Learn to Code: For the Animals

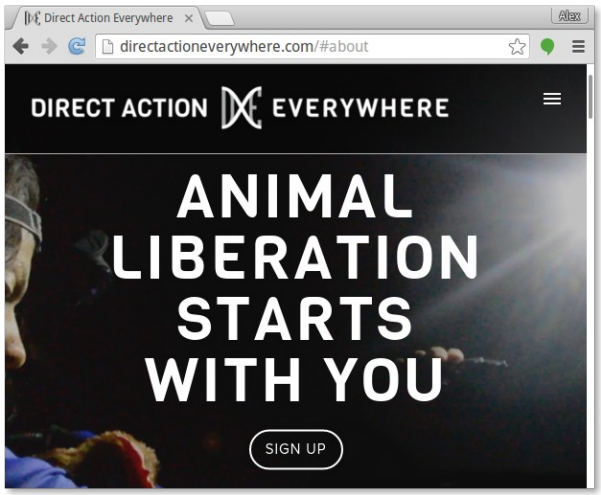
Web Basics

presented by

DIRECT ACTION EVERYWHERE
PHILADELPHIA

Websites are made up of **HTML**, **CSS**, and **JavaScript** code. There can be more to it than that, but for now we're going to focus on **front-end** code. HTML, CSS, and JavaScript are front-end code. The front-end is the part of the website you can see. It's code that a web browser can directly understand.

Pictured: the front-end of directactioneverywhere.com



Websites can also have **back-end** code. Back-end code can be used to generate front-end code dynamically. While front-end code is limited in its capability, the capability of back-end code is virtually limitless. Any programming language can be used to write back-end code, such as C++, Python, Ruby, C, Lisp, Fortran, and even Brainfuck. More on that later.

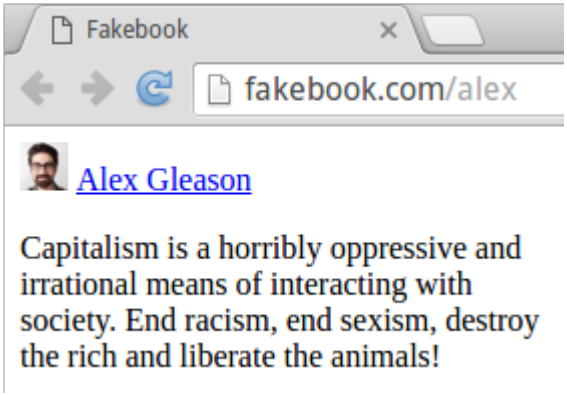
Front-end Technology

HTML

Hypertext markup language. HTML is used to create the layout of a web page. It determines the order that elements appear in, the nesting of elements, and the names you assign to elements.

Pictured: some HTML sample code

```
<div class="facebook-post">
  <img class="profile-pic-thumb"
  src="alex.jpg">
  <a class="name"
  href="/profile/alex">Alex Gleason</a>
  <div class="post">
    <p>Capitalism is a horribly oppressive
    and irrational means of interacting
    with society. End racism, end sexism,
    destroy the rich and liberate the
    animals!</p>
  </div>
</div>
```



CSS

Cascading style sheet. CSS is used to produce the style of a web page. It determines the colors of things, sizes of shapes and text, position of elements, and more.

```
body {
  background: blue;
  color: white;
  font-family: "Open Sans";
}
a { color: yellow; }
```

JavaScript

JavaScript is a programming language that allows you to make things happen on the page programatically. It's often used to produce things like image sliders and other interactive elements. JavaScript is a full fledged programming language which can also be used outside of the context of a website to build desktop software, system scripts, and more.

Part 1: HTML

HTML comes in the form of **tags**. Tags make up page **elements**. For instance, you'd use the `p` tag to create a paragraph. “p” stands for paragraph. Like so:

```
<p>This is a paragraph!</p>
<p>This is another paragraph.</p>
```

There is an **opening tag** and a **closing tag**. The closing tag will always be the same as the opening tag, except there will be a `/` in it. Most HTML elements have closing tags, but some of them like `<br>` (line break) exist by themselves. In between tags, there will be some text, or some other tags.

```
<p>Want me to tell you something <strong>awesome</strong>?</p>
```

Above is an example of nested tags. The `strong` tag pair is nested within the `p` tags. `strong` makes text appear bold by default.

Common tags

`<h1>Header 1</h1>`
Used to make top level headers. This is usually the title of the page.

`<h2>Header 2...</h2>`
The trend continues from `h1` to `h6`.

`<div>Divider</div>`
A generic divider element. Use this to divide up your page layout into distinct parts.

`<a>Anchor</a>`
There are used to make links and define parts of the page that can be skipped to.

`<p>Paragraph</p>`
Make paragraphs of text.

Strong
Represents important text which is bold by default.

Emphasis
Represents emphasized text which is italic by default.

Attributes

Attributes let you add additional information to tags. Some tags require an attribute in order to be used for their intended purpose.

For example:

```
<div class="post">
```

In this case, `class="post"` is an attribute. `class` is the **attribute name**, and `post` is the **attribute value**.

Multiple Attributes

Multiple attributes are allowed. They're separated by a space. For instance:

```
<a class="special-link" href="http://mysite.com/">
```

In this case, `class="special-link"` and `href="http://mysite.com/"` are both attributes.

Images

```
<img src="picture.jpg">
```

Links

```
<a href="http://github.com">GitHub</a>
```

Basic HTML page structure

This is the most basic valid HTML page structure for a web page.

```
<!DOCTYPE html>
<html>
     <head>
          <title>The Page Title</title>
     </head>
     <body>
          <!-- Your code here -->
     </body>
</html>
```

Part 2: CSS

CSS targets HTML elements using **selectors**. A selector can find and search for one or more HTML elements based on their tag, ID attribute value, class attribute value, nesting within other elements, and more. For instance:

```
p { color: red; }
```

The selector is the `p`. It selects all `<p>` elements and sets their text color to red.

```
.post { color: green; }
```

This selector finds elements with a `class="post"` attribute and sets their text color to green. `class` attributes are used to set a name on elements; you can call it anything you want. class attributes are leveraged by CSS and JavaScript to find certain elements by the class name you choose.

```
.post p a.special-link { color: blue; }
```

This selector will find `<a>` tags which have a `class="special-link"` attribute. Those `<a>` tags must be the child of a `<p>` element. That `<p>` element must be the child of an element with a `class="post"` attribute. The selected elements will have their color set to blue.

Properties

Once you select some elements with CSS, you may style them using **properties**. In the examples above, the properties are the parts that look like `color: red;`

A group of properties are surrounded by curly braces `{}`. **Property names** have a colon `:` at the end, and **property values** have a semi-colon `;` at the end. Multiple properties may appear between the braces, and are usually separated onto their own lines.

```
a.button {
     padding: 15px;
     background: red;
     color: white;
     border-radius: 4px;
}
```

Common properties

`color`
Sets the color of text.
values: `red`, `blue`, etc... `#ffffff`, `#000000`, etc.

`background-color`
Sets the background color of an element.
values: `red`, `blue`, etc... `#ffffff`, `#000000`, etc.

`font-size`
Font size of text.
values: `24px`, `50%`, `2em`, etc.

`font-weight`
The thickness of text.
values: `bold`, `normal`, `300`, `700`, etc.

`font-family`
Name of the font you want to use.
values: `Arial`, `Helvetica`, `sans`, `sans-serif`, etc.

`text-align`
Aligns text on the screen.
values: `left`, `right`, `center`

`margin`
Adds spacing around an element.
values: `10px`, `20px`, etc.

`padding`
Adds spacing inside an element.
values: `10px`, `20px`, etc.