CS 646 Android Mobile Application Development
Spring Semester, 2015
Doc 10 Http, JSON
March 2, 2015

Android Programming: Big Nerd Ranch

Chapter 26 HTTP & Background Tasks

Covers HTTP & AsyncTask

Chapter 27 Loopers, Handlers & HandlerThread

Data and Smart Phones

Smart Phones have network connections

Web browser gives user web page

How does app get data from server?

Ways to transport data on the Network

HTTP

- Web scraping
- **Data as content**

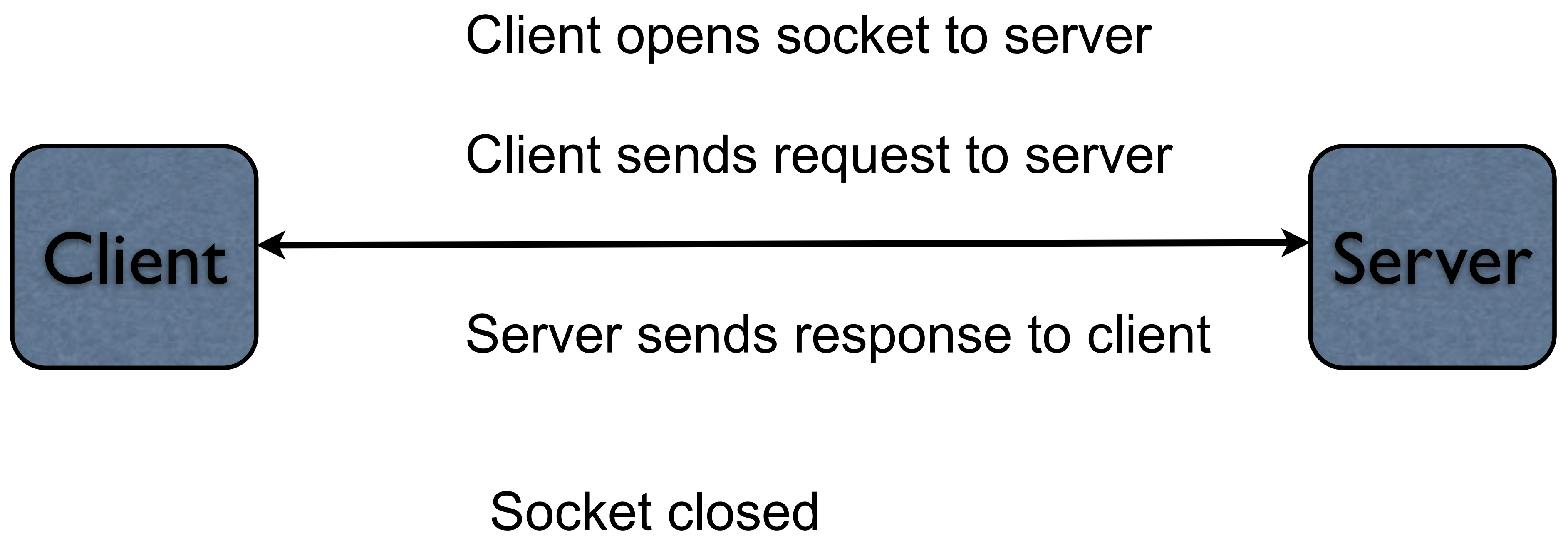
SOAP
XML-RPC
RMI

Network Programming

- Open a network socket etc
- See CS 580 - Client Server Programming

HTTP Basics

Http Basic Operation


Client opens socket to server
Client sends request to server
Client
Server
Server sends response to client
Socket closed

HTTP Protocol

Requests to server have headers and body

Responses have headers and body

Web browsers
- hide protocol from users
- Just show rendered web pages

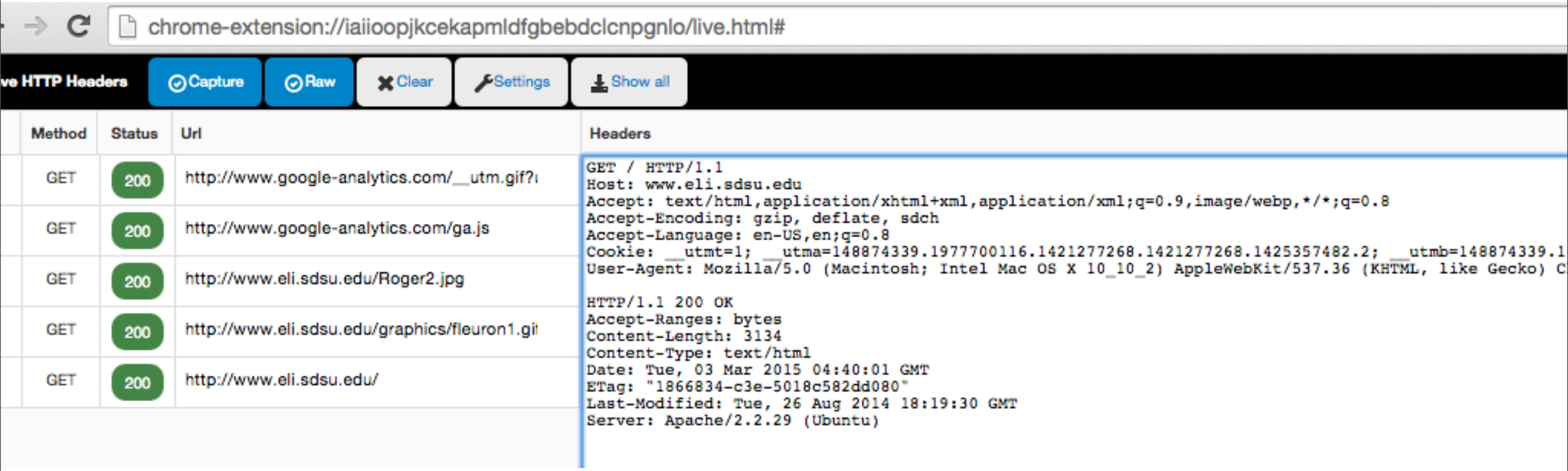
Behind the Scenes with Chrome

Add Live HTTP Headers - chrome extension

Open the Extension

Then load a web page

Headers - www.eli.sdsu.edu



chrome-extension://iaiioopjkcekapmldfgbebdclcnpgnlo/live.html#
ve HTTP Headers
Capture
Raw
Clear
Settings
Show all

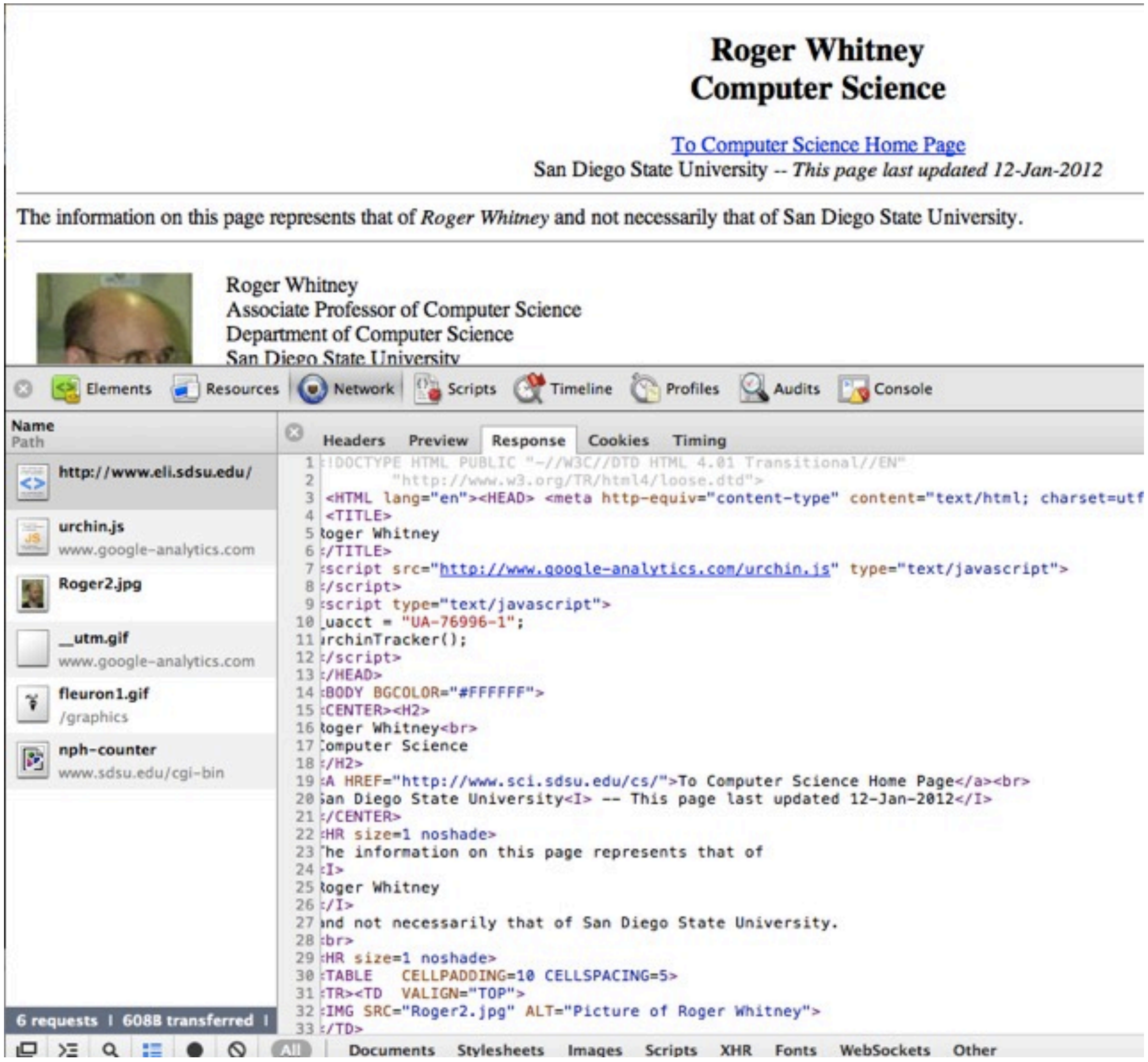
Method	Status	Url
GET	200	http://www.google-analytics.com/__utm.gif?
GET	200	http://www.google-analytics.com/ga.js
GET	200	http://www.eli.sdsu.edu/Roger2.jpg
GET	200	http://www.eli.sdsu.edu/graphics/fleuron1.gi
GET	200	http://www.eli.sdsu.edu/

Headers

```
GET / HTTP/1.1
Host: www.eli.sdsu.edu
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
Cookie: __utmt=1; __utma=148874339.1977700116.1421277268.1421277268.1425357482.2; __utmb=148874339.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_2) AppleWebKit/537.36 (KHTML, like Gecko) C

HTTP/1.1 200 OK
Accept-Ranges: bytes
Content-Length: 3134
Content-Type: text/html
Date: Tue, 03 Mar 2015 04:40:01 GMT
ETag: "1866834-c3e-5018c582dd080"
Last-Modified: Tue, 26 Aug 2014 18:19:30 GMT
Server: Apache/2.2.29 (Ubuntu)
```

Body - www.eli.sdsu.edu



Roger Whitney
Computer Science

To Computer Science Home Page
San Diego State University -- *This page last updated 12-Jan-2012*

The information on this page represents that of *Roger Whitney* and not necessarily that of San Diego State University.

Roger Whitney
Associate Professor of Computer Science
Department of Computer Science
San Diego State University

Name / Path
http://www.eli.sdsu.edu/
urchin.js www.google-analytics.com
Roger2.jpg
__utm.gif www.google-analytics.com
fleuron1.gif /graphics
nph-counter www.sdsu.edu/cgi-bin

6 requests | 608B transferred |

```
!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
        "http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD> <meta http-equiv="content-type" content="text/html; charset=utf
<TITLE>
oger Whitney
/TITLE>
script src="http://www.google-analytics.com/urchin.js" type="text/javascript">
/script>
script type="text/javascript">
uacct = "UA-76996-1";
rchinTracker();
/script>
/HEAD>
BODY BGCOLOR="#FFFFFF">
CENTER><H2>
oger Whitney<br>
omputer Science
/H2>
A HREF="http://www.sci.sdsu.edu/cs/">To Computer Science Home Page</a><br>
an Diego State University<I> -- This page last updated 12-Jan-2012</I>
/CENTER>
HR size=1 noshade>
he information on this page represents that of
I>
oger Whitney
/I>
nd not necessarily that of San Diego State University.
br>
HR size=1 noshade>
TABLE   CELLPADDING=10 CELLSPACING=5>
TR><TD  VALIGN="TOP">
IMG SRC="Roger2.jpg" ALT="Picture of Roger Whitney">
/TD>
```

HTTP protocol by hand

You can use telnet to take directly to a web server

netcat is better but most people already have telnet

Example

```
Al pro 20->telnet www.sdsu.edu 80
GET Trying 130.191.8.198...
Connected to www.sdsu.edu.
Escape character is '^]'.
GET /index.html HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 02 Feb 2012 21:38:15 GMT
Server: Apache/2.2.14 (Ubuntu)
Last-Modified: Thu, 02 Feb 2012 21:35:10 GMT
ETag: "55f9b-1af87-4b801f878ff80"
Accept-Ranges: bytes
Content-Length: 110471
Vary: Accept-Encoding
Connection: close
Content-Type: text/html


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Types of HTTP Requests

GET
POST
PUT
HEAD
DELETE
TRACE
OPTIONS
CONNECT
PATCH

GET
    Retrieving web pages

POST
    Sending data to server

HTTP Request and Response

- Request
  - URL of page we want
  - Headers with information
  - Body - extra information

- Response
  - Headers with information
  - Body - the actual response
    - Normally html of web page
    - Can be anything

Using HTTP for Data

Easier than doing it with just network programming

Can use web server as back end

Existing clients code to make request and get response

Helps avoid fire walls

Data Formats

Html is not meant for data

Use

- XML or JSON

XML

XML

```
<?xml version="1.0" ?>
<CATALOG>
        <CD>
                <TITLE>Empire Burlesque</TITLE>
                <ARTIST>Bob Dylan</ARTIST>
                <COUNTRY>USA</COUNTRY>
                <COMPANY>Columbia</COMPANY>
                <PRICE>10.90</PRICE>
                <YEAR>1985</YEAR>
        </CD>
        <CD>
                <TITLE>Hide your heart</TITLE>
                <ARTIST>Bonnie Tyler</ARTIST>
                <COUNTRY>UK</COUNTRY>
                <COMPANY>CBS Records</COMPANY>
                <PRICE>9.90</PRICE>
                <YEAR>1988</YEAR>
        </CD>
</CATALOG>
```

XML & Android

Android three XML parsers

- W3C DOM
- SAX
- XML Pull Parser

Why not use XML?

Verbose

- Slower
- Consumes more resources

JSON

JSON

http://www.json.org/

JavaScript Object Notation

data-interchange format

rfc 4627

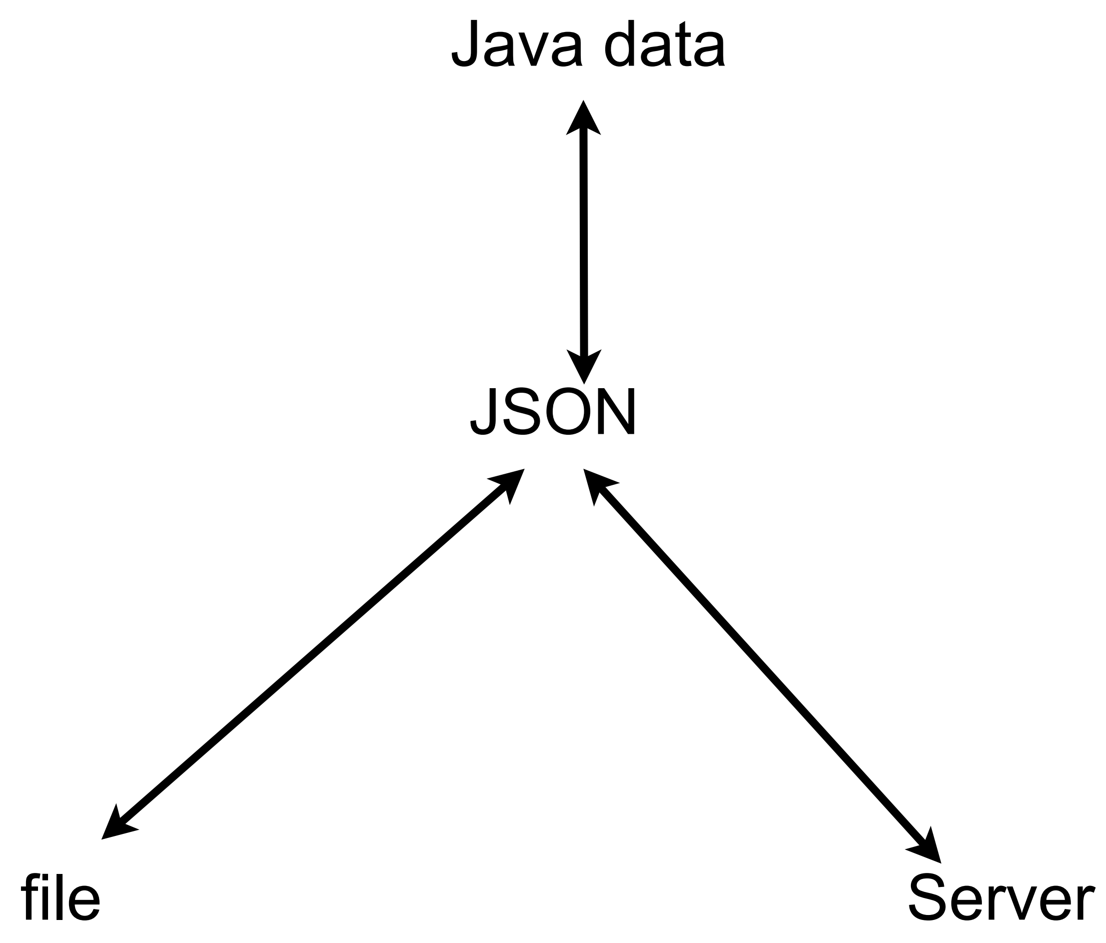
Maps to/from strings

- null
- true, false
- number
- string
- array
- objects

Implementations in

- C, C++, C#, D, E, Java, Objective C
- Cold Fusion, Delphi, Erlang, Haskell
- JavaScript, Lisp, LotusScript, Perl,
- PHP, Pike, Prolog, Python, Ruby, Smalltalk

Data Interchange



Number & String & Constants

123
32.4
0.12
2.3e3
+5
-6
5.93e-2

"a string"

"a string with \" a quote char"

\ escape - normal special char

    \n
    \t
    \\
    \/
    \b
    \f
    \u

null
true
false

Arrays

```
[1, 2, 3]

["cat", 2, true]

[23.4, ["dog", null], 10]
```

JSONArrays can hold any valid JSON data type

Object (Dictionary)

```
{"key": "value"}
    keys have to be strings
    value can be any legal JSON data type
```

```
{"name": "Roger", "age": 21}
```

```
{"id":2,"office":"GMCS 407B","phone":"619-594-6191","email":"beck@cs.sdsu.edu","rating":{"average":5.0,"totalRatings":1},"firstName":"Dr. Leland","lastName":"Beck"}
```

Valid JSON Document

One top level item

- Array
- Object

How to Convert

$$\begin{tikzcd}\text{Java data} \arrow[d, leftrightarrow] \\ \text{JSON}\end{tikzcd}$$

Android uses standard code from

http://www.json.org/java/index.html

JSON in Java

Included in Android

http://www.json.org/java/index.html

A Java JSON library

Main Classes

JSONObject
    Deals with top level JSON object

JSONArray
    Deals with top level JSON array

JSONObject

```
try {
    JSONObject json = new JSONObject();
    json.put("lowerBound", 18);
    json.put("upperBound", 139);
    json.put("name", "cat");

    String objectString = json.toString();
                          // "{"lowerBound":18,"upperBound":139, "name":"cat"}"

    JSONObject newJson = new JSONObject(objectString);

    int bound = newJson.getInt("lowerBound"); // 18
} catch (JSONException error) {
    Log.i("rew", "error", error);
}
```

JSONObject

Main methods

getX(String key)

put(String key, Y value)

Where X =
- Boolean
- Double
- Int
- JSONArray
- JSONObject
- Long
- String

Where Y =
- int
- long
- boolean
- double
- Object

JSONArray

```
JSONArray data = new JSONArray();
data.put(5);
data.put("Cat");
try {
    int value = data.getInt(0);
    Log.i("rew", "value " + value);
} catch (JSONException e) {
    e.printStackTrace();
}

String dataString = data.toString();
```

HTTP In Android

Using the Internet

In Manifest file

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Android Thread Rules

Don't block the UI thread

Activity code runs on the UI thread
Create threads to perform long operations

Do not access the Android UI toolkit from outside the UI thread

Use the following to access UI thread
- Activity.runOnUiThread(Runnable)
- View.post(Runnable)
- View.postDelayed(Runnable, long)

Actually these are the rules for all UI frameworks that I have used

Apache HttpComponents HttpClient

http://hc.apache.org/

Java classes that implements Http 1.1 protocol

- Talks to web server
- Gets responses for you

Does not

- Render web page
- Run Javascript
- Handle CSS

HttpClients & Android

HttpClient is included in Android

So can use in Android code

In past you could run this on main thread

Newer system will not allow this

AndroidHttpClient

HttpClient with reasonable default settings for Android

Create with
    AndroidHttpClient.newInstance (String userAgent)

Will not run on main thread

AndroidHttpClient

Basic usage

- Create HttpGet object with url to visit
- Create BasicResponseHandler object to handle response
- Call execute on AndroidHttpClient

GET example

```
public void runNetworkCode(View button) {
		String url = "http://www.eli.sdsu.edu/";
		String userAgent = null;
		HttpClient httpclient = AndroidHttpClient.newInstance(userAgent);
		HttpGet getMethod = new HttpGet(url);
		try {
			ResponseHandler<String> responseHandler = new 
BasicResponseHandler();
			String responseBody = httpclient.execute(getMethod, 
responseHandler);
			Log.i("rew", responseBody);
		} catch (Throwable t) {
			Log.i("rew","did not work", t);
		}
		httpclient.getConnectionManager().shutdown();
		return null;
}
```

But the Main thread issue

```
public void runNetworkCode(View button) {
		String url = "http://www.eli.sdsu.edu/";
		String userAgent = null;
		HttpClient httpclient = AndroidHttpClient.newInstance(userAgent);
		HttpGet getMethod = new HttpGet(url);
		try {
			ResponseHandler<String> responseHandler = new
BasicResponseHandler();
			String responseBody = httpclient.execute(getMethod,
responseHandler);
```

Will not run in main thread

So use AsyncTask

```
class HttpClientTask extends AsyncTask<Void, Void, Void> {
	protected Void doInBackground(Void... arg0) {
		String url = "http://www.eli.sdsu.edu/";
		HttpClient httpclient = AndroidHttpClient.newInstance(null);
		HttpGet getMethod = new HttpGet(url);
		try {
			ResponseHandler<String> responseHandler =
		 new BasicResponseHandler();
			String responseBody = httpclient.execute(getMethod,
responseHandler);
			Log.i("rew", responseBody);
		} catch (Throwable t) {
			Log.i("rew","did not work", t);
		}
		httpclient.getConnectionManager().shutdown();
		return null;
	}
}
```

Using the AsyncTask

```
public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        HttpClientTask task = new HttpClientTask();
        task.execute();
    }
```

But not very useful

AsyncTask only access one URL

All work done in other thread so can not interact with any UI elements

Improved Version

```
public class MainActivity extends Activity {
    HttpClient httpclient;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void onResume() {
        super.onResume();
        String userAgent = null;
        httpclient = AndroidHttpClient.newInstance(userAgent);
        HttpClientTask task = new HttpClientTask();
        String url = "http://www.eli.sdsu.edu/";
        task.execute(url);
    }
```

```
public void onPause() {
	super.onPause();
	httpclient.getConnectionManager().shutdown();
}
```

New HttpClientTask

```
class HttpClientTask extends AsyncTask<String, Void, String> {

        protected String doInBackground(String... urls) {
            try {
                ResponseHandler<String> responseHandler =
             new BasicResponseHandler();
                HttpGet getMethod = new HttpGet(urls[0]);
                String responseBody = httpclient.execute(getMethod,
responseHandler);
                return responseBody;
            } catch (Throwable t) {
                Log.i("rew","did not work", t);
            }
            return null;
        }

        public void onPostExecute(String result) {
            Log.i("rew", result);  //here you could put contents into UI element
        }
```

Output

```
03-14 12:21:51.066: I/rew(10688): <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN"
03-14 12:21:51.066: I/rew(10688):         "http://www.w3.org/TR/html4/loose.dtd">
03-14 12:21:51.066: I/rew(10688):  <HTML lang="en"><HEAD>   <meta http-
equiv="content-type" content="text/html; charset=utf-8">
03-14 12:21:51.066: I/rew(10688):  <TITLE>
03-14 12:21:51.066: I/rew(10688): Roger Whitney
03-14 12:21:51.066: I/rew(10688): </TITLE>
03-14 12:21:51.066: I/rew(10688): <script src="http://www.google-analytics.com/
urchin.js" type="text/javascript">
03-14 12:21:51.066: I/rew(10688): </script>
03-14 12:21:51.066: I/rew(10688): <script type="text/javascript">
03-14 12:21:51.066: I/rew(10688): _uacct = "UA-76996-1";
03-14 12:21:51.066: I/rew(10688): urchinTracker();
03-14 12:21:51.066: I/rew(10688): </script>
03-14 12:21:51.066: I/rew(10688): </HEAD>
03-14 12:21:51.066: I/rew(10688): <BODY BGCOLOR="#FFFFFF">
```

HttpClient & shutdown

```
httpclient.getConnectionManager().shutdown();
```

Client maintains pool of network connections for better performance

Close connections when done
    Best in onPause

Don't close connections after each request

Get Example with JSON

http://www.eli.sdsu.edu/courses/fall09/cs696/examples/names.json

Returns

```
[{"firstname":"Roger","lastname":"Whitney"},
{"firstname":"Robert","lastname":"Edwards"},
{"firstname":"Kris","lastname":"Stewart"}]
```

No Change in doInBackground

```
class HttpClientTask extends AsyncTask<String, Void, String> {

        @Override
        protected String doInBackground(String... urls) {
            try {
                ResponseHandler<String> responseHandler =
             new BasicResponseHandler();
                HttpGet getMethod = new HttpGet(urls[0]);
                String responseBody = httpclient.execute(getMethod,
responseHandler);
                return responseBody;
            } catch (Throwable t) {
                Log.i("rew","did not work", t);
            }
            return null;
        }
```

Handing JSON in onPostExecute

```
public void onPostExecute(String jsonString) {
    try {
        JSONArray data = new JSONArray(jsonString);
        JSONObject firstPerson = (JSONObject) data.get(0);
        String firstName = firstPerson.getString("firstname");
        String lastName = firstPerson.getString("lastname");
        Log.i("rew", firstName + " " + lastName);
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

Giving the task the URL

```
    public void onResume() {
        super.onResume();
        String userAgent = null;
        httpclient = AndroidHttpClient.newInstance(userAgent);
        HttpClientTask task = new HttpClientTask();
        String url = "http://www.eli.sdsu.edu/courses/fall09/cs696/examples/
names.json";
        task.execute(url);
    }
```

Post Example

```
public void runNetworkCode(View button) {
		String url = "http://bismarck.sdsu.edu/rateme/comment/32";
		HttpClient httpclient = new DefaultHttpClient();
		HttpPost postMethod = new HttpPost(url);
		StringEntity comment;
		try {
			comment = new StringEntity("hi dad", HTTP.UTF_8);
		} catch (UnsupportedEncodingException e) {
			Log.i("rew", e.toString());
			return;
		}
		postMethod.setHeader("Content-Type", "application/json;charset=UTF-8");
		postMethod.setEntity(comment);
		try {
			HttpResponse responseBody = httpclient.execute(postMethod);
		} catch (Throwable t) {
			Log.i("rew", t.toString());
		}
		httpclient.getConnectionManager().shutdown();
	}
```

Uploading File using Post

```
class HttpClientTask extends AsyncTask<String, Void, String> {
    protected String doInBackground(String... urls) {
        try {
            ResponseHandler<String> responseHandler = new
BasicResponseHandler();
            HttpPost getMethod = new HttpPost(urls[0]);
            FileEntity photo = new FileEntity(getFileStreamPath("bird"), "image/jpeg");
            getMethod.setEntity(photo);
            String responseBody = httpclient.execute(getMethod, responseHandler);
            return responseBody;
        } catch (Throwable t) {
            Log.i("rew","did not work", t);
        }
        return null;
    }

    public void onPostExecute(String result) {
        Log.i("rew", result);
    }
}
```

Assumes that one has created a file called 'birds' in the default location on Android device/emulator