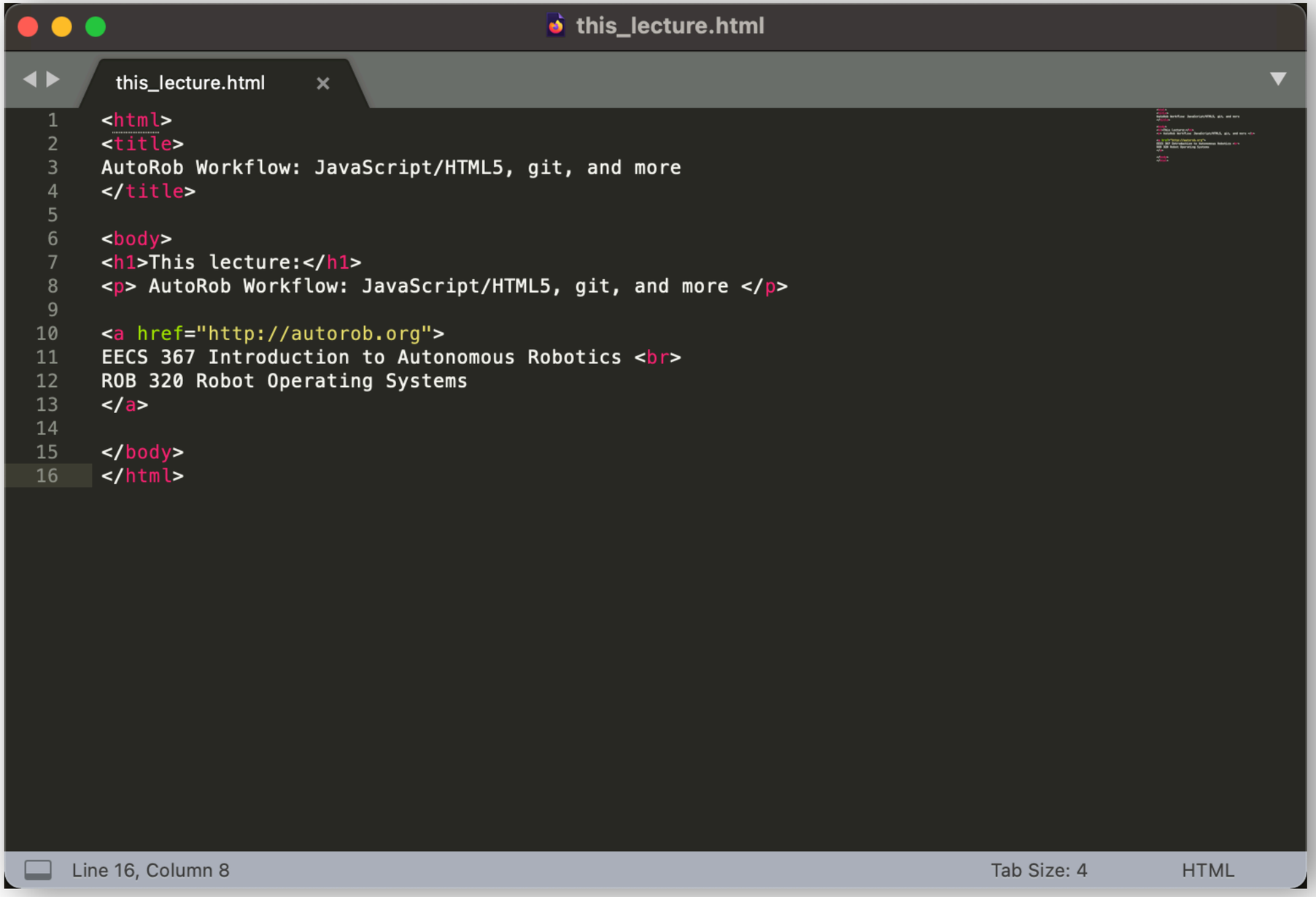
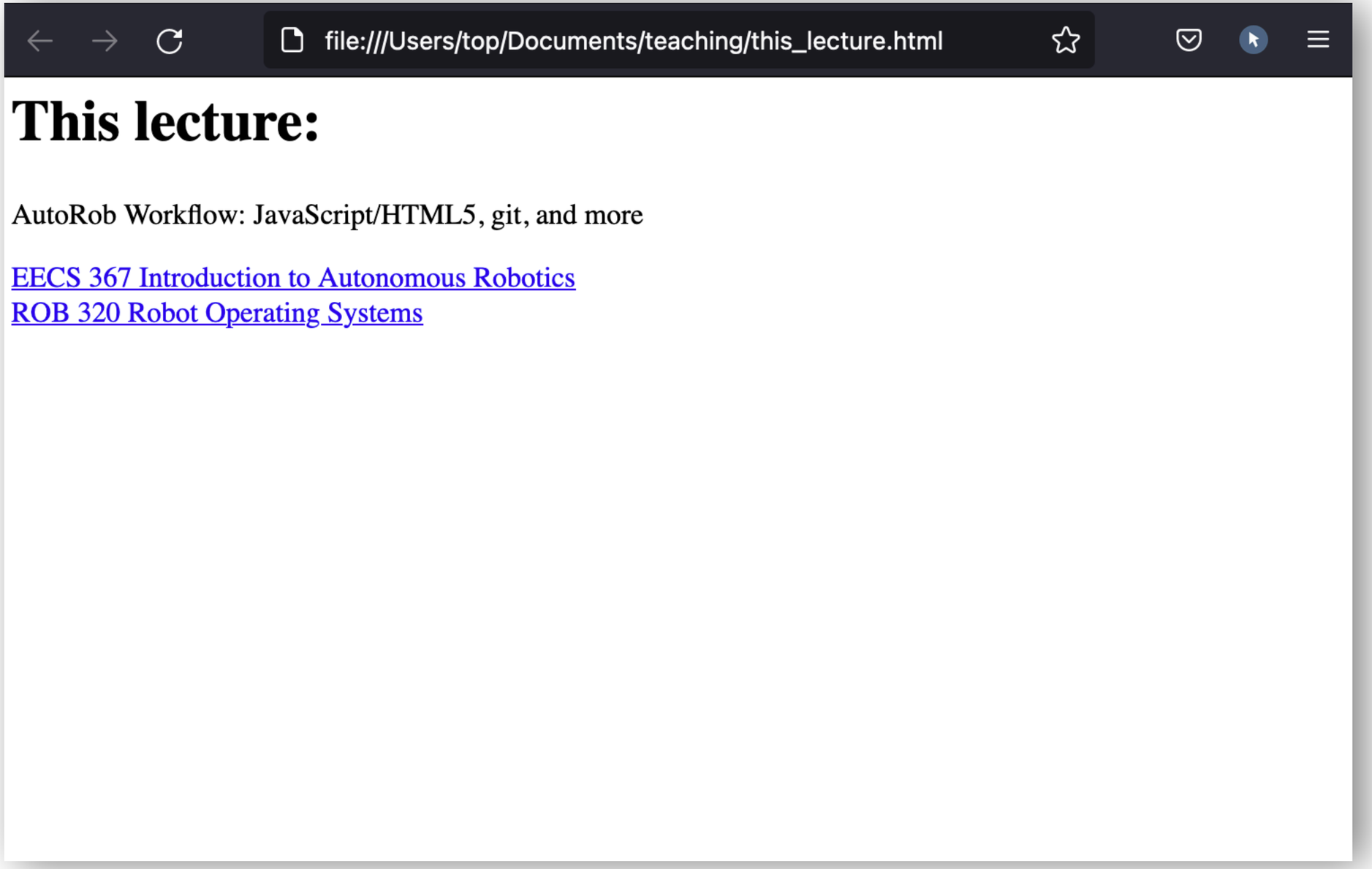
```
<html>
<title>
AutoRob Workflow: JavaScript/HTML5, git, and more
</title>

<body>
<h1>This lecture:</h1>
<p> AutoRob Workflow: JavaScript/HTML5, git, and more </p>

<a href="http://autorob.org">
EECS 367 Introduction to Autonomous Robotics <br>
ROB 320 Robot Operating Systems
</a>

</body>
</html>
```

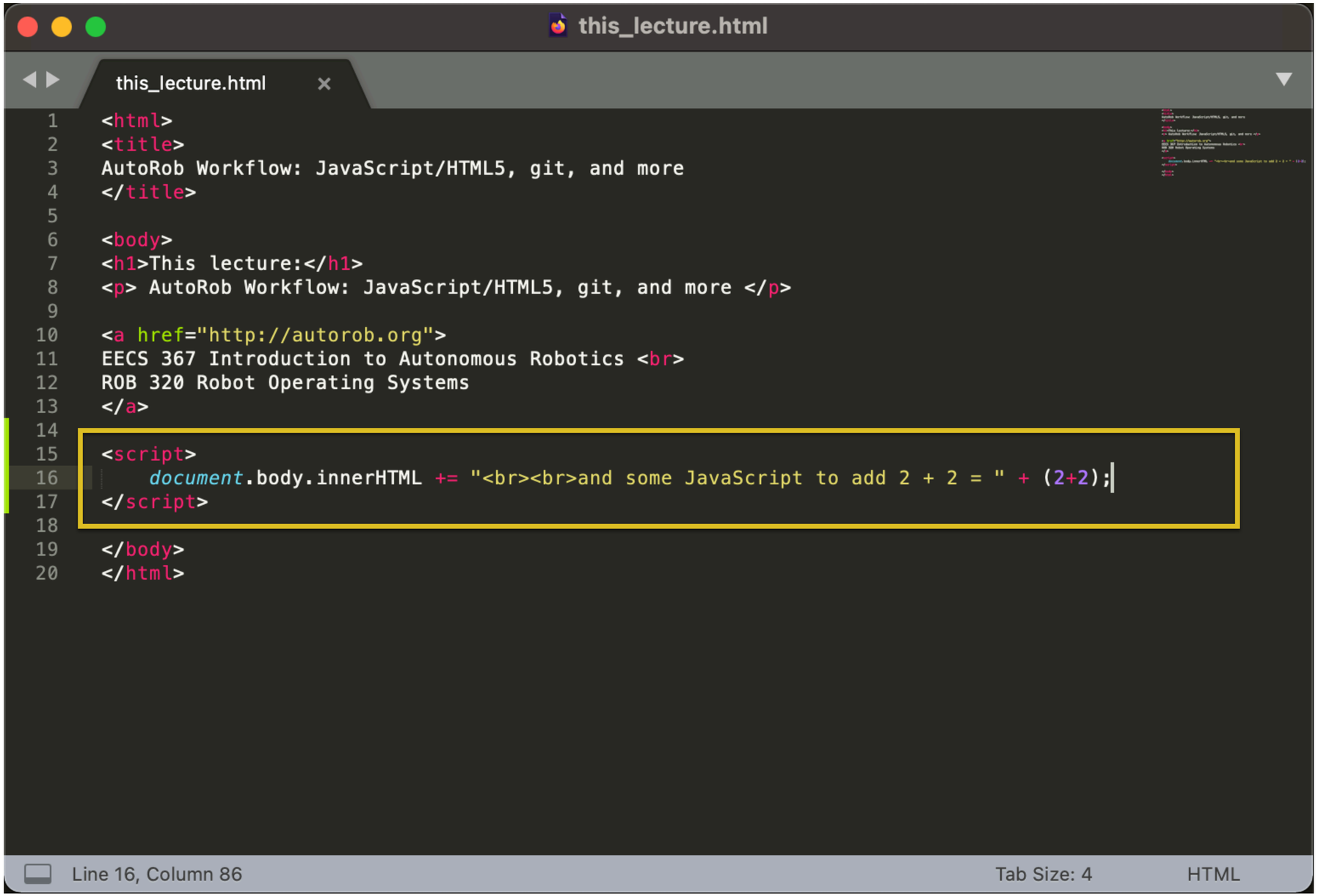


sublime text editor

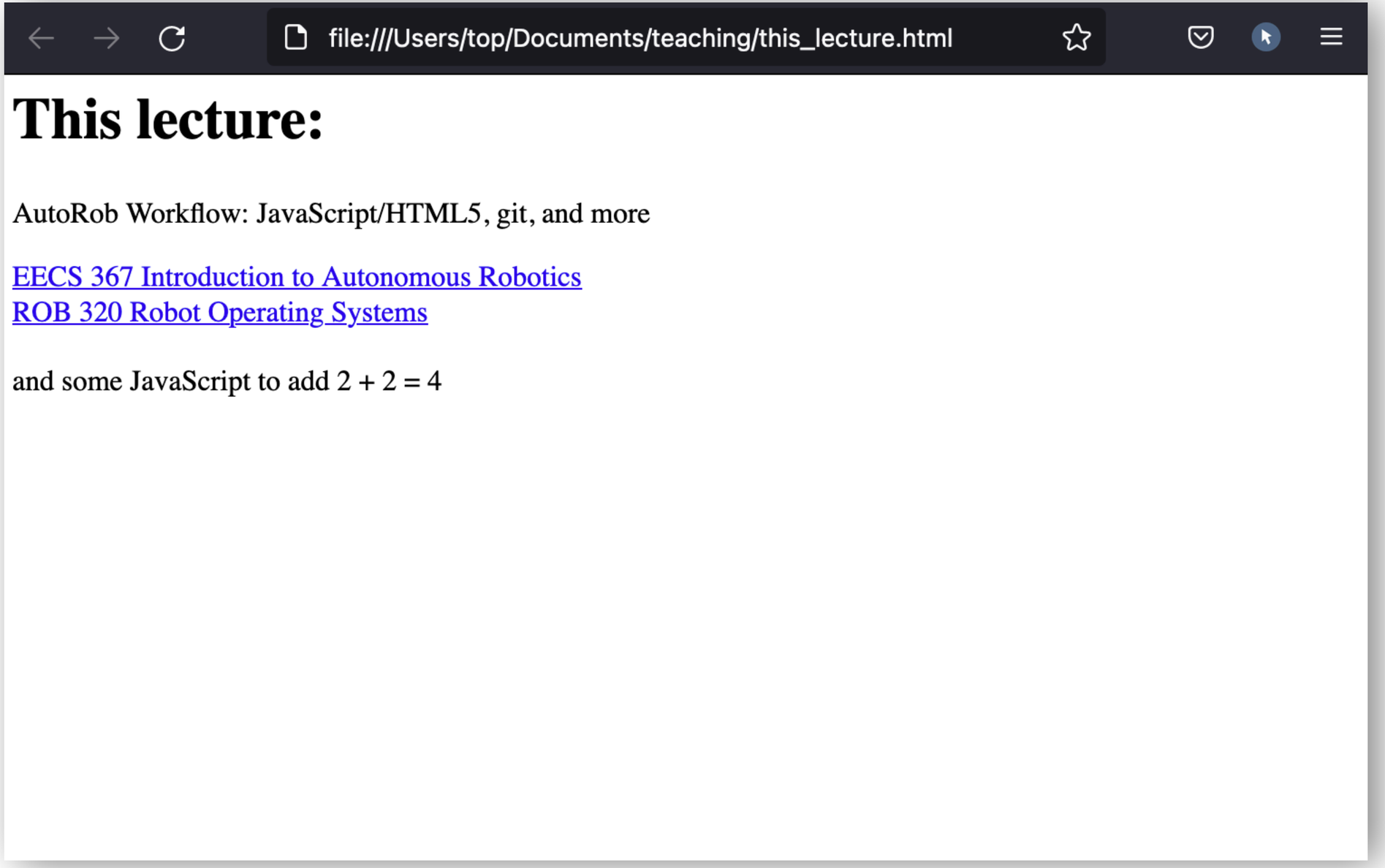
Put this text into a file named “this_lecture.html”



Open “this_lecture.html” in a web browser



Change “this_lecture.html” with JavaScript code to execute

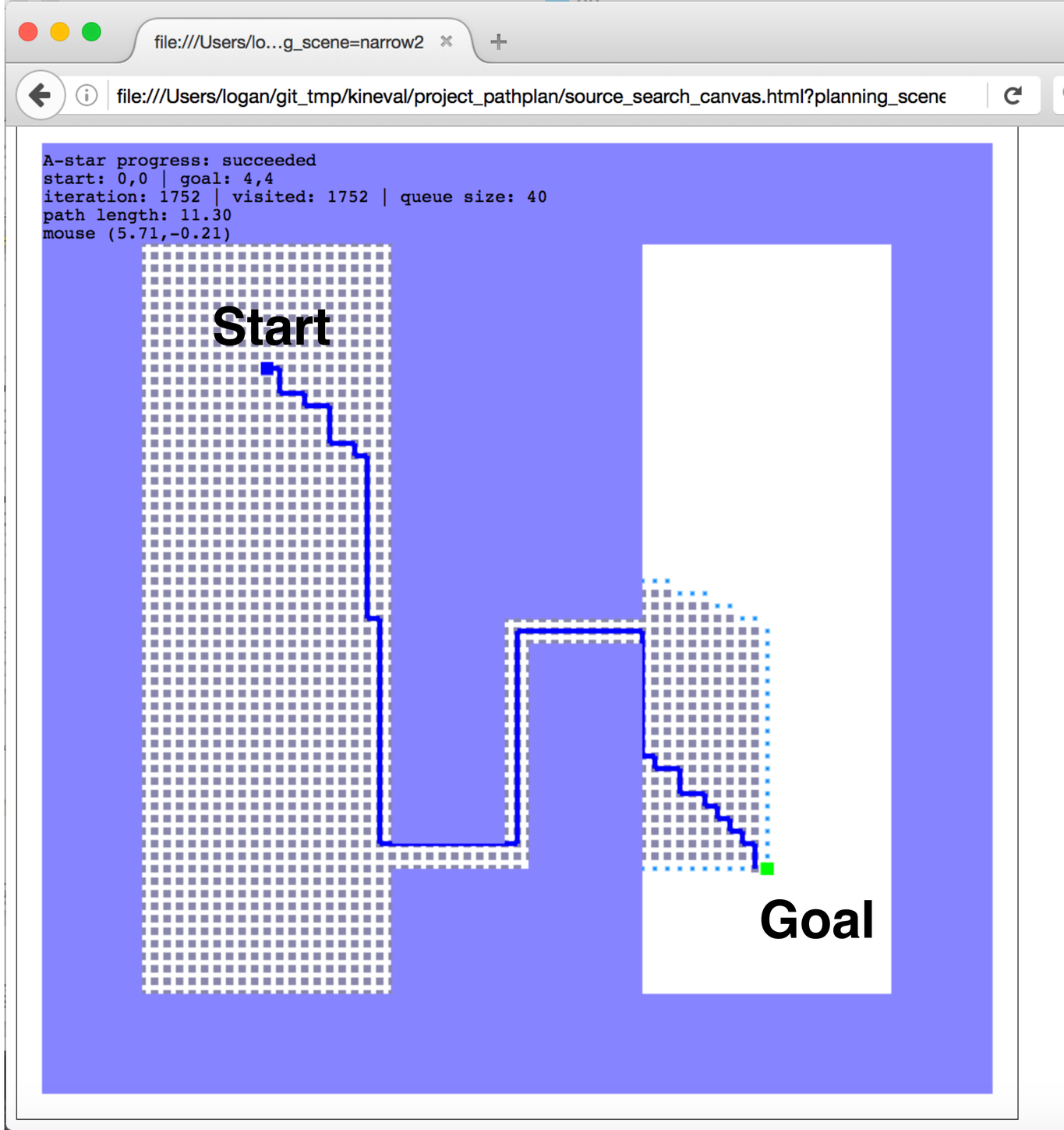


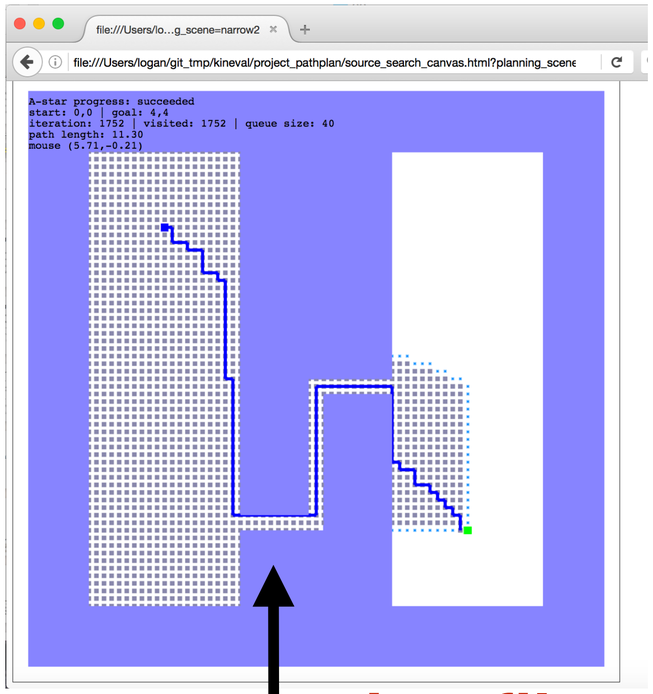
Reload “this_lecture.html” in browser to see result of this code

Similar workflow for Project 1

Project 1:
2D Path Planning

- A-star algorithm for search in a specified 2D world
- Implement planner in JavaScript/HTML5 file
- View planner run time behavior in web browser
- Submit by committing code to your git repository

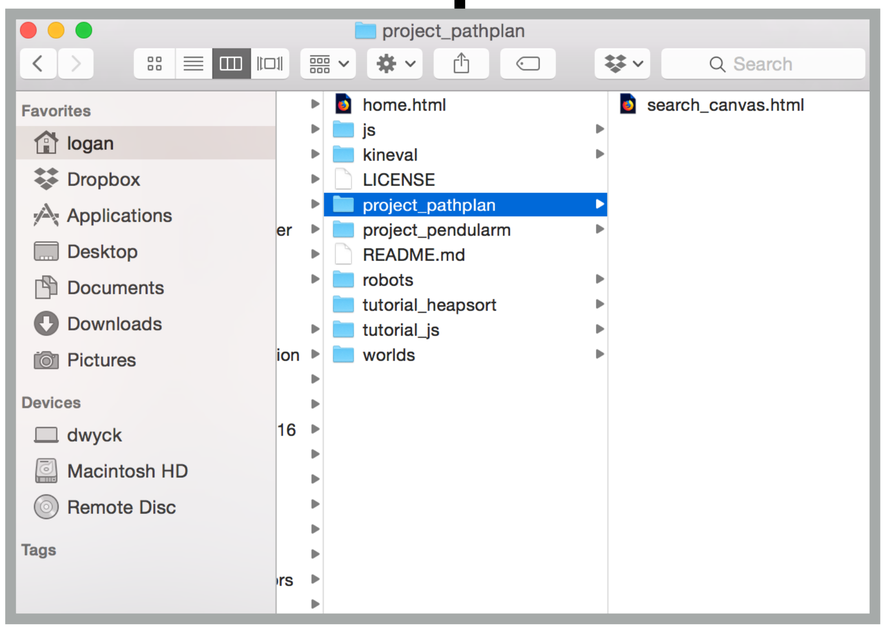




Browser

See HTML and JS code working

view file

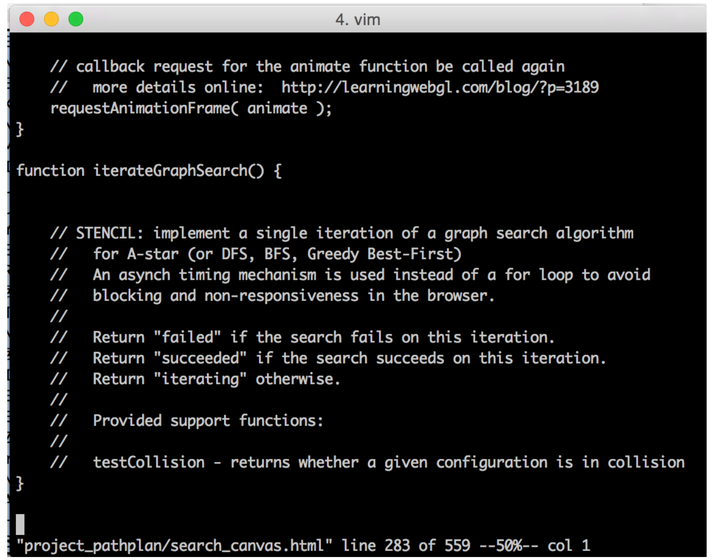


Source code

HTML and JS files containing your code

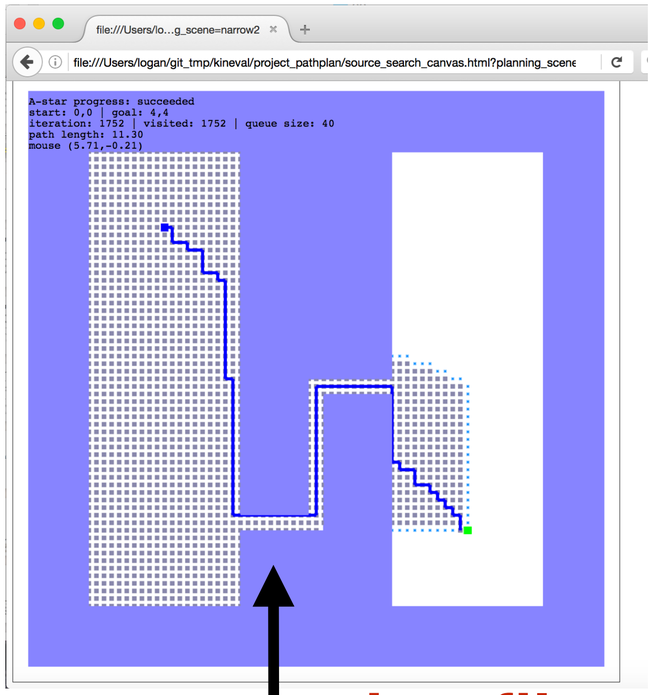
Text editor

Make changes to HTML and JS code

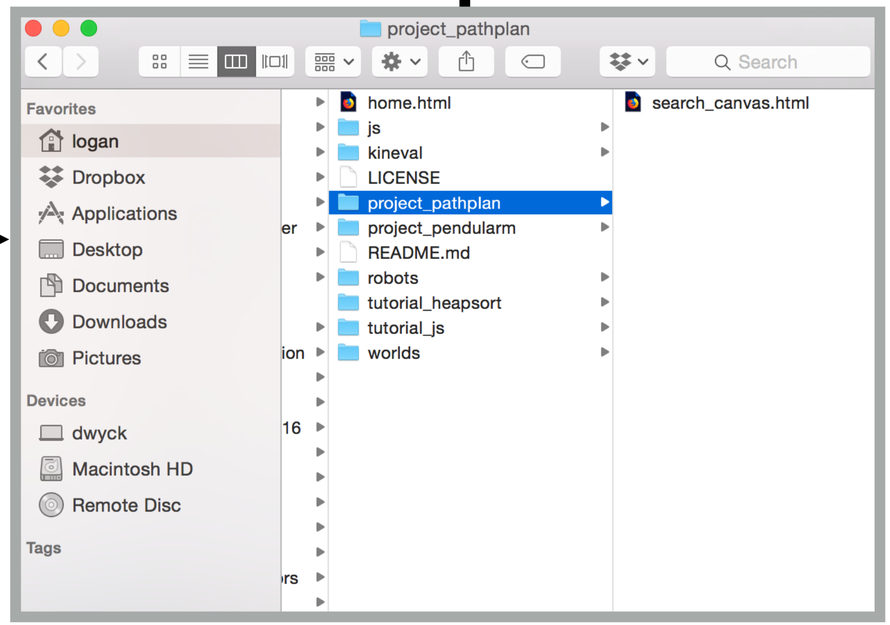


open file

save file



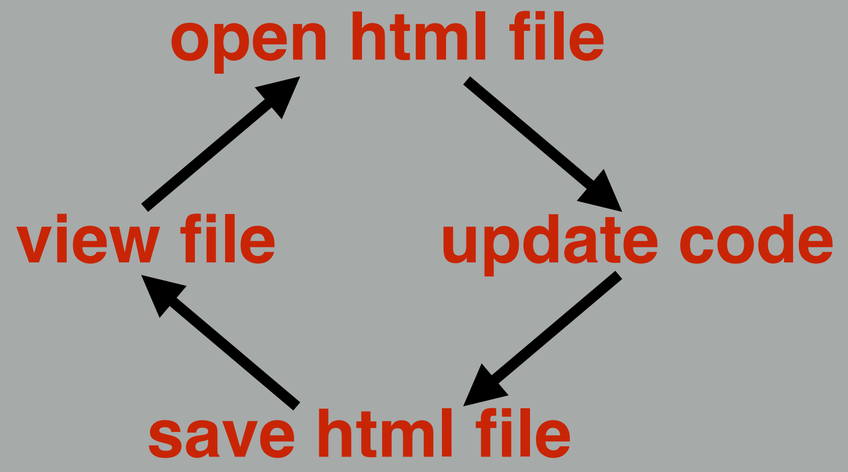
view file



Source code

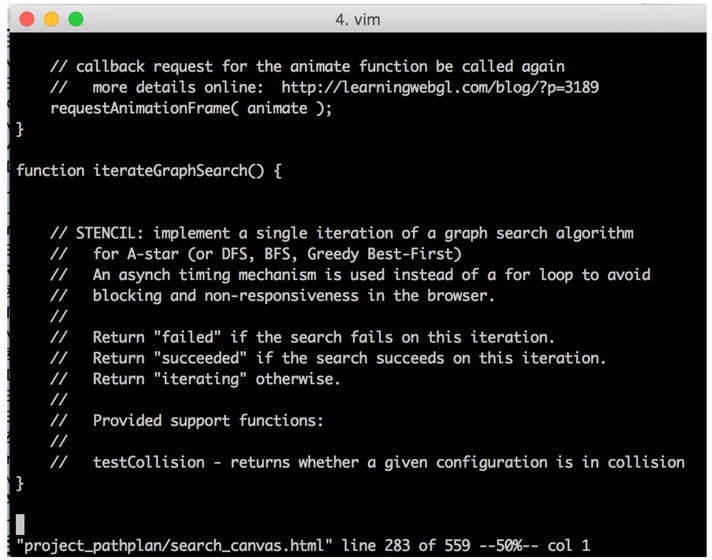
HTML and JS files containing your code

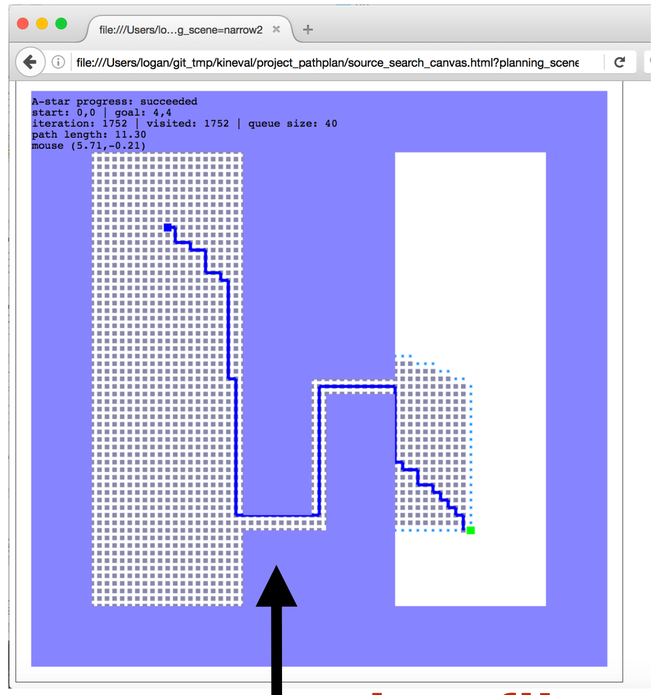
Coding process



Text editor

Make changes to HTML and JS code

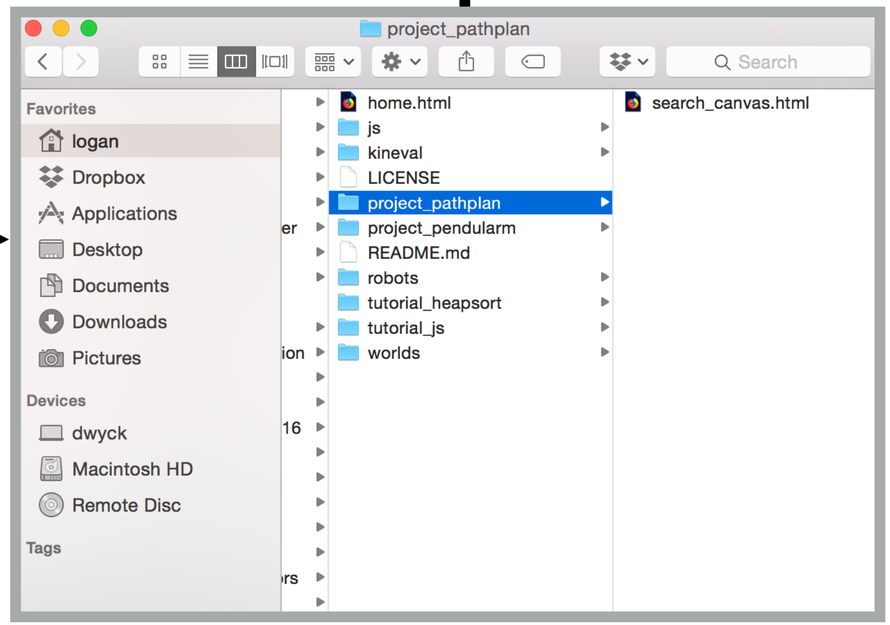




view file

open file

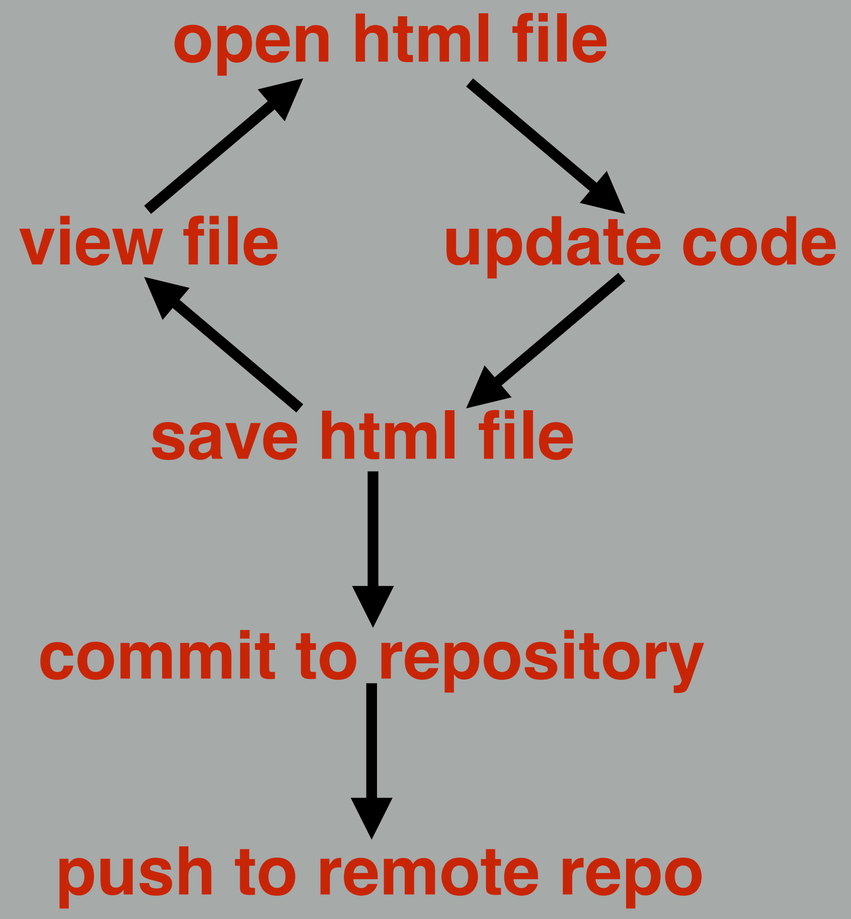
save file



Source code

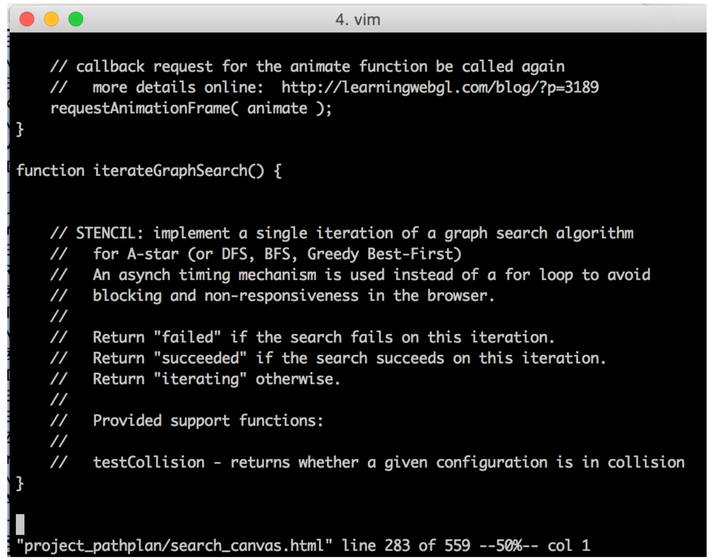
HTML and JS files containing your code

Coding process



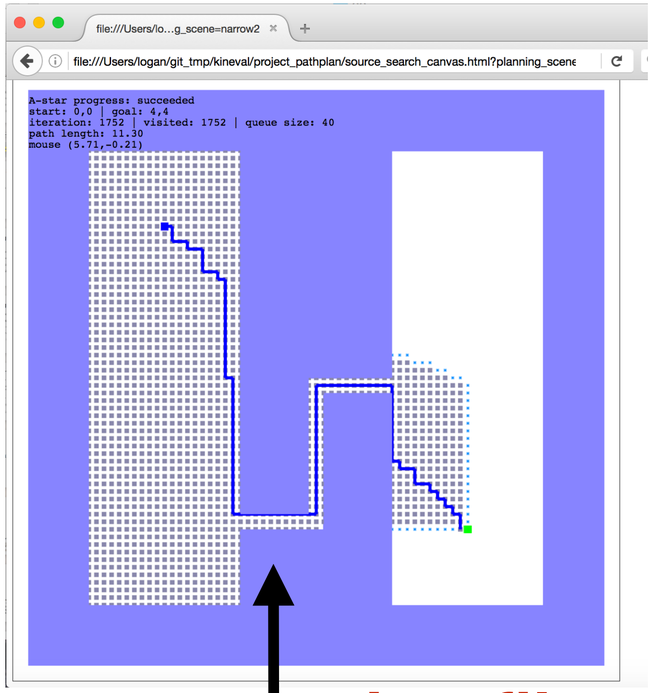
Text editor

Make changes to HTML and JS code

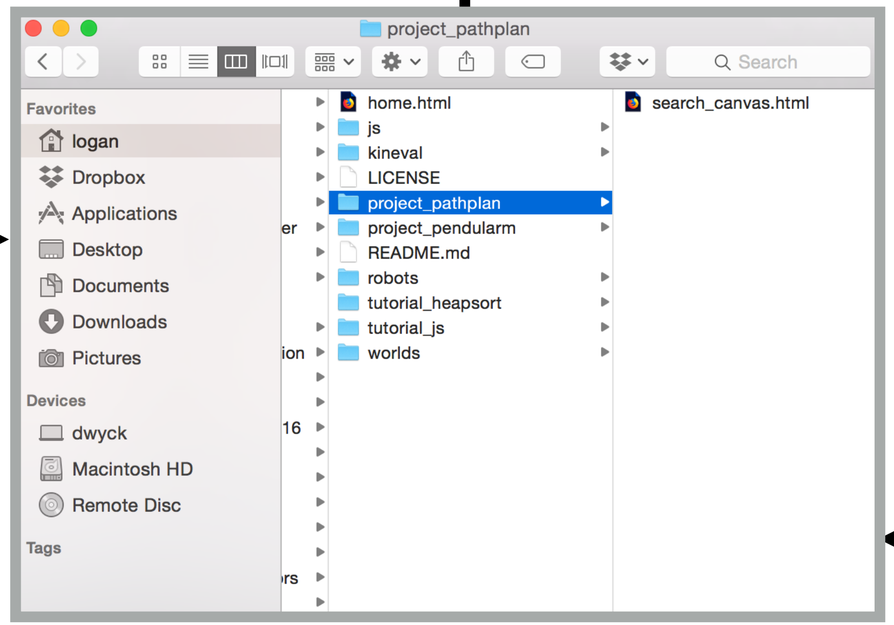


open file

save file



view file

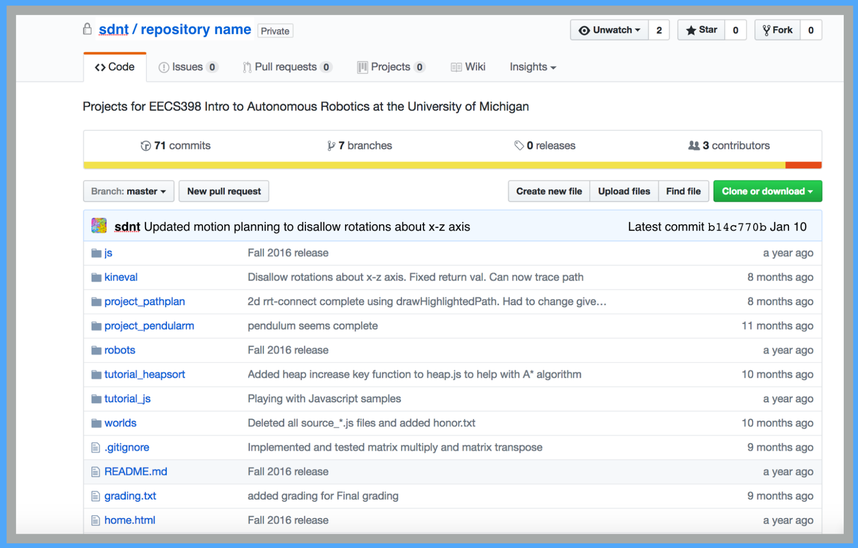


Source code

HTML and JS files containing your code

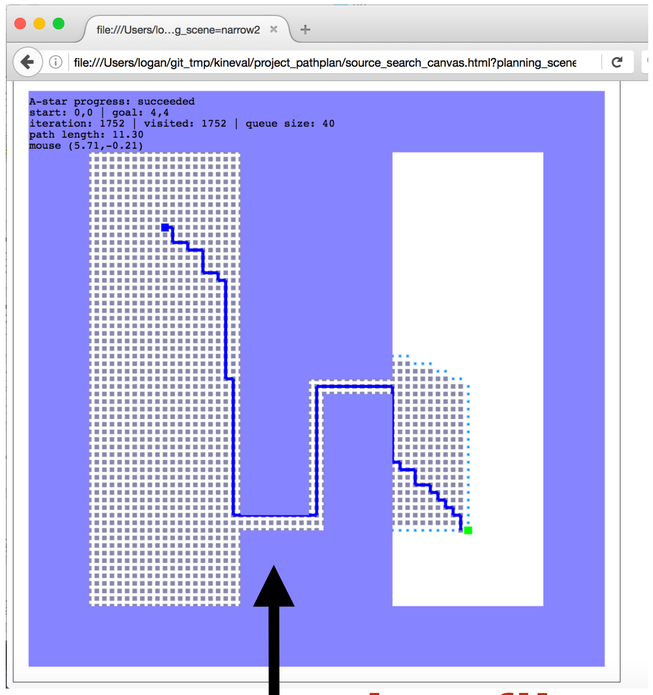
git repository

store history of code changes and pull grading

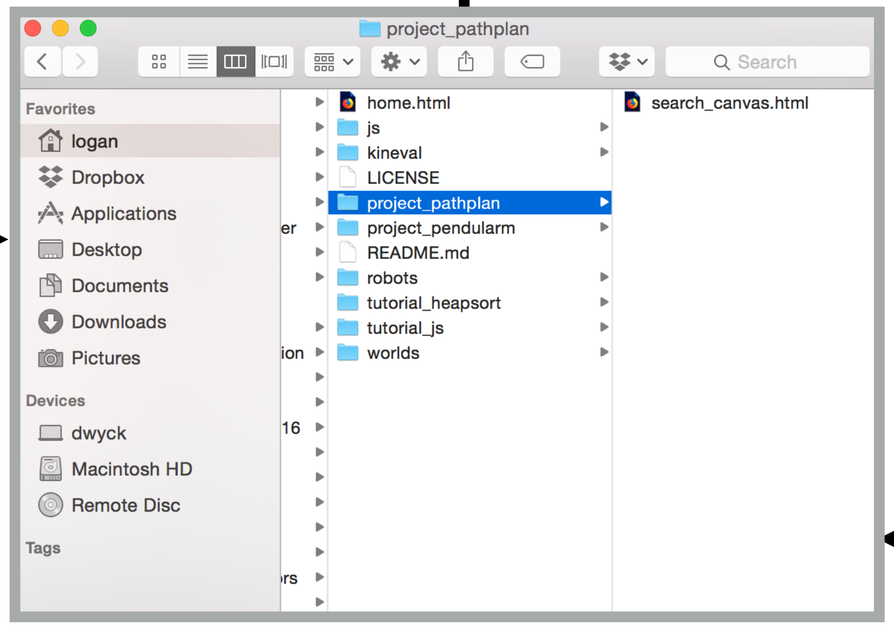


push repo

Pull repo

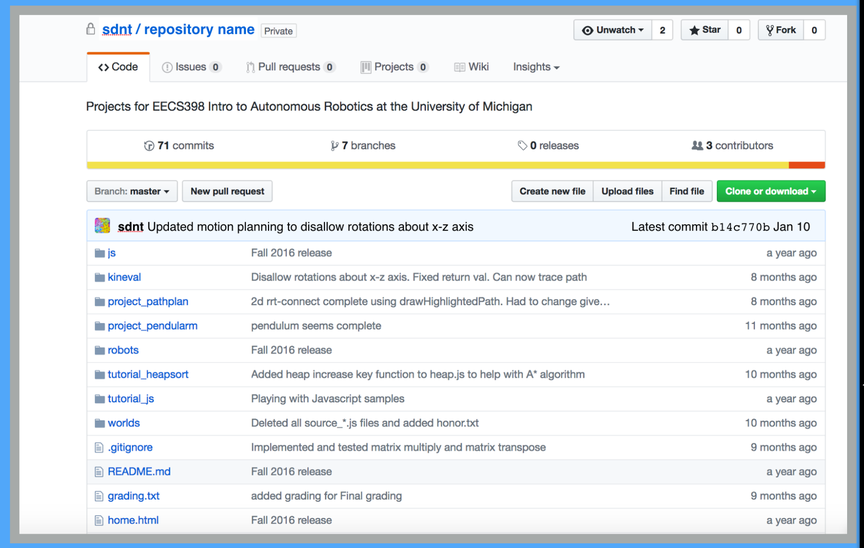


view file



git repository

store history of code changes and pull grading

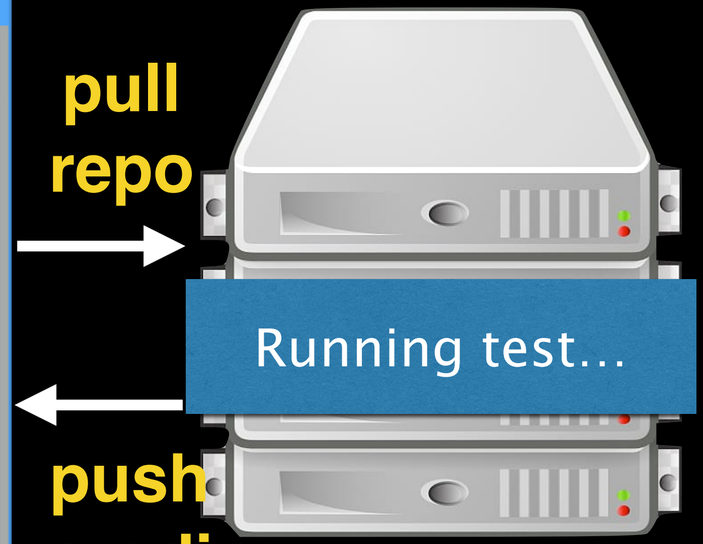


push repo

Pull repo

CI Grader

Continuous Integration testing of project code

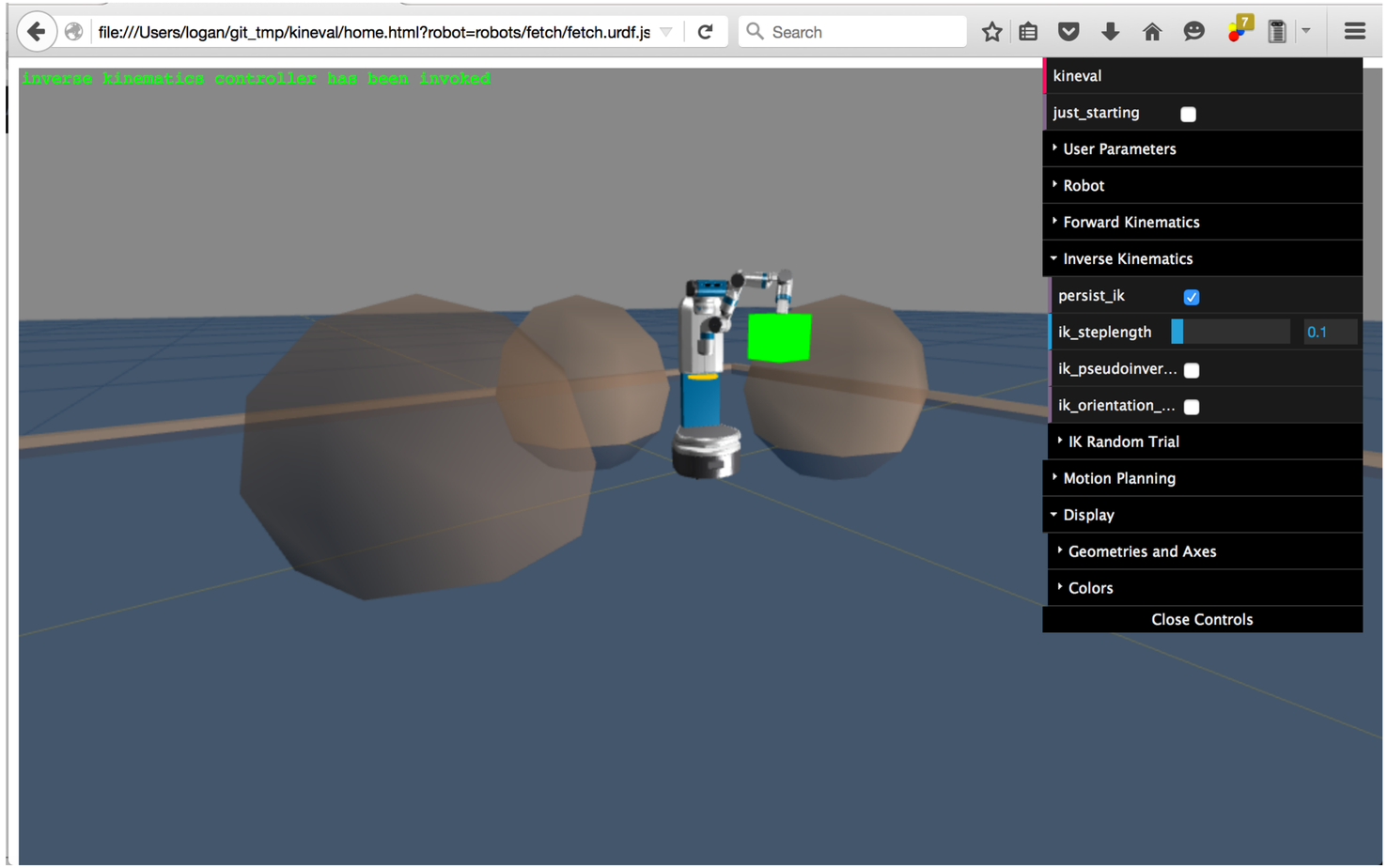


This is new to AutoRob for remote work.

All grades reviewed by course staff

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, …)



AutoRob

Introduction to Autonomous Robotics
Michigan EECS 367

Robot Operating Systems
Michigan ROB 320

Winter 2022

Flipped Classroom Hybrid COVID-19 Edition

https://github.com/autorob-WN22/kineval-stencil

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

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

Watch 0 Fork 7 Star 0

https://github.com/autorob-WN22/kineval-stencil

Code Pull requests Actions Projects Wiki Security Insights

master 1 branch 0 tags Go to file Add file Code Use this template

This branch is 1 commit ahead of autorob:master. Contribute

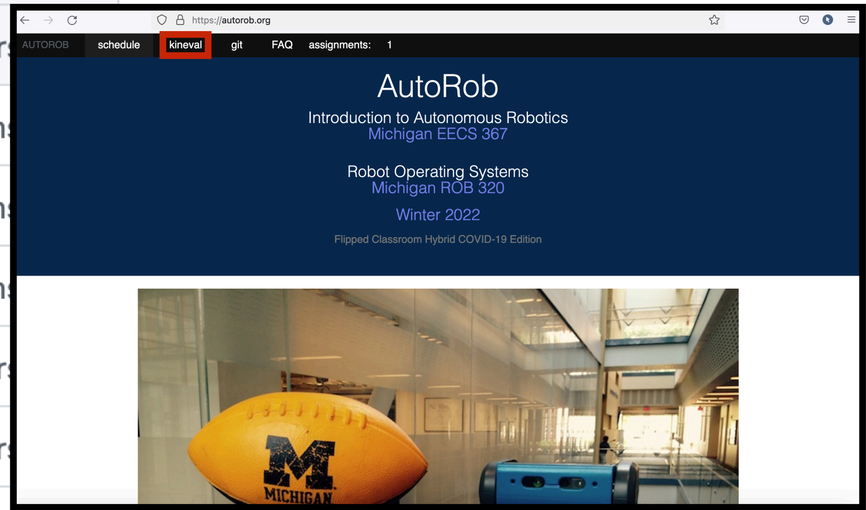
tonyop Update LICENSE 26c8fb8 10 days ago 23 commits

js	initial commit Fall 2018	3 years
kineval	Add matrix requirement for IK	15 months
project_pathplan	Makes assignment 6 drawing work more like assignment 1 for...	14 months
project_pendularm	fixed control set to 0 and 2d array problem in pendulum2.html	16 months
robots	initial commit Fall 2018	3 years
tutorial_heapsort	initial commit Fall 2018	3 years

About

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

Readme



Why JavaScript/HTML5?

Why JavaScript/HTML5?

Spectrum of programming languages

Spectrum of programming languages

C

C++ (maybe)

Matlab
(for numerics)

Python

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

Dissemination

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 (sandboxed run time 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 (JavaScript is C without discipline)
- Cryptic debugging messages

JavaScript is C without discipline

- If you are fluent in C or a “classical” programming language, JavaScript has prototyped objects and minor syntactical differences
    - your instincts and stackoverflow should be enough
- If you are familiar with Matlab or a “scientific” language, you may need time to become familiar with syntax and some data structures
- If you are new to programming: EECS 402 or EECS 280 or ROB 502 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

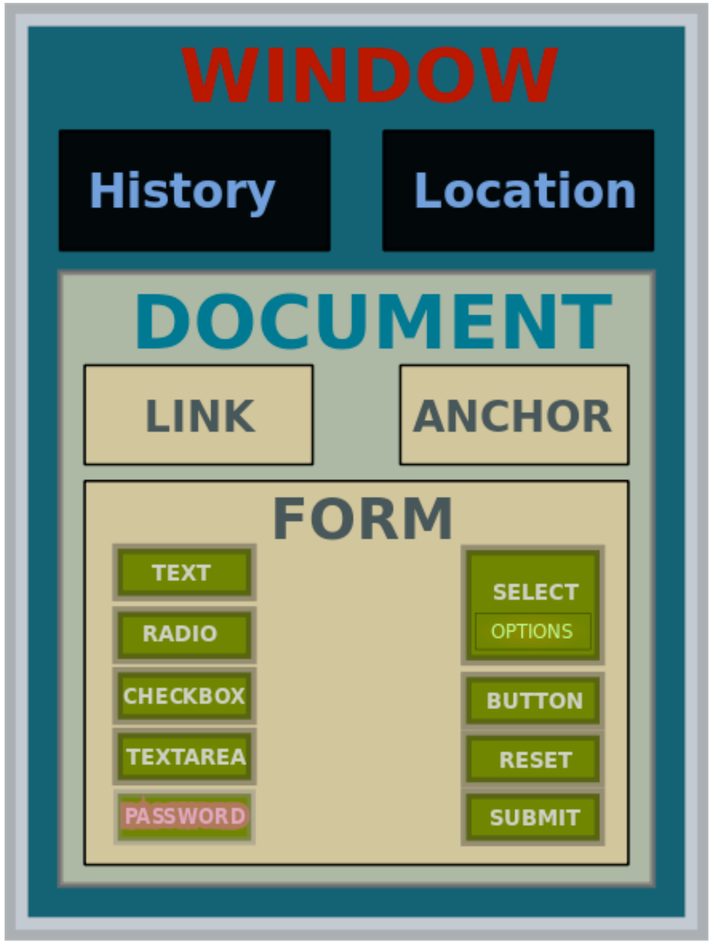
What is JavaScript/HTML5?

- **HyperText Markup Language** (HTML5): a markup language for expressing web pages as documents
  - Web browsers read HTML files and render to display
  - 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)

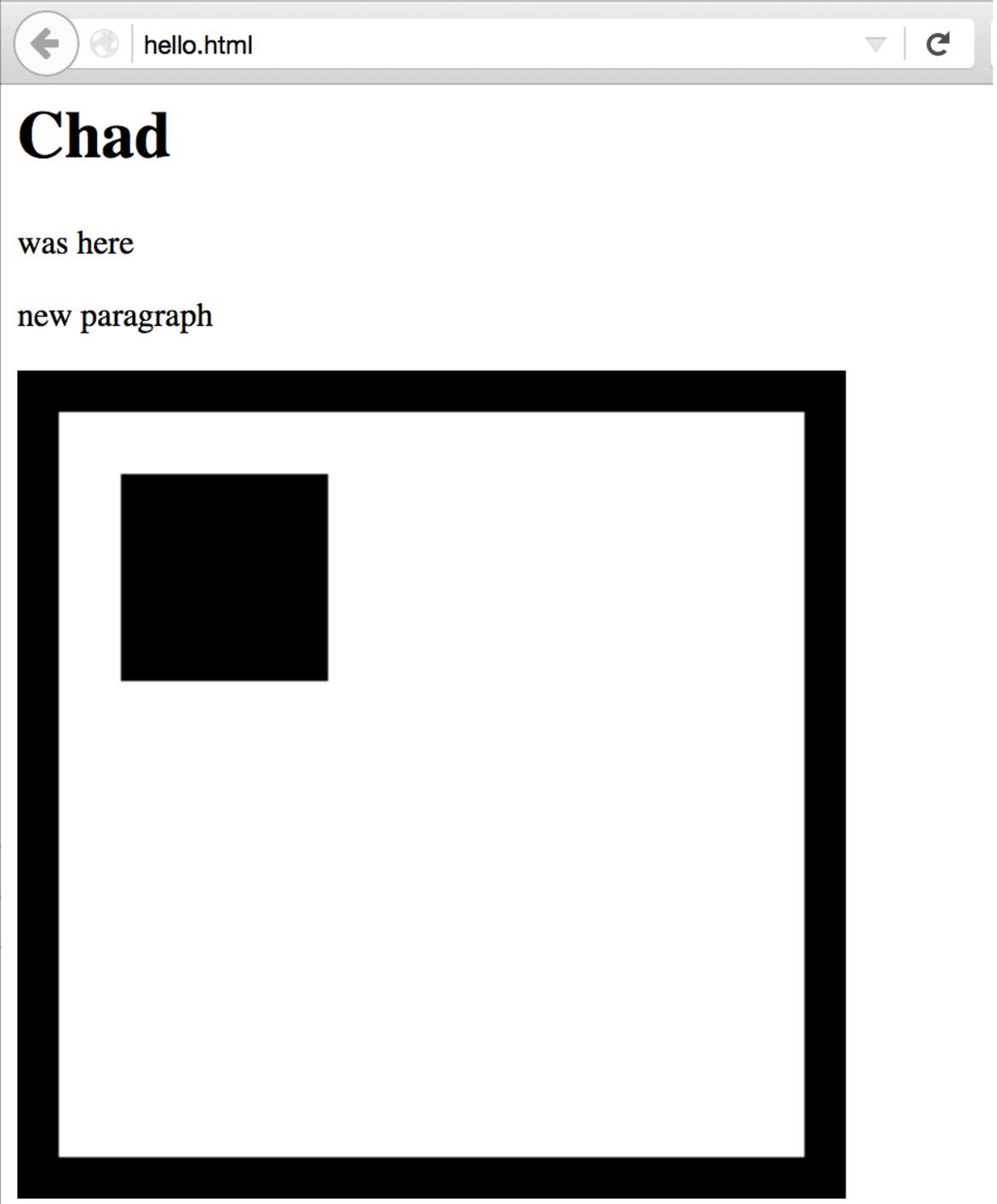
- HTML document defined by elements nested within markup tags (e.g.,
 <h1>text</h1>)
- DOM provides programmatic access to these elements as JavaScript objects
- DOM provides 2 important global objects:
  - "**window**" is the DOM root for a browser tab (a global variable "x" is actually "window.x")
  - "**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

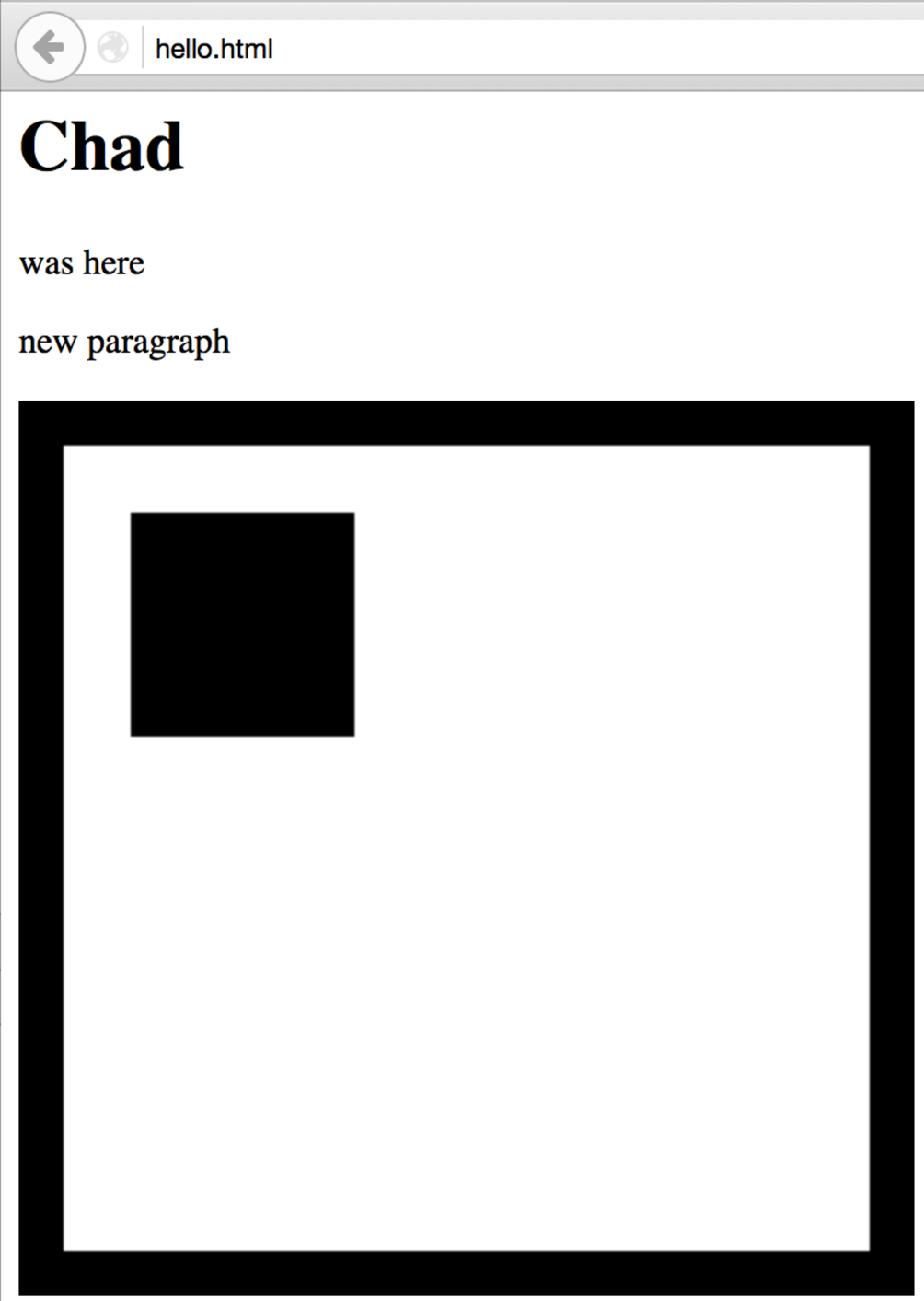
Preview slides from lectures during the Fall 2020 offering of AutoRob are provided. These preview slides will be replaced with recorded lectures for Winter 2022 as the videos become available.

Date	Topic	Reading	Project I Quiz
Jan 5	Initialization: "So, where is my robot?", "What is a Robot OS?", Course administration and logistics [Lecture Video] [Session recording (UM only)] What is a robot?: Brief history and definitions for robotics	Spong Ch.1 Corke Ch.1	Setup git repository Out: Path Planning
Jan 7	Lab Session: Git-ing started with git, JavaScript, and KinEval [Session recording (UM only)]		
	JavaScript/HTML5 hello example		
Jan 10	shortest paths, A-star, Priority queues and binary heaps [Lecture Video]	Wikipedia	
Jan 12	JavaScript and AutoRob workflow: Project workflow with git, JS/HTML5 tutorial, Document Object Model, Version Control, LaTeX math mode, Licensing, Michigan Honor License	Crockford, HTML Sandbox, hello.html, JavaScript by Example, hello_anim, hello_anim_text	
Jan 14	367 Lab: KinEval Path Planning code overview		
	Week 3		
Jan 17	**No course meeting** - Martin Luther King, Jr. Day UM Martin Luther King Jr. Symposium Help broaden participation in computing and robotics		
Jan 19	Dynamical Simulation: Simple pendulum, Lagrangian equation(s) of motion, Initial value problem, Explicit integrators: Euler, Verlet, and	Spong Ch.7 I Corke Ch.9 Euler's Method	



hello example

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

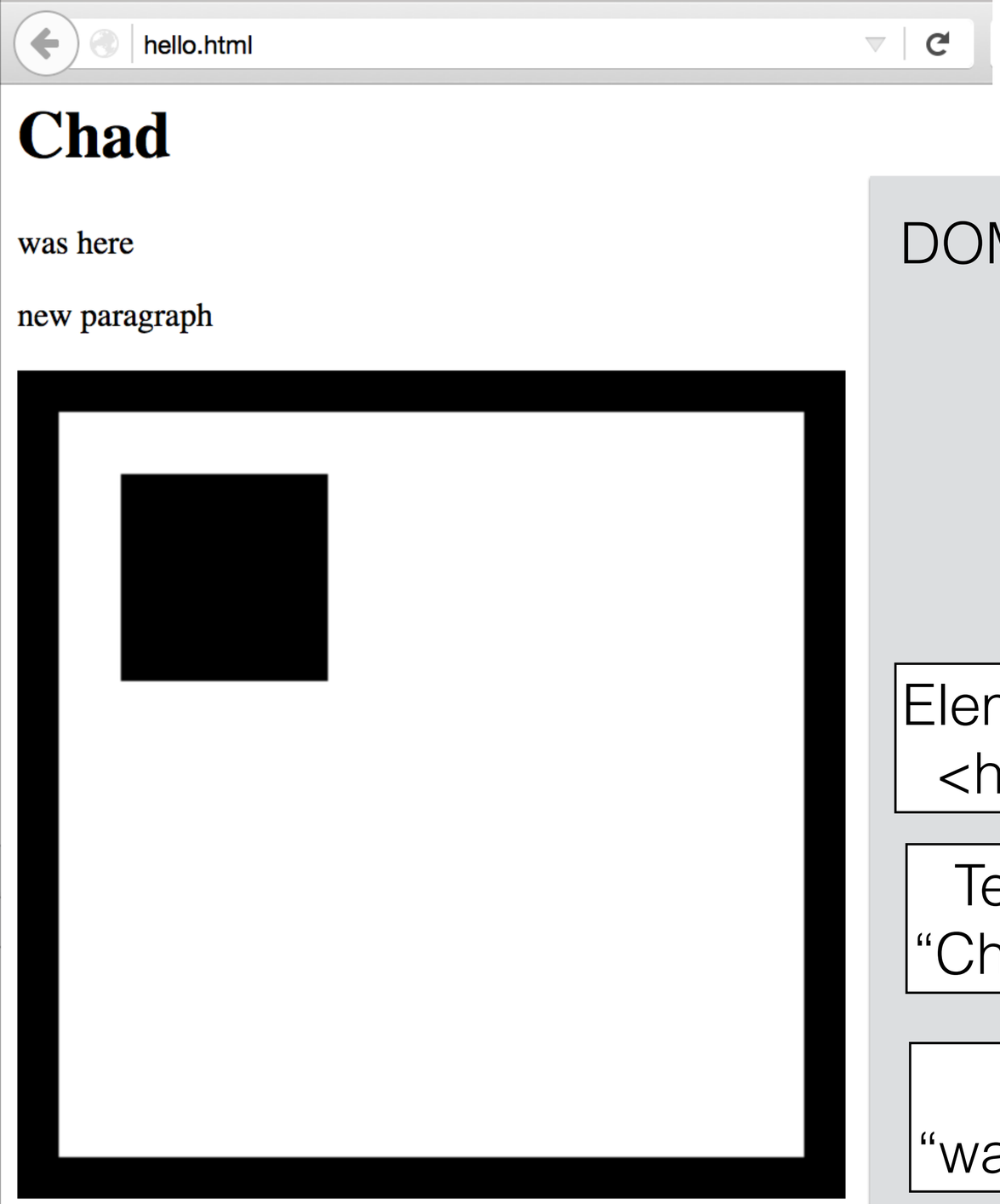


hello example

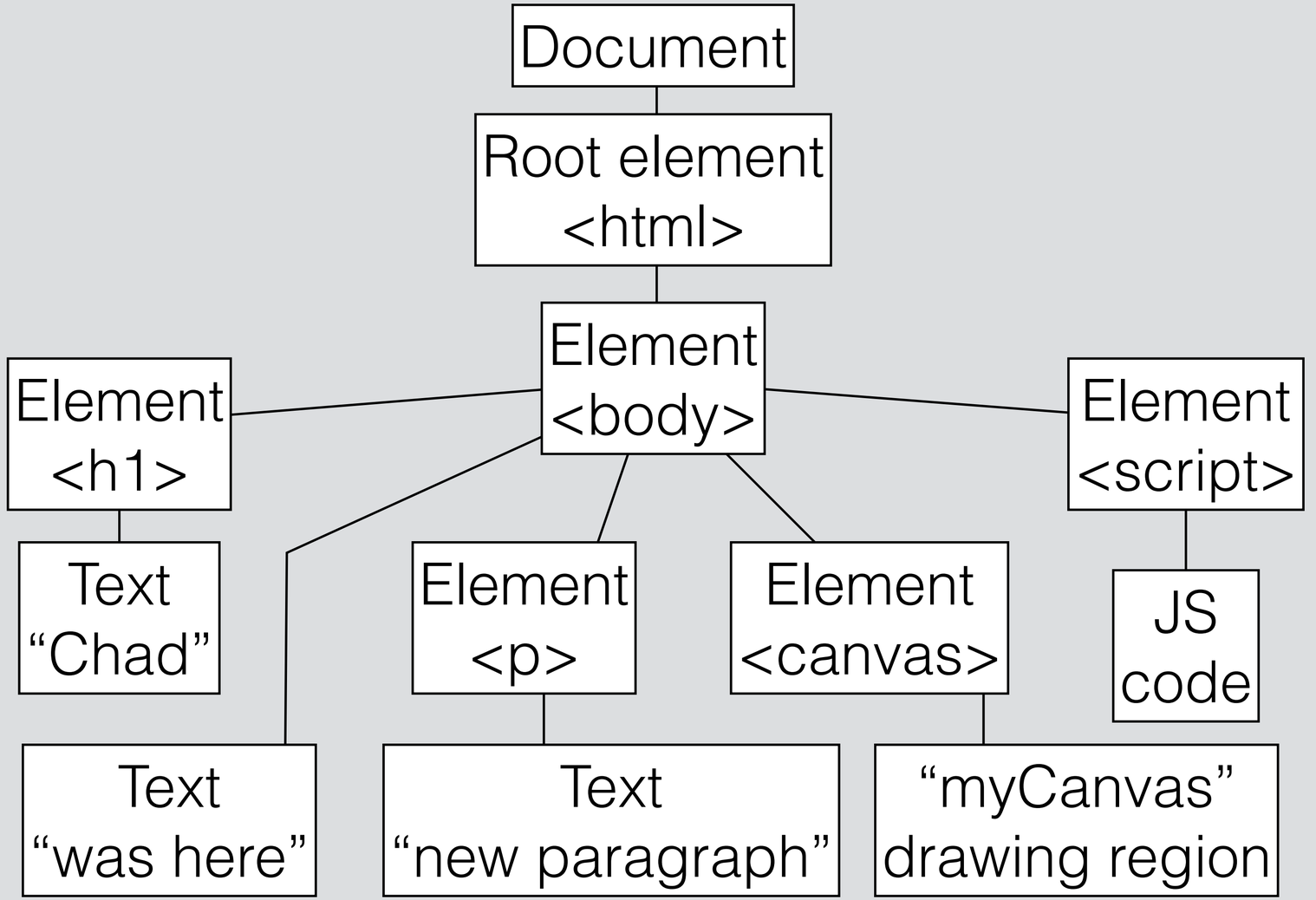
- http://autorob.github.io/examples/hello.html

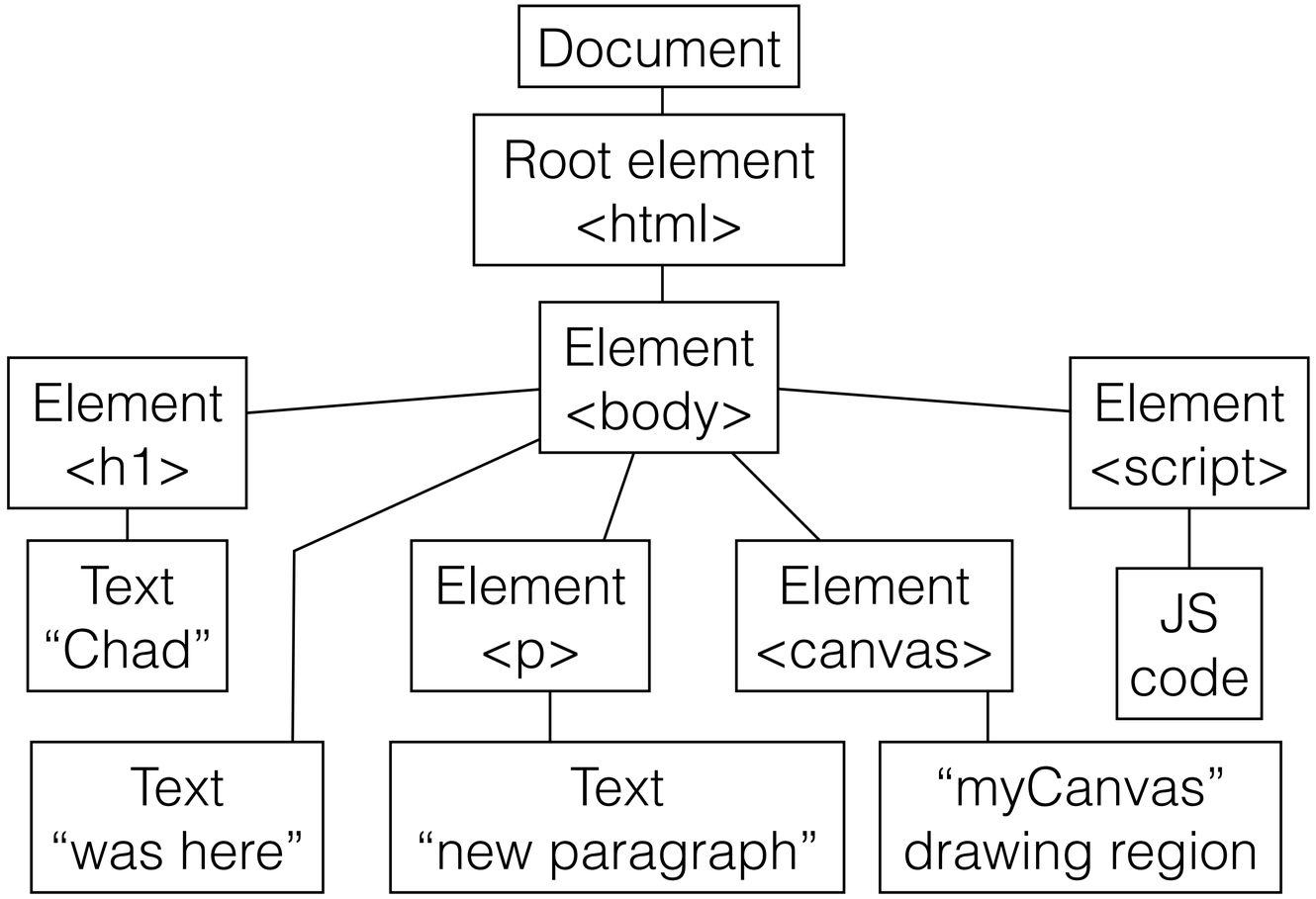
Loading…

hello example



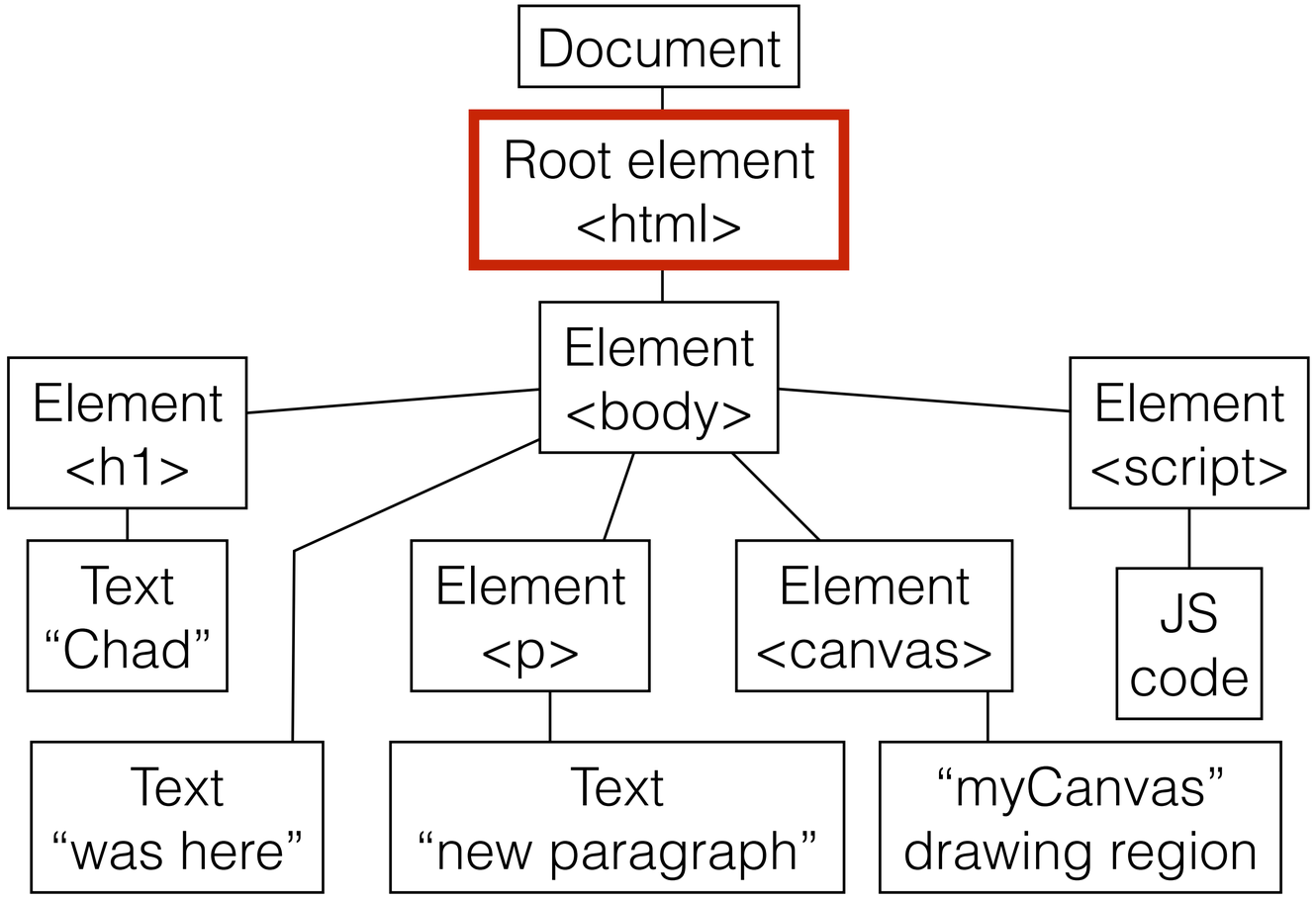
DOM created by browser upon loading hello.html





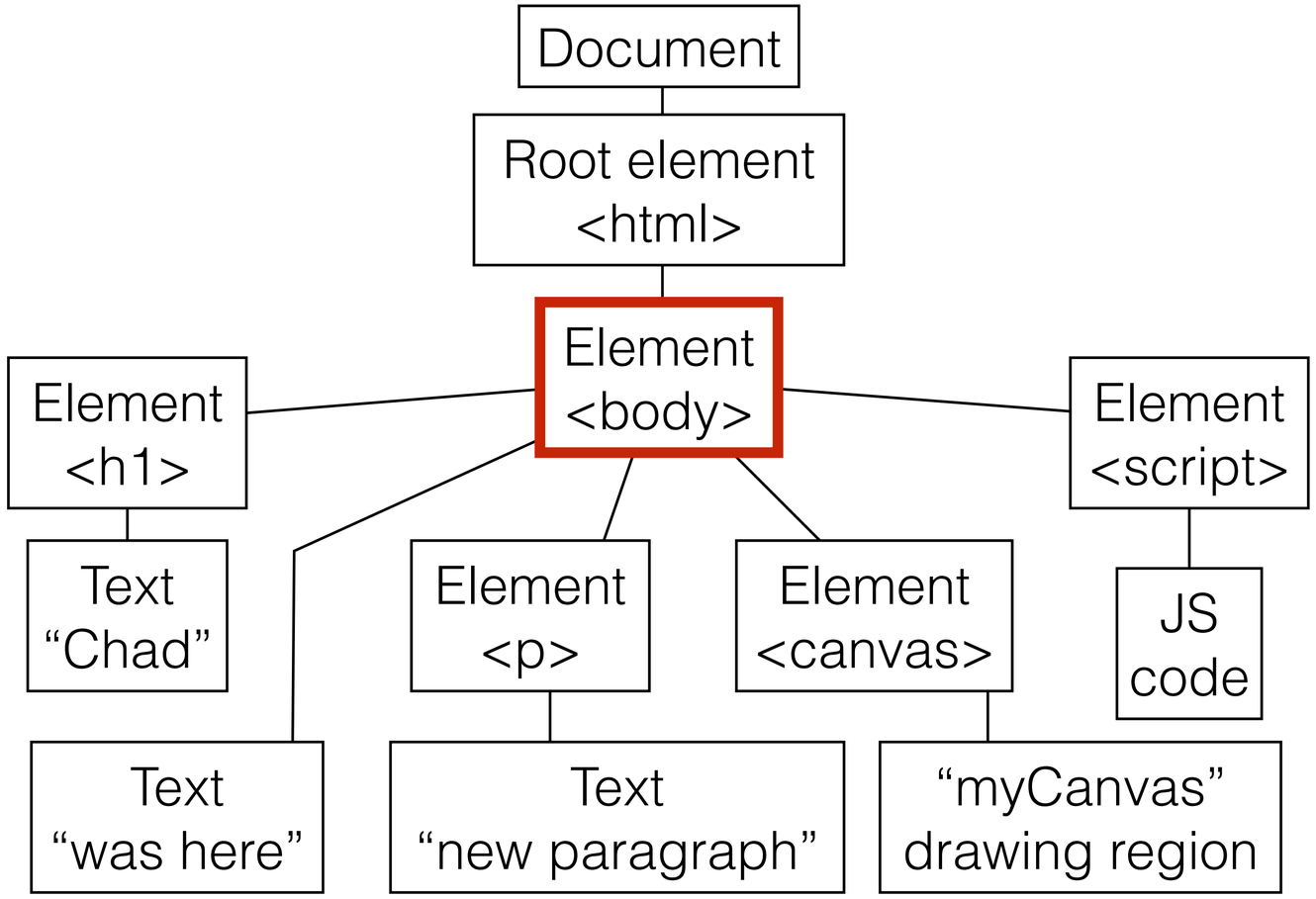

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

hello.html file



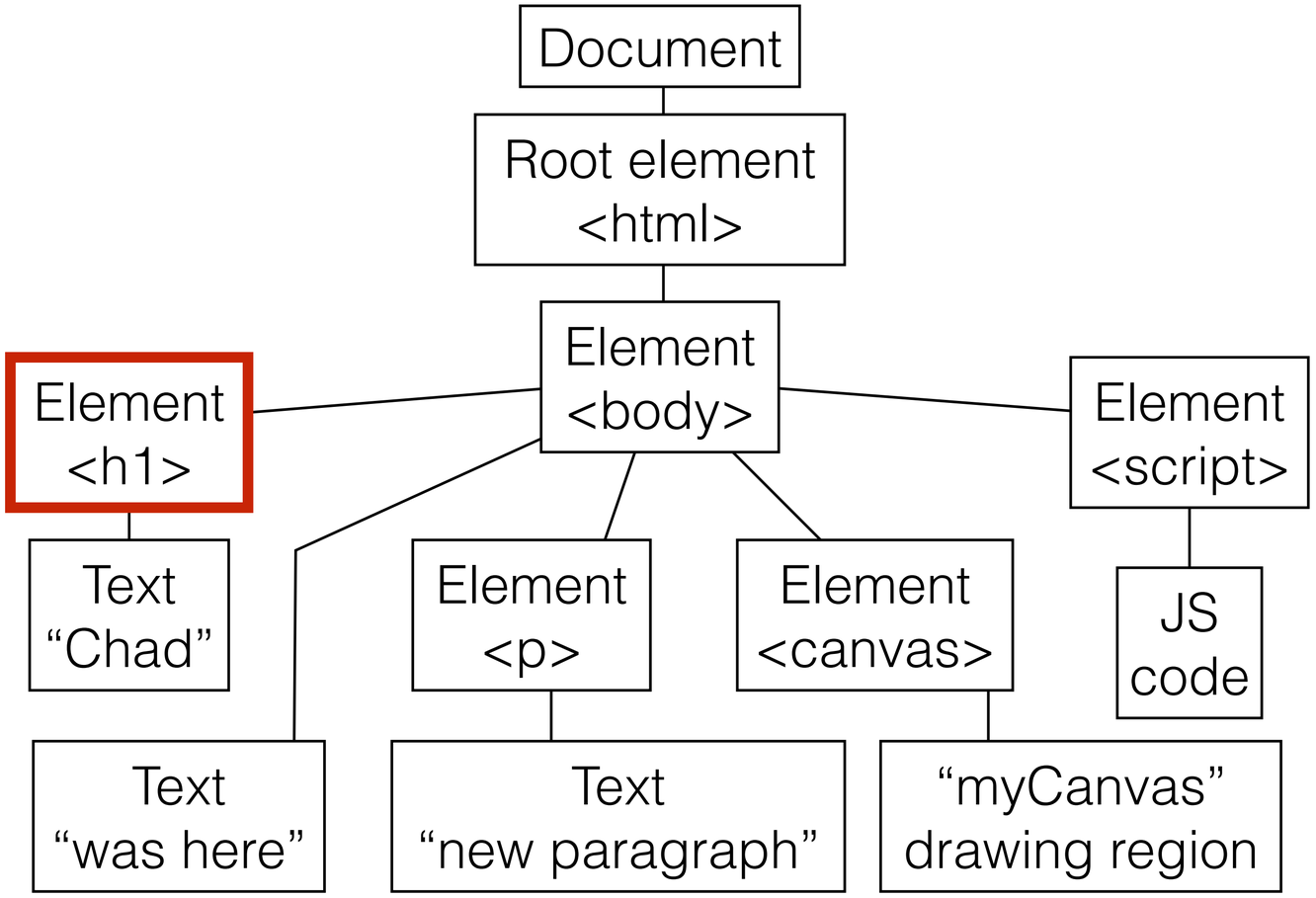

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

hello.html file



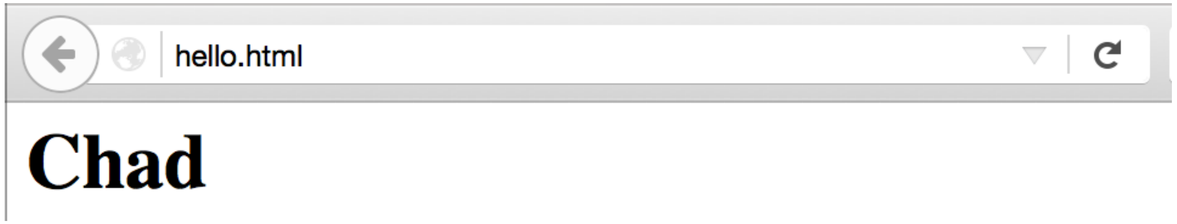

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

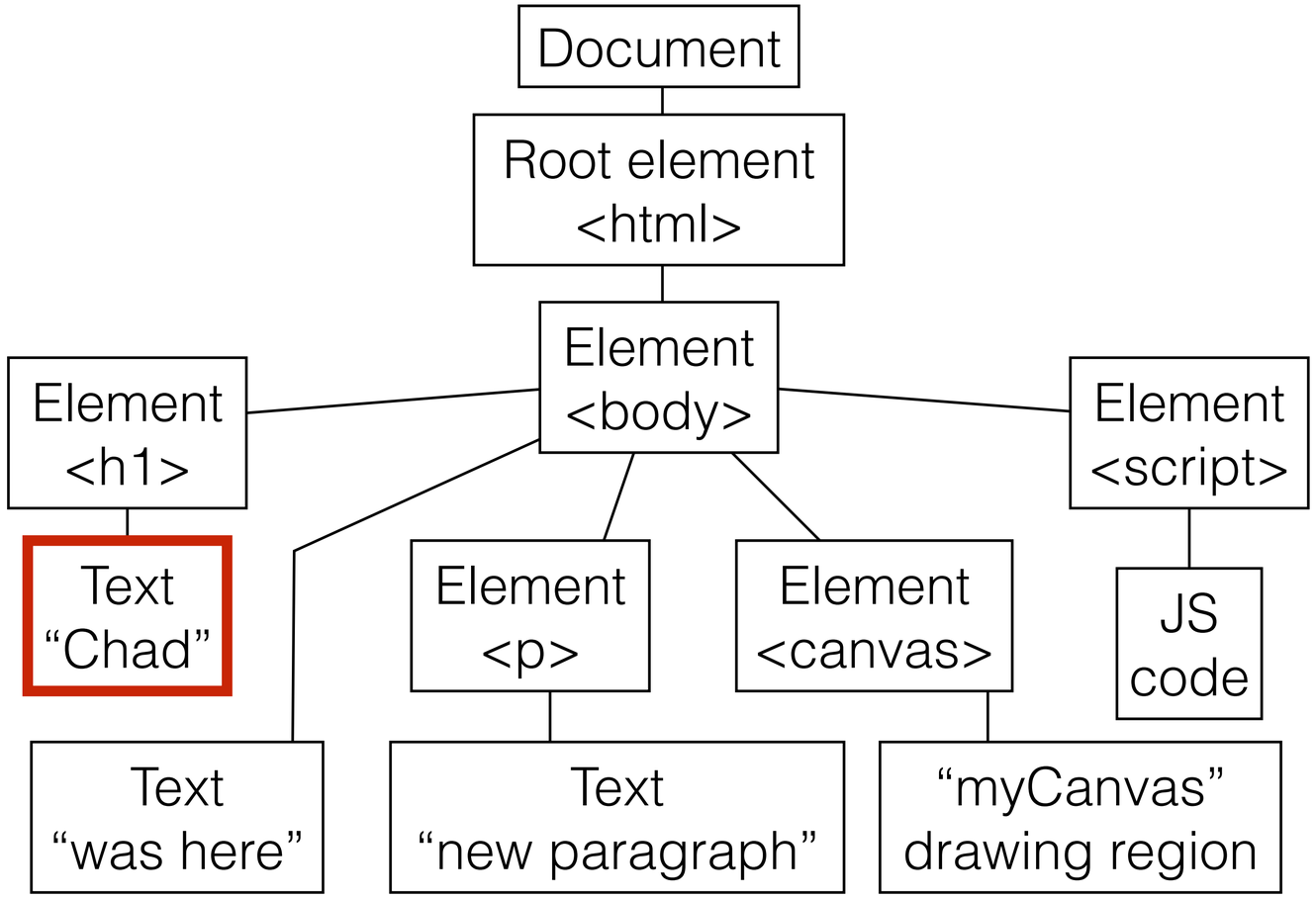
hello.html file




```
<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 --> new paragraph </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>
```

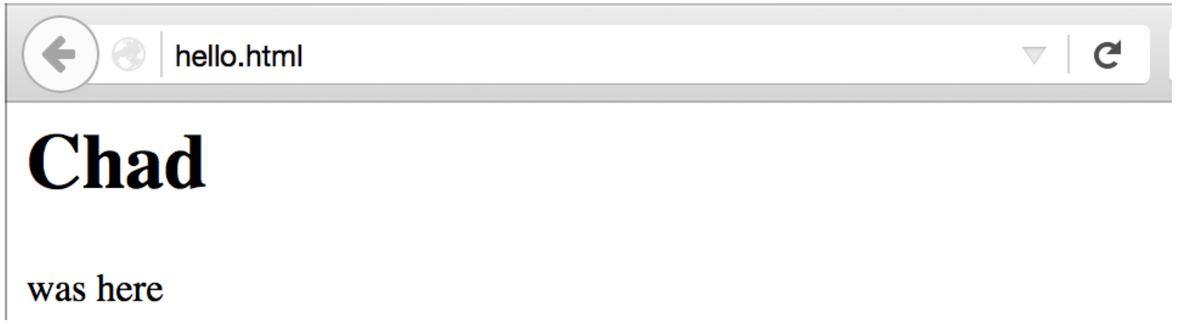
hello.html file

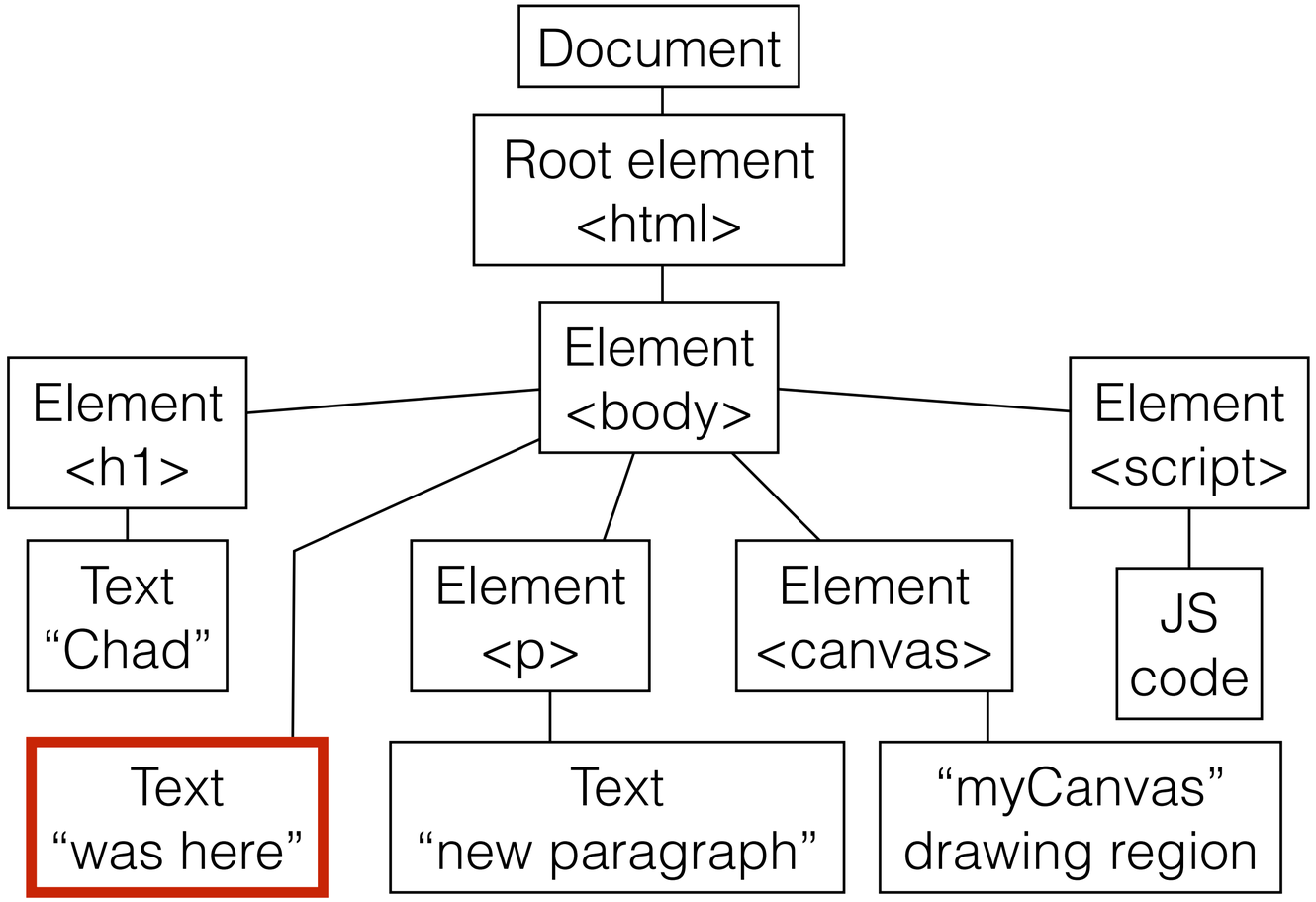





```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

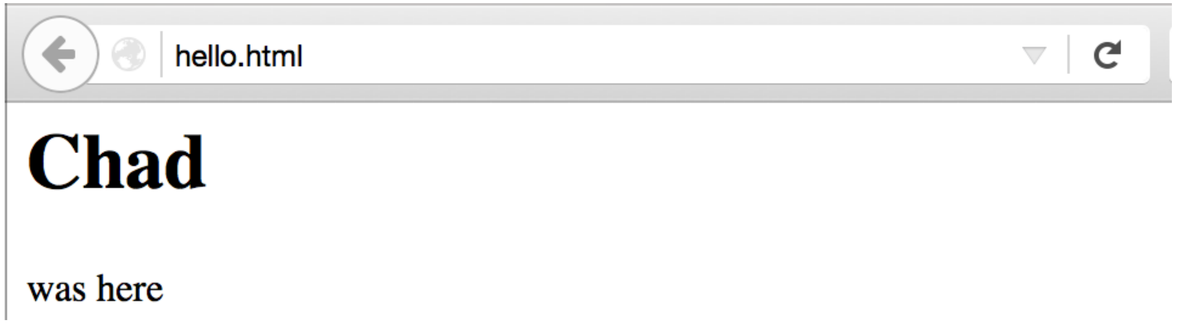
hello.html file

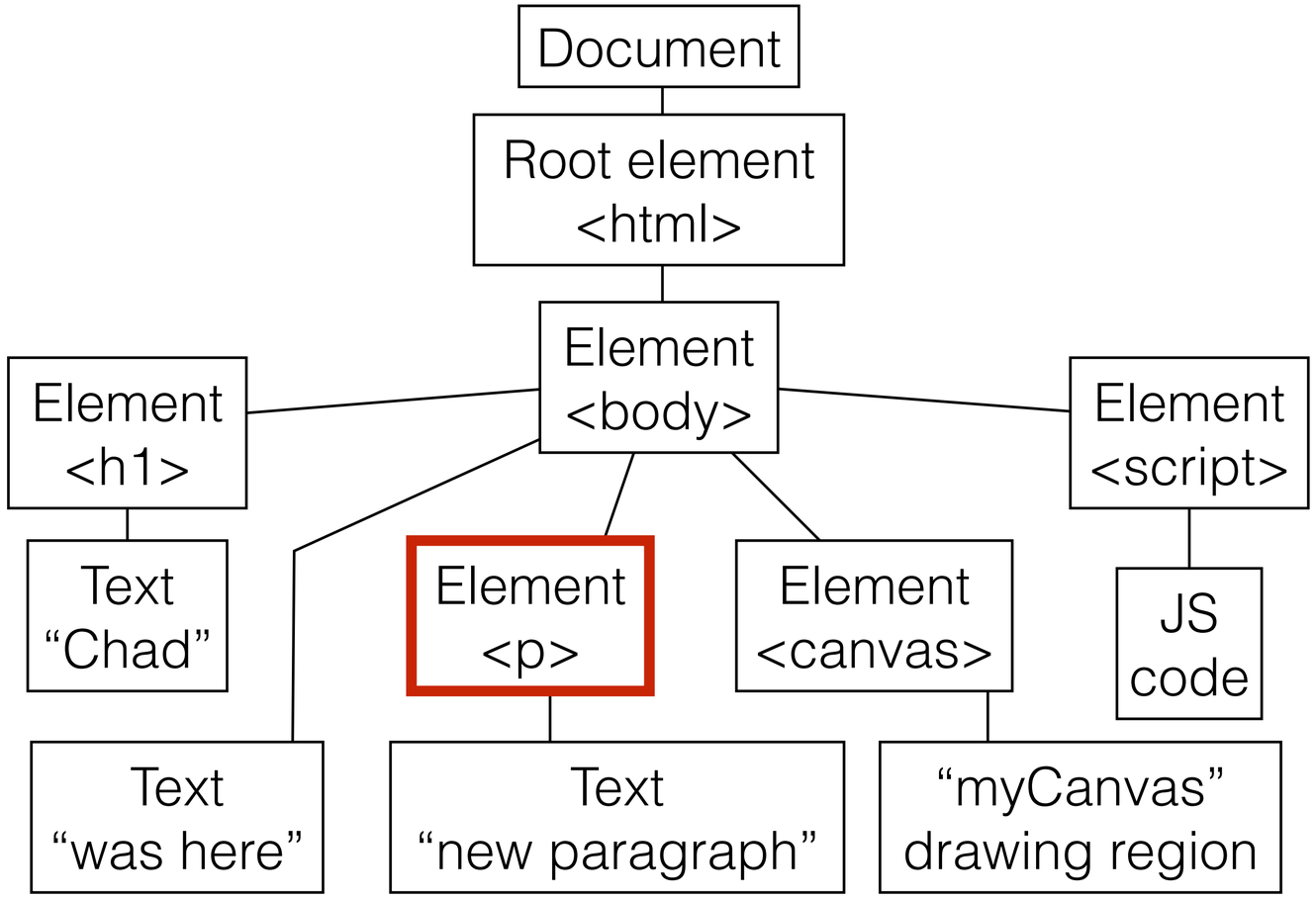





```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

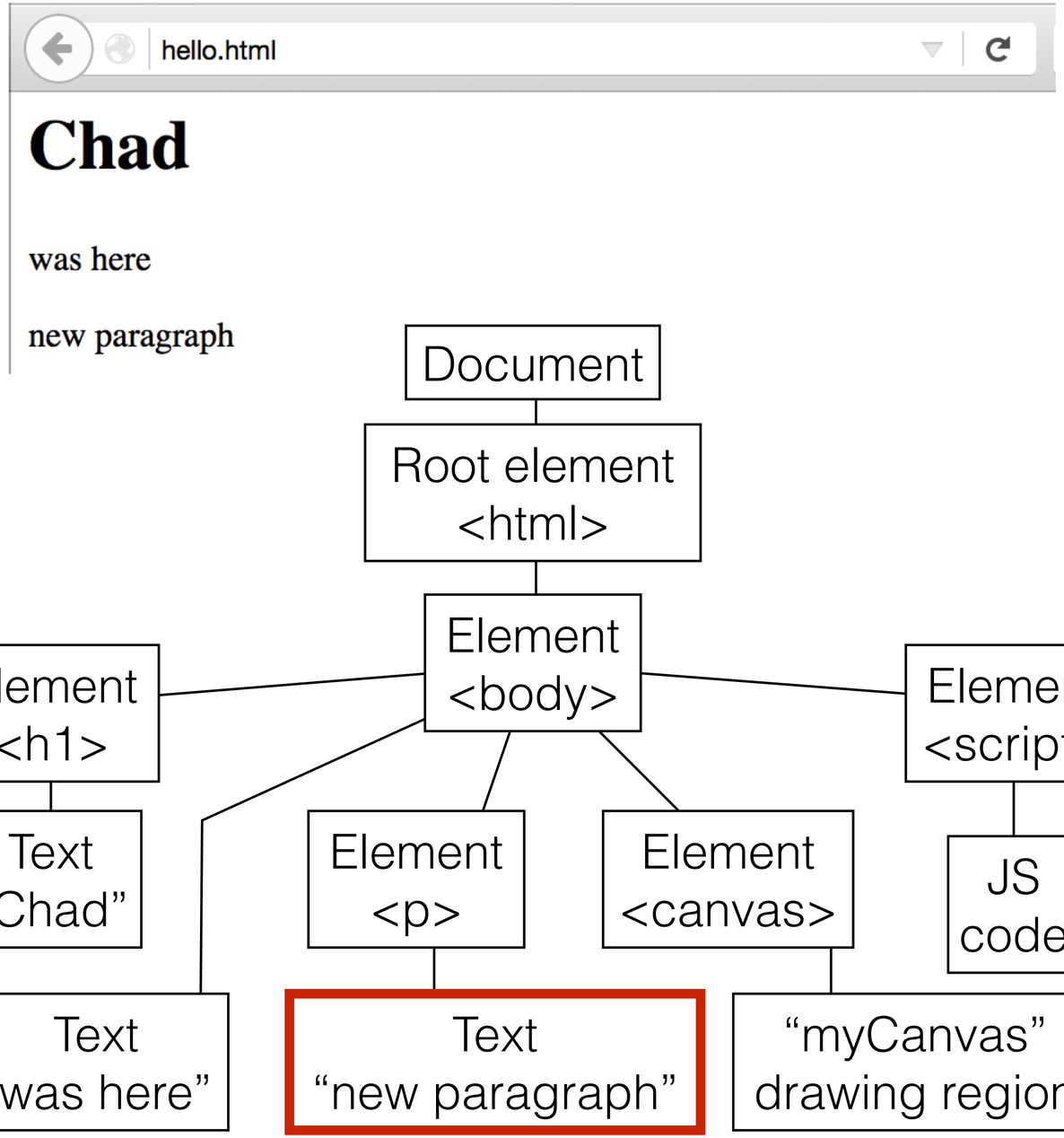
hello.html file






```
<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 --> new paragraph </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>
```

hello.html file



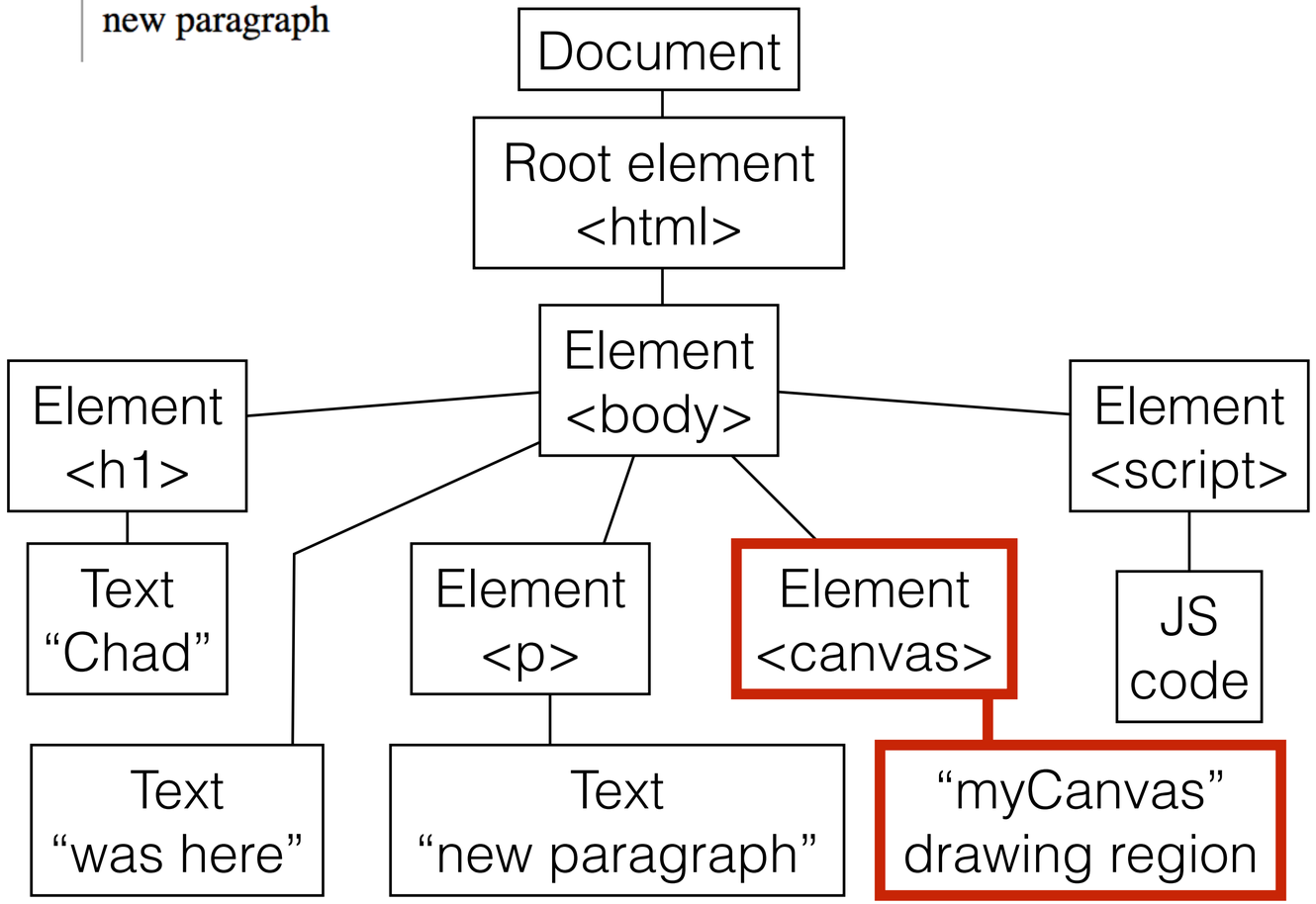

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

hello.html file

Chad

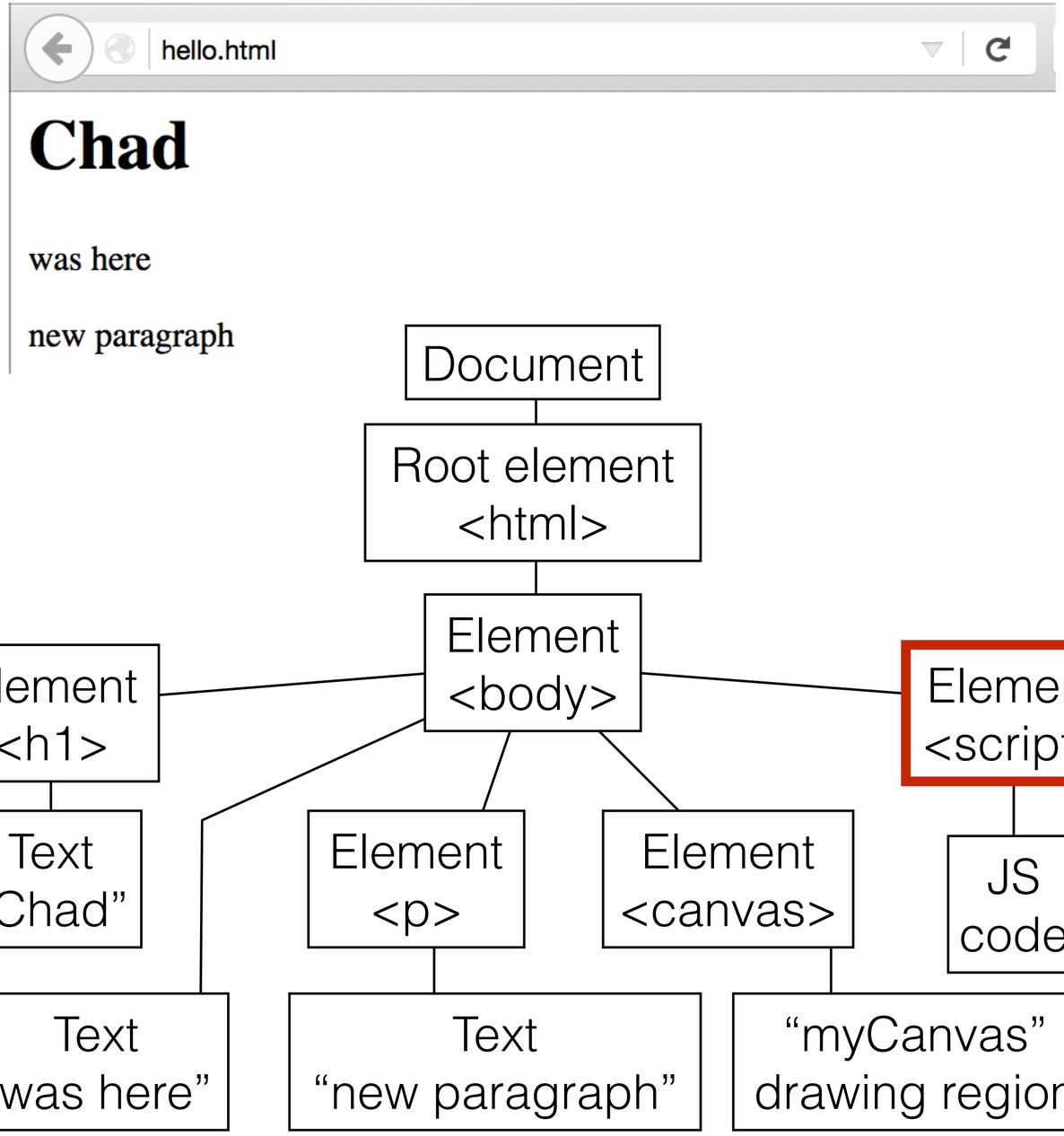
was here

new paragraph



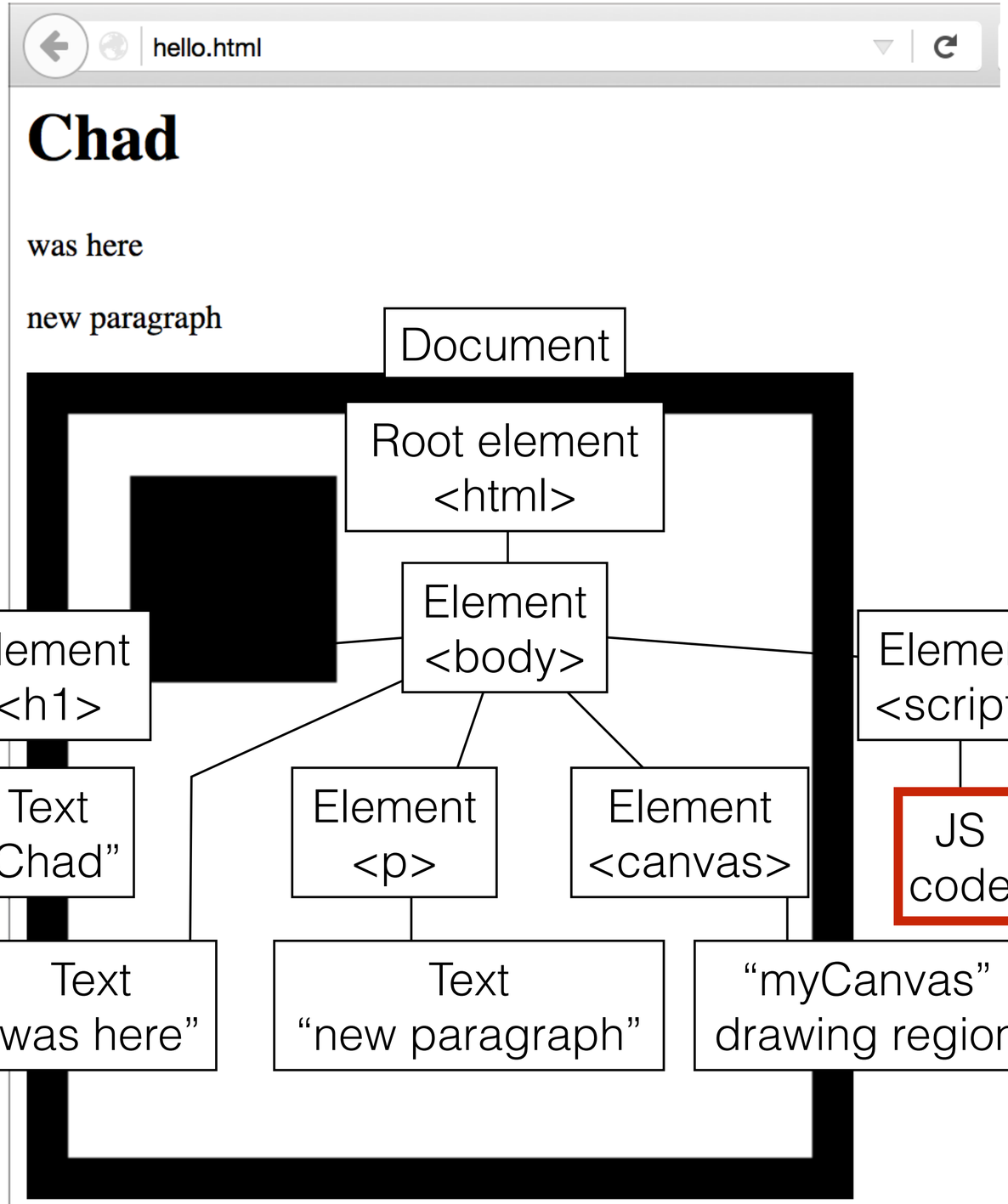

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

hello.html file




```
<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 --> new paragraph </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>
```

hello.html file



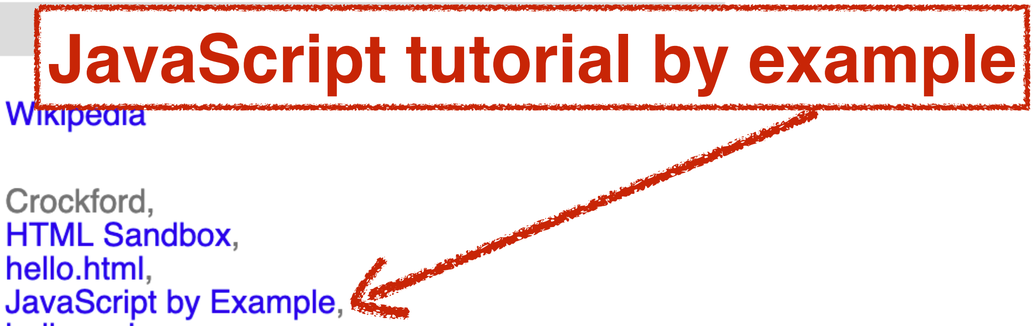

```
<html> <body> <!-- this is a comment in HTML.  it is ignored -->

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

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

<p>   <!-- start a new paragraph --> new paragraph </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>
```

hello.html file

JavaScript:
the quick and dirty

Preview slides from lectures during the Fall 2020 offering of AutoRob are provided. These preview slides will be replaced with recorded lectures for Winter 2022 as the videos become available.

Date	Topic	Reading	Project I Quiz
Jan 5	Initialization: "So, where is my robot?", "What is a Robot OS?", Course administration and logistics [Lecture Video] [Session recording (UM only)] What is a robot?: Brief history and definitions for robotics	Spong Ch.1 Corke Ch.1	Setup git repository Out: Path Planning
Jan 7	Lab Session: Git-ing started with git, JavaScript, and KinEval [Session recording (UM only)]		
	Week 2		
Jan 10	Path Planning: Navigation as graph search, DFS, BFS, Dijkstra shortest paths, A-star, Priority queues and binary heaps [Lecture Video]	Wikipedia	
Jan 12	JavaScript and AutoRob workflow: Project workflow with git, JS/HTML5 tutorial, Document Object Model, Version Control, LaTeX math mode, Licensing, Michigan Honor License	Crockford, HTML Sandbox, hello.html, JavaScript by Example, hello_anim, hello_anim_text	
Jan 14	367 Lab: KinEval Path Planning code overview		
	Week 3		
Jan 17	**No course meeting** - Martin Luther King, Jr. Day UM Martin Luther King Jr. Symposium Help broaden participation in computing and robotics		
Jan 19	Dynamical Simulation: Simple pendulum, Lagrangian equation(s) of motion, Initial value problem, Explicit integrators: Euler, Verlet, and	Spong Ch.7 I Corke Ch.9 Euler's Method	



Quick JavaScript Code-by-Example Tutorial

Chad Jenkins (ocj)

```
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
```

Open “View Source” in browser to see this code

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

    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>
```

```
      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 -->
<html>

    <!-- The onload property will execute upon the browser completely loading
         the body of the HTML document.  This onload routine will call the
         function "tutorialJSCoding" that is in the "js_overview.js" file.
    -->
    <body onload="tutorialJSCoding()">
        <p>
        Quick JavaScript Code-by-Example Tutorial <br>
        <i><a href="ocj.name">Chad Jenkins (ocj)</a></i>
        </p>

        <p>
        <br><br><br>
        (Remember to open the <a href="https://webmasters.stackexchange.com/questions/8525/how-do-i-open-the-javascript-console-i
        </p>
    </body>

    <head>
        <title>Quick JavaScript Code-by-Example Tutorial</title>

        <!-- the script tag contains JavaScript code that the browser will
             execute, either as code inside the tag markers or inside the
             file specified in the src property
        -->
        <script type="text/javascript" src="js_overview.js"> </script>

        <!-- this tag removes the annoying error message about garbled text -->
        <meta charset="UTF-8">
    </head>
</html>
```

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”

Equal sign (=) will assign a value to a variable

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

Double forward slash (//) will ignore the remainder of a line

Variable Assignment Examples

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

assigns string “this is a global variable” in a variable named stringGlobal

variable name	variable value
stringGlobal	“this is a global variable”

```
var booleanLocal = true;
```

assigns boolean value of true in a variable named booleanLocal

variable name	variable value
booleanLocal	true

```
numberLocal = 20 - 18;
```

assigns numeric value of 20 minus 18 in a variable named numberLocal

variable name	variable value
numberLocal	2

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 variables as “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

JavaScript Object Notation (JSON)

- Objects in JavaScript can be alternatively assigned through JSON https://en.wikipedia.org/wiki/JSON

The same object created through JavaScript statements and separately in JSON

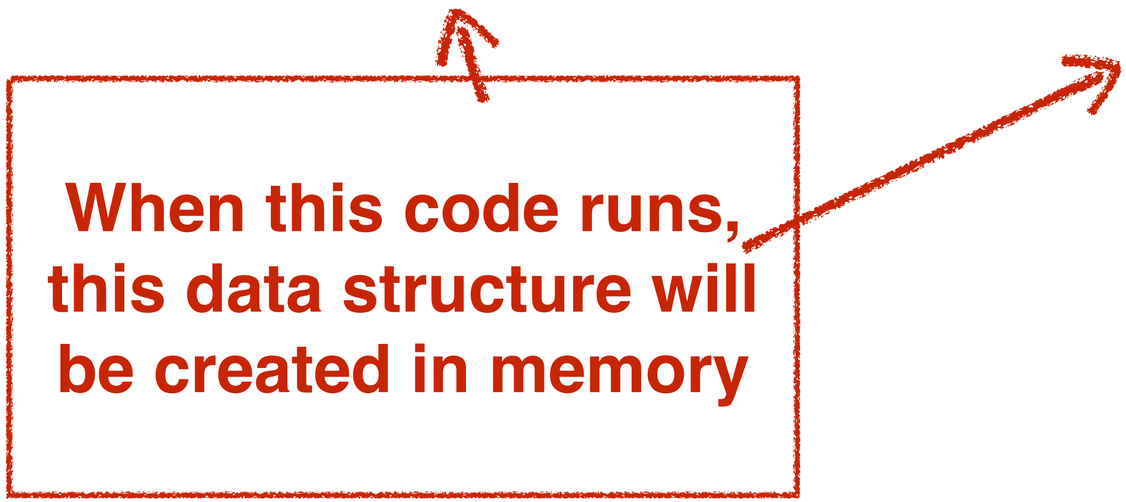
```
myObject = {};  // objects can also be created dynamically

// 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"
```

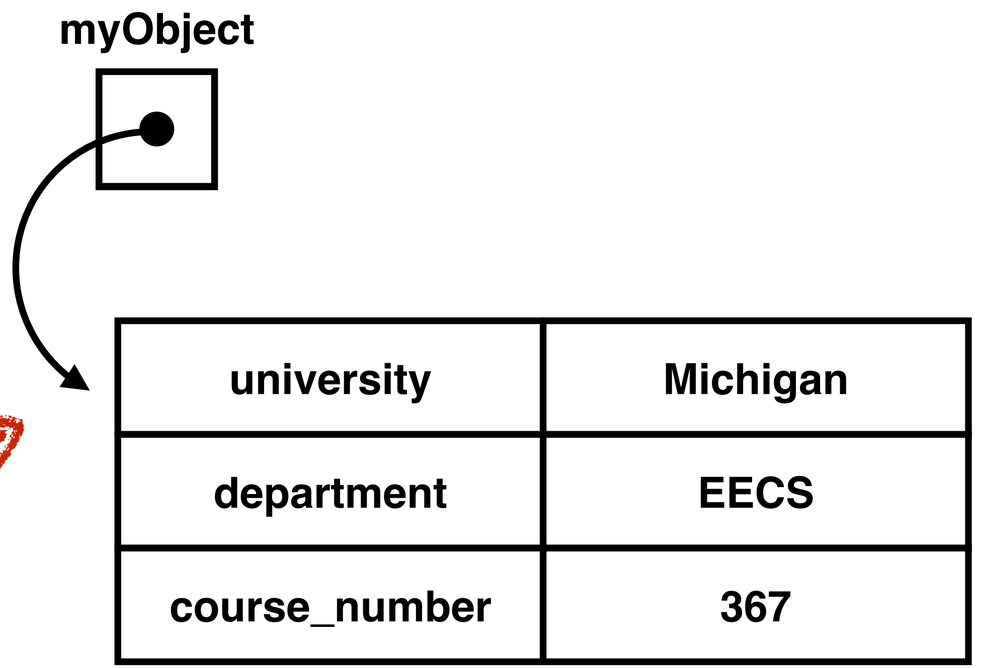
```
myObject = {
    university : "Michigan",
    department : "EECS",
    course_number : 367
}
```

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"
```





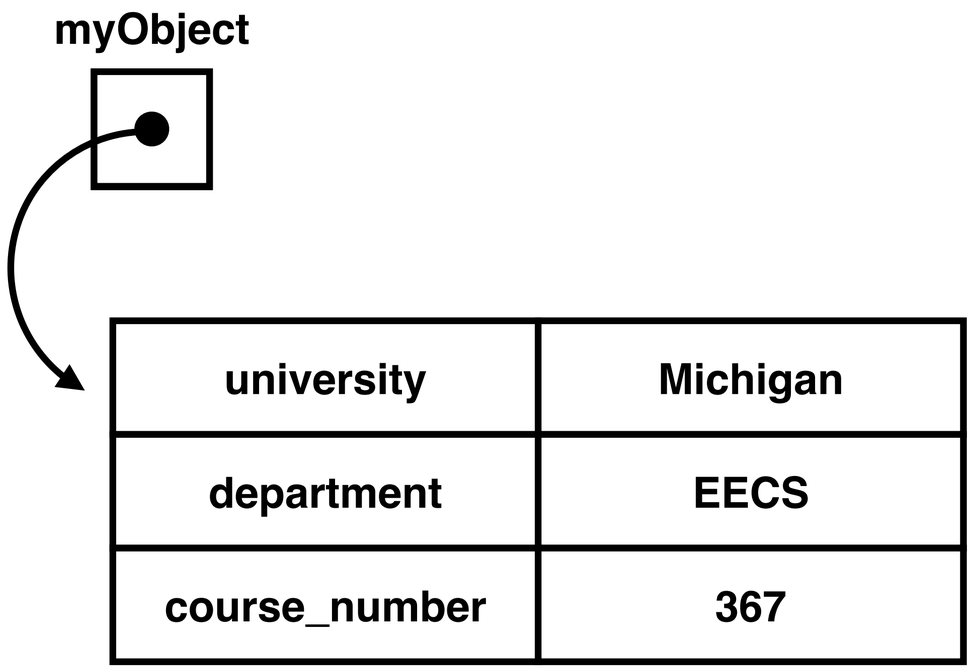
Objects and References

- 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;
```

what will happen if we assign this structure to a new variable?



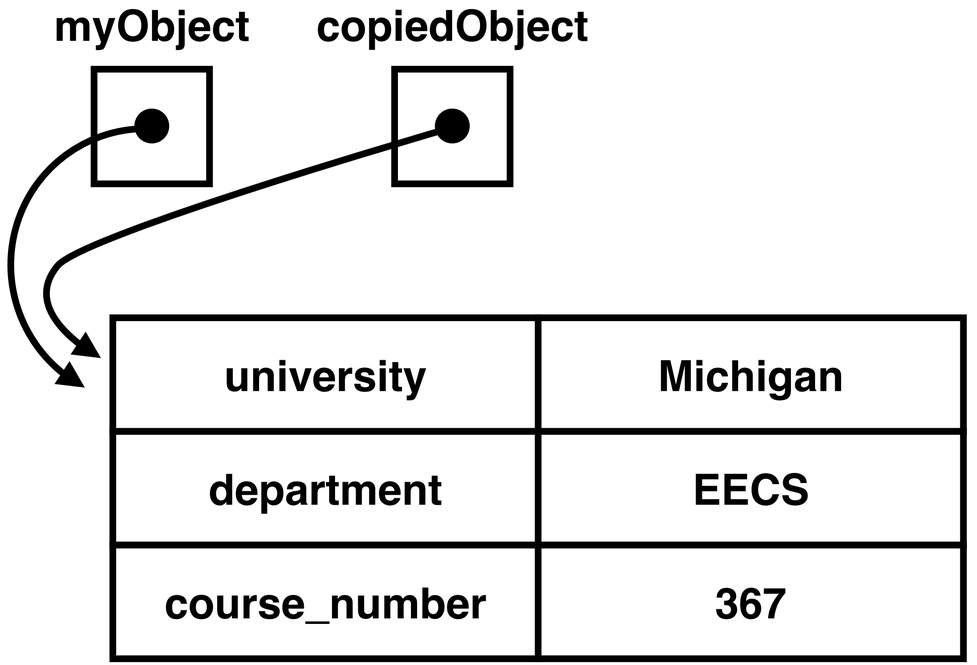
Objects and References

- 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;
```

what will happen if we assign this structure to a new variable?



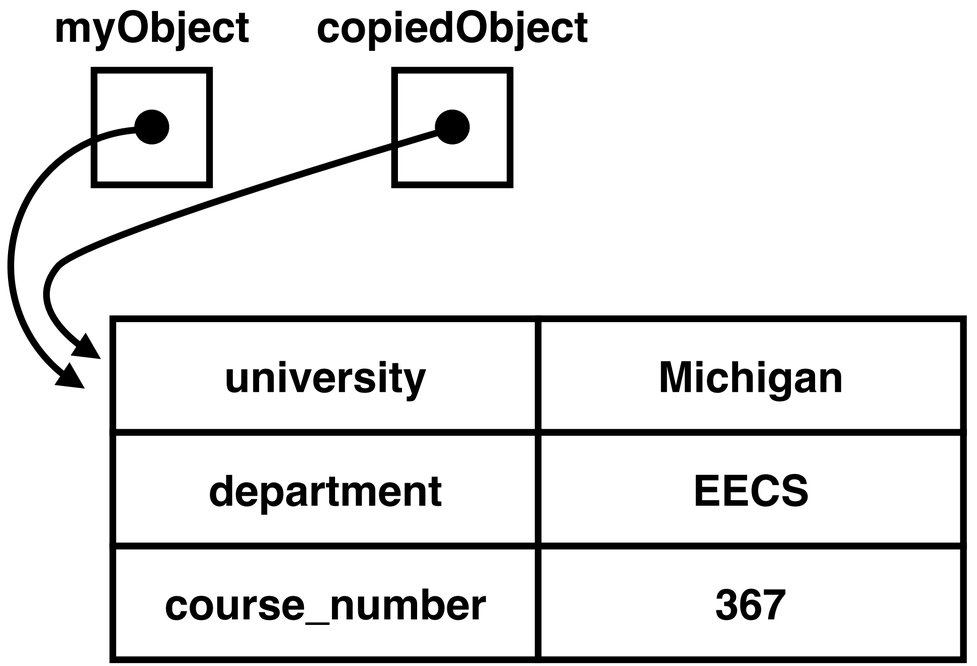
Objects and References

- 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;
```

what will happen if we modify the new variable?



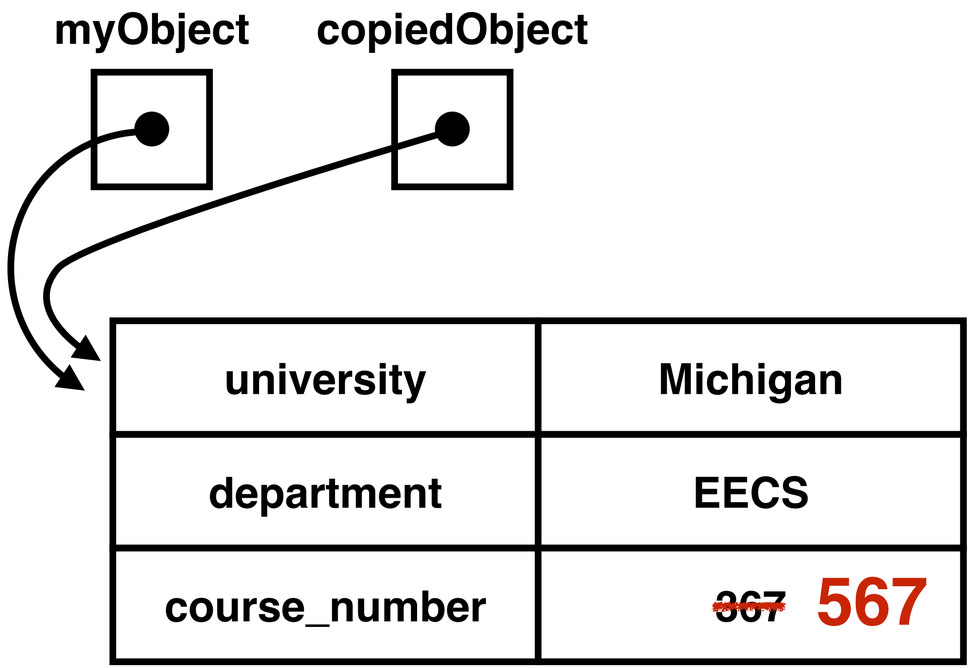
Objects and References

- 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;
```

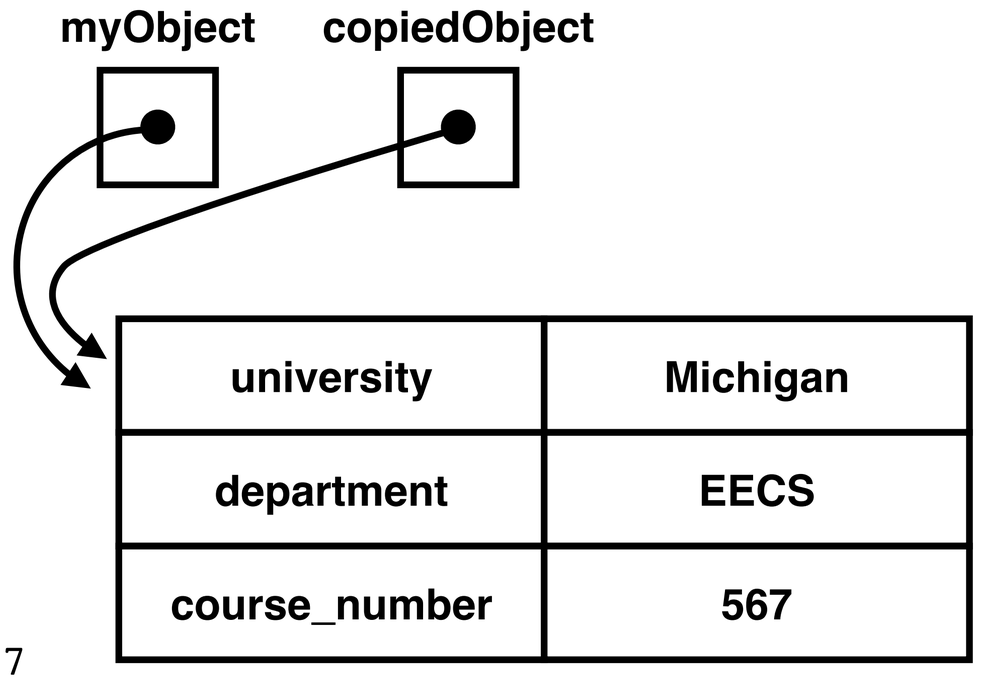
what will happen if we modify the new variable?



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
```

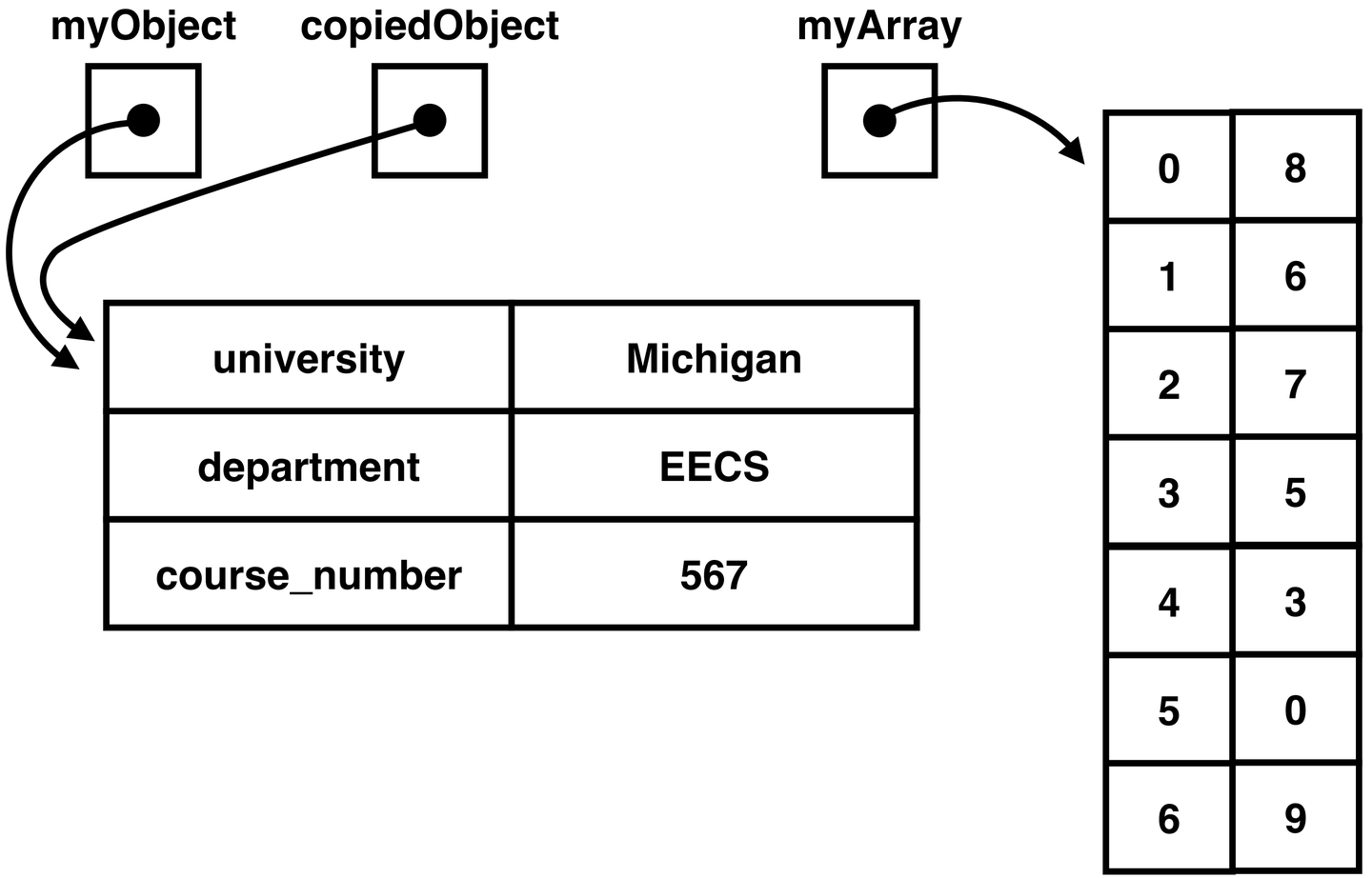


This result can be verified in the browser console

Arrays

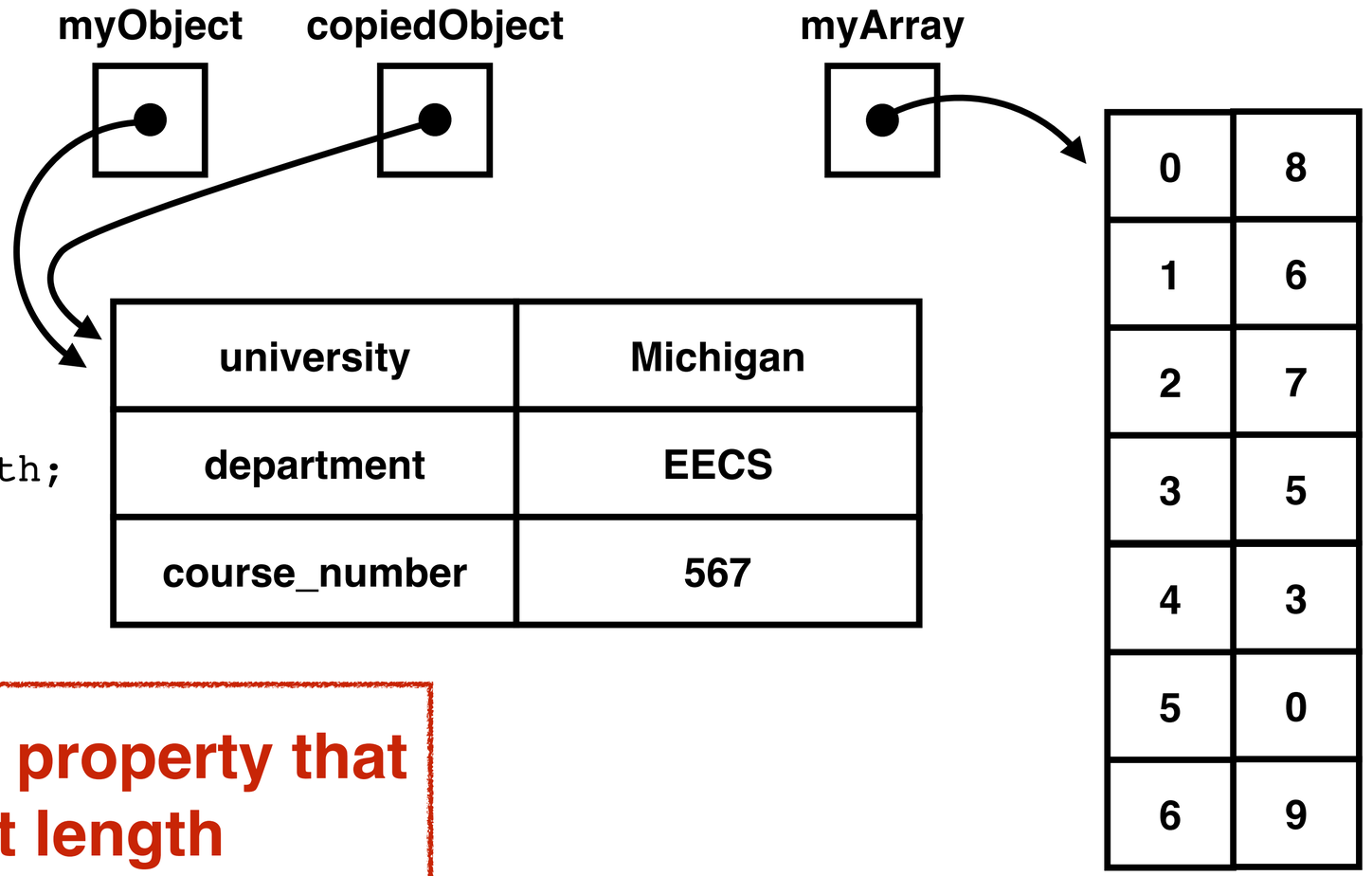
- An array is an instance of an object data type with numeric keys

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



Arrays

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



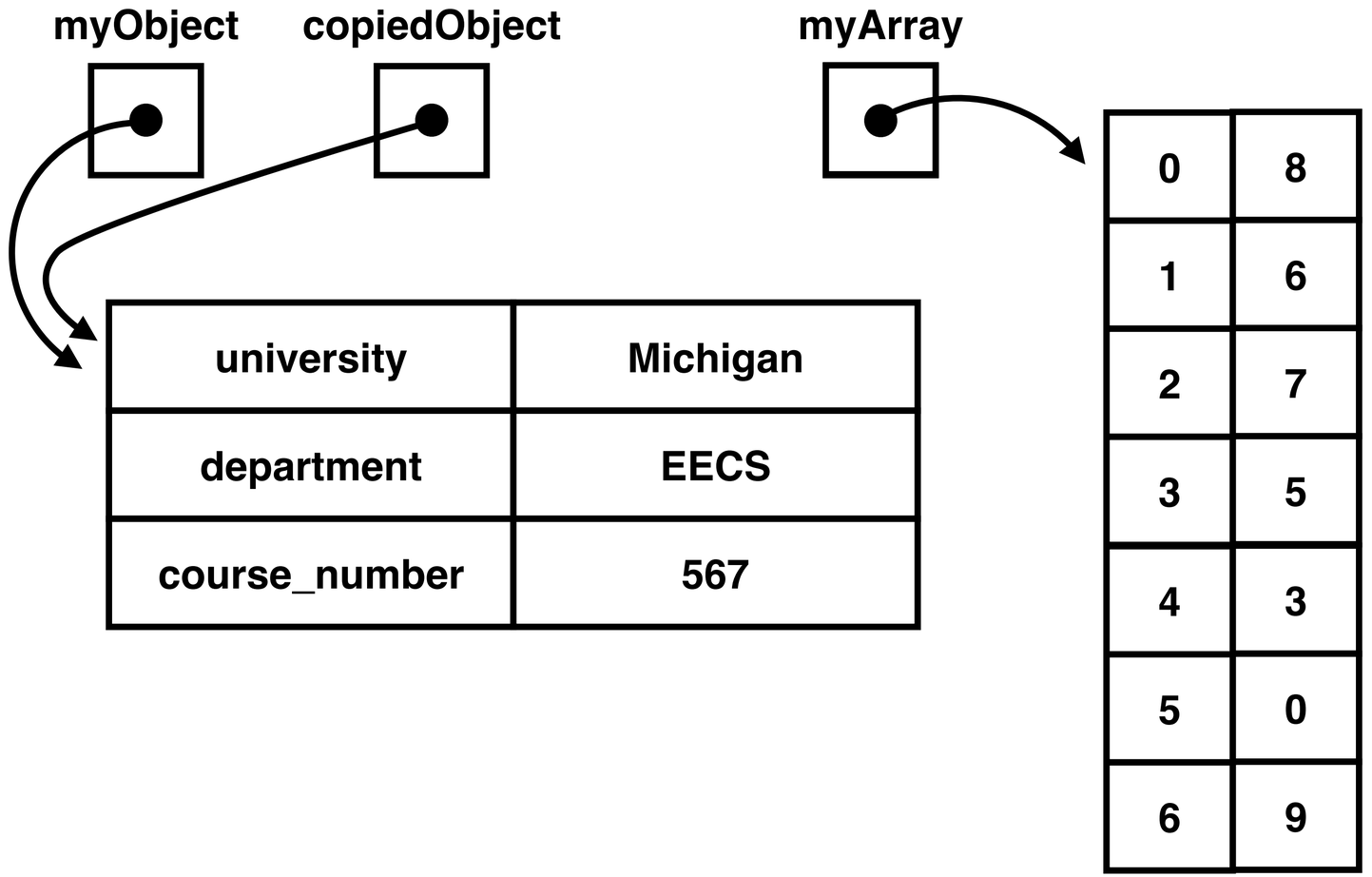

```
myArray = [8,6,7,5,3,0,9];

// this statement will output 7
numberOfArrayElements = myArray.length;
```

All arrays have a `.length` property that maintains its current length

Nesting Objects in Objects

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

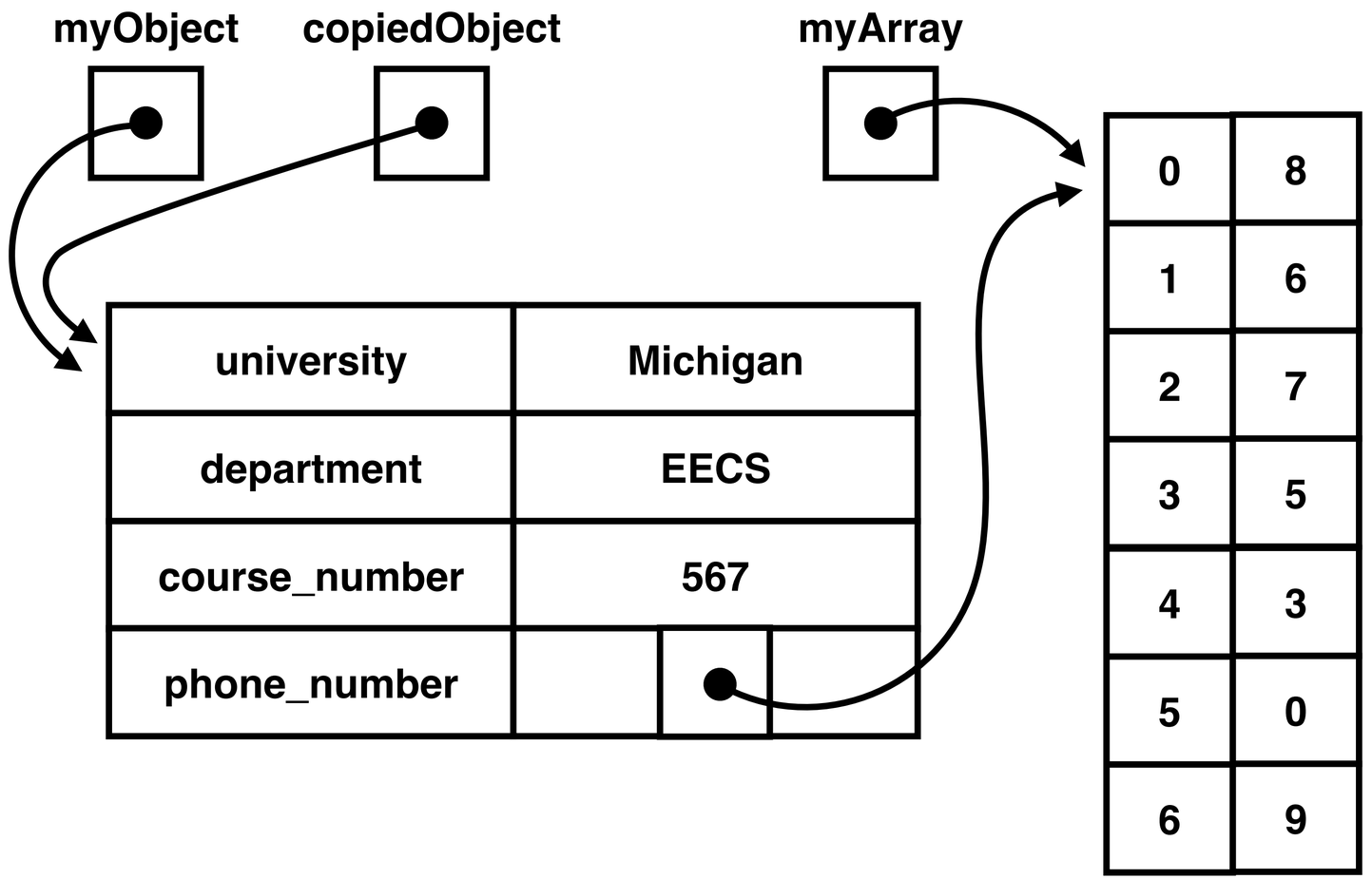
```
myArray = [8,6,7,5,3,0,9];

myObject.phone_number = myArray;
```



Nesting Objects in Objects

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

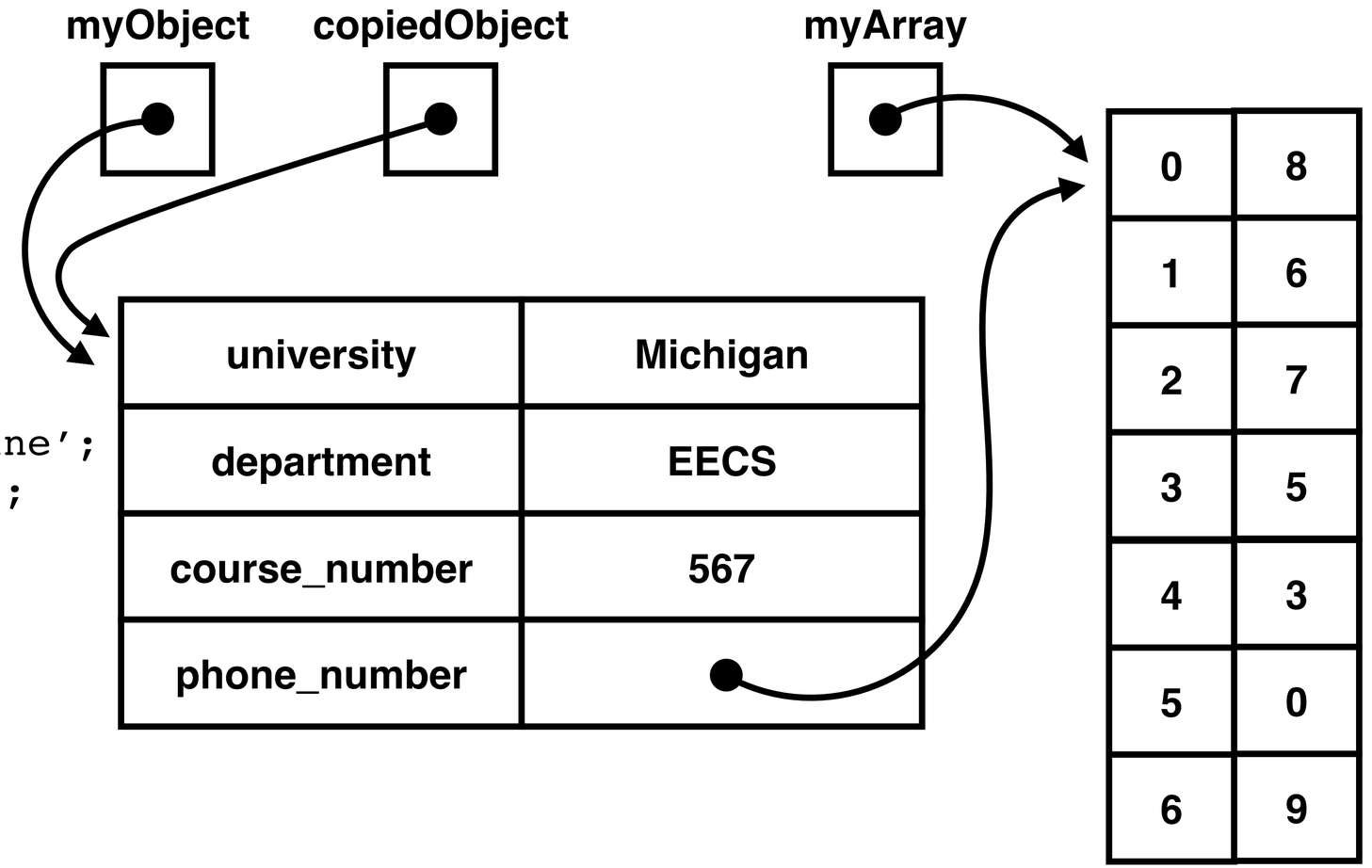
```
myArray = [8,6,7,5,3,0,9];

myObject.phone_number = myArray;
```



Dynamic Typing

- The type of a variable changes upon assignment

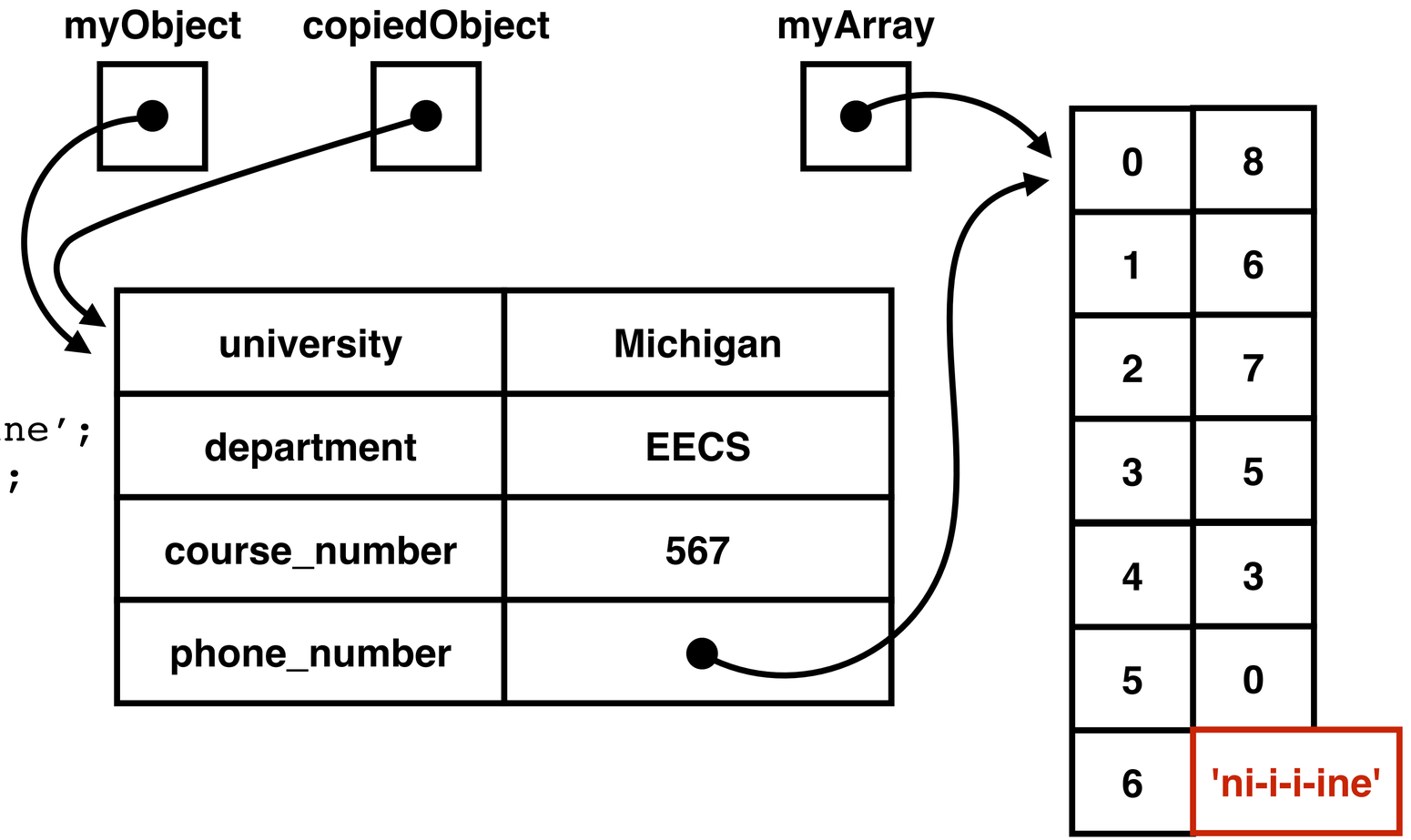
```
myArray = [8,6,7,5,3,0,9];

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

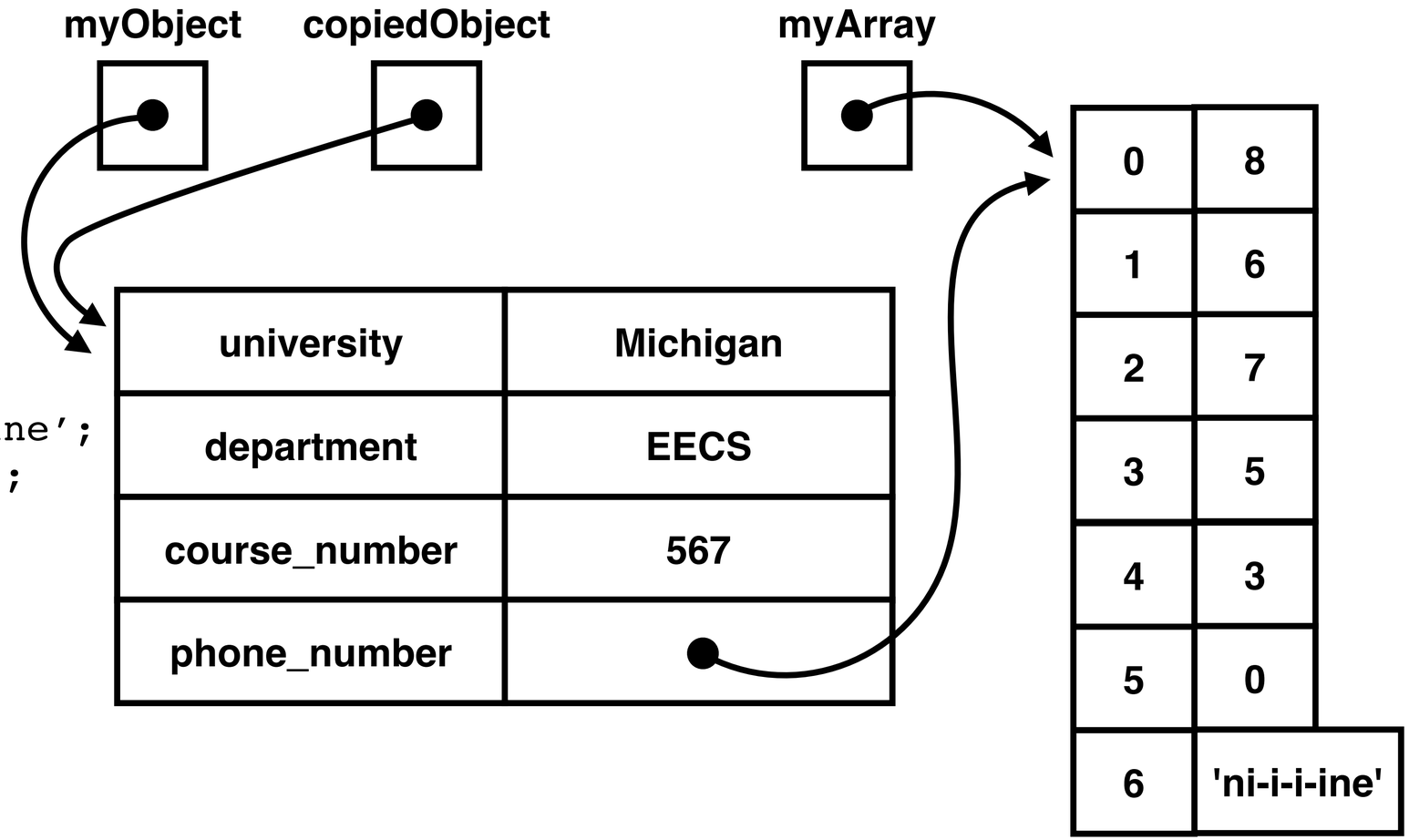
```
myArray = [8,6,7,5,3,0,9];

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

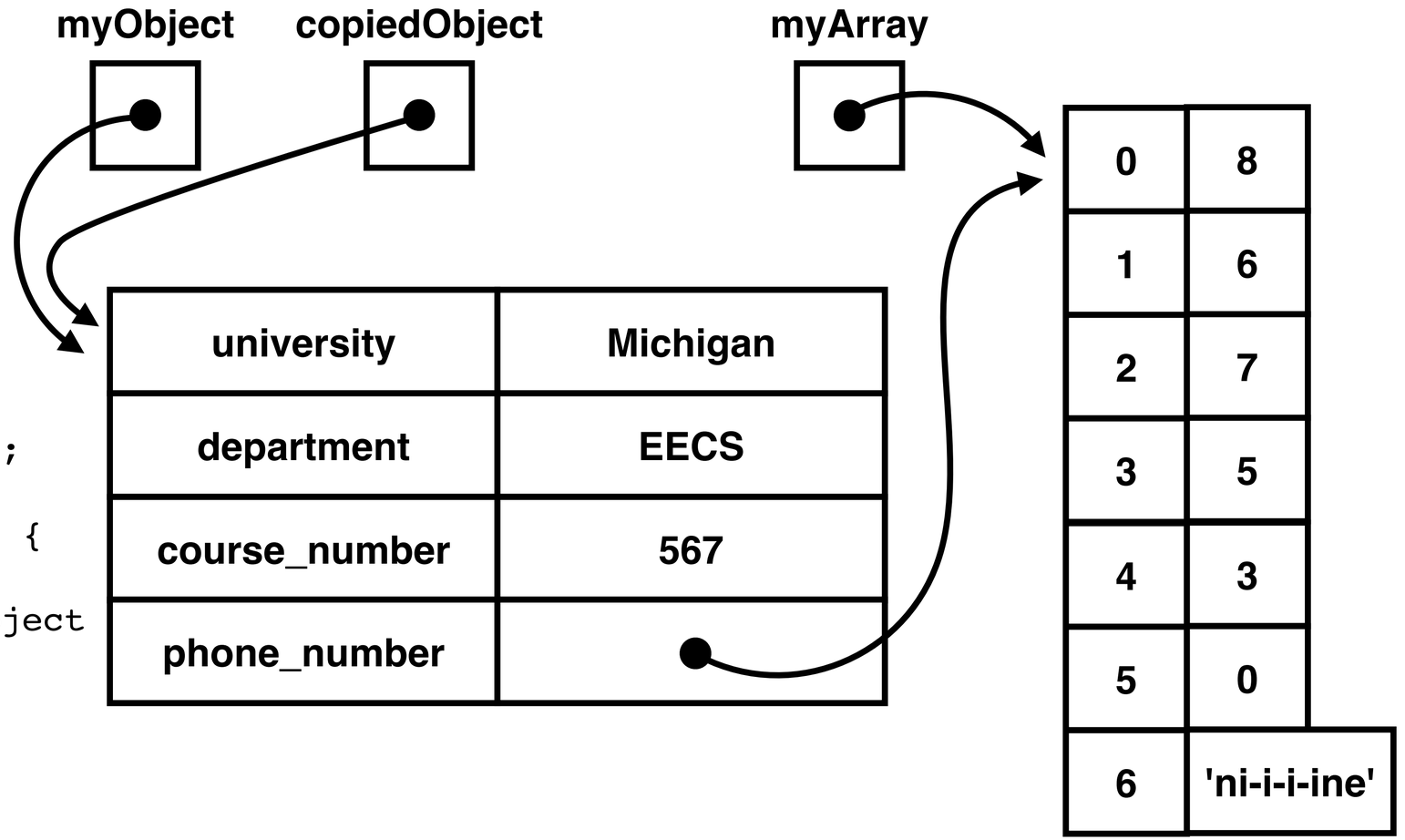
```
myArray = [8,6,7,5,3,0,9];

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]);
    // console output: 867530ni-i-ine
}

// output the course section to console
if (myObject.course_number === 367) {
    console.log('undergraduate section');
}
else if (myObject.course_number === 567) {
    console.log('old graduate section');
    // console will output this for myObject
}
else {
    console.log('ROB 511 is awesome');
}
```



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

Strict equality (===) will not attempt to match types

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

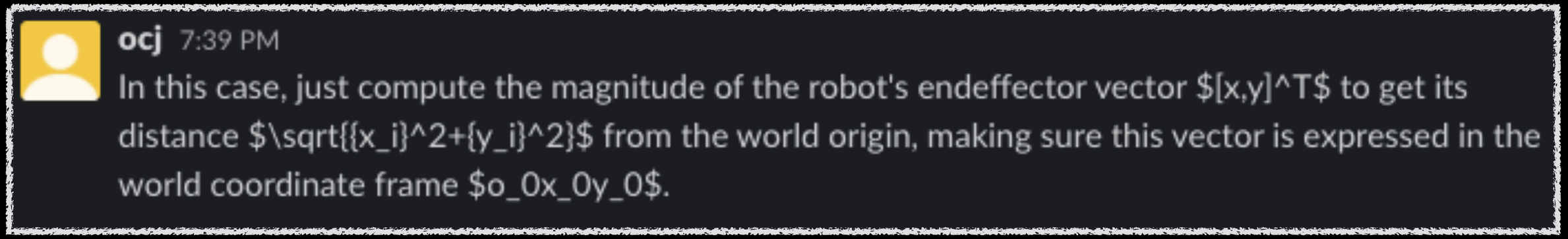
Brief Tangent

LaTeX Math Mode

Our default convention for writing mathematical expressions in text.

For example…

For example…


ocj 7:39 PM
In this case, just compute the magnitude of the robot's endeffector vector $[x,y]^T$ to get its distance $\sqrt{{x_i}^2+{y_i}^2}$ from the world origin, making sure this vector is expressed in the world coordinate frame $o_0x_0y_0$.

For example…

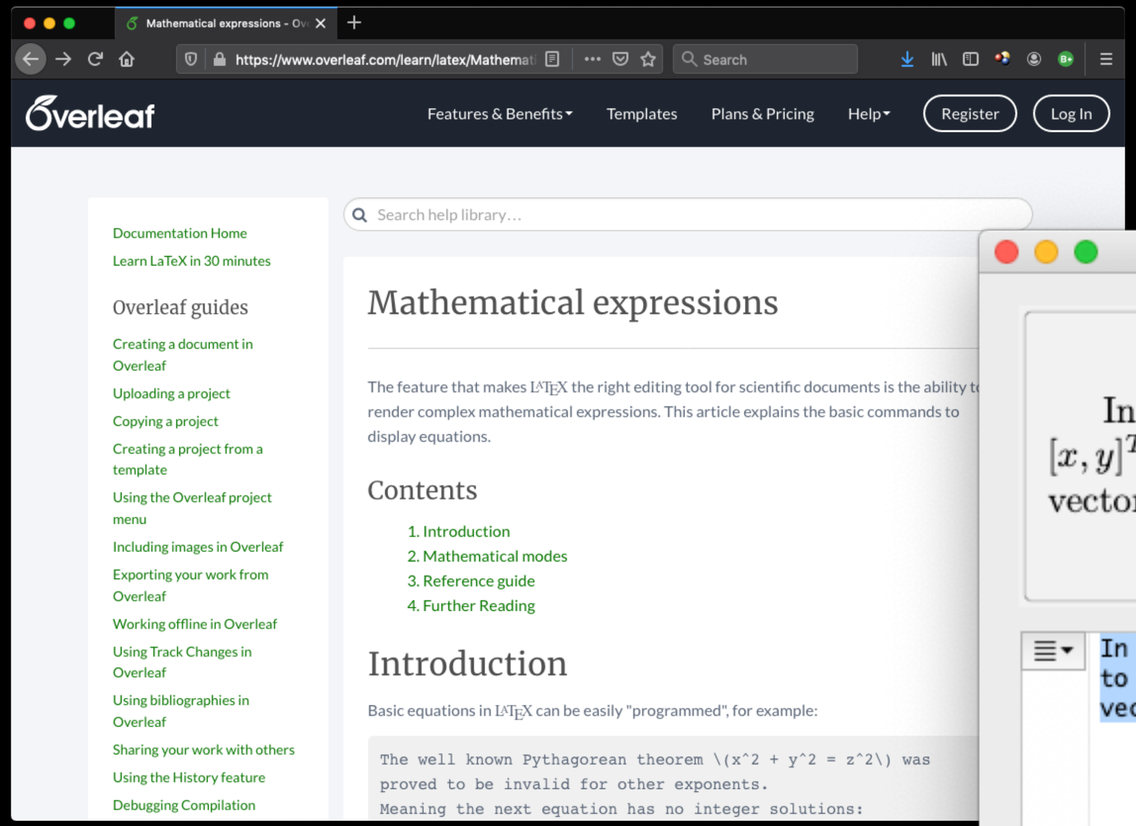
ocj 7:39 PM

In this case, just compute the magnitude of the robot's endeffector vector $[x,y]^T$ to get its distance $\sqrt{{x_i}^2+{y_i}^2}$ from the world origin, making sure this vector is expressed in the world coordinate frame $o_0x_0y_0$.

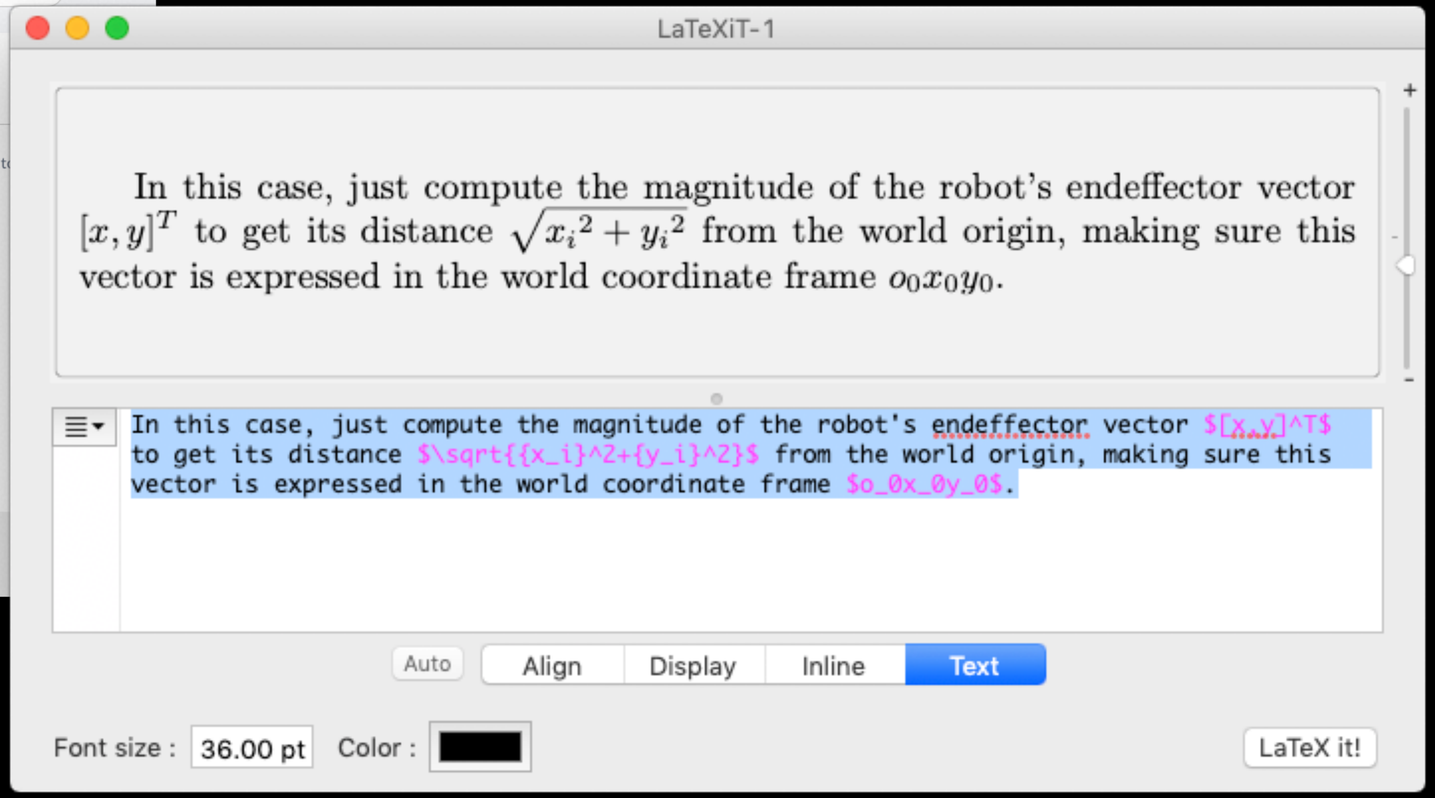
is meant to be read as…

In this case, just compute the magnitude of the robot's endeffector vector $[x, y]^T$ to get its distance $\sqrt{{x_i}^2 + {y_i}^2}$ from the world origin, making sure this vector is expressed in the world coordinate frame $o_0x_0y_0$.

Overleaf LaTeX Math Mode Guide



LaTeXiT Equation Editor



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*

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Function call stack

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

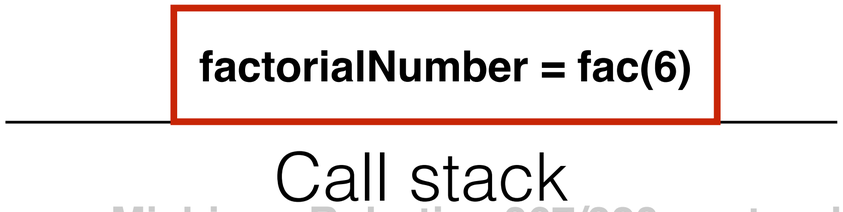
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Function call fac(6) stack pushes new local variable scope

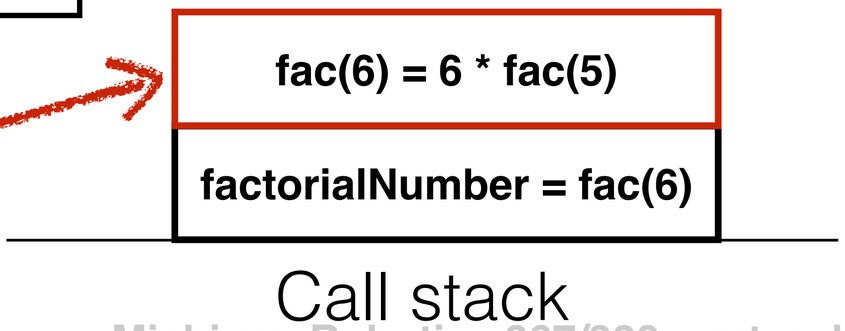


Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Function call fac(6) stack pushes new local variable scope

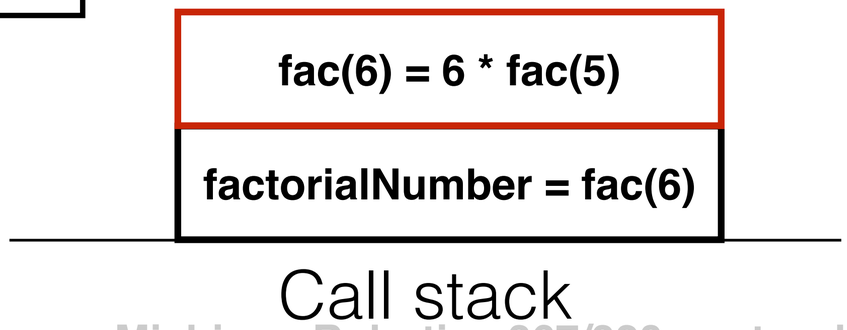


Function call stack

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

```
// define a function to compute the factorial of a numbe
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Recursive call fac(5) pushes new local variable scope

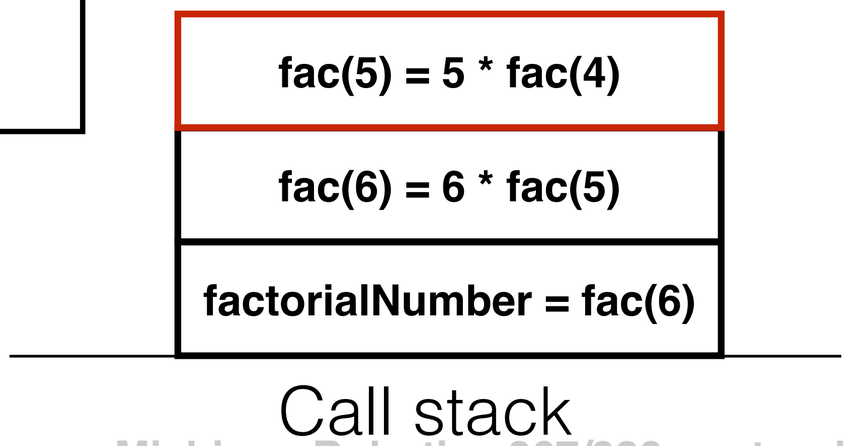


Function call stack

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

```
// define a function to compute the factorial of a numbe
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Recursive call fac(5) pushes new local variable scope

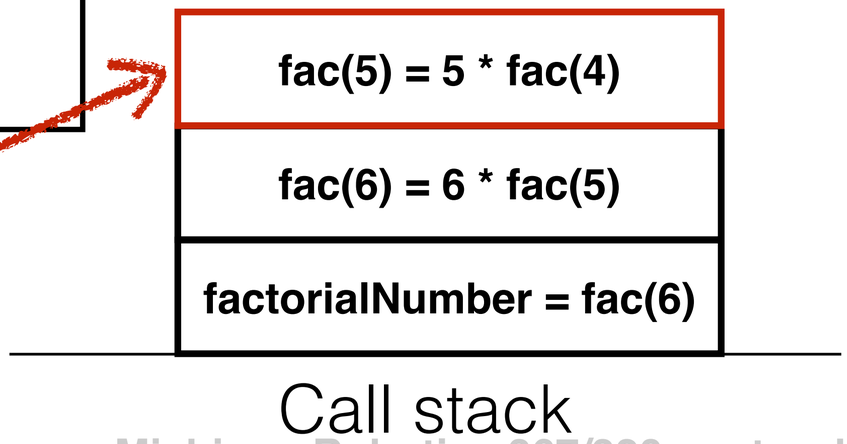


Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

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

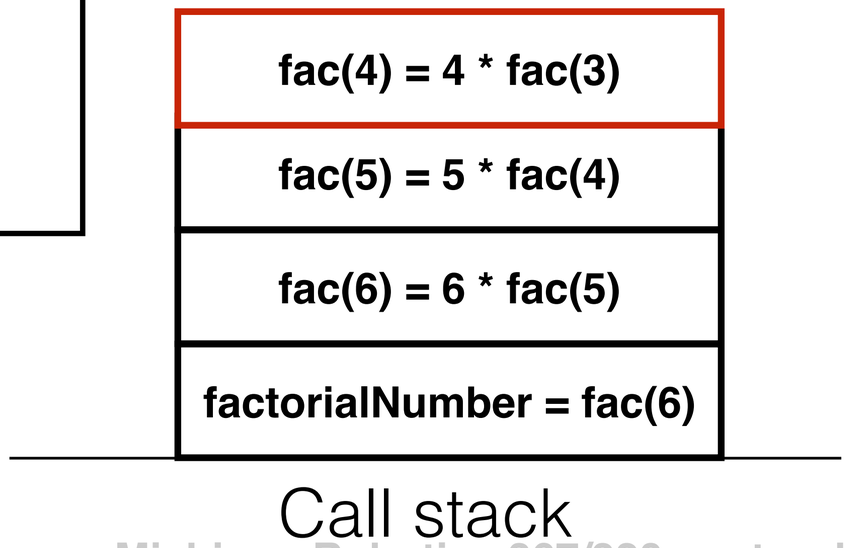


Function call stack

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

```
// define a function to compute the factorial of a numbe
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Recursive call fac(4) pushes new local variable scope

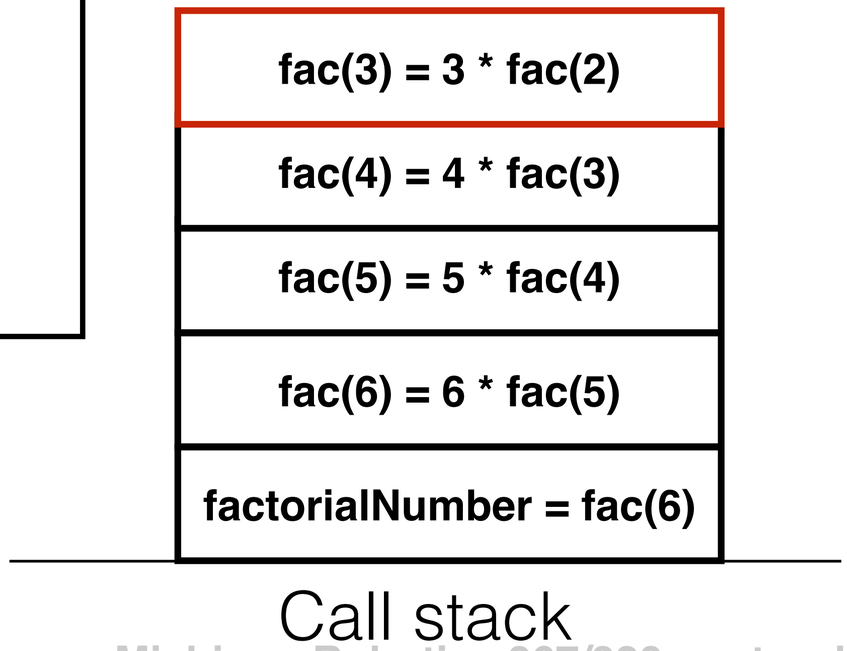


Function call stack

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

```
// define a function to compute the factorial of a numbe
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

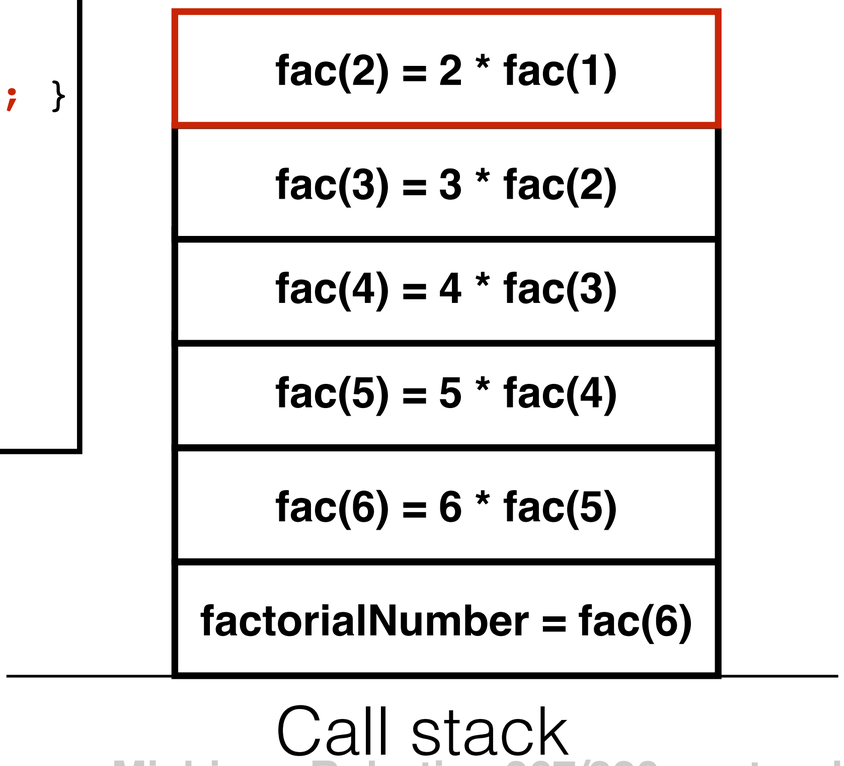
Recursive call fac(3) pushes new local variable scope



Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

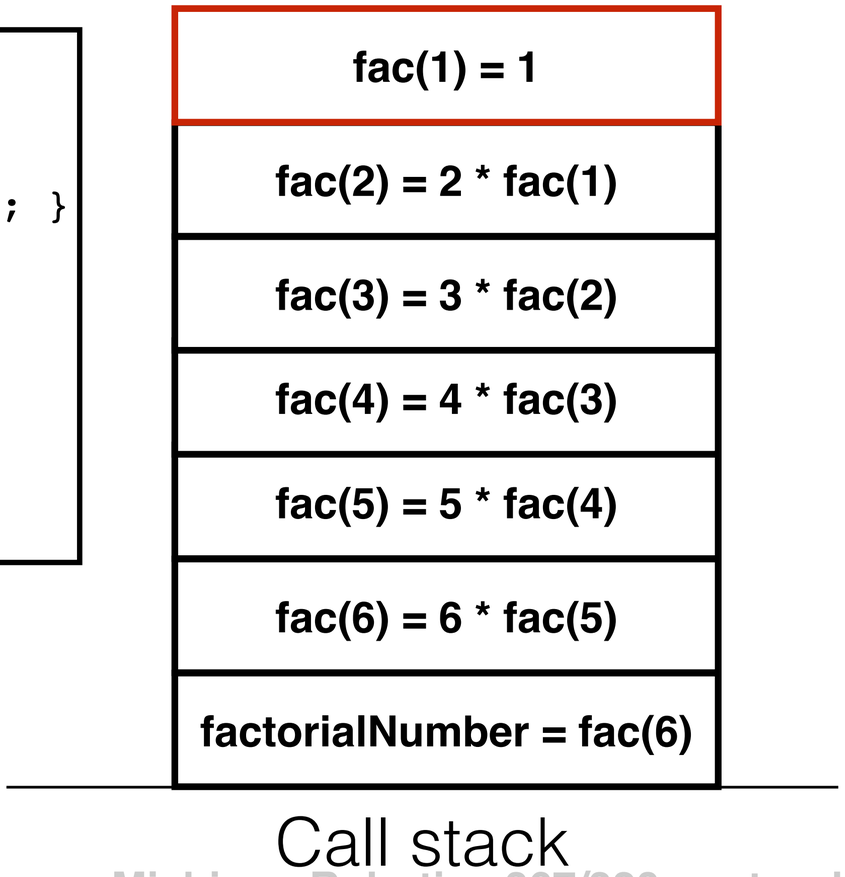


Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

Recursive calls stop when base condition encountered

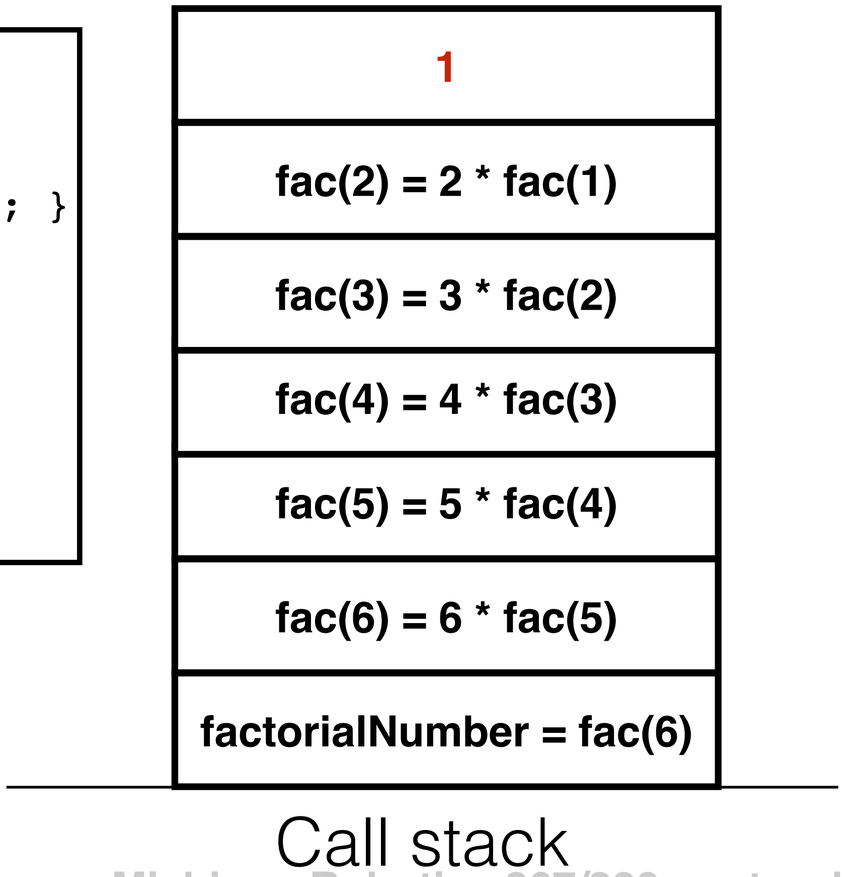


Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

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

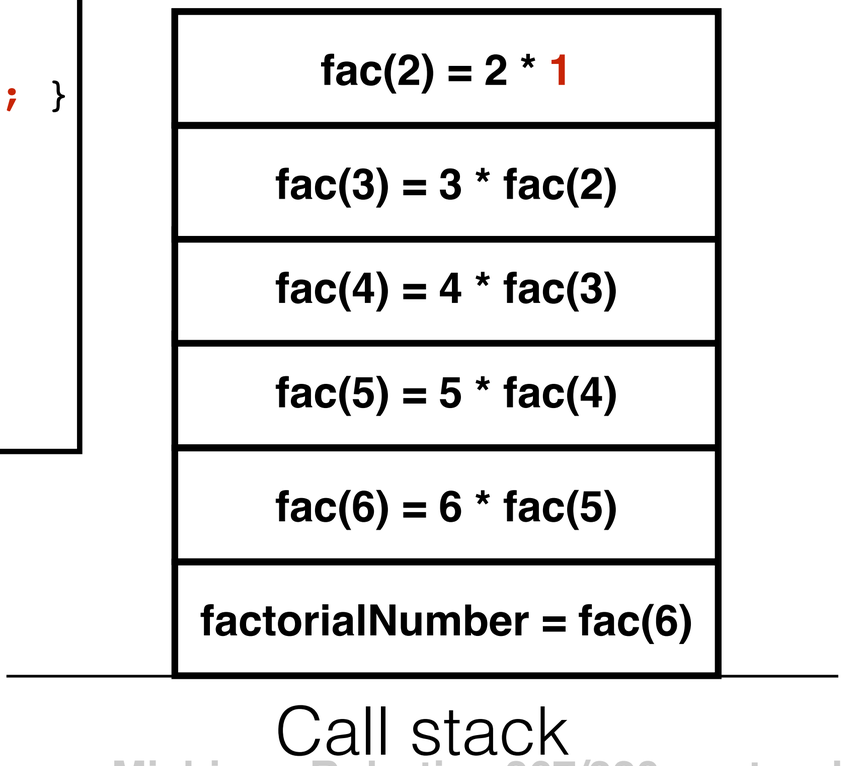


Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

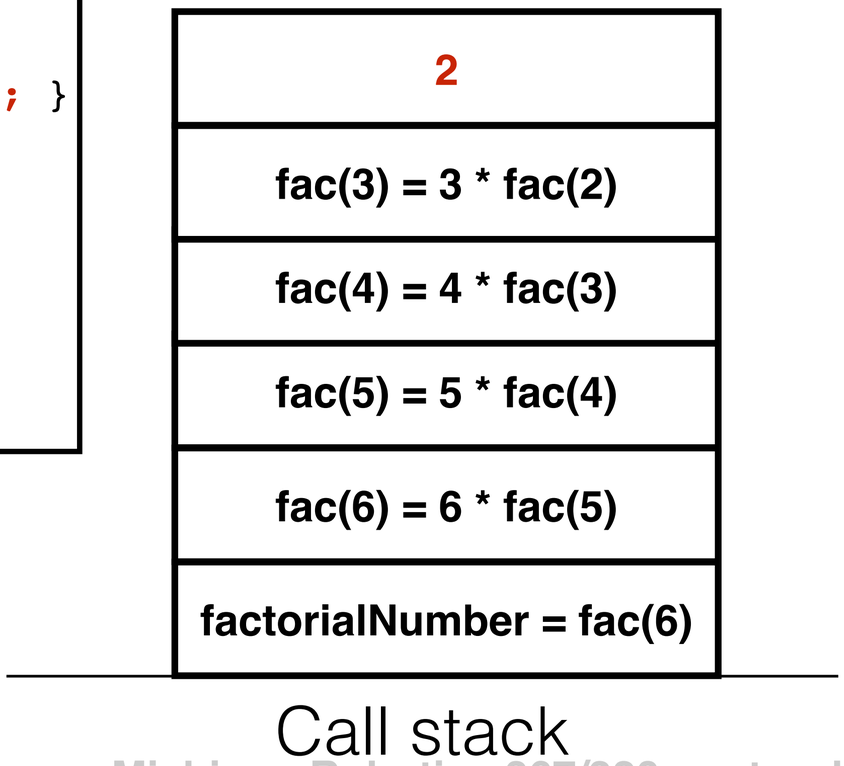
Stack pop from call stack returns a constant value to calling function



Function call stack

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

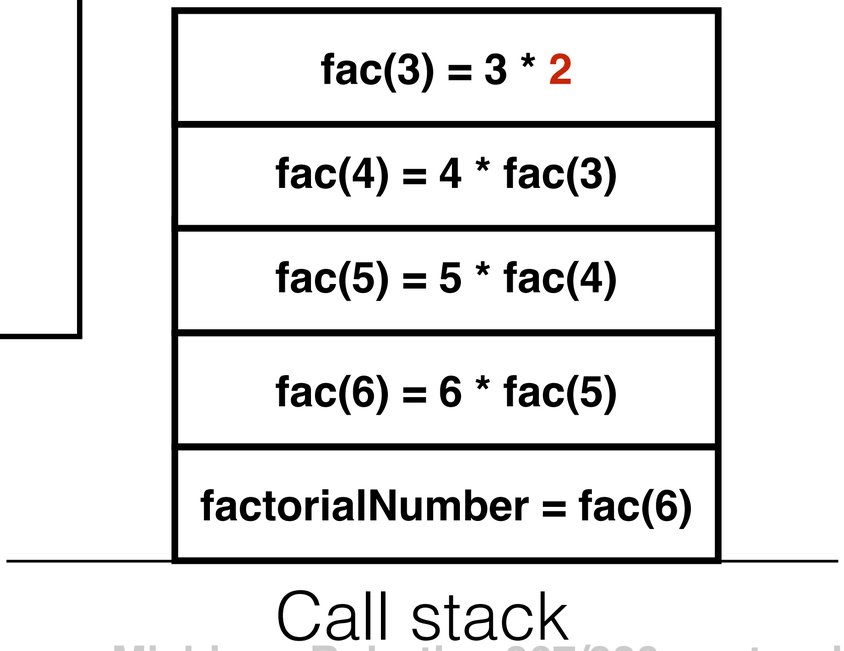
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

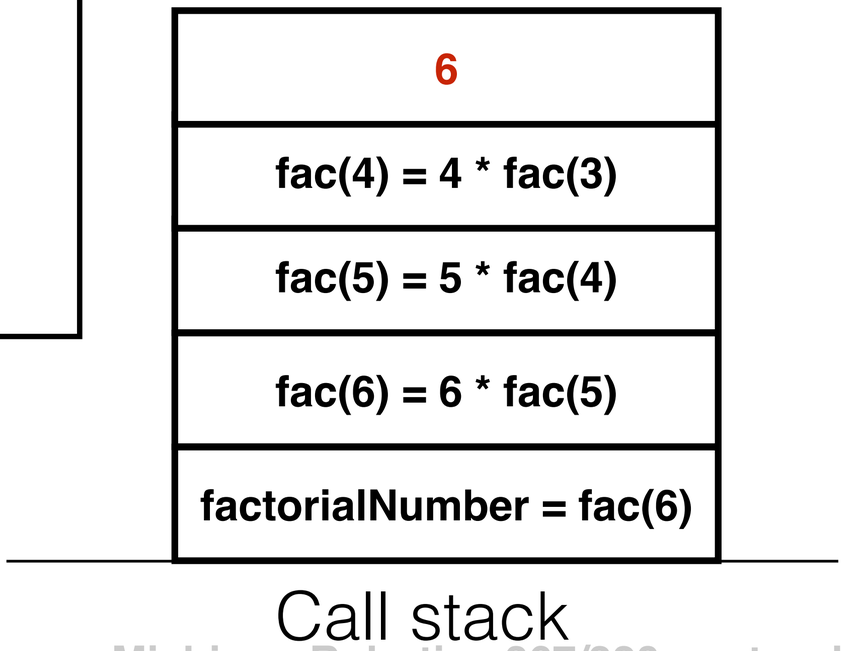
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

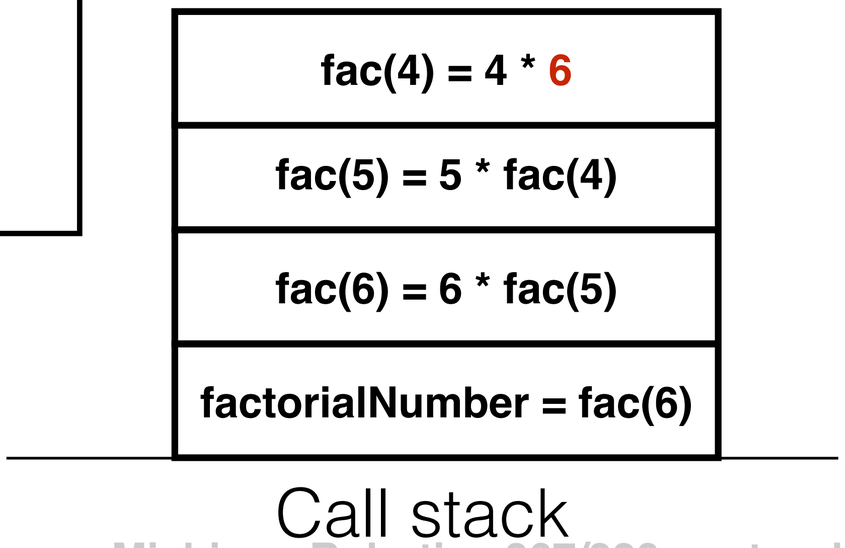
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

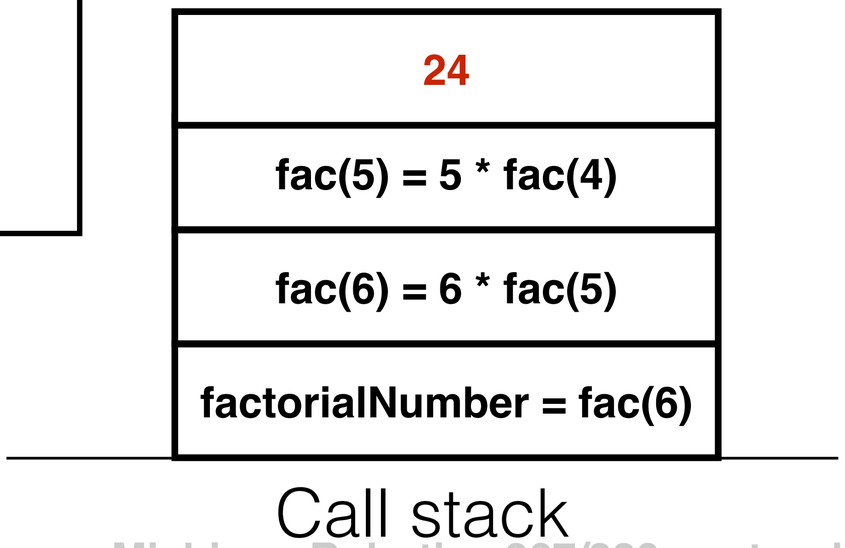
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

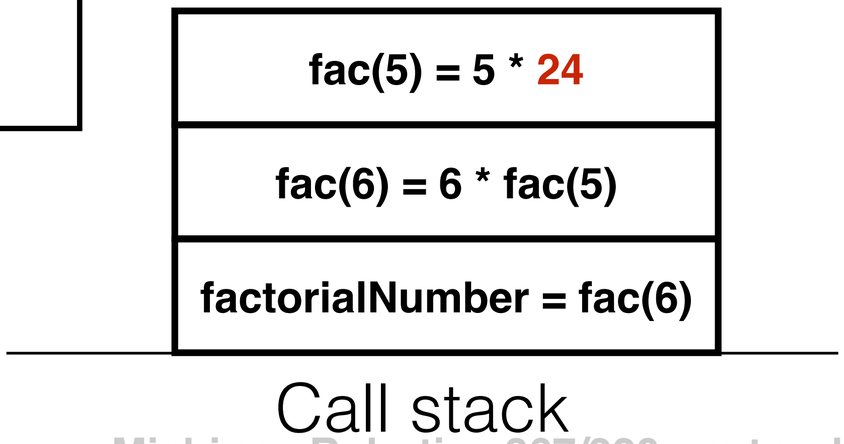
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

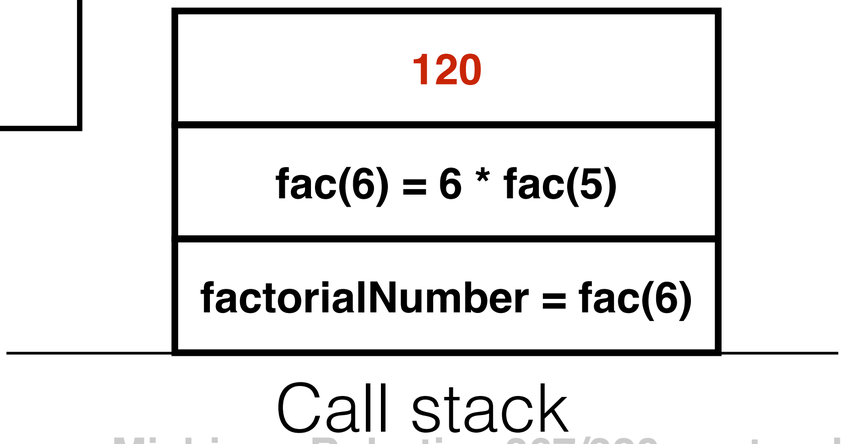
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

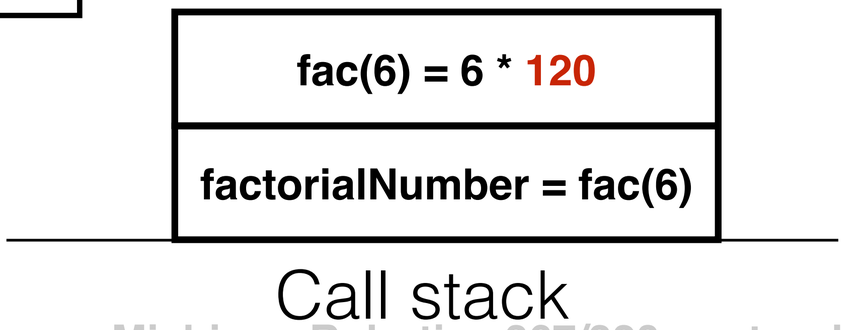
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

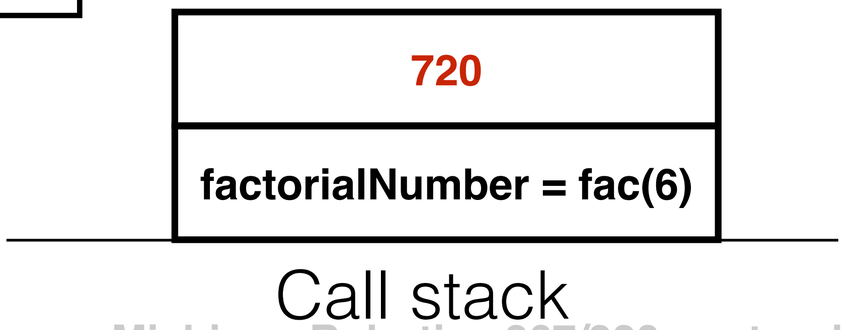
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```



Function call stack

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

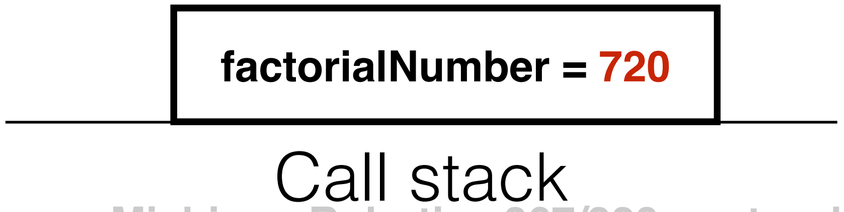
```
// define a function to compute the factorial of a number
fac = function factorialFunction(inputNumber) {
   if (inputNumber > 1) // recursive case
       { return inputNumber * factorialFunction(inputNumber-1); }
   else { return 1; }  // base case
}

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

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```

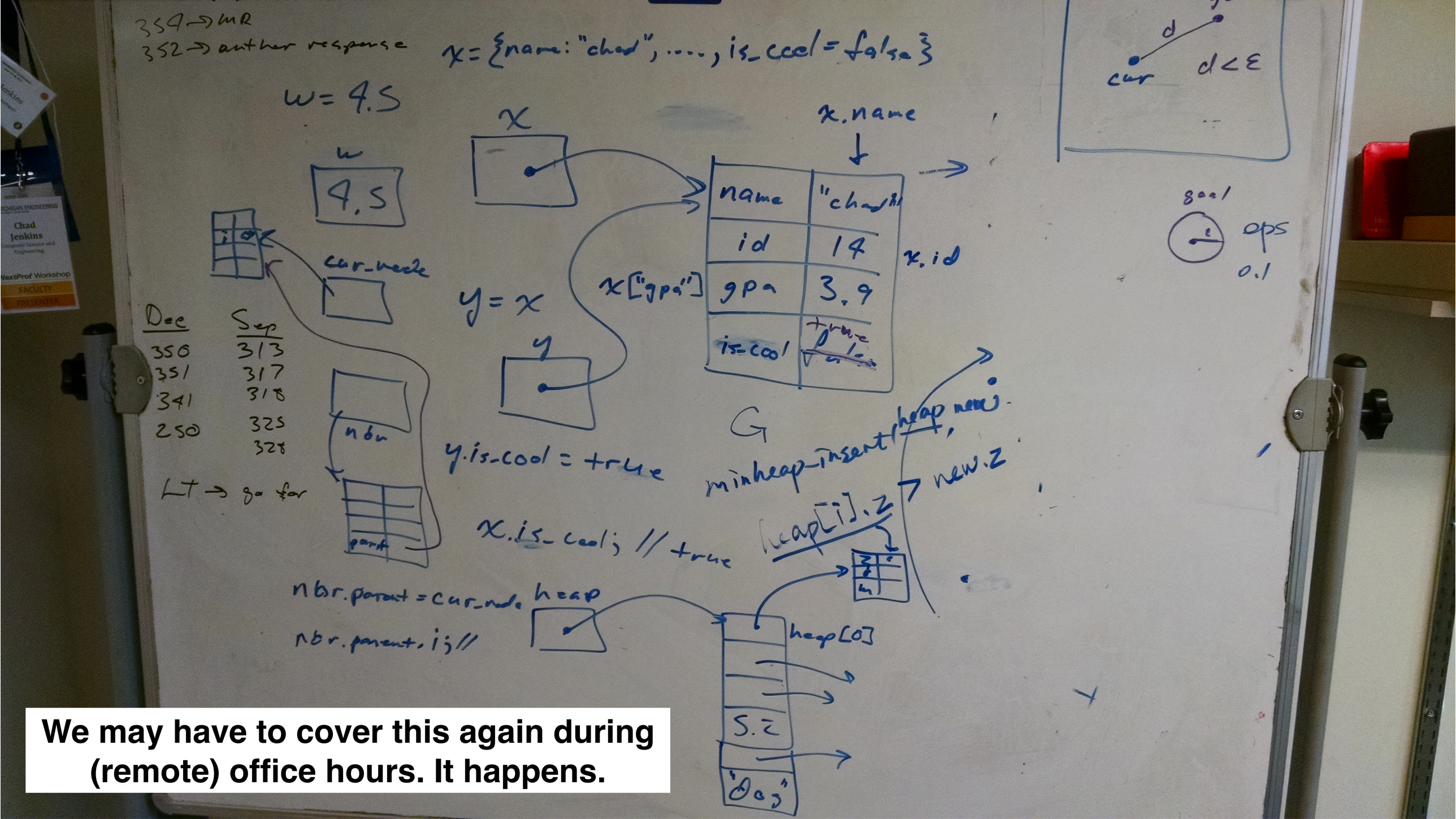


Function call stack

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

```
// define a function to compute the factorial of a number
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 . . .

// evaluate this function of the number 6
factorialNumber = fac(6);  // 720
```





We may have to cover this again during (remote) office hours. It happens.

The browser console
can be a good friend

The Browser Console

- Provided by the web browser to:
  - Log information associated with running 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

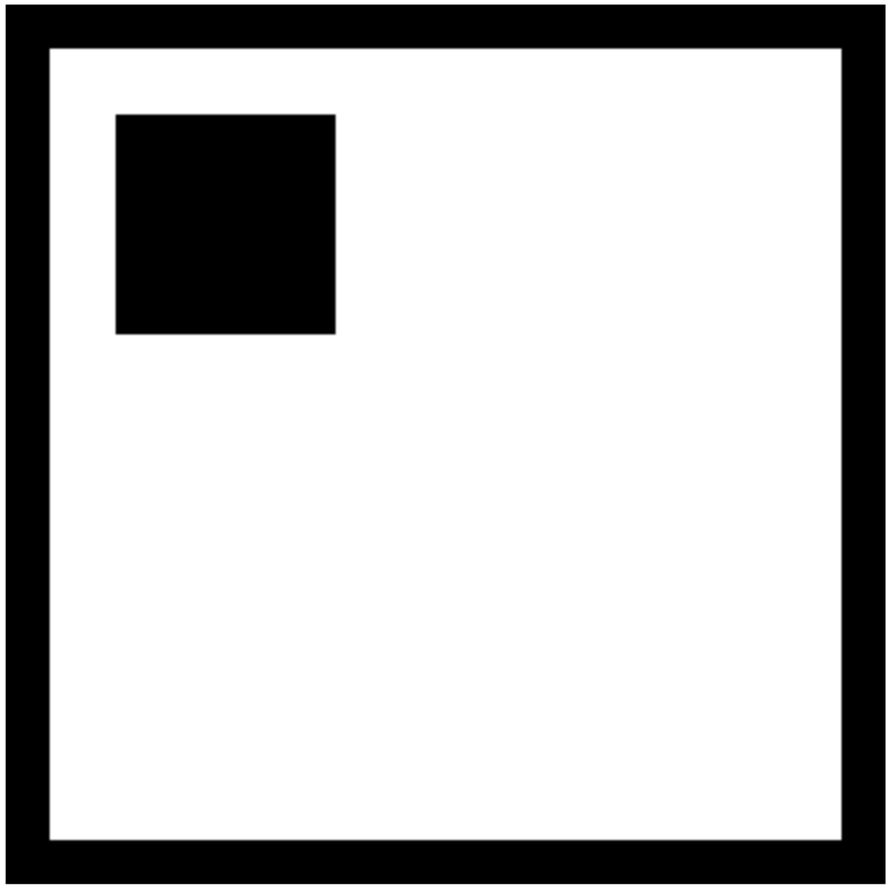
hello.html

html hello world

Chad

was here

new paragraph



Cut

Window

int

nd

loper

gn in to S

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 | Perf

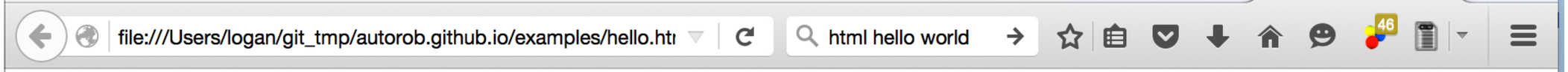
Net | CSS | JS | Security | Logging | Clear

```
>> 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



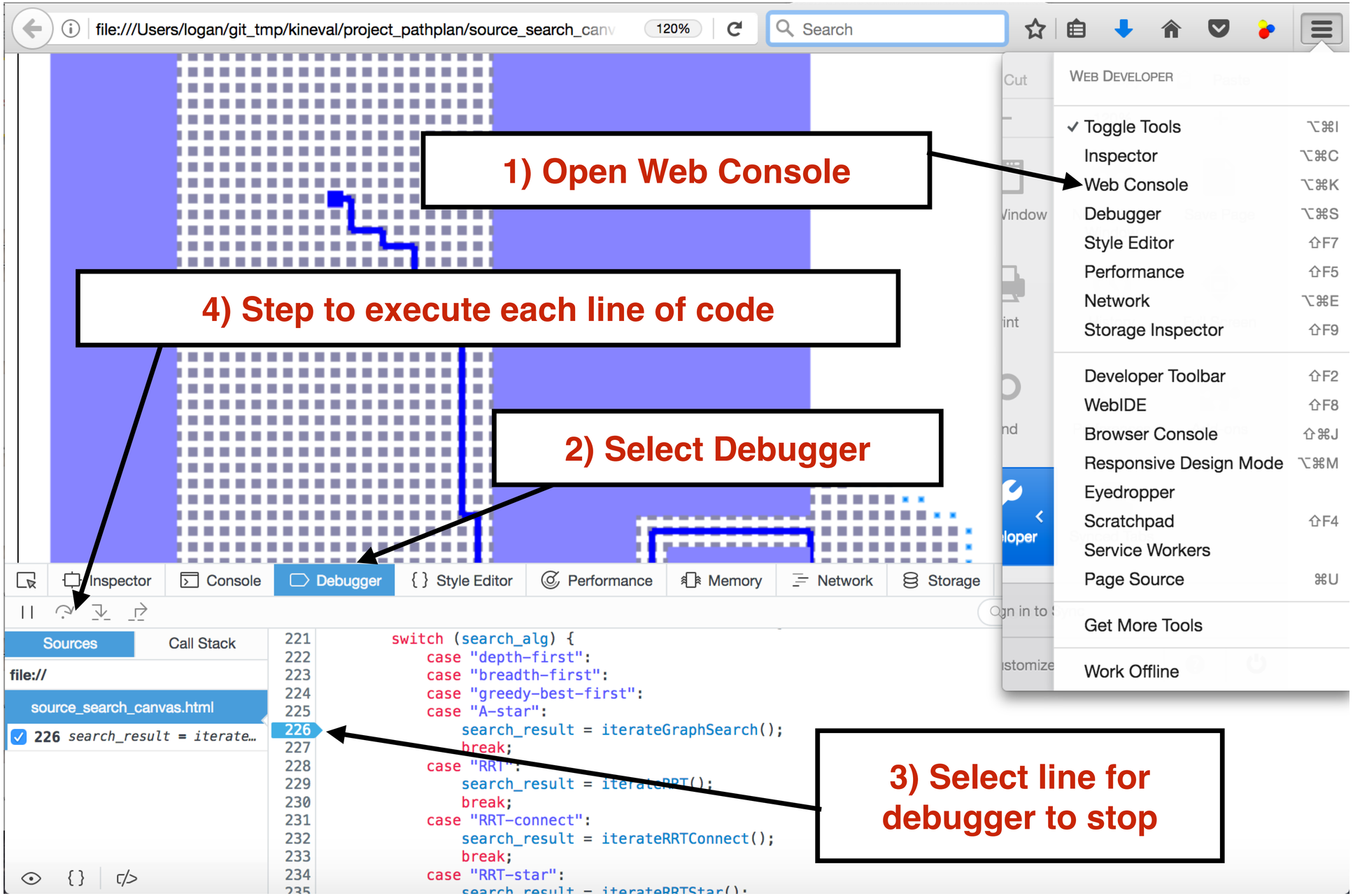
wiped everything out and replaced with more things to say 0 1 2 3 4 5 6 7 8 9

```
>> 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 ⌥⌘I
Inspector ⌥⌘C
Web Console ⌥⌘K
Debugger ⌥⌘S
Style Editor ⇧F7
Performance ⇧F5
Network ⌥⌘E
Storage Inspector ⇧F9
Developer Toolbar ⇧F2
WebIDE ⇧F8
Browser Console ⇧⌘J
Responsive Design Mode ⌥⌘M
Eyedropper
Scratchpad ⇧F4
Service Workers
Page Source ⌘U
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…
221 switch (search_alg) {
222 case "depth-first":
223 case "breadth-first":
224 case "greedy-best-first":
225 case "A-star":
226 search_result = iterateGraphSearch();
227 break;
228 case "RRT":
229 search_result = iterateRRT();
230 break;
231 case "RRT-connect":
232 search_result = iterateRRTConnect();
233 break;
234 case "RRT-star":

Let’s try an animation example

Preview slides from lectures during the Fall 2020 offering of AutoRob are provided. These preview slides will be replaced with recorded lectures for Winter 2022 as the videos become available.

Date	Topic	Reading	Project I Quiz
Jan 5	Initialization: "So, where is my robot?", "What is a Robot OS?", Course administration and logistics [Lecture Video] [Session recording (UM only)] What is a robot?: Brief history and definitions for robotics	Spong Ch.1 Corke Ch.1	Setup git repository Out: Path Planning
Jan 7	Lab Session: Git-ing started with git, JavaScript, and KinEval [Session recording (UM only)]		
	Week 2		
Jan 10	shortest paths, A-star, Priority queues and binary heaps [Lecture Video]	Wikipedia	
Jan 12	JavaScript and AutoRob workflow: Project workflow with git, JS/HTML5 tutorial, Document Object Model, Version Control, LaTeX math mode, Licensing, Michigan Honor License	Crockford, HTML Sandbox, hello.html, JavaScript by Example, hello_anim, hello_anim_text	
Jan 14	367 Lab: KinEval Path Planning code overview		
	Week 3		
Jan 17	**No course meeting** - Martin Luther King, Jr. Day UM Martin Luther King Jr. Symposium Help broaden participation in computing and robotics		
Jan 19	Dynamical Simulation: Simple pendulum, Lagrangian equation(s) of motion, Initial value problem, Explicit integrators: Euler, Verlet, and	Spong Ch.7 I Corke Ch.9 Euler's Method	

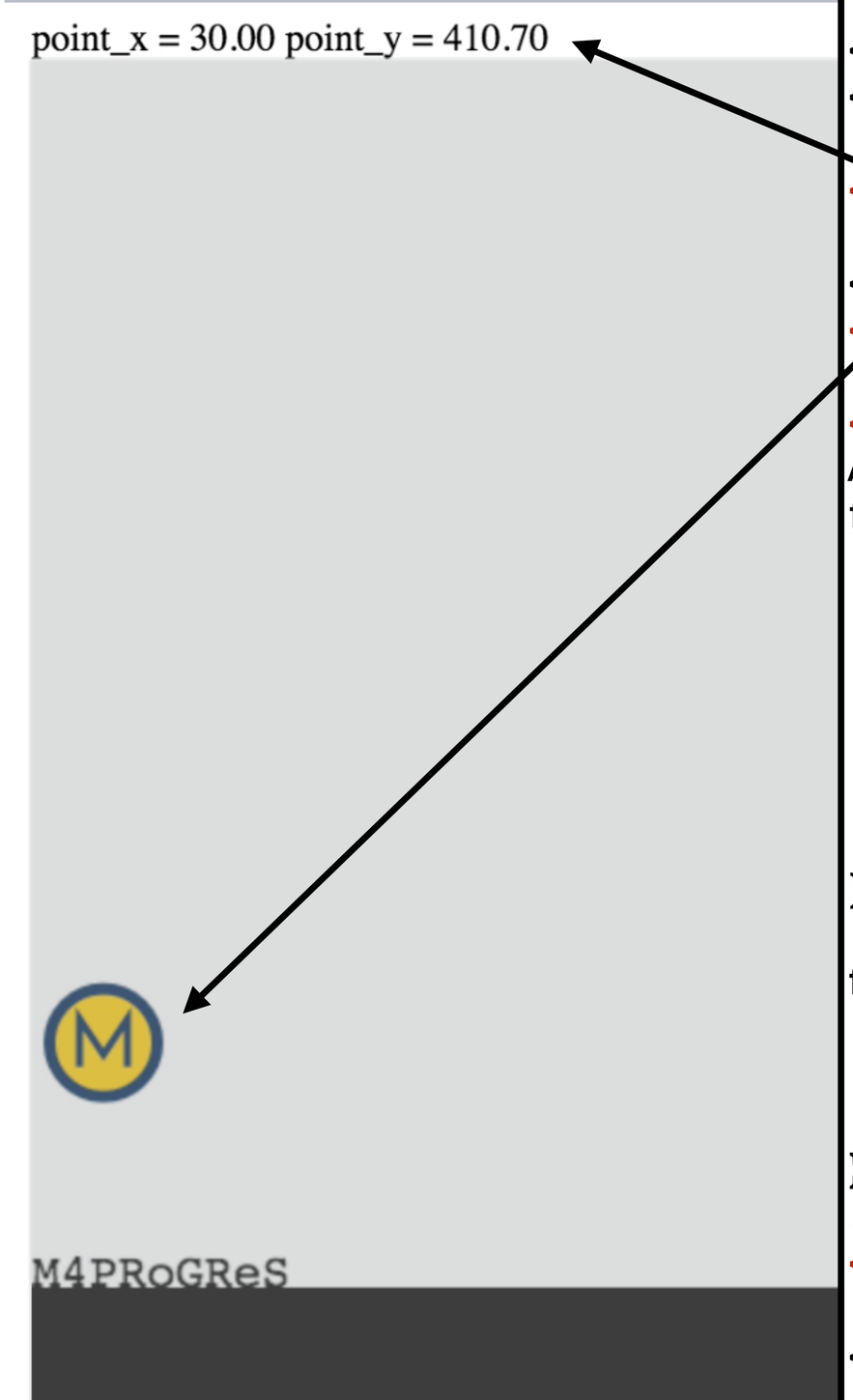
JavaScript/HTML5 animation example

point_x = 30.00 point_y = 410.70

hello_anim example

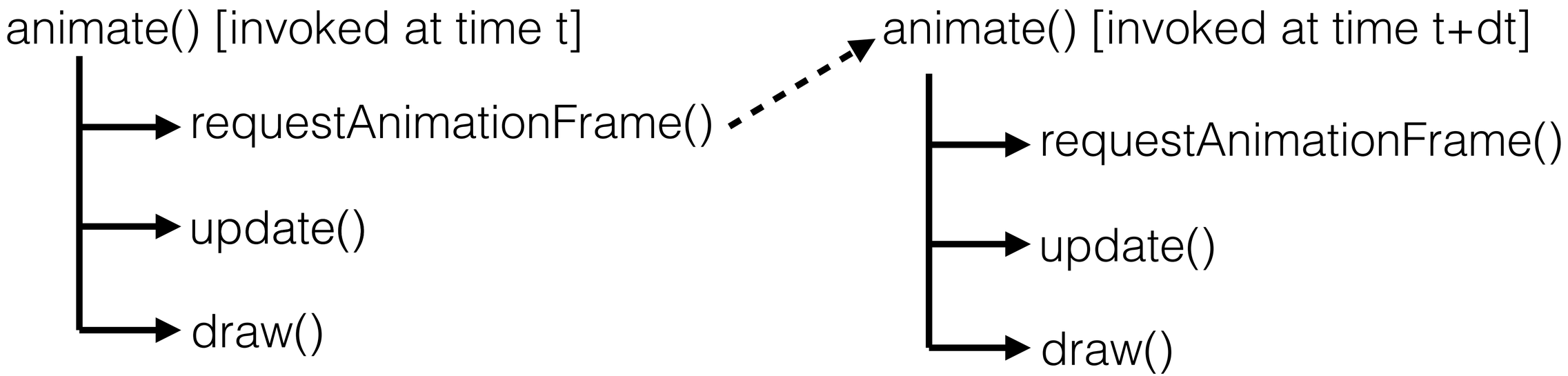
http://autorob.org/examples/hello_anim.html

M4PRoGReS




```
<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.

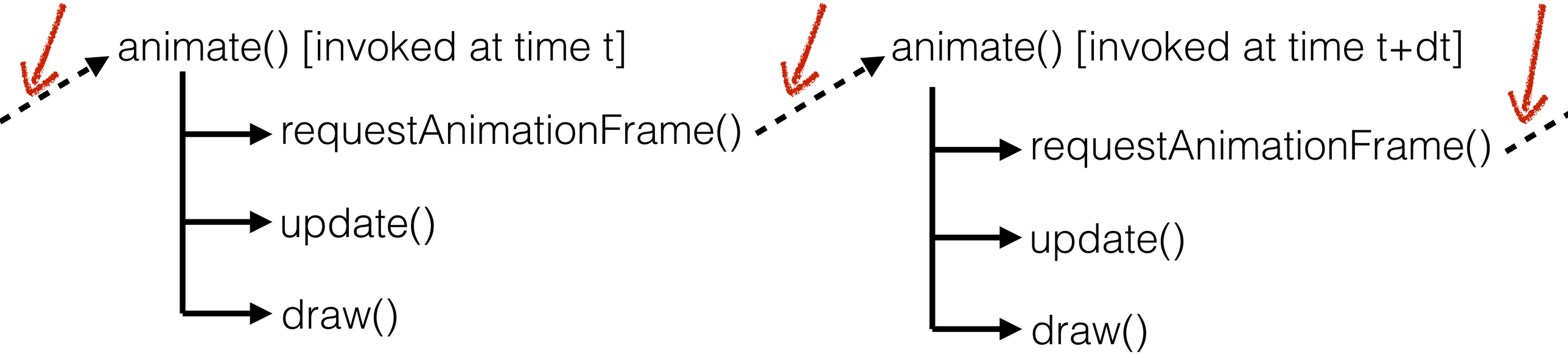
IMPORTANT to avoid code that blocks in animate()

```
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
```

callback from time t-dt

callback from time t

callback from time t+dt



requestAnimationFrame() will have browser call animate() again.

IMPORTANT to avoid code that blocks in animate()

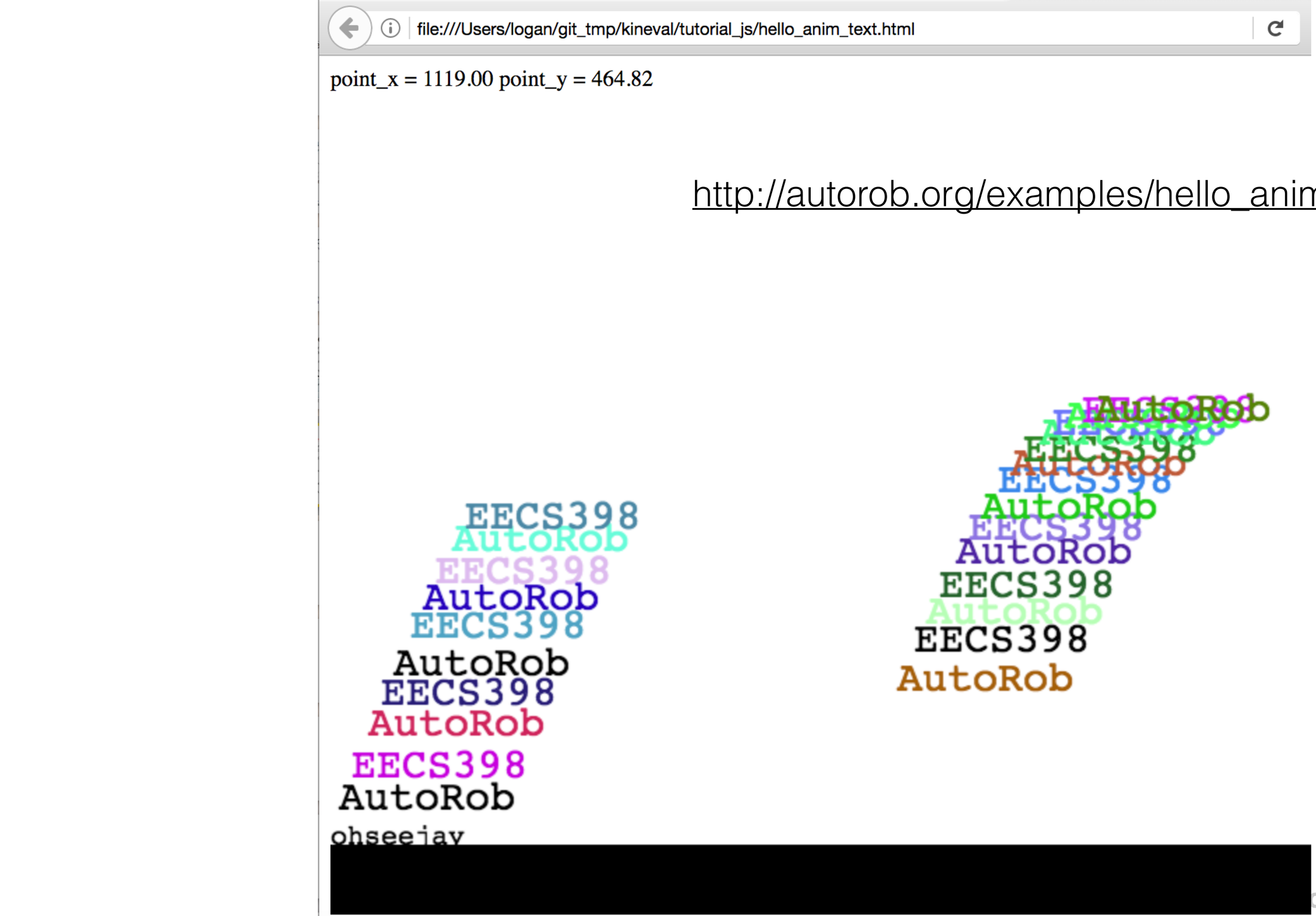
point_x = 30.00 point_y = 410.70

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 object

Run one more animation example on your own


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

Many examples available online

GitHub, Inc. (US) https://github.com/odestcj/superquadric/

Search

This repository Search Pull requests Issues Gist

odestcj / superquadric

Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settin

super quick superquadric Implementation of Barr's superquadric surface point visualization

2 commits 1 branch 0 releas

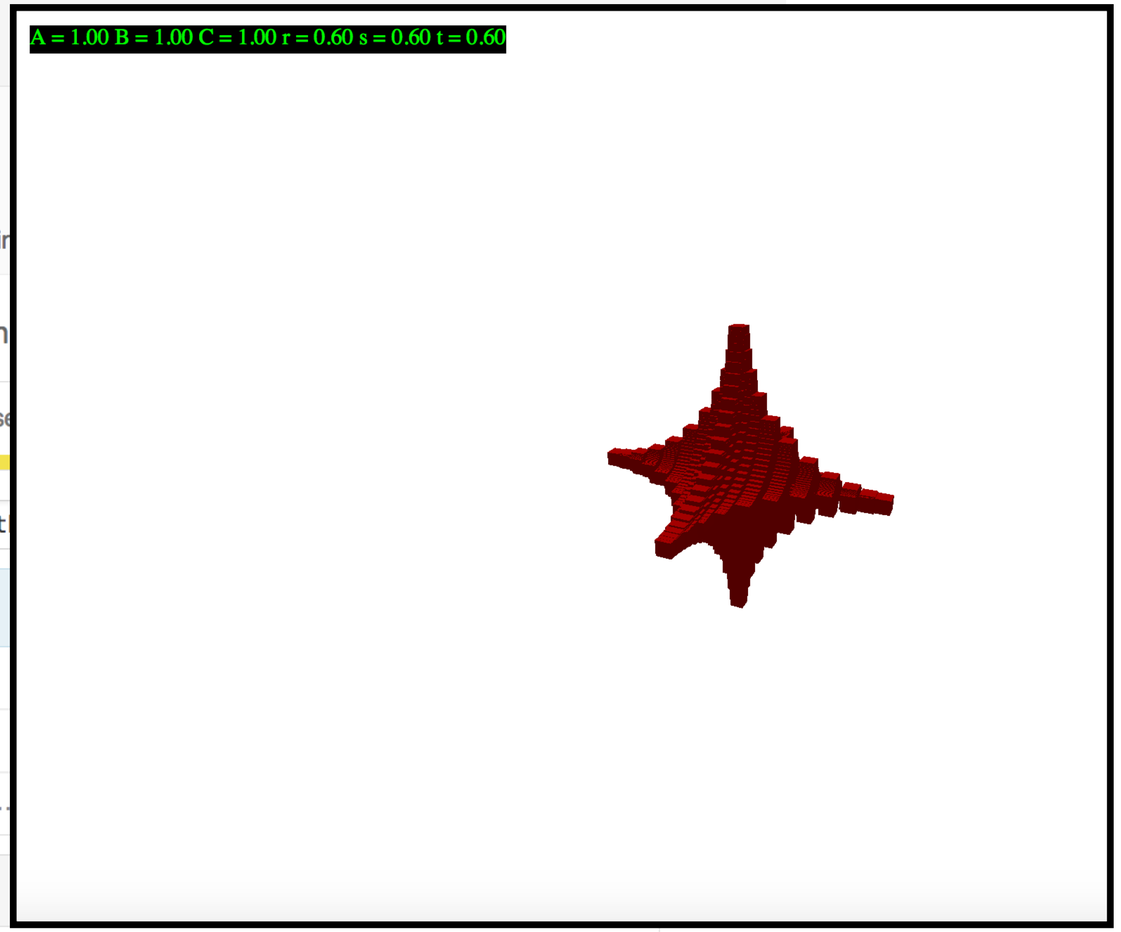
Branch: master New pull request New file Find file SSH git@git

odestcj fixed Math.sign not being in Chrome and keyboard input; added paramet…

js	Initial commit, working version of superquadric, but lighting and tes…
README	Initial commit, working version of superquadric, but lighting and tes…
superquadric.html	fixed Math.sign not being in Chrome and keyboard input; added paramet…

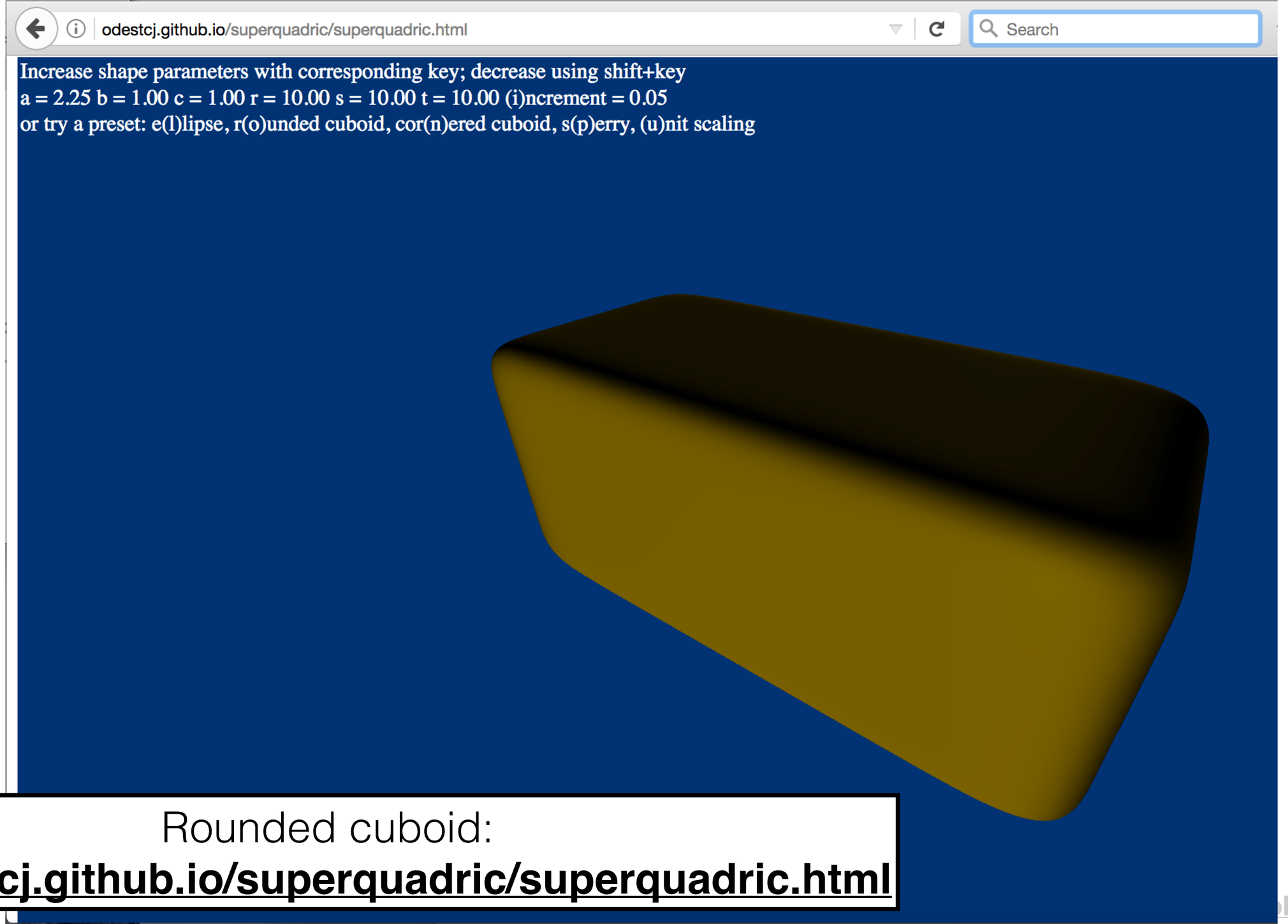
README

```
super quick superquadric
Implementation of Barr's superquadric surface point vi
in HTML5/JavaScript and threejs

@author odestcj / https://github.com/odestcj

Change view by click-and-drag mouse

Increase shape parameters with r,s,t keys;
Decrease shape parameters with R,S,T
```



Another example:

https://github.com/odestcj/superquadric



Rounded cuboid:

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

KinEval Stencil Overview

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

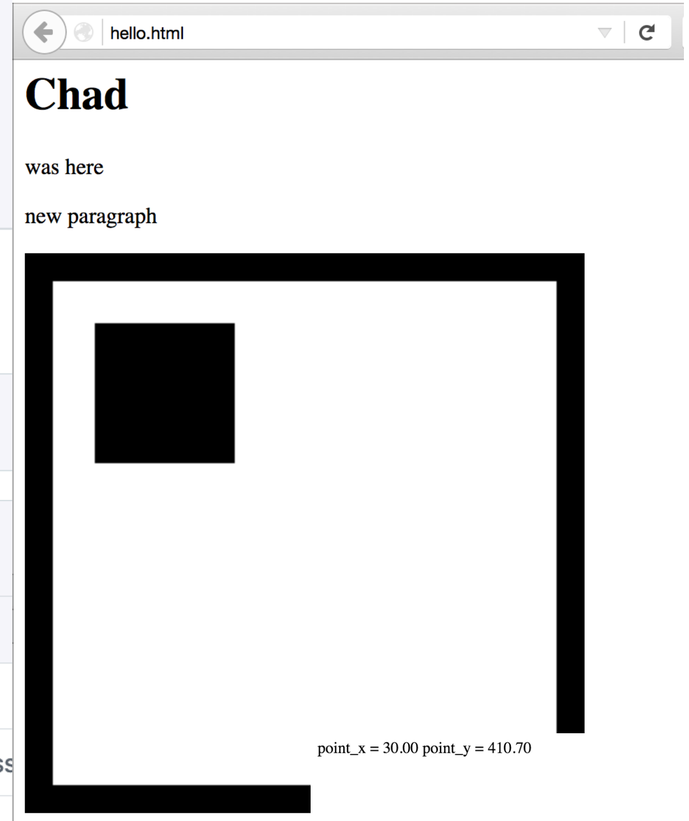
Code | Pull requests | Actions | Projects | Wiki | Security

master | 1 branch | 0 tags

This branch is up to date with master.

tonyop Update LICENSE

js	initial commit Fall 2018
kineval	Add matrix requirement for IK
project_pathplan	Makes assignment 6 drawing work more like ass
project_pendularm	fixed control set to 0 and 2d array problem in pe
robots	initial commit Fall 2018
tutorial_heapsort	initial commit Fall 2018
tutorial_js	initial commit Fall 2018
worlds	initial commit Fall 2018
LICENSE	Update LICENSE
README.md	initial commit Fall 2018
advanced_extensions.html	Add simple advanced_extensions.html
home.html	Factorize kineval stencil for FK gradining



Tutorial Examples

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

Code | Pull requests | Actions | Projects | Wiki | Security

master | 1 branch | 0 tags

This branch is up to date with master.

tonyop Update LICENSE

js	initial commit Fall 2018
kineval	Add matrix requirement for IK
project_pathplan	Makes assignment 6 drawing work more like ass
project_pendularm	fixed control set to 0 and 2d array problem in pe
robots	initial commit Fall 2018
tutorial_heapsort	initial commit Fall 2018
tutorial_js	initial commit Fall 2018
worlds	initial commit Fall 2018
LICENSE	Update LICENSE
README.md	initial commit Fall 2018
advanced_extensions.html	Add simple advanced_extensions.html
home.html	Factorize kineval stencil for FK gradining

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

Project 1:
2D Path Planning

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

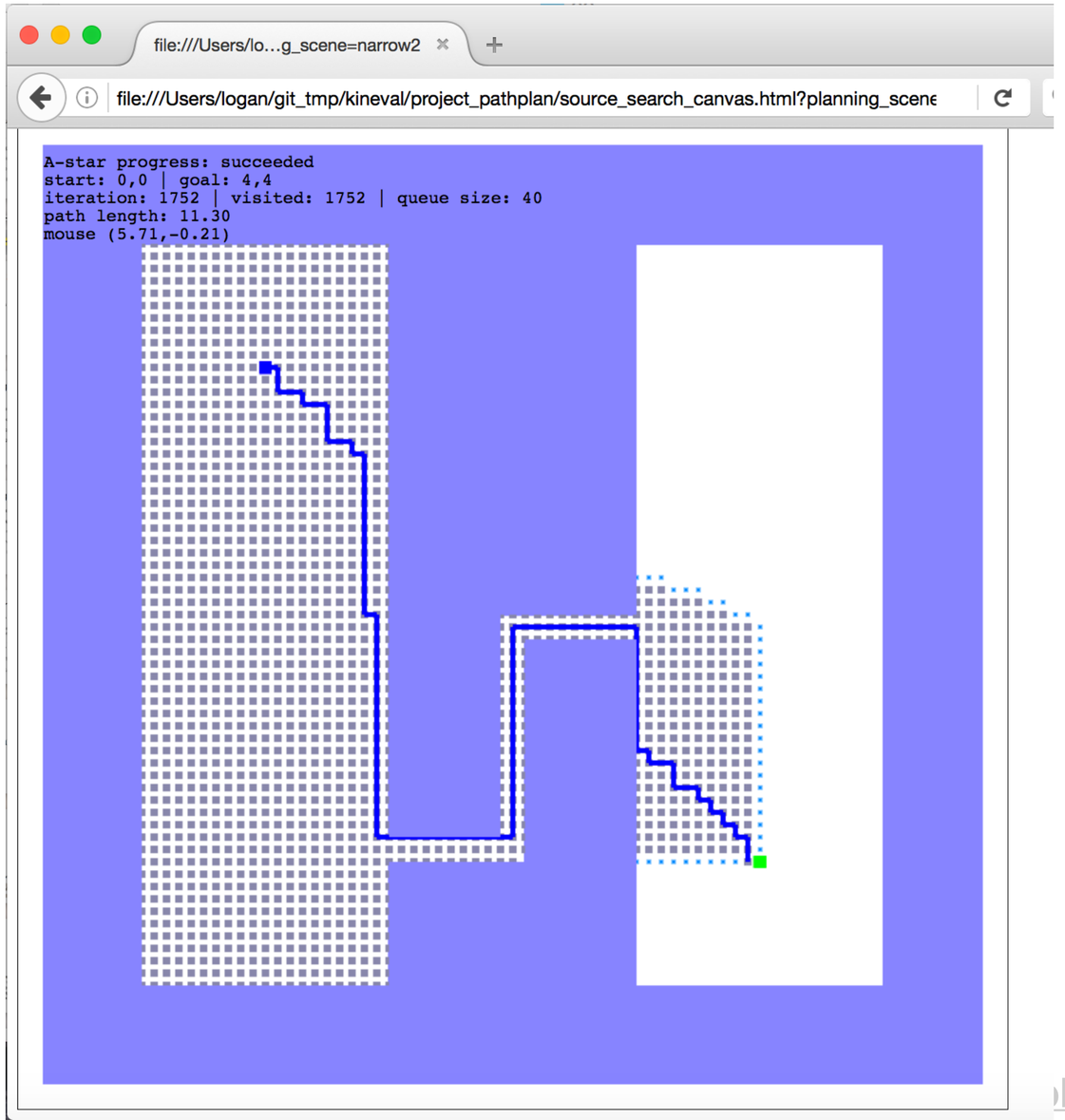
Code | Pull requests | Actions | Projects | Wiki | Security

master | 1 branch | 0 tags

This branch is up to date with master.

tonyop Update LICENSE

js	initial commit Fall 2018
kineval	Add matrix requirement for IK
project_pathplan	Makes assignment 6 drawing work more like ass
project_pendularm	fixed control set to 0 and 2d array problem in pe
robots	initial commit Fall 2018
tutorial_heapsort	initial commit Fall 2018
tutorial_js	initial commit Fall 2018
worlds	initial commit Fall 2018
LICENSE	Update LICENSE
README.md	initial commit Fall 2018
advanced_extensions.html	Add simple advanced_extensions.html
home.html	Factorize kineval stencil for FK gradining



Project 2: Pendulum Dynamics/Control

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

Code | Pull requests | Actions | Projects | Wiki | Security

master | 1 branch | 0 tags

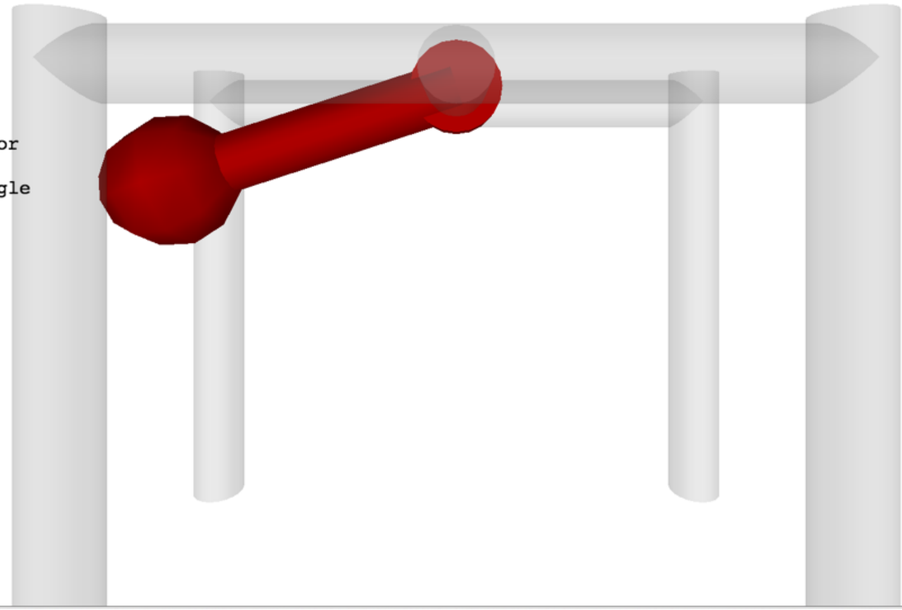
This branch is up to date with master.

tonyop Update LICENSE

js	initial commit Fall 2018
kineval	Add matrix requirement for IK
project_pathplan	Makes assignment 6 drawing work more like as
project_pendularm	fixed control set to 0 and 2d array problem in p
robots	initial commit Fall 2018
tutorial_heapsort	initial commit Fall 2018
tutorial_js	initial commit Fall 2018
worlds	initial commit Fall 2018
LICENSE	Update LICENSE
README.md	initial commit Fall 2018
advanced_extensions.html	Add simple advanced_extensions.html
home.html	Factorize kineval stencil for FK gradining

file:///Users/logan/git_tmp/kineval/pendularm/pendularm1.html

```
System
t = 162.00 dt = 0.05
integrator = velocity verlet
x = -1.26
x_dot = -0.00
x_desired = -1.26

Servo: active
u = -37.32
kp = 1500.00
kd = 15.00
ki = 150.10

Pendulum
mass = 2.00
length = 2.00
gravity = 9.81

Keys
[0-4] - select integrator
a/d - apply user force
q/e - adjust desired angle
c - toggle servo
s - disable servo
```



Projects 3-6: Robot Modeling and Control

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

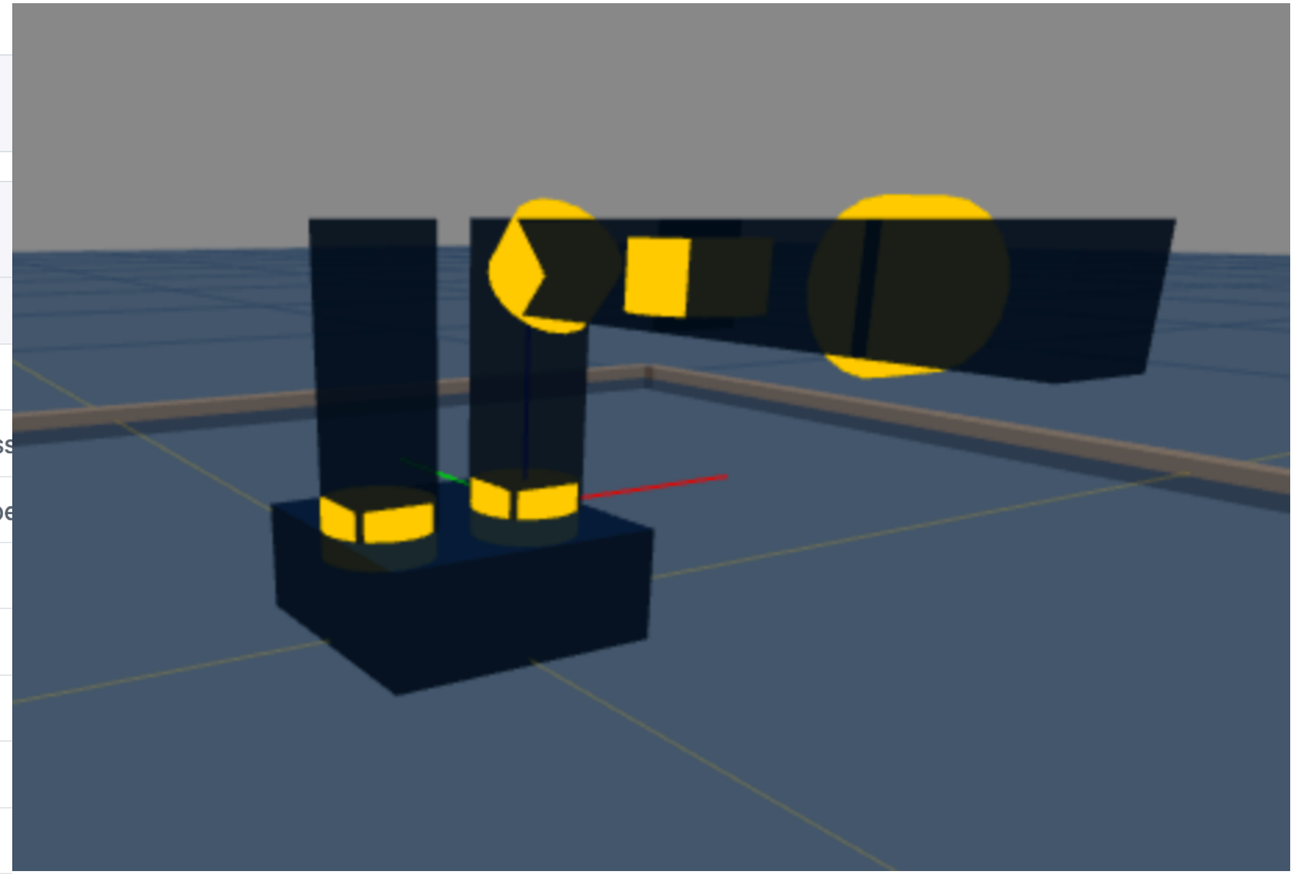
Code | Pull requests | Actions | Projects | Wiki | Security

master | 1 branch | 0 tags

This branch is up to date with master.

tonyop Update LICENSE

js	initial commit Fall 2018
kineval	Add matrix requirement for IK
project_pathplan	Makes assignment 6 drawing work more like ass
project_pendularm	fixed control set to 0 and 2d array problem in pe
robots	initial commit Fall 2018
tutorial_heapsort	initial commit Fall 2018
tutorial_js	initial commit Fall 2018
worlds	initial commit Fall 2018
LICENSE	Update LICENSE
README.md	initial commit Fall 2018
advanced_extensions.html	Add simple advanced_extensions.html
home.html	Factorize kineval stencil for FK gradining



Who owns the code?

Both of us

master | 1 branch | 0 tags | Go to file | Add file | Code | Use this template

This branch is 1 commit ahead of autorob:master. Contribute

tonyop Update LICENSE 26c8fb8 10 days ago 23 commits

js	initial commit Fall 2018	3 years ago
kineval	Add matrix requirement for IK	15 months ago
project_pathplan	Makes assignment 6 drawing work more like assignment 1 for familia...	14 months ago
project_pendularm	fixed control set to 0 and 2d array problem in pendulum2.html	16 months ago
robots	initial commit Fall 2018	3 years ago
tutorial_heapsort	initial commit Fall 2018	3 years ago
tutorial_js	initial commit Fall 2018	3 years ago
worlds	initial commit Fall 2018	3 years ago
LICENSE	Update LICENSE	10 days ago
README.md	initial commit Fall 2018	3 years ago

About

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

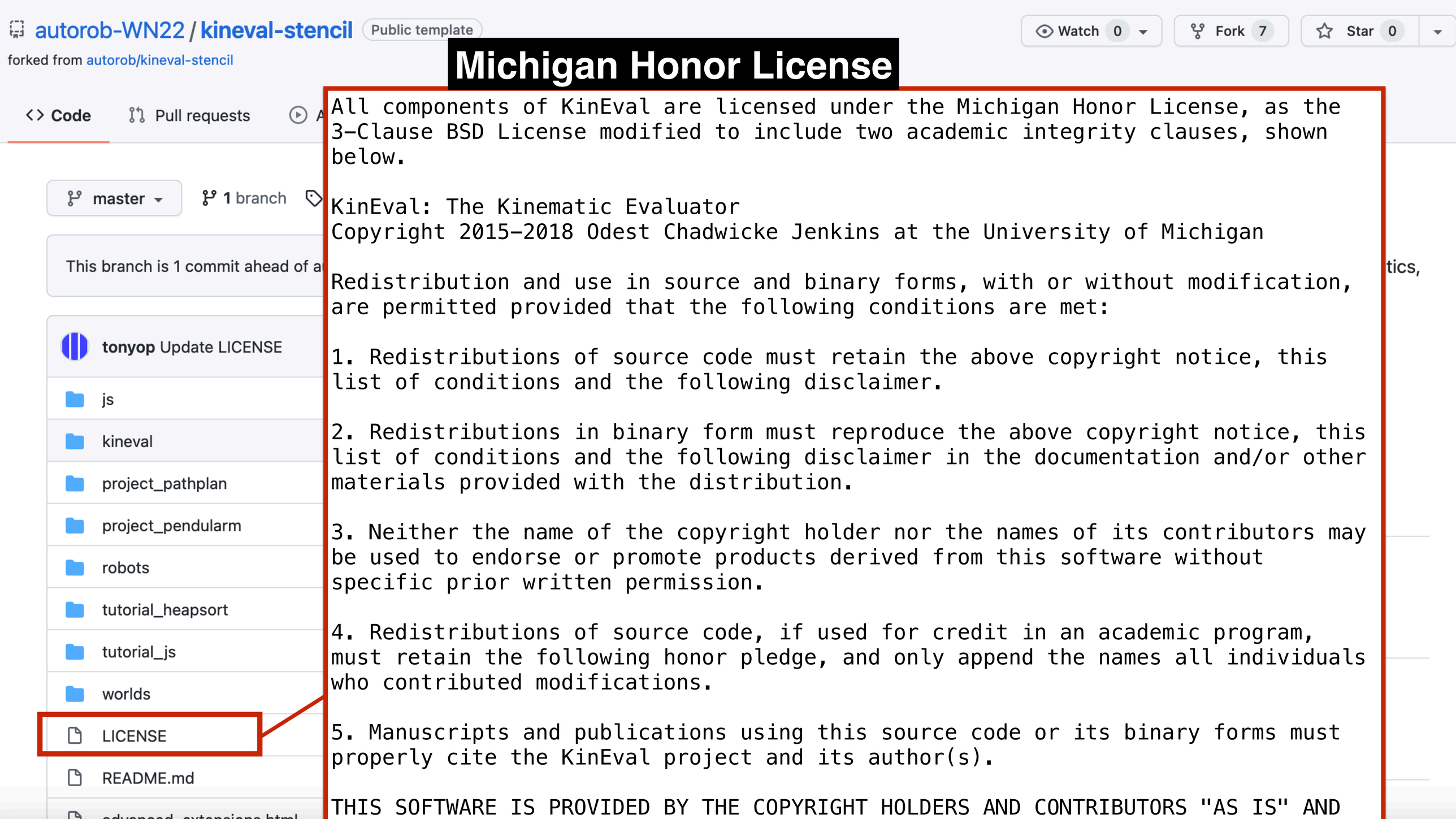
Readme
View license
0 stars
0 watching
7 forks

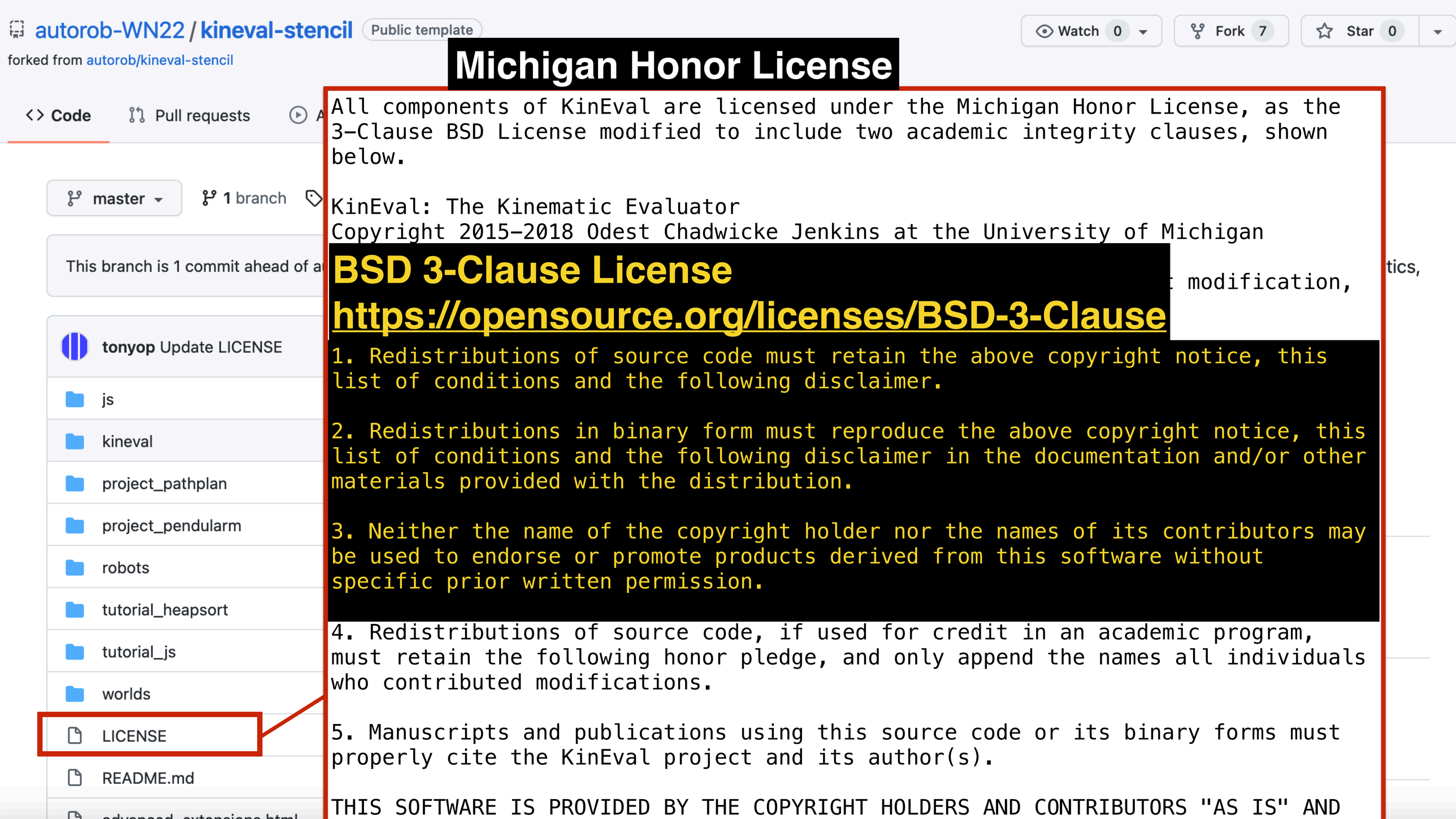
Releases

No releases published

Packages

No packages published


autorob-WN22 / kineval-stencil
Public template
forked from autorob/kineval-stencil
Watch 0
Fork 7
Star 0
Code
Pull requests
master
1 branch
This branch is 1 commit ahead of a
tonyop Update LICENSE
js
kineval
project_pathplan
project_pendularm
robots
tutorial_heapsort
tutorial_js
worlds
LICENSE
README.md
tics,
Michigan Honor License
All components of KinEval are licensed under the Michigan Honor License, as the 3-Clause BSD License modified to include two academic integrity clauses, shown below.
KinEval: The Kinematic Evaluator
Copyright 2015-2018 Odest Chadwicke Jenkins at the University of Michigan
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
4. Redistributions of source code, if used for credit in an academic program, must retain the following honor pledge, and only append the names all individuals who contributed modifications.
5. Manuscripts and publications using this source code or its binary forms must properly cite the KinEval project and its author(s).
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND


autorob-WN22 / kineval-stencil
Public template
forked from autorob/kineval-stencil
Watch 0
Fork 7
Star 0
Code
Pull requests
master
1 branch
This branch is 1 commit ahead of a
tonyop Update LICENSE
js
kineval
project_pathplan
project_pendularm
robots
tutorial_heapsort
tutorial_js
worlds
LICENSE
README.md
Michigan Honor License
All components of KinEval are licensed under the Michigan Honor License, as the 3-Clause BSD License modified to include two academic integrity clauses, shown below.
KinEval: The Kinematic Evaluator
Copyright 2015-2018 Odest Chadwicke Jenkins at the University of Michigan
BSD 3-Clause License
https://opensource.org/licenses/BSD-3-Clause
1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
4. Redistributions of source code, if used for credit in an academic program, must retain the following honor pledge, and only append the names all individuals who contributed modifications.
5. Manuscripts and publications using this source code or its binary forms must properly cite the KinEval project and its author(s).
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

Watch 0 Fork 7 Star 0

Code Pull requests A

master 1 branch

This branch is 1 commit ahead of a

tonyop Update LICENSE

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

Michigan Honor License

All components of KinEval are licensed under the Michigan Honor License, as the 3-Clause BSD License modified to include two academic integrity clauses, shown below.

KinEval: The Kinematic Evaluator
Copyright 2015-2018 Odest Chadwicke Jenkins at the University of Michigan

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Michigan Engineering Honor Code
https://bulletin.engin.umich.edu/rules/

4. Redistributions of source code, if used for credit in an academic program, must retain the following honor pledge, and only append the names all individuals who contributed modifications.

5. Manuscripts and publications using this source code or its binary forms must properly cite the KinEval project and its author(s).

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND

mes of its contributors may
is software without

tics,

autorob-WN22 / kineval-stencil Public template

forked from autorob/kineval-stencil

Watch 0 Fork 7 Star 0

Code Pull requests A

master 1 branch

This branch is 1 commit ahead of a

tonyop Update LICENSE

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

Michigan Honor License

All components of KinEval are licensed under the Michigan Honor License, as the 3-Clause BSD License modified to include two academic integrity clauses, shown below.

KinEval: The Kinematic Evaluator
Copyright 2015-2018 Odest Chadwicke Jenkins at the University of Michigan

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

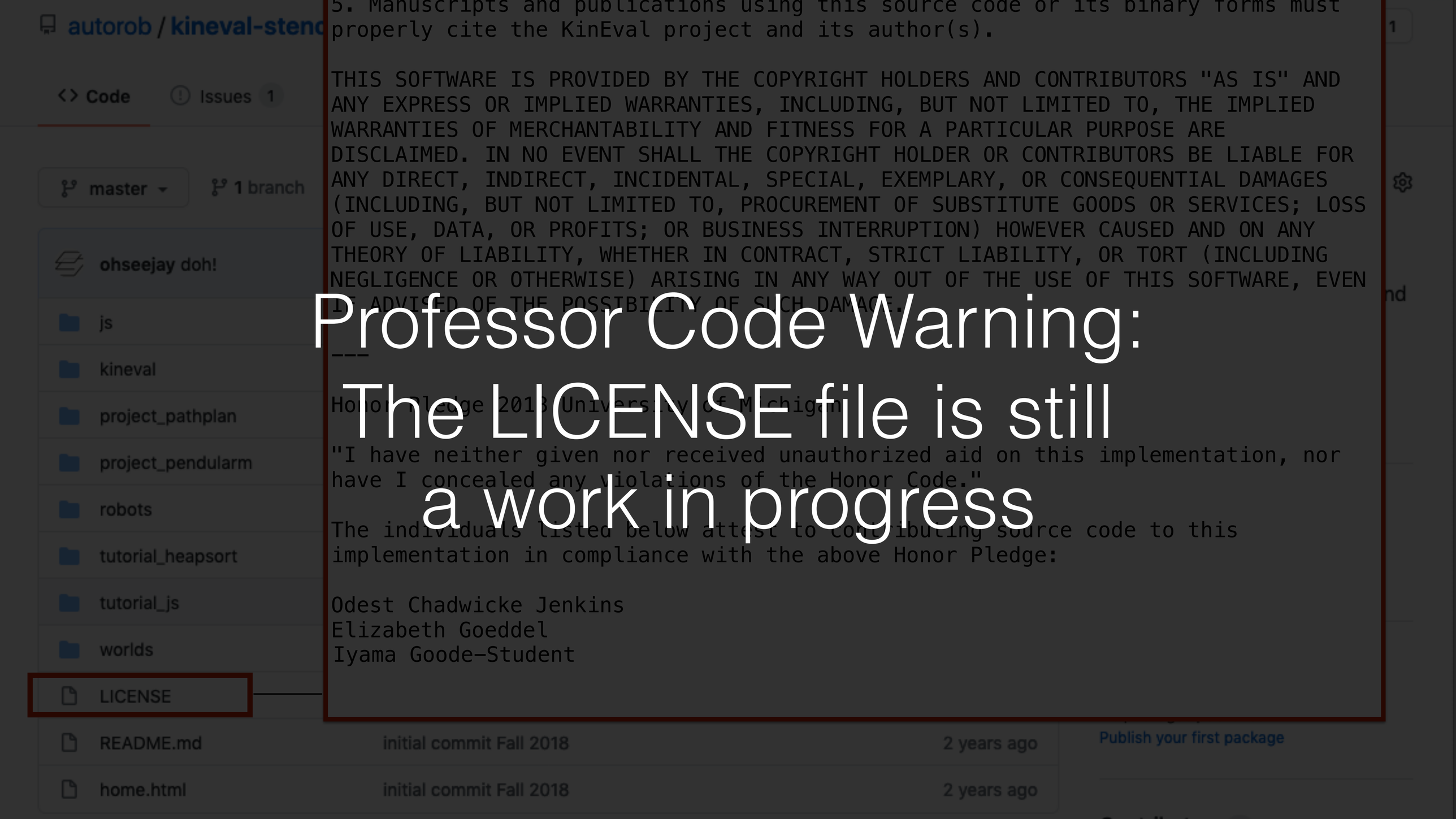
Attribution similar to CC BY 4.0 License
https://creativecommons.org/licenses/by/4.0/

5. Manuscripts and publications using this source code or its binary forms must properly cite the KinEval project and its author(s).

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND

```
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS
OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

Honor Pledge 2018 University of Michigan

"I have neither given nor received unauthorized aid on this implementation, nor
have I concealed any violations of the Honor Code."

The individuals listed below attest to contributing source code to this
implementation in compliance with the above Honor Pledge:

Odest Chadwicke Jenkins
Elizabeth Mamantov Goeddel
Xiaotong Chen
Zheming Zhou
Anthony Opipari
[STUDENT: add your name here]
```

```
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS
OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

Honor Pledge 2018 University of Michigan

"I have neither given nor received unauthorized aid on this implementation, nor
have I concealed any violations of the Honor Code."

The individuals listed below attest to contributing source code to this
implementation in compliance with the above Honor Pledge:

Odest Chadwicke Jenkins
Elizabeth Mamantov Goeddel
Xiaotong Chen
Zheming Zhou
Anthony Opipari
Iyama Goode-Student
```



Professor Code Warning: The LICENSE file is still a work in progress

Wrap up

- This week's lab
  - Walkthrough of assignment 1 stencil
- Next week's lectures
  - MLK Day (no course meeting)
  - Dynamical Simulation
- Assignment 1 (Path Planning) is due January 21, 11:59pm