Web Development

COMP 431 / COMP 531

Third-Party Services

Scott E Pollack, PhD

April 14, 2016

Summer intern?

Part IIb – Back End Development

Looking for volunteers to present on Tuesday 4/19

- COMP 531 Paper and Presentations 4/21
  - Due Thursday 4/21 before class
- Homework Assignment 8 (Final Full Web App)
  - Due Thursday 4/28

Part IIb

~~Authorization~~

~~Security~~

~~OAuth/2~~

~~Scalability~~

Service APIs

Integrating

Assignment 8: Full Web App

- Finalize your social networking application

- Post query returns posts for user and followed users
- OAuth login option for users: link and unlink accounts Inclass 23
- Redis cache for session Inclass 24
- Permit image uploads and persist in datastore Inclass 25

Third-Party Services: *APIs and SDKs*

- Why roll your own?
  - You know every piece of the puzzle
  - Customized to your site

- Why use a service?
  - Custom software not easily transferrable
  - Not known to new team members
  - Tried and tested
  - You don’t have the expertise or time
  - Continual updates and improvements

Web Analytics

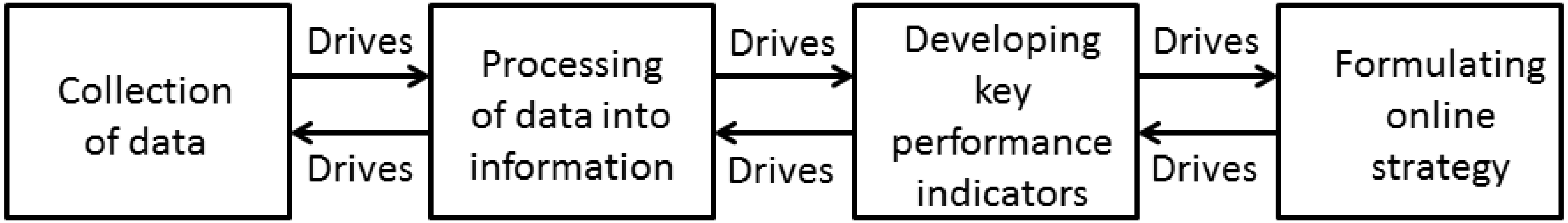
- Collect, analyze, and report web traffic
- Use to enchance and optimize your site
- Also can be used for market research

- Off-site analytics
  - news about your website in the internet
- On-site analytics
  - scrape server logs
  - page tagging with JavaScript

IP address
Request location
Sessions = collection of requests by user
Track cookies

IP address
Count cached page loads
Event tagging, i.e., mouse clicks!
Cookies, sessions, etc

Basic Steps of Web Analytics Process



Collection of data

Drives

Drives

Processing of data into information

Drives

Drives

Developing key performance indicators

Drives

Drives

Formulating online strategy

Typically, counts.

Basically, data collection

Examples:

- Time stamp
- Referral URL
- Query terms

Typically, ratios.

Data becomes metrics.

Examples:

- Time on page
- Bounce rate
- Unique visitors

Counts and ratios infused with business strategy.

Examples:

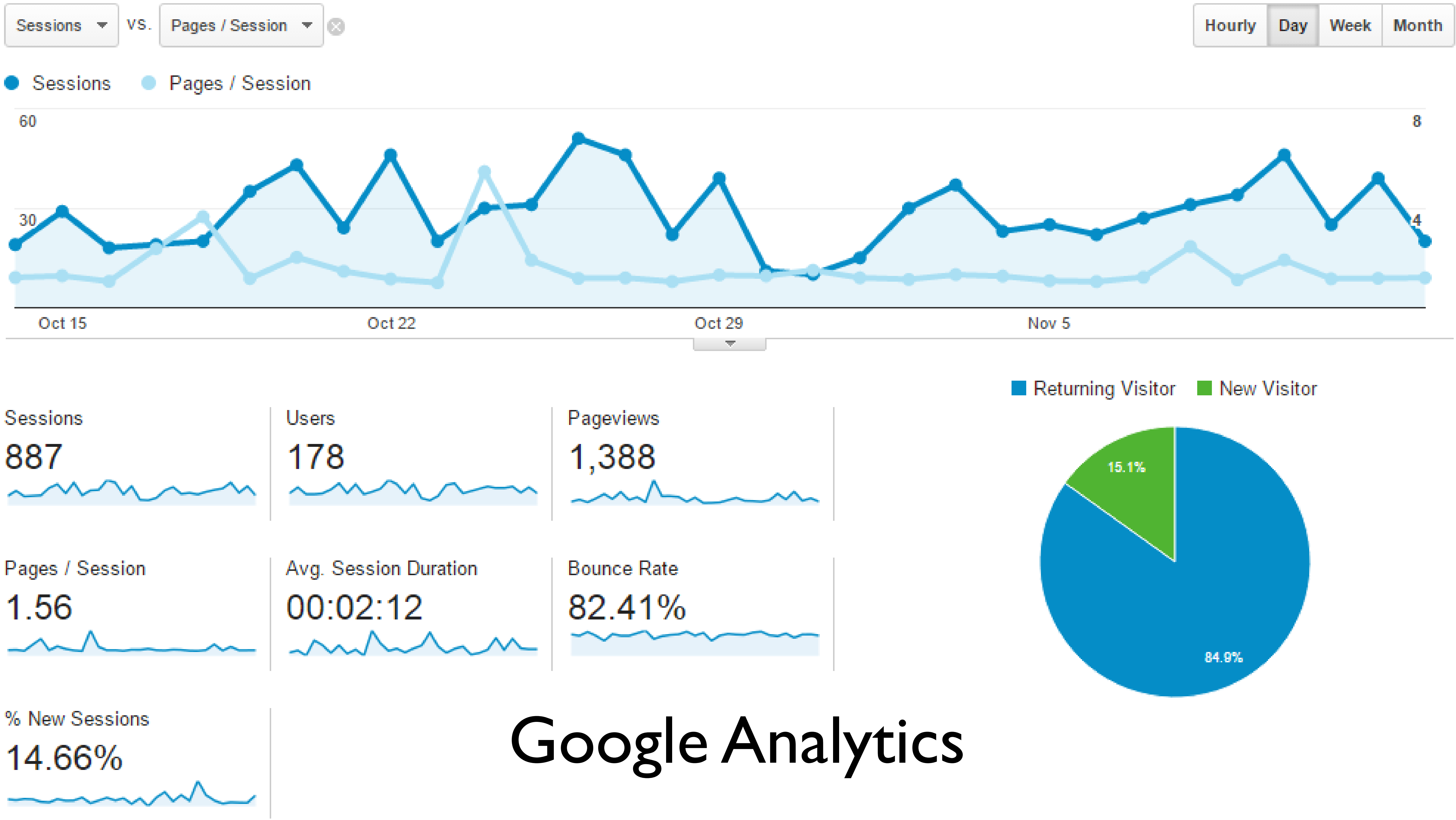
- Conversion rate
- Average order value
- Task completion rate

Online goals, objectives, or standards for organization.

Examples:

- Save money
- Make money
- Marketshare

Google Analytics



Sessions
887

Users
178

Pageviews
1,388

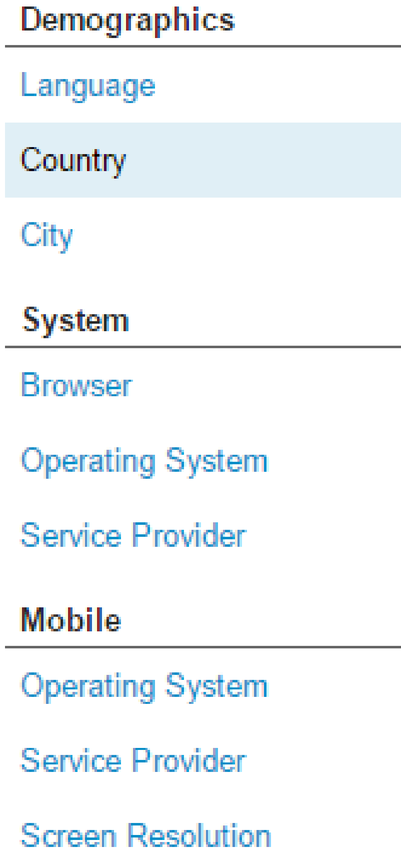
Pages / Session
1.56

Avg. Session Duration
00:02:12

Bounce Rate
82.41%

% New Sessions
14.66%

Google Analytics



Demographics

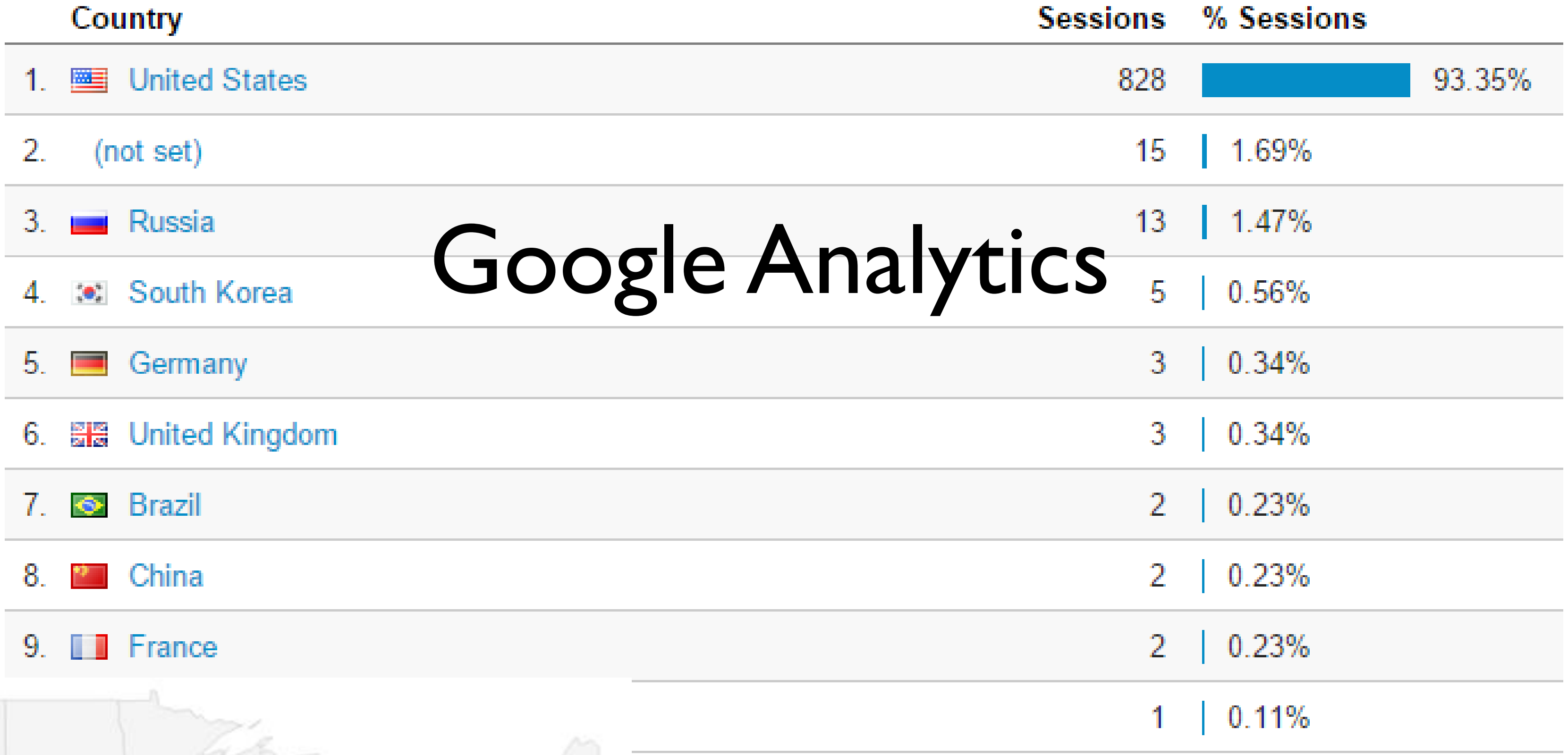
- Language
- Country
- City

System

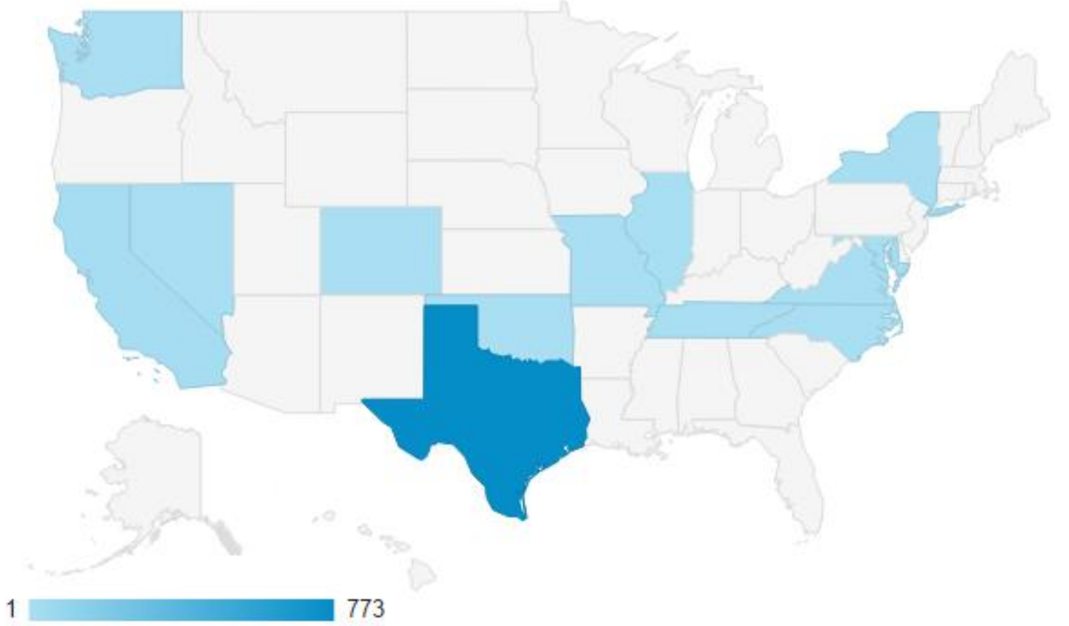
- Browser
- Operating System
- Service Provider

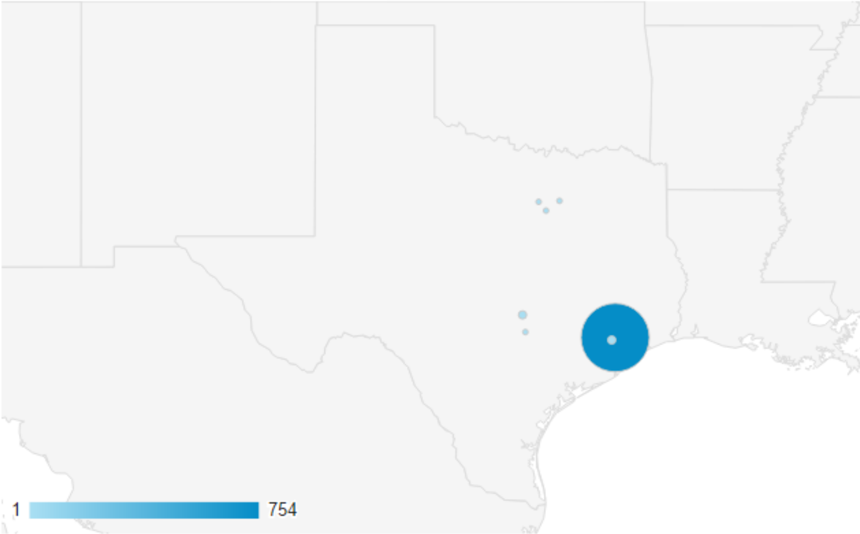
Mobile

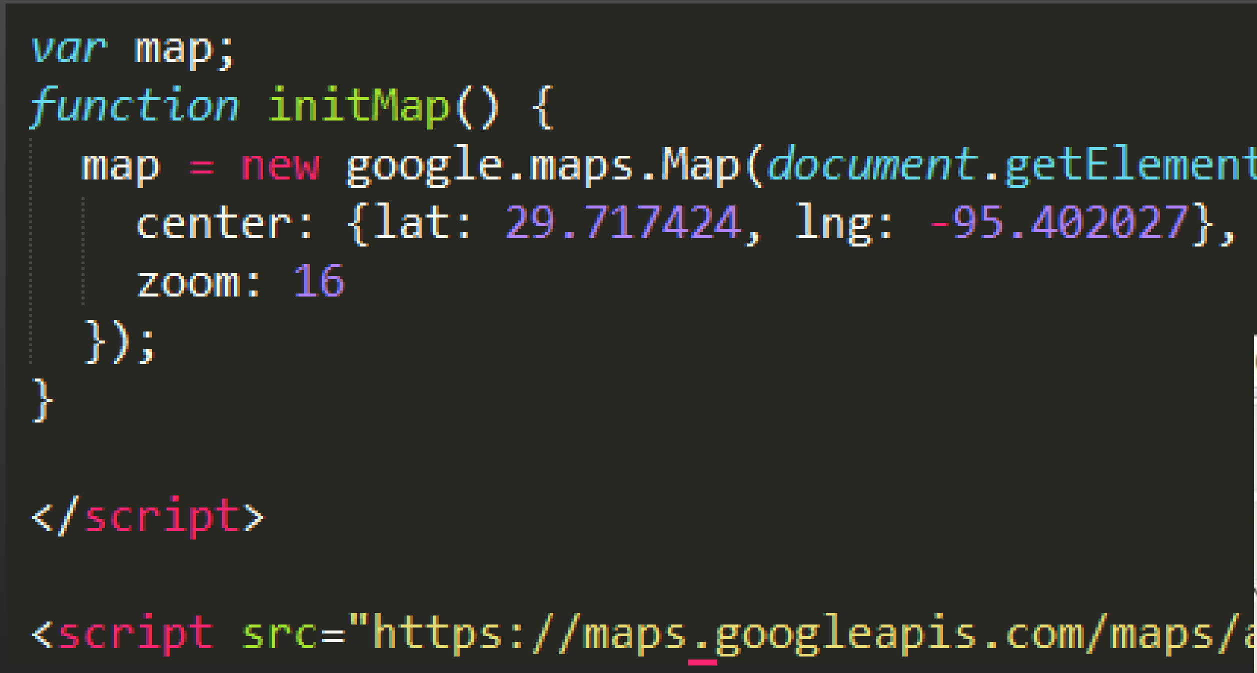
- Operating System
- Service Provider
- Screen Resolution



	Country	Sessions	% Sessions
1.	🇺🇸 United States	828	93.35%
2.	(not set)	15	1.69%
3.	🇷🇺 Russia	13	1.47%
4.	🇰🇷 South Korea	5	0.56%
5.	🇩🇪 Germany	3	0.34%
6.	🇬🇧 United Kingdom	3	0.34%
7.	🇧🇷 Brazil	2	0.23%
8.	🇨🇳 China	2	0.23%
9.	🇫🇷 France	2	0.23%
		1	0.11%

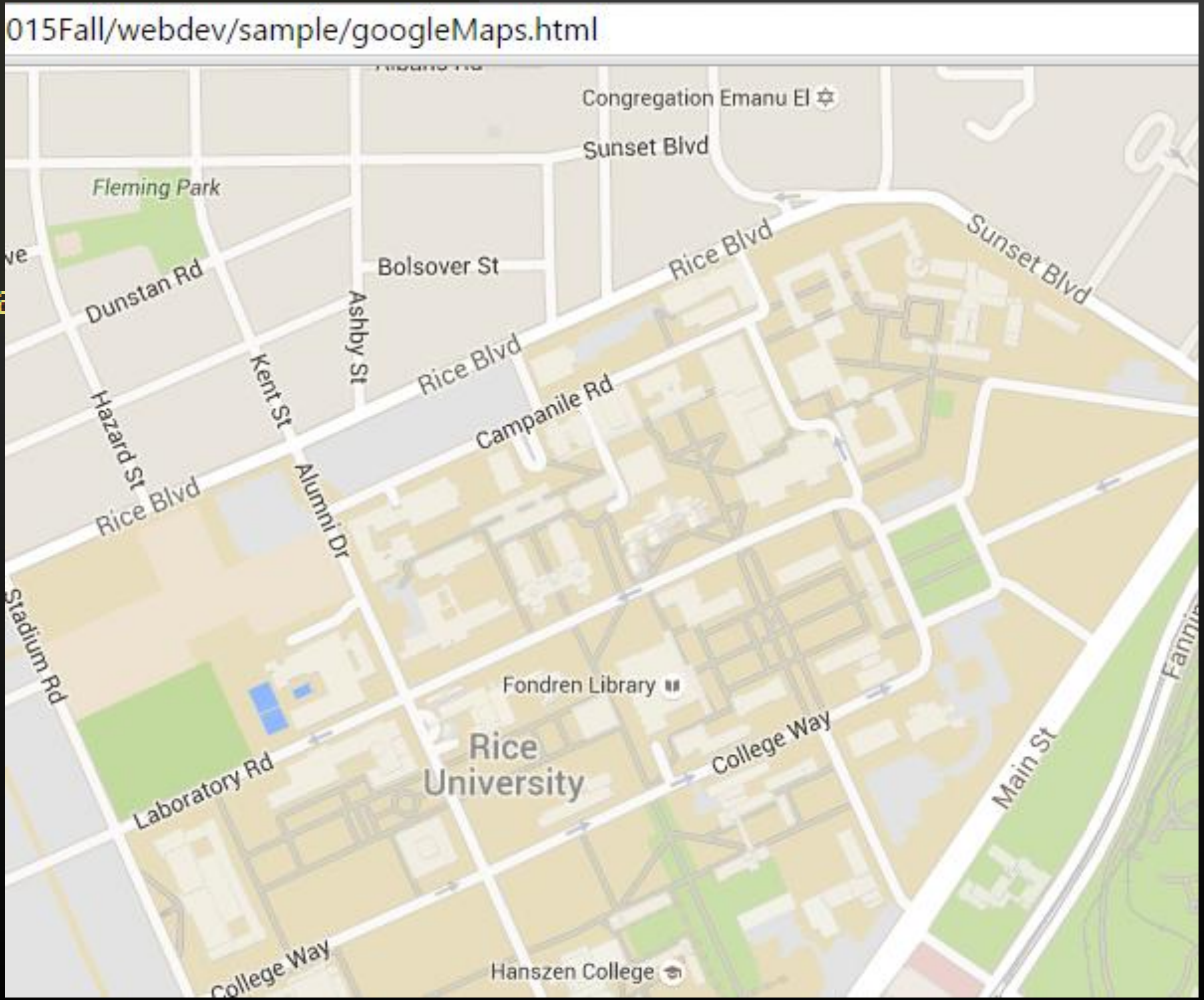






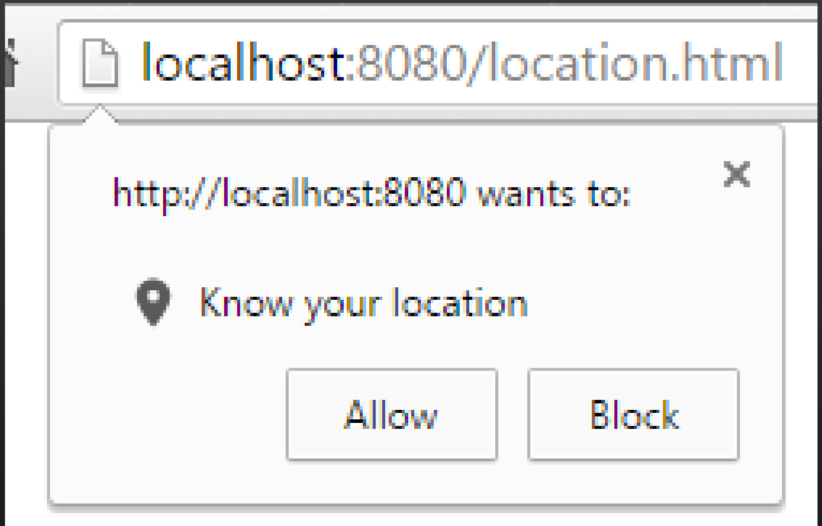
```
var map;
function initMap() {
  map = new google.maps.Map(document.getElementById('map'), {
    center: {lat: 29.717424, lng: -95.402027},
    zoom: 16
  });
}

</script>

<script src="https://maps.googleapis.com/maps/a
```

Google Maps API



015Fall/webdev/sample/googleMaps.html

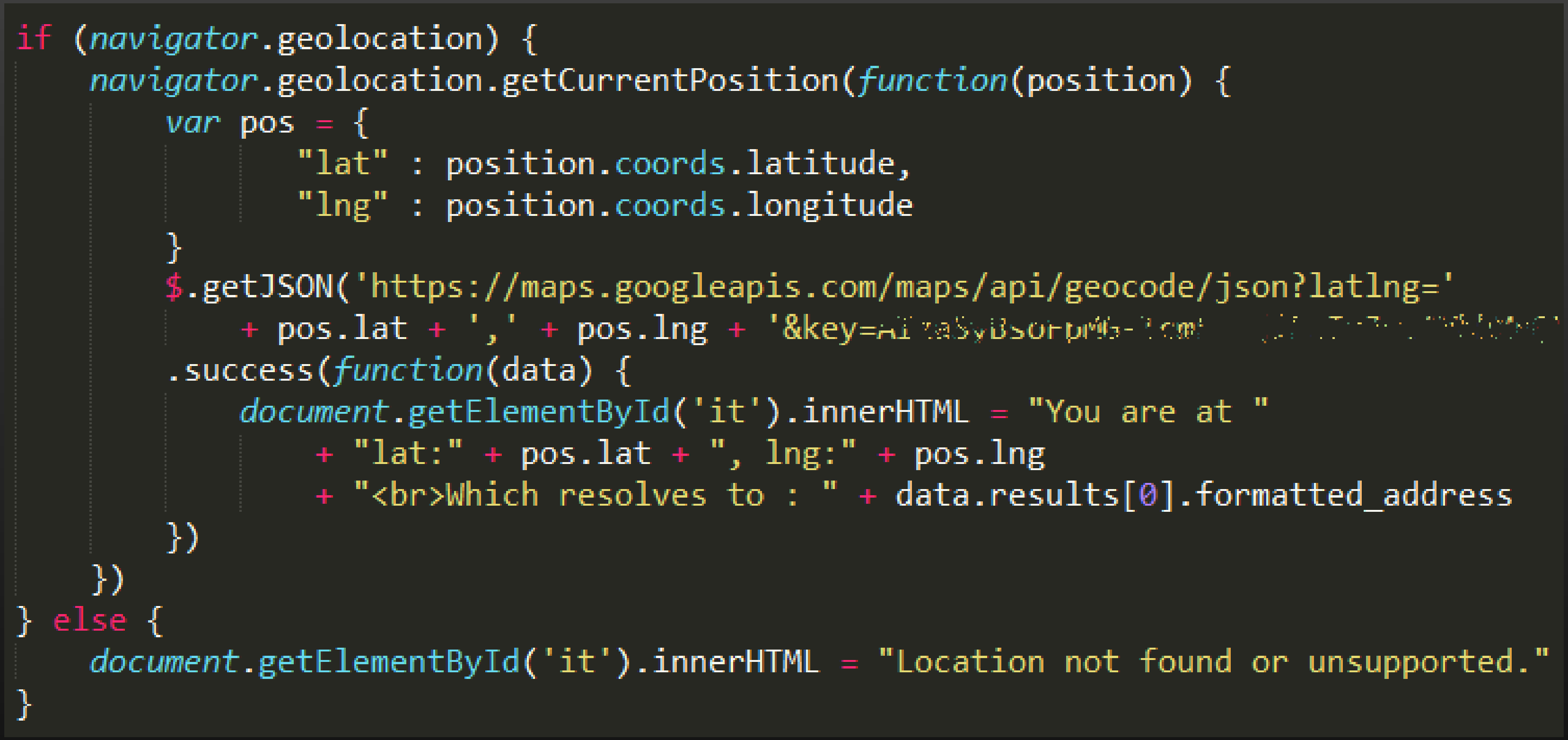
HTML5 Location



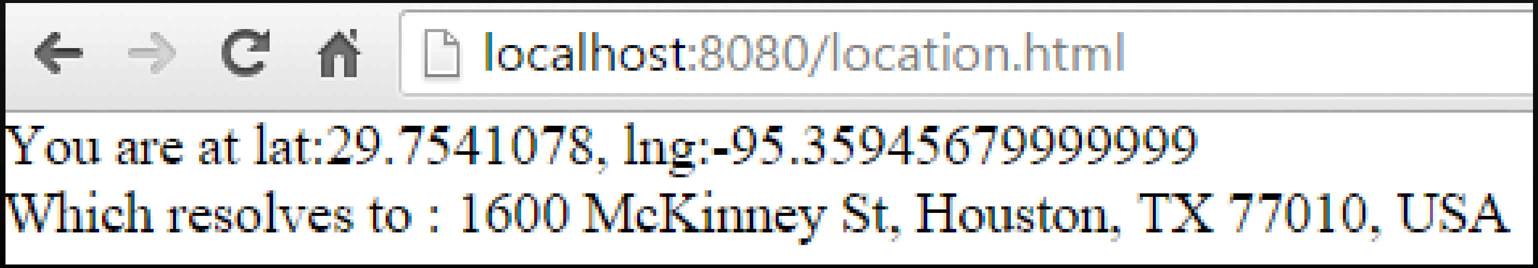
localhost:8080/location.html

http://localhost:8080 wants to:

Know your location



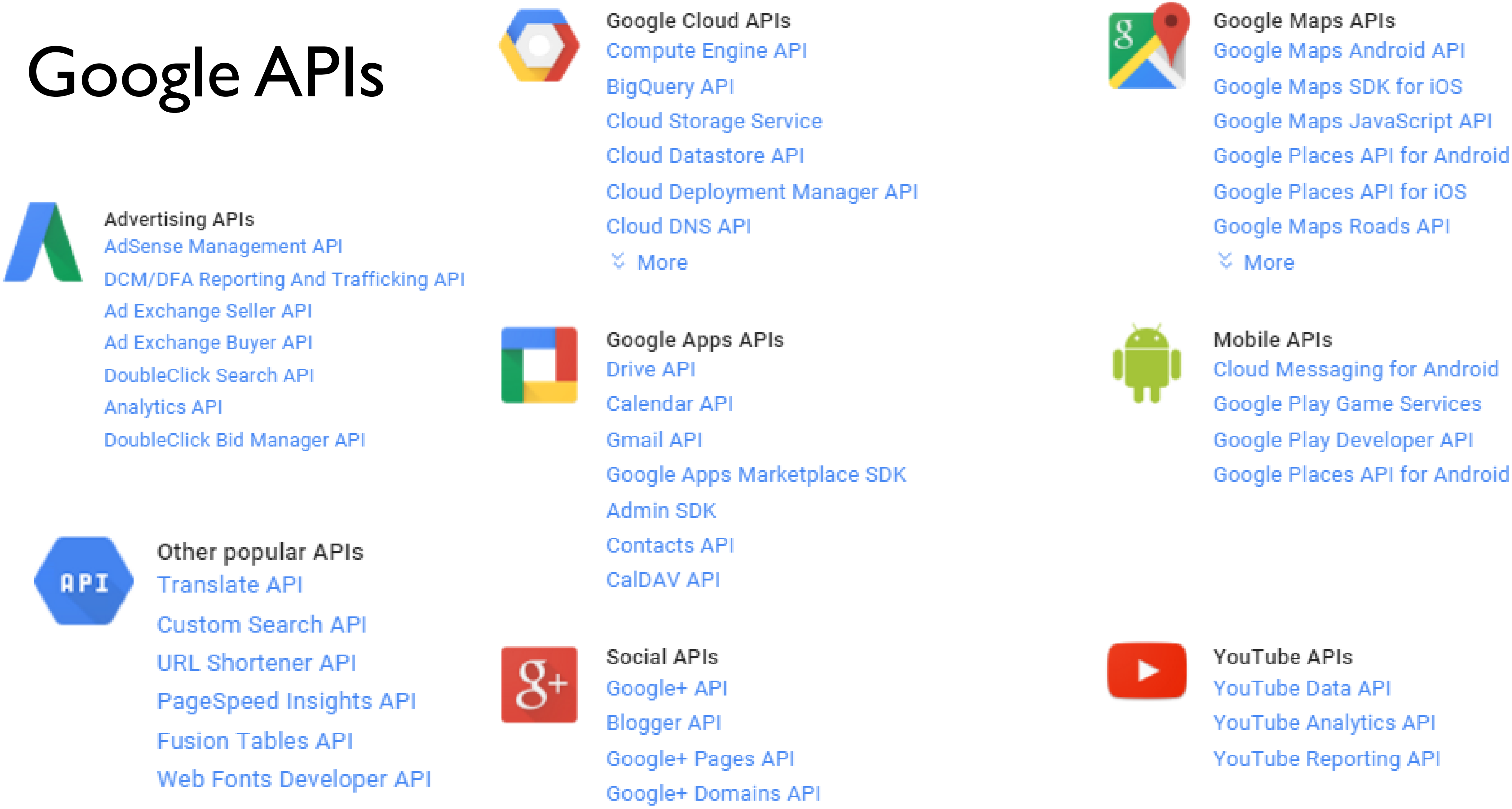
```
if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(function(position) {
        var pos = {
                "lat" : position.coords.latitude,
                "lng" : position.coords.longitude
        }
        $.getJSON('https://maps.googleapis.com/maps/api/geocode/json?latlng='
            + pos.lat + ',' + pos.lng + '&key=[redacted]
        .success(function(data) {
            document.getElementById('it').innerHTML = "You are at "
                + "lat:" + pos.lat + ", lng:" + pos.lng
                + "<br>Which resolves to : " + data.results[0].formatted_address
        })
    })
} else {
    document.getElementById('it').innerHTML = "Location not found or unsupported."
}
```



localhost:8080/location.html

```
You are at lat:29.7541078, lng:-95.35945679999999
Which resolves to : 1600 McKinney St, Houston, TX 77010, USA
```

Google APIs



Advertising APIs
- AdSense Management API
- DCM/DFA Reporting And Trafficking API
- Ad Exchange Seller API
- Ad Exchange Buyer API
- DoubleClick Search API
- Analytics API
- DoubleClick Bid Manager API

Other popular APIs
- Translate API
- Custom Search API
- URL Shortener API
- PageSpeed Insights API
- Fusion Tables API
- Web Fonts Developer API

Google Cloud APIs
- Compute Engine API
- BigQuery API
- Cloud Storage Service
- Cloud Datastore API
- Cloud Deployment Manager API
- Cloud DNS API
- More

Google Apps APIs
- Drive API
- Calendar API
- Gmail API
- Google Apps Marketplace SDK
- Admin SDK
- Contacts API
- CalDAV API

Social APIs
- Google+ API
- Blogger API
- Google+ Pages API
- Google+ Domains API

Google Maps APIs
- Google Maps Android API
- Google Maps SDK for iOS
- Google Maps JavaScript API
- Google Places API for Android
- Google Places API for iOS
- Google Maps Roads API
- More

Mobile APIs
- Cloud Messaging for Android
- Google Play Game Services
- Google Play Developer API
- Google Places API for Android

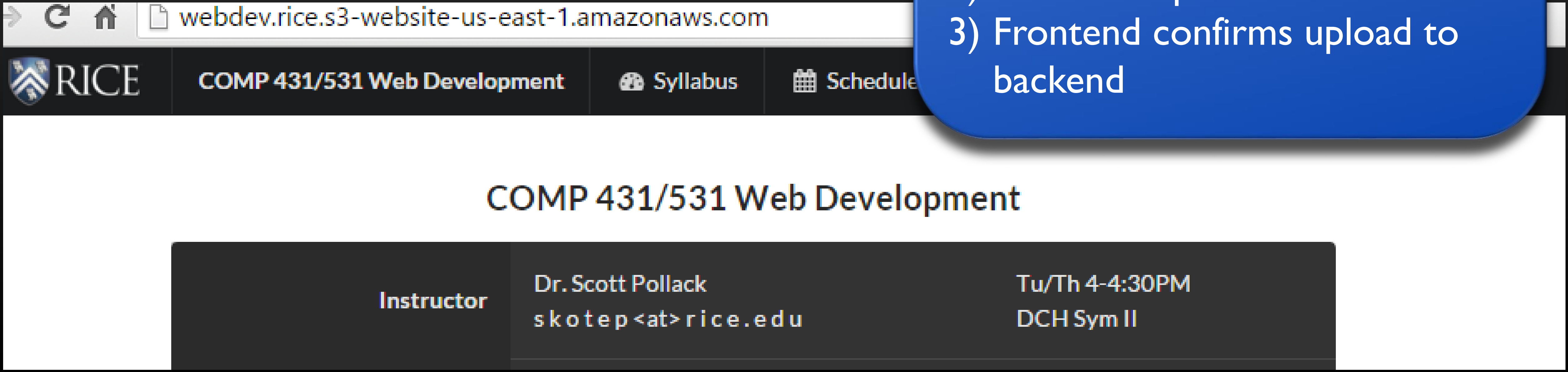
YouTube APIs
- YouTube Data API
- YouTube Analytics API
- YouTube Reporting API

Amazon Web Services: *Simple Storage Service (S3)*

- S3 is composed of buckets
- "blobs" go in the buckets
- Buckets can be permissioned
- We can even web serve from a bucket

Frontend uploads directly to S3 instead of Heroku backend

1) Frontend GETs signed request from backend
2) Frontend uploads file to S3
3) Frontend confirms upload to backend



webdev.rice.s3-website-us-east-1.amazonaws.com

RICE | COMP 431/531 Web Development | Syllabus | Schedule

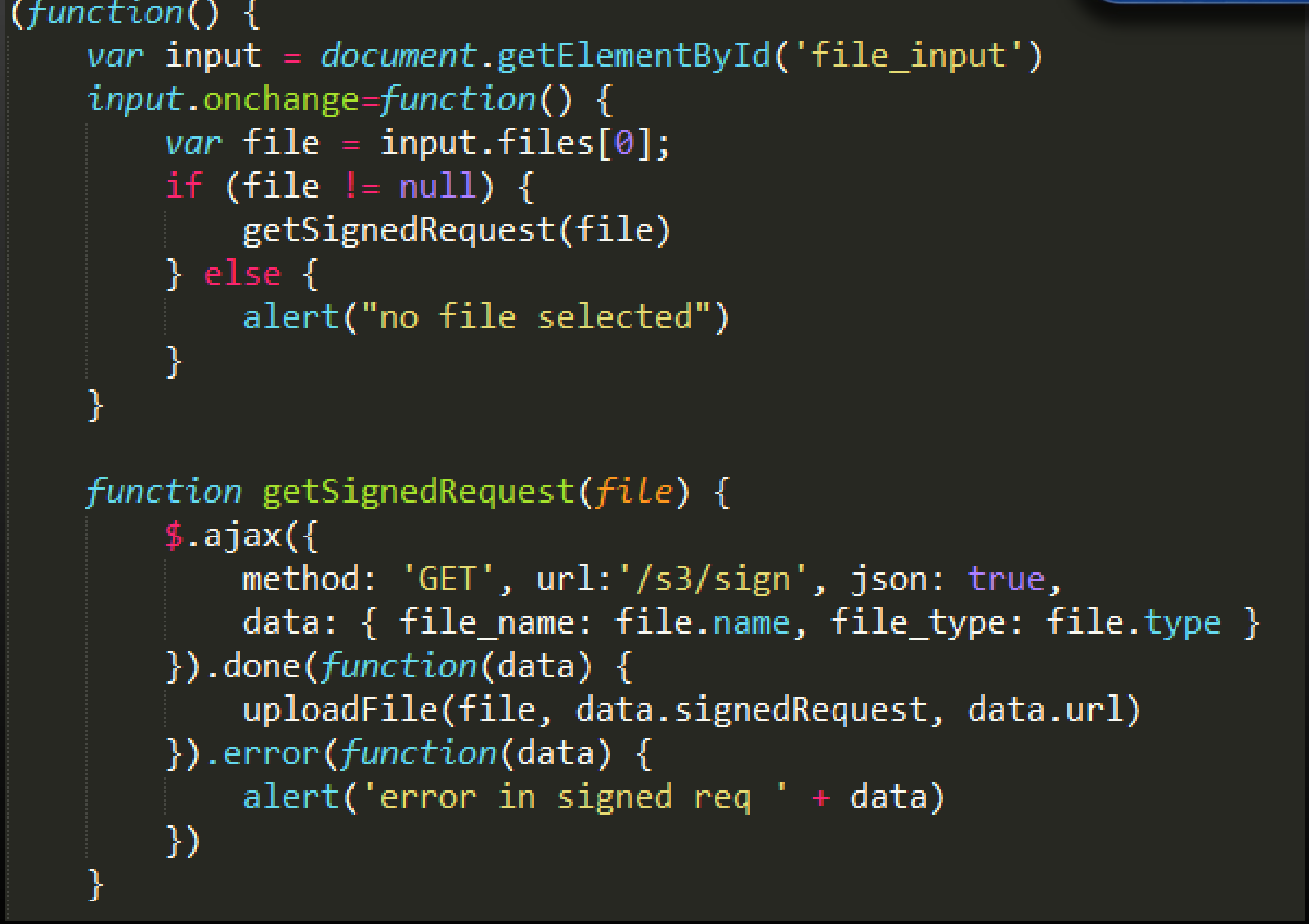
COMP 431/531 Web Development

Instructor	Dr. Scott Pollack skotep<at>rice.edu	Tu/Th 4-4:30PM DCH Sym II

S3 Upload

Frontend uploads directly to S3 instead of Heroku backend

1) Frontend GETs signed request from backend
2) Frontend uploads file to S3
3) Frontend confirms upload to backend

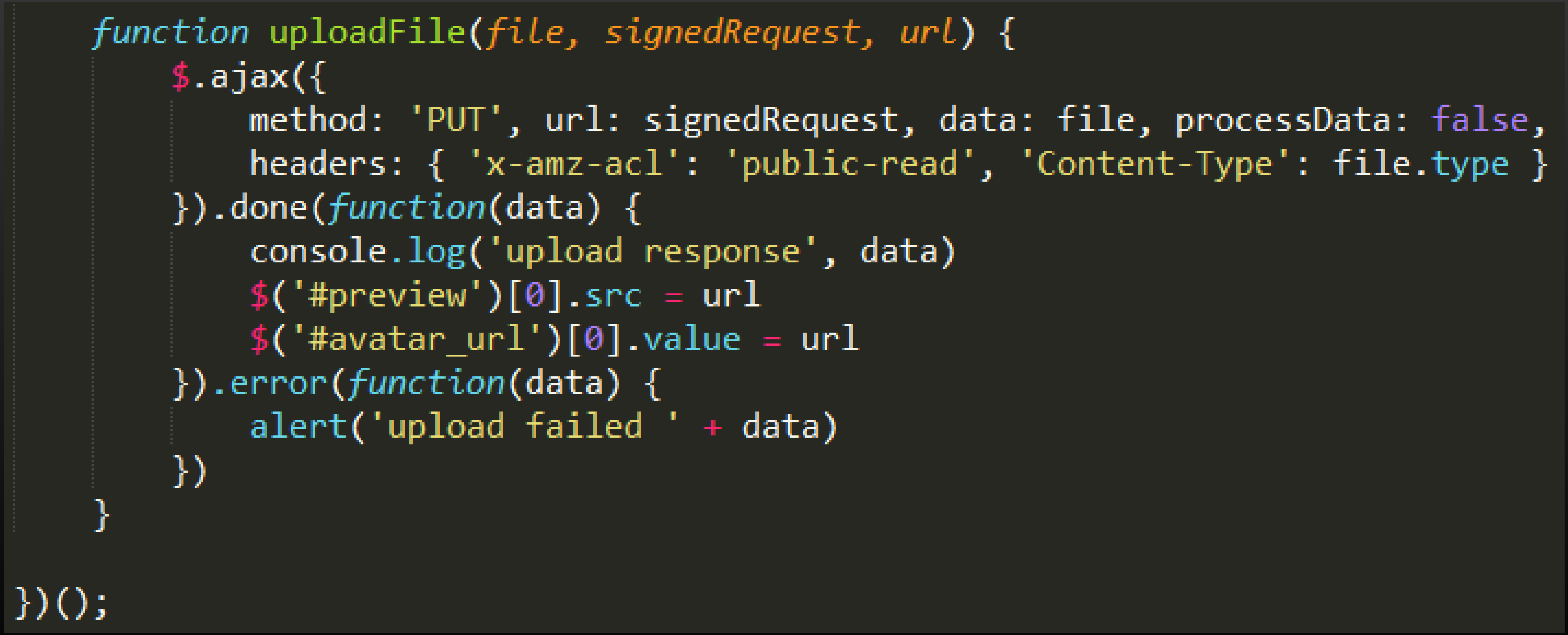


```
(function() {
    var input = document.getElementById('file_input')
    input.onchange=function() {
        var file = input.files[0];
        if (file != null) {
            getSignedRequest(file)
        } else {
            alert("no file selected")
        }
    }

    function getSignedRequest(file) {
        $.ajax({
            method: 'GET', url:'/s3/sign', json: true,
            data: { file_name: file.name, file_type: file.type }
        }).done(function(data) {
            uploadFile(file, data.signedRequest, data.url)
        }).error(function(data) {
            alert('error in signed req ' + data)
        })
    }
```

S3 Upload

Frontend uploads directly to S3 instead of Heroku backend

1) Frontend GETs signed request from backend
2) Frontend uploads file to S3
3) Frontend confirms upload to backend

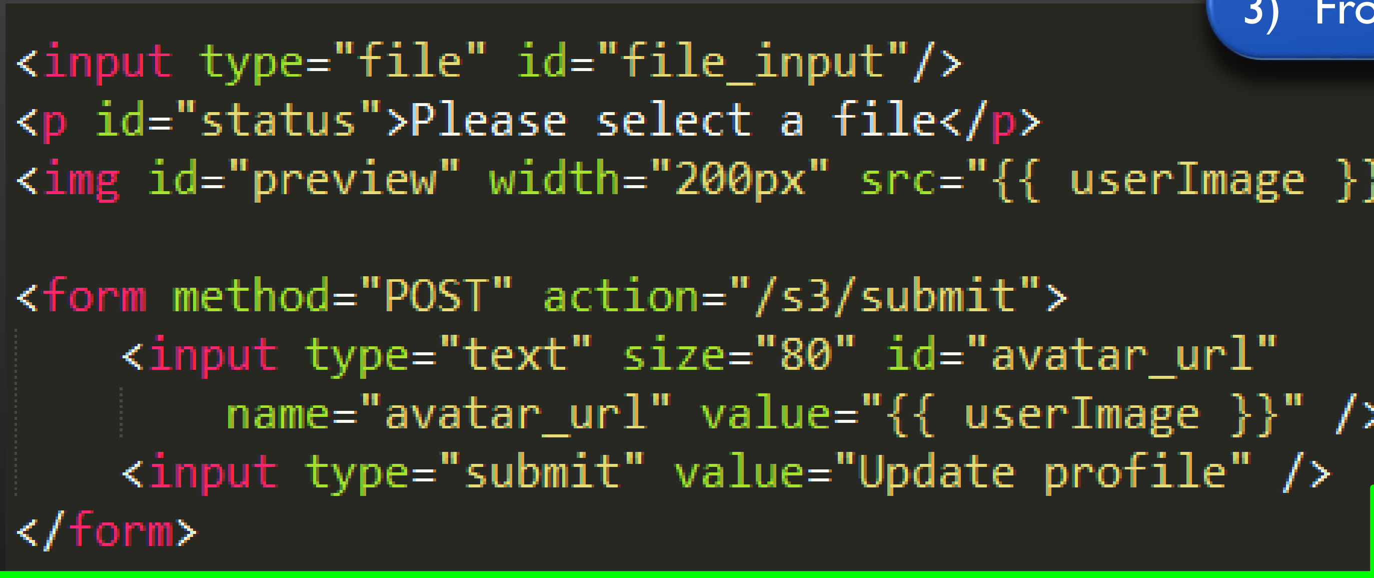


```
    function uploadFile(file, signedRequest, url) {
        $.ajax({
            method: 'PUT', url: signedRequest, data: file, processData: false,
            headers: { 'x-amz-acl': 'public-read', 'Content-Type': file.type }
        }).done(function(data) {
            console.log('upload response', data)
            $('#preview')[0].src = url
            $('#avatar_url')[0].value = url
        }).error(function(data) {
            alert('upload failed ' + data)
        })
    }

})();
```

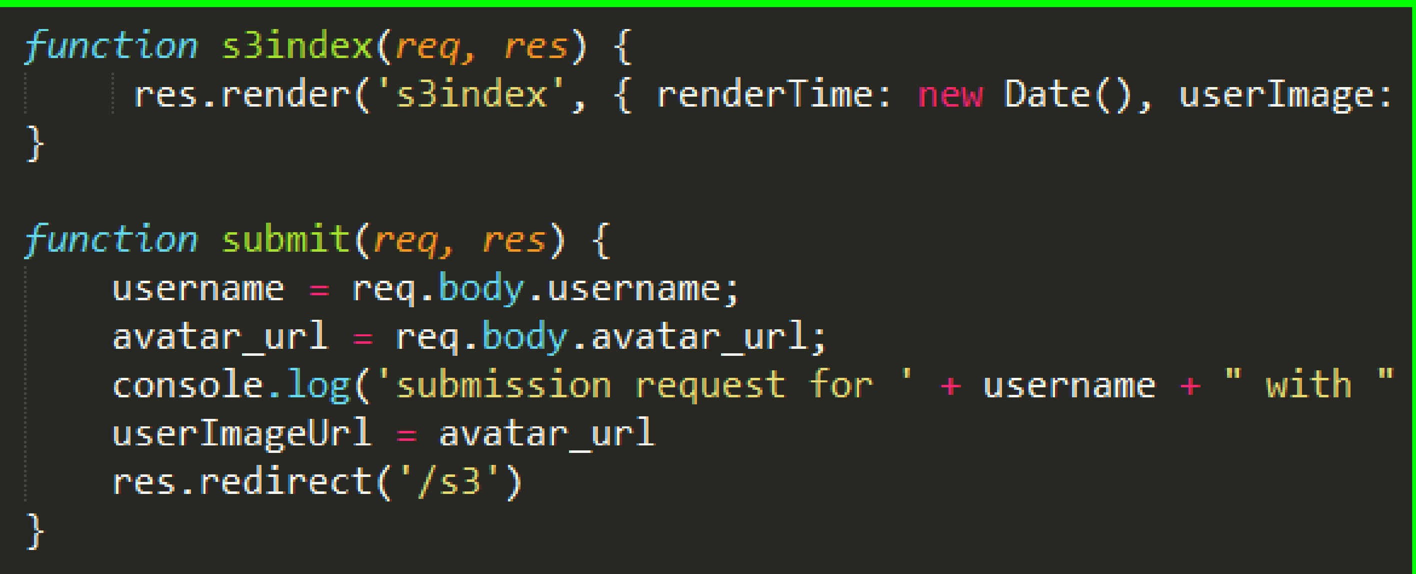
S3 Upload

Frontend uploads directly to S3 instead of Heroku backend

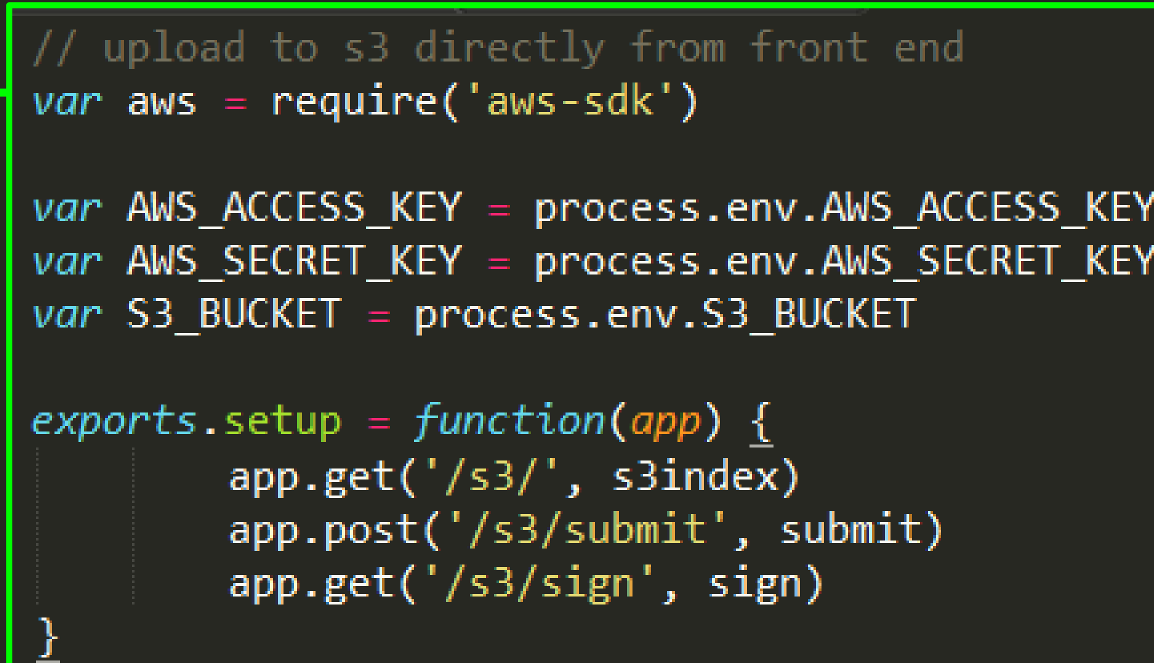
1) Frontend GETs signed request from backend
2) Frontend uploads file to S3
3) Frontend confirms upload to backend



```html
<input type="file" id="file_input"/>
<p id="status">Please select a file</p>
<img id="preview" width="200px" src="{{ userImage }}" />

<form method="POST" action="/s3/submit">
    <input type="text" size="80" id="avatar_url"
        name="avatar_url" value="{{ userImage }}" /><br>
    <input type="submit" value="Update profile" />
</form>
```



```javascript
function s3index(req, res) {
    res.render('s3index', { renderTime: new Date(), userImage:
}

function submit(req, res) {
    username = req.body.username;
    avatar_url = req.body.avatar_url;
    console.log('submission request for ' + username + " with "
    userImageUrl = avatar_url
    res.redirect('/s3')
}
```

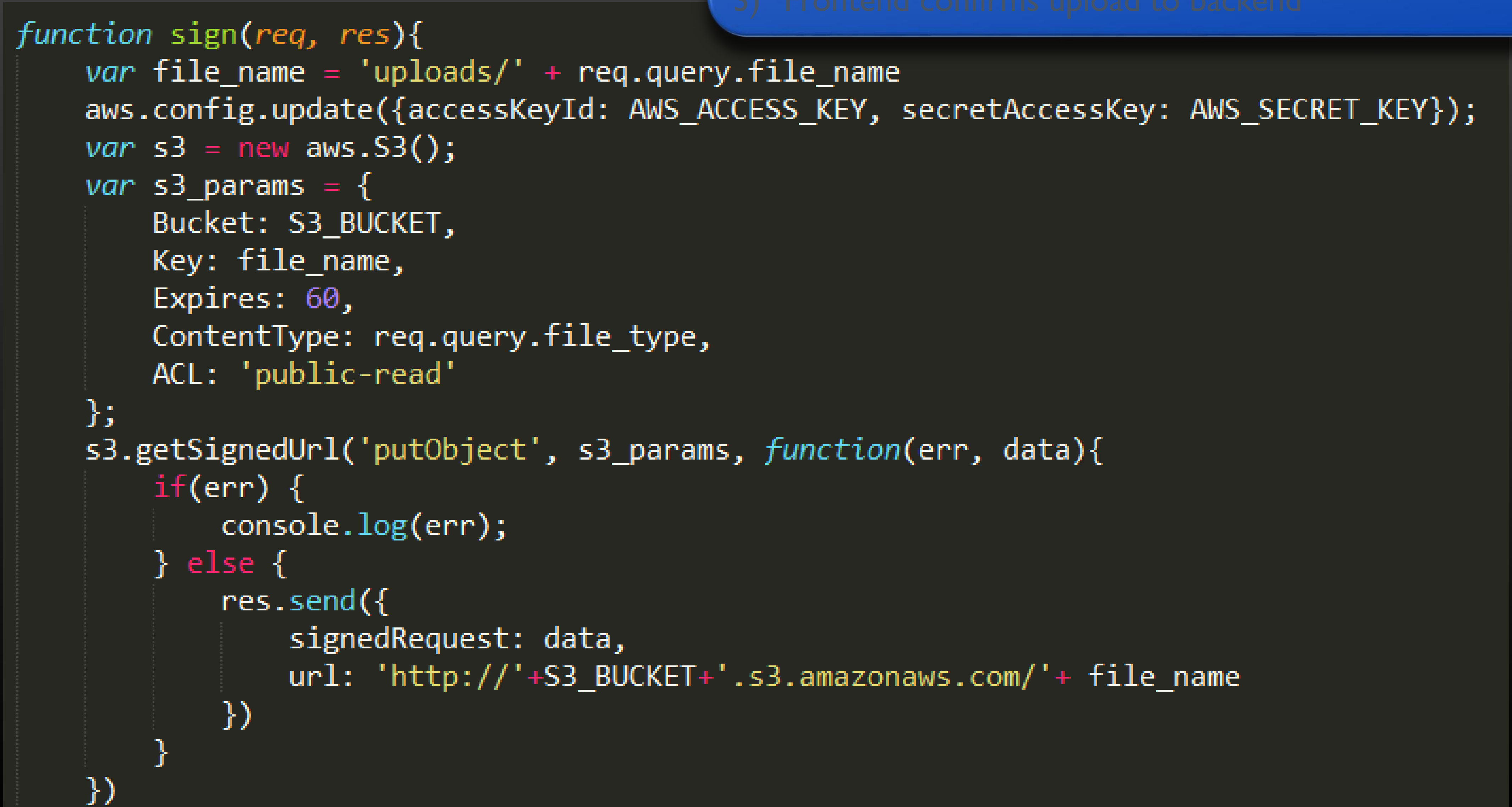


```javascript
// upload to s3 directly from front end
var aws = require('aws-sdk')

var AWS_ACCESS_KEY = process.env.AWS_ACCESS_KEY
var AWS_SECRET_KEY = process.env.AWS_SECRET_KEY
var S3_BUCKET = process.env.S3_BUCKET

exports.setup = function(app) {
        app.get('/s3/', s3index)
        app.post('/s3/submit', submit)
        app.get('/s3/sign', sign)
}
```

S3 Upload

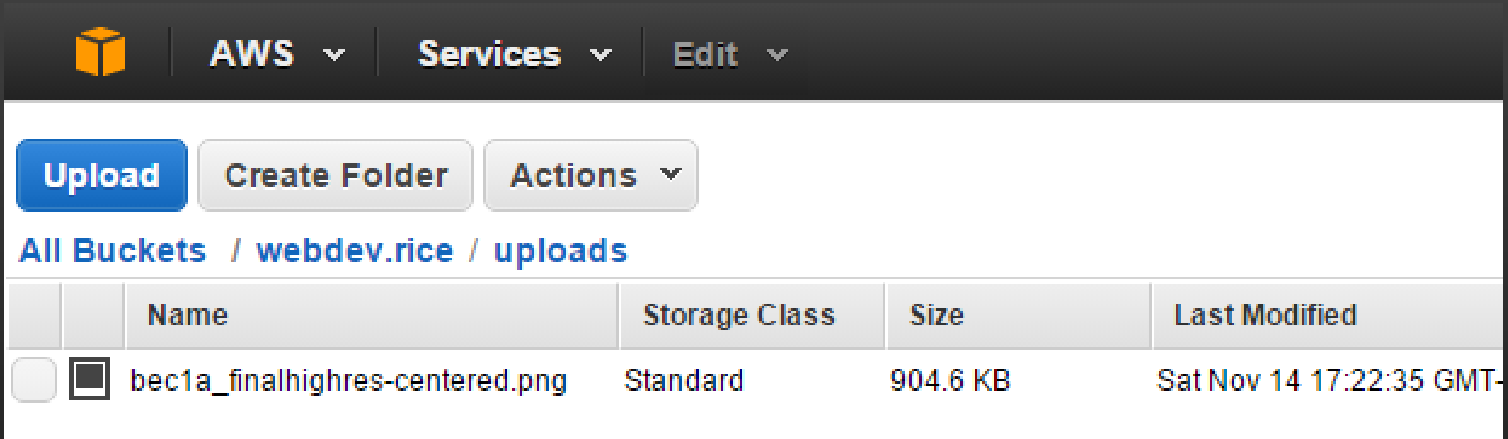
Frontend uploads directly to S3 instead of Heroku backend

1) Frontend GETs signed request from backend
2) Frontend uploads file to S3
3) Frontend confirms upload to backend



```
function sign(req, res){
    var file_name = 'uploads/' + req.query.file_name
    aws.config.update({accessKeyId: AWS_ACCESS_KEY, secretAccessKey: AWS_SECRET_KEY});
    var s3 = new aws.S3();
    var s3_params = {
        Bucket: S3_BUCKET,
        Key: file_name,
        Expires: 60,
        ContentType: req.query.file_type,
        ACL: 'public-read'
    };
    s3.getSignedUrl('putObject', s3_params, function(err, data){
        if(err) {
            console.log(err);
        } else {
            res.send({
                signedRequest: data,
                url: 'http://'+S3_BUCKET+'.s3.amazonaws.com/'+ file_name
            })
        }
    })
```

AWS S3 Upload



All Buckets / webdev.rice / uploads

		Name	Storage Class	Size	Last Modified
○	■	bec1a_finalhighres-centered.png	Standard	904.6 KB	Sat Nov 14 17:22:35 GMT-

This article was contributed by Will Webberley

Will is a computer scientist and is enthused by nearly all aspects of the technology domain. He is specifically interested in mobile and social computing and is currently a researcher in this area at Cardiff University.

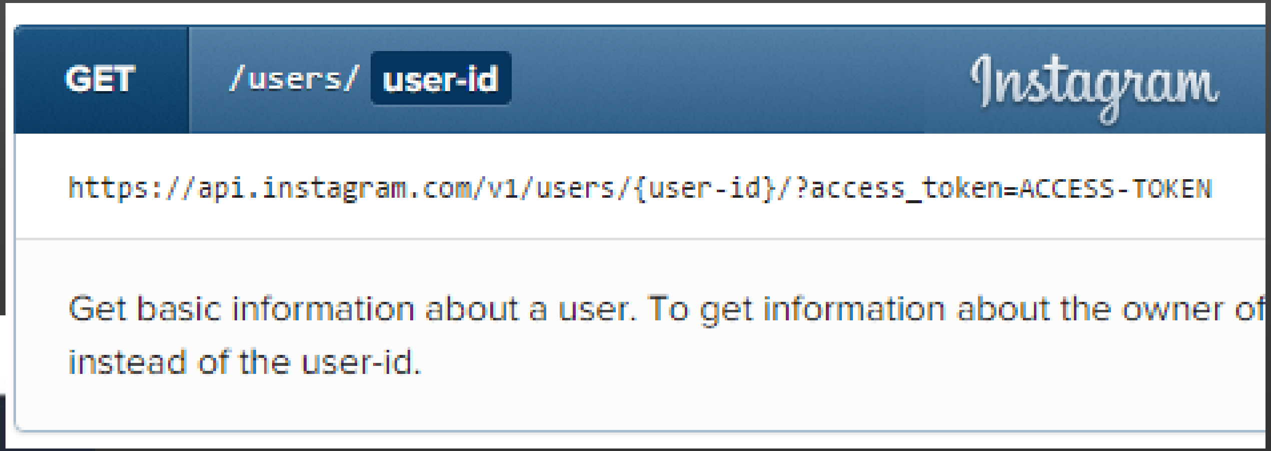
Direct to S3 File Uploads in Node.js

Last updated 29 September 2015

https://devcenter.heroku.com/articles/s3-upload-node

See also perhaps http://www.cheynewallace.com/uploading-to-s3-with-angularjs/

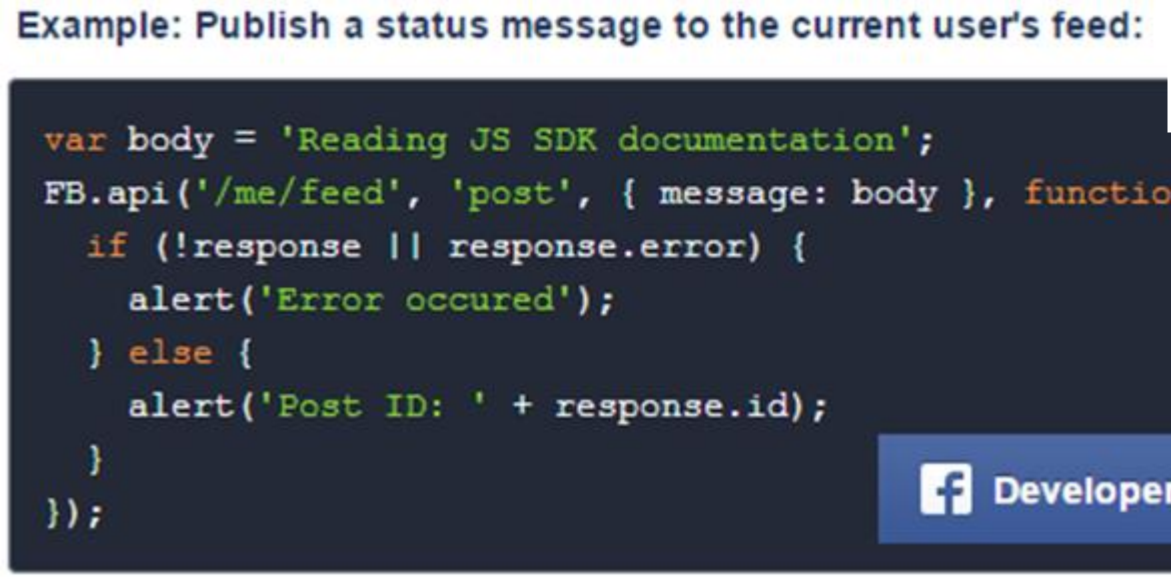
More APIs



GET /users/ user-id Instagram

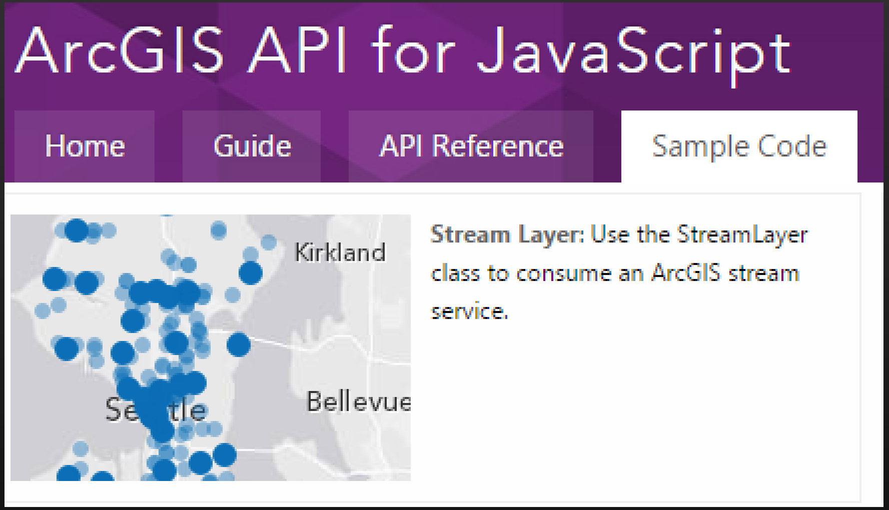
```
https://api.instagram.com/v1/users/{user-id}/?access_token=ACCESS-TOKEN
```

Get basic information about a user. To get information about the owner of
instead of the user-id.



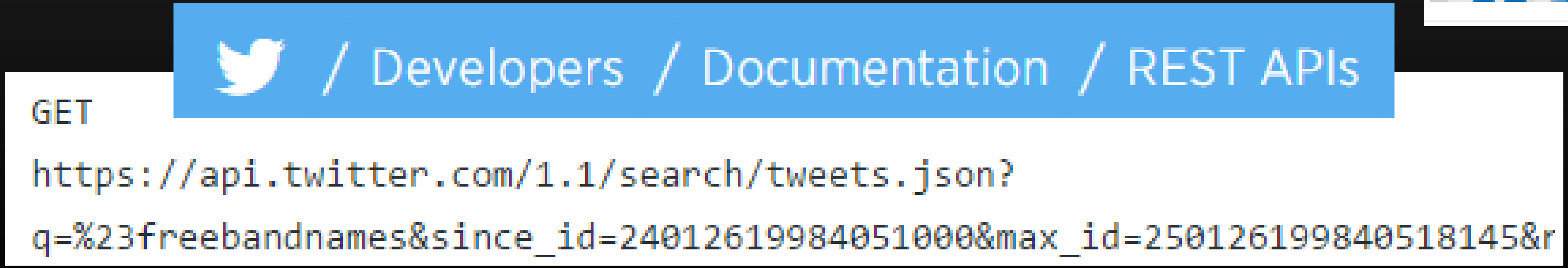
Example: Publish a status message to the current user's feed:

```
var body = 'Reading JS SDK documentation';
FB.api('/me/feed', 'post', { message: body }, function(re
  if (!response || response.error) {
    alert('Error occured');
  } else {
    alert('Post ID: ' + response.id);
  }
});
```



ArcGIS API for JavaScript

Stream Layer: Use the StreamLayer class to consume an ArcGIS stream service.



```
GET
https://api.twitter.com/1.1/search/tweets.json?
q=%23freebandnames&since_id=24012619984051000&max_id=250126199840518145&r
```

Shepherd School of Music
Collegium Concert

Monday, April 18
7:30pm

Hirsch Hall
Alice Pratt Brown Hall

In-Class Exercise: Adding Cloudinary

```
# npm install cloudinary multer --save
# npm install (dotenv|dot-env) --save-dev
# heroku addons:create cloudinary:starter
# heroku config | grep CLOUDINARY >> .env  [or add to .env.json]
https://www.clear.rice.edu/comp431/sample/uploadCloudinary.js
```

```
// add this to index.js
if (process.env.NODE_ENV !== "production") {
    require('dotenv').load()  -OR-  require('dot-env')
}

require('./uploadCloudinary.js').setup(app)
```

Spin up your server and navigate to **/image** try an image upload

Implement the **PUT /picture** endpoint

Turn in your js source for PUT /picture (and anything related)

COMP431-S16:inclass-25