Defining Front-End Web Programming

(https://github.com/learn-co-curriculum/phase-1-defining-frontend-web-programming) (https://github.com/learn-co-curriculum/phase-1-defining-frontend-web-programming/issues/new)

Learning Goals

- Define web programming
- Contrast web pages versus web applications
- Identify a reference example: "Liking on social media"
- Identify the "Three Pillars of Web Programming"

Introduction

The phrase "Front-End Web Programming" is used in different ways by different people. In this lesson, we're going to choose a definition of "web programming" and show an example that demonstrates Flatiron's "three pillars" of web programming. We will explore each of those pillars in detail in upcoming lessons.

What Exactly Does "Web Programming" Mean Anyway?

Web programming, at its heart, is:

- Creating documents with HTML and styling/positioning the documents' content with CSS
- Using JavaScript to provide interactivity
- Using JavaScript to notify remote servers

Web Page vs. Web Application

When a web page has a lot of JavaScript code, the page feels closer to a computer application, so some people may call it a "**web application**." However, it's worth noting that there is ***no clear distinction*** between a "web page" and a "web application." For instance, there's no rule like

"When there are three or more actions it's a web application!" Different people draw the boundary differently. More or less, we call a web page an "application" when it feels "rich."

Identify a Reference Example: Liking on Social Media

As we move through the rest of this material, we're going to use *one, tiny* interaction as our shared or "reference" example.

> **Web Programming Example**: "Favoriting" a social media post.

Regardless of the social media site (Instagram, Pinterest, Facebook, LinkedIn, Twitter), the interaction goes something like this:

Step 1: The site renders some HTML content that is styled using CSS

Step 2: You see the content and decide to show your approval of it

Step 3: You *click* some visual element meant to show approval (heart, thumbs-up, +1, etc.). For example:

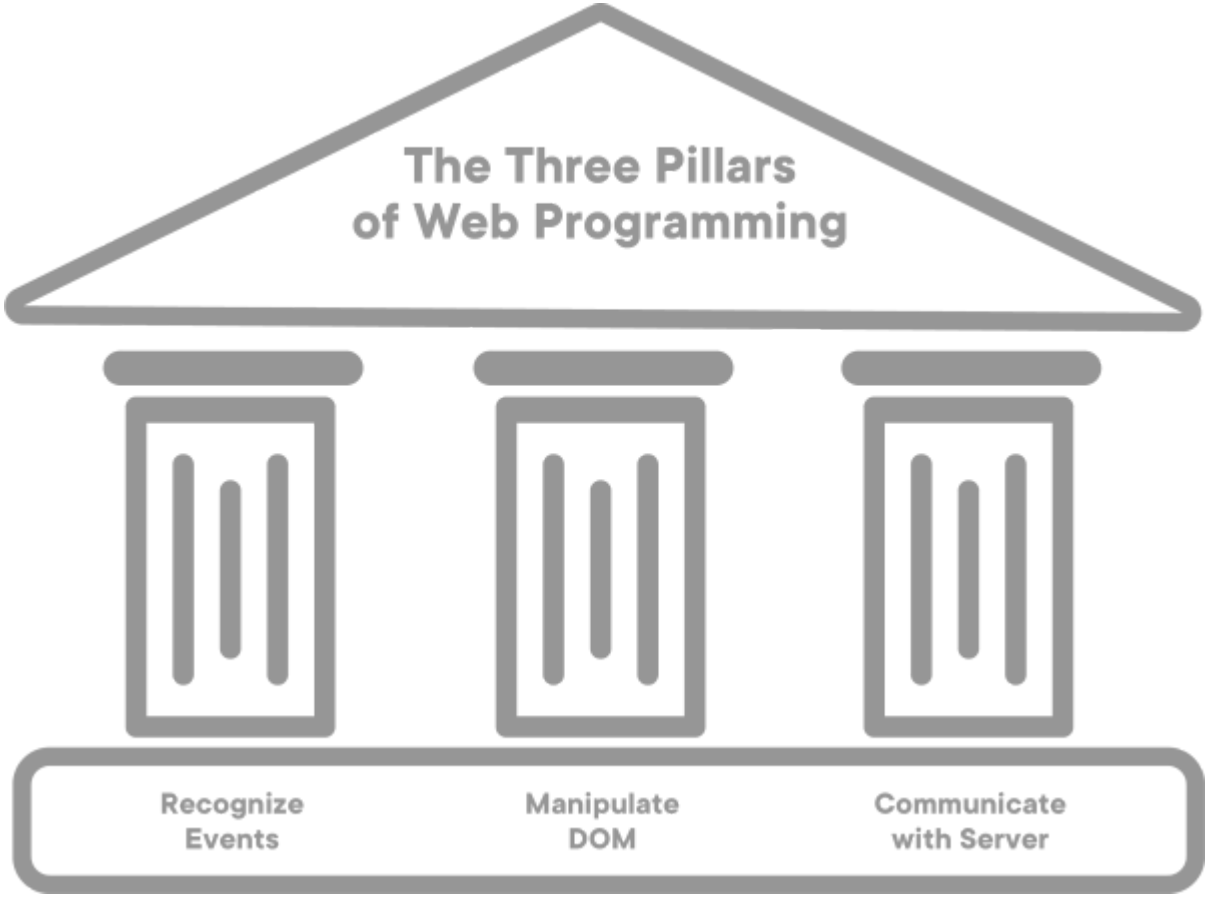
Step 4: The visual element *updates* (animates, goes from empty to full, jiggles, etc) like:

Step 5: Behind the scenes, the application *tells the provider* that this post has gained your approval so that the central provider can store this information and use it later (for example, to notify the post author that you liked their post).

If all goes as it should, the entire interaction only takes a second or two. But even this small interaction demonstrates all the concepts of front-end web programming.

Flatiron's "Three Pillars of Web Programming"

To help us learn web programming in three distinct phases, this curriculum is broken down into three essential "pillars":



- **Recognize Events**
- **Manipulate the DOM**
- **Communicate with the Server**

Let's take another look at steps 3 through 5 of our Favoriting a Social Media Post example. Note that we *italicized* a verb in each step. Those words exemplify the activity of one of the "pillars" we must learn in order to make web applications.

- Step 3 showed **Recognizing JS events**: Your *click* action on the empty heart tells JavaScript to do work
- Step 4 showed **Manipulating the DOM**: the work JavaScript was told to do was to *update* the screen to make the heart "look clicked"

- Step 5 showed **Communicating with the server**: the work JavaScript was told to do was to *tell the social media company* that you approved of this content

> **THINKING ABOUT LEARNING:** These "pillars" are not something professional developers outside Flatiron School recognize. You won't go into a tech interview and be asked: "Name the three pillars of web programming." These abstractions are a way to help learners (you!) recognize how what you're learning fits into three major activity areas. That said, if an interviewer asked you how to debug a web program and you said your strategy would be based on ensuring three "critical areas" were working, we think your interview answer would be off to a good start!

Now you know what's going on when you click that heart! The next lessons will focus on explaining each of these "pillars" in more detail. After you've worked your way through them, your new "web programmer" eyes will have you looking at your favorite sites very differently.

Conclusion

Web Programming consists of creating documents with HTML, styling/positioning the documents' content with CSS, and using JavaScript to respond to events. We can break down the JavaScript part of web programming into three "pillars": recognizing events, updating the DOM, and informing the server. Now that we've seen how these pillars are connected in the abstract, we are ready to dive into seeing how they work together in detail.