Lecture 1

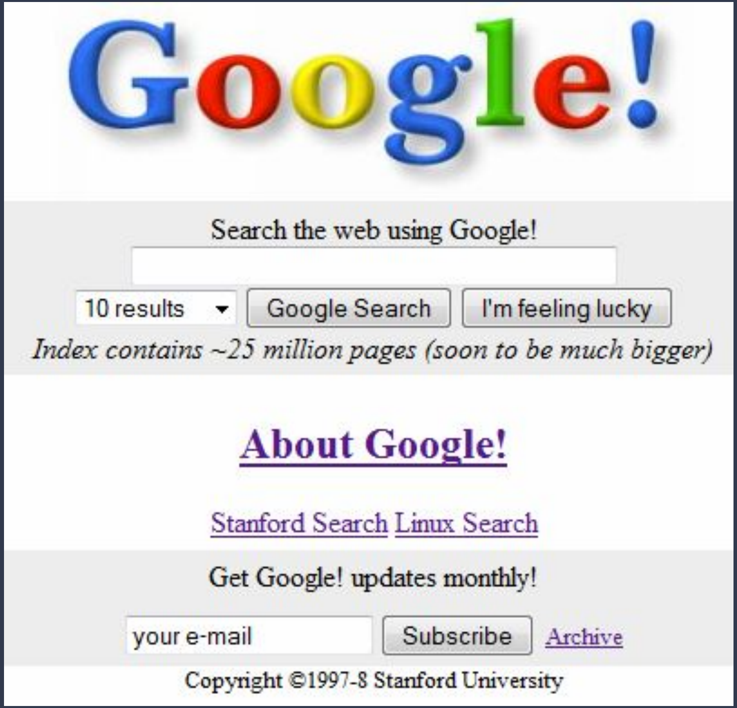
Introduction to JavaScript

Topics

- History of JavaScript
- Browser Debugging Tools
- The JavaScript Console
    - Console.log
    - Syntax Examples

JavaScript History

Static Web


Search the web using Google!

Index contains ~25 million pages (soon to be much bigger)

About Google!

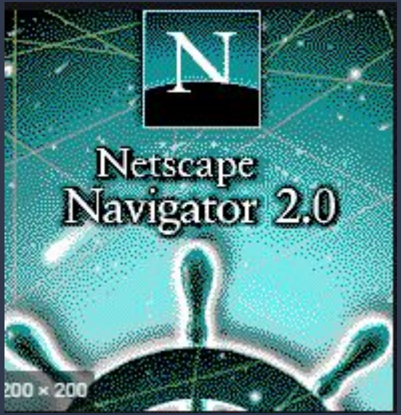
Stanford Search Linux Search

Get Google! updates monthly!

Copyright ©1997-8 Stanford University

- Prior to JavaScript, Web pages were static with HTML and CSS with the occasional plug-in or Java Applet.
- HTML and CSS are markup languages. Markup languages are used to describe and define elements within a document.

JavaScript is Born



- JavaScript was created in just 10 days, by Brendan Eich while working on Netscape Navigator (one of the Internet's first browsers) in 1995

- It was nicknamed Mocha during development, and ultimately named JavaScript to piggyback on the popularity of Java (another programming language)

What is a Programming Language?

```
<link rel=
<script type="text/javas
<script type="text/javascript">
(function(){
  onLoaded: function(request) {
    if (request.name == 'log_error') return;
    log.trace("Ajax.Request: " + (request.name || request.url.substr(0, 30
      )) + "...");
  },
  onComplete: function(request) {
    if (request.name == 'log_error') return;
  },
  onException: function(request, e) {
    if (request.name == 'log_error') return;
    log.fatal(request.url + ': ' + e.name + ' | ' + e.message + ' | ' +
      .stack);
```

- JavaScript is a programming language
- Programming languages are used to communicate instructions to a machine
- Programming languages are used to communicate to a machine
- Programming languages can be used to control the behavior of a machine and express algorithms

Definitions

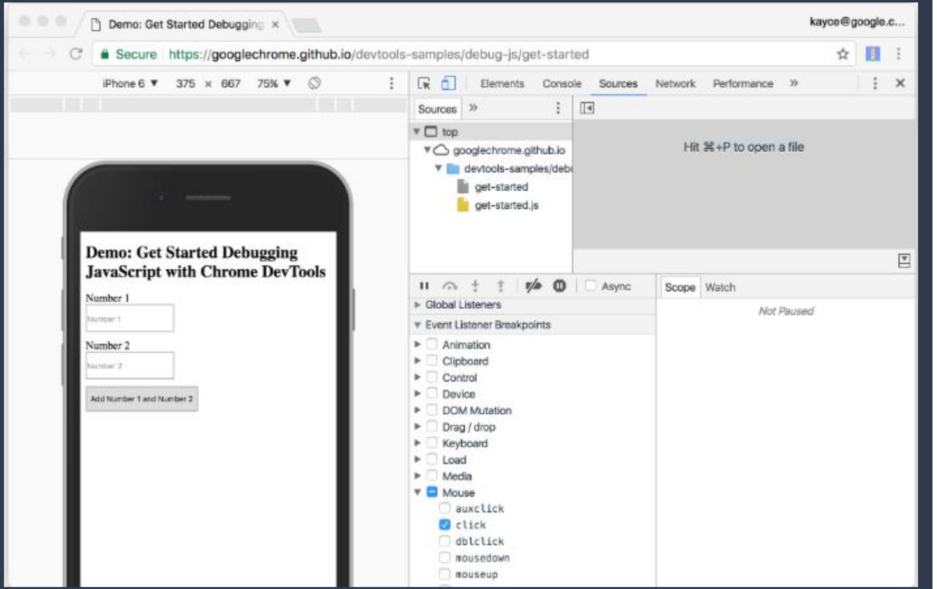
ecma INTERNATIONAL

https://www.ecma-international.org/

- JavaScript (JS) - a high level, dynamic, untyped and interpreted programming language created original for web browsers
- ECMA International - an international non-profit standards organization *(European Computer Manufacturers Association)*
- ECMAScript (ES) - scripting-language specification standardized by ECMA International. (*Implemented well known languages such as JavaScript, JScript and ActionScript)*
- ES2015 (ES6) - the newest version of ECMAScript

Debugging Tools

Browser Developer Tools



https://developers.google.com/web/tools/chrome-devtools
https://developers.google.com/web/tools/chrome-devtools/javascript

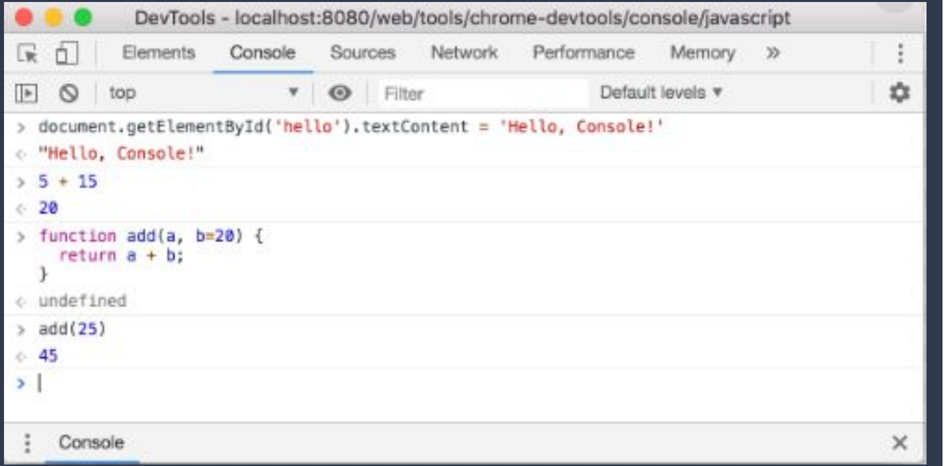
- All modern browser ie. Chrome, Firefox, Microsoft Edge support Web development tools to allow web developers to test and debug their code

- Web **development tools** allow **developers** to work with a variety of web technologies, including HTML, CSS, the DOM, JavaScript, and other components that are handled by the web **browser**

- Chrome DevTools is a set of web developer tools built directly into the Google Chrome browser. DevTools can help you edit pages on-the-fly and diagnose problems quickly.

JavaScript Console

JavaScript Console



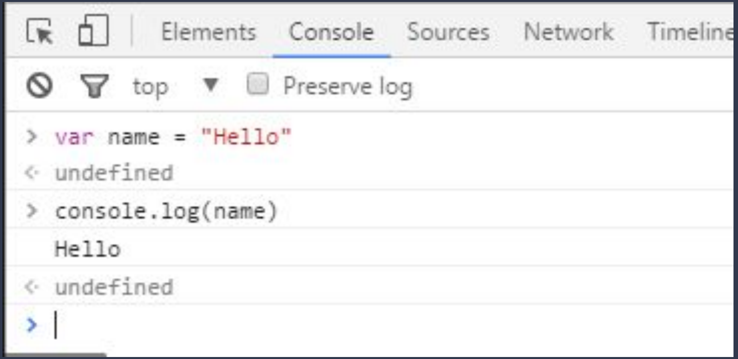
```
> document.getElementById('hello').textContent = 'Hello, Console!'
< "Hello, Console!"
> 5 + 15
< 20
> function add(a, b=20) {
    return a + b;
  }
< undefined
> add(25)
< 45
> |
```

- All modern browser ie. Chrome, Firefox, Microsoft Edge support Web development tools to allow web developers to test and debug their code

- Web **development tools** allow **developers** to work with a variety of web technologies, including HTML, CSS, the DOM, JavaScript, and other components that are handled by the web **browser**

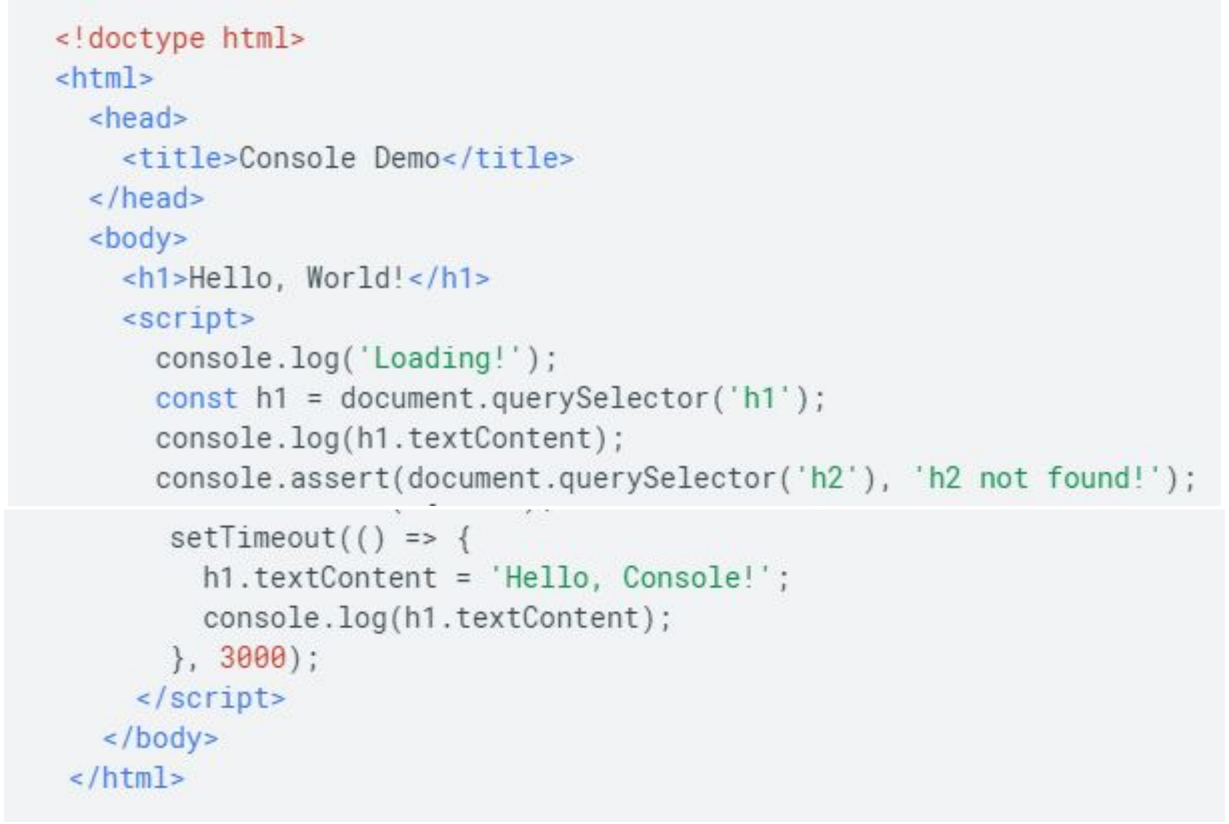
- Chrome DevTools is a set of web developer tools built directly into the Google Chrome browser. DevTools can help you edit pages on-the-fly and diagnose problems quickly.

console.log



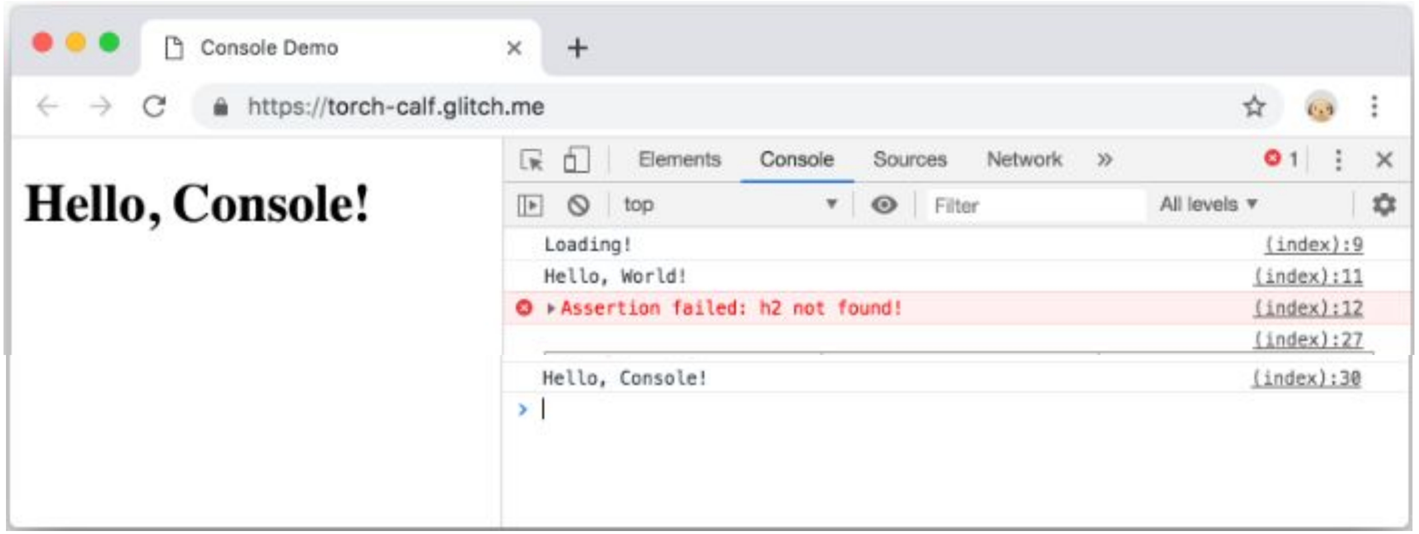
```
> var name = "Hello"
<· undefined
> console.log(name)
  Hello
<· undefined
> |
```

- Web developers often log messages to the Console to make sure that their JavaScript is working as expected.
- To log a message, you insert an expression like `console.log('Hello, Console!')` into your JavaScript.
- When the browser executes your JavaScript and sees an expression like that, it knows that it's supposed to log the message to the Console.



```html
<!doctype html>
<html>
  <head>
    <title>Console Demo</title>
  </head>
  <body>
    <h1>Hello, World!</h1>
    <script>
      console.log('Loading!');
      const h1 = document.querySelector('h1');
      console.log(h1.textContent);
      console.assert(document.querySelector('h2'), 'h2 not found!');
      setTimeout(() => {
        h1.textContent = 'Hello, Console!';
        console.log(h1.textContent);
      }, 3000);
    </script>
  </body>
</html>
```

Example of HTML5 markup and JavaScript console.log()



```
Loading!                                        (index):9
Hello, World!                                   (index):11
▸Assertion failed: h2 not found!                (index):12
                                                (index):27
Hello, Console!                                 (index):30
>
```

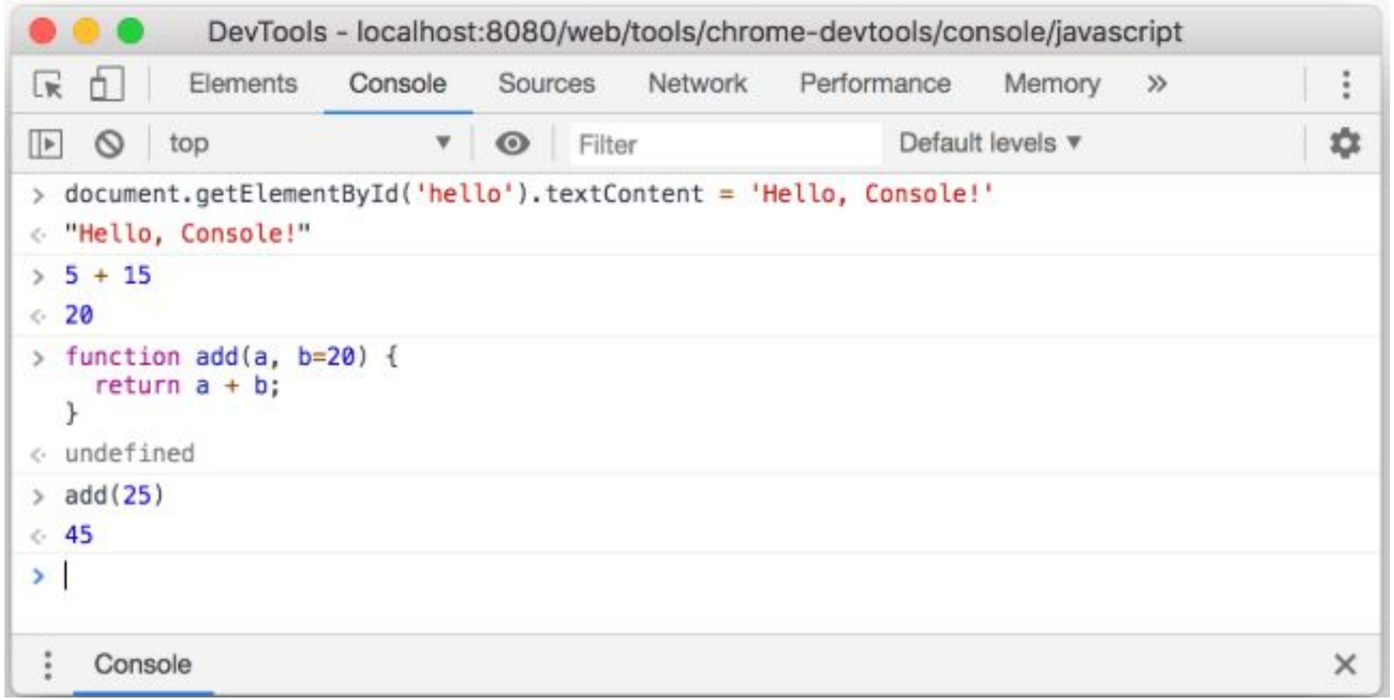
JavaScript console.log output in Chrome Console Window

Run JavaScript in Console

- The **Console** is a REPL, which stands for Read, Evaluate, Print, and Loop. It reads the JavaScript that you type into it, evaluates your code, prints out the result of your expression, and then loops back to the first step.

- View and change the page's JavaScript or DOM
  - Type `document.getElementById('hello').textContent = 'Hello, Console!'` in the **Console** and then press Enter to evaluate the expression. Notice how the text inside the button changes.

- Run arbitrary JavaScript that's not related to the page
  - Sometimes, you just want a code playground where you can test some code, or try out new JavaScript features you're not familiar with. The Console is a perfect place for these kinds of experiments.



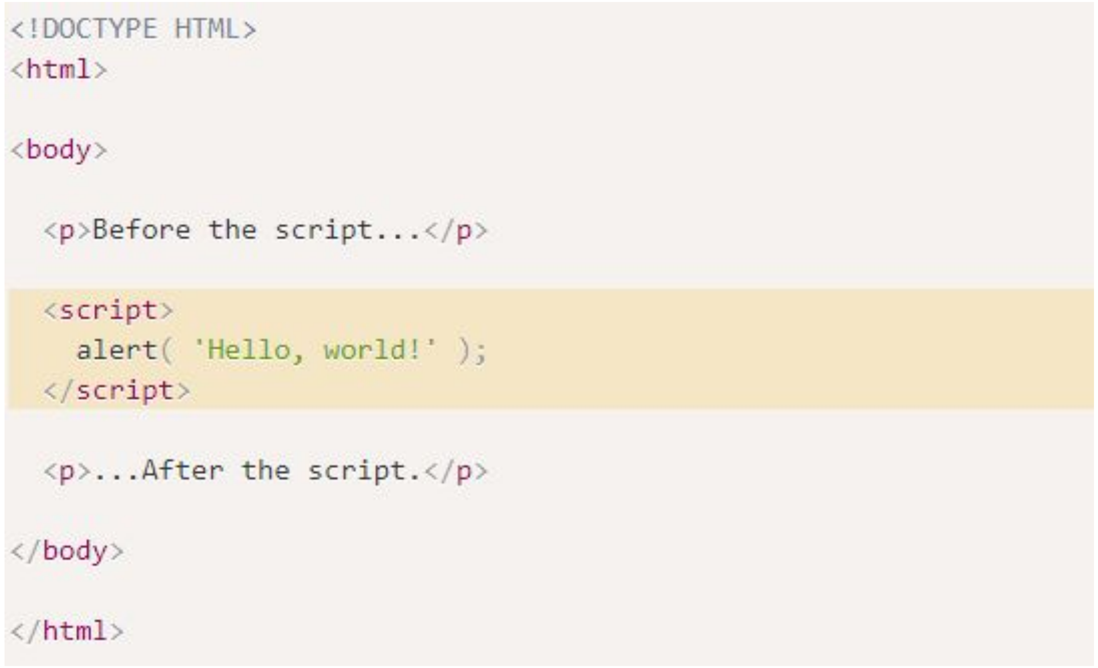
```
> document.getElementById('hello').textContent = 'Hello, Console!'
< "Hello, Console!"
> 5 + 15
< 20
> function add(a, b=20) {
    return a + b;
  }
< undefined
> add(25)
< 45
>
```

Example of running JavaScript in the Console

Script Tag

The Script Tag

JavaScript programs can be inserted into any part of an HTML document with the help of the <script> tag.



```
<!DOCTYPE HTML>
<html>

<body>

  <p>Before the script...</p>

  <script>
    alert( 'Hello, world!' );
  </script>

  <p>...After the script.</p>

</body>

</html>
```

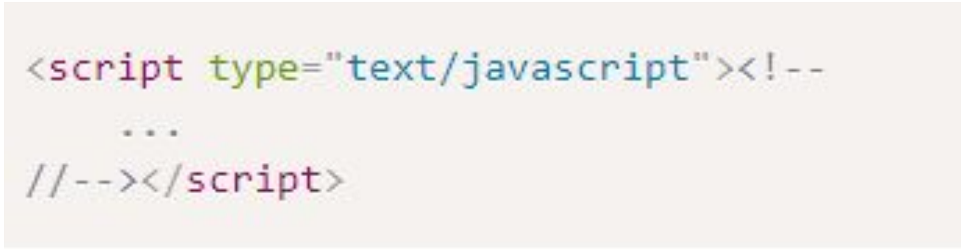
Modern Markup

The <script> tag has a few attributes that are rarely used nowadays but can still be found in old code:

The type attribute: <script type=…>

The old HTML standard, HTML4, required a script to have a type. Usually it was type="text/javascript". It’s not required anymore. Also, the modern HTML standard totally changed the meaning of this attribute.

In really ancient books and guides, you may find comments inside <script> tags, like this:



```
<script type="text/javascript"><!--
    ...
//--></script>
```

External Scripts

If we have a lot of JavaScript code, we can put it into a separate file.

Script files are attached to HTML with the `src` attribute:

```html
<script src="/path/to/script.js"></script>
```

We can give a full URL as well. For instance:

```html
<script src="https://cdnjs.cloudflare.com/ajax/libs/lodash.js/4.17.11/lodash.js"></script>
```