PROGRAMMING FUNDAMENTALS

LESSON #2

PRESENTED BY

ANDREW BUNTINE

WHO AM I, ANYWAY?

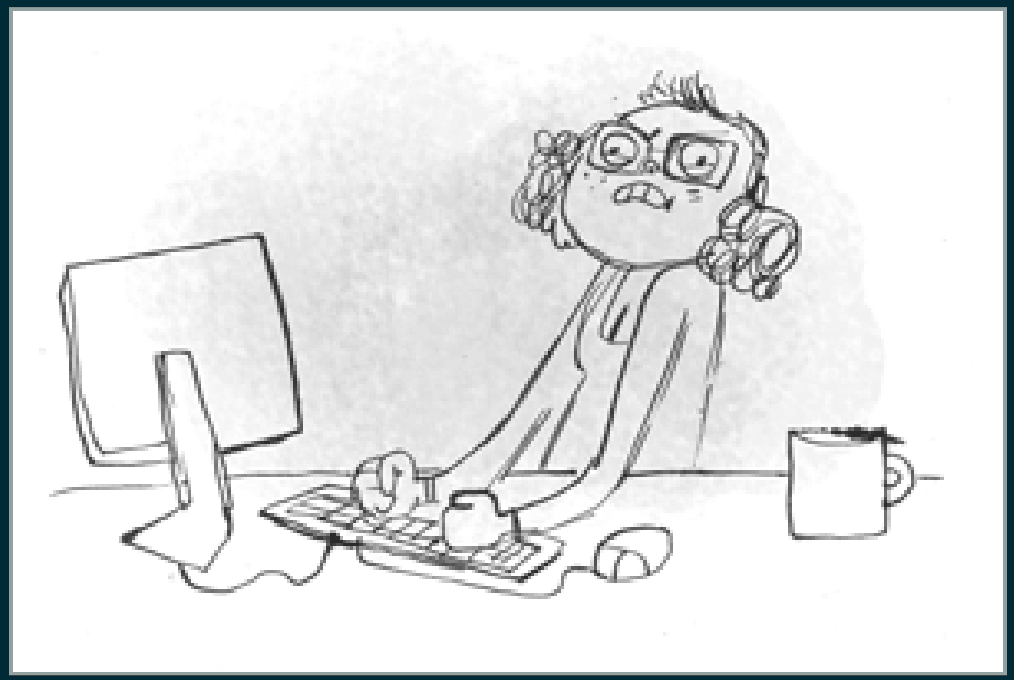
- Technical Director at Hardhat
- 15 years experience
- Web developer
- Indie game developer
- bunts.io
- github.com/buntine

WHAT IS THE GOAL?

DEMYSTIFY THE LINGO

CREATE BASIC WEBSITES

SURVEY OF TECHNOLOGIES



WHY ARE YOU HERE?

1. INTRODUCTION

- Vocabulary
- Development process
- Basics of the Web
- Coding introduction

2. FRONT END

- HTML
- CSS
- Javascript

3. BACK END

- Ruby
- Rails
- Make a web app

4. BECOME A GOD

- Put it all together

GOALS FOR TODAY

- Recap of lesson #1
- Setup your development environment
- HTML
- Hacking
- CSS
- More hacking
- Javascript
- Image formats
- Overview of some tools

RECAP

WHAT IS PROGRAMMING?

PROGRAMMING

The process of writing instructions that can be understood by a computer.

PROGRAMMING LANGUAGE

An intermediary language which can be understood by both computers and by Human beings.

WHAT IS WEB DEVELOPMENT?

WEB DEVELOPMENT

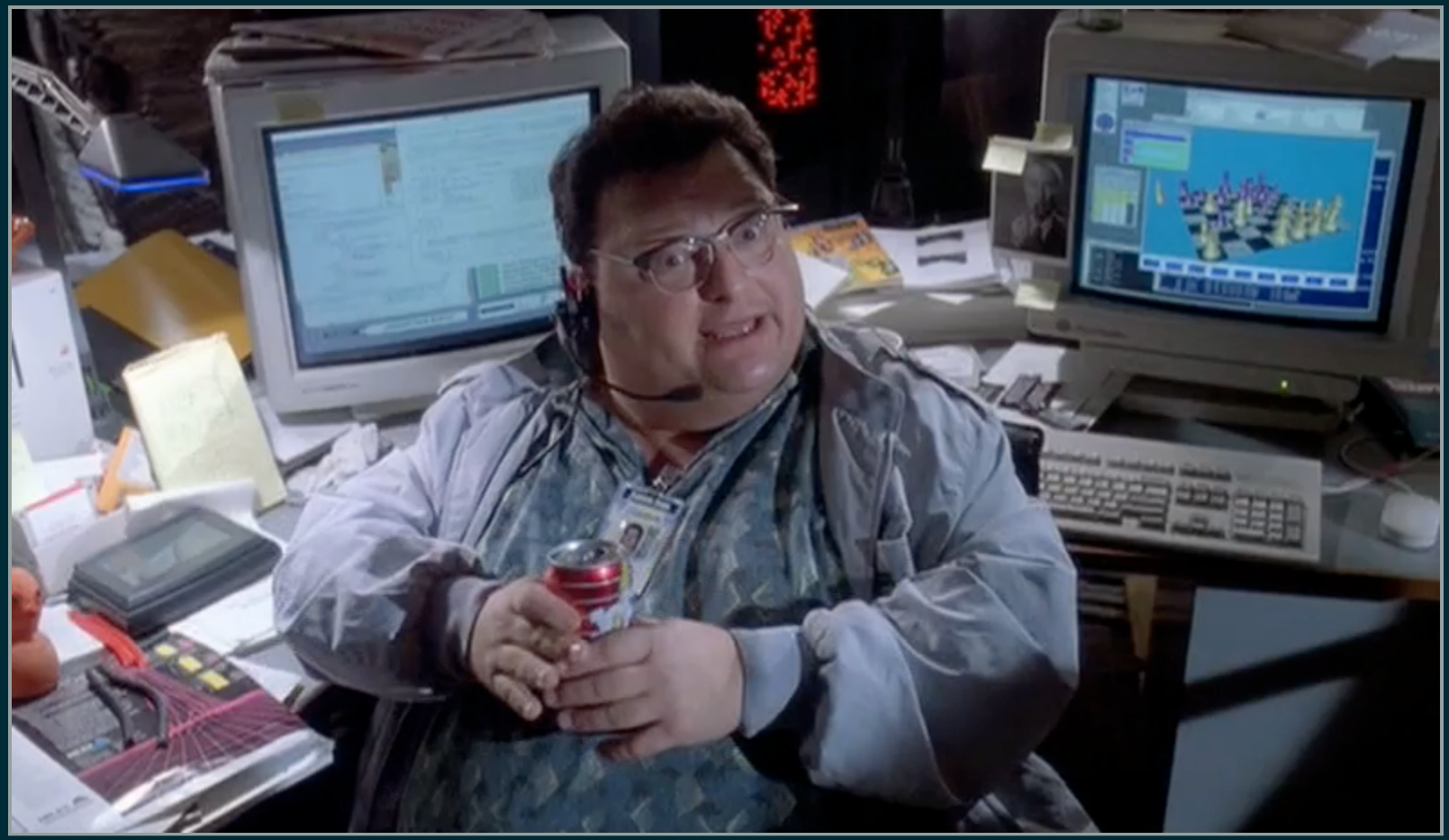
- The technical process of creating software that runs on the Web
- It's a broad term that covers several separate jobs!
- Typically, all of the non-design aspects of creating web applications

FRONT-END WEB DEVELOPMENT

- Translate designs into web pages
- Work primarily in the Web Browser
- Main tools are (typically) HTML, CSS and Javascript

BACK-END WEB DEVELOPMENT

- Create software that provides Website functionality
- Work primarily on the server
- Main tool vary greatly

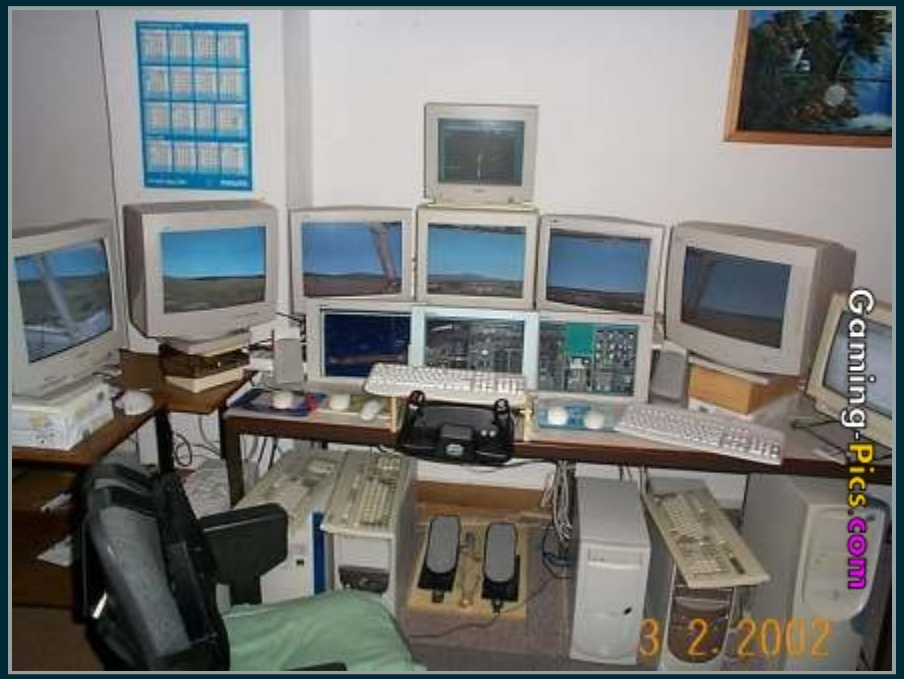


WE HAD A QUICK LOOK AT

- HTML
- CSS
- Javascript
- Ruby
- Version control (Git)
- APIs

OK, MOVING ON...

HACK TIME!



FOLDER SETUP

- Create a folder called "my_website"
- Inside that, create "images" and "css" folders
- Open "my_website" in Sublime (or your editor of choice)

HELLO, WORLD!

- Create a new file
- Type this:

```
<!DOCTYPE html>
<html>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

- Save it as "index.html"
- Open it in your Web Browser

1978

The first "Hello, World" program is published in the seminal textbook "The C Programming Language"

HTML

WHAT IS IT?

- Hypertext Markup Language
- Defines Webpage structure
- Represents a "tree" data structure
- Parsed by the Web Browser and turned into visual elements
- Not a "programming" language:
  - No control flow
  - No internal state
  - No arithmatic or logic

WHAT DOES IT LOOK LIKE?

```
<!DOCTYPE html>
<html>
  <body>
    <h1>This is a heading</h1>
    <div>
      <img src="/some/image.jpg" alt="Alternate text" />
      <ul>
        <li>List item goes here</li>
        <li class="special">Another list item</li>
      </ul>
    </div>
  </body>
</html>
```

WTF?

DOCTYPES

- Tell the Web Browser what type of document it is receiving
- HTML4 and XHTML have several, HTML5 only has one
- It used to be optional, but now HTML5 requires it!

HTML ELEMENT

- The <html> element is the root node of a document.
- Remember, an HTML document is represented as a tree.
- <html> is the trunk!

HEAD ELEMENT

- Generally the first child element of <html> in a document.
- <head> houses the page title, meta elements, scripts, stylesheets, etc.
- It does not display to the user, so don't put images, headings, links in there.

BODY ELEMENT

- Generally the second child element of <html> in a document.
- <body> houses the "renderable" document content.
- **Do** put images, links and headings in here!

BUT WAIT, THERE'S MORE!

- Over 100 tags/elements in HTML5.
- And most have rules about where they go!
- You can usually get away with remembering a smaller subset.

STRUCTURE OF AN ELEMENT

```
<element attr1="value1" attr2="value"> ... </element>
```

SOME EXAMPLES!

HEADINGS

```
<h1>Biggest heading</h1>
<h2>Bigger heading</h2>
<h3>Big heading</h3>
<h4>Small heading</h4>
<h5>Smaller heading</h5>
<h6>Smallest heading</h6>
```

PARAGRAPHS

```
<p>This is a paragraph.</p>
<p>And so it this!</p>
```

LINKS / ANCHORS

```
<a href="http://www.google.com/">Go to Google.com</a>
<a href="/cheese.html">My Cheese webpage</a>
```

IMAGES

```
<img src="/some/path/to/image.png" alt="Alternate text here">
```

UNORDERED LISTS

```
<ul>
  <li>This is a dot point</li>
  <li>So is this</li>
  <li>
    <p>I like cheese</p>
    <img src="/my/cheese.png" alt="Blue cheese">
  </li>
  <li>Last but not least</li>
</ul>
```

FORMS

```
<form method="post" action="/some/page">
  <ul>
    <li>
      <label for="name">Your name:</label>
      <input type="text" id="name">
    </li>
    <li>
      <label for="email">Your email:</label>
      <input type="text" id="email">
    </li>
  </ul>
</form>
```

CSS

WHAT IS IT?

- Cascading Style Sheets
- Applies styling rules to webpages (colours, fonts, borders, etc)
- Separates document content from document presentation
  - Allows us to abstract styling out of document, so large changes are easier
- CSS rules are weighted so as to "cascade" through a document
- All browsers have default stylesheets that we can overwrite

WHAT DOES IT LOOK LIKE?

```
selector:pseudo, #another {
  property: value;
  another: value;
}
```

WTF?

SELECTORS

- A simple syntax to "target" one or more document elements.
- We can by very specific or very general.
- Multiple selectors can be separated by commas.

PROPERTIES

- Signify the name of a property we want to set.
- Naming conventions are kind of bad.
- Some browsers use differing names for the same property.

VALUES

- Signify values for a designated property.
- The range of valid values is dictated by the property.

PROPERTY / VALUE EXAMPLES

POSITIONING

```
img {
  position: static; /* Default */
  position: relative; /* Relative to static position */
  position: absolute; /* Exact positioning */
  position: fixed; /* Ignore scrolling */
}
```

FLOATING

```
img {
  float: none; /* Default */
  float: left; /* Float to the left of text */
  float: right; /* Float of the right of text */
}
```

COLOURS

```
div {
  color: #ff00ff;
  color: #fff;
  color: rgb(255, 0, 100);
  color: rgba(124, 35, 99.6);
  color: green;
}
```

FONTS

```
a {
  font: bold 15px "Sans-Serif";

  font-weight: bold;
  font-size: 15px;
  font-face: Sans-Serif;
  color: green;
}
```

BACKGROUNDS

```
div {
  background: red url('/some/bg.png') no-repeat;

  background-color: red;
  background-image: url('/some/bg.png');
  background-repeat: no-repeat;
}
```

SELECTOR EXAMPLES

UNIVERSAL SELECTOR

```
* {
  text-align: center;
}
```

ELEMENT SELECTOR

```
p {
  text-align: center;
}

h3 {
  color: red;
}
```

DESCENDENT SELECTOR

```
ul li img {
  border: 2px solid red;
}

p img {
  float: left;
  border: 4px solid #ffffff;
}
```

CLASS SELECTOR

```
.intro {
  color: red;
}

.serious {
  font-style: italic;
}
```

```
<p class="intro">
  This is <span class="serious">very serious</span>. Good bye!
</p>
```

ID SELECTOR

```
#footer {
  text-decoration: underline;
  color: green;
}

#footer img {
  border: 1px solid green;
}
```

```
<div id="footer">
  <p>This is green text with an underline</p>
  <img src="/image.png" alt="This image has a border">
</div>
```

PSEUDO SELECTOR

```
a:hover {
  color: #cccbbb;
}

div#header a.home:hover {
  color: blue;
}
```

```
<a href="/contact">This is #cccbbb on hover</a>
<div id="header">
  <a class="nav" href="/">This is blue on hover</a>
</div>
```

JAVASCRIPT

WHAT IS IT?

- General-purpose programming language
- Developed by Brenden Eich at Netscape in 1995
- Implemented in all major browsers, with slight differences
- Used to control behaviour and interact with the user
- Standardised under the name ECMAScript

WHAT DOES IT LOOK LIKE?

```
var hello_world,
    hello = function(name) {
      return "Hello, " + name;
    }

hello_world = hello("World!");

console.log(hello_world);
```

WHAT MAKES IT A PROGRAMMING LANGUAGE?

- It's **"Turing Complete"**!
- It provides constructs that we can use to create **algorithms**

POPULAR LIBRARIES

- JQuery (DOM wrapper, AJAX, etc)
- Three.js (Graphics)
- AngularJS (Frontend framework)
- Ember.js (Frontend framework)
- React.js (UI framework)

JQUERY

- Simplifies Javascript programming significantly
- Abstracts away much of the cross-device messiness
- Animations become very simple (no trigonometry requiired!)
- AJAX requests become a lot simpler to fire and handle

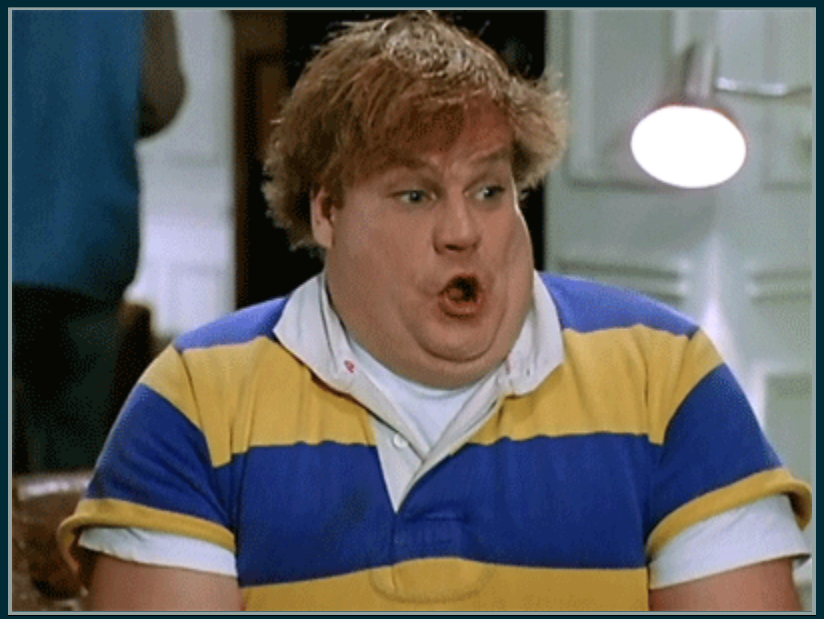
PLAIN JAVASCRIPT

```
var heading = document.getElementById("header");

heading.innerHTML = "Hello, world!";
heading.style.backgroundColor = "#eee456";
```

JAVASCRIPT + JQUERY

```
$("#heading")
  .html("Hello, world!")
  .css("background-color", "#eee456");
```



LEARN JAVASCRIPT FIRST!

- JQuery does a lot of **magic** under the hood
- Make sure you understand Javascript well before learning JQuery
- Including Jquery requires the browser to interpret an extra ~20,000 lines of code

IMAGE FORMATS

PNG

- Portable Network Graphics
- Supports transparency
- Lossless data compression, but high compression means more decoding
- Great for photos and images with lots of colour variations

JPG

- Joint Photography Experts Group
- No transparency
- Lossy compression
- Great for photos and images with lots of colour variations

GIF

- Graphics Interchange Format
- Supports transparency (no semi-transparency)
- Supports animation
- Very limited colour palette (256)
- Limited compression, so big filesize

SVG

- Scalable Vector Graphics format
- Supports transparency
- Best suited to flat colours and geometric shapes
- Can be scaled to any size without lost quality or increased file size

JPG AND SVG ARE USUALLY ENOUGH!

- JPG is great for photo-esque images
- SVG is great for simpler shapes, logos, etc

REVIEW

QUESTIONS?

WHAT'S NEXT?

BACK-END WEB DEVELOPMENT

THANK YOU!