HACK YALE

< ADVANCED JAVASCRIPT />

WWW.HACKYALE.COM

HACK YALE

< ADVANCED JAVASCRIPT />

DAY 1

THE FUN BEGINS

WELL HELLO THERE

The Agenda

- Introductions
- What is this HackYale?
- The Internet
- The Javascript

INTRODUCTIONS

HACK YALE

INTRODUCTIONS

RAFI KHAN

- Sixth time teaching HackYale
- Senior, Pierson, Computer Science Major
- Google.org, iXperience, GAKKO
- <3 Javascript

INTRODUCTIONS

YOU ALL

- Budding programmers, all-star designers
- Coding is more fun with friends!

WHAT IS THIS HACKYALE?

PROGRAMMING + HACKING CULTURE

HACKYALE

Practical, not as theoretical / academic as a Yale CS class

- Zero -> prototype
- Not training CTOs
- Preparing you with the tools to train yourself to do whatever you want

WHY HACKYALE?

Good ideas + good developers = good tech companies

- Yale ⊃ many students with good ideas
- Yale ⊅ many students who can implement those ideas

THE ADVANCED JS CLASS

- Become a better Javascript programmer
- Learn about what more you need to learn
  - Frameworks, libraries, etc…
- Build your ideas better and faster

GOALS

Focus on processes and psychology of web development more than content

- The idea is your responsibility
- Learn by doing; learn by immersion. Lots of implementation, lots of coding
- Memorization as the emergent byproduct of experience

GOALS

We can't make you successful developers

We can equip you with a kernel of knowledge and key resources with which to make yourselves successful developers

GIVE WHAT YOU CAN

AND MAKE TIME TO GIVE A LOT

HACK YALE

HACKING CULTURE

THE "DO-IT-YOURSELF" ATTITUDE

HACK YALE

ON TO TECHNOLOGY

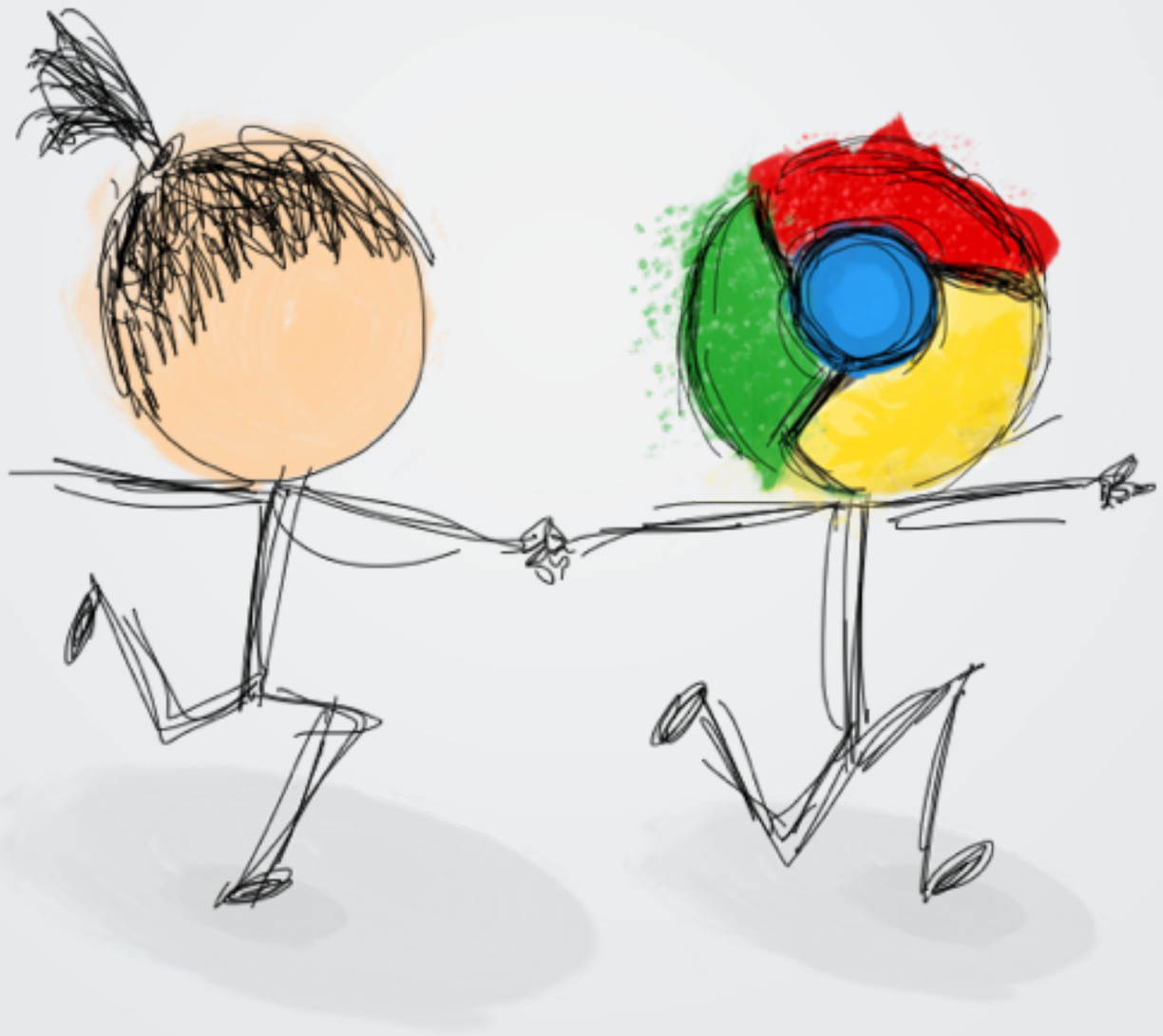
HACK YALE

KEY CONCEPTS

BEFORE WE START

KEY CONCEPT 1

GOOGLE IS YOUR FRIEND



HACK YALE

KEY CONCEPT 1

80% of web development is knowing where to look

Most common answer = Google

- Things to Google:
  - Error messages
  - Syntax
  - Entire problems. Ex: “javascript dropdown menu”

WHAT DO WE DO WHEN WE ENCOUNTER A PROBLEM WE CAN'T IMMEDIATELY SOLVE?

HACK YALE

GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE
GOOGLE GOOGLE GOOGLE GOOGLE GOOGLE

HACK YALE

GOOGLE IS YOUR FRIEND

BUT I'M HERE FOR YOU TOO!

WE'LL GO FAST, BUT THIS IS AN INTERACTIVE CLASS.

PLEASE STOP ME WHENEVER YOU HAVE A QUESTION.

KEY CONCEPT 2

Code is meant for humans to read

- ***Extremely*** important to be clear and concise
- Rely on conventions
  - camelCase instead of spaces, start with lower case…
- Use comments

KEY CONCEPT 3

There are two parts to learning to code

- Concept
  - What you can do
- Syntax and implementation
  - How to do it

WE WILL TEACH YOU CONCEPTS

But the implementation is on you!

- The fastest way to learn is practice, practice, practice
  - Making mistakes helps a lot, too
- Please, please, please, follow along examples in class
  - If you're bored, think of how you can make it more interesting for yourself

THE INTERNET

IT'S VERY INTERESTING

WHAT IS THE INTERNET?

- A Big Wire!
  - Literally a wire that traverses the globe
- Think of the world as a neighborhood:
  - The internet is like the streets

HOW DO WE FIND THINGS?

- Host names and IP addresses
  - facebook.com
  - also 173.252.110.27

COOL! WHAT CAN I DO WITH IT?

- Send requests!
  - To get information
  - Or send information
- A GET request is for getting information
- A POST request is for sending information

A WEBPAGE

THE WAY HUMANS VIEW
INFORMATION THAT WE
RECEIVED FROM THE INTERNET

HACK YALE

IMAGINE FOR A SEC...

THAT A WEBSITE IS A HUMAN BODY

HACK YALE

HTML

The “bones”

- The “content” of the Internet
- Builds the layout, structure and connections
- All the “information” that you can see

CSS

The “skin” or “physical features”

- The “style” of the Internet
- Defines how HTML elements look
  - Width, height, color, position…
- Not covered in this course

JAVASCRIPT

The “muscles”

- The “interaction” or “animation” of the Internet
  - Makes HTML elements interact with one another
    - And with other pages
  - Enables logic, user-based behavior and communication with the rest of the internet
    - Basically, everything *interesting*


THE DOM
ADOMINATION

MORE DEFINITIONS

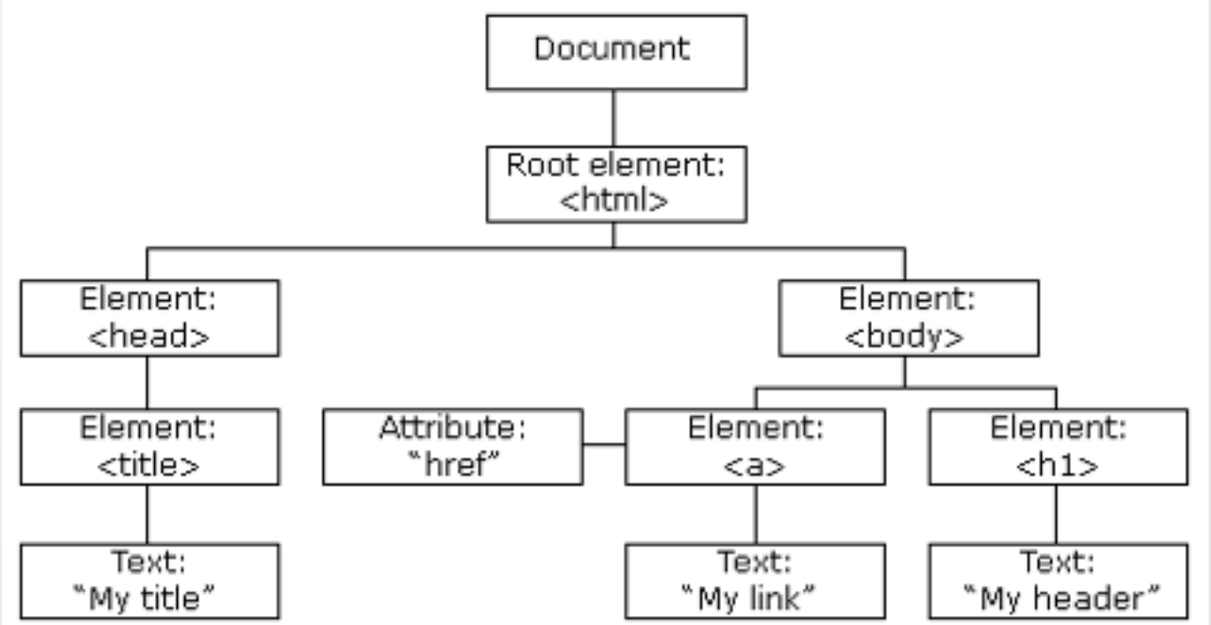
- It is a standard for accessing, changing, adding, and deleting HTML elements and their attributes and values.
- A tree of all HTML elements and attributes that allows for traversal and manipulation.

THE DOM TREE

An HTML Document can be represented as a tree

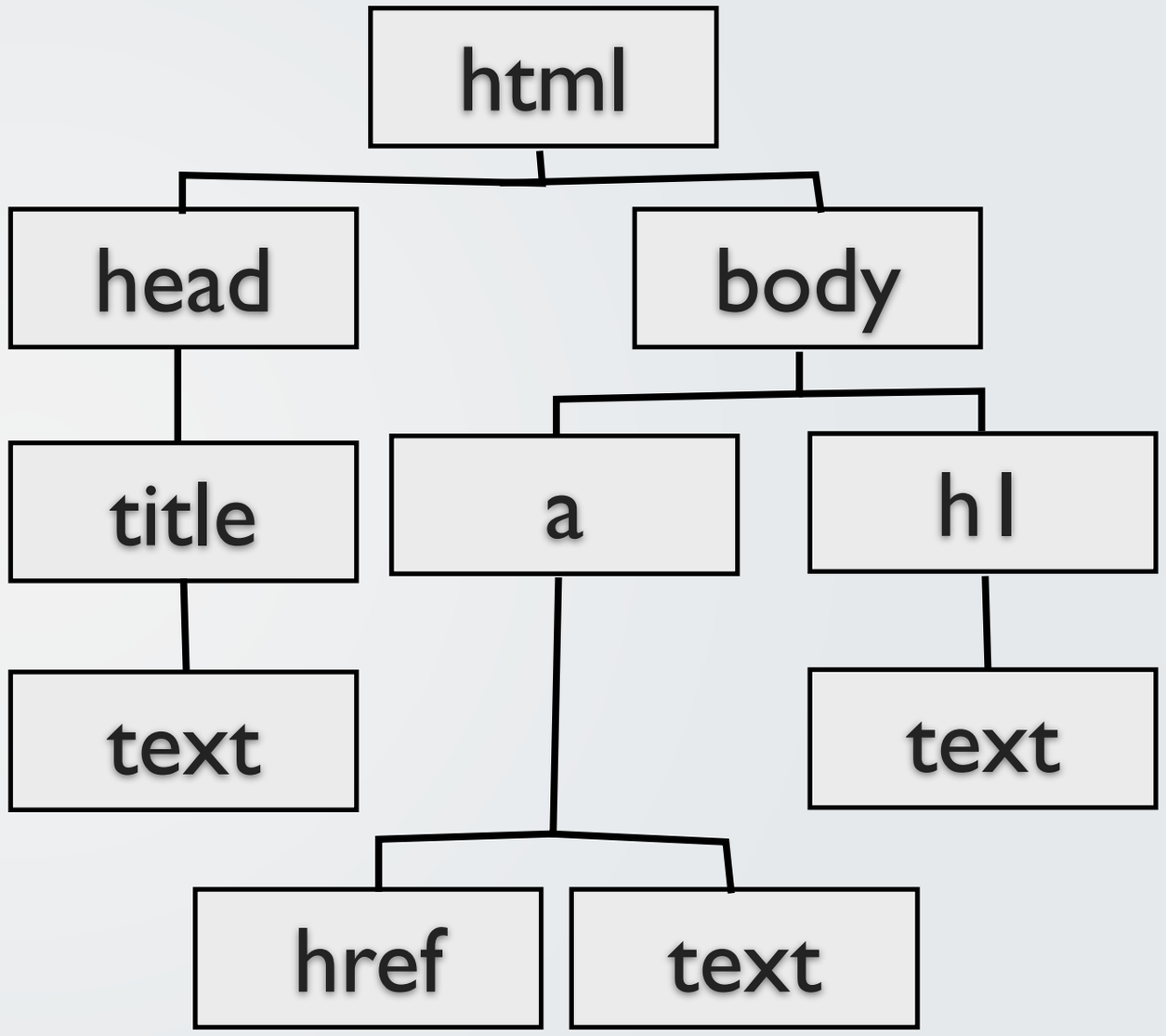
```
<!DOCTYPE html>
<html>
<head>
    <title>My title</title>
</head>
<body>
    <a href="/destination">My link</a>
    <h1>My header</h1>
</body>
</html>
```

=>



HOW THE TREE WORKS

```
<!DOCTYPE html>
<html>
<head>
    <title>My title</title>
</head>
<body>
    <a href="/destination">My link</a>
    <h1>My header</h1>
</body>
</html>
```



USING THE DOM

IT'S EASY TO ACCESS, ALTER, ADD,
AND DELETE ELEMENTS,
ATTRIBUTES, AND TEXT.

THE DOM IN NATIVE JS

WE COULD LEARN ABOUT IT…BUT

THE DOM IN NATIVE JS

We have jQuery

WE COULD LEARN ABOUT IT...


JQUERY
A WHOLE NEW WEB

WHAT IS JQUERY?

JAVASCRIPT LIBRARY: a set of predefined functions that we can mix into normal JavaScript code

WHAT IS JQUERY?

JAVASCRIPT LIBRARY: a set of predefined functions that we can mix into normal JavaScript code

DOCUMENT TRAVERSING: edit the DOM by selecting nodes then adding, removing or altering them

WHAT IS JQUERY?

JAVASCRIPT LIBRARY: a set of predefined functions that we can mix into normal JavaScript code

DOCUMENT TRAVERSING: edit the DOM by selecting nodes then adding, removing or altering them

EVENT HANDLING: simple hookups for listening to and acting upon events that happen in the browser

WHAT IS JQUERY?

JAVASCRIPT LIBRARY: a set of predefined functions that we can mix into normal JavaScript code

DOCUMENT TRAVERSING: edit the DOM by selecting nodes then adding, removing or altering them

EVENT HANDLING: simple hookups for listening to and acting upon events that happen in the browser

AJAX: load content from your server without a page refresh or blocking user action

JQUERY: CSS SELECTOR REVIEW

- Basic selectors:
  - `div` => all elements with tag “td”
  - `.red` => all elements with class “red”
  - `#big` => all elements with id “big”
- Combining selectors
  - `.foo.bar` => class foo and class bar
  - `foo bar` => all `bars` that have an ancestor foo
  - `foo > bar` => all bars that have a parent foo

PRACTICE

- All elements with class “big” and “red”
- All <li> that have class “red”
- All <td> that are the immediate children of <tr>s with class “special”
- All <li> that are descendants of a <ul> with ID “some-list”

THE JAVASCRIPT

FINALLY!

THE WORKFLOW

- Create a folder called “hackyale” on your Desktop (or somewhere safe)
- In it, create a folder called “week1”
- Open Sublime and open that folder (file -> open)
- Create script.js
- Type “alert(“hello world”);” and save it
- Open script.js in Google Chrome

OOPS!

- In week1, create “index.html”
- Make it look like the following:

index.html × script.js ×

```
1  <!DOCTYPE html>
2  <head>
3    <script src="./script.js"></script>
4  </head>
5  <body>
6  </body>
7  </html>
```

THE CHROME CONSOLE

- View -> Developer -> Javascript Console (command-opt-j)
- You can just type Javascript into here as well
  - Comes loaded with all the Javascript on the page!
  - But it's cumbersome to write multiple lines
- Error messages will show up here

LETS PLAY AROUND

VAR, FUNCTIONS, THE SIX TYPES

HACK YALE

KEY LESSONS: 1

- Javascript is a *too permissive*
- You can do lots of things and it won't complain
  - Leave off var, forget semicolons…
- DON'T!
  - These lead to weird bugs that are very hard to find
  - Also the sign of a novice programmer

KEY LESSONS: 2

- Javascript is a *weakly typed* language
  - A variable can be any type
    - And can also change types
- This allows for a lot of flexibility
  - In arrays and objects, for example
- But can also be dangerous

KEY LESSONS: 3

- In Javascript functions are *first-class objects*
  - They can be assigned to variables!
  - They can also be passed as parameters to other functions
- You can also create *anonymous functions*
  - These are extremely common, and you'll get used to them

PRACTICE

- Write a function helloUniverse() that prints out “hello universe” to the console. Call it and verify it works.
- Write a function callTwice that takes a parameter a function and calls it twice. Call it with helloUniverse as a parameter
- Call callTwice with an anonymous function that prints out “hello back!”

EVENTS

YOU'RE INVITED!

THE BIGGEST THING ABOUT JS

- Javascript is event-based!
- Code does *not* always run in code-line order
  - *Asynchronous* functions start a task, and run it in the background
    - When they finish, they call a *callback* function

AN EXAMPLE

```
1  var num = doSomeCalculation();  // Runs first
2  var result = internetTask(num); // Runs second
3  console.log(result);            // Is null!
```

THE SOLUTION

```
1  var num = doSomeCalculation(); // first
2  var callback = function(res) { // second
3    console.log(res);            // fourth
4  };
5  internetTask(callback);        // third
```

THE SOLUTION: ALTERNATE

```
1  var num = doSomeCalculation(); // first
2  internetTask(function(res) {   // second
3    console.log(res);            // third
4  });
```

PRACTICE

SELECT A NODE

CREATE A NEW NODE

APPEND TO FIRST NODE

HACK YALE

PRACTICE

GET DATA FROM THE REDDIT API

HACK YALE

JAVASCRIPT OBJECTS

HACK YALE

WHAT ARE OBJECTS, ANYHOW?

One of the six (6) Javascript data types

A collection of properties and methods

- Properties: variables attached to an object
- Methods: functions attached to an object

OBJECTS ARE WRITTEN IN JSON

Javascript Object Notation

- Key-value pairs
  - Keys are usually strings, but can be “anything”
    - Arrays are just objects where keys are numbers
    - Can’t be null, undefined, and some other values
  - Values can be anything, including other objects

DOT NOTATION

In Javascript, we can access *properties* and *methods* of objects using dot notation

- SYNTAX: `myObject.my_prop or myObject.my_method()`
- We have already seen this! Remember `console.log`?
  - **console** is an *object*
  - **log** is a *method* of the console object

SYNTAX IN DEPTH

To create a singular instance of an object:

```
var myPerson = {
    // Properties of myPerson
    firstname: "John",
    lastname: "Smith",
    age: 23,

    // Methods of myPerson
    fullname: function(){
        return this.firstname + " " + this.lastname;
    }
}

console.log( myPerson.age ); // -> 23
console.log( myPerson.firstname ); // -> John
console.log( myPerson.fullname() ); // -> John Smith
```

BRACKET NOTATION

We can also access *properties* and *methods* of objects using bracket notation

- SYNTAX: `myObject["my_prop"] or myObject["my_method"]()`
- We prefer dot syntax because it is easier to type and read
- Sometimes, however, bracket notation will be more convenient, so keep it in mind

METHOD CHAINING

Using dot notation, we can run multiple methods in sequence on one object

- This allows us to keep our code DRY when appropriate
- But, going overboard can make your code hard to read and maintain

METHOD CHAINING EXAMPLE

```
var rooms = "Master Suite|Living Room|Dining Room|Bathroom";
rooms = rooms.split( "|" ).sort().join( ", " );
// -> Bathroom, Dining Room, Living Room, Master Suite
```

THESE TWO ARE EQUIVALENT

```
var rooms = "Master Suite|Living Room|Dining Room|Bathroom";
rooms = rooms.split( "|" );
rooms = rooms.sort();
rooms = rooms.join( ", " );
// -> Bathroom, Dining Room, Living Room, Master Suite
```