Week 5

DEALING WITH DATA

DATA FORMAT

JSON

= JavaScript Object Notation

```
{
    "example": "this is some JSON!",
    "arrayExample": [
        {
            "note": "property name must be in quotes",
            "and": "no functions",
            "nuvar test mberExample": 123,
            "booleanExample": true
        }
    ]
}
```

JSON

Difference between JSON and an actual JavaScript Object: JSON is plain text.

We need to convert between one and the other:

```
var obj = { test: 123 },
    objToJson = JSON.stringify(obj) // "{ "test": 123 }"

var jsonString = '{ "test": 234 }',
    jsonToObj = JSON.parse(jsonString) // Object {test: 234}
```

JSON

Why?

Because http protocol doesn't understand JavaScript Objects. With JSON we can send the data across the internet as plain text.

It's also because JavaScript Objects are easy to work with in code. (XML is horrible)

JSON API

= URLs that give you JSON instead of web pages.

```
// a hypothetical example
http://api.example.com/posts


// would give you:
{
    "posts": [
        { "title": "...", "content": "..." },
        { "title": "...", "content": "..." },
        ...
    ]
}
```

HOW TO GET DATA

HTTP

HTTP is a protocol: HyperText Transfer Protocol

You send something to the server in a specific format, and expect to get something back in the same format.

An HTTP request is simply plain text which looks like this:

```
GET /index.html HTTP/1.1
Host: www.example.com
```

The response could look like this:

```
HTTP/1.1 200 OK
 Date: Mon, 23 May 2005 22:38:34 GMT
 Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
 Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
 Etag: "3f80f-1b6-3e1cb03b"
 Accept-Ranges:  none
 Content-Length: 438
 Connection: close
 Content-Type: text/html; charset=UTF-8
```

AJAX

Basically it means getting some data from an API, then update some DOM elements without refreshing the page.

Why?

Data only = faster transport across the wire

No refresh = fewer requests, no need to re-render the page, no need to re-run the JavaScript => snappier interaction => happier user

AJAX

Of course you don't need to manually write and decode all that in JavaScript. There's something called `XMLHttpRequest.`

```
var xhr = new XMLHttpRequest()

// the last true argument means we want this request
// to be asynchronous
xhr.open("GET", 'http://some.api.here?apiKey=your-key', true)

xhr.onload = function () {
    // do something with xhr.responseText
}

xhr.send() // this is the moment the request is actually sent
```

AJAX

That was still not that intuitive to use.
The easy way is to do it with jQuery...

```
$.ajax({
    url: 'http://api.tumblr.com/v2/blog/something.tumblr.com/posts',
    data: {
        'apiKey': 'your-api-key'
    },
    complete: function (response) {
        // do something with the response
    }
})
```

WHAT DO I DO WITH THE DATA?

A common case is to loop through things and build DOM elements:

```
// suppose data is what you got back from the API
data.posts.forEach(buildPost)

function buildPost (postData) {
    var node = document.createElement('div')
    node.innerHTML =
        '<h2>' + postData.title + '</h2>'
        + '<p>' + postData.content + '</p>'
    return node
}
```