

Engineering Robust Server Software

AJAX

Wrap up Intro to Server-side Web: AJAX

- Just did intro to Django
 - Everything so far: page reload to communicate with server
- Real website:
 - Interactive without page reload
 - AJAX: Asynchronous Javascript and XML (*)
 - (*) May not actually involve XML.

AJAX Basics

```
function someJSFun() {  
    //whatever code...  
  
    var xhttp = new XMLHttpRequest();
```

This is the object to contact
the server and get a response...

AJAX Basics

```
function someJSFun() {  
    //whatever code...  
  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        //some other code in here...  
    };  
};
```

Set its
onreadystatechange
to be notified when stuff happens

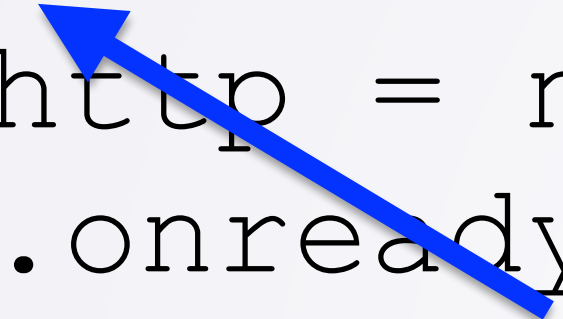
AJAX Basics

```
function someJSFun() {  
    //whatever code...  
  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        //some other code in here...  
    };  
};
```

Yes, you can write one function
inside another.
JavaScript has **lexical scope**.
This makes a **closure**.

AJAX Basics

```
function someJSFun() {  
    //whatever code...  
    var xyz = something;  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        ...xyz...  
    };  
};
```



AJAX Basics

```
function someJSFun() {  
    //whatever code...  
  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        //some other code in here...  
    };  
    xhttp.open("GET", "/api/foo/bar/42", true);  
}
```

.open() specifies where to connect:
HTTP Request Method
URL to request
Asynchronous (usually true)

AJAX Basics

```
function someJSFun() {  
    //whatever code...  
  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        //some other code in here...  
    };  
    xhttp.open("GET", "/api/foo/bar/42", true);  
    xhttp.send();  
}
```

.send() makes the actual request.

**Will make callback to our function
when state changes**

AJAX Basics

```
xhttp.onreadystatechange = function() {
```

```
} ;
```

Now let us look inside our
ready state change callback

AJAX Basics

```
xhttp.onreadystatechange = function() {  
    if (this.readyState == 4
```

```
} ;
```

Typically inspect `this.readyState` first

`this` is our XMLHttpRequest

readyState: 0–4. 4 is Done

AJAX Basics

```
xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {
```

```
    }  
};
```

May also want to inspect

this.status (HTML response status)

200 = OK

AJAX Basics

```
xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
        ...this.responseText...
```

```
    }  
};
```

Once we have our response,
generally want to use

`this.responseText`

which has the text we received

AJAX Basics

```
xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
        var resp = JSON.parse(this.responseText) ;  
  
    }  
};
```

If our response is JSON, can use

JSON.parse to turn into JavaScript object!

Server Side

- On server side:
 - Need to set up URL

```
url(r'^api/foo/(?P<thing>[a-zA-Z]+[0-9]+)/(?P<num>[0-9])', foo.apiView)
```

- And write view

```
def apiView(request, thing, num) :  
    ans = computeAThing(thing, num)  
    return HttpResponse(json.dumps({'foo' : ans}),  
                        content_type="application/json")
```

Server Side

- On server side:
 - Need to set up URL
`url(r'^api/foo/(?P<thing>[a-zA-Z]+[0-9]+)/(?P<num>[0-9])', foo.apiView)`
- And write view

```
def apiView(request, thing, num) :  
    ans = computeAThing(thing, num)  
    return HttpResponse(json.dumps ({ 'foo' : ans } ),  
                        content_type="application/json")
```

Server Side

- On server side:
 - Need to set up URL
`url(r'^api/foo/(?P<thing>[a-zA-Z]+[0-9]+)/(?P<num>[0-9])', foo.apiView)`
- And write view

```
def apiView(request, thing, num) :  
    ans = computeAThing(thing, num)  
    return HttpResponse(json.dumps({'foo' : ans}),  
                        content_type="application/json")
```