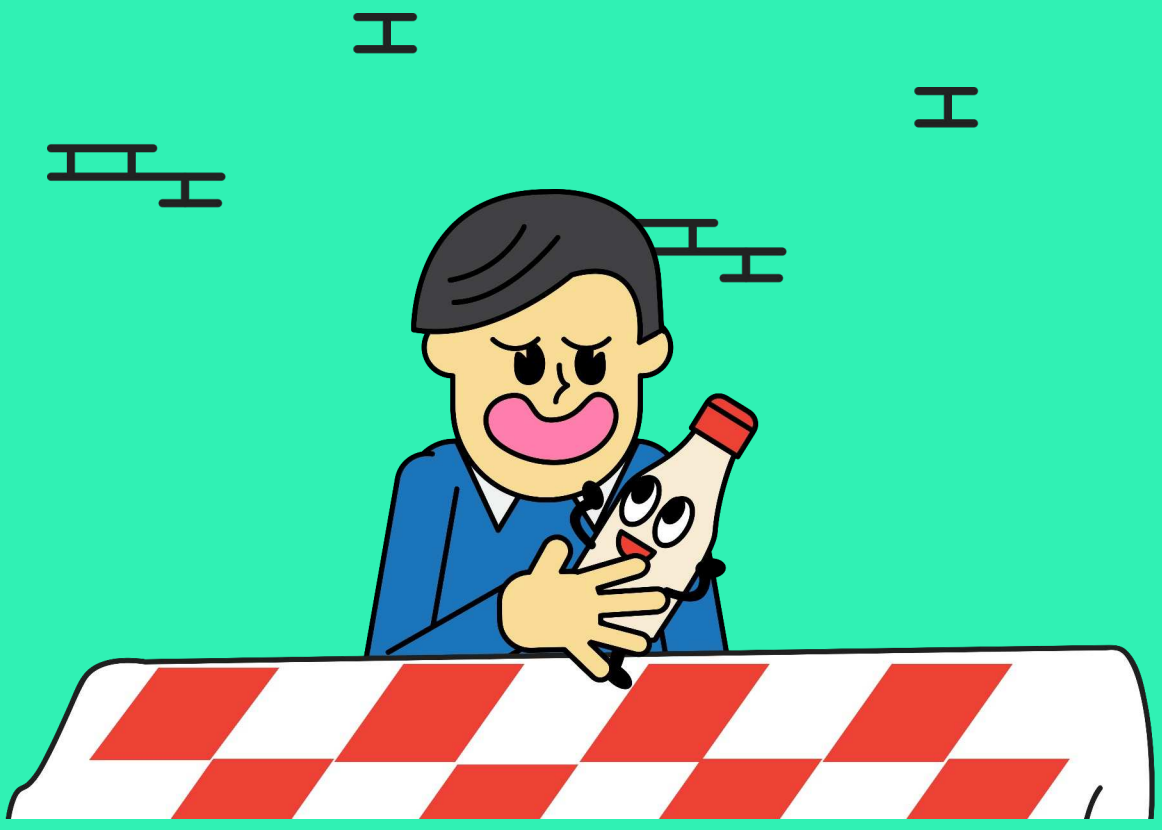
This is intuitive and predictable, but it can take quite a bit of time. What if it takes ages to be seated? Then you'd have to wait to place your food order *and* you’re unable to ask for mayo until you have food.

So Node.js favors asynchronous requests - it's like being at a fast food place. You can ask to be seated, place your food order and ask for mayo, and multiple servers will each go off to do those things.

You don't have to wait before asking for the next thing - each server carries out their task as soon as they can.

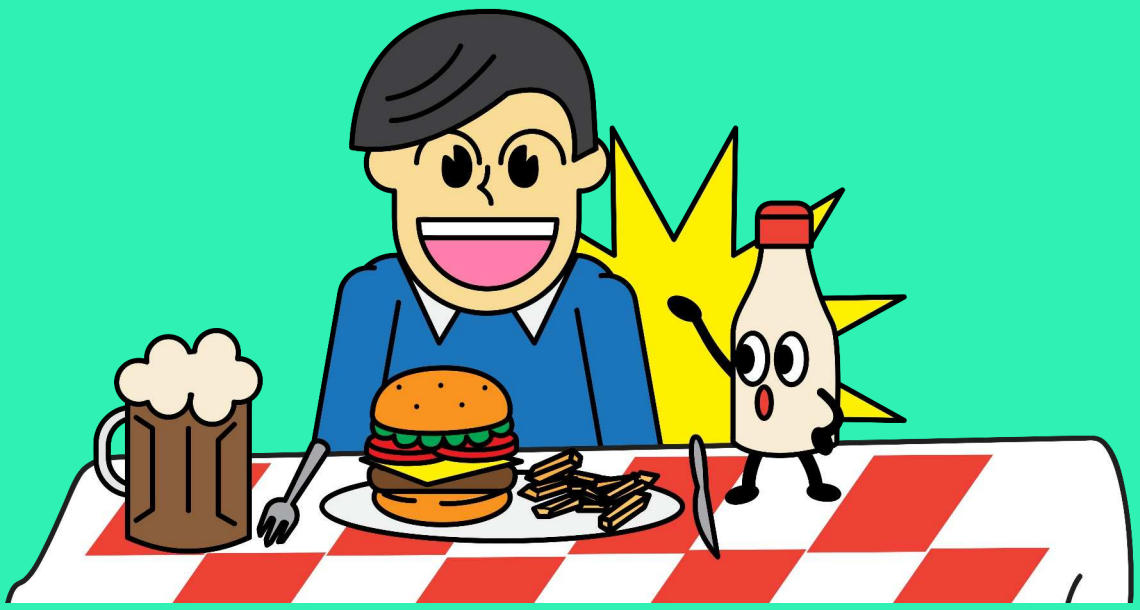
This is quicker, but it can be confusing because you don't know the order in which the tasks will be completed. It could be that the server getting the mayo returns before you've gotten your food. Or your food could arrive before you've even been seated.

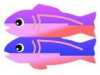
Awkward.



So Node.js 7.6 and up supports Async/Await, which gives you the predictability of synchronous requests with the efficiency of asynchronous. You can request to be seated, order your food and get mayo. They'll go and do all three at once, but you can specify that you want to be seated before your food arrives and you're given mayo.

When creating an app, this means your app can handle requests really quickly and the code you need to write is as simple as it can be.





glitch.com/~async-await-playground

The async-await-playground project allows you to experiment with code implementing Async/Await functionality.

Top Tips for New Coders

Coding is "easier when you do it with a community of people who are just as passionate about coding as you."

Saron Yitbarek, Founder of CodeNewbie

"Don't hesitate to ask people dumb questions. Your questions may not be so dumb after all."

Tomomi Imura, Developer Relations at Slack

"A lot of beginners under-appreciate the degree to which googling things is part of a programmer's job."

Steve Klabnik, Docs team lead on Rust at Mozilla

"Consuming information is one thing, but if you don't build things you won't remember it."

Brian Bondy, Co-Founder at Brave, prev. Senior Engineer at Khan Academy and Mozilla

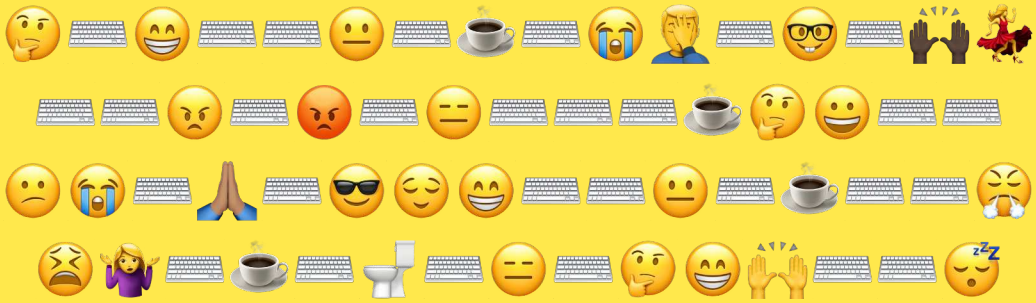
"Don't waste your time on technologies you don't enjoy. It's much more fun to code on things that you want to!"

Cassidy Williams, Developer Voice Programs Lead at Amazon

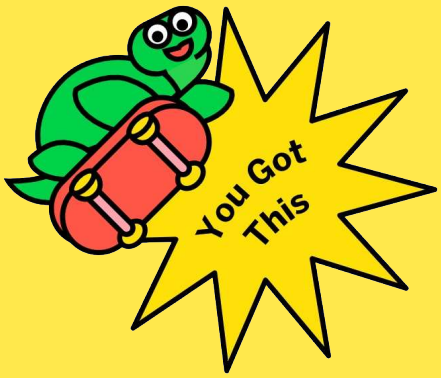
"Good developers will quickly reach a point where just writing correct code is not enough; they've got to teach others how to use it. Communication is key."

Eric Lippert, Software Engineer at Facebook and author of 'Essential C#'

Coding be like:



It's a 🎢. Hang in there.



Credits

Illustration: Good Boy Graphics

Layout: Gareth Wilson

Content: Jenn Schiffer, Gareth Wilson

Edited by: Maurice Cherry

Guest Contributors: Cassidy Williams, Saron Yitbarek, Steve Klabnik