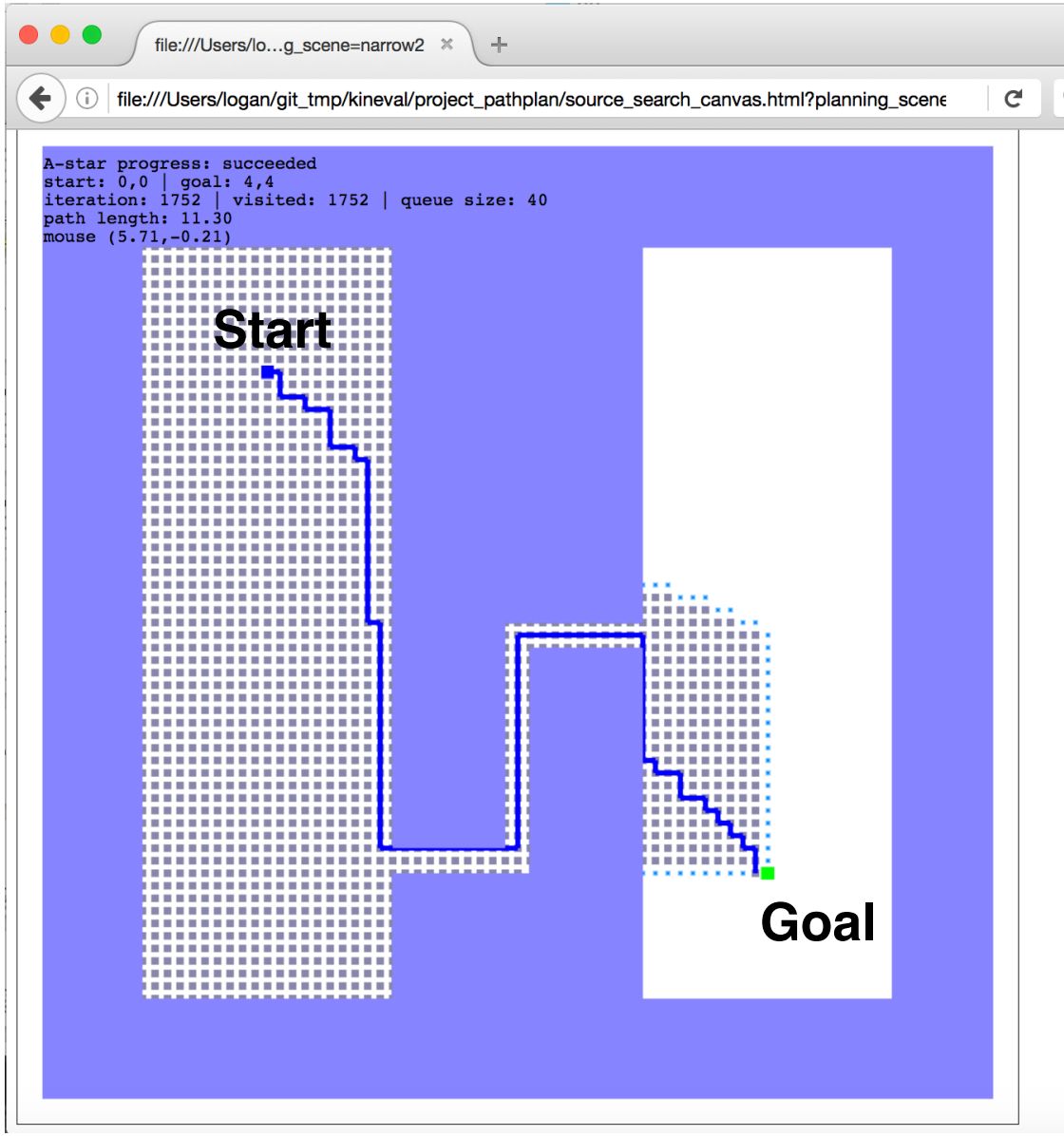
```
<html>

<title>JavaScript/HTML5 and git Tutorial</title>

<body>
<a href="autorob.org">
EECS 398 Introduction to Autonomous Robotics <br>
ME/EECS 567 ROB 510 Robot Kinematics and Dynamics
</a>
</body>
</html>
```

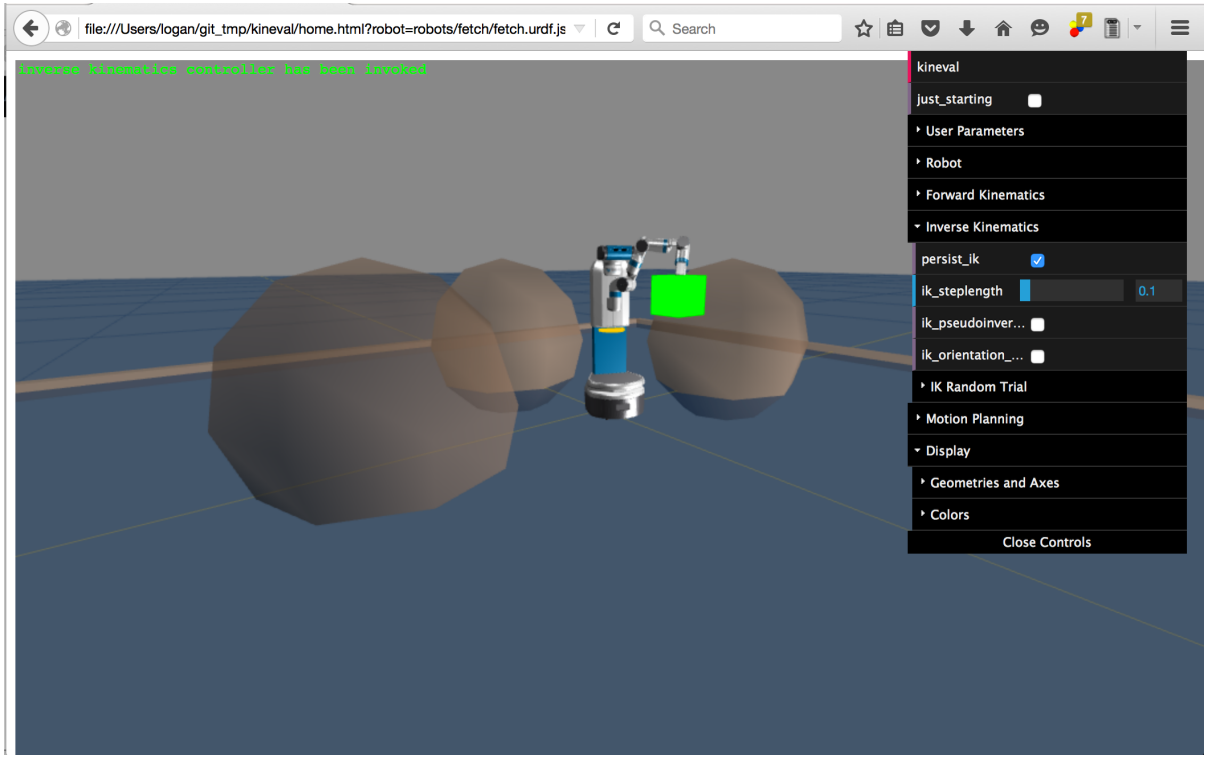
Project 1:
2D Path Planning

- A-star algorithm for search in a given 2D world
- Implement in JavaScript/ HTML5
- Heap data structure for priority queue
- Grad: DFS, BFS, Greedy
- Submit through your git repository



AutoRob and JavaScript/HTML5

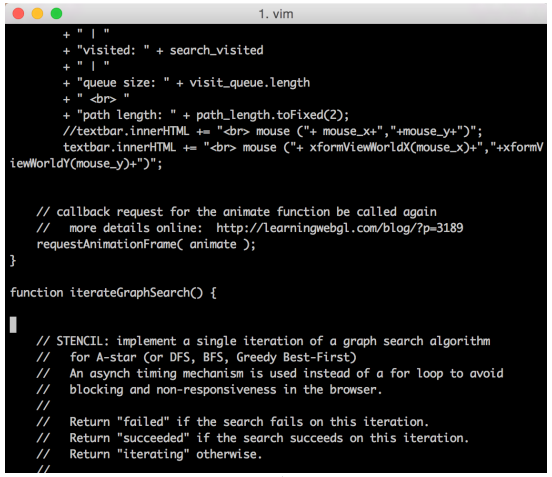
- AutoRob course projects implemented for web browsers using JavaScript/HTML5
- KinEval code stencil was created for this purpose
- Works in commonly-used modern web browsers (Firefox, Chrome, Opera, …)



Work flow

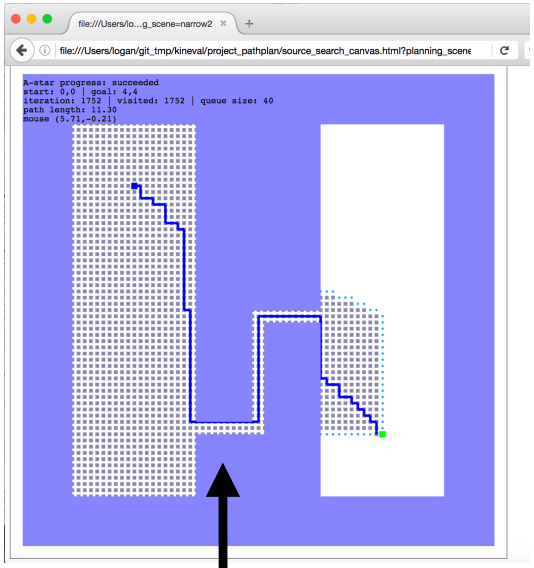
Text editor

Make changes to HTML and JS code



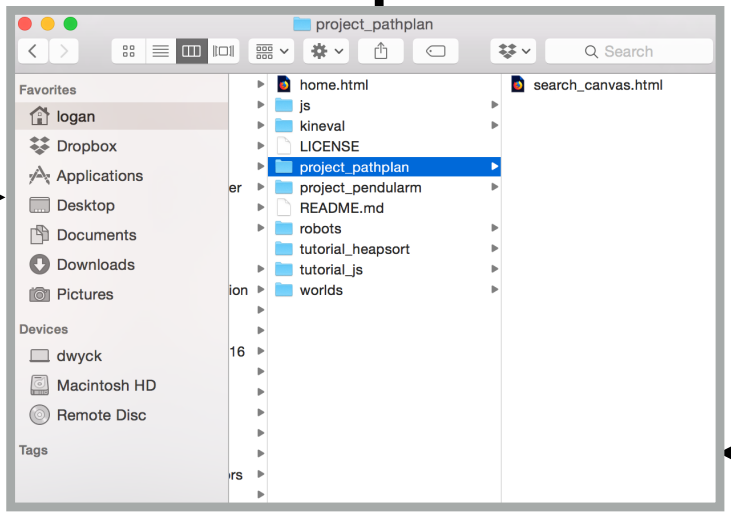
Source code

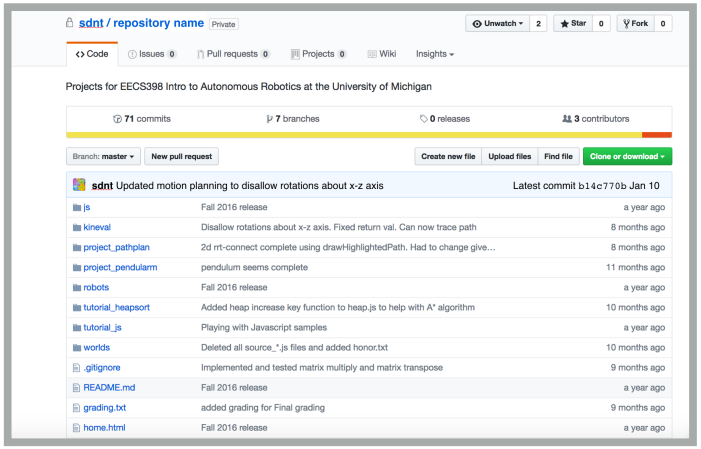
HTML and JS files containing your code



Browser

See HTML and JS code working





git repository

store history of changes and pull grading

autorob.org

Search

AUTOROB | schedule | kineval | git | assignments: | 1

AutoRob

Introduction to Autonomous Robotics
Michigan EECS 398

Robot Kinematics and Dynamics
Michigan ME 567 EECS 567 ROB 510

Fall 2018

GitHub, Inc. (US) https://github.com/autorob/ki

Search or jump to... / Pull requests Issues Marketplace Explore

autorob / **kineval-stencil**

Unwatch 2 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wi

https://github.com/autorob/kineval-stencil

Stencil code for KinEval (Kinematic Evaluator) for robot control, kinematics, decision, and dynamics in JavaScript/HTML5 Edit

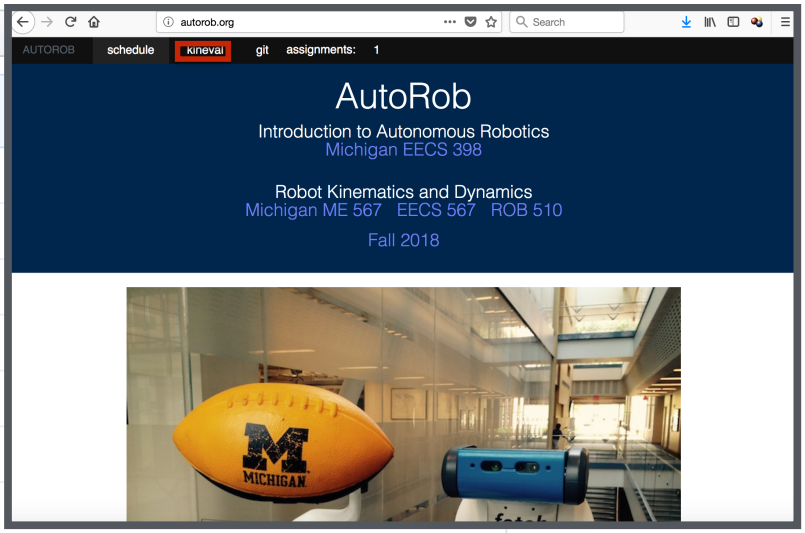
Add topics

2 commits **1** branch **0** releases **2** contributors

Branch: master New pull request Create new file

odestcj initial commit Fall 2018

js	initial commit Fall 2018	
kineval	initial commit Fall 2018	
project_pathplan	initial commit Fall 2018	
project_pendularm	initial commit Fall 2018	
robots	initial commit Fall 2018	
tutorial_heapsort	initial commit Fall 2018	
tutorial_js	initial commit Fall 2018	
worlds	initial commit Fall 2018	4 hours ago



Why JavaScript/HTML5?

Why JavaScript/HTML5?

Spectrum of programming languages

Spectrum of programming languages

C

C++ (maybe)

Matlab
(for numerics)

JavaScript

“Get it done right”

Performance

Robustness

Reusability

Suboptimal tradeoffs

Readability
(e.g., scoping by whitespace)

Mixed Performance
(e.g., compiling down to C)

Overhead Cost
(e.g., complex build and run time)

“Get it done quickly”

Rapid development

Visualization/UI

Distributability

JavaScript has Pros and Cons

Pros

Cons

+

- It's free! (and open)
- Portability (Browsers are everywhere!)
- Excellent UI and visualization
  (see threejs.org for examples)
- Reasonable learning curve
  (No complicated build process)
- Translates to C++ style thinking
- Weak typing (**JavaScript is C without discipline**)
- Live introspection and coding

-

- Network access limited to HTTP
  (for security)
- Limited file I/O (for security)
- Floating point issues
  (all numbers represented in IEEE 754)
- Speed and efficiency
  (typically JavaScript is interpreted or compiled to intermediate code)
- Weak typing
- Cryptic debugging messages

JavaScript is C without discipline

- If you are fluent in C or a “classical” programming language, JavaScript has prototyping and minor syntactical differences
    - your instincts and stackoverflow should be enough
- If you are familiar with Matlab or an “scientific” language, you may need to spend time to familiarize yourself with syntax and some data structures
- If you are new to programming, EECS 402 or EECS 280 are highly recommended to be taken in parallel or as a prerequisite

What is JavaScript/HTML5

What is JavaScript/HTML5

- The essential technologies for creating and rendering web pages are HTML5, JavaScript, and CSS
- These technologies structure the run-time environment of web browsers

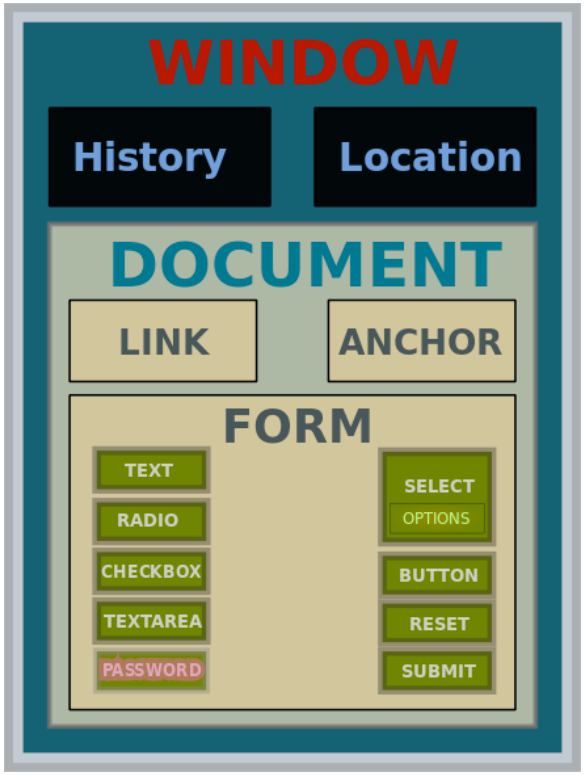
What is JavaScript/HTML5

- **HyperText Markup Language** (HTML5): a markup language for expressing web pages as documents
  - Web browsers read HTML files and render to display
  - Based on the Document Object Model representation

- **JavaScript** (formally ECMAScript): a high-level, dynamic, untyped, interpreted programming language for making “dynamic” web pages
- **Cascading Style Sheets** (CSS): a style sheet language used for describing the presentation of a document

Document Object Model (DOM)

- HTML document defined by elements nested within markup tags (e.g.,
 <h1>text</h1>)
- DOM provides programmatic access to these elements as JavaScript objects
- DOM provides 2 important global objects:
  - "**window**" is the DOM root for a browser tab (a global variable "x" is actually "window.x")
  - "**document**" maintains the current state of the document; auto-populated upon loading
- Each element has a "style" property for CSS



Start with a simple example

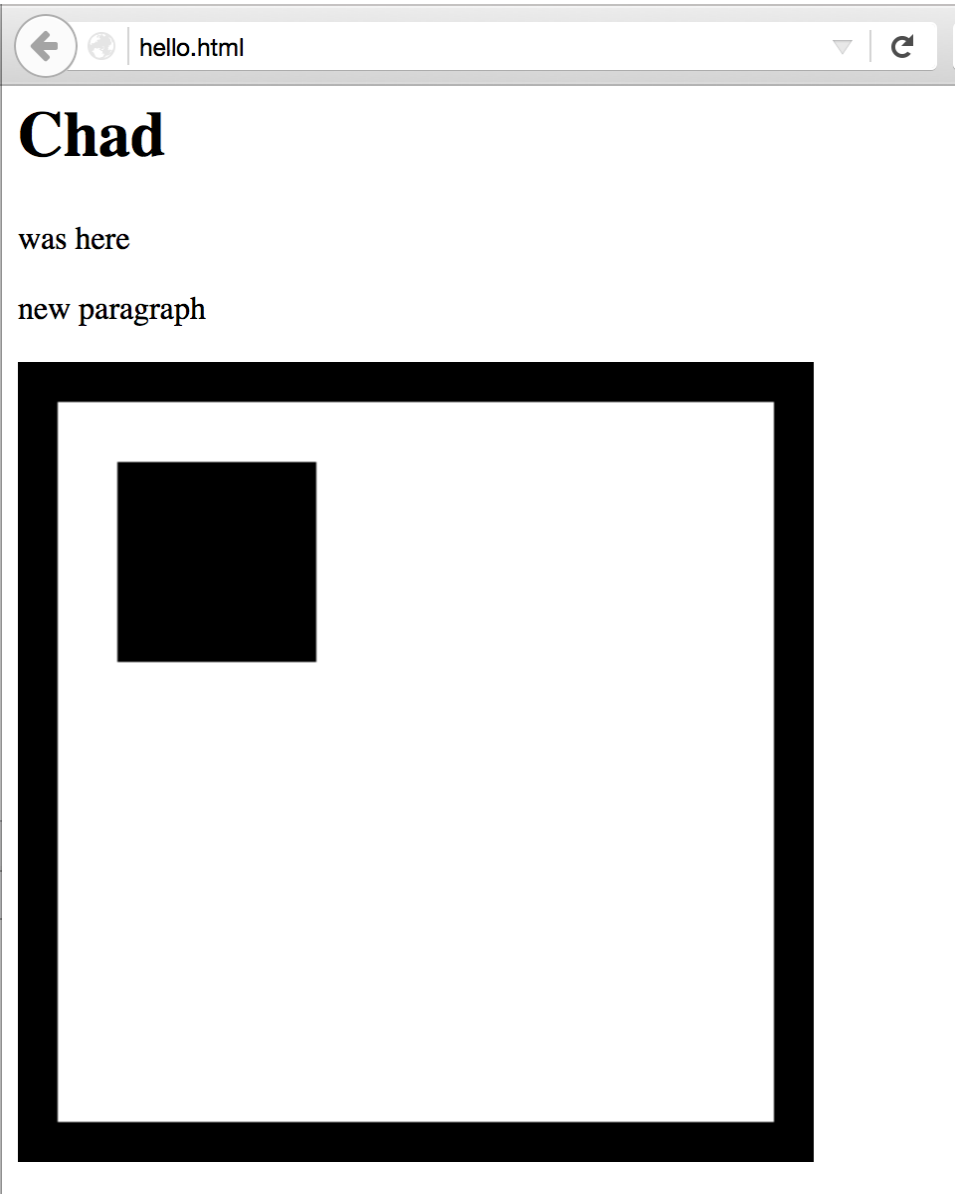
AUTOROB schedule kineval git assignments: 1

Course Schedule (tentative and subject to change)

Note: Assignment descriptions will have updated assignment due dates. Assignment due dates listed in the schedule are merely a guide.

Date	Topic	Reading	Project
Sep 5	Initialization: Course overview, Robotics roadmap, Path planning quick start	Spong Ch.1 Corke Ch.1	Setup git repository
	Week 2		
Sep 10	Path Planning: DFS, BFS, A-star, Greedy best first	Wikipedia	Out: Path Planning
	JavaScript and git tutorial: Heap sort example	Crockford, HTML Sandbox, hello.html (source), JavaScript by Example (source), hello_anim (source), hello_anim_text (source)	
Sep 12	Pendulum Simulation and Numerical Integration: Lagrangian equation(s) of motion, Initial value problem, Explicit integrators: Euler, Verlet, and Runge-Kutta 4	Euler's Method Verlet Integration, Runge-Kutta; Witkin&Baraff 1998: Dynamics Witkin&Baraff 1998: Integrators	
	Week 3		

JAVASCRIPT/HTML5 HELLO EXAMPLE



hello example

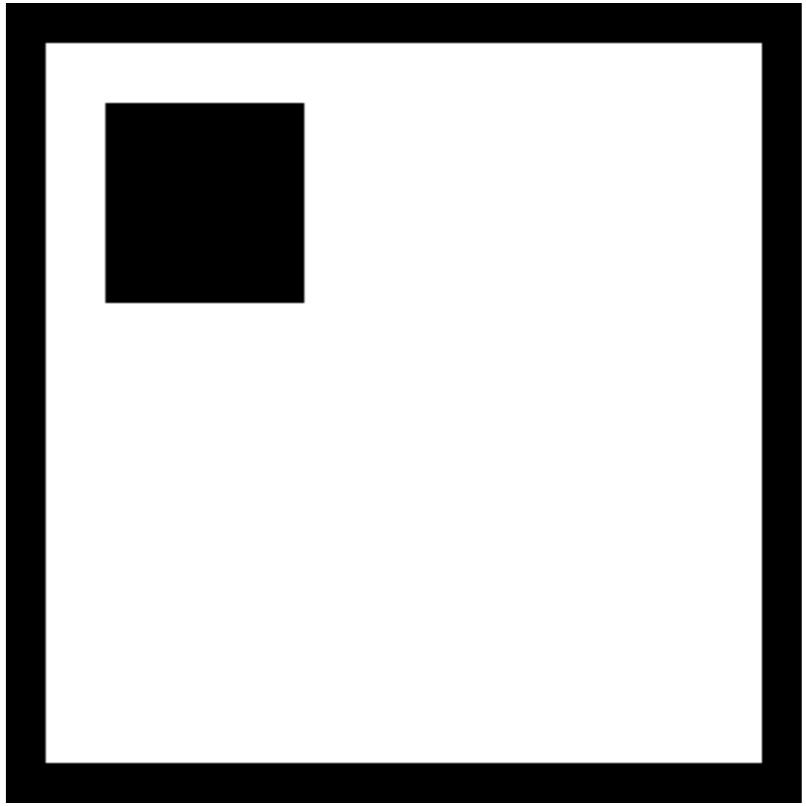
- http://autorob.github.io/examples/hello.html

hello.html

Chad

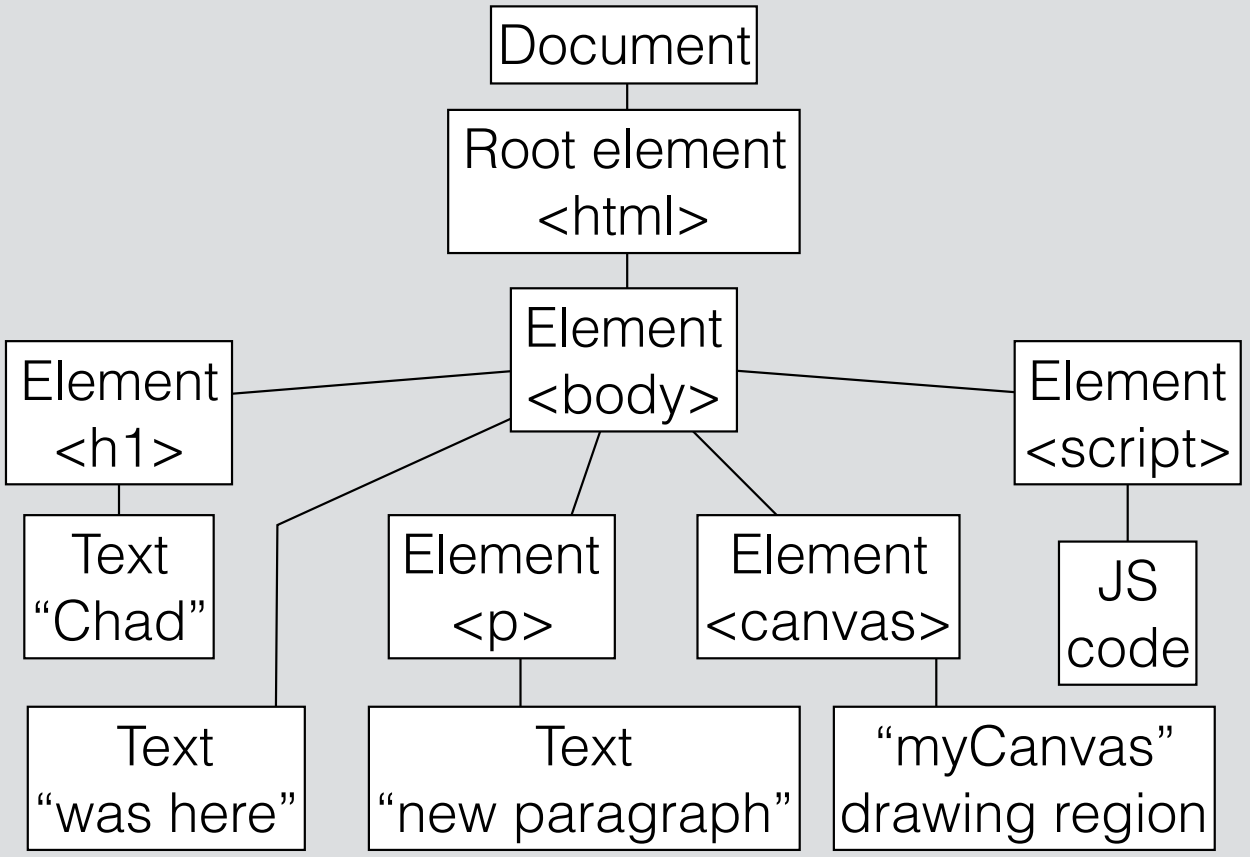
was here

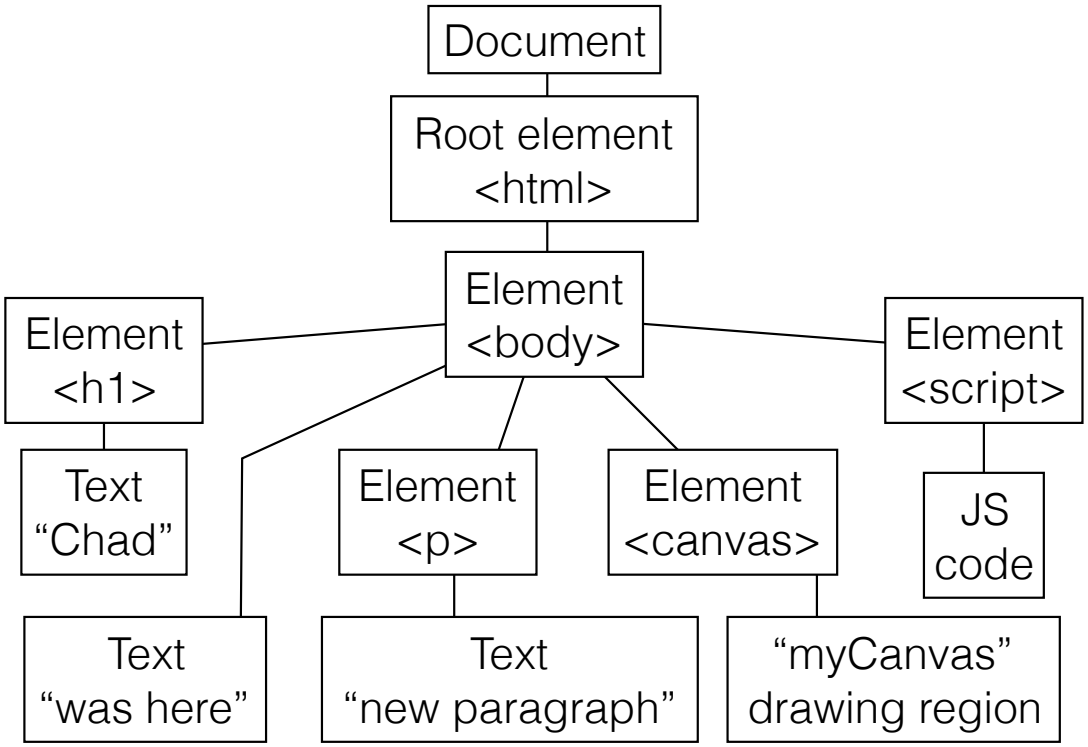
new paragraph



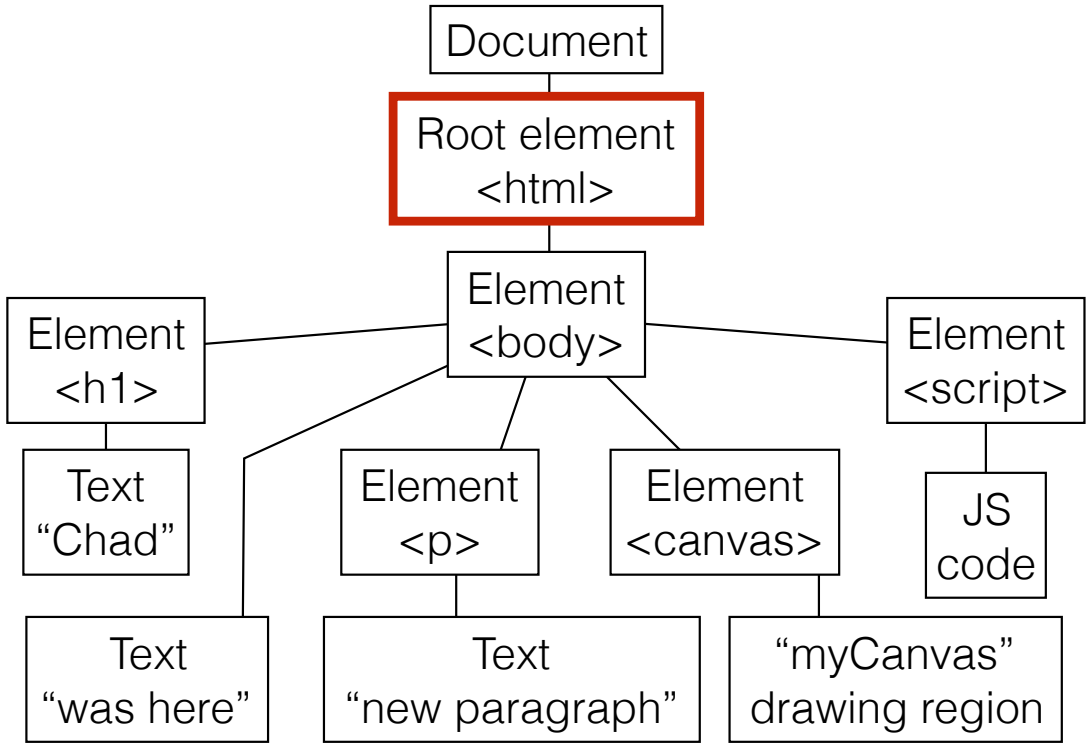
hello example

DOM created by browser upon loading hello.html

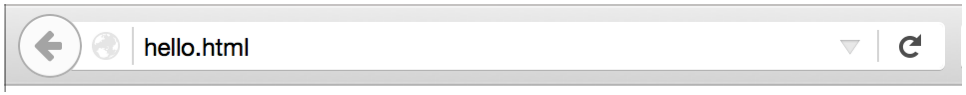


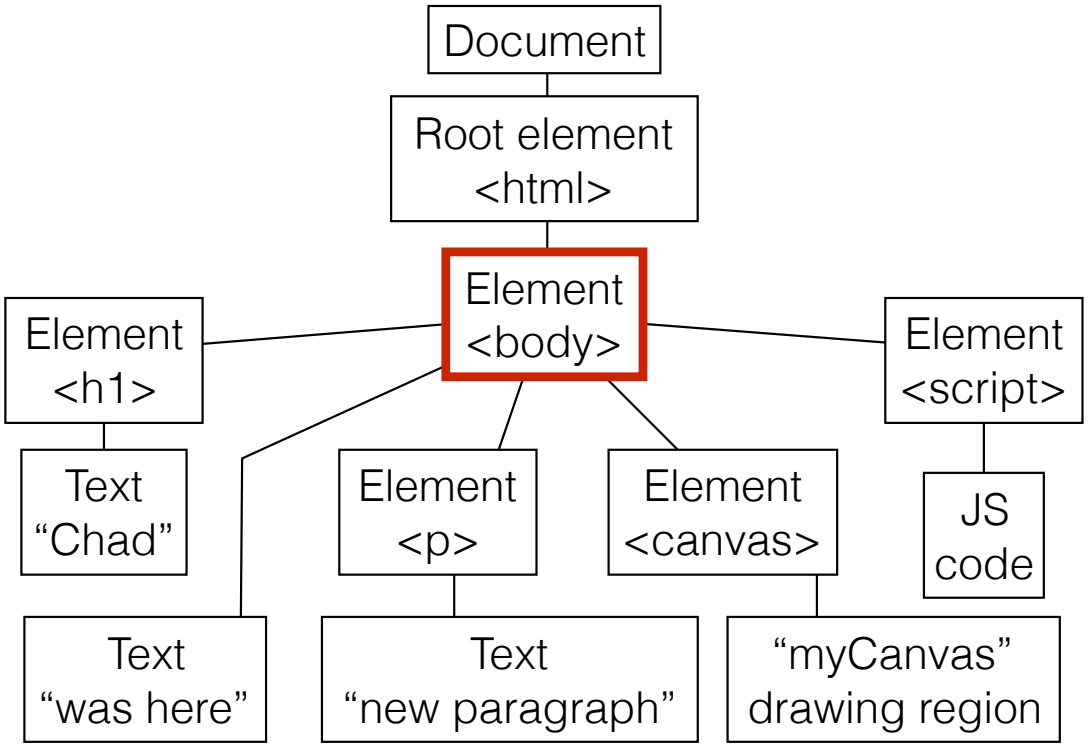



```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>   <!-- say your name big -->

was here   <!-- say something smaller -->

<p>   <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```

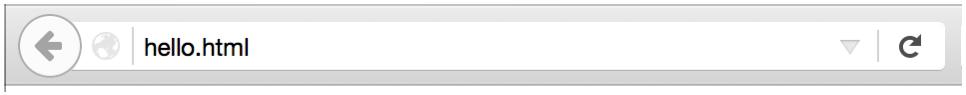


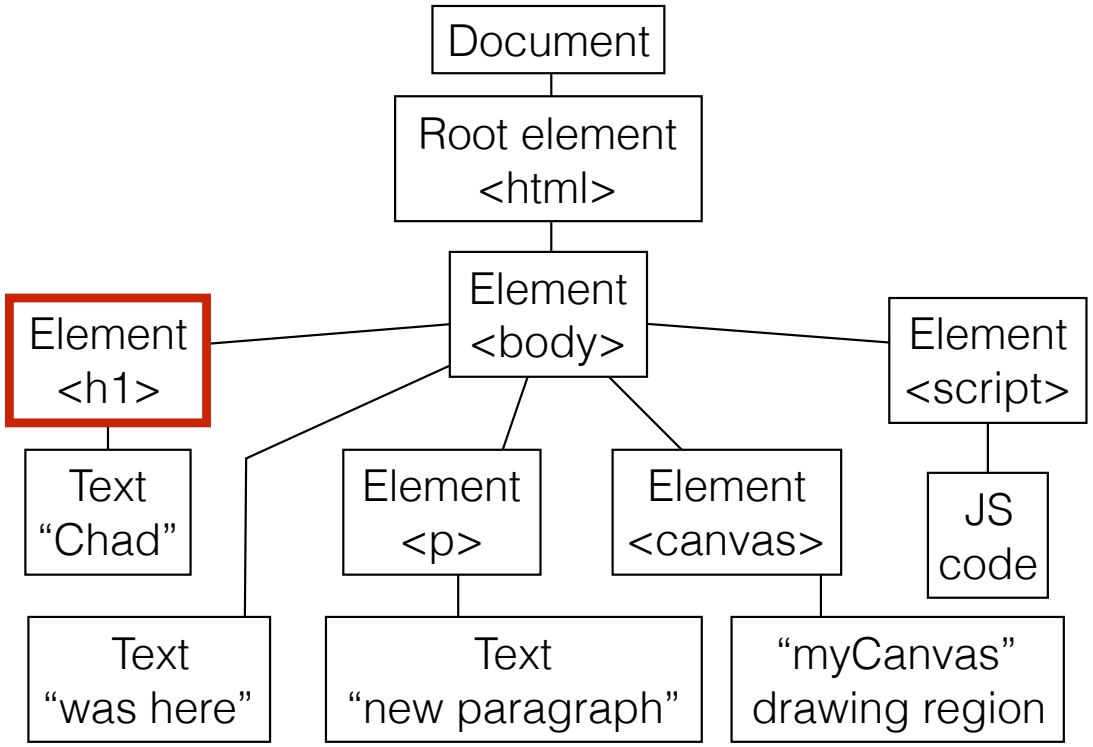

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>   <!-- say your name big -->

was here   <!-- say something smaller -->

<p>   <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```



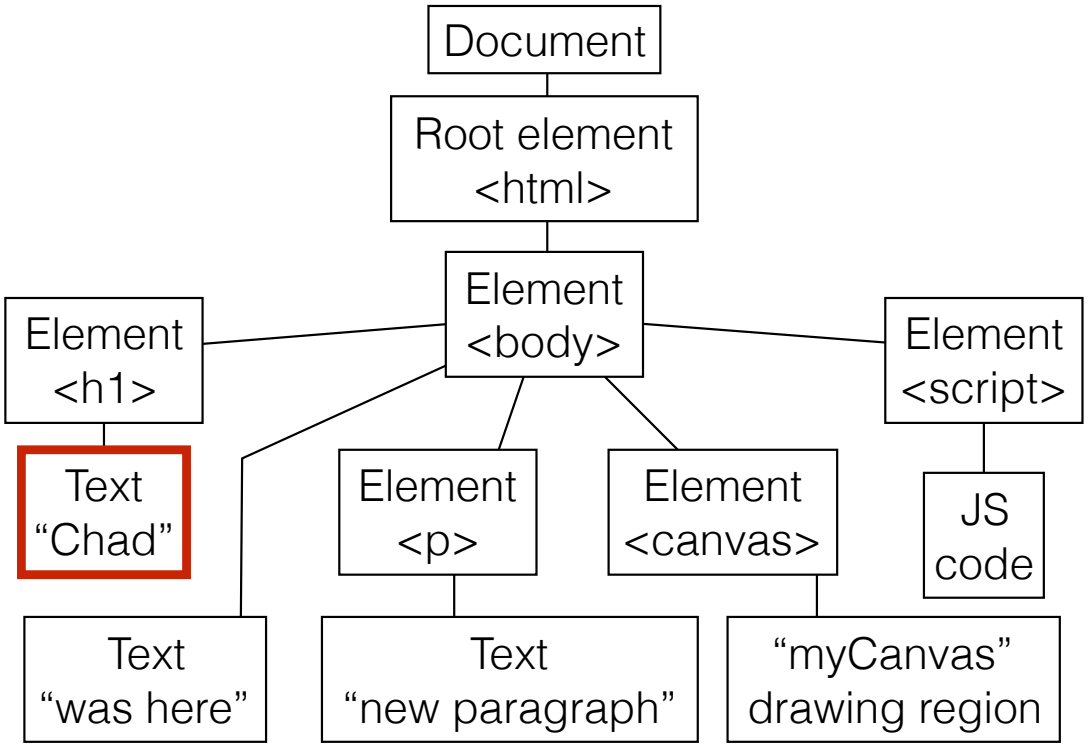



```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>  <!-- say your name big -->

was here  <!-- say something smaller -->

<p>  <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```

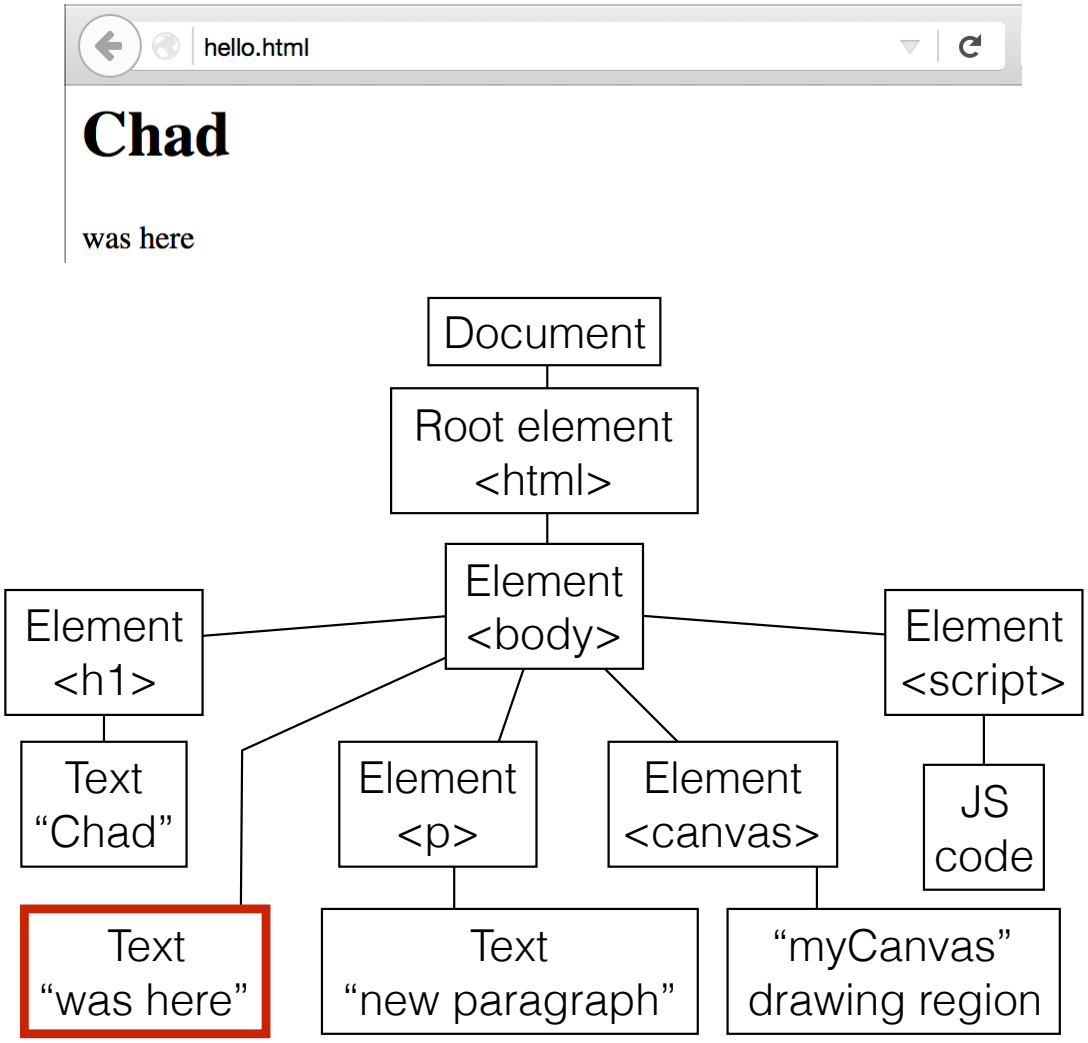




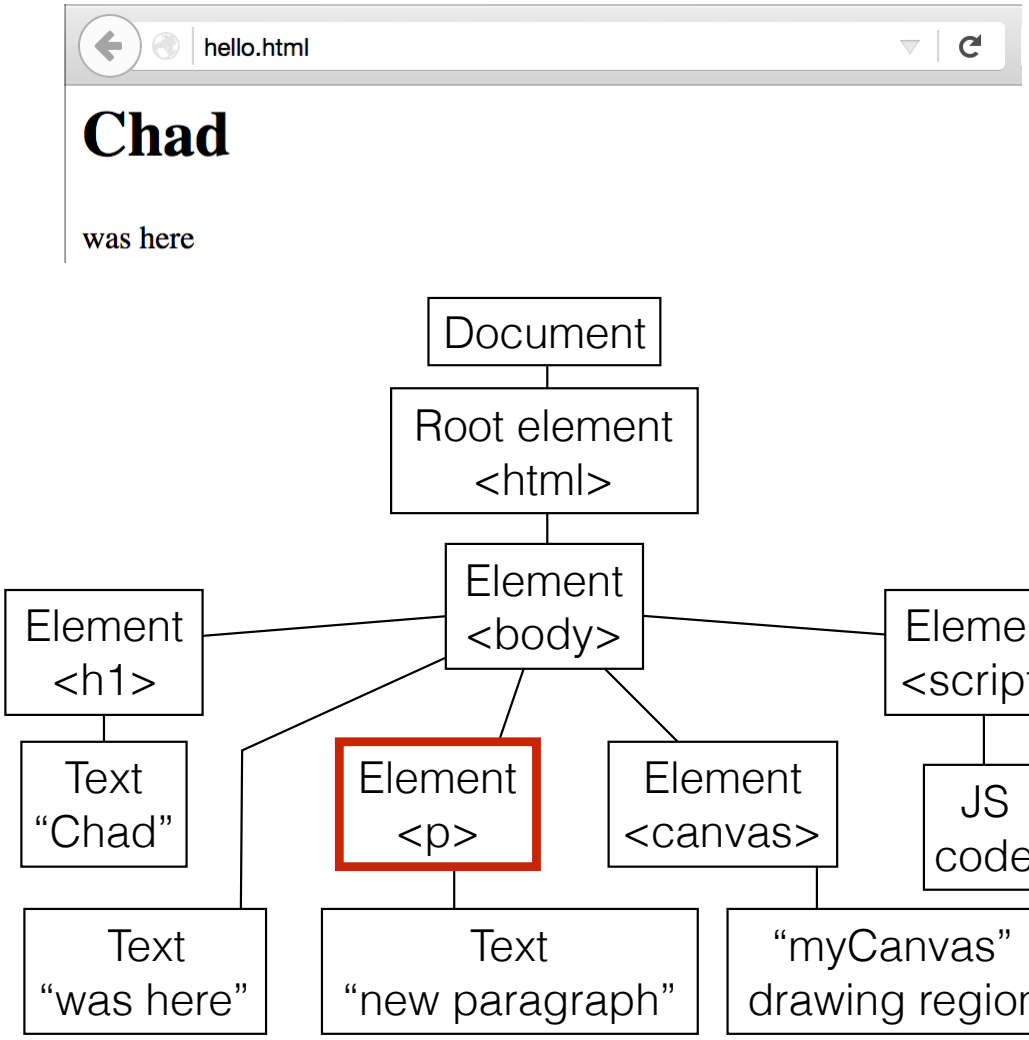

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>  <!-- say your name big -->

was here   <!-- say something smaller -->

<p>   <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```



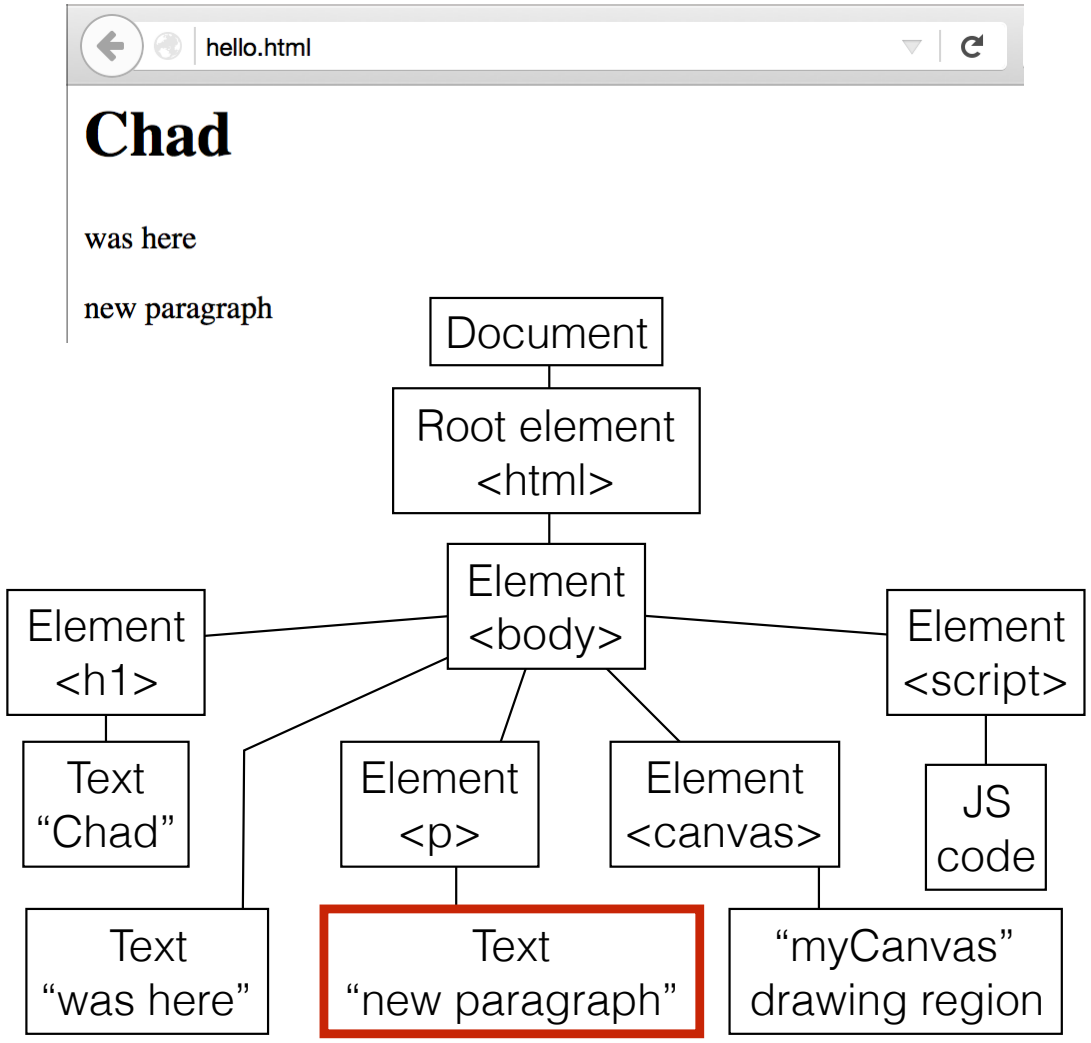

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>   <!-- say your name big -->

was here   <!-- say something smaller -->

<p>   <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```



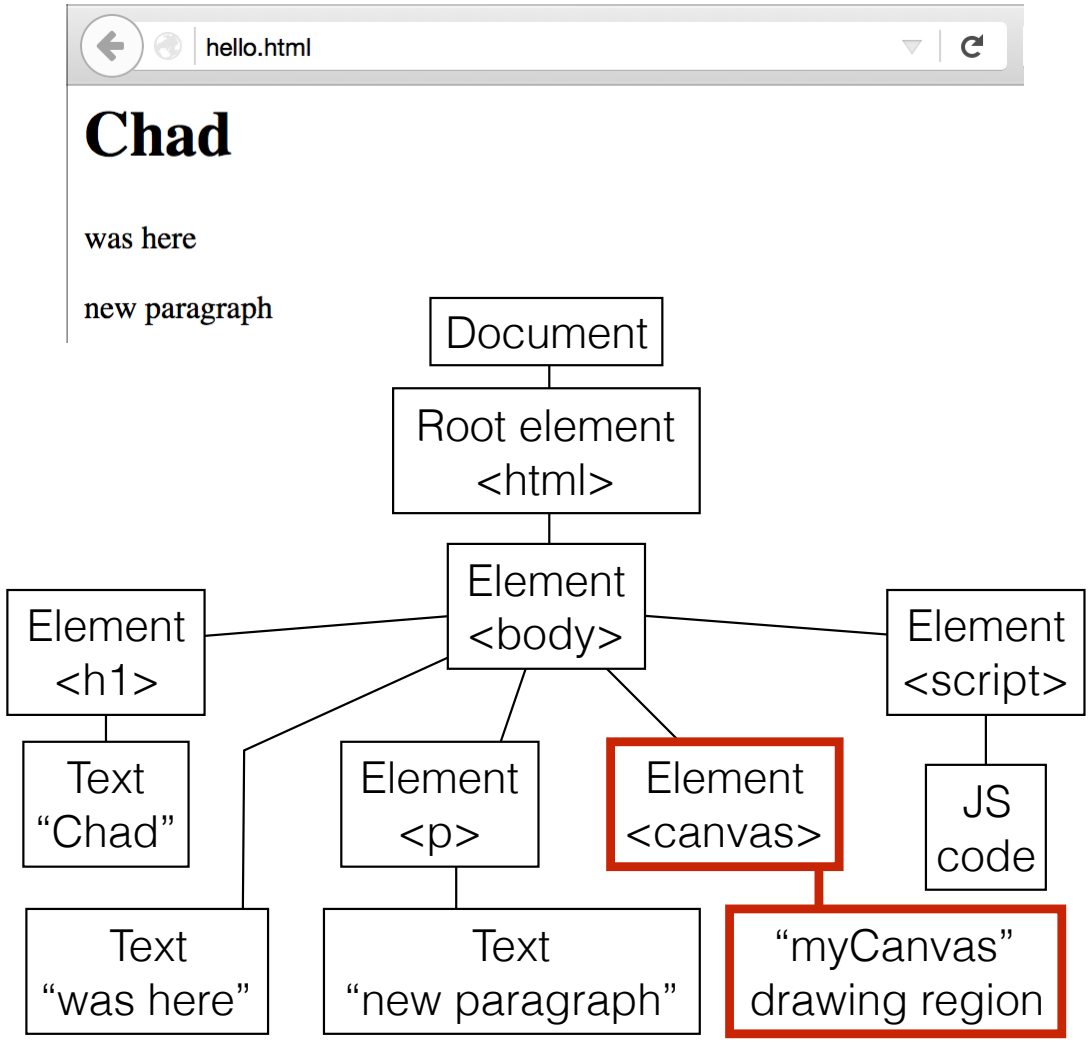

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>  <!-- say your name big -->

was here  <!-- say something smaller -->

<p>  <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```



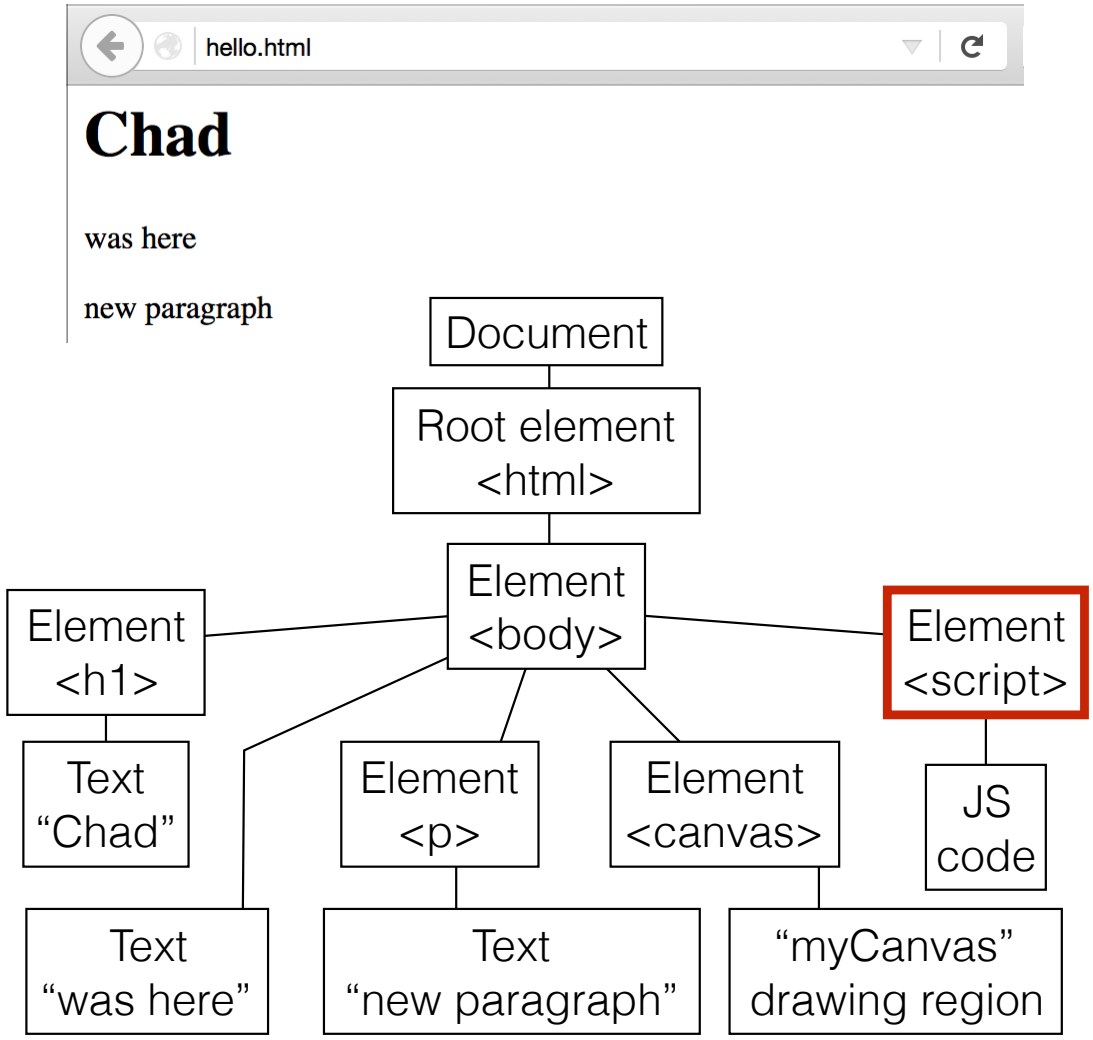

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>  <!-- say your name big -->

was here  <!-- say something smaller -->

<p>  <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```



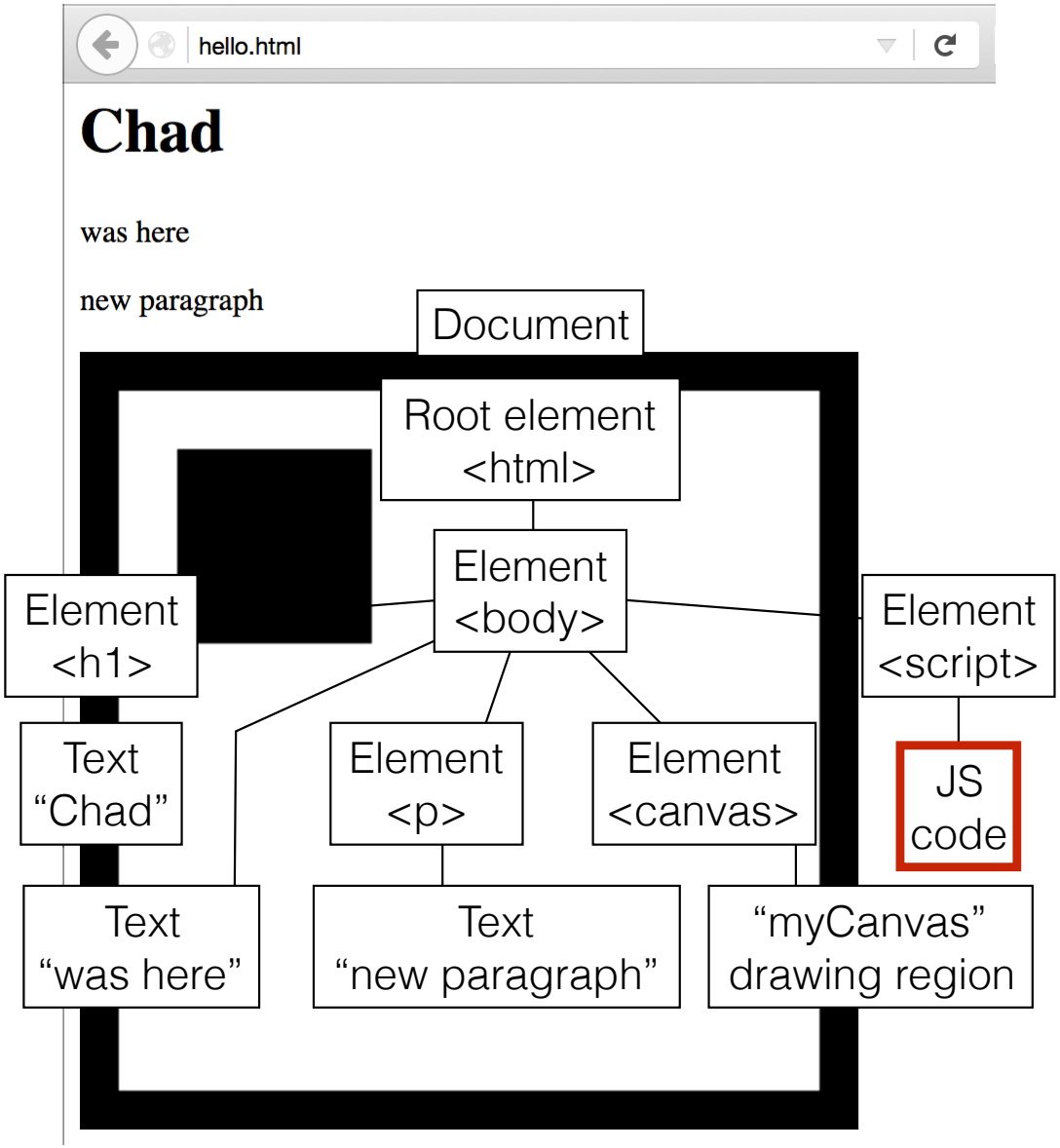

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>   <!-- say your name big -->

was here   <!-- say something smaller -->

<p>   <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```




```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>  <!-- say your name big -->

was here  <!-- say something smaller -->

<p>  <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```




```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>  <!-- say your name big -->

was here  <!-- say something smaller -->

<p>  <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```

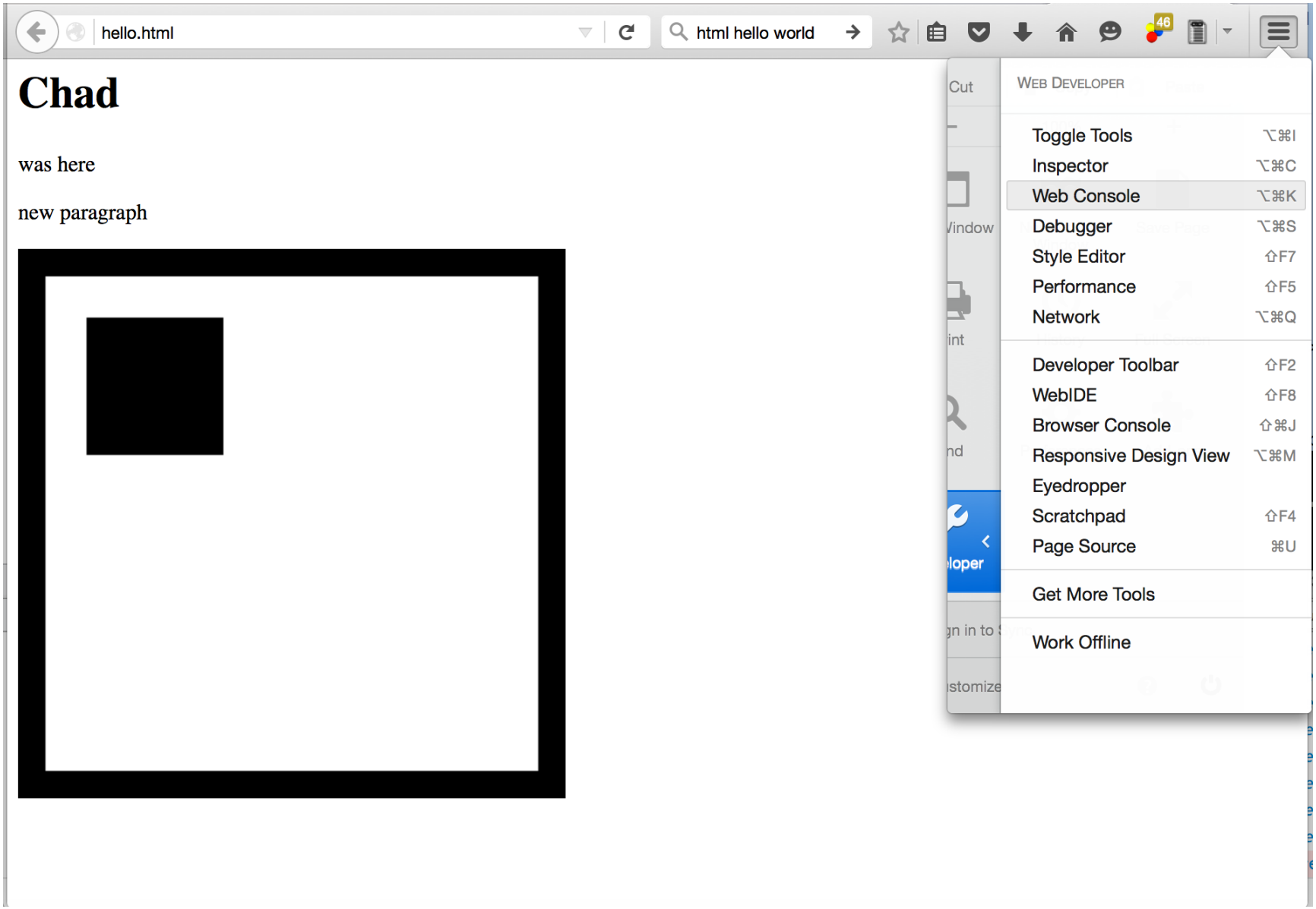


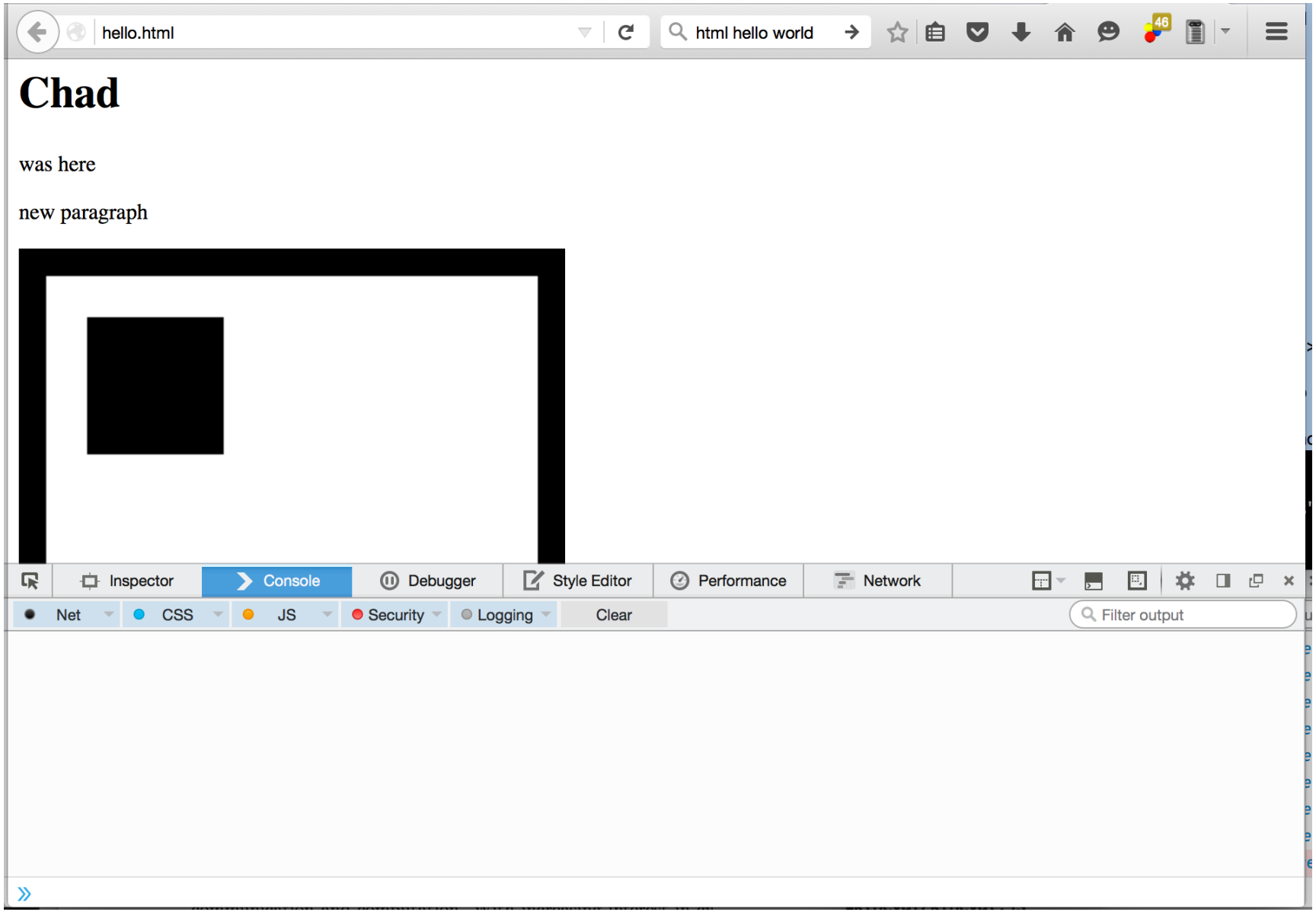

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

<h1>Chad</h1>  <!-- say your name big -->

was here   <!-- say something smaller -->

<p>  <!-- start a new paragraph --> some paragraph text </p>

<!-- create a element for drawing -->
<canvas id="myCanvas" width="400" height="400"></canvas>

<!-- create an element with JavaScript code to execute -->
<script>
   // this is a comment in JavaScript.  it is ignored

   // grab the canvas HTML element for drawing
   var canvas = document.getElementById("myCanvas");

   // grab the canvas drawing context
   var ctx = canvas.getContext("2d");

   // draw rectangles
   ctx.fillRect(50,50,100,100);
   ctx.fillRect(0,0,20,400);
   ctx.fillRect(0,0,400,20);
   ctx.fillRect(0,380,400,20);
   ctx.fillRect(380,0,20,400);
</script>
</body> </html>
```

Using the browser console

The Browser Console

- Provided by the web browser to:
  - Log information associated with a web page: errors, warnings, explicit logging messages, network requests, security errors, etc.
  - Live interaction with a web page by executing JavaScript expressions in the context of the page


hello.html
html hello world
Chad
was here
new paragraph
Cut
Window
int
nd
loper
gn in to
stomize
WEB DEVELOPER
Toggle Tools ⌥⌘I
Inspector ⌥⌘C
Web Console ⌥⌘K
Debugger ⌥⌘S
Style Editor ⇧F7
Performance ⇧F5
Network ⌥⌘Q
Developer Toolbar ⇧F2
WebIDE ⇧F8
Browser Console ⇧⌘J
Responsive Design View ⌥⌘M
Eyedropper
Scratchpad ⇧F4
Page Source ⌘U
Get More Tools
Work Offline


hello.html
html hello world
Chad
was here
new paragraph
Inspector
Console
Debugger
Style Editor
Performance
Network
Net
CSS
JS
Security
Logging
Clear
Filter output

file:///Users/logan/git_tmp/autorob.github.io/examples/hello.ht

html hello world

Chad

was here

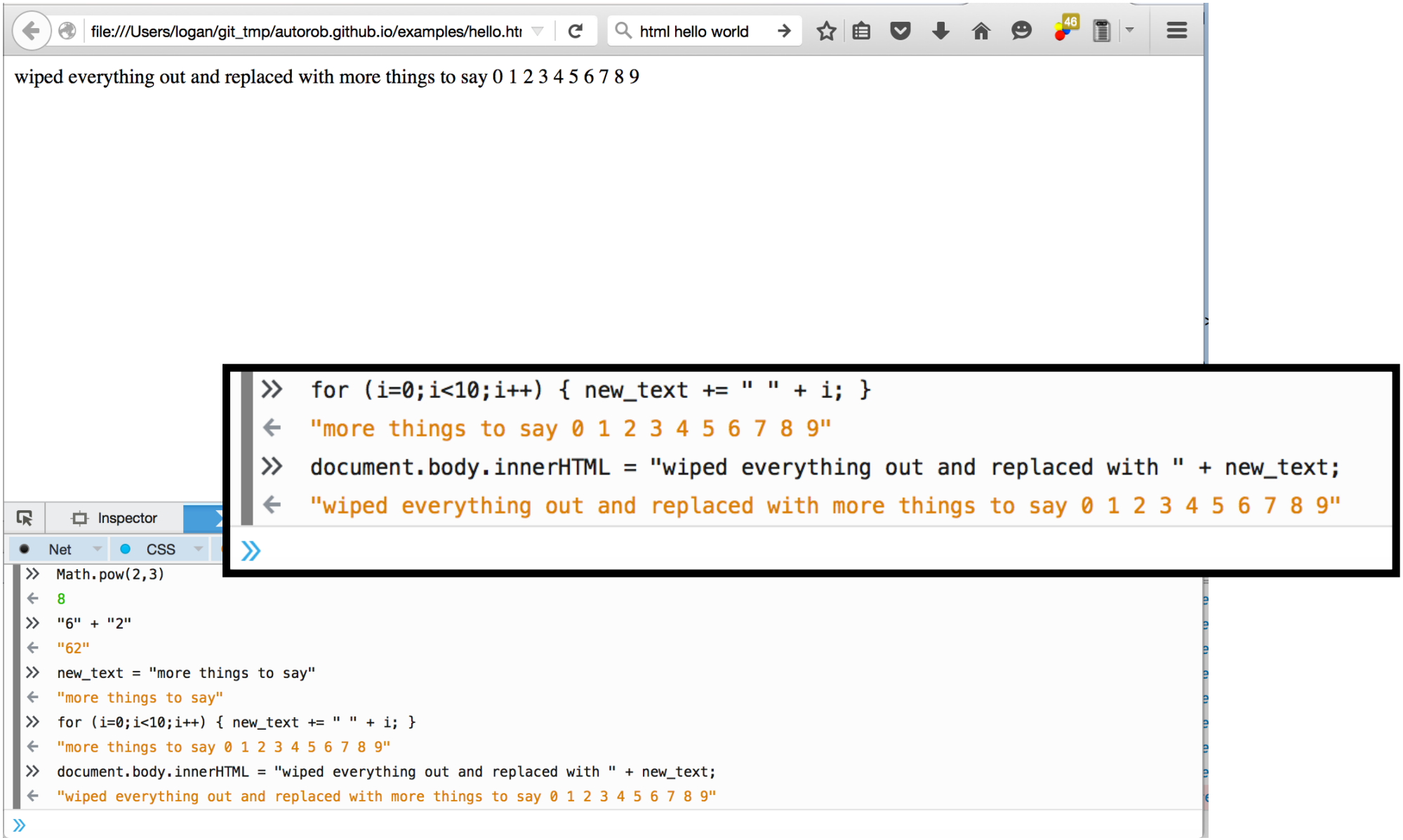
some paragraph text

Inspector | Console | Debugger | Style Editor | Per

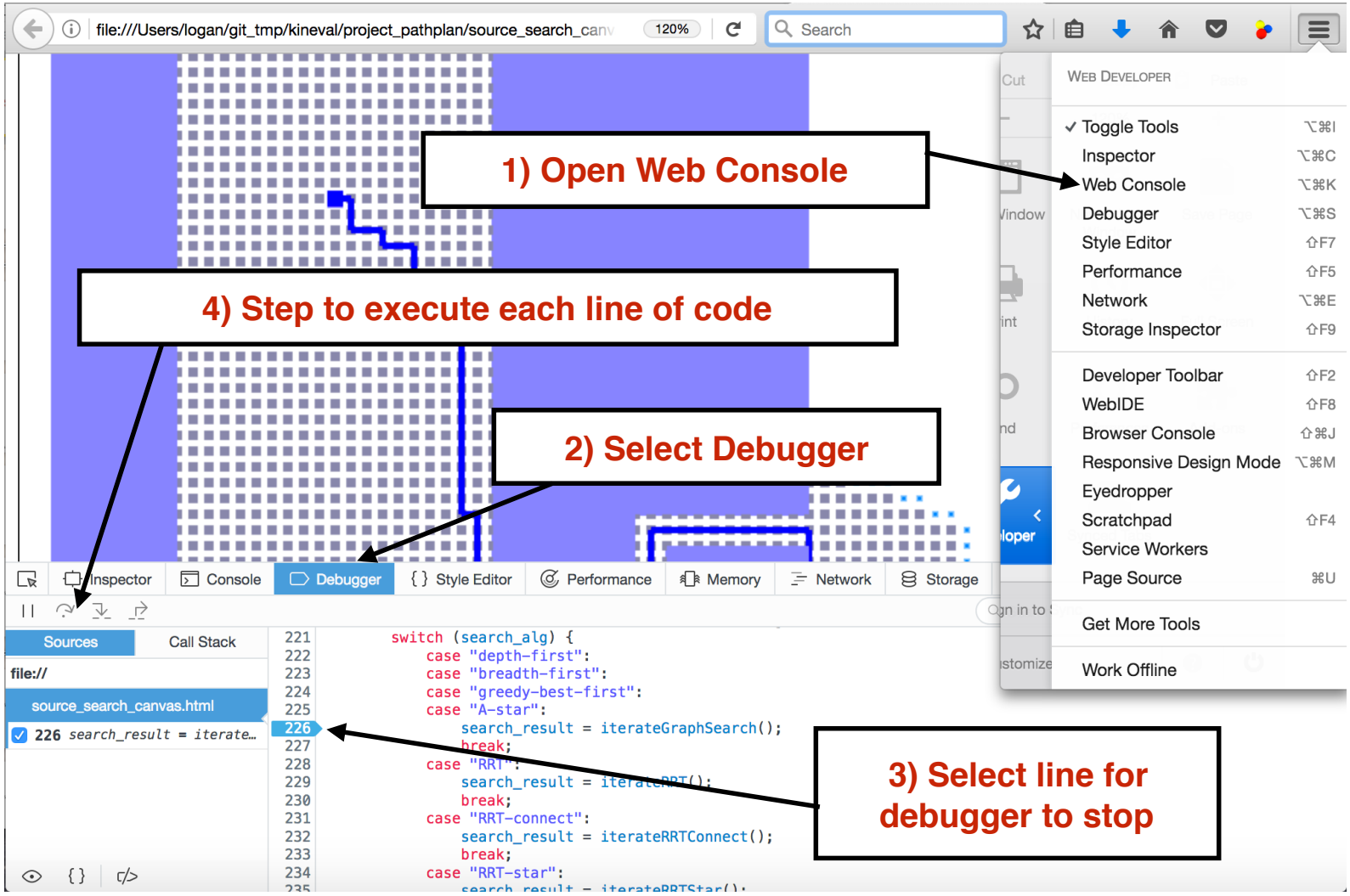
Net | CSS | JS | Security | Logging | Clear

```
» 6+2
← 8
» Math.pow(2,3)
← 8
» "6" + "2"
← "62"
» new_text = "more things to say"
← "more things to say"
» for (i=0;i<10;i++) { new_text += " " + i; }
← "more things to say 0 1 2 3 4 5 6 7 8 9"
» expressions entered here are evaluated live
```


```
» 6+2
← 8
» Math.pow(2,3)
← 8
» "6" + "2"
← "62"
» new_text = "more things to say"
← "more things to say"
» for (i=0;i<10;i++) { new_text += " " + i; }
← "more things to say 0 1 2 3 4 5 6 7 8 9"
» expressions entered here are evaluated live
```


file:///Users/logan/git_tmp/autorob.github.io/examples/hello.ht
html hello world
wiped everything out and replaced with more things to say 0 1 2 3 4 5 6 7 8 9
for (i=0;i<10;i++) { new_text += " " + i; }
"more things to say 0 1 2 3 4 5 6 7 8 9"
document.body.innerHTML = "wiped everything out and replaced with " + new_text;
"wiped everything out and replaced with more things to say 0 1 2 3 4 5 6 7 8 9"
Inspector
Net
CSS
Math.pow(2,3)
8
"6" + "2"
"62"
new_text = "more things to say"
"more things to say"
for (i=0;i<10;i++) { new_text += " " + i; }
"more things to say 0 1 2 3 4 5 6 7 8 9"
document.body.innerHTML = "wiped everything out and replaced with " + new_text;
"wiped everything out and replaced with more things to say 0 1 2 3 4 5 6 7 8 9"

Using the browser debugger

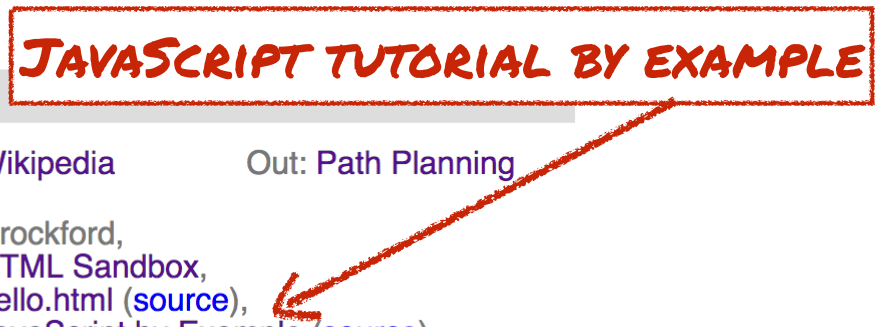

file:///Users/logan/git_tmp/kineval/project_pathplan/source_search_can
120%
Search
1) Open Web Console
4) Step to execute each line of code
2) Select Debugger
3) Select line for debugger to stop
Web Developer
Toggle Tools
Inspector
Web Console
Debugger
Style Editor
Performance
Network
Storage Inspector
Developer Toolbar
WebIDE
Browser Console
Responsive Design Mode
Eyedropper
Scratchpad
Service Workers
Page Source
Get More Tools
Work Offline
Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Sources
Call Stack
file://
source_search_canvas.html
226 search_result = iterate…
switch (search_alg) {
case "depth-first":
case "breadth-first":
case "greedy-best-first":
case "A-star":
search_result = iterateGraphSearch();
break;
case "RRT":
break;
case "RRT-connect":
search_result = iterateRRTConnect();
break;
case "RRT-star":

JavaScript:
the quick and dirty

Course Schedule (tentative and subject to change)

Note: Assignment descriptions will have updated assignment due dates. Assignment due dates listed in the schedule are merely a guide.

Date	Topic	Reading	Project
Sep 5	Initialization: Course overview, Robotics roadmap, Path planning quick start What is a robot?: Brief history and definitions for robotics	Spong Ch.1 Corke Ch.1	Setup git repository
	Week 2		
Sep 10	Path Planning: DFS, BFS, A-star, Greedy best first	Wikipedia	Out: Path Planning
	JavaScript and git tutorial: Heap sort example	Crockford, HTML Sandbox, hello.html (source), JavaScript by Example (source), hello_anim (source), hello_anim_text (source)	
Sep 12	Pendulum Simulation and Numerical Integration: Lagrangian equation(s) of motion, Initial value problem, Explicit integrators: Euler, Verlet, and Runge-Kutta 4	Euler's Method Verlet Integration, Runge-Kutta; Witkin&Baraff 1998: Dynamics Witkin&Baraff 1998: Integrators	
	Week 3		




autorob.org/examples/js_overview.html
Search
Quick JavaScript Code-by-Example Tutorial
Chad Jenkins (ocj)
Open "View Source" in browser to see this code
autorob.org|autorob.online|autorob.github.io
--> Please review the tutorialJSCoding() function
autorobObject contents:
AutoRob university is Michigan
AutoRob department is EECS
AutoRob course_number is 367-002
AutoRob subject is autonomous_robotics
AutoRob string_containing_the_word_subject is an irrelevant property
AutoRob phoneArray is 8,6,7,5,3,0,ni-i-i-ine
AutoRob instructor is ocj
AutoRob printCourseInfo is function myFunction(inputObject) { // create array that will be returned var outputArray = []; //
Object.keys() method returns an array of top-level keys in an object myObjectKeys = Object.keys(inputObject); // format and output
strings for each key/value element of myObject for (i=0;i

view-source:http://autorob.org/examples/js_over

Search

```
<!--|\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/|
|\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/|
||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/
/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\

    tutorialJSCoding | HTML front page

    Quick JavaScript Code-by-Example Tutorial

    @author ohseejay / https://github.com/ohseejay
                     / https://bitbucket.org/ohseejay

    Chad Jenkins
    Laboratory for Perception RObotics and Grounded REasoning Systems
    University of Michigan

    License: Michigan Honor License

|\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/|
||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/
/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\
\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\-->

<!-- IMPORTANT: Remember to open the console

     For this tutorial, it will be very helpful to open the "console" for
     introspection of variables and other items in the browser run-time
     environment.  At the time this code was written, the console can be
     opened from the keyboard:

     Firefox Web Console: Control-Shift-K or (Mac OS X) Option-Command-K;
     Chrome JavaScript Console: Control-Shift-J or (Mac OS X) Option-Command-J;
     Opera Developer Tools: Control-Shift-I or (Mac OS X) Option-Command-I
     Safari Web Inspector: Option-Command-I (after enabling Develop mode)
     Internet Explorer Developer Tools: F12
-->

<!-- DOCTYPE specifies of the document type as HTML -->
<!DOCTYPE html>

<!-- Start tag for the HTML document -->
```

```
31      Firefox Web Console: Control-Shift-K or (Mac OS X) Option-Command-K;
32      Chrome JavaScript Console: Control-Shift-J or (Mac OS X) Option-Command-J;
33      Opera Developer Tools: Control-Shift-I or (Mac OS X) Option-Command-I
34      Safari Web Inspector: Option-Command-I (after enabling Develop mode)
35      Internet Explorer Developer Tools: F12
36  -->
37
38  <!-- DOCTYPE specifies of the document type as HTML -->
39  <!DOCTYPE html>
40
41  <!-- Start tag for the HTML document -->
42  <html>
43
44      <!-- The onload property will execute upon the browser completely loading
45           the body of the HTML document.  This onload routine will call the
46           function "tutorialJSCoding" that is in the "js_overview.js" file.
47      -->
48      <body onload="tutorialJSCoding()">
49          <p>
50          Quick JavaScript Code-by-Example Tutorial <br>
51          <i><a href="ocj.name">Chad Jenkins (ocj)</a></i>
52          </p>
53
54          <p>
55          <br><br><br>
56          (Remember to open the <a href="https://webmasters.stackexchange.com/questions/8525/how-do-i-open-the-javascript-console-i
57          </p>
58      </body>
59
60      <head>
61          <title>Quick JavaScript Code-by-Example Tutorial</title>
62
63          <!-- the script tag contains JavaScript code that the browser will
64               execute, either as code inside the tag markers or inside the
65               file specified in the src property
66          -->
67          <script type="text/javascript" src="js_overview.js"> </script>
68
69          <!-- this tag removes the annoying error message about garbled text -->
70          <meta charset="UTF-8">
71      </head>
72  </html>
73
74
```

EXAMINE TUTORIALJSCODING() FUNCTION

```
/*|\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\
|\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/|
||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/
/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\

    tutorialJSCoding | JavaScript support functions

    Quick JavaScript Code-by-Example Tutorial

    @author ohseejay / https://github.com/ohseejay
                     / https://bitbucket.org/ohseejay

    Chad Jenkins
    Laboratory for Perception RObotics and Grounded REasoning Systems
    University of Michigan

    License: Michigan Honor License

|\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/|
||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/
/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\
\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/||\/*/


/* Function definition for main JavaScript Tutorial routine, which will be
   invoked once the body of the document "js_overview.html" is loaded by the
   browser.
*/
function tutorialJSCoding() {

    /*** JAVASCRIPT TUTORIAL: Variables and Data structures ***/

    /* JavaScript variables are implicitly defined upon assignment, without
       the need for explicit declaration.  Variables are globally scoped by
       default, unless locally scoped using the "var" reserved word at first
       assignment.  Primitive types for JavaScript variables include number,
       boolean, and string.
    */

    stringGlobal = "this is a global variable";
```

EXAMINE TUTORIALJSCODING() FUNCTION

JavaScript Variables

- JavaScript has primitive data types for Number, String, Boolean, etc.
- Variables become declared when they are first assigned
- Variables are globally scoped by default, unless first used with “var”

```
stringGlobal = "this is a global variable";
var booleanLocal = true;
numberLocal = 20 - 18;
```

EQUAL SIGN (=) WILL ASSIGN A VALUE TO A VARIABLE

JavaScript Data Structures

- Object is the core data type for more complex data structures
- Object is an associative data structure; it stores a collection of "keys" (or "properties") each with an associated "value"

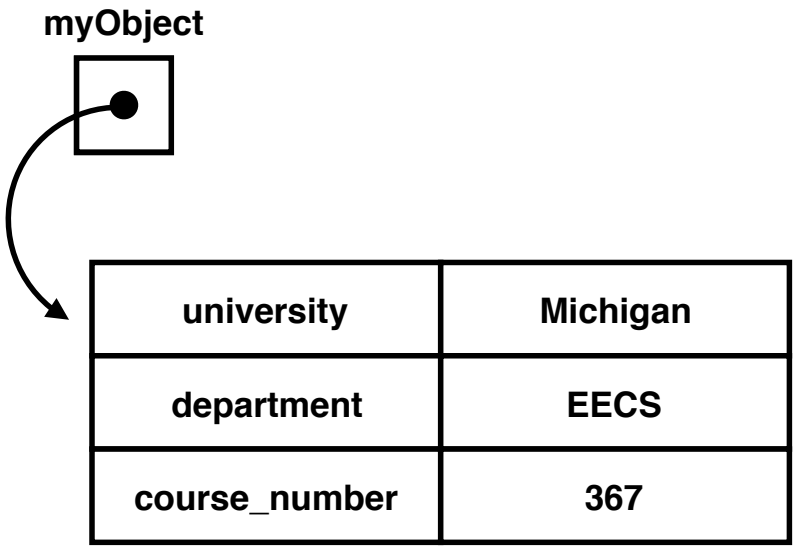
```
myObject = {};  // objects can also be created dynamically

// create object property "university" with an assignment of "Michigan"
myObject.university = "Michigan"; // this variable is of type "string"

// equivalent to myObject.department = "EECS";
myObject["department"] = "EECS"; // this variable is of type "string"

myObject.course_number = 367;  // this variable is of type "number"
```

DOUBLE FORWARD SLASH (//) WILL COMMENT THE REMAINDER OF A LINE

Objects and References

- An object variable is not itself a data structure, but rather a reference to a data structure

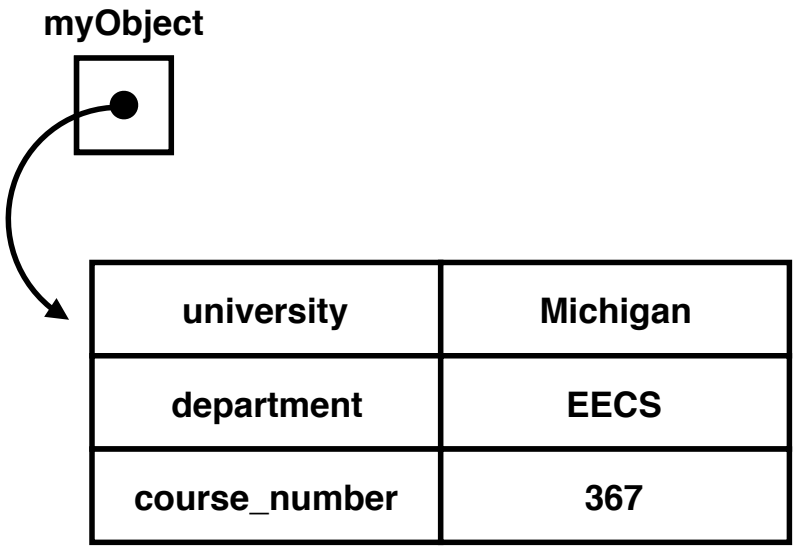
```
myObject = {};  // objects can also be created dynamically

// create object property "university" with an assignment of "Michigan"
myObject.university = "Michigan"; // this variable is of type "string"

// equivalent to myObject.department = "EECS";
myObject["department"] = "EECS"; // this variable is of type "string"

myObject.course_number = 367;  // this variable is of type "number"
```



Objects and References

- An object variable is not itself a data structure, but rather a reference to a data structure

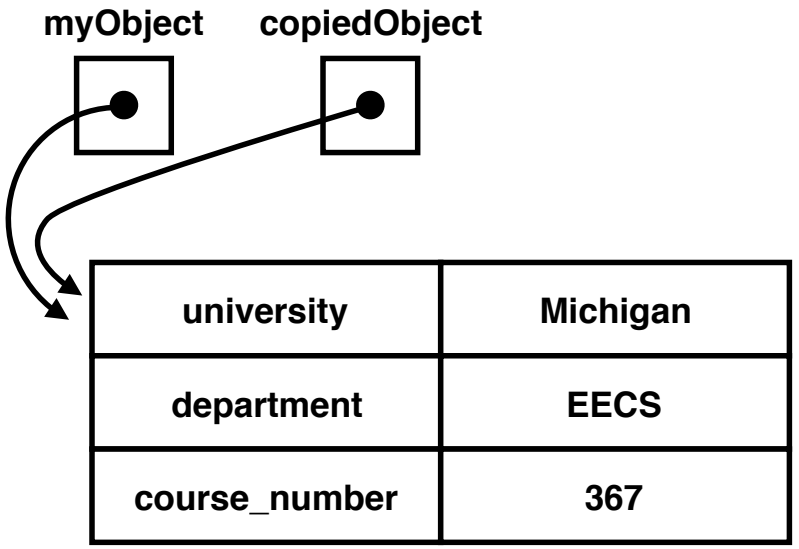
```
myObject = {};  // objects can also be created dynamically

// create object property "university" with an assignment of "Michigan"
myObject.university = "Michigan"; // this variable is of type "string"

// equivalent to myObject.department = "EECS";
myObject["department"] = "EECS"; // this variable is of type "string"

myObject.course_number = 367;  // this variable is of type "number"


copiedObject = myObject;
```



Objects and References

- An object variable is not itself a data structure, but rather a reference to a data structure

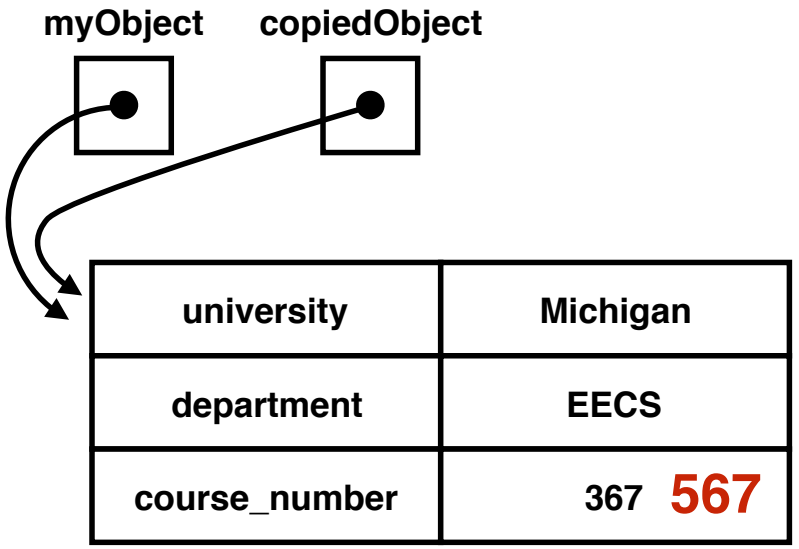
```
myObject = {};  // objects can also be created dynamically

// create object property "university" with an assignment of "Michigan"
myObject.university = "Michigan"; // this variable is of type "string"

// equivalent to myObject.department = "EECS";
myObject["department"] = "EECS"; // this variable is of type "string"

myObject.course_number = 367;  // this variable is of type "number"


copiedObject = myObject;
```



Objects and References

- An object variable is not itself a data structure, but rather a reference to a data structure

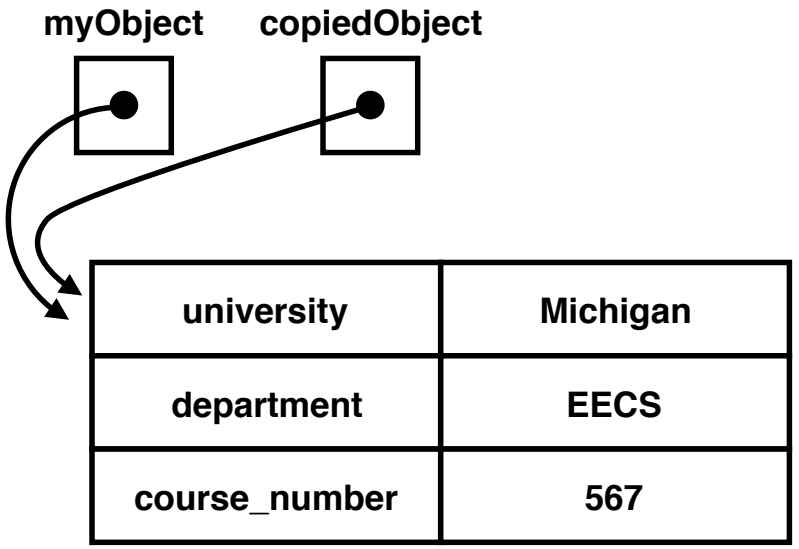
```
myObject = {};  // objects can also be created dynamically

// create object property "university" with an assignment of "Michigan"
myObject.university = "Michigan"; // this variable is of type "string"

// equivalent to myObject.department = "EECS";
myObject["department"] = "EECS"; // this variable is of type "string"

myObject.course_number = 367;  // this variable is of type "number"


copiedObject = myObject;


copiedObject.course_number = 567;
```



Objects and References

- An object variable is not itself a data structure, but rather a reference to a data structure

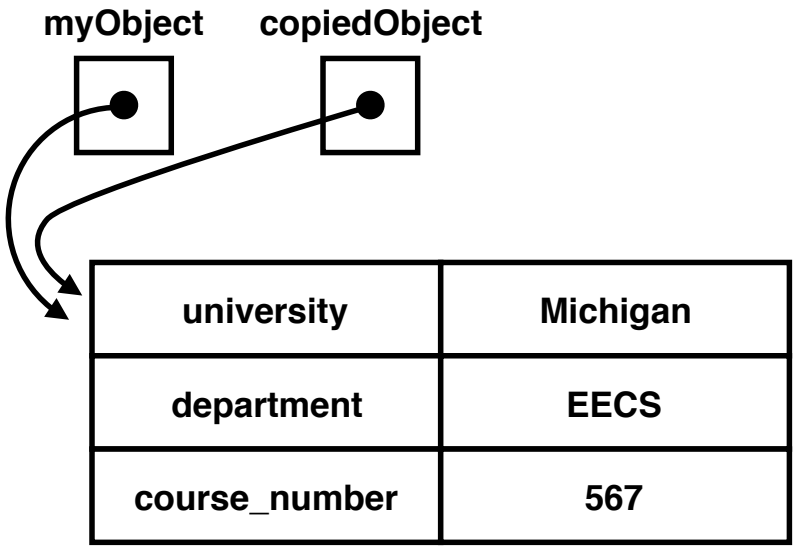
```
myObject = {};  // objects can also be created dynamically

// create object property "university" with an assignment of "Michigan"
myObject.university = "Michigan"; // this variable is of type "string"

// equivalent to myObject.department = "EECS";
myObject["department"] = "EECS"; // this variable is of type "string"

myObject.course_number = 367;  // this variable is of type "number"


copiedObject = myObject;


copiedObject.course_number = 567;
```



Objects and References

- An object variable is not itself a data structure, but rather a reference to a data structure

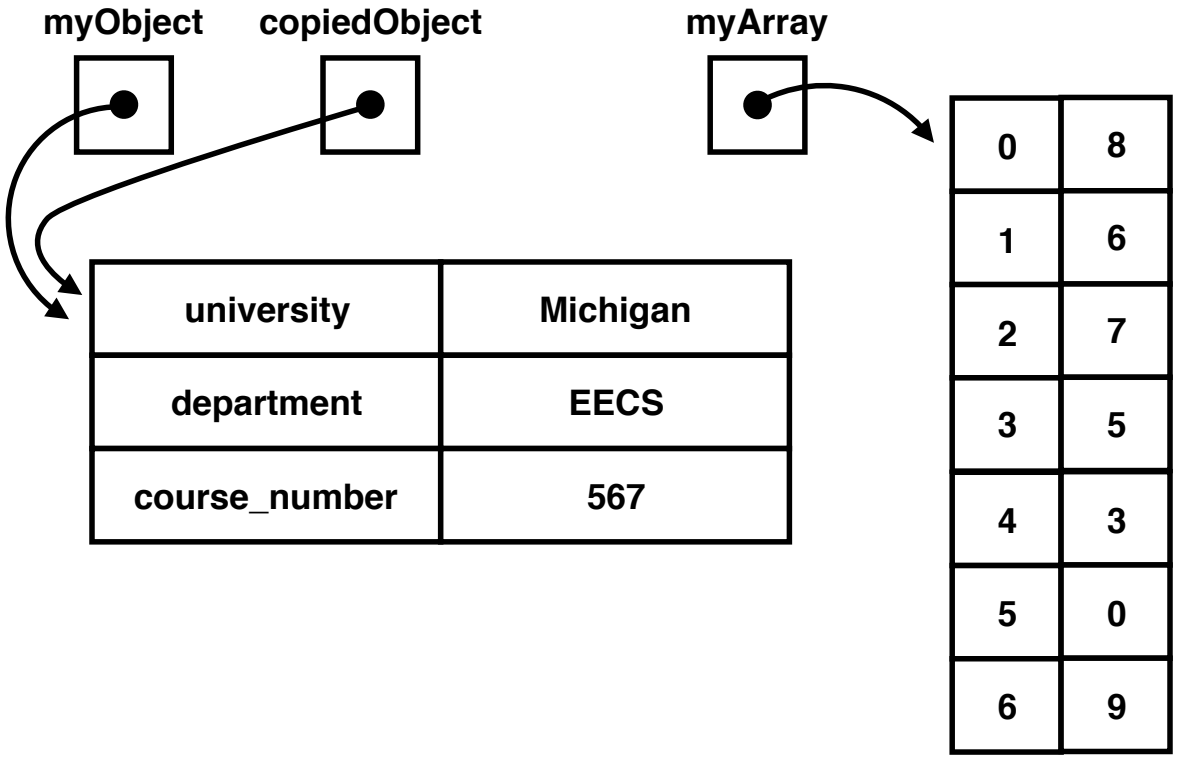
```
myObject = {};  // objects can also be created dynamically

// create object property "university" with an assignment of "Michigan"
myObject.university = "Michigan"; // this variable is of type "string"

// equivalent to myObject.department = "EECS";
myObject["department"] = "EECS"; // this variable is of type "string"

myObject.course_number = 367;  // this variable is of type "number"


copiedObject = myObject;


copiedObject.course_number = 567;


console.log(myObject.course_number); // this will be 567
```



Arrays

- An array is an instance of an object data type with numeric indices

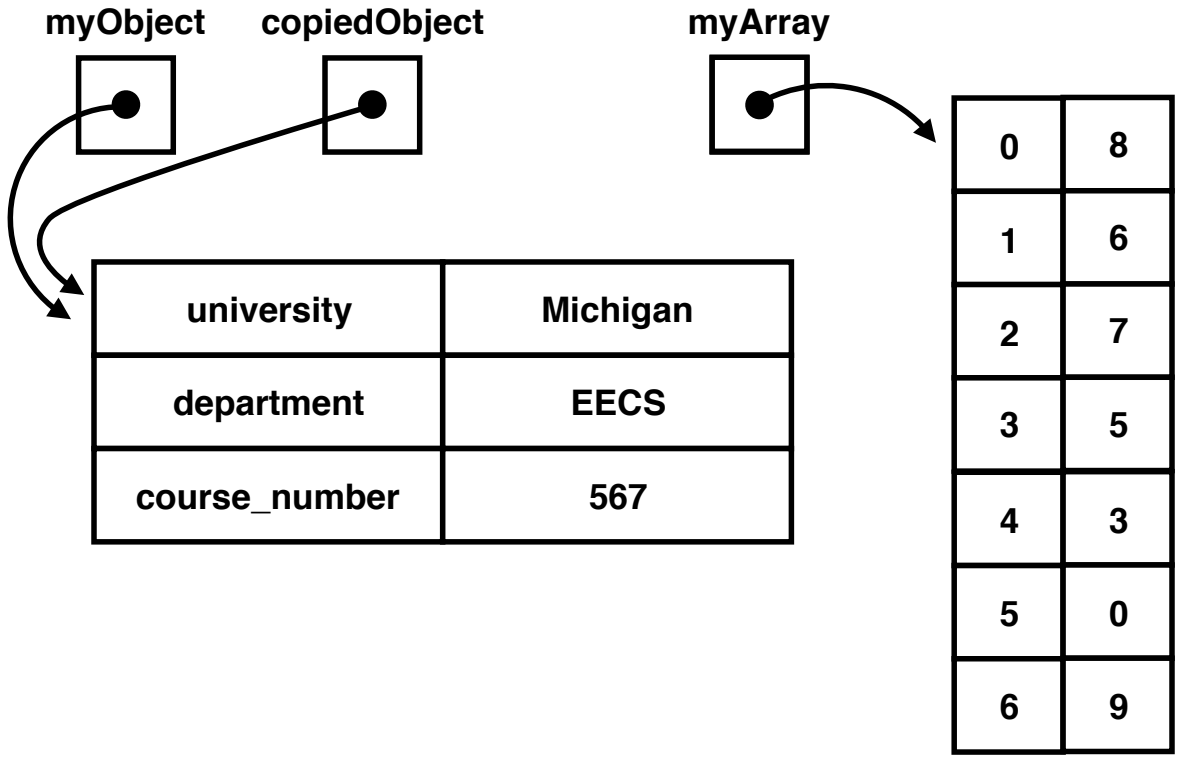
```
myArray = [8,6,7,5,3,0,9];
```



Nesting Objects in Objects

- An object can be used as the value for another object's property

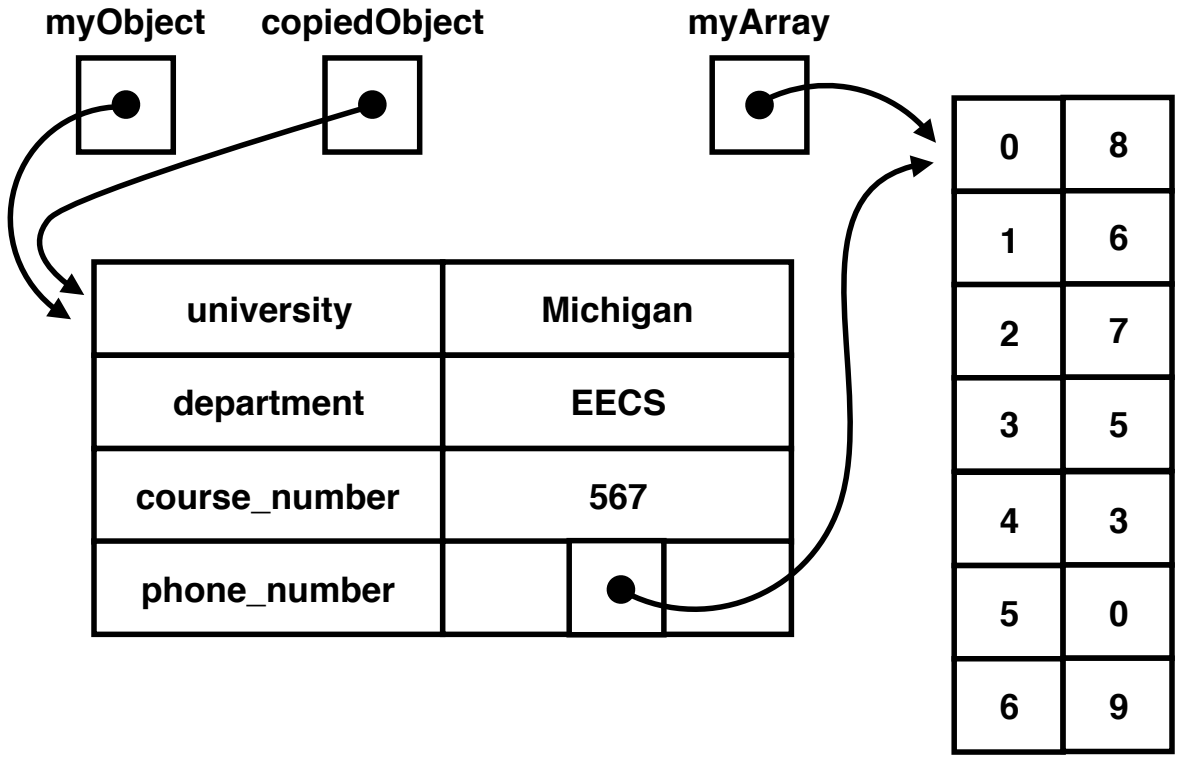
```
myArray = [8,6,7,5,3,0,9];

myObject.phone_number = myArray;
```



Nesting Objects in Objects

- An object can be used as the value for another object's property

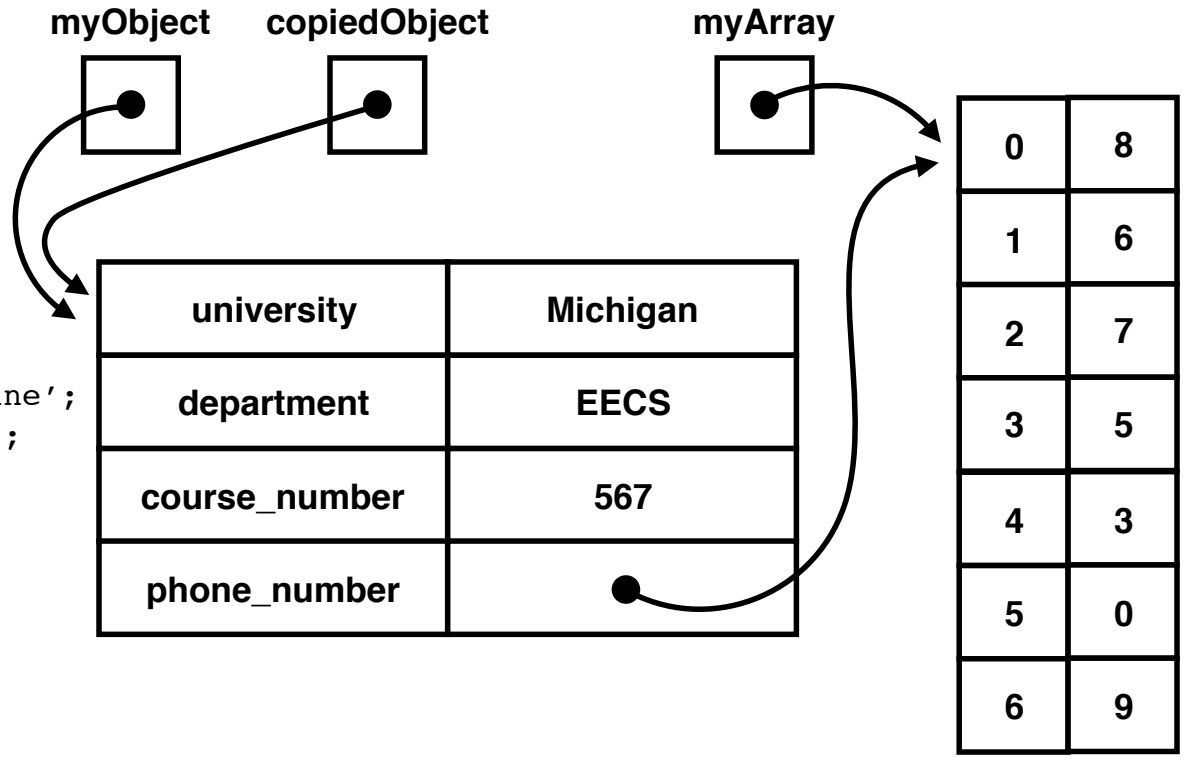
```
myArray = [8,6,7,5,3,0,9];

myObject.phone_number = myArray;
```



Dynamic Typing

- The type of a variable changes upon assignment

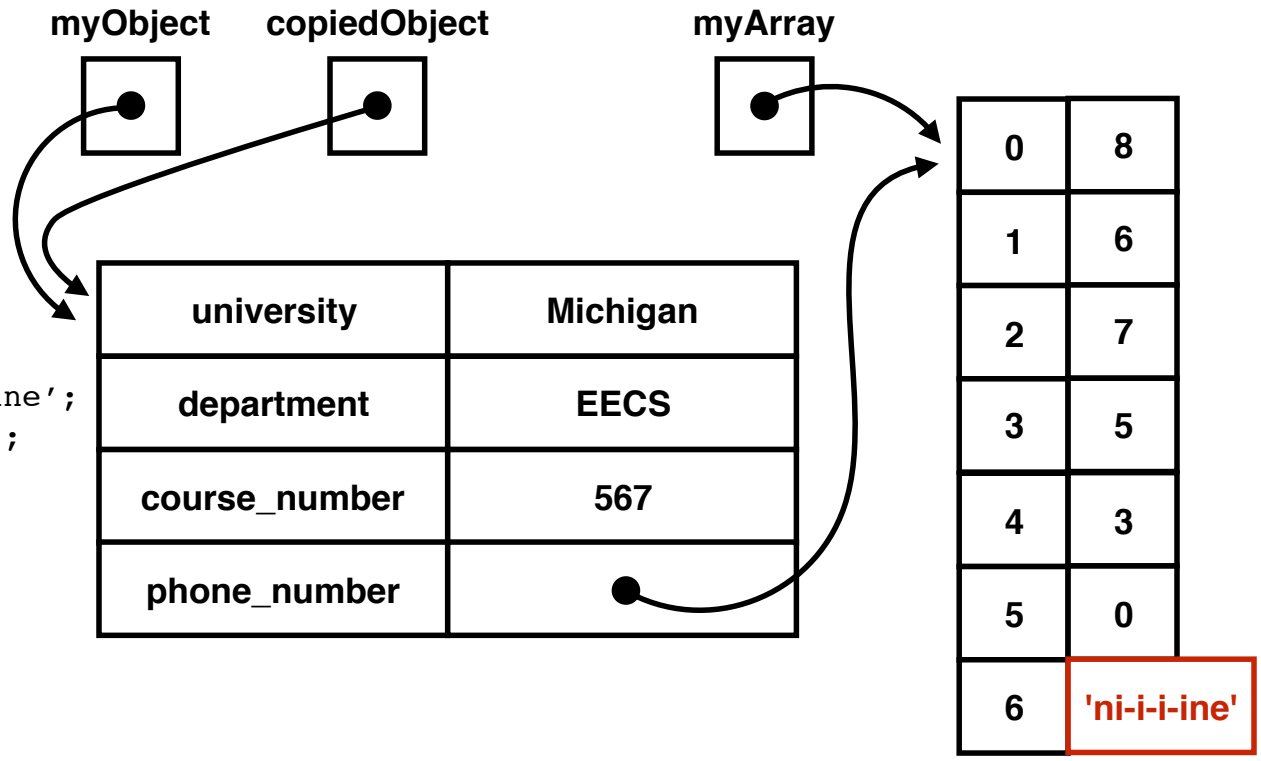
```
myArray = [8,6,7,5,3,0,9];

myObject.phone_number = myArray;

myObject.phone_number[6] = 'ni-i-i-ine';
// same as myArray[6] = 'ni-i-i-ine';
```



Dynamic Typing

- The type of a variable changes upon assignment

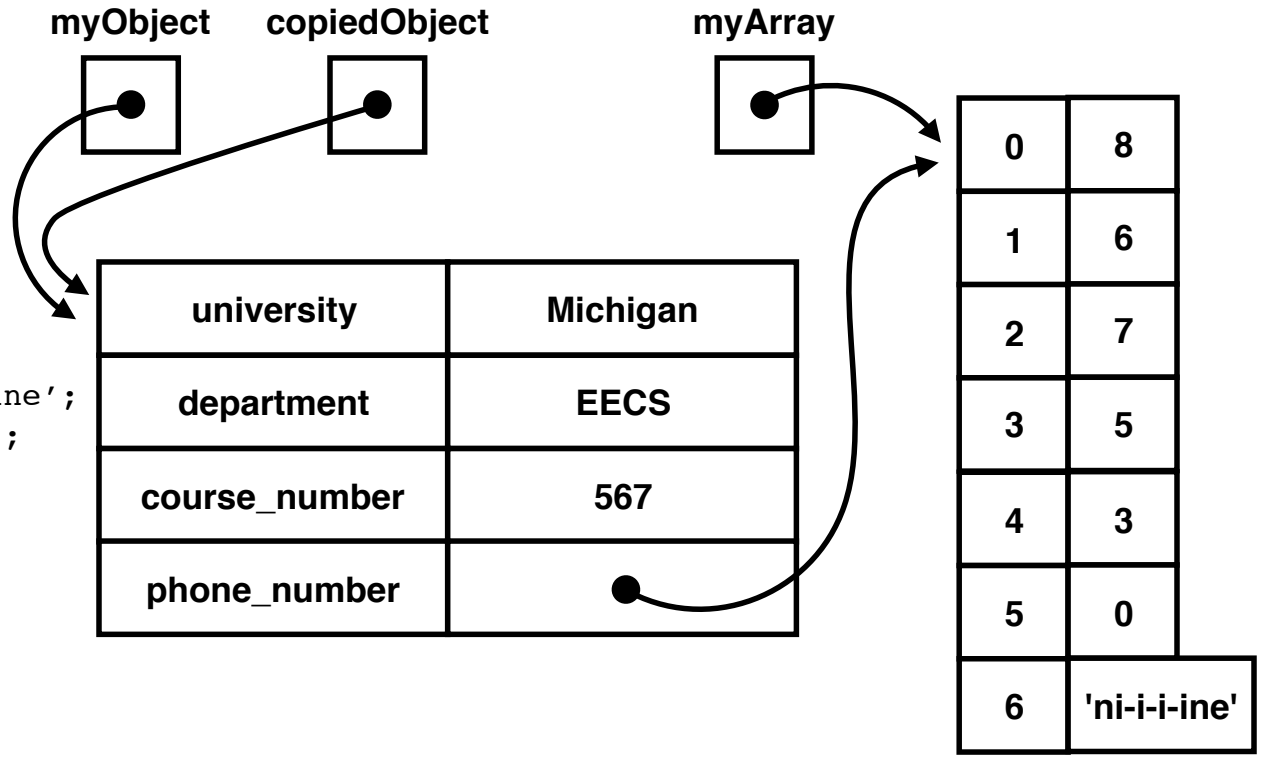
```
myArray = [8,6,7,5,3,0,9];

myObject.phone_number = myArray;

myObject.phone_number[6] = ‘ni-i-i-ine’;
// same as myArray[6] = ‘ni-i-i-ine';
```



Dynamic Typing

- The type of a variable changes upon assignment

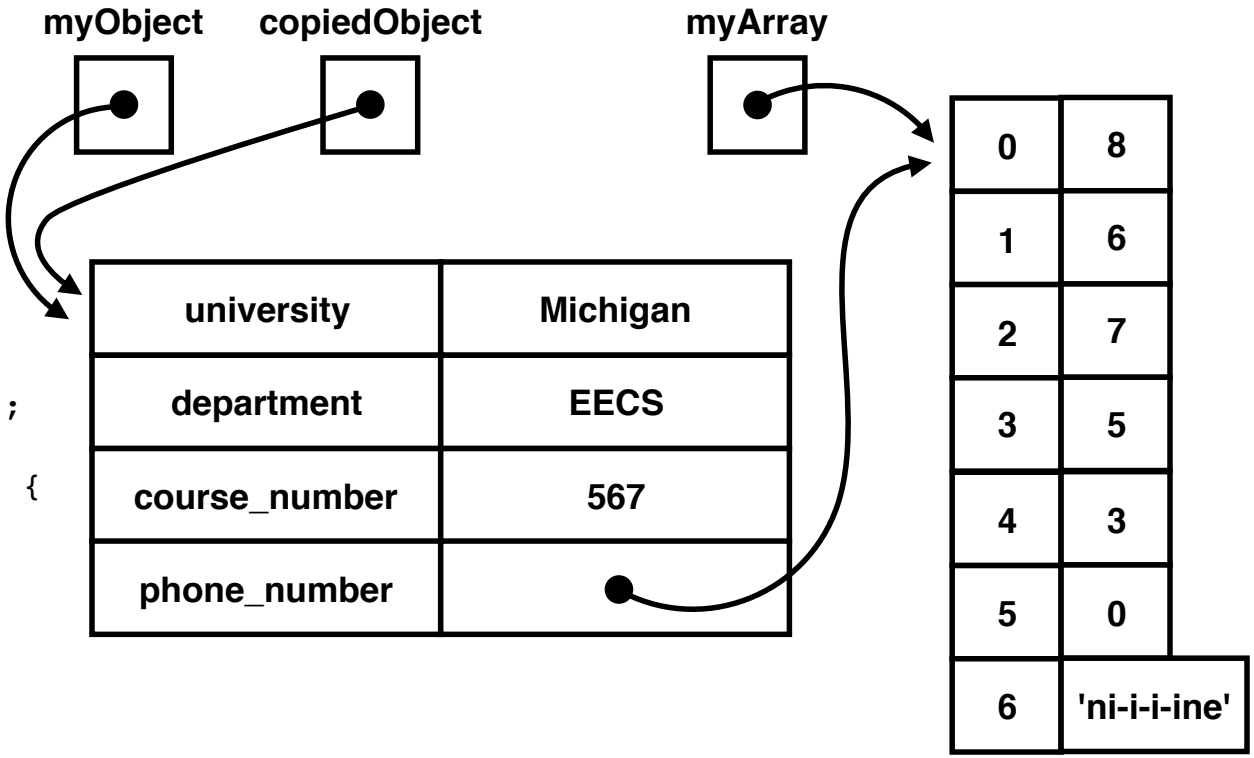
```
myArray = [8,6,7,5,3,0,9];

myObject.phone_number = myArray;

myObject.phone_number[6] = ‘ni-i-i-ine’;
// same as myArray[6] = ‘ni-i-i-ine';
```



Control Statements

- JavaScript supports C-style “if-else” statements and “for” loops

```
// output the phone number to console
var i;  // iterator local variable
for (i=0; i<myArray.length; i++) {
    console.log(myArray[i]);
}


// output the course section to console
if (myObject.course_number ▪▪▪ 367) {
    console.log('undergraduate section');
}
else if (myObject.course_number === 567) {
    console.log('graduate section');
    // message output for myObject
}
else {
    console.log('ROB 510 maybe?');
}
```



Operators

- JavaScript supports C-style operators with order of precedence
  - grouping: `/*` open comment; `*/` close comment; `//` comment to end of line; `(` open parenthesis; `)` close parenthesis
  - increment/decrement: `++` increment variable; `--` decrement variable
  - arithmetic: `*` multiplication; `/` division; `%` modulus; `+` addition/concatenation; `-` subtraction
  - comparison: `<` less than; `<=` less than or equal; `>` greater than; `>=` greater than or equal; `==` equality; `!=` inequality; `===` strict equality; `!===` strict inequality; `&&` logical AND; `||` logical OR
  - assignment: `=` assignment; `+=` add to variable; `*=` multiply to variable

PLUS (+) IS OVERLOADED TO ADD NUMBERS AND CONCATENATE STRINGS

STRICT EQUALITY (===) WILL NOT ATTEMPT TO MATCH TYPES

Functions

- A function is an object type that modularly executes a set of statements with given parameters and (optionally) returns a variable.
- Unlike C, JavaScript functions do not declare a return type

```
// a simple function declaration that returns the sum of two given numbers
function sum(a,b) {
   return a + b;
}

// at some later point in the execution of the code . . .

// function call to add two numbers
sumNumber = sum(3,67);  // 70

// function call to concatenate two strings
sumString = sum("3","67");  // "367"
```

Recursion

- JavaScript supports recursion (i,e., a function calling itself)
- Consider factorial example: *6! = 6 * 5 * 4 * 3 * 2 * 1 = 720*

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

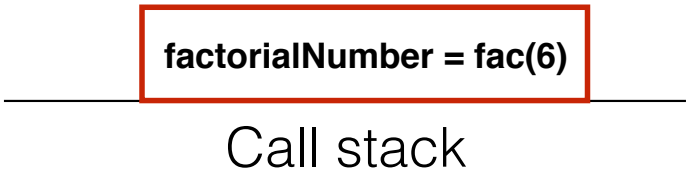
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

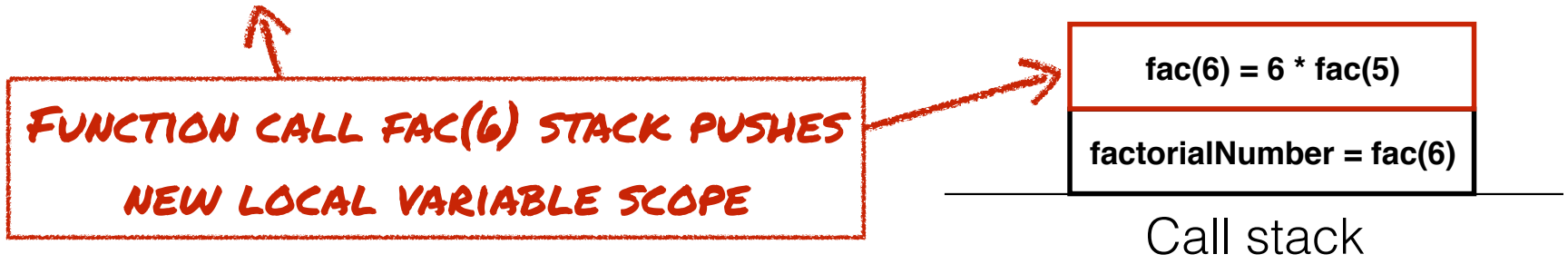
FUNCTION CALL FAC(6) STACK PUSHES NEW LOCAL VARIABLE SCOPE



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

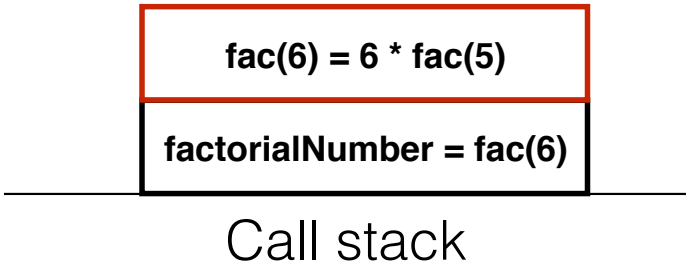


Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

RECURSIVE CALL FAC(5) PUSHES NEW LOCAL VARIABLE SCOPE

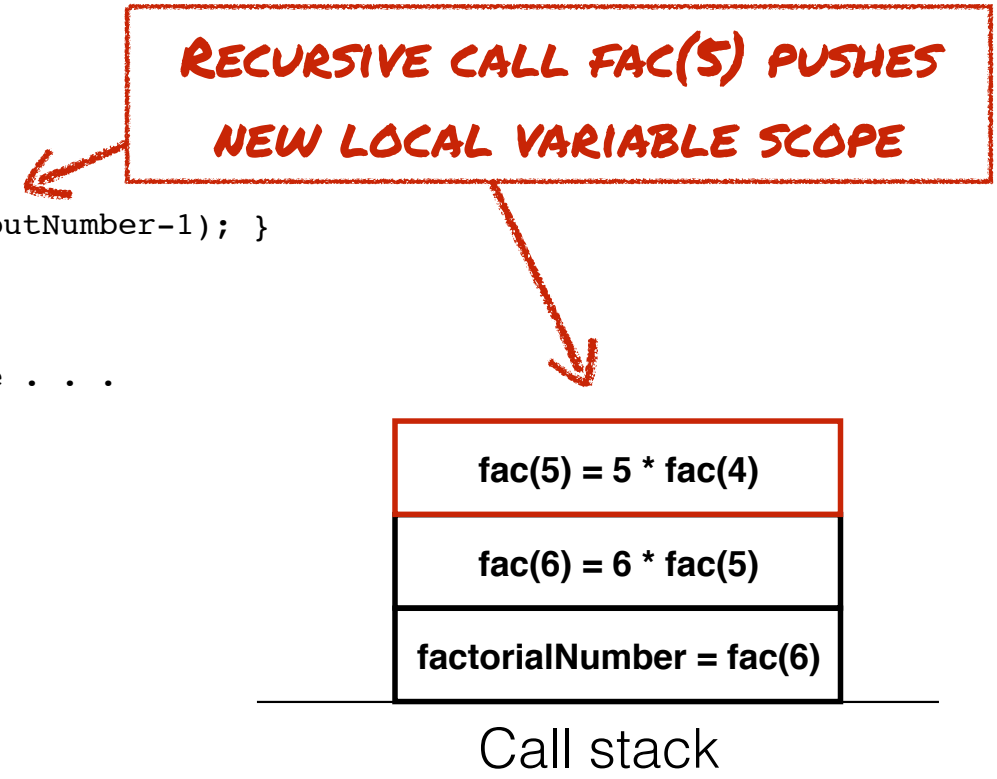
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

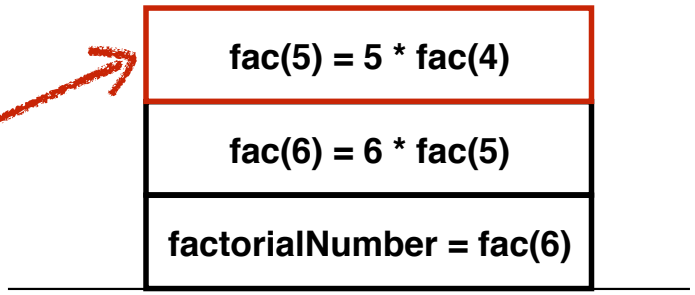


Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

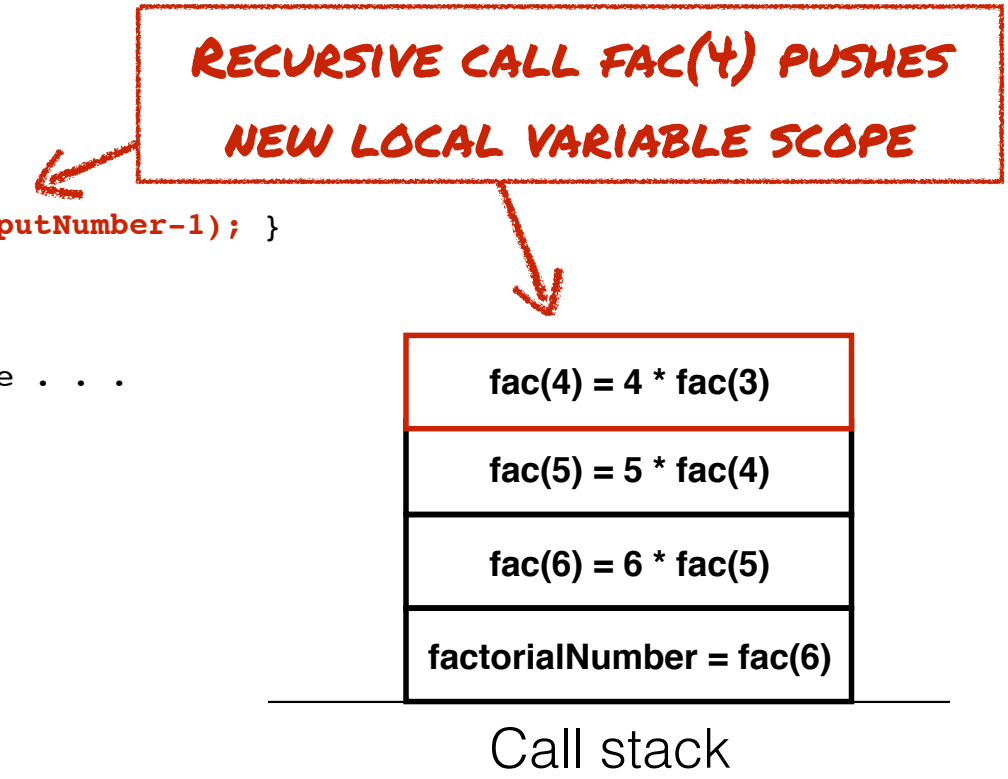
inputNumber variable in fac(5) is different variable than inputNumber in fac(6)



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

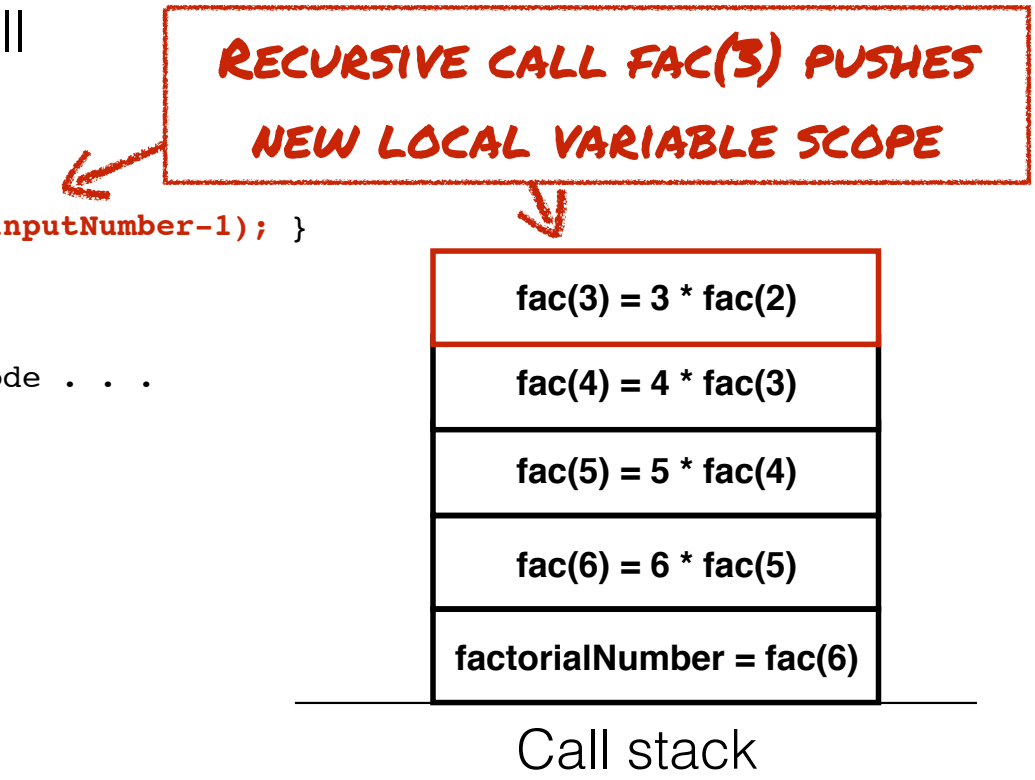
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

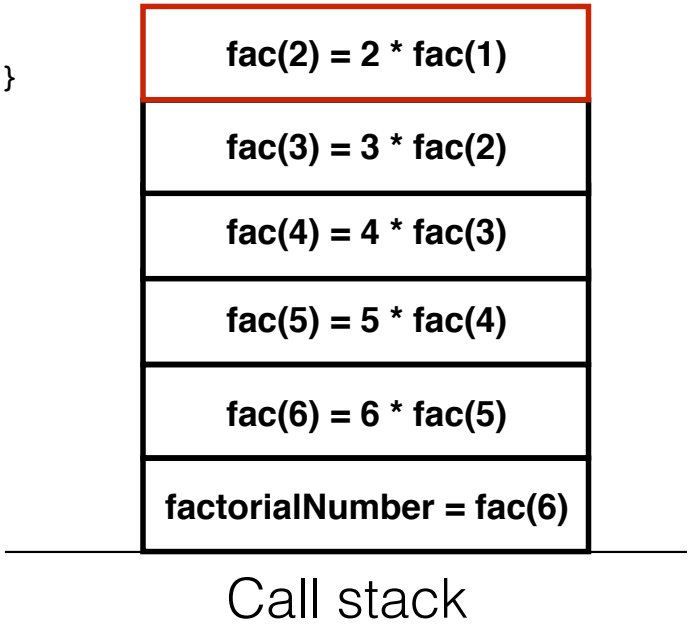
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

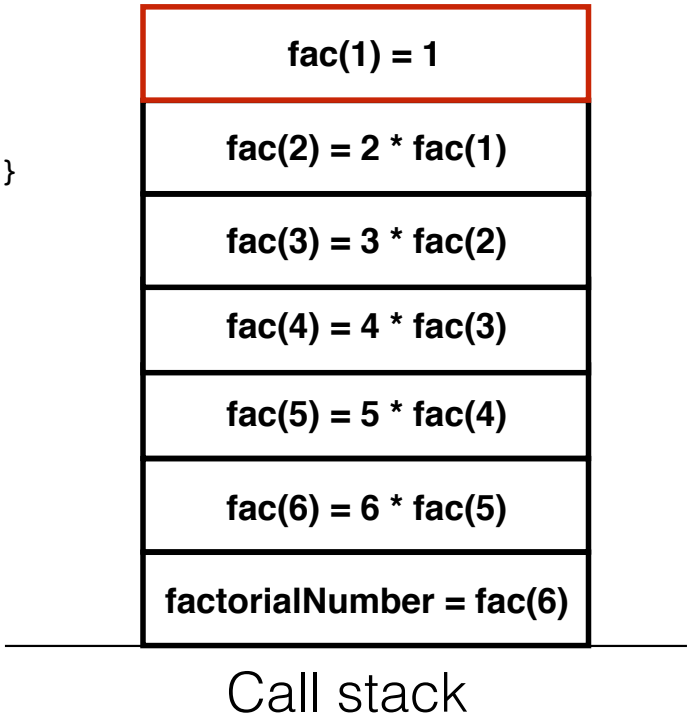


Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

RECURSIVE CALLS STOP WHEN BASE CONDITION ENCOUNTERED

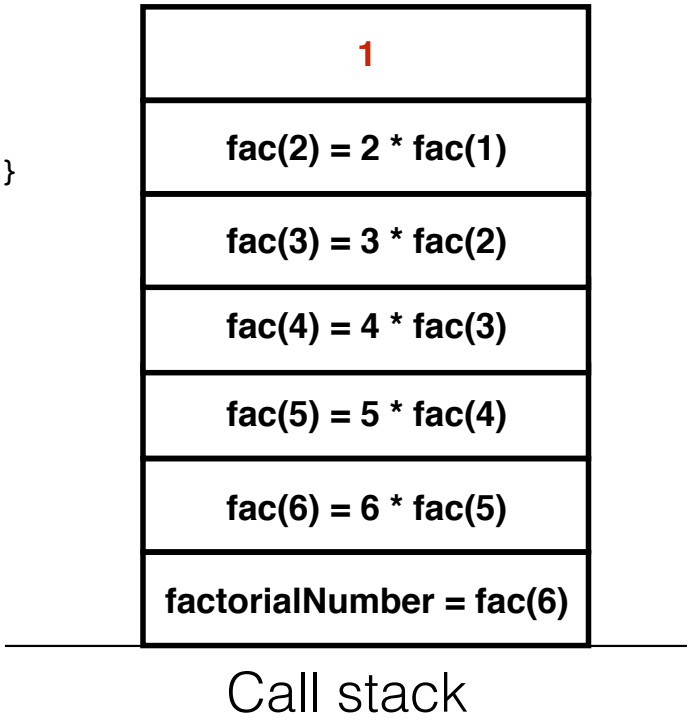


Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

Stack pop from call stack returns a constant value to calling function

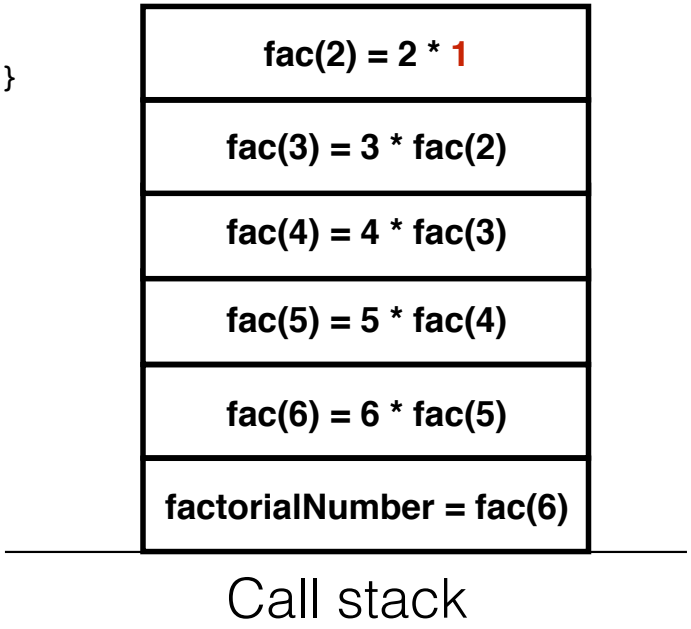


Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

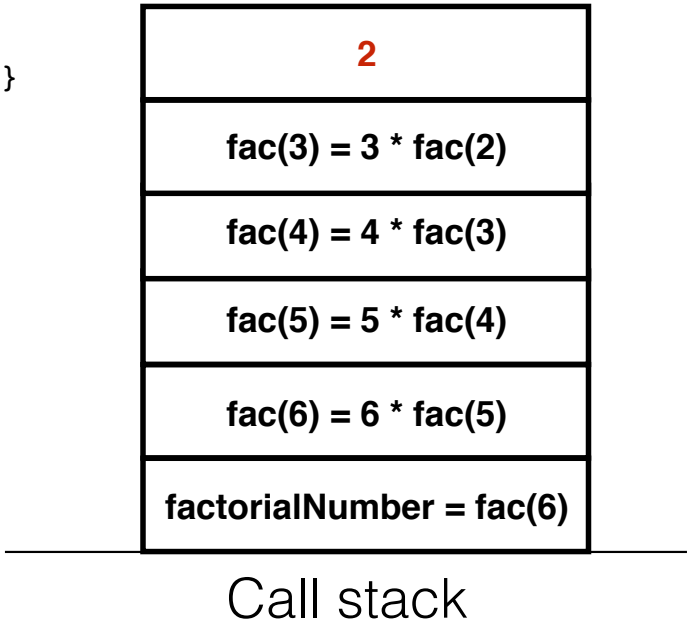
STACK POP FROM CALL STACK RETURNS A CONSTANT VALUE TO CALLING FUNCTION



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

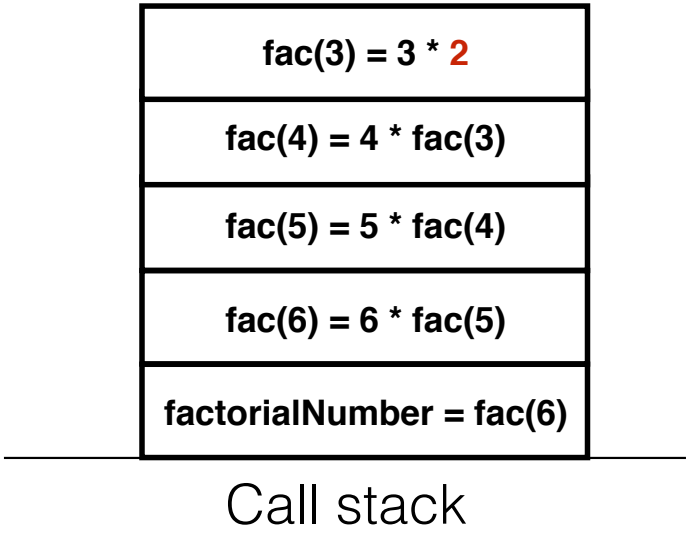
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

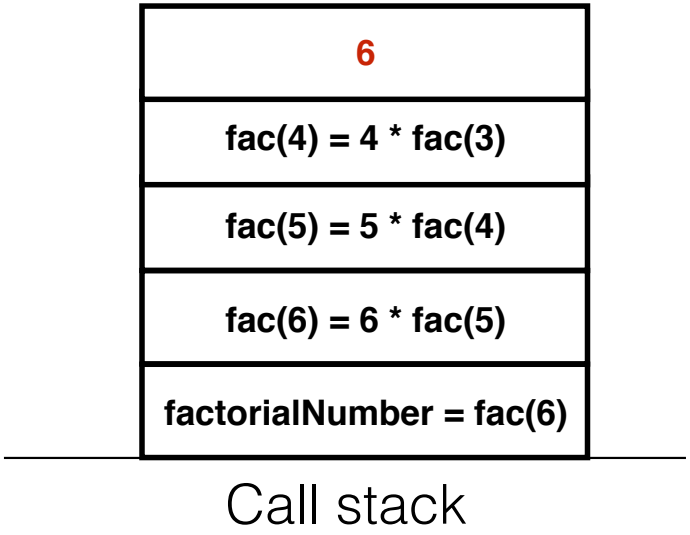
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

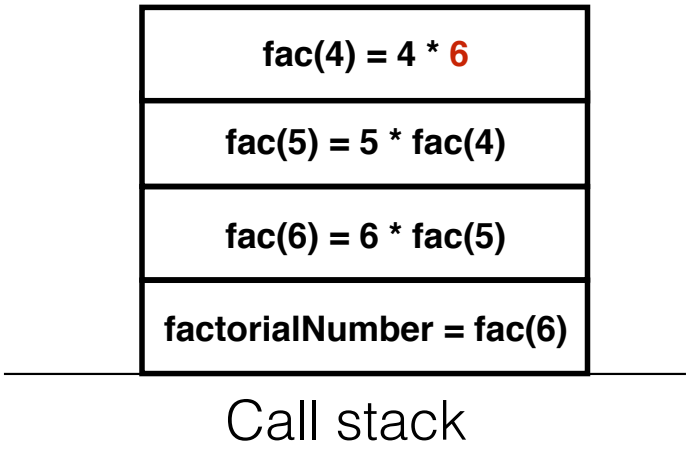
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

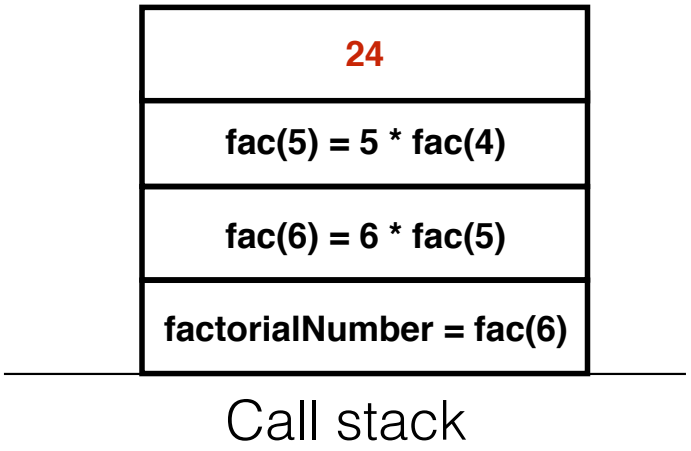
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

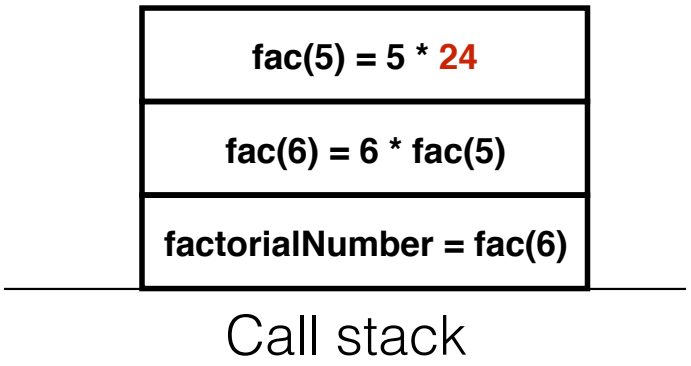
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

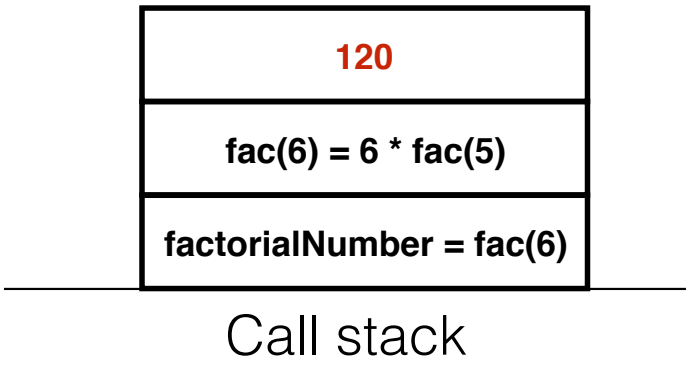
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

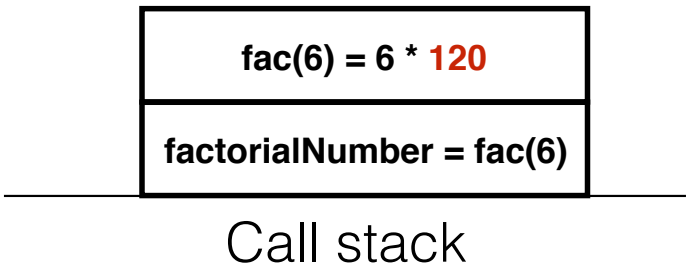
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

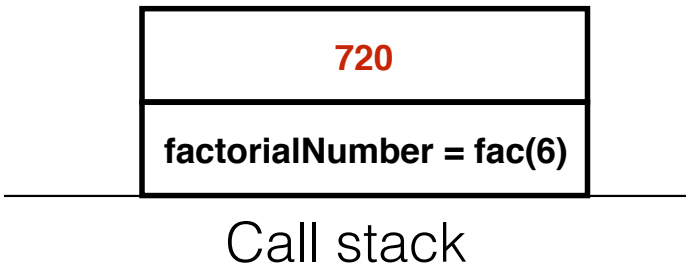
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

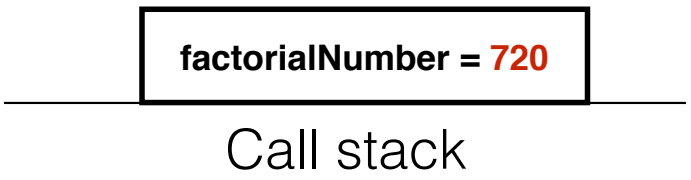
```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```



Function call stack

- JavaScript maintains a call stack that stores the state of variables scoped local to each function call

```
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  //base case
}

// at some later point in the execution of the code . . .

factorialNumber = fac(6);  // 720
```

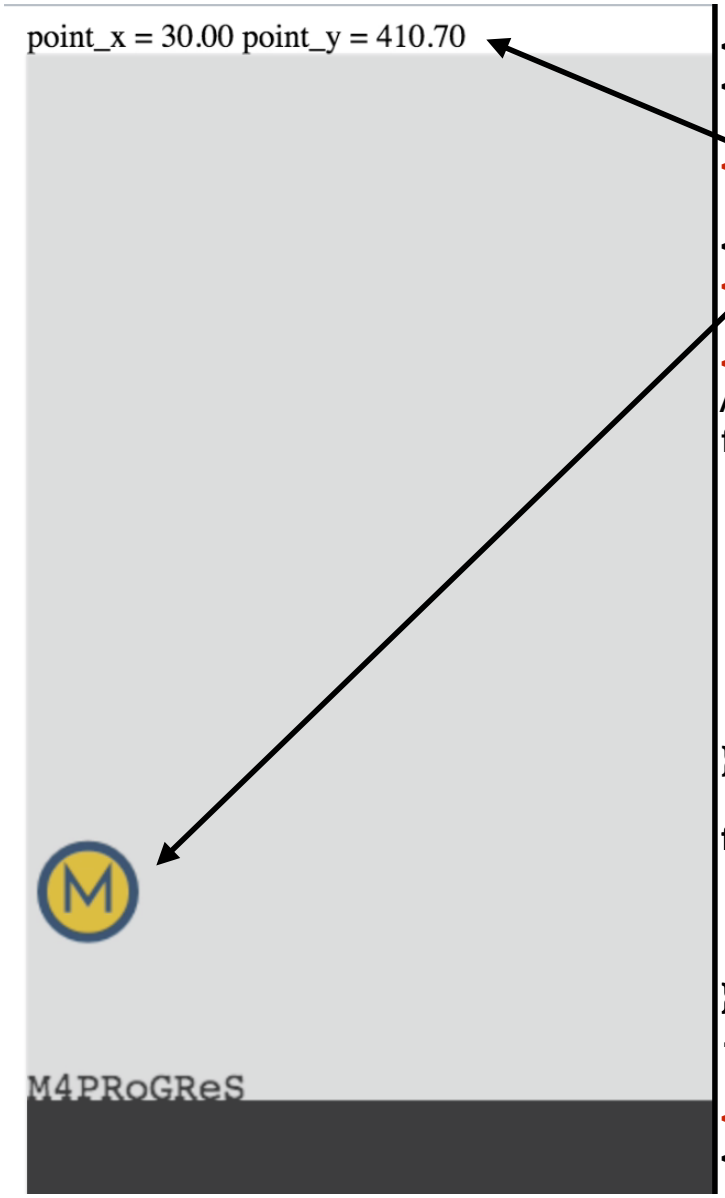


Let’s try an animation example

point_x = 30.00 point_y = 410.70

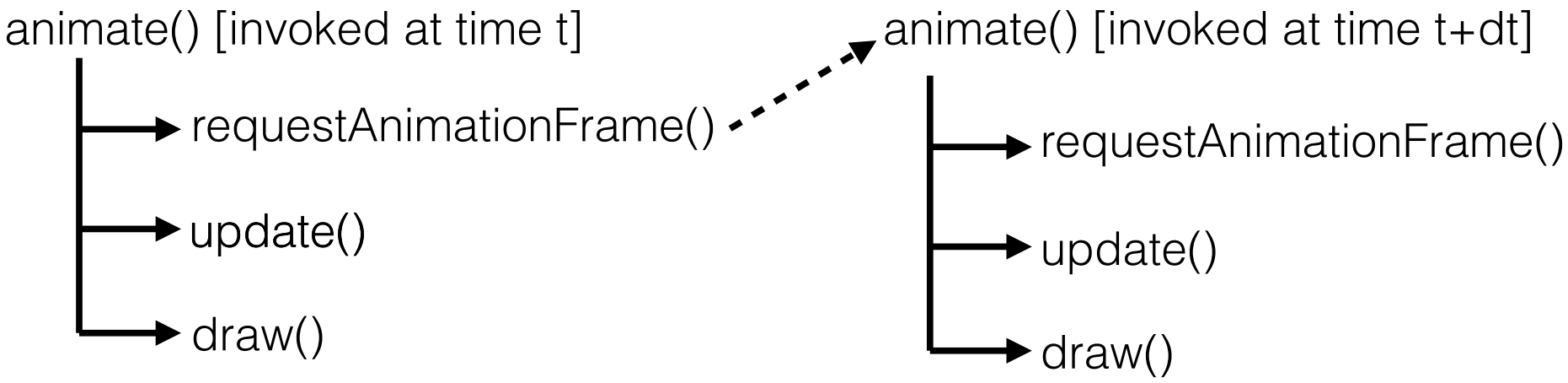
hello_anim example

- http://autorob.github.io/examples/hello_anim.html



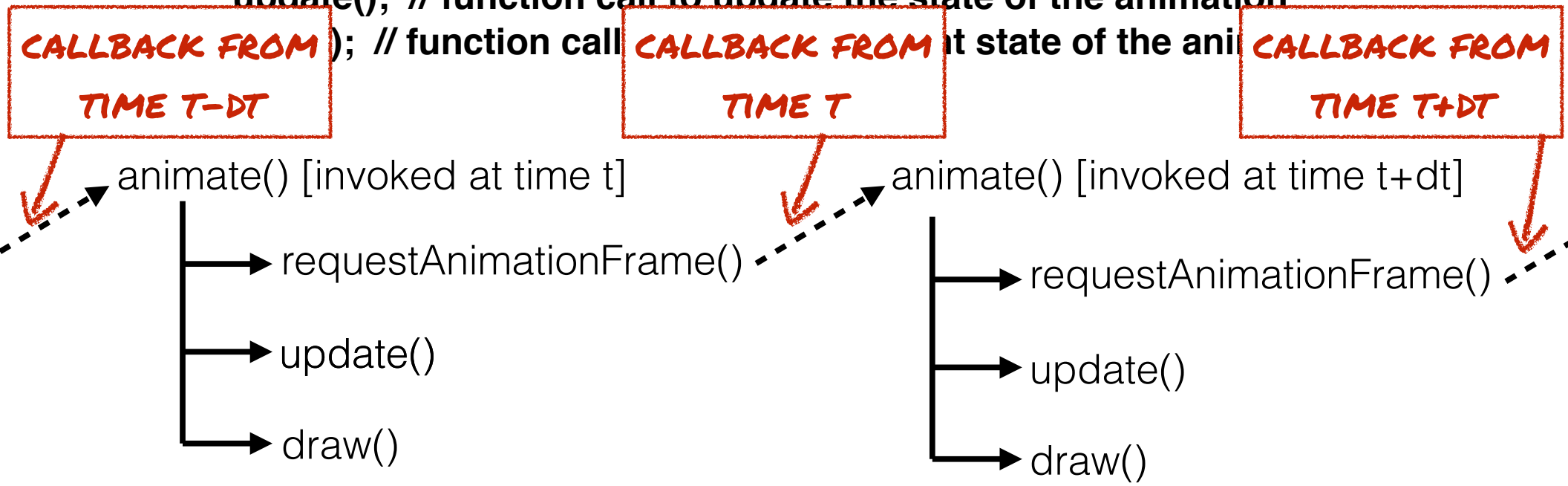

```
<html> <body onload=init()>
<!-- init function will be called when body loaded -->

<div id="text_output"> going to put some text here </div>

<!-- create a element for drawing -->
<canvas id="draw_canvas" width=1000 height="400"></canvas>

<script>
// define a function for initialization as: function name_of_function { function_code }
function init() {

    // create a JavaScript object named "point" with two attributes
    //   specifying the horizontal and vertical location of the circle
    point = {x: 50, y: 50}

    // function call to start the animation loop
    animate();
}

function animate() {
   requestAnimationFrame(animate);  // requests next time step
   update();  // function call to update the state of the animation
   draw();  // function call to draw the current state of the animation
}
...

</script>
</body> </html>
```

```
function animate() {
  requestAnimationFrame(animate);  // requests next time step
  update();  // function call to update the state of the animation
  draw();  // function call to draw the current state of the animation
}
```



requestAnimationFrame() will have browser call animate() again.
IMPORTANT to avoid code that blocks in animate()

```
function animate() {
   requestAnimationFrame(animate);  // requests next time step
   update();  // function call to update the state of the animation
   );  // function call                t state of the ani
```



requestAnimationFrame() will have browser call animate() again.
IMPORTANT to avoid code that blocks in animate()

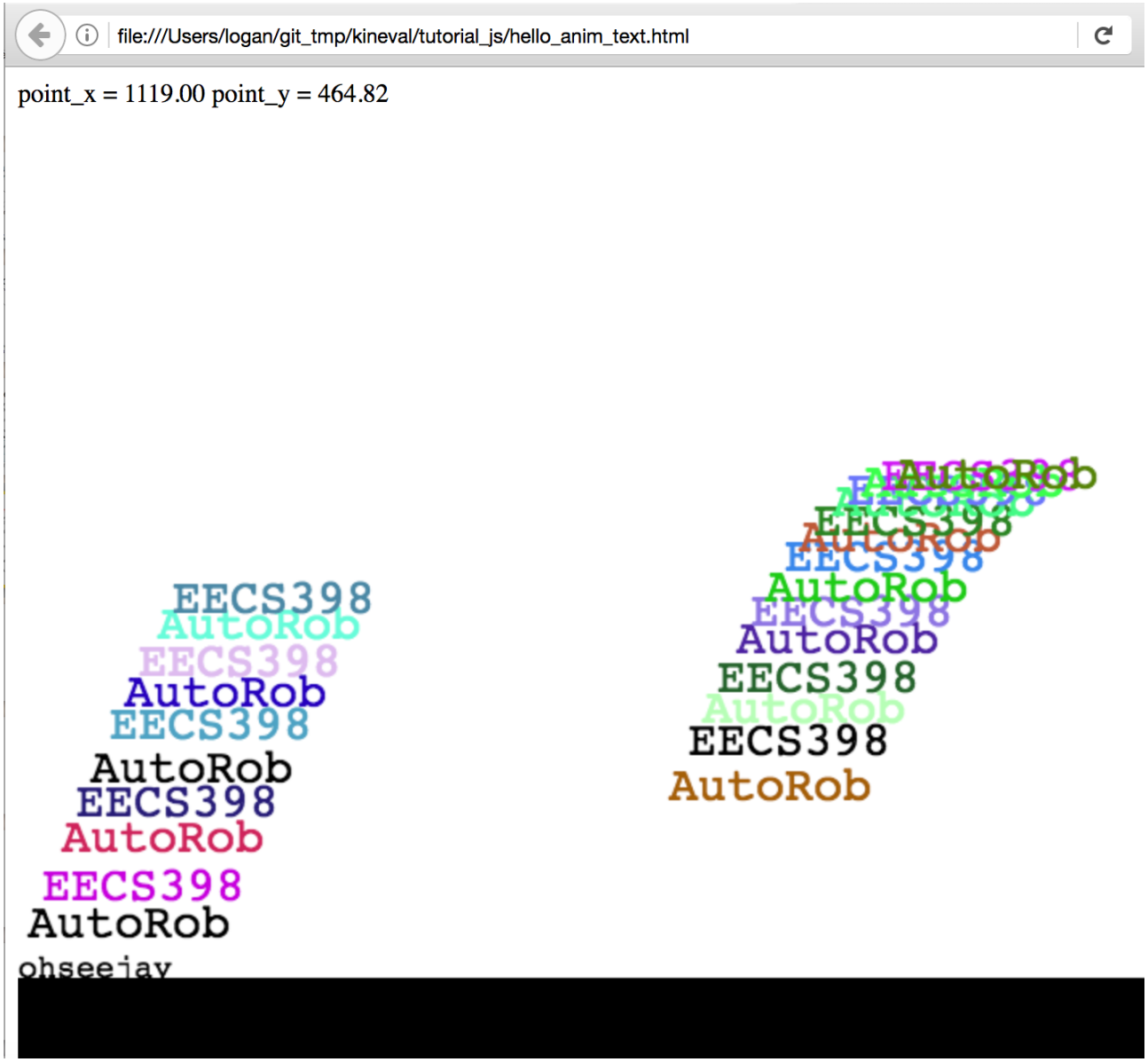
point_x = 30.00 point_y = 410.70

M4PRoGReS

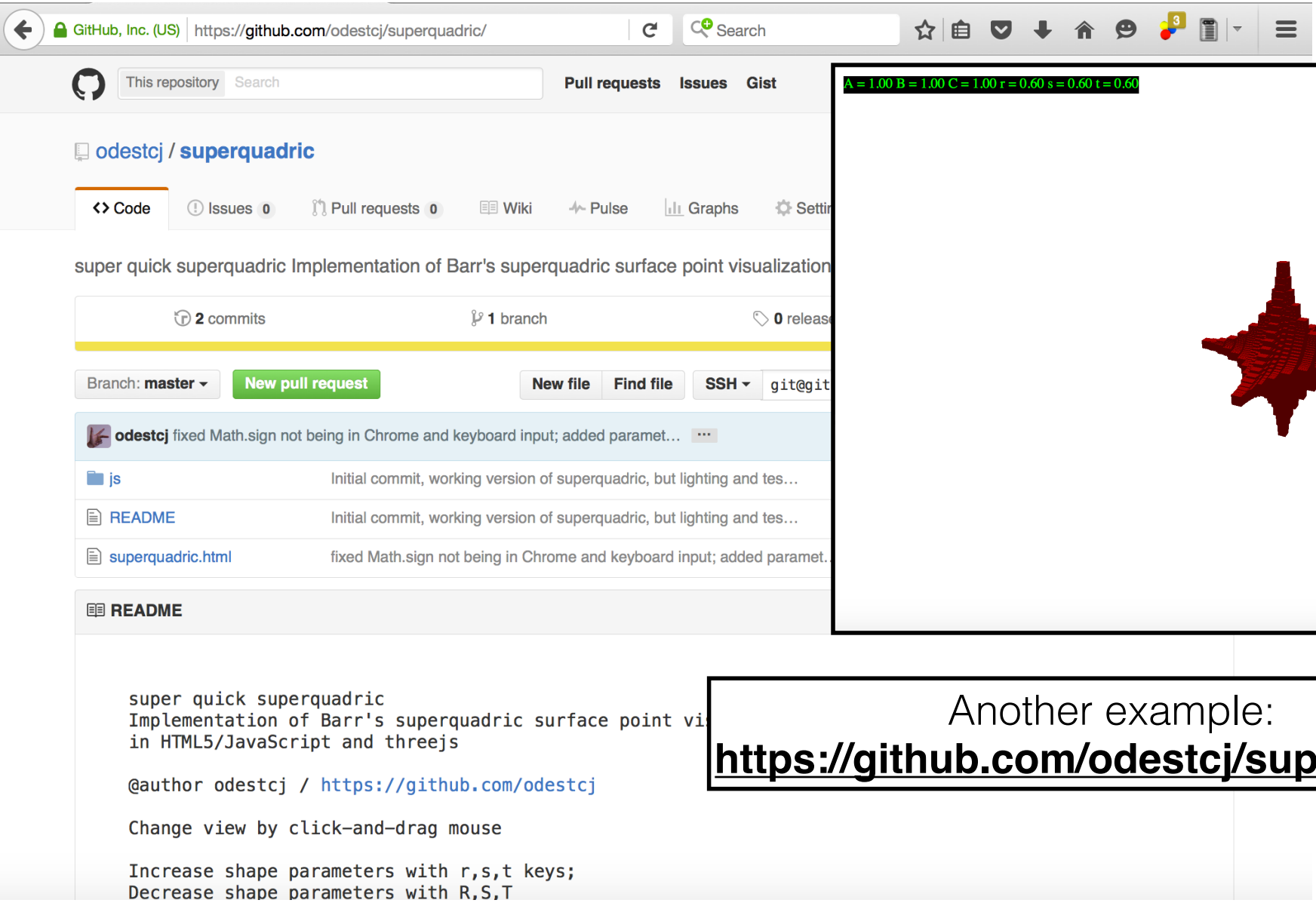
```
...

function update() {

   // get a reference to the canvas element "draw_canvas" in the document.
   var canvas = document.getElementById("draw_canvas");

   // update the size of the canvas based on dimensions of browser windows
   //   note: window is a global object for the browser window
   canvas.width = window.innerWidth;
   canvas.height = window.innerHeight-50;

   // move the circle forward by assignment
   point.x = point.x + 5;

   // if statement conditionally executes with roughly this structure:
   //   if (condition) { code } else if (condition) {code} else {code}

   // if the circle is at the extent of the canvas, move it back to the start
   if (point.x > canvas.width) {
       point.x = 0;
   }

   // make the circle look like it bouncing using a sin function
   //   note: the Math object has a number of useful functions
   point.y = (canvas.height-60)-Math.abs((canvas.height/2)*Math.sin(point.x/
(canvas.width*0.1)));}

...
```

RETURNS A REFERENCE TO ANY DOM ELEMENT

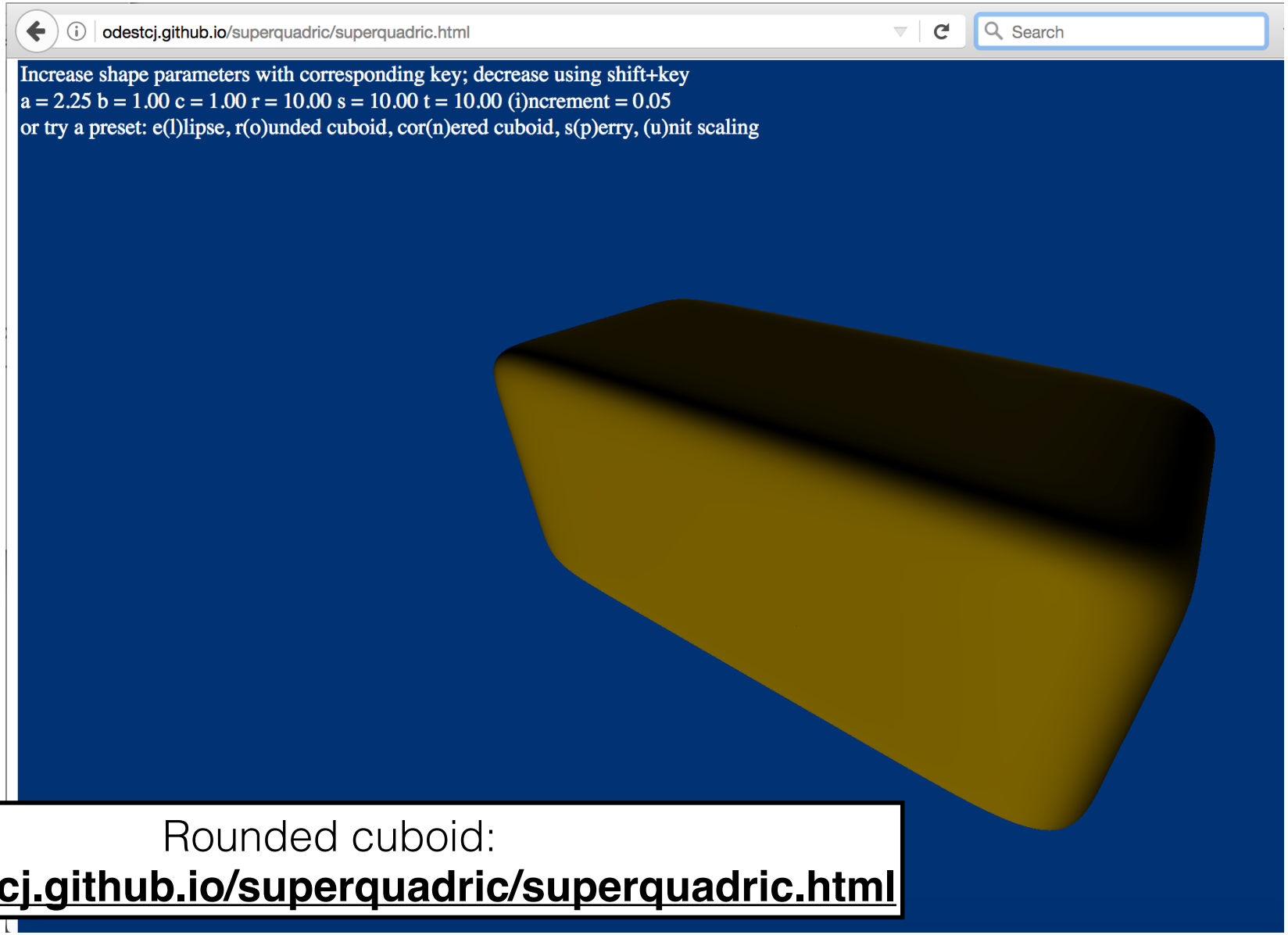
One more animation example


file:///Users/logan/git_tmp/kineval/tutorial_js/hello_anim_text.html
point_x = 1119.00 point_y = 464.82
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
EECS398
AutoRob
ohseejay

Many examples available online



Another example:
https://github.com/odestcj/superquadric



Rounded cuboid:
http://odestcj.github.io/superquadric/superquadric.html

GitHub, Inc. (US) https://github.com/autorob/ki

Search or jump to... / Pull requests Issues Marketplace Explore

autorob / **kineval-stencil**

Unwatch 2 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wi

https://github.com/autorob/kineval-stencil

Stencil code for KinEval (Kinematic Evaluator) for robot control, kinematics, decision, and dynamics in JavaScript/HTML5 Edit

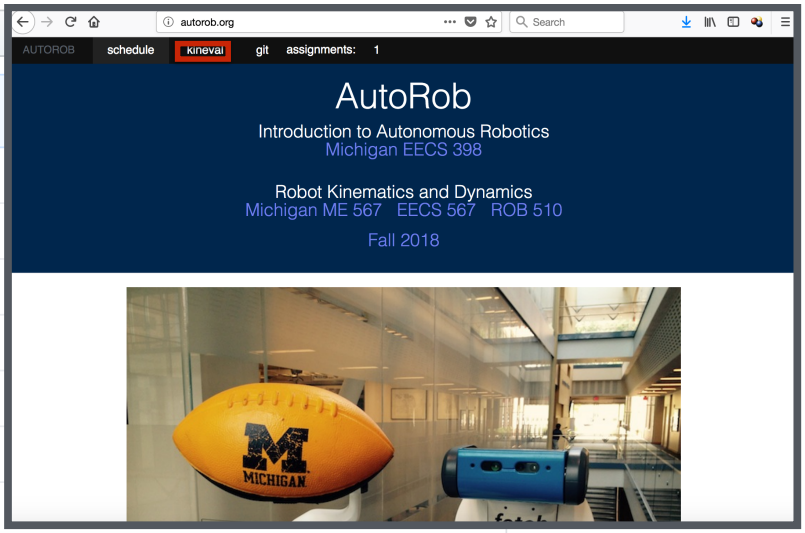
Add topics

2 commits **1** branch **0** releases **2** contributors

Branch: master New pull request Create new file

odestcj initial commit Fall 2018

js	initial commit Fall 2018	
kineval	initial commit Fall 2018	
project_pathplan	initial commit Fall 2018	
project_pendularm	initial commit Fall 2018	
robots	initial commit Fall 2018	
tutorial_heapsort	initial commit Fall 2018	
tutorial_js	initial commit Fall 2018	
worlds	initial commit Fall 2018	4 hours ago



What is version control?

What is version control?

- Maintains a past history of changes for your code (or any project)
- History of changes (or “commits”) maintained in a repository
- Basic workflow
  - Code is “checked out” (or “pulled”) from a repository, then modified
  - These updates are then “checked in” (or “committed”) to the repository
  - Repository maintains history as “diffs”, the changes between before and after checking in a commit

For example… our TED talk

https://www.ted.com/talks/henry_evans_and_chad_jenkins_meet_the_robots_for_humanity?language=en

TED Watch Discover Attend Participate About Search... Log in

Henry Evans and Chad Jenkins:

Meet the robots for humanity

TEDxMidAtlantic · 10:21 · **Filmed** Oct 2013

28 subtitle languages

View interactive transcript

Share this idea

Facebook Twitter Email Embed More

1,145,408 Total views

Share this talk and track your influence!

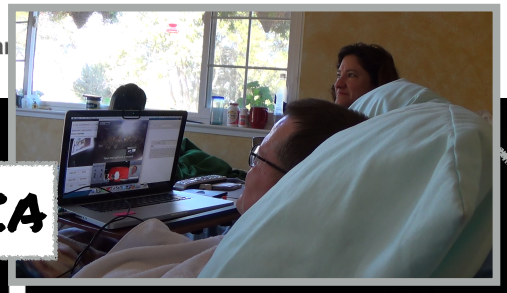
...bot to

https://www.ted.com/talks/henry_evans_and_chad_jenkins_meet_the_robots_for_humanity?language=en#

TED Talks are free thanks to support from

https://www.ted.com/talks/henry_evans_and_chad_jenkins_meet_the_robots_for_humanity?language=en

TED Watch Discover Attend Par Search... Log in



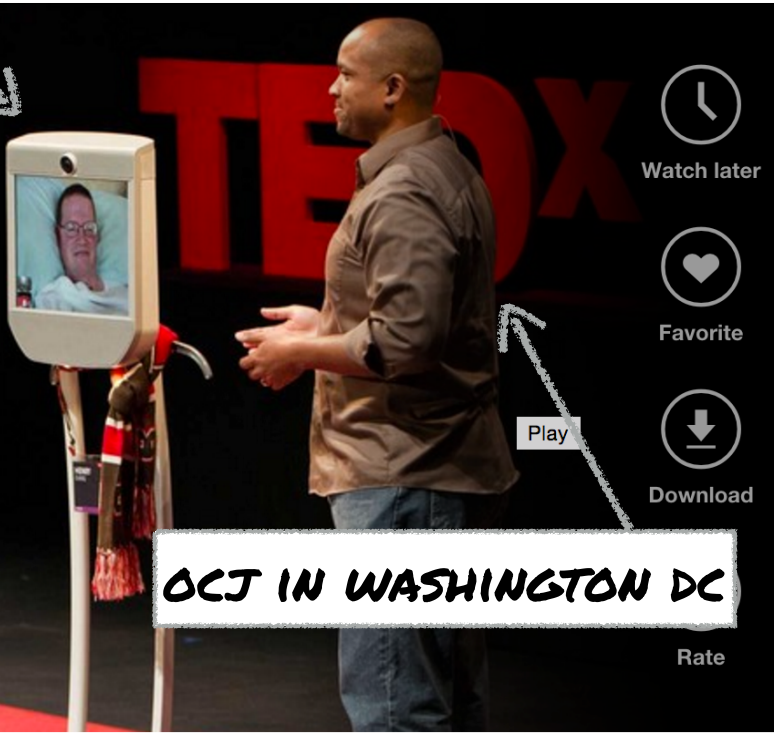
HENRY EVANS IN PALO ALTO CA

Meet the robots for humanity

TEDxMidAtlantic · 10:21 · **Filmed** Oct 2013

28 subtitle languages

View interactive transcript



Watch later

Favorite

Download

OCJ IN WASHINGTON DC

Rate

Share this idea

Facebook Twitter Email Embed More

1,145,408 Total views

Share this talk and track your influence!

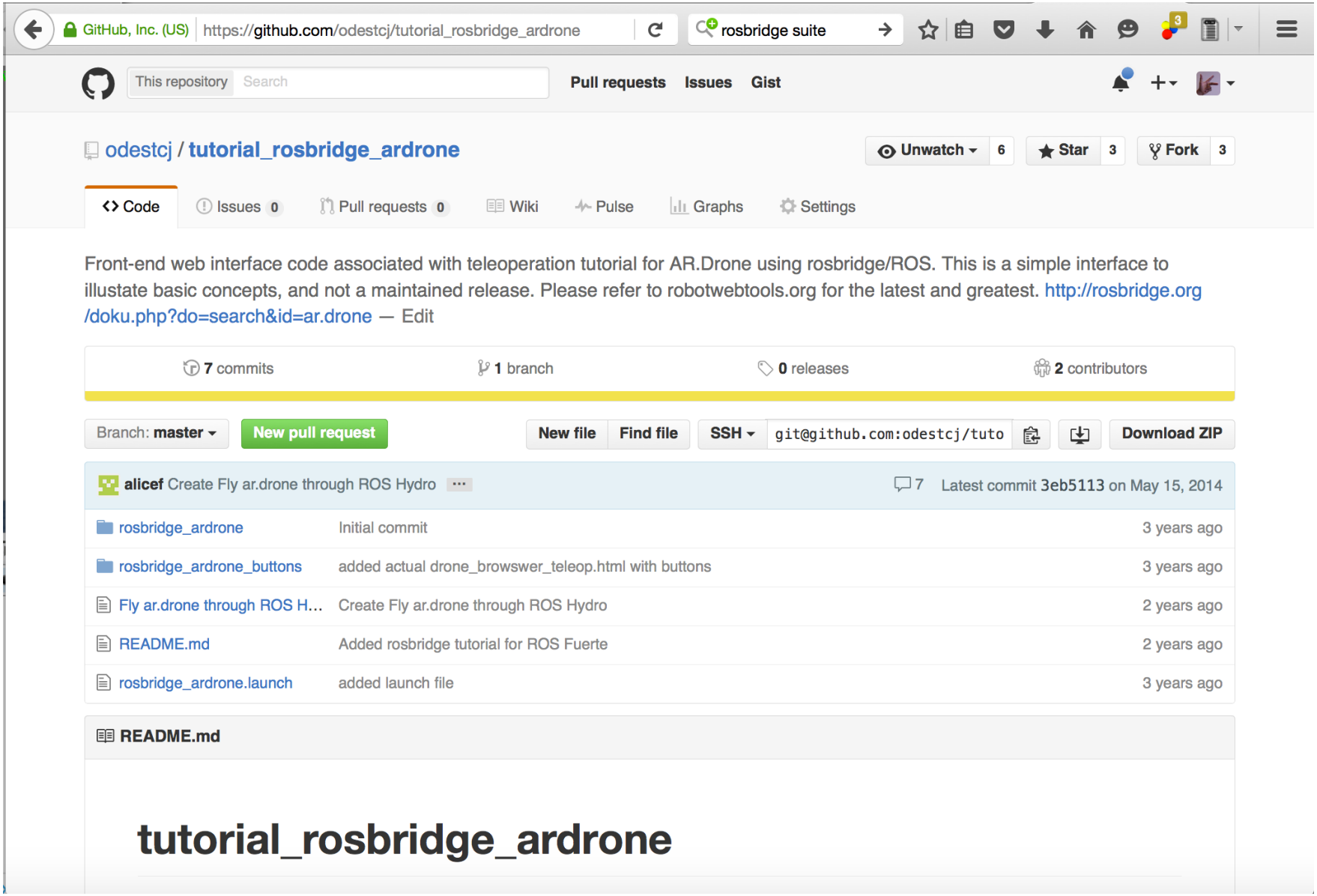
TED Talks are free thanks to support from Infosys®

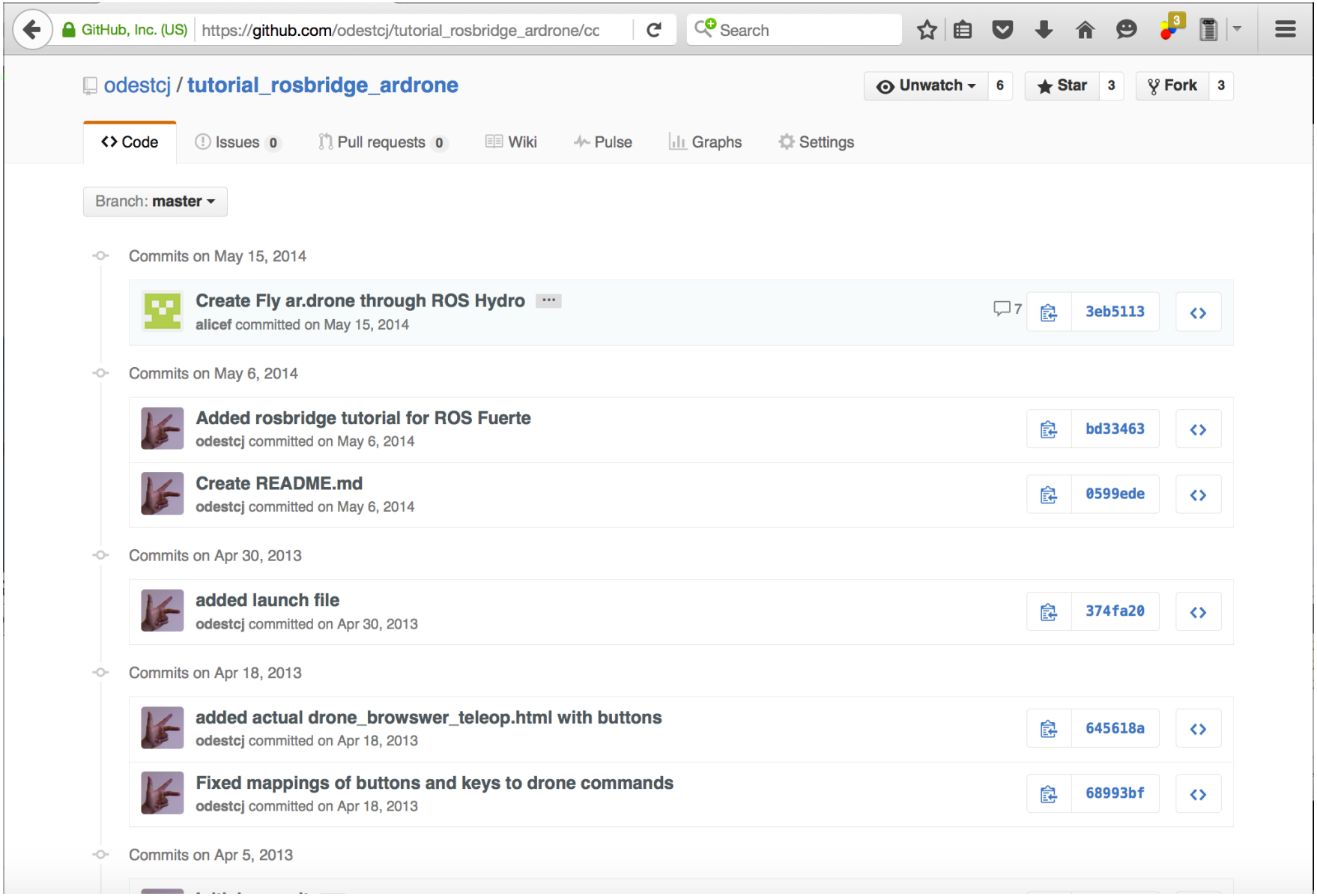
bot to

https://www.ted.com/talks/henry_evans_and_chad_jenkins_meet_the_robots_for_humanity?language=en#

TED Talks are free thanks to support from


IN-BROWSER DRONE CONTROL
Your microphone is muted.


GitHub, Inc. (US) https://github.com/odestcj/tutorial_rosbridge_ardrone
rosbridge suite
This repository Search
Pull requests Issues Gist
odestcj / tutorial_rosbridge_ardrone
Unwatch 6 Star 3 Fork 3
Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settings
Front-end web interface code associated with teleoperation tutorial for AR.Drone using rosbridge/ROS. This is a simple interface to illustate basic concepts, and not a maintained release. Please refer to robotwebtools.org for the latest and greatest. http://rosbridge.org/doku.php?do=search&id=ar.drone — Edit
7 commits 1 branch 0 releases 2 contributors
Branch: master New pull request New file Find file SSH git@github.com:odestcj/tuto Download ZIP
alicef Create Fly ar.drone through ROS Hydro 7 Latest commit 3eb5113 on May 15, 2014
rosbridge_ardrone Initial commit 3 years ago
rosbridge_ardrone_buttons added actual drone_browswer_teleop.html with buttons 3 years ago
Fly ar.drone through ROS H... Create Fly ar.drone through ROS Hydro 2 years ago
README.md Added rosbridge tutorial for ROS Fuerte 2 years ago
rosbridge_ardrone.launch added launch file 3 years ago
README.md
tutorial_rosbridge_ardrone


GitHub, Inc. (US)
https://github.com/odestcj/tutorial_rosbridge_ardrone/cc
Search
odestcj / tutorial_rosbridge_ardrone
Unwatch 6
Star 3
Fork 3
Code
Issues 0
Pull requests 0
Wiki
Pulse
Graphs
Settings
Branch: master
Commits on May 15, 2014
Create Fly ar.drone through ROS Hydro
alicef committed on May 15, 2014
7
3eb5113
Commits on May 6, 2014
Added rosbridge tutorial for ROS Fuerte
odestcj committed on May 6, 2014
bd33463
Create README.md
odestcj committed on May 6, 2014
0599ede
Commits on Apr 30, 2013
added launch file
odestcj committed on Apr 30, 2013
374fa20
Commits on Apr 18, 2013
added actual drone_browswer_teleop.html with buttons
odestcj committed on Apr 18, 2013
645618a
Fixed mappings of buttons and keys to drone commands
odestcj committed on Apr 18, 2013
68993bf
Commits on Apr 5, 2013

Large open source projects…

robotwebtools.org

3D INTERACTIONS
USING THE LATEST IN WEBGL

MULTI-PLATFORM SUPPORT
HARNESSING THE POWER OF ROS

TOWARDS COMPATIBILITY
MORE BROWSERS, MORE ROBOTS.

ROBOT WEB ARCHITECTURE

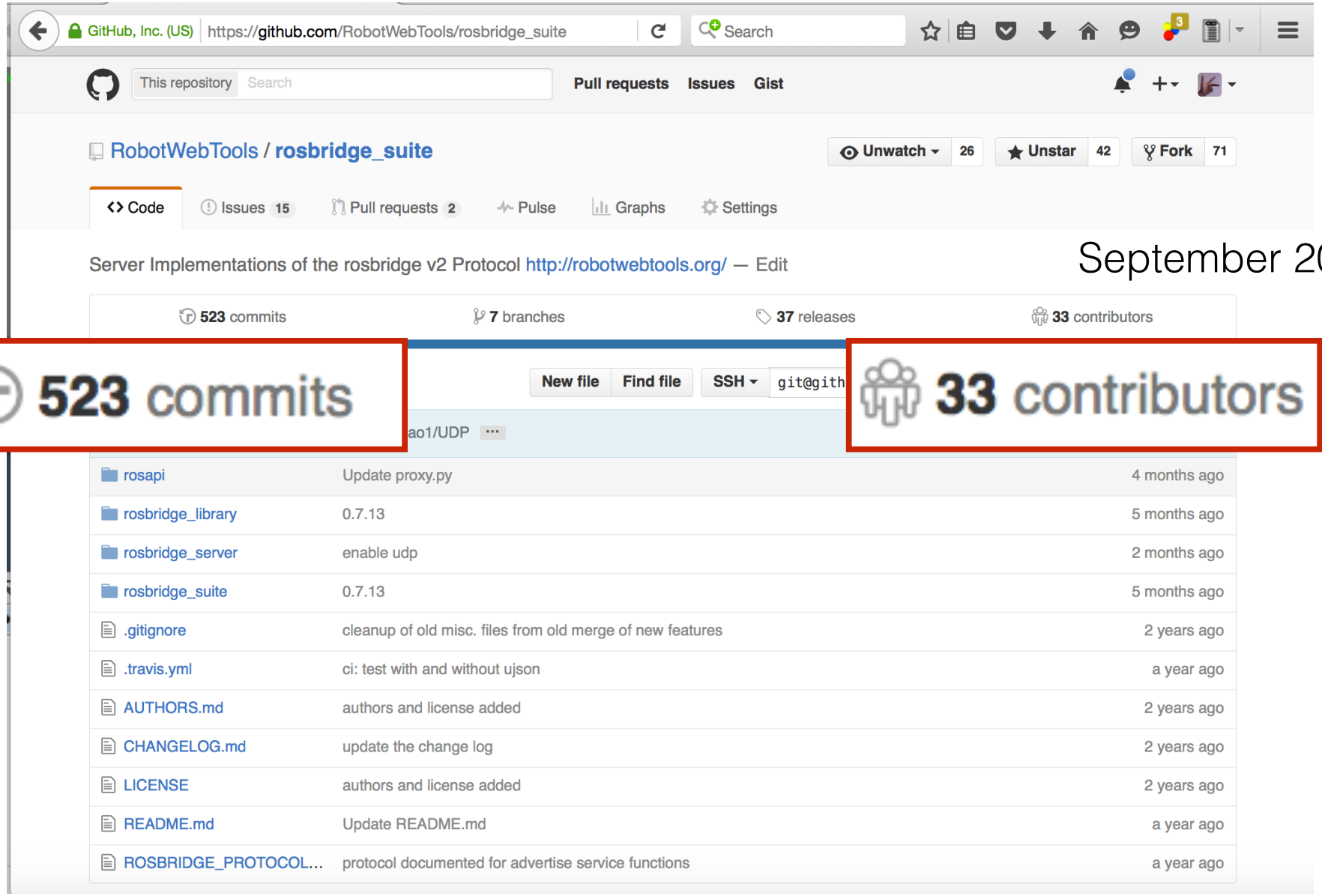
BRIDGING ROBOTS AND THE WEB

A variety of routes are available for architecting a robot web

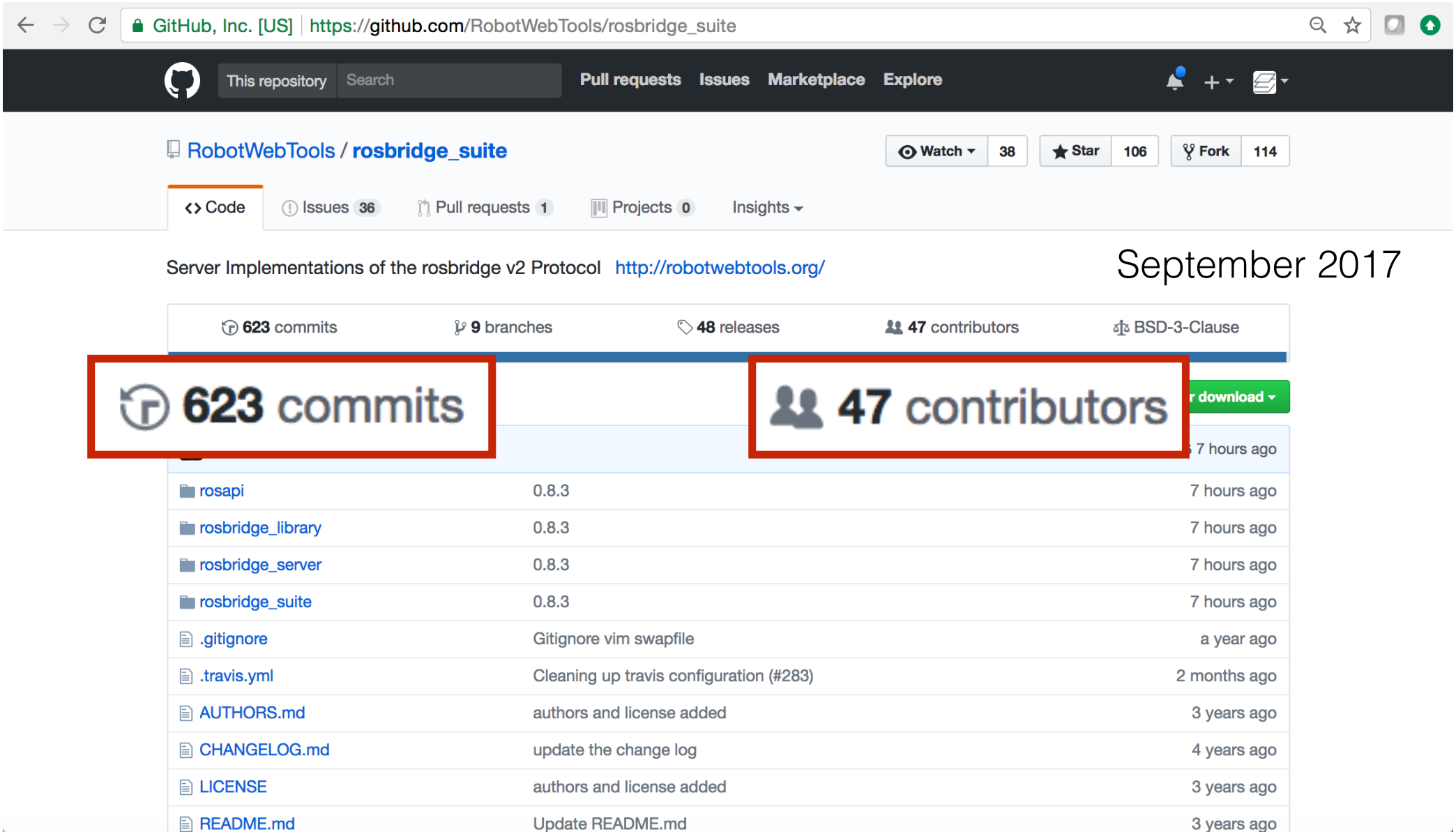
ROSBRIDGE AS A TRANSPORT

USING JSON TO SPEAK TO YOUR ROBOT

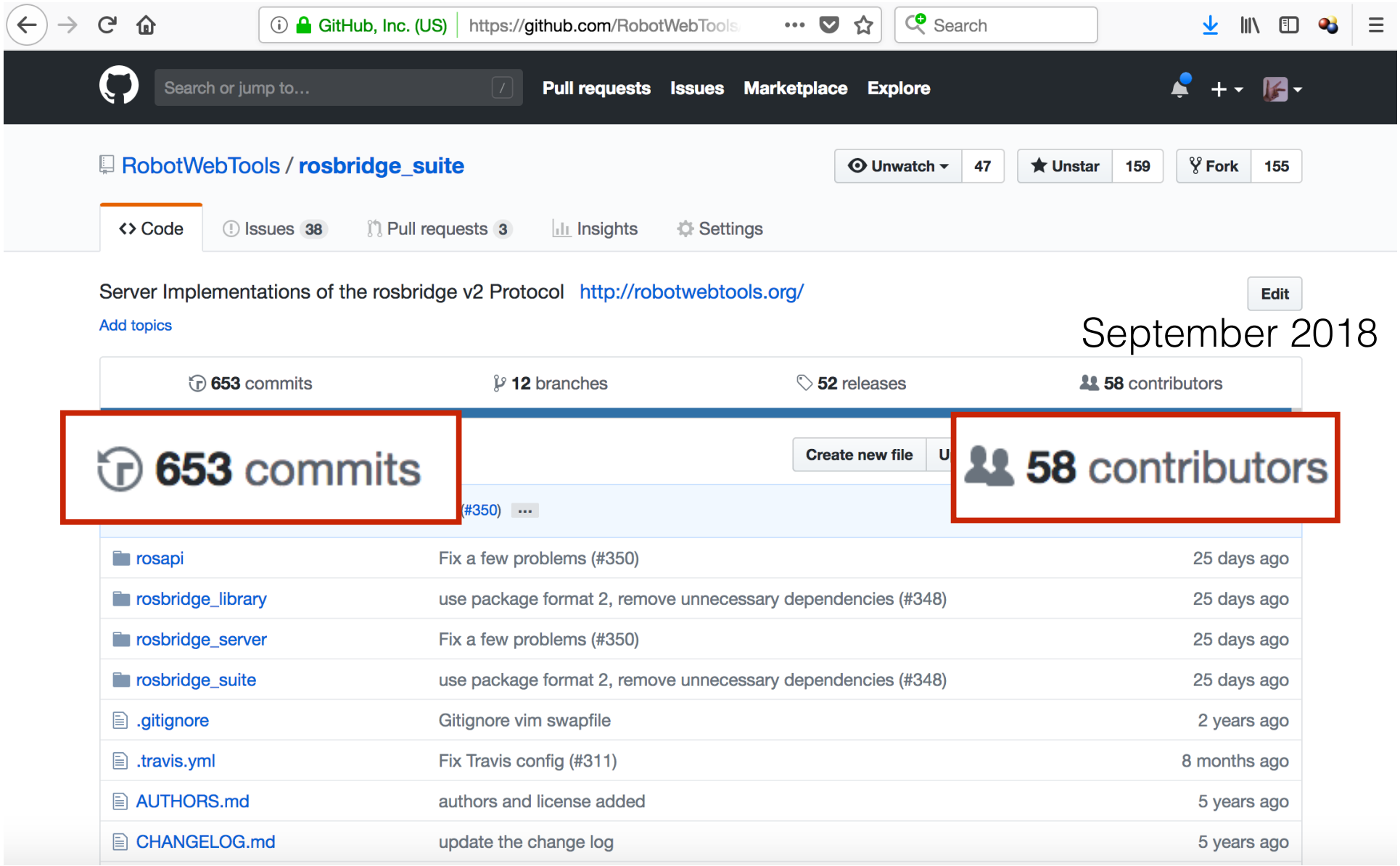
While ROS works great for applications on the robot, another layer is

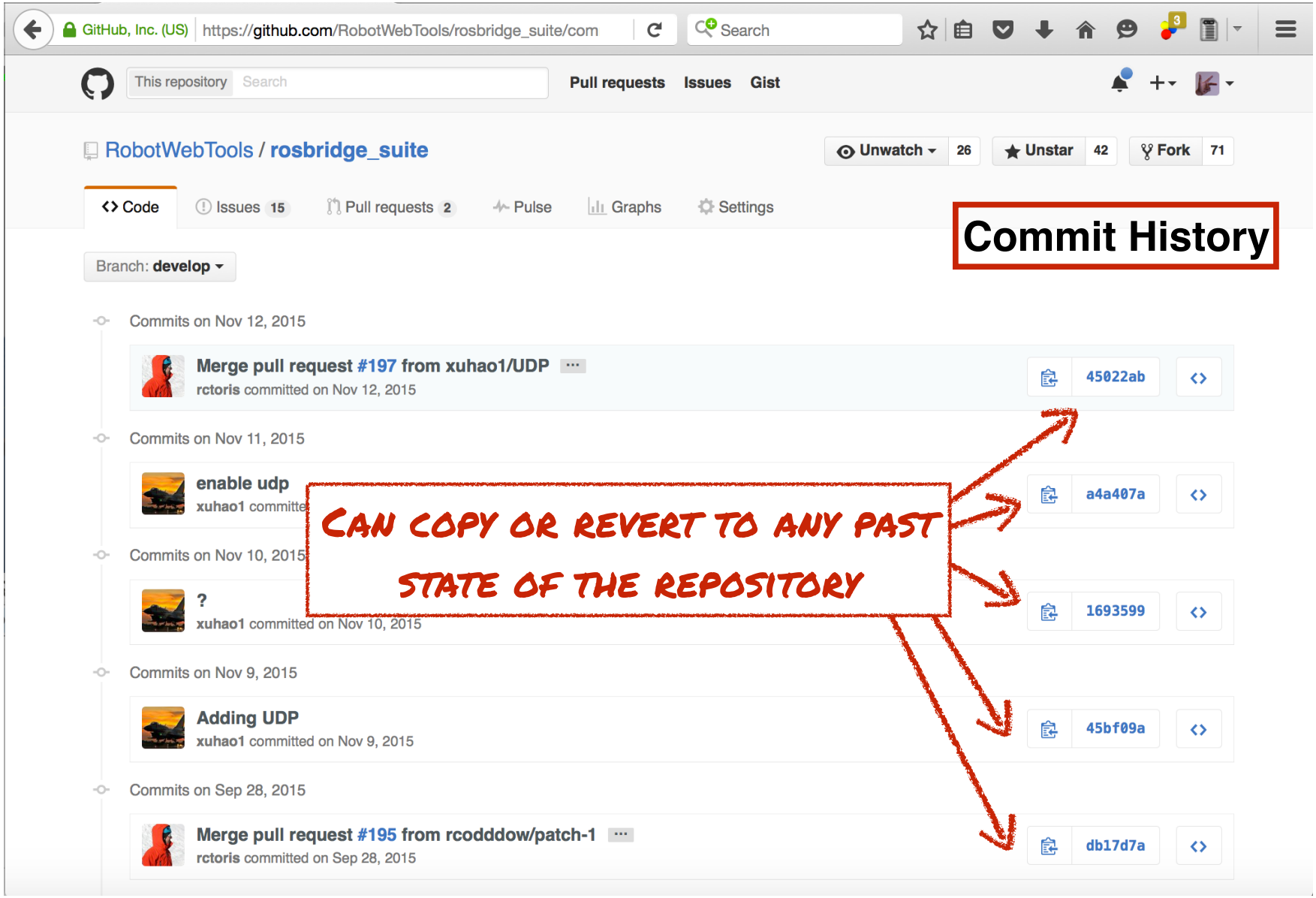


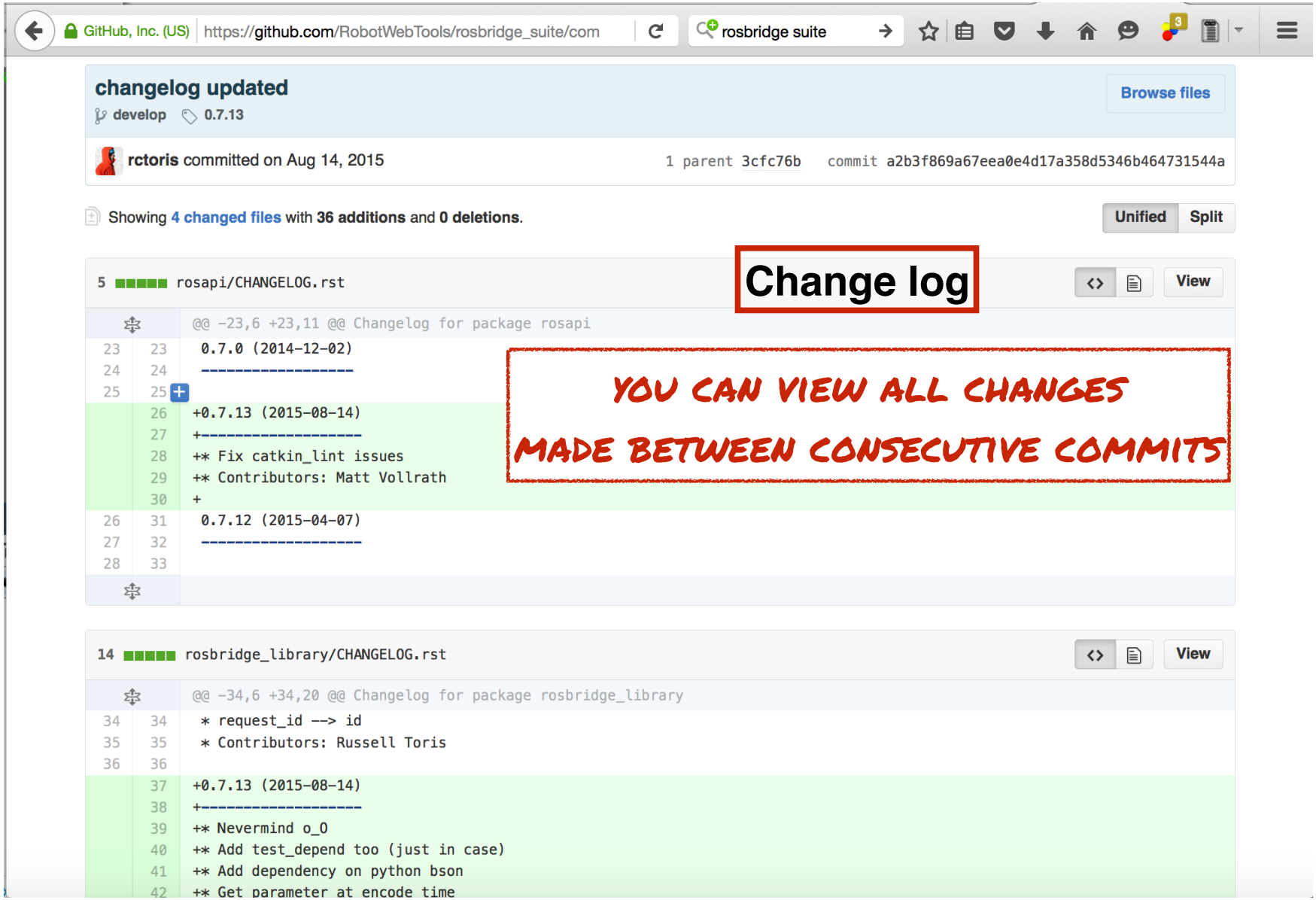
September 2016



September 2017


GitHub, Inc. (US)
https://github.com/RobotWebTools
Search
Search or jump to…
Pull requests
Issues
Marketplace
Explore
RobotWebTools / rosbridge_suite
Unwatch 47
Unstar 159
Fork 155
Code
Issues 38
Pull requests 3
Insights
Settings
Server Implementations of the rosbridge v2 Protocol http://robotwebtools.org/
Edit
Add topics
September 2018
653 commits
12 branches
52 releases
58 contributors
653 commits
Create new file
58 contributors
(#350)
rosapi Fix a few problems (#350) 25 days ago
rosbridge_library use package format 2, remove unnecessary dependencies (#348) 25 days ago
rosbridge_server Fix a few problems (#350) 25 days ago
rosbridge_suite use package format 2, remove unnecessary dependencies (#348) 25 days ago
.gitignore Gitignore vim swapfile 2 years ago
.travis.yml Fix Travis config (#311) 8 months ago
AUTHORS.md authors and license added 5 years ago
CHANGELOG.md update the change log 5 years ago


GitHub, Inc. (US)
https://github.com/RobotWebTools/rosbridge_suite/com
Search
This repository
Search
Pull requests
Issues
Gist
RobotWebTools / rosbridge_suite
Unwatch 26
Unstar 42
Fork 71
Code
Issues 15
Pull requests 2
Pulse
Graphs
Settings
Commit History
Branch: develop
Commits on Nov 12, 2015
Merge pull request #197 from xuhao1/UDP
rctoris committed on Nov 12, 2015
45022ab
Commits on Nov 11, 2015
enable udp
xuhao1 committe
a4a407a
Commits on Nov 10, 2015
?
xuhao1 committed on Nov 10, 2015
1693599
Commits on Nov 9, 2015
Adding UDP
xuhao1 committed on Nov 9, 2015
45bf09a
Commits on Sep 28, 2015
Merge pull request #195 from rcodddow/patch-1
rctoris committed on Sep 28, 2015
db17d7a
CAN COPY OR REVERT TO ANY PAST STATE OF THE REPOSITORY


GitHub, Inc. (US)
https://github.com/RobotWebTools/rosbridge_suite/com
rosbridge suite
changelog updated
develop
0.7.13
Browse files
rctoris committed on Aug 14, 2015
1 parent 3cfc76b
commit a2b3f869a67eea0e4d17a358d5346b464731544a
Showing 4 changed files with 36 additions and 0 deletions.
Unified
Split
Change log
5 rosapi/CHANGELOG.rst
View
@@ -23,6 +23,11 @@ Changelog for package rosapi
0.7.0 (2014-12-02)
+0.7.13 (2015-08-14)
+* Fix catkin_lint issues
+* Contributors: Matt Vollrath
+
0.7.12 (2015-04-07)
YOU CAN VIEW ALL CHANGES
MADE BETWEEN CONSECUTIVE COMMITS
14 rosbridge_library/CHANGELOG.rst
View
@@ -34,6 +34,20 @@ Changelog for package rosbridge_library
* request_id --> id
* Contributors: Russell Toris
+0.7.13 (2015-08-14)
+* Nevermind o_O
+* Add test_depend too (just in case)
+* Add dependency on python bson
+* Get parameter at encode time

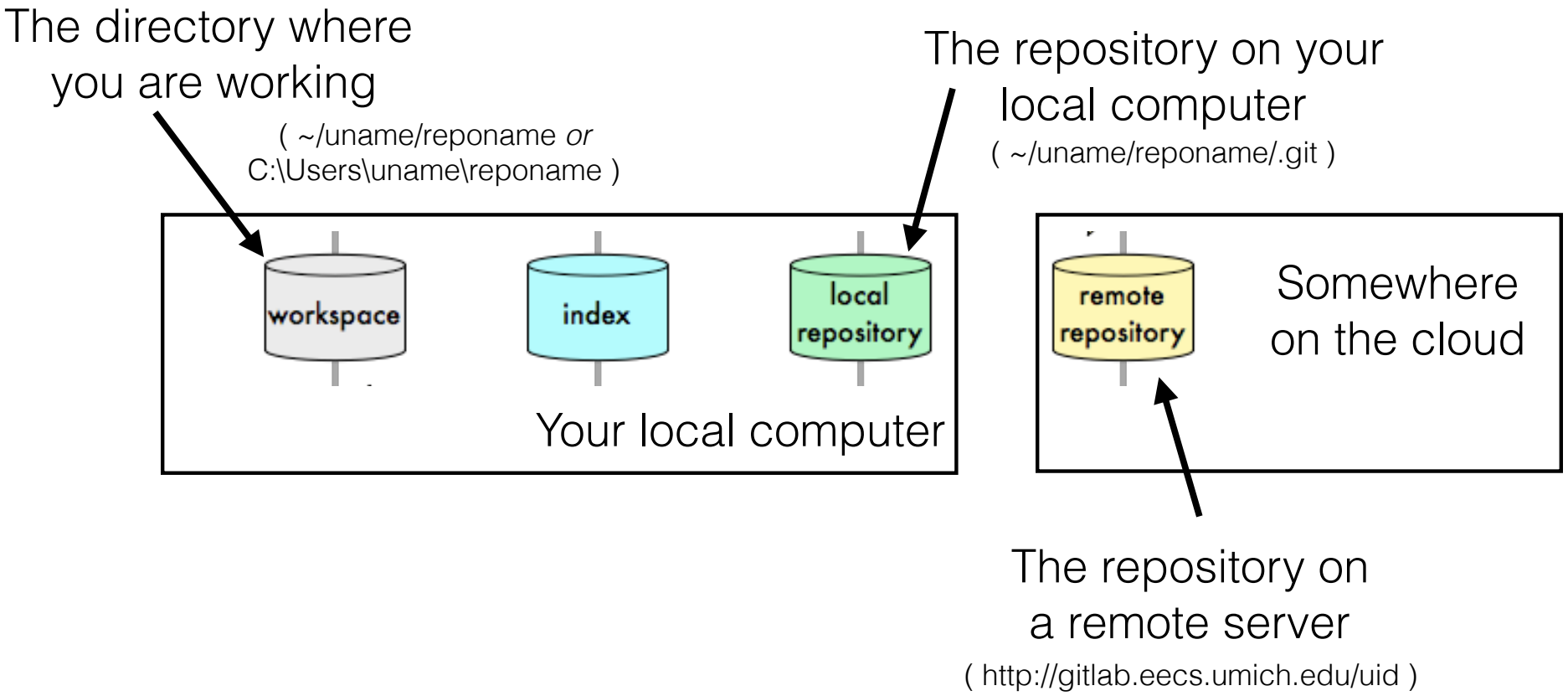
Version control options

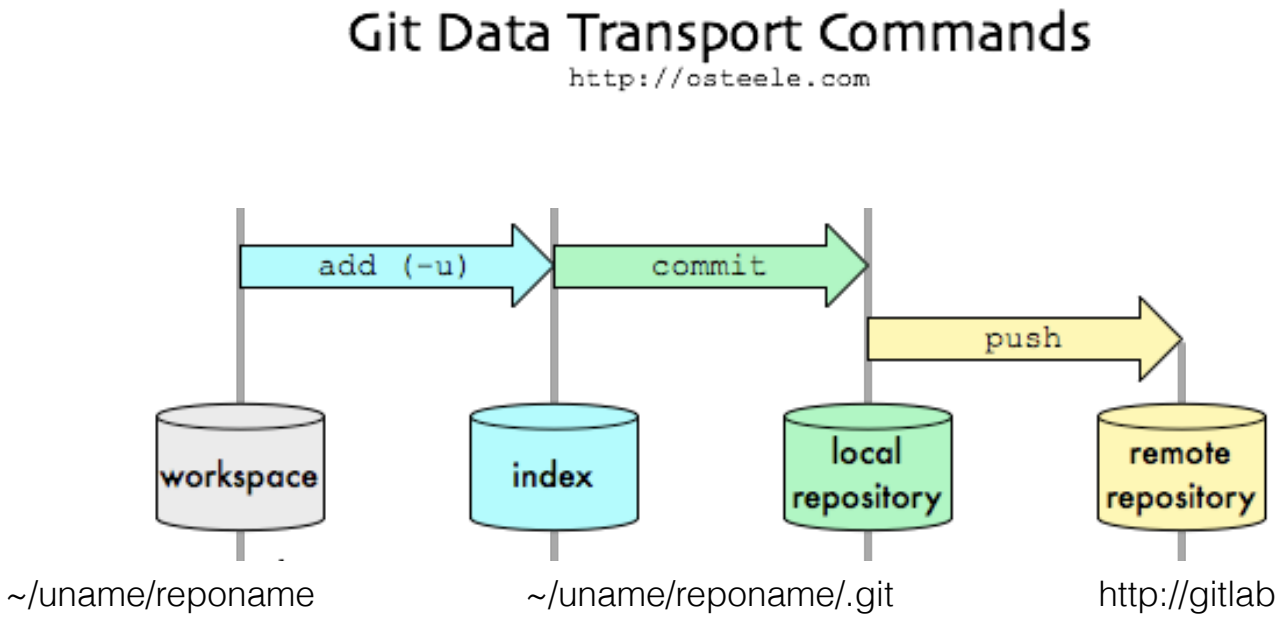
Version control options

- Concurrent Versioning System (CVS): very old school
- Subversion (SVN): still in significant use
- Mercurial (hg)
- **git: used in AutoRob**

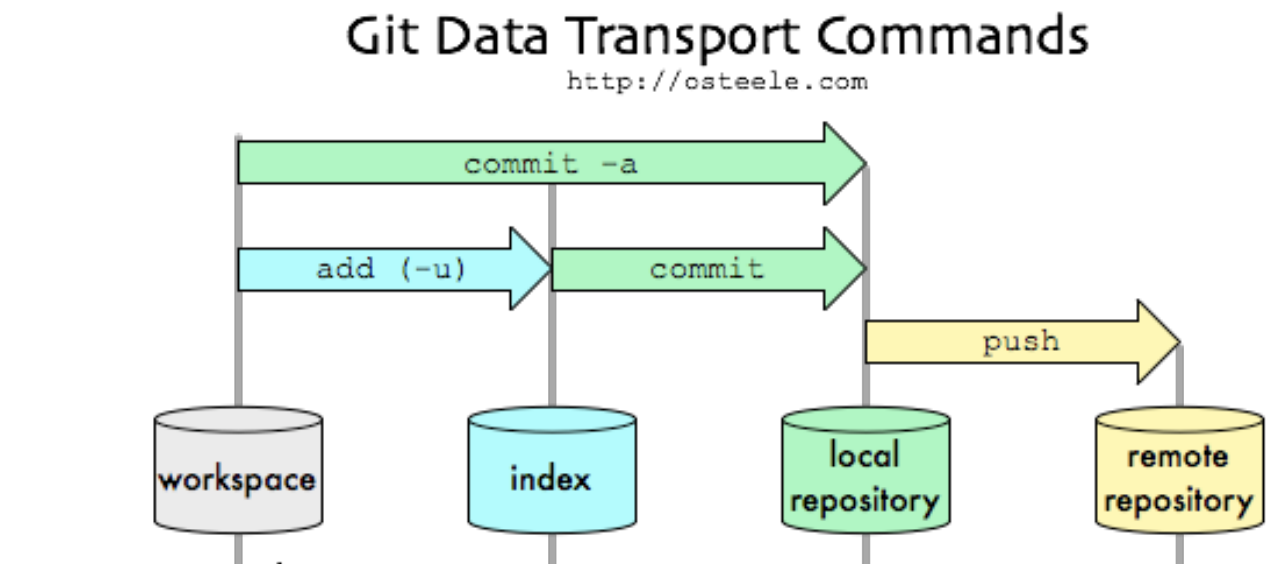
Git Data Transport Commands

http://osteele.com



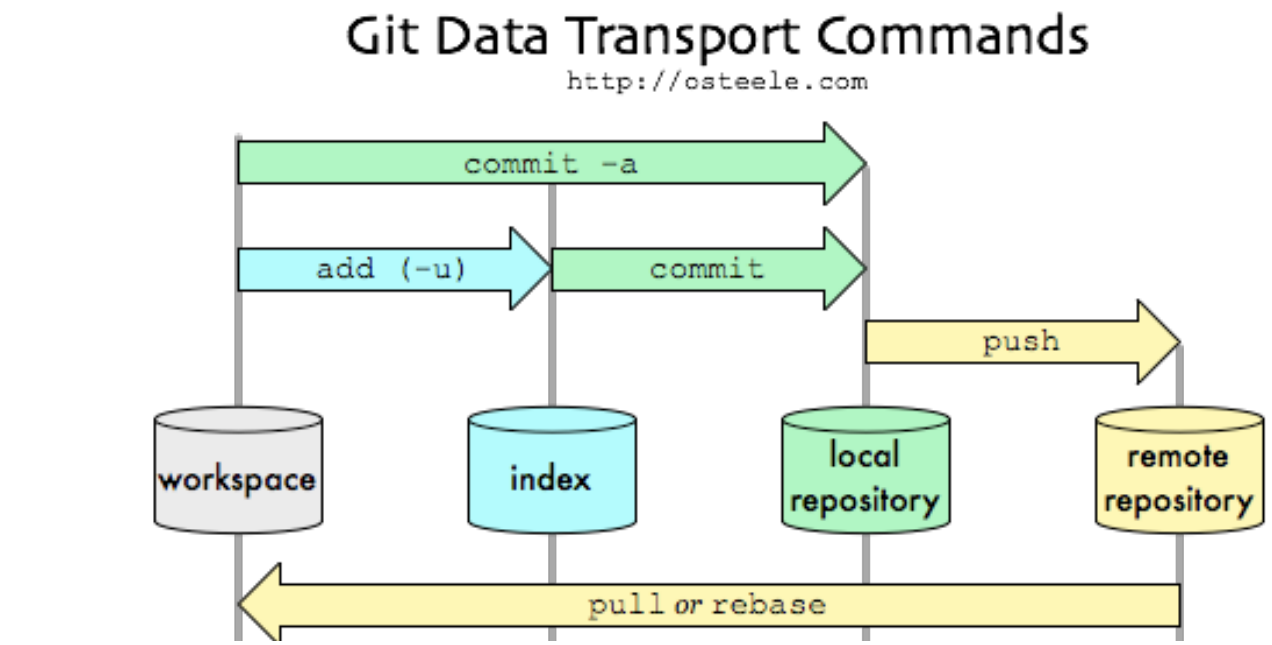


After making local changes, you can add, commit, and push to your remote repository



~/uname/reponame ~/uname/reponame/.git http://gitlab.eecs.umich.edu/~uid

If there are no files to add, just commit and push



A pull command updates the local workspace with changes from the remote repository

git basics: commands

- Push completed project to repository (or just to update)
  - add files to a repository: **`git add <file listing>`**
  - commit changes to local repo: **`git commit -a -m "<msg>"`**
  - push local changes to a remote repository: **`git push`**
- Pull to updates your local repository (and workspace) from remote
  - pull remote changes to a local repository: **`git pull`**

git stencil quick start

- Create a git repository on github, bitbucket, or EECS gitlab website
- Install git on your machine
  - https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
  - OSX: https://code.google.com/p/git-osx-installer/
- Clone the repository to your machine:

```
git clone https://github.com/yourid/yourrepo.git
```

git stencil quick start

- Clone and copy contents of stencil repository the repository to your machine:

```
git clone https://github.com/autorob/kineval-stencil
```

- Copy contents of stencil repository to your repository

```
cd <your repository directory>
cp -r <stencil repository directory>/* .
```

- Commit these changes and then push them to your remote repository

```
cd <your repository directory>
git commit -a -m “initial commit”
git push
```

https://github.com/ohseejay/kineval-stencil-f16

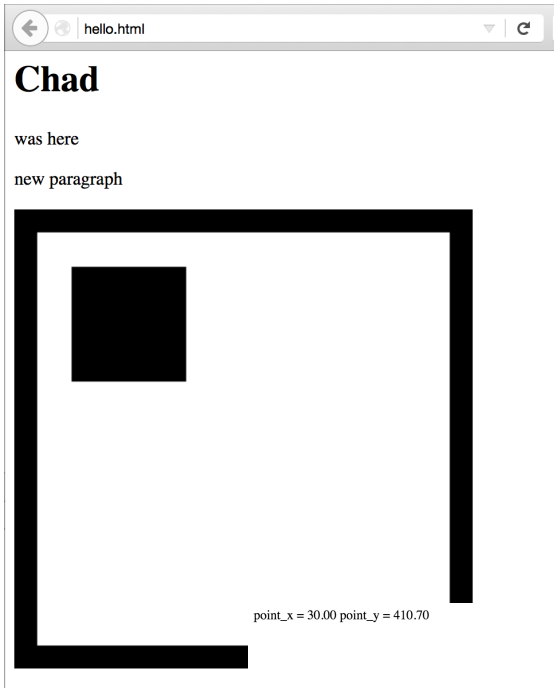
2 commits 1 branch

Branch: master New pull request

odestcj Fall 2016 release

- js
- kineval
- project_pathplan
- project_pendularm
- robots
- tutorial_heapsort
- tutorial_js
- worlds
- README.md
- home.html

README.md



Tutorial Examples

https://github.com/ohseejay/kineval-stencil-f16

2 commits 1 branch

Branch: master New pull request

odestcj Fall 2016 release

- js
- kineval
- project_pathplan
- project_pendularm
- robots
- tutorial_heapsort
- tutorial_js
- worlds
- README.md
- home.html

README.md

My Heap Sort

file:///Users/logan/git_tmp/kineval/tutorial_heapsort/source_heap

Search

My Heap Sort

```
                                78
                  100                           459
        797                319             3157            1165
   4631      3413      1472      8465      5949      3911      4921      2575
8320 9772 5185 3865 6409
```

check
numbers to sort: 5949 8320 1472 6409 3865 4921 3157 459 5185 797 8465 3911 2575 100 1165 4631 9772 3413 78 319
heap (insert 5949): 5949
heap (insert 8320): 5949 8320
heap (insert 1472): 1472 8320 5949
heap (insert 6409): 1472 6409 5949 8320
heap (insert 3865): 1472 3865 5949 8320 6409
heap (insert 4921): 1472 3865 4921 8320 6409 5949
heap (insert 3157): 1472 3865 3157 8320 6409 5949 4921
heap (insert 459): 459 1472 3157 3865 6409 5949 4921 8320
heap (insert 5185): 459 1472 3157 3865 6409 5949 4921 8320 5185
heap (insert 797): 459 797 3157 3865 1472 5949 4921 8320 5185 6409
heap (insert 8465): 459 797 3157 3865 1472 5949 4921 8320 5185 6409 8465
heap (insert 3911): 459 797 3157 3865 1472 3911 4921 8320 5185 6409 8465 5949
heap (insert 2575): 459 797 2575 3865 1472 3157 4921 8320 5185 6409 8465 5949 3911
heap (insert 100): 100 797 459 3865 1472 3157 2575 8320 5185 6409 8465 5949 3911 4921
heap (insert 1165): 100 797 459 3865 1472 3157 1165 8320 5185 6409 8465 5949 3911 4921 2575
heap (insert 4631): 100 797 459 3865 1472 3157 1165 4631 5185 6409 8465 5949 3911 4921 2575 8320
heap (insert 9772): 100 797 459 3865 1472 3157 1165 4631 5185 6409 8465 5949 3911 4921 2575 8320 9772
heap (insert 3413): 100 797 459 3413 1472 3157 1165 4631 3865 6409 8465 5949 3911 4921 2575 8320 9772 5185
heap (insert 78): 78 100 459 797 1472 3157 1165 4631 3413 6409 8465 5949 3911 4921 2575 8320 9772 5185 3865
heap (insert 319): 78 100 459 797 319 3157 1165 4631 3413 1472 8465 5949 3911 4921 2575 8320 9772 5185 3865 6409
heap (extract 78): 100 319 459 797 1472 3157 1165 4631 3413 6409 8465 5949 3911 4921 2575 8320 9772 5185 3865
heap (extract 100): 319 797 459 3413 1472 3157 1165 4631 3865 6409 8465 5949 3911 4921 2575 8320 9772 5185
heap (extract 319): 459 797 1165 3413 1472 3157 2575 4631 3865 6409 8465 5949 3911 4921 5185 8320 9772
heap (extract 459): 797 1472 1165 3413 6409 3157 2575 4631 3865 9772 8465 5949 3911 4921 5185 8320
heap (extract 797): 1165 1472 2575 3413 6409 3157 4921 4631 3865 9772 8465 5949 3911 8320 5185
heap (extract 1165): 1472 3413 2575 3865 6409 3157 4921 4631 5185 9772 8465 5949 3911 8320
heap (extract 1472): 2575 3413 3157 3865 6409 3911 4921 4631 5185 9772 8465 5949 8320
heap (extract 2575): 3157 3413 3911 3865 6409 5949 4921 4631 5185 9772 8465 8320
heap (extract 3157): 3413 3865 3911 4631 6409 5949 4921 8320 5185 9772 8465
heap (extract 3413): 3865 4631 3911 5185 6409 5949 4921 8320 8465 9772
heap (extract 3865): 3911 4631 4921 5185 6409 5949 9772 8320 8465
heap (extract 3911): 4631 5185 4921 8320 6409 5949 9772 8465
heap (extract 4631): 4921 5185 5949 8320 6409 8465 9772
heap (extract 4921): 5185 6409 5949 8320 9772 8465
heap (extract 5185): 5949 6409 8465 8320 9772
heap (extract 5949): 6409 8320 8465 9772
heap (extract 6409): 8320 9772 8465
heap (extract 8320): 8465 9772
heap (extract 8465): 9772
heap (extract 9772):
sorted: 78 100 319 459 797 1165 1472 2575 3157 3413 3865 3911 4631 4921 5185 5949 6409 8320 8465 9772

Project 0:
Heap sort

ub.io

https://github.com/ohseejay/kineval-stencil-f16

2 commits 1 branch

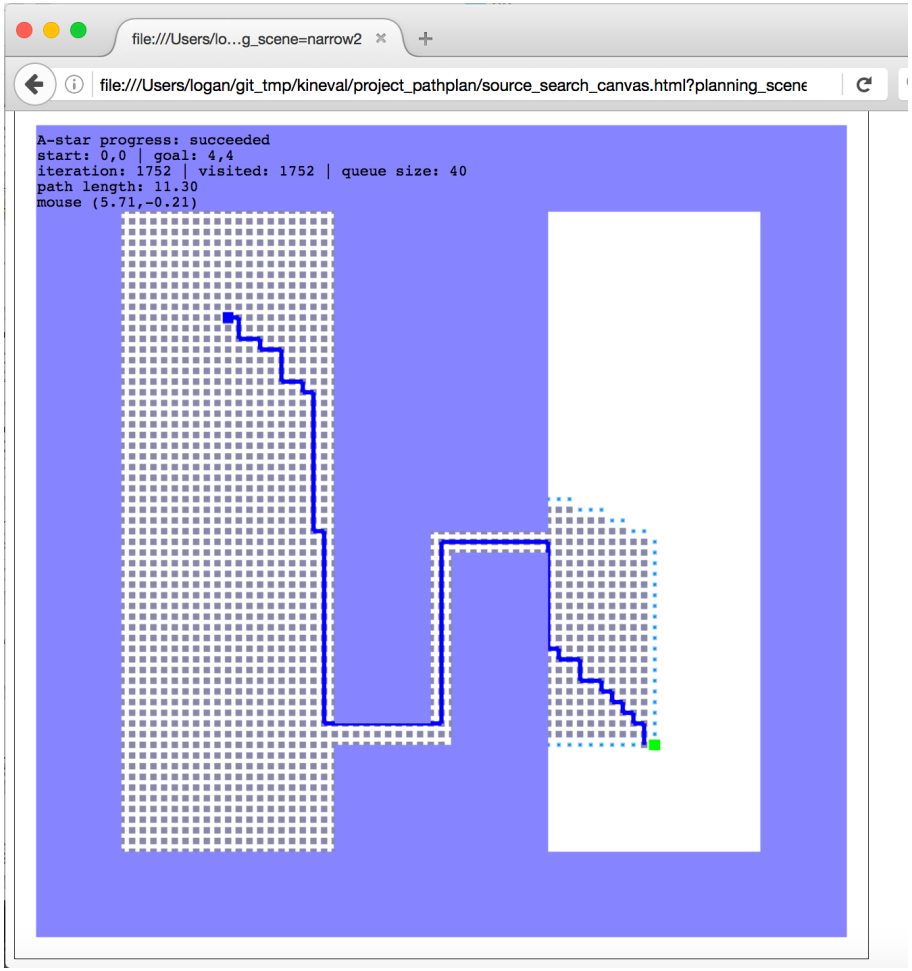
Branch: master New pull request

odestcj Fall 2016 release

- js
- kineval
- project_pathplan
- project_pendularm
- robots
- tutorial_heapsort
- tutorial_js
- worlds
- README.md
- home.html

README.md

Project 1:
2D Path Planning



https://github.com/ohseejay/kineval-stencil-f16

2 commits 1 branch

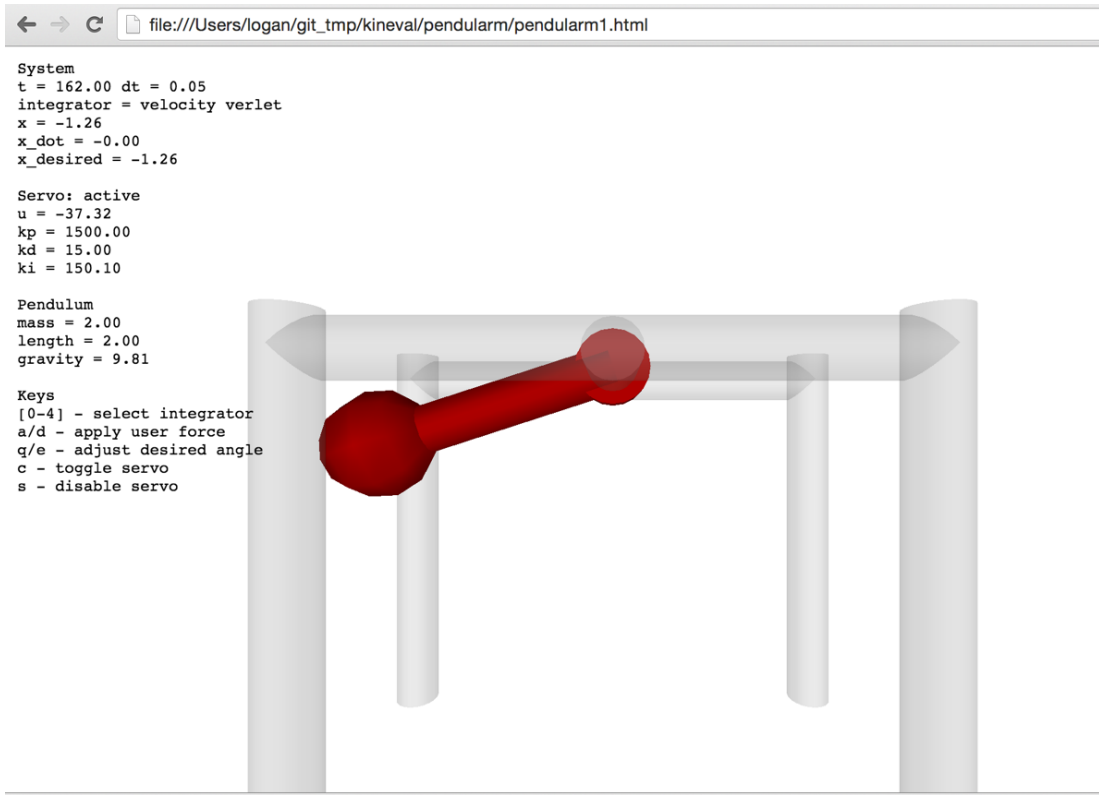
Branch: master New pull request

odestcj Fall 2016 release

- js
- kineval
- project_pathplan
- project_pendularm
- robots
- tutorial_heapsort
- tutorial_js
- worlds
- README.md
- home.html

README.md

Project 2: Pendularm



https://github.com/ohseejay/kineval-stencil-f16

2 commits 1 branch

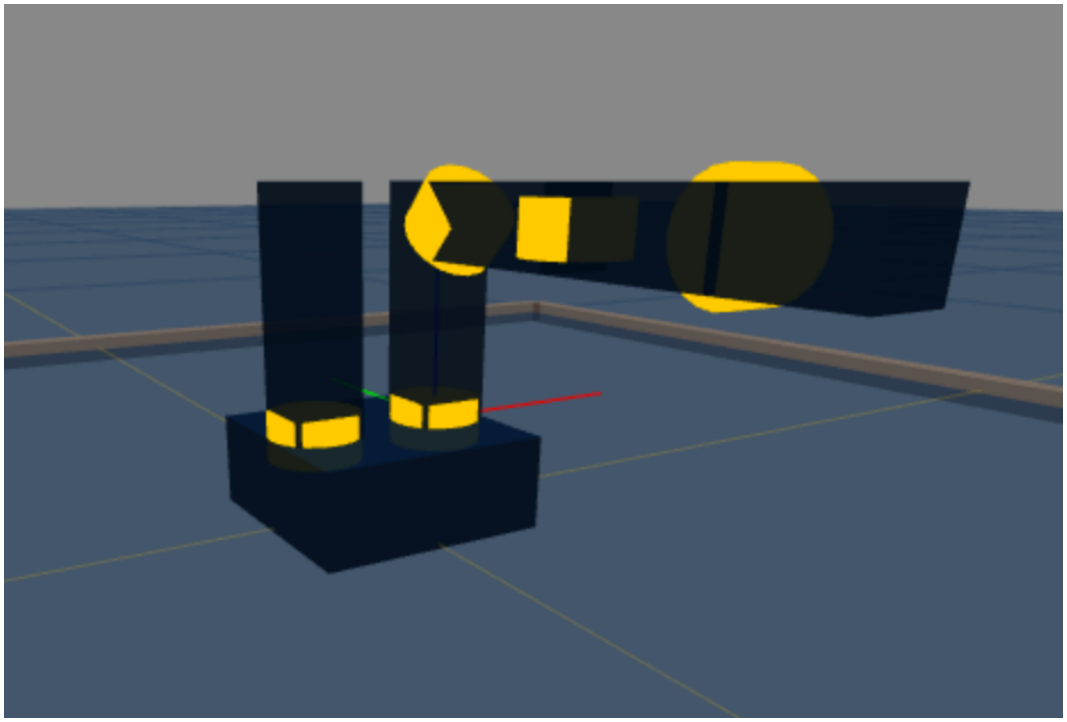
Branch: master New pull request

odestcj Fall 2016 release

- js
- kineval
- project_pathplan
- project_pendularm
- robots
- tutorial_heapsort
- tutorial_js
- worlds
- README.md
- home.html

README.md

Projects 3-6: KinEval



Branching

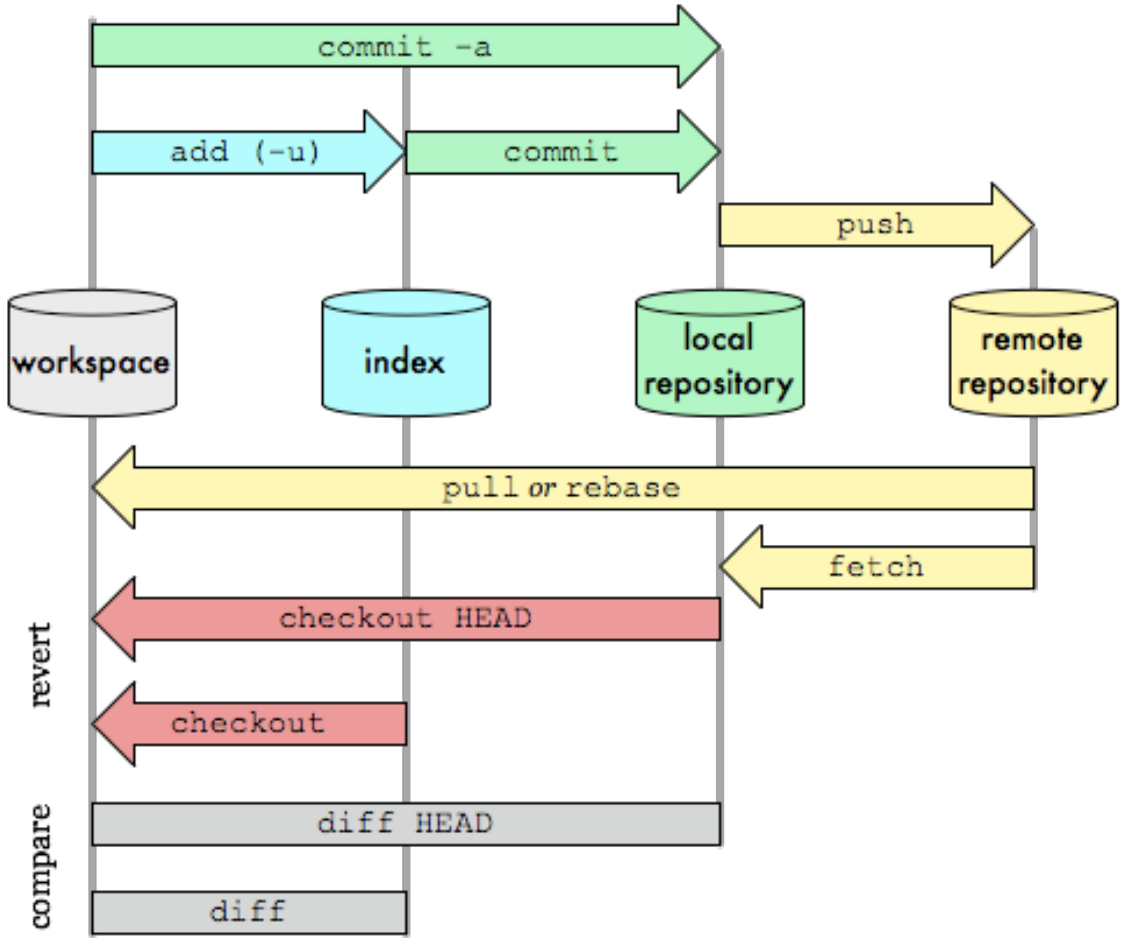
- Helpful for projects with multiple collaborators
- Allows different versions to be modified in parallel
- Branches are tagged with a descriptive name
- A branch is “checked out” into the local workspace
- Conflicts between branches must be resolved in merging (“pull request”)

git basics: commands

- create a local copy of a repository branch:
  - `git clone -b <branch_name> <repo url>`
- switch workspace to a branch of a local repository:
  - `git checkout <branch_name>`

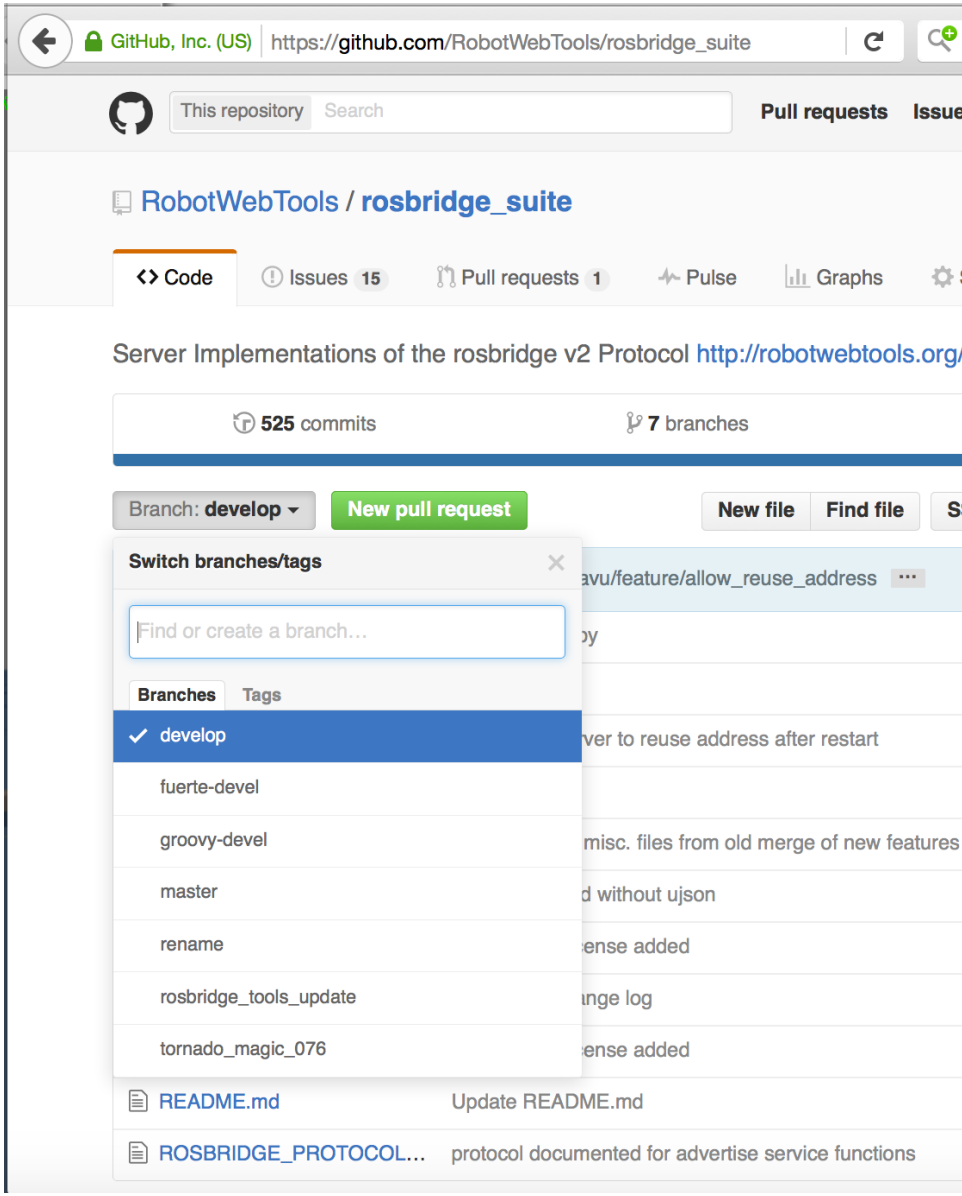
Git Data Transport Commands

http://osteele.com



AutoRob branches

- There will be no collaborator conflicts.
  - You contribute code to the **master** branch. We contribute grading.
- But, you need to keep working while your submitted projects are graded
- You can create a new branch to build upon your code in parallel to grading
- Once complete, your new branch can be merged back into master branch



Foreshadowing: Project 3

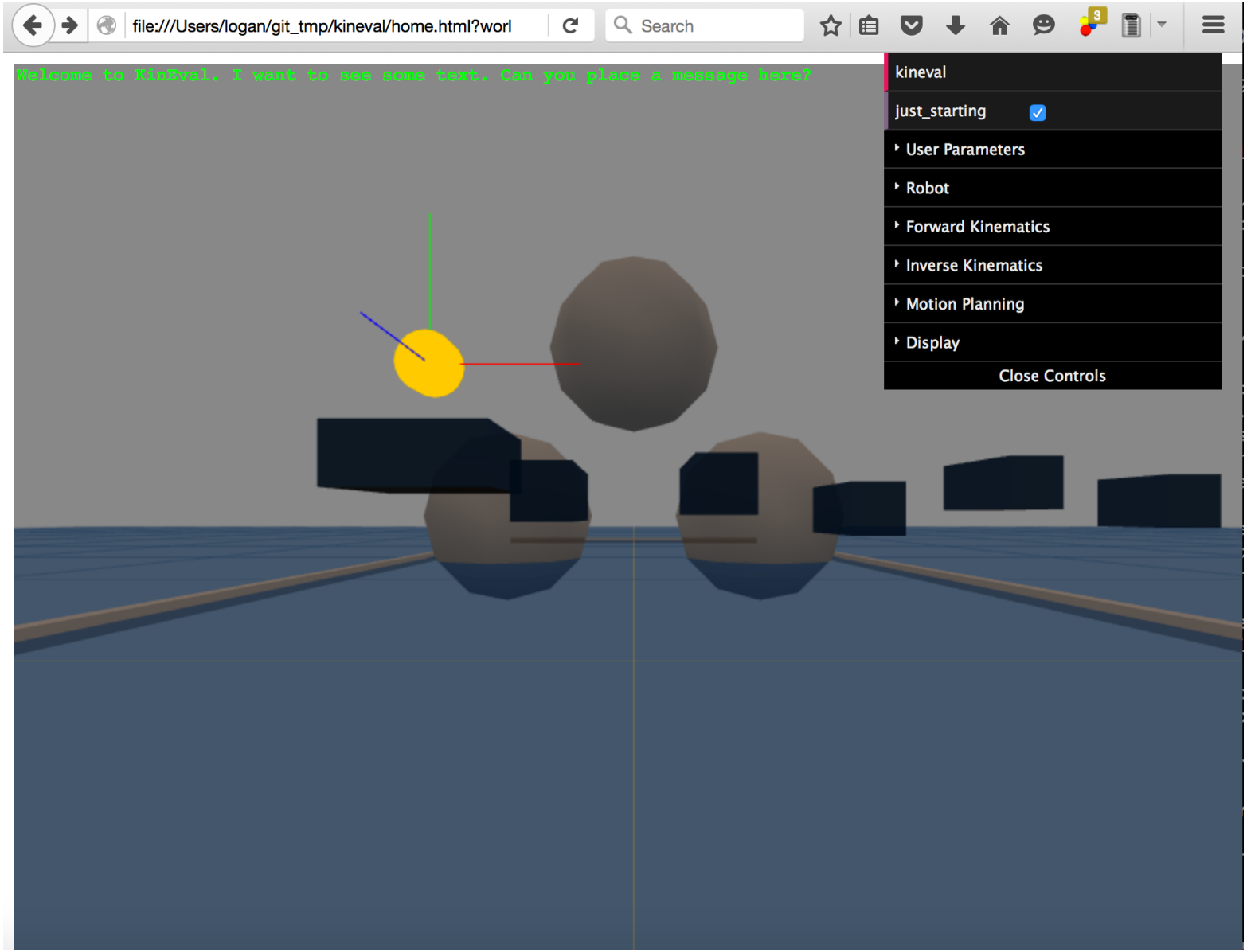
- create a local copy of a repository: `git clone <repo url>`
  - for KinEval, you should now see repo contents in cloned directory
  - view “home.html” in a web browser
  - examine “kineval/kineval_startingpoint.js”
- Note: you might need to clone stencil into a temporary directory and copy into a clone of the repository you have created

Running KinEval

- In Firefox browser, simply open “home.html”
- For other browsers, “home.html” may need to be served to avoid throwing a “security” error
  - If you have ***python***, run the simpleHTTPServer from the directory containing “home.html”
    - python -m SimpleHTTPServer
    - point browser to http://localhost:8000/

Running KinEval

- In Firefox browser, simply open “home.html”
- For other browsers, “home.html” may need to be served to avoid throwing a “security” error
  - If you have ***nodejs***, install and run simple-server module from the directory containing “home.html”
    - npm install simple-server
    - node simple-server
    - point browser to http://localhost:3000/home.html


file:///Users/logan/git_tmp/kineval/home.html?worl
Search
Welcome to KinEval. I want to see some text. Can you place a message here?
kineval
just_starting
User Parameters
Robot
Forward Kinematics
Inverse Kinematics
Motion Planning
Display
Close Controls


Happy hacking!