Searching Earthquake Data

Introduction

Duke
UNIVERSITY

Earthquake Data

- Using earthquake data as basis for study
  - Object orientation with classes
  - Searching, sorting, understanding data
  - Parsing and transforming data

Duke
UNIVERSITY

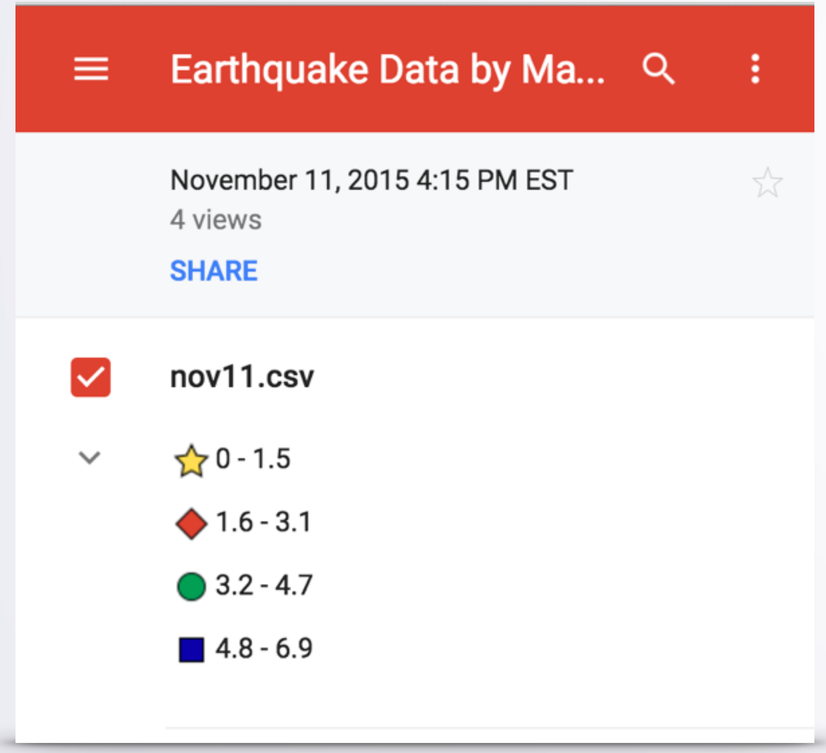
Earthquake Data

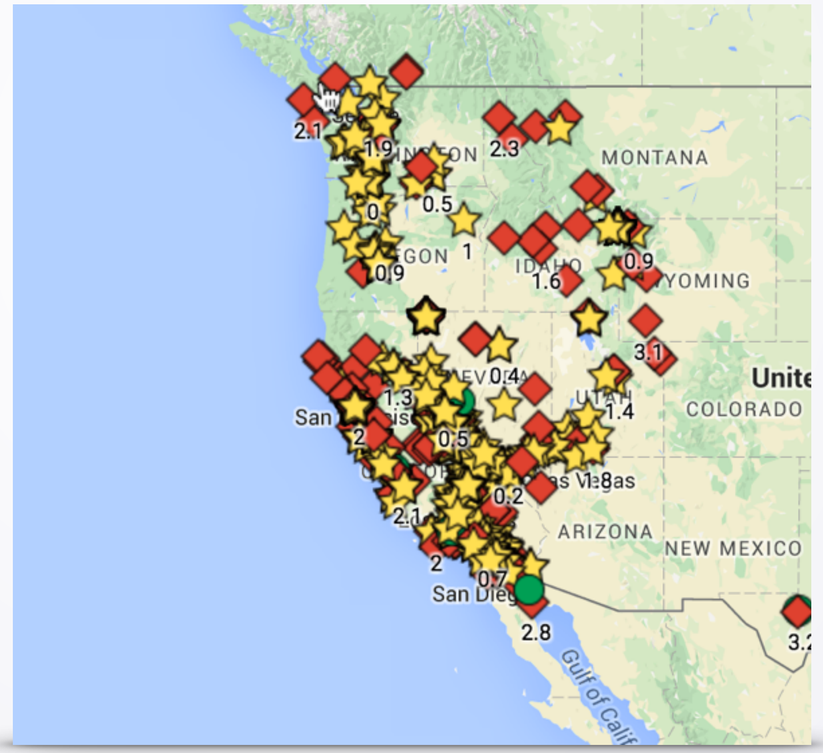
- Using earthquake data as basis for study
  - Object orientation with classes
  - Searching, sorting, understanding data
  - Parsing and transforming data
- Transition to further study
  - Capstone Project
  - UCSD Specialization
  - On your own

Duke
UNIVERSITY

Real-time Data, November 11, 2015

- http://bit.ly/dukesoftware-quake11
  Visualizing earthquake data in California, United States

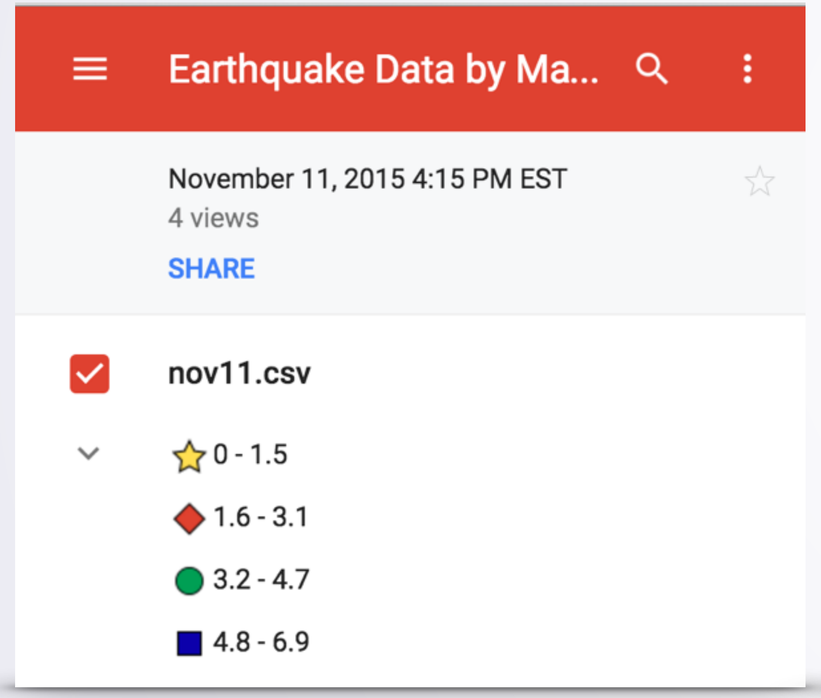


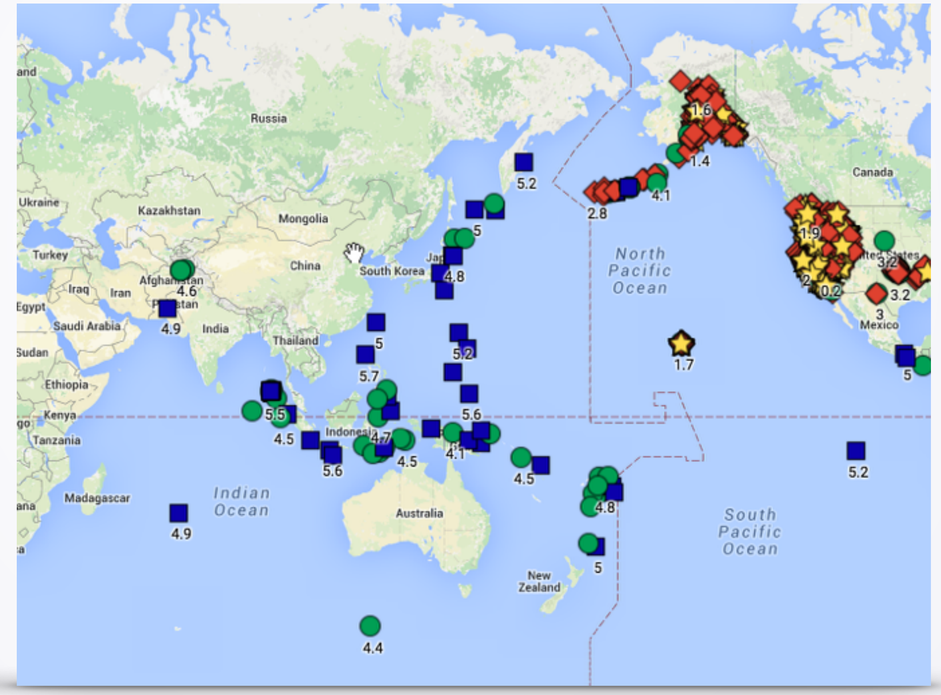


Duke UNIVERSITY

World/Asian Quakes, 11/11/2015

- Larger magnitude quakes ocean based?
  - Creating maps by creating CSV files





Transforming and Using Data

- Data from http://earthquake.usgs.gov is in XML format. JSON another data standard

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:georss=
<entry><id>urn:earthquake-usgs-gov:us:1000309d</id>
<title>M 4.4 - 106km NNE of Tobelo
<dt>Depth</dt><dd>9.84 km (6.11 mi)</dd></dl>]]></summary>
<georss:point>-16.3676 -173.2287</georss:point>
<georss:elev>-9840</georss:elev>
```

Transforming and Using Data

- Data from http://earthquake.usgs.gov is in XML format. JSON another data standard
  - Widespread standards, tricky to parse

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:georss=
<entry><id>urn:earthquake-usgs-gov:us:1000309d</id>
<title>M 4.4 - 106km NNE of Tobelo
<dt>Depth</dt><dd>9.84 km (6.11 mi)</dd></dl>]]></summary>
<georss:point>-16.3676 -173.2287</georss:point>
<georss:elev>-9840</georss:elev>
```

Transforming and Using Data

- Data from http://earthquake.usgs.gov is in XML format. JSON another data standard
  - Widespread standards, tricky to parse
- We need CSV format for Google Maps

```
Latitude,Longitude,Magnitude,Info
60.08,-152.76,2.50,45km S of Redoubt Volcano, Alaska
36.33,-115.76,1.30,24km NE of Pahrump, Nevada
33.83,-117.00,1.00,7km NW of San Jacinto, California
33.83,-117.00,0.80,7km NW of San Jacinto, California
```

Transforming and Using Data

- Data from http://earthquake.usgs.gov is in XML format. JSON another data standard
  - Widespread standards, tricky to parse
- We need CSV format for Google Maps
  - Also used in other mapping services

```
Latitude,Longitude,Magnitude,Info
60.08,-152.76,2.50,45km S of Redoubt Volcano, Alaska
36.33,-115.76,1.30,24km NE of Pahrump, Nevada
33.83,-117.00,1.00,7km NW of San Jacinto, California
33.83,-117.00,0.80,7km NW of San Jacinto, California
```

Transforming and Using Data

- Data from http://earthquake.usgs.gov is in XML format. JSON another data standard
  - Widespread standards, tricky to parse
- We need CSV format for Google Maps
  - Also used in other mapping services
  - Transforming data common application

```
Latitude,Longitude,Magnitude,Info
60.08,-152.76,2.50,45km S of Redoubt Volcano, Alaska
36.33,-115.76,1.30,24km NE of Pahrump, Nevada
33.83,-117.00,1.00,7km NW of San Jacinto, California
33.83,-117.00,0.80,7km NW of San Jacinto, California
```

Parsing and Transforming Data

- Parsing isn't simple, so let's use an API we created to return collection of QuakeEntry objects

```
public void createCSV(){
    EarthQuakeParser parser = new EarthQuakeParser();
    String source = "http://earthquake.usgs.gov/…";
    ArrayList<QuakeEntry> list  = parser.read(source);
    dumpCSV(list);
    System.out.println("# quakes read: "+list.size());
}
```

Parsing and Transforming Data

- Parsing isn't simple, so let's use an API we created to return collection of QuakeEntry objects

```
public void createCSV(){
    EarthQuakeParser parser = new EarthQuakeParser();
    String source = "http://earthquake.usgs.gov/…";
    ArrayList<QuakeEntry> list  = parser.read(source);
    dumpCSV(list);
    System.out.println("# quakes read: "+list.size());
}
```

Parsing and Transforming Data

- Parsing isn't simple, so let's use an API we created to return collection of QuakeEntry objects
  - Parse file or URL, testing or real time

```
public void createCSV(){
    EarthQuakeParser parser = new EarthQuakeParser();
    String source = "http://earthquake.usgs.gov/…";
    ArrayList<QuakeEntry> list  = parser.read(source);
    dumpCSV(list);
    System.out.println("# quakes read: "+list.size());
}
```

Parsing and Transforming Data

- Parsing isn't simple, so let's use an API we created to return collection of QuakeEntry objects
  - Parse file or URL, testing or real time

```
public void createCSV(){
    EarthQuakeParser parser = new EarthQuakeParser();
    String source = "http://earthquake.usgs.gov/…";
    ArrayList<QuakeEntry> list  = parser.read(source);
    dumpCSV(list);
    System.out.println("# quakes read: "+list.size());
}
```

Parsing and Transforming Data

- Parsing isn't simple, so let's use an API we created to return collection of QuakeEntry objects
  - Parse file or URL, testing or real time

```
public void createCSV(){
    EarthQuakeParser parser = new EarthQuakeParser();
    String source = "http://earthquake.usgs.gov/…";
    ArrayList<QuakeEntry> list  = parser.read(source);
    dumpCSV(list);
    System.out.println("# quakes read: "+list.size());
}
```

QuakeEntry and Location Classes

- QuakeEntry holds relevant data from USGS XML feed
  - Location
  - Magnitude
  - Depth
  - Description

```
public class QuakeEntry {

    private Location myLocation;
    private String title;
    private double depth;
    private double magnitude;

    public QuakeEntry(…){

    }
```

QuakeEntry and Location Classes

- QuakeEntry holds relevant data from USGS XML feed
  - Location
  - Magnitude
  - Depth
  - Description

```
public class QuakeEntry {

    private Location myLocation;
    private String title;
    private double depth;
    private double magnitude;

    public QuakeEntry(…){

    }
```

- Location is a separate class
  - One class often uses another
  - Location code adapted from Android standard