

## ECS 89

6/2

## Announcements

- Assignment due tomorrow evening – submit using SmartSite
- Final in this room, Wds June 11, 8AM
- Course evaluations - <https://eval.ucdavis.edu>
  
- Today:
  - Asynchronous loading of data
  - Ajax

## Dynamic Web pages

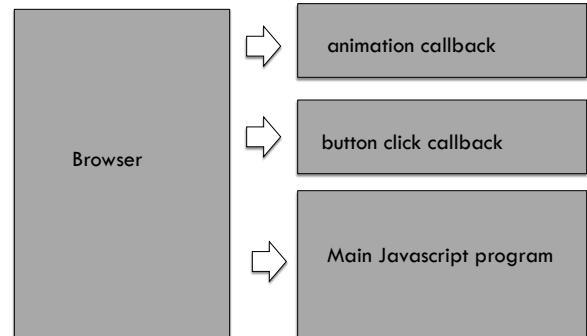
- Javascript changes Web pages in the browser, no server involvement
- But what if you want new information from the server?
- Can always get a new Web page
- But a complicated Web page can take forever to load
- Why does it take forever to load?

## Asynchronous loading

- Can't use data until it is loaded
- Could just wait, but usually user does not want browser locked up while waiting
- When many things need to load, get them all working at once
- This is the default for video and sound clips, very common for images
- usually a good idea when you go out to the server, since you never know how long that will take (or if you will ever get a response!)

## Events and callbacks

- We define functions to be run when events occur, eg. onclick, for every animation frame...
- These are **callback functions**
- The browser is always running. When an event for which we have a callback registered occurs, it stops whatever it is doing and calls the function
- Asynchronous loading of data uses events and callbacks
- Start the load, and then an event occurs when the data is (fully or partially) loaded



## Asynchronously loading an image

```
var arblmg = document.createElement("img");
arblmg.src = "arboretum.jpg";
arblmg.addEventListener("load", drawImage);
```

- Images might take a while to load
- Instead of waiting, define a callback function
- “load” is the event that happens when the image finishes loading
- Here function drawImage will be called when the “load” event occurs

## HTMLMediaObject

- <audio> and <video> HTML5 elements handle asynchronous loading of audio and video
- Use load command to load, then play when the “loadeddata” event occurs.
- This isn’t when the whole thing is loaded, just when the browser figures enough has been loaded to play the whole thing without having to stop and wait.

## Ajax

- Asynchronous loading of arbitrary text data
- “Asynchronous Javascript And XML”
- XML (X markup language) is a data format, like csv; similar to HTML in that all data is labeled with tags.
- XML possibly being supplanted by JSON (Javascript Object Notation), which looks a little like a Python dictionary
- Ajax can actually be used with any text data format, XML is optional

## XMLHttpRequest object

- Constructs an HTTP request to get some data
  - Can use GET or POST
  - Need to give URL; URL has to be from the same server as the original Web page
- ```
req.open("GET", "ajax_info.txt", true);
```
- “true” means asynchronous request. If you use “false”, the javascript program stops at this command until the load is complete

## Using an XMLHttpRequest object

```
var req = new XMLHttpRequest();
req.onreadystatechange = changeTheText; // callback
req.open("GET","ajax_info.txt",true); // header
req.send();
```

- “onreadystatechange” is the event
- Called whenever status of the XMLHttpRequest object is changed

## (side note – scope!)

- In the example code, the XMLHttpRequest state change callback is defined inside the onclick callback
- This is so that it can share the XMLHttpRequest object
- Technically, it is a method of the function object loadXMLDoc; no problem with this in Javascript

## More typical Ajax application

---

- Getting data from server database
- Rather than get a whole Web page that shows the new data, just get the data itself and let Javascript display it
- I'll use the database response we already have as an example...
- This is producing HTML that is getting stuck into the page, but it could be producing data in any format, that the Javascript program uses however it wants