

JavaScript: Functions, Methods & Objects

CISC-2350-R01 | Fall 2017 | Week 12-2

Ruta Kruliauskaite

Today's Agenda

- Attendance
- Web review presentations (Sean, Noelle)
- Homework Show & Tell
- Final Assignment & Timeline
- JavaScript Functions
- JavaScript Objects, Methods & Events
- Homework Assignment

Web Review Presentations

Homework show & tell

<http://bit.ly/2xzLHvt>

The rest of the semester

Remaining classes

Week 13, 11/20 (Monday)

- Final project concepts presentations
- Conditional statements (If, else, else if)
- For & while loops

Week 13, 11/27 (Thursday)

- No class, Thanksgiving

Week 14, 11/27 (Monday)

- Final project in-class workshop

Week 14, 11/30 (Thursday):

- Time
- The Canvas
- Multiple JavaScript files & more examples

Week 15, 12/4 (Monday)

- Overview of different JavaScript libraries: p5js, express, etc.
- Design process & looking ahead

Week 15, 12/7 (Thursday)

- FINAL PROJECT PRESENTATIONS

Final Project Assignment

Final Project Assignment (due December 7)

Task: to create a personal website for your creative work.

**Final projects will be presented in class on Thursday,
December 7th.**

Final project is worth 30% of the final grade in the course.

Final Project Assignment (due December 7)

You can make a website about your creative project or a hobby that you really love such as cooking or traveling.

The subject matter of your site is open ended but should reflect your academic interests, your creative practice or some other interest of yours.

Some examples can include: an interactive resume, a project gallery site, an interactive media representation of your creative / academic work, a single page personal statement. It's up to you.

The form of your site should emerge from the content and it must be thoughtful and well designed. This means creating something more than a straightforward portfolio page layout.

Think of it as something that you'd show to potential employers or a gallery interested in your work to show off your work and your personality.

Final Project Assignment (due December 7)

The project must:

- Have a well-thought through concept and appropriate form to implement it
- Have a well-thought out and cohesive visual design, using color, external fonts, icons and images
- Have a responsive design (using Bootstrap is highly recommended) and it should be well implemented
- Utilize extensive media elements (up to you how many and which ones)
- Contain a degree of button/link transitions and CSS animation
- Have interactivity using JavaScript (invite users to click something that changes design elements, etc.)

Note: If you did this assignment for the midterm you must meet with me to discuss how you will take yours to the next level technically and conceptually

Final Project Assignment (due December 7)

Technical requirements:

- Original content - well thought through, proofread, etc.
- HTML structure with CSS styling
- CSS - use of classes and IDs, as well as pseudo classes such as :hover and :active
- CSS Fonts, Styles and Colors
- CSS Positioning and Layout (responsive)
- JavaScript interactivity
- Visual Design - I want you to thoughtfully use color, iconography and type, minimal will not work for the final
- Structure - folder structure, external stylesheets, clean code with comments

Note: I am not putting a page minimum on the site, but it should be a complete site without dead links. Scale it appropriately for what you can achieve. You will be graded on the complexity of the project (i.e. if it's only one page, it needs to be complex, some kind of infinite scroll sites, etc.).

Final Project Assignment (due December 7)

References and inspiration:

- [Robby Leonardi Interactive Resume](#)
- [Making of Robby Leonardi's Resume](#)
- [CSS Awards](#)
- [Awwwards Creative Web Portfolios](#)
- [Semplice Labs showcase](#)
- [Siteinspire Portfolio Gallery](#)
- [Raf Simons](#)
- [HAWRAF](#)
- Dribbble and Designspiration for design inspiration

Note: a lot of these websites use advanced Javascript or other advanced techniques, so look at them for visual and interaction reference. Talk to me to make sure the scale of your project is appropriate and achievable.

Final Project Assignment (due December 7)

Project structure:

1. Concept
2. Content preparation
3. Wireframe how you want to make it
4. Design
5. Code
6. Test
7. Iterations
8. Final presentation

Final Project Assignment Timeline

Monday, November 20th:

- Concepts ready to present
- In-class 1 min presentations (1 slide ready)
- Office hours after class available

Thursday, November 23rd:

- NO CLASS, Thanksgiving

Monday, November 27th:

- Content ready
- Wireframes ready
- Visual Design ideas ready: options for use of color, iconography and fonts
- In-class workshop for feedback (3min presentations ready)
- Office hours after class available

Thursday, November 30th:

- First website drafts ready (HTML structure & CSS)
- Office hours after class available if necessary

Monday, December 4th:

- More functional websites ready (improving CSS, adding JavaScript functionality)
- Office hours after class available

Thursday, December 7th:

- Final in-class presentations
- 5min presentations ready
- Double class w/ guest critics

1. Functions

Functions group a series of statements together to perform a specific task.

Terms you should know

1. When you ask function to perform its task, you **call** a function
2. Some functions need to be provided with information in order to achieve a given task - pieces of information passed to a function are called **parameters**
3. When you write a function and expect it to provide you with an answer, the response is known as **return value**

Declaring a function

1. To create a function you give it a name and write statements inside to achieve a task you want it to achieve

The diagram illustrates the components of a function declaration. It shows the code: `function buttonClicked() { console.log("hello"); }`. Labels with leader lines identify the parts: 'function keyword' points to 'function', 'function name' points to 'buttonClicked()', 'code statement' points to the line 'console.log("hello");', and 'code block inside curly braces' points to the entire block between the opening and closing curly braces.

```
function keyword      function name
      |               |
      |               |
function buttonClicked() {
code statement  console.log("hello");
      |
      |
code block inside curly braces
```

Calling a function

1. You call a function by writing its name `buttonClicked()`;

```
1 function buttonClicked() {  
3   console.log("hello");  
  }  
2 buttonClicked();  
4 //code after
```

Declaring functions with parameters

1. Sometimes a function need specific information to perform it's task (parameters)
2. Inside the function the parameters act as variables

```
function countTotal(itemNumber, price) {  
    return itemNumber * price;  
}
```

parameters

|

parameters are used like variables inside the function

Calling functions with parameters

1. When you call a function that has parameters, you need to specify the values it should take in
2. Those values are called **arguments**
3. Arguments are written inside parentheses when you call a function
4. Arguments can be provided as values or as variables

```
function countTotal(itemNumber, price) {  
  return itemNumber * price;  
}
```

```
countTotal(7,15); //will return 105
```

```
itemNumber = 7;  
price = 15;
```

```
countTotal(itemNumber, price); // will return 105
```

Using “return” in a function

1. **return** is used to return a value to the code that called the function
2. Interpreter leaves the function when return is used and goes back to the statement that called it

```
function countTotal(itemNumber, price) {  
    return itemNumber * price;  
}  
console.log("hello");
```

Variable scope

1. Where you declare a variable affects where it can be used within your code
2. If it's declared inside a function, it can only be used inside that function
3. It's known as variable's **scope**

Local variables

1. When a variable is created *inside* a function
2. It's called **local** variable or **function-level** variable
3. It has **local scope** or **function-level scope**
4. It cannot be accessed outside the function

```
function countTotal(itemNumber, price) {  
    var total = itemNumber * price;  
    return total;  
}
```

Global variables

1. When a variable is created *outside* a function
2. It can be used anywhere in the script
3. It's called **global** variable
4. It has **global scope**
5. They take up more memory than local variables

```
function countTotal(itemNumber, price) {  
    var total = itemNumber * price;  
    return total;  
}
```

```
var totalPrice = countTotal(7, 15); // global variable
```

2. Objects

Objects group together a set of variables and functions to create a model of something you would recognise from the real world.

Properties & methods

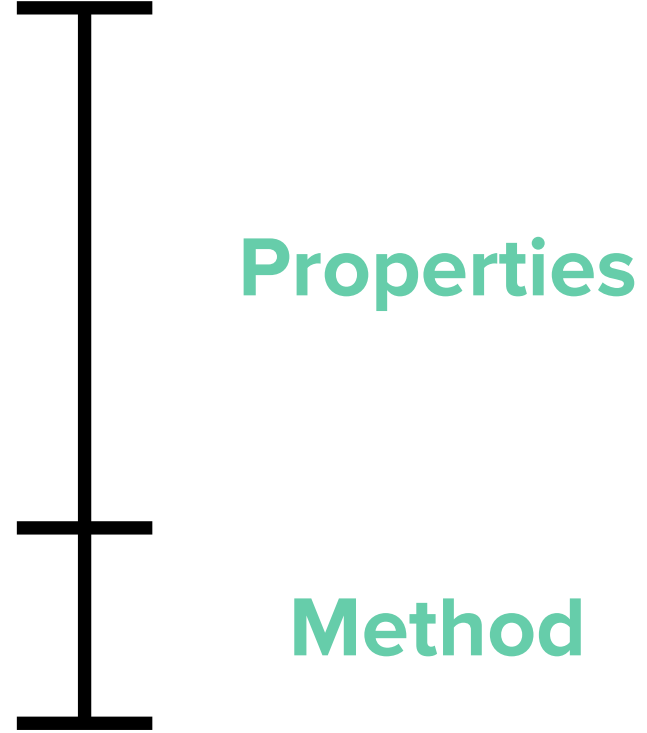
1. In an object:
 1. Variables become known as properties
 2. Functions become known as methods
2. **Properties** tell us about the object, such as the height of a chair and how many chairs there are
3. Methods represent tasks that are associated with the object, e.g. count the height of all chairs by adding all heights together

Properties & methods

1. In an object:
 1. Variables become known as properties
 2. Functions become known as methods
2. **Properties** tell us about the object, such as the height of a chair and how many chairs there are
3. Methods represent tasks that are associated with the object, e.g. count the height of all chairs by adding all heights together

Properties & objects

```
var hotel = {  
  name: "Radisson",  
  rooms: 55,  
  booked: 15,  
  spa: false  
  roomTypes: ["single", "twin", "suite"],  
  checkAvailability: function() {  
    return this.rooms - this.booked;  
  }  
}
```



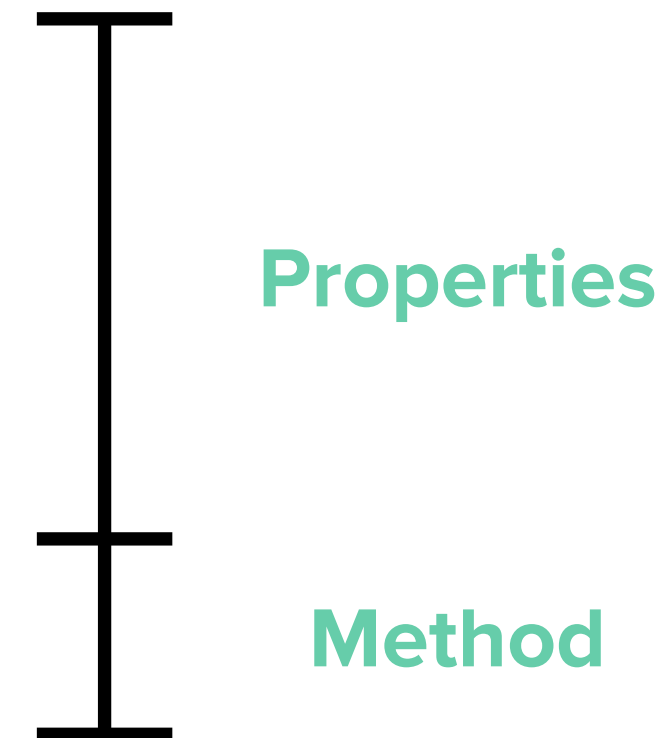
Properties

Method

1. name, rooms, booked - are **keys** in the object
2. Object cannot have two keys with the same name
3. Value can be anything you want (string, number, boolean, function, another object)

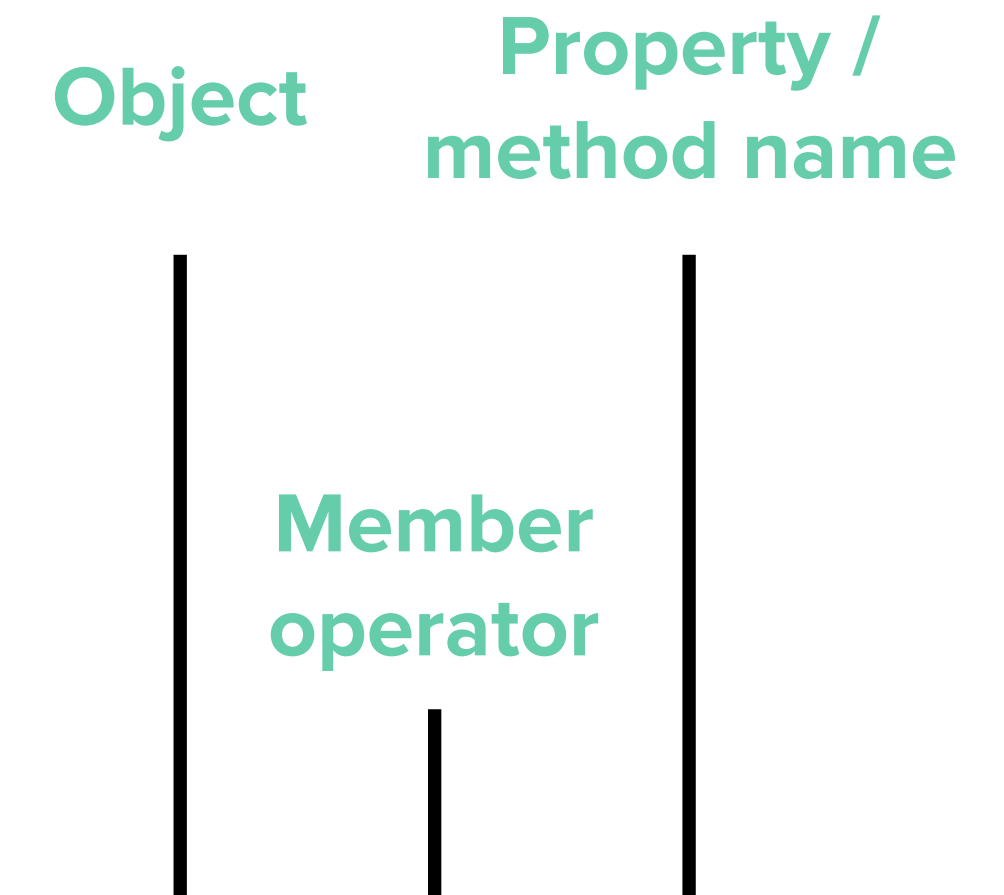
Creating an object: literal notation

```
var hotel = {  
  name: "Radisson",  
  rooms: 55,  
  booked: 15,  
  spa: false  
  roomTypes: ["single", "twin", "suite"],  
  checkAvailability: function() {  
    return this.rooms - this.booked;  
  }  
}
```



1. Object is stored in a variable called **hotel**
2. Each key is separated from its value by a colon :
3. Each property and method is separated by a comma ,
4. **this** keyword indicates that “rooms” and “booked” properties should be taken from **this object**

Accessing an object



1. There are two ways to access properties in an object

1. Using dot notation
2. Using square brackets

```
var hotel = {  
  name: "Radisson",  
  rooms: 55,  
  booked: 15,  
  spa: false  
  roomTypes: ["single", "twin", "suite"],  
  checkAvailability: function() {  
    return this.rooms - this.booked;  
  }  
}
```

dot notation

```
var hotelName = hotel.name;  
var roomsAvailable =  
  hotel.checkAvailability();
```

square brackets

```
var hotelName =  
  hotel["name"];  
var roomsAvailable =  
  hotel["checkAvailability"]  
  ();
```

Creating an object: constructor notation

```
var hotel = new Object();  
  hotel.name = "Radisson";  
  hotel.rooms = 55;  
  hotel.booked = 15;  
  hotel.spa = false;  
  hotel.roomTypes = ["single", "twin", "suite"];  
  hotel.checkAvailability = function() {  
    return this.rooms - this.booked;  
  }
```



Properties

Method

1. The **new** keyword creates a blank object
2. Then you can add properties and method using dot notation

Updating an object

```
var hotel = {  
  name: "Radisson",  
  rooms: 55,  
  booked: 15,  
  spa: false  
  roomTypes: ["single", "twin",  
    "suite"],  
  checkAvailability: function() {  
    return this.rooms -  
      this.booked;  
  }  
}
```

dot notation

```
hotel.name = "Hilton";
```

square brackets

```
hotel["name"] = "Hilton";
```

Deleting property from an object

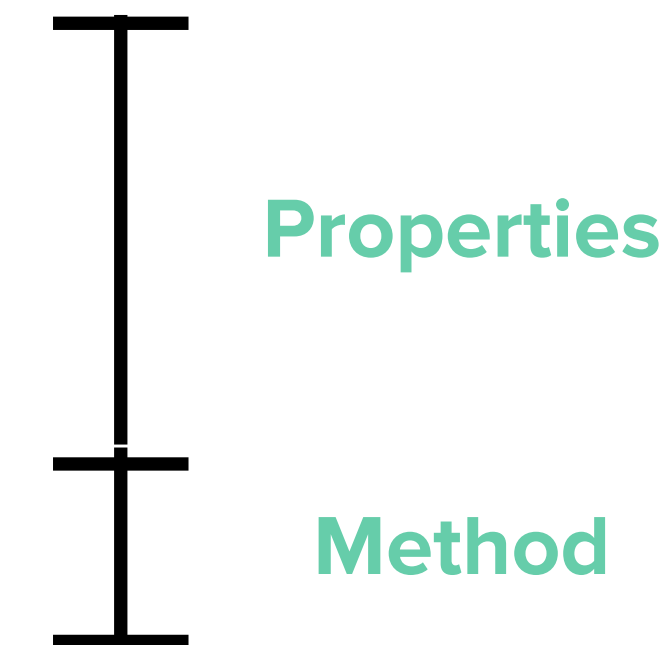
```
var hotel = {  
  name: "Radisson",  
  rooms: 55,  
  booked: 15,  
  spa: false  
  roomTypes: ["single", "twin",  
    "suite"],  
  checkAvailability: function() {  
    return this.rooms -  
      this.booked;  
  }  
}
```

delete hotel.name;

Creating many objects: constructor notation

1. Sometimes you will want several objects to represent similar thing
2. You can use function as a template to create several objects
3. First we need to create template with object's properties and methods

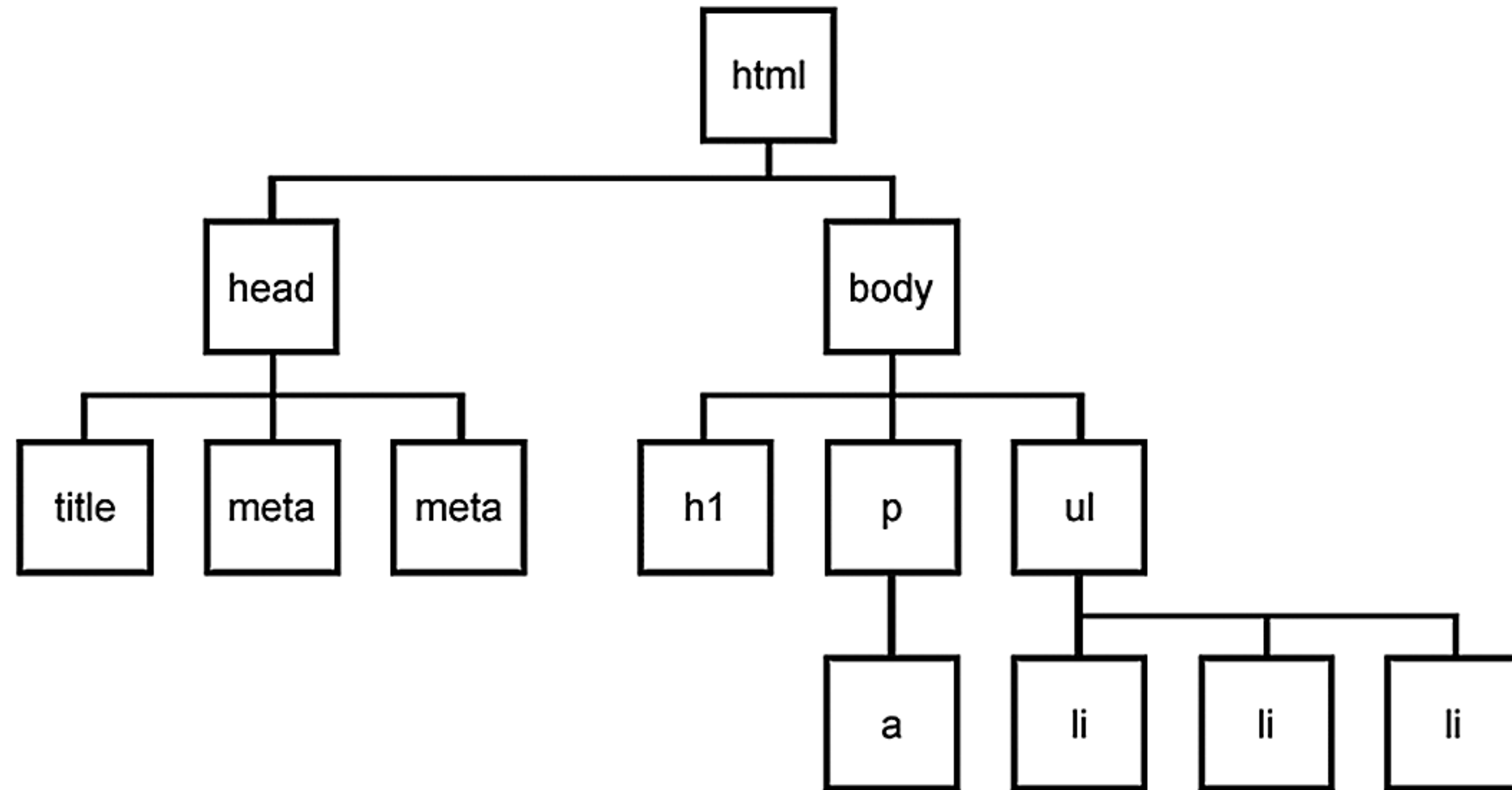
```
function Hotel(name, rooms, booked) {  
  this.name = name;  
  this.rooms = rooms;  
  this.booked = booked;  
  this.checkAvailability = function() {  
    return this.rooms - this.booked;  
  }  
}
```



```
var radissonHotel = new Hotel("Radisson", 55, 15);  
var hiltonHotel = new Hotel("Hilton", 60, 60);
```

3. Built-in DOM events

When an HTML document is loaded into a web browser, it becomes a document object.



The document object provides properties and methods to access all node objects, from within JavaScript.

Built-in DOM events

Mouse Events

Event	Description	DOM
<u>onclick</u>	The event occurs when the user clicks on an element	2
<u>oncontextmenu</u>	The event occurs when the user right-clicks on an element to open a context menu	3
<u>ondblclick</u>	The event occurs when the user double-clicks on an element	2
<u>onmousedown</u>	The event occurs when the user presses a mouse button over an element	2
<u>onmouseenter</u>	The event occurs when the pointer is moved onto an element	2
<u>onmouseleave</u>	The event occurs when the pointer is moved out of an element	2
<u>onmousemove</u>	The event occurs when the pointer is moving while it is over an element	2
<u>onmouseover</u>	The event occurs when the pointer is moved onto an element, or onto one of its children	2
<u>onmouseout</u>	The event occurs when a user moves the mouse pointer out of an element, or out of one of its children	2
<u>onmouseup</u>	The event occurs when a user releases a mouse button over an element	2

Built-in DOM events

Keyboard Events

Event	Description	DOM
<u>onkeydown</u>	The event occurs when the user is pressing a key	2
<u>onkeypress</u>	The event occurs when the user presses a key	2
<u>onkeyup</u>	The event occurs when the user releases a key	2

Built-in DOM events

Media Events

Event	Description	DOM
<u>onabort</u>	The event occurs when the loading of a media is aborted	3
<u>oncanplay</u>	The event occurs when the browser can start playing the media (when it has buffered enough to begin)	3
<u>oncanplaythrough</u>	The event occurs when the browser can play through the media without stopping for buffering	3
<u>ondurationchange</u>	The event occurs when the duration of the media is changed	3
<u>onemptied</u>	The event occurs when something bad happens and the media file is suddenly unavailable (like unexpectedly disconnects)	3
<u>onended</u>	The event occurs when the media has reach the end (useful for messages like "thanks for listening")	3
<u>onerror</u>	The event occurs when an error occurred during the loading of a media file	3
<u>onloadeddata</u>	The event occurs when media data is loaded	3
<u>onloadedmetadata</u>	The event occurs when meta data (like dimensions and duration) are loaded	3
<u>onloadstart</u>	The event occurs when the browser starts looking for the specified media	3
<u>onpause</u>	The event occurs when the media is paused either by the user or programmatically	3
<u>onplay</u>	The event occurs when the media has been started or is no longer paused	3
<u>onplaying</u>	The event occurs when the media is playing after having been paused or stopped for buffering	3

Built-in DOM events documentation

1. Full list with examples:
 1. https://www.w3schools.com/jsref/dom_obj_event.asp
 2. <https://developer.mozilla.org/en-US/docs/Web/Events>

Built-in DOM properties and methods

1. HTML DOM **methods** are *actions* you can perform (on HTML Elements).
2. HTML DOM **properties** are *values* (of HTML Elements) that you can set or change.

```
document.getElementById("hello").innerHTML = "Hello!";
```

getElementById -> method

innerHTML -> property

Built-in DOM properties and methods

Finding HTML Elements

Method	Description
<code>document.getElementById(<i>id</i>)</code>	Find an element by element id
<code>document.getElementsByTagName(<i>name</i>)</code>	Find elements by tag name
<code>document.getElementsByClassName(<i>name</i>)</code>	Find elements by class name

Built-in DOM properties and methods

Changing HTML Elements

Method	Description
<code>element.innerHTML = new html content</code>	Change the inner HTML of an element
<code>element.attribute = new value</code>	Change the attribute value of an HTML element
<code>element.setAttribute(attribute, value)</code>	Change the attribute value of an HTML element
<code>element.style.property = new style</code>	Change the style of an HTML element

Built-in DOM properties and methods

Adding and Deleting Elements

Method	Description
<code>document.createElement(<i>element</i>)</code>	Create an HTML element
<code>document.removeChild(<i>element</i>)</code>	Remove an HTML element
<code>document.appendChild(<i>element</i>)</code>	Add an HTML element
<code>document.replaceChild(<i>element</i>)</code>	Replace an HTML element
<code>document.write(<i>text</i>)</code>	Write into the HTML output stream

Built-in DOM properties and methods documentation

1. Full list with examples:
 1. https://www.w3schools.com/js/js_htmlDOM_document.asp
 2. https://www.w3schools.com/js/js_htmlDOM.asp

Homework assignment

Homework

By Sunday, November 19, 6pm

1. Web review presentations: Sayed, Md

1. Post website links you'll be presenting to Slack by Sunday 6pm

2. Review the class slides

3. Thoroughly read documentation on DOM events, properties and methods:

1. https://www.w3schools.com/jsref/dom_obj_event.asp
2. <https://developer.mozilla.org/en-US/docs/Web/Events>
3. https://www.w3schools.com/js/js_htmlDOM_document.asp
4. https://www.w3schools.com/js/js_htmlDOM.asp

4. Choose 2-3 built-in events, properties, methods and experiment with them by adding them to your website (can be either one of the past homework or something new):

- Should be at least **two** built-in events
- Also proper use of methods and properties
- Make sure to clearly comment your code and process, so it's easy to follow
- Use variables and other things we've learned so far in class
- Make sure you have proper file structure
- The website should be responsive (using Bootstrap or other framework we used)
- Upload your code to a new folder on Github and post two links on #general channel on Slack:
 - Link to your project's repository (folder)
 - Working link to the Github page (yourusername.github.io/pathstofolder)

5. Final Project assignment:

1. Come up with the concept for your project
2. Prepare 1min presentation to talk about it in class, you should have 1 slide while you talk
 1. Email me with 1 slide by Sunday 6pm, so I can put all of the slides into one presentation and you can present from your seat