CSCB20 – Week 11

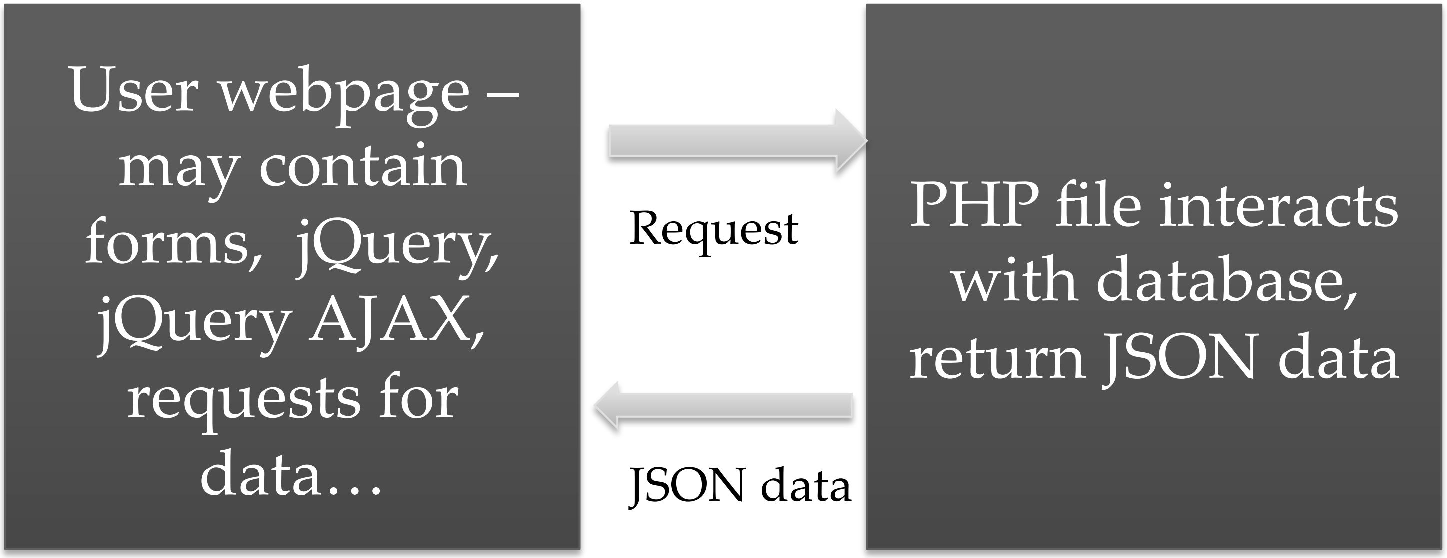
Introduction to Database and Web Application Programming

*Anna Bretscher**

Winter 2017

*thanks to Alan Rosselet for providing the slides these are adapted from.

This Week



User webpage – may contain forms, jQuery, jQuery AJAX, requests for data…

Request

JSON data

PHP file interacts with database, return JSON data

HTML Forms

Most interesting Web apps are based on data, e.g.:

- Google's compilation of search indices of crawled Web content
- Amazon's product catalog and product reviews
- Facebook's user pages
- Flickr's image catalog
- YouTube's video collection

Many of these apps provide Application Programming Interfaces (API), enabling programs to work with their data, e.g. Amazon product catalog

Many apps allow users to enter their own data, e.g. upload videos to YouTube, upload photos to Flickr

Often control Web app behavior with parameter values

HTML Forms

- Collect user input
  - for processing by client-side scripts, e.g. jQuery
  - for transmission to server-side programs
- Constructed out of widgets/controls, e.g.:
  - text boxes
  - checkboxes
  - pull-down select menus
  - radio buttons
  - submit and reset buttons
- Each widget/control has a value
- When a form is submitted, widget values are collected together and sent to server

HTML Forms: `<form>`

`<form>` element encompasses entire input-form structure

```
<form action="URL" method="GET" >
```

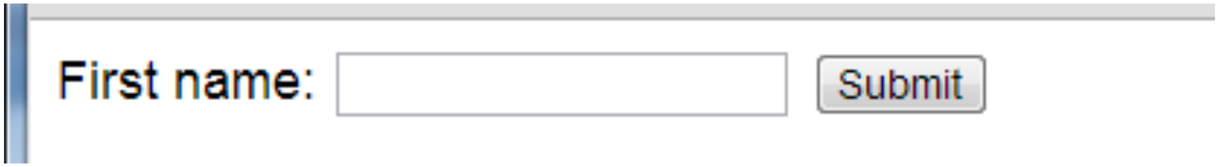
Attributes:

- `action`
  - identifies URL of server-side program to be invoked in response to submit
- `method`
  - either "GET" (default) or "POST"

A single Web page may contain many forms

Forms: <form>

```
<form action="search.php" method="GET" >
 <div>
    First Name: <input type="text” id="firstname"/>
               <input type="submit" />
   </div>
</form>
```



form fields can be wrapped in a `<div>` element or a `<fieldset>` block element.

Forms: `<input>`

Example:

```
<input type="radio” id="cardtype”
    name="cardtype" value="amex" />
```

Attributes:

- type: text boxes, checkboxes, radio buttons, user-defined buttons, file, hidden, password, reset and submit buttons
- name, id: for reference e.g. by script, HTTP query parameter
- value: default for checkboxes and radio buttons, initial text for others

Forms: <input>

Text inputs can have a placeholder value displayed within the text-input field to prompt the user, e.g.

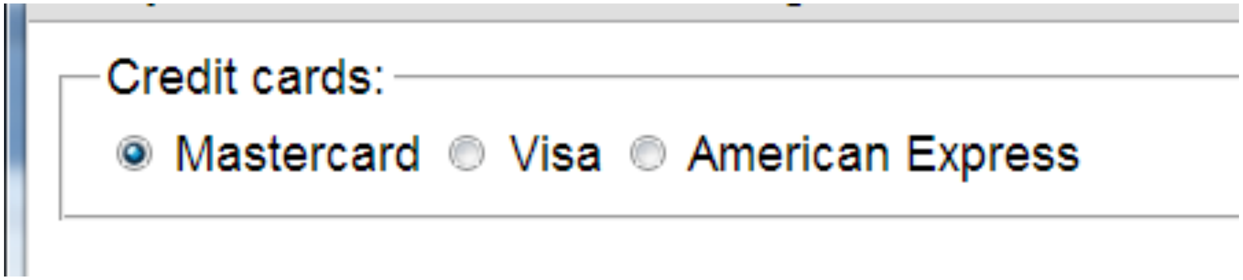
```
<input type="text" id="firstname”
name="firstname" placeholder="First Name"/>
```

controls can be labelled with <label> element. “for” attribute connects label to matching input-field “id”. e.g.:

```
<label for="cardtype">Card Type</label>
<input type="radio" id="cardtype"
name="cardtype" value="amex" />
```

Forms: <fieldset>, <legend>

```
<fieldset>
  <legend>Credit cards:</legend>
  <input type="radio" name="cc" value="visa"
                  checked="checked" /> Visa
  <input type="radio" name="cc"
            value="mastercard" /> MasterCard
  <input type="radio" name="cc"
            value="amex" /> American Express
</fieldset>
```



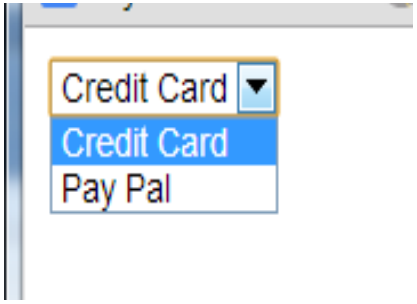
Credit cards:

● Mastercard ○ Visa ○ American Express

fieldset groups related-input set and puts a border around them

legend provides a caption for the form fields

Forms: <select>

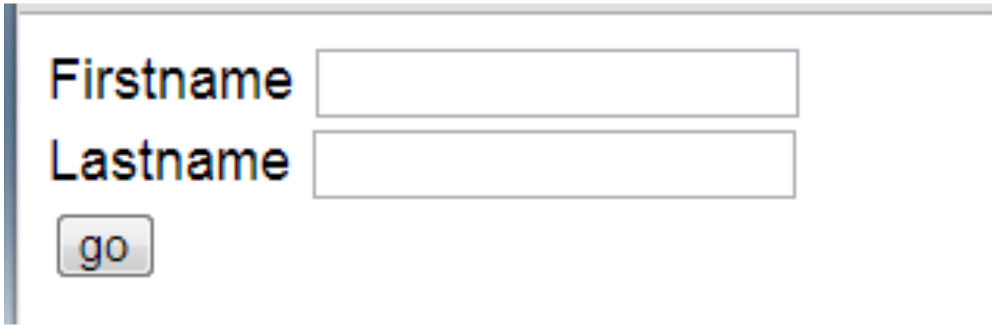


- <select> element encompasses:
  - menus
- attributes:
  - name, id – for referencing, e.g. by script
  - size – for drop down and scrolling lists
  - multiple – allow multiple values to be selected
- <option> element children of <select> define the allowed values for the menu
- e.g.:

```
<select>
    <option value=“1”>Credit Card</option>
    <option value=“2”>Pay Pal</option>
  </select>
```

Forms: type hidden

```
<label>Firstname
  <input type="text" name="firstname" /></label><br/>
<label>Lastname
  <input type="text" name="lastname" /></label><br/>
<input type=“hidden" name="fullname" />
<input type="submit" value="go" />
```



Note hidden form fields are still passed to the server as parameter values, referenced by their name fields

Forms: submit and reset types

```
<input type="reset" … />
```

- clears all user selections/data and returns form to initial state

```
<input type="submit" value="filter" />
```

- encode all widget/control values into a string value (essentially a parameter list), called the "query string"; example:

  name1=value1&name2=value2&name3=value3

- pass that string value (parameter list) to an application on the server, as specified in form "action" attribute

  ```
  URL?name1=value1&name2=value2&name3=value3
  ```

- In PHP on server, e.g.: `$_REQUEST["name1"]`

URL-encoding

Form data appended to URL as

```
?name1=value1&name2=value2&…
```

A few complications:

- e.g. “=”, “&”, “|”, “/”, or “<“ character in value
- space character in value

Browser automatically “encodes” these special cases, and on server side, PHP automatically “decodes” them

- e.g. space character is encoded as “%20”

HTTP: the Web’s protocol

Used “behind the scenes” to send browser requests and server responses

Provides “methods” that effect how client data is passed to the server – forms have a “method” attribute to set the HTTP method.

GET: intended for *read-only requests*, such as static Web page retrieval, or simple server-program calls

POST: intended for transmitting general-case client data to the server

HTTP: GET and POST

```
<form action="URL" method="GET" >
```

or

```
<form action="URL" method=”POST" >
```

GET appends URL-encoded data to the end of the server URL as shown on previous slide

- not private
- easily tampered by user
- length limits

POST places URL-encoded data in the “body” of the request

- more private (not visible in browser)
- harder to tamper
- no length limits

jQuery and Forms

Detecting event types

Suppose a user makes a choice from a `<select>` element; how do you detect that they made a choice, and how can you determine what they chose?

```
... header with script tags ...
$("mymenu").change(showMe);
function showMe() {
  alert(this.value + "was chosen");
}
</script>
<select id="mymenu">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="opel">Opel</option>
  <option value="audi">Audi</option>
</select>
```

Dealing with forms

- An HTML form element wraps a set of input types, such as buttons, input-text fields, “radio” buttons, checkboxes

```
<form method="GET" action="search.php">
… <input type="submit" value="go"> … other fields
</form>
```

- When the user clicks on the input element with `type “submit”`, which is displayed as a button, the values of the form’s fields are submitted to the URL named by the “`action`” parameter.
- Sometimes you may need to interrupt that normal submit behavior, e.g. to perform validation or some other action using the form fields.

Dealing with forms

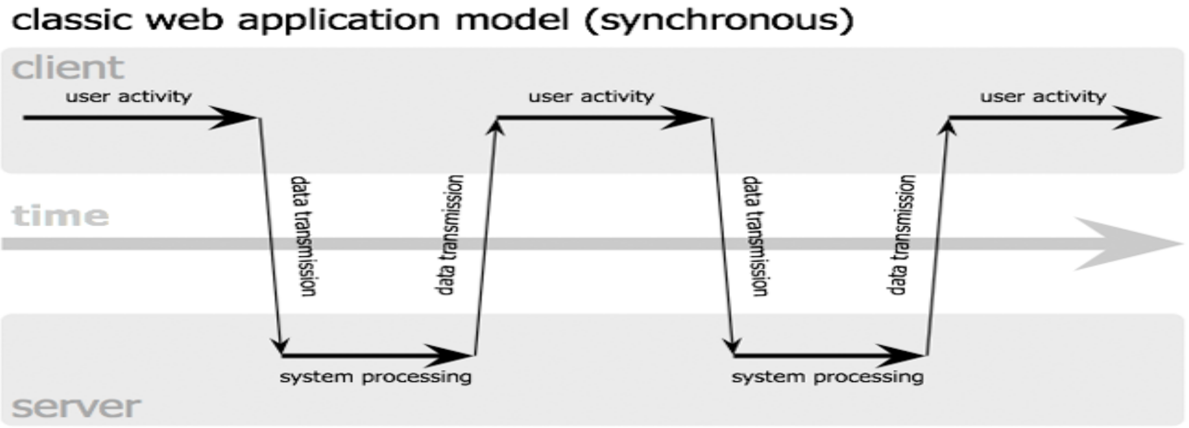
```
<form id="f1" method="GET" action="search.php">
… <input type="submit" value="go"> … other fields
</form>

$(function() {
  $("#f1").submit(getSomething);
});

function getSomethine(event) {
   event.preventDefault(); // suspend submit
   /* code to invoke myquery.php script to get
    info from database to create <select> menu */
}
```

You may
need this in
your A3

Ajax: Asynchronous Requests

- Traditional Web applications follow the underlying HTTP protocol model of *request and response* page-flow.
- A client Web page makes a request to the server, which results in a new Web page being returned by the server, replacing the original page in the browser.

The client blocks while awaiting the server response. We refer to this kind of request as synchronous



Ajax: Asynchronous Requests

Nothing wrong with synchronous requests when building Web apps that consist of a sequence of page views (traditional view-click-view-click cycle)

Sometimes we want Web apps that should not block (freeze) while waiting for the server – we need a different model.

Leads to AJAX…

Web 2.0, RIA (Rich Internet Apps)

RIA/Web-2.0 apps attempt to bridge gap between "native" (desktop) apps and traditional Web apps

Goals:

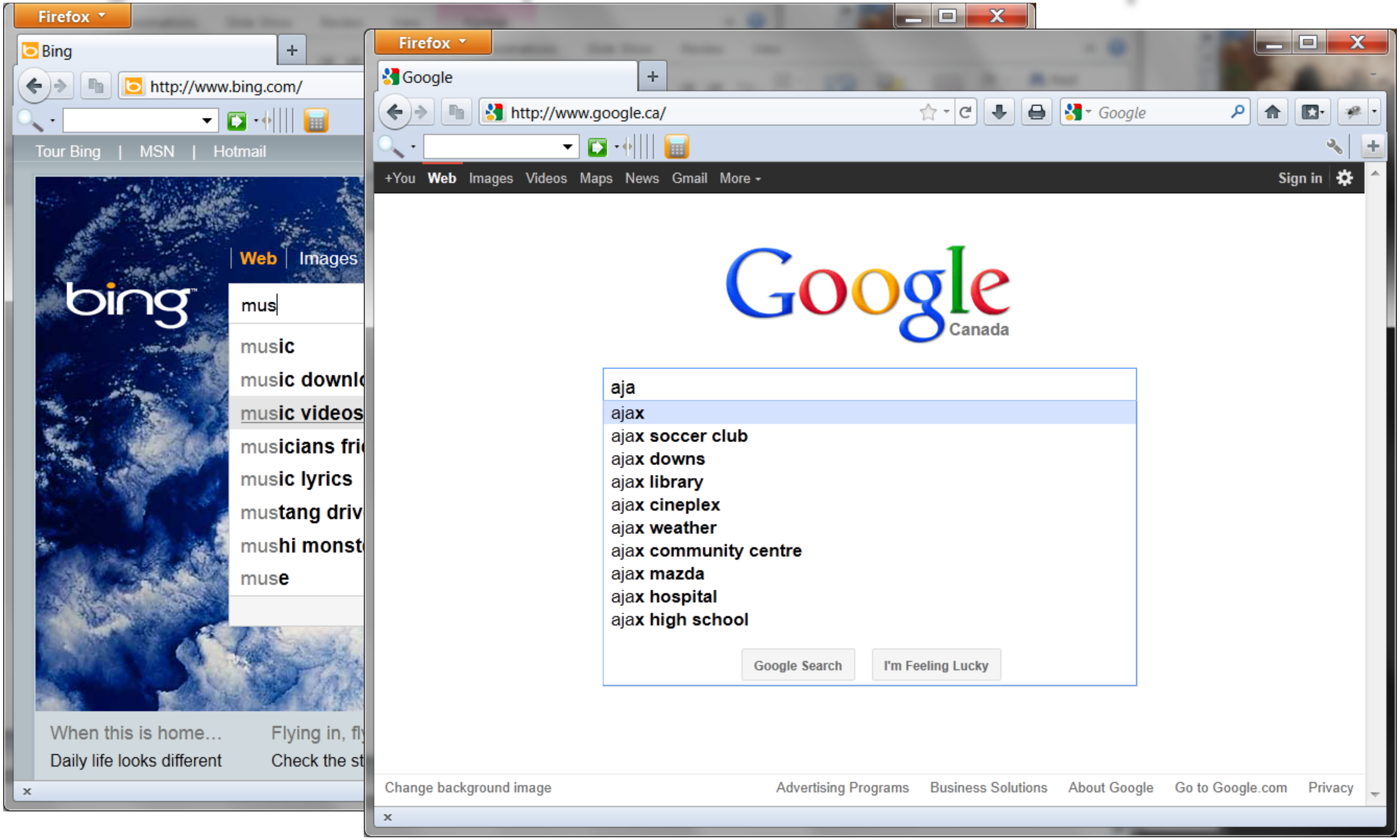
- more interactivity (can achieve some of this with JS and the DOM without server-side interaction)
- would like to achieve performance similar to native (desktop) applications

A major change is a shift from page-oriented interaction to "*event-driven*" programming where events result in updates to *portions* of pages.

Ajax Use

- Many mainstream apps now utilize Ajax to improve the user experience, e.g. GMail, Google maps, Google finance page, Yahoo search, Flickr
- What is Ajax used for?
  - real-time *semantic form-data validation* (not just regex checks, but actual value checks against e.g. a server DB
  - Auto-completion (e.g. Google suggest, Yahoo search)
  - dynamic page interaction, e.g. user drag/pan results in new data display (e.g. Google maps, Google finance)
  - auto-refresh of page data (e.g., real-time stock quotes embedded within document)
  - interactive user-to-user communications, e.g. Google talk

Ajax example: autocomplete



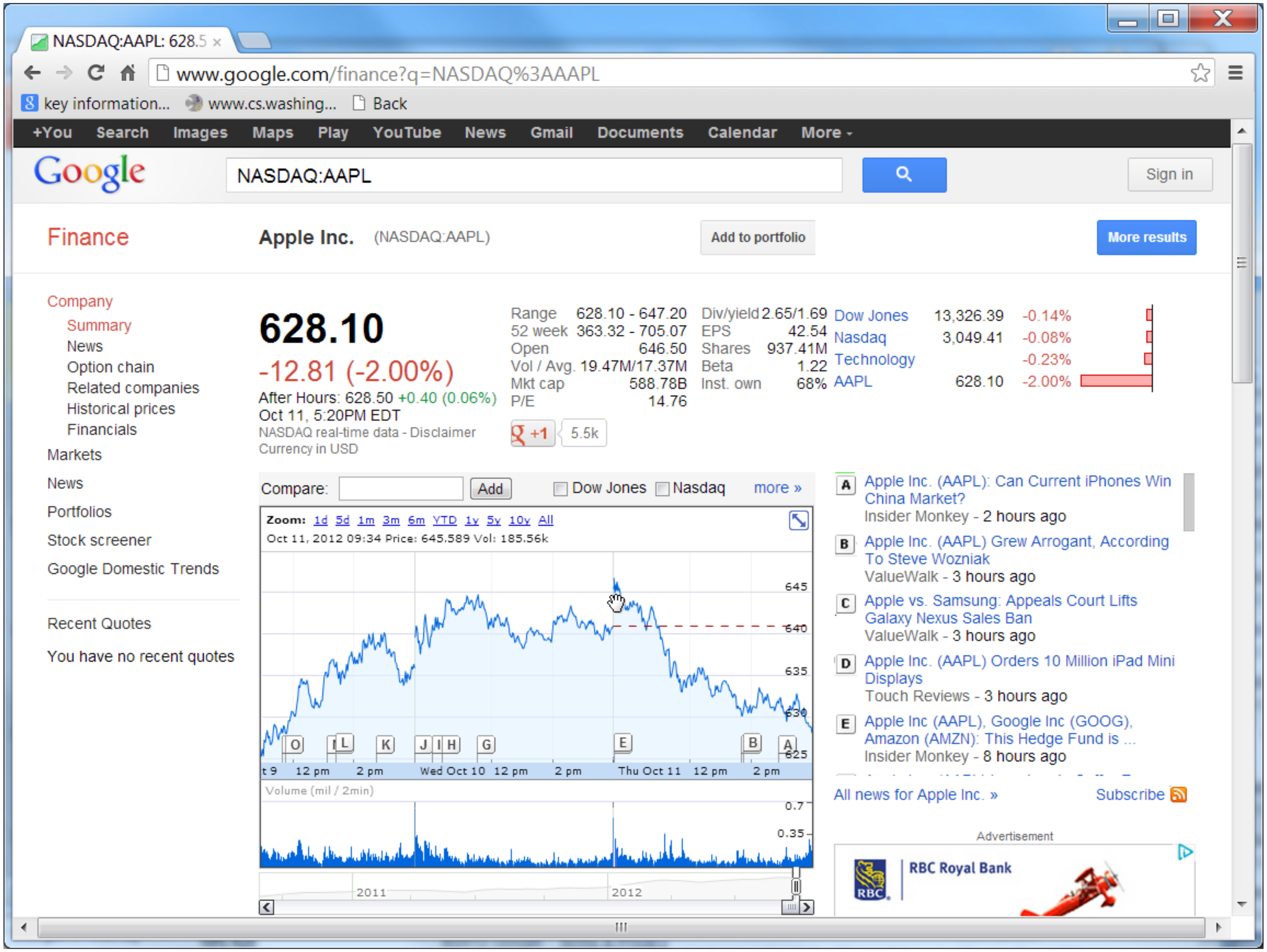
mus

music
music downl
music videos
musicians fri
music lyrics
mustang driv
mushi monst
muse

aja

ajax
ajax soccer club
ajax downs
ajax library
ajax cineplex
ajax weather
ajax community centre
ajax mazda
ajax hospital
ajax high school

Ajax example: live data updates



NASDAQ:AAPL: 628.5 ×

www.google.com/finance?q=NASDAQ%3AAAPL

key information... www.cs.washing... Back

+You Search Images Maps Play YouTube News Gmail Documents Calendar More

Google NASDAQ:AAPL Sign in

Finance **Apple Inc.** (NASDAQ:AAPL) Add to portfolio More results

Company
- Summary
- News
- Option chain
- Related companies
- Historical prices
- Financials

Markets

News

Portfolios

Stock screener

Google Domestic Trends

Recent Quotes

You have no recent quotes

628.10

-12.81 (-2.00%)

After Hours: 628.50 +0.40 (0.06%)
Oct 11, 5:20PM EDT
NASDAQ real-time data - Disclaimer
Currency in USD

Range	628.10 - 647.20	Div/yield	2.65/1.69
52 week	363.32 - 705.07	EPS	42.54
Open	646.50	Shares	937.41M
Vol / Avg.	19.47M/17.37M	Beta	1.22
Mkt cap	588.78B	Inst. own	68%
P/E	14.76		

+1 5.5k

Dow Jones	13,326.39	-0.14%
Nasdaq	3,049.41	-0.08%
Technology		-0.23%
AAPL	628.10	-2.00%

Compare: Add □ Dow Jones □ Nasdaq more »

Zoom: 1d 5d 1m 3m 6m YTD 1y 5y 10y All

Oct 11, 2012 09:34 Price: 645.589 Vol: 185.56k

A Apple Inc. (AAPL): Can Current iPhones Win China Market?
Insider Monkey - 2 hours ago

B Apple Inc. (AAPL) Grew Arrogant, According To Steve Wozniak
ValueWalk - 3 hours ago

C Apple vs. Samsung: Appeals Court Lifts Galaxy Nexus Sales Ban
ValueWalk - 3 hours ago

D Apple Inc. (AAPL) Orders 10 Million iPad Mini Displays
Touch Reviews - 3 hours ago

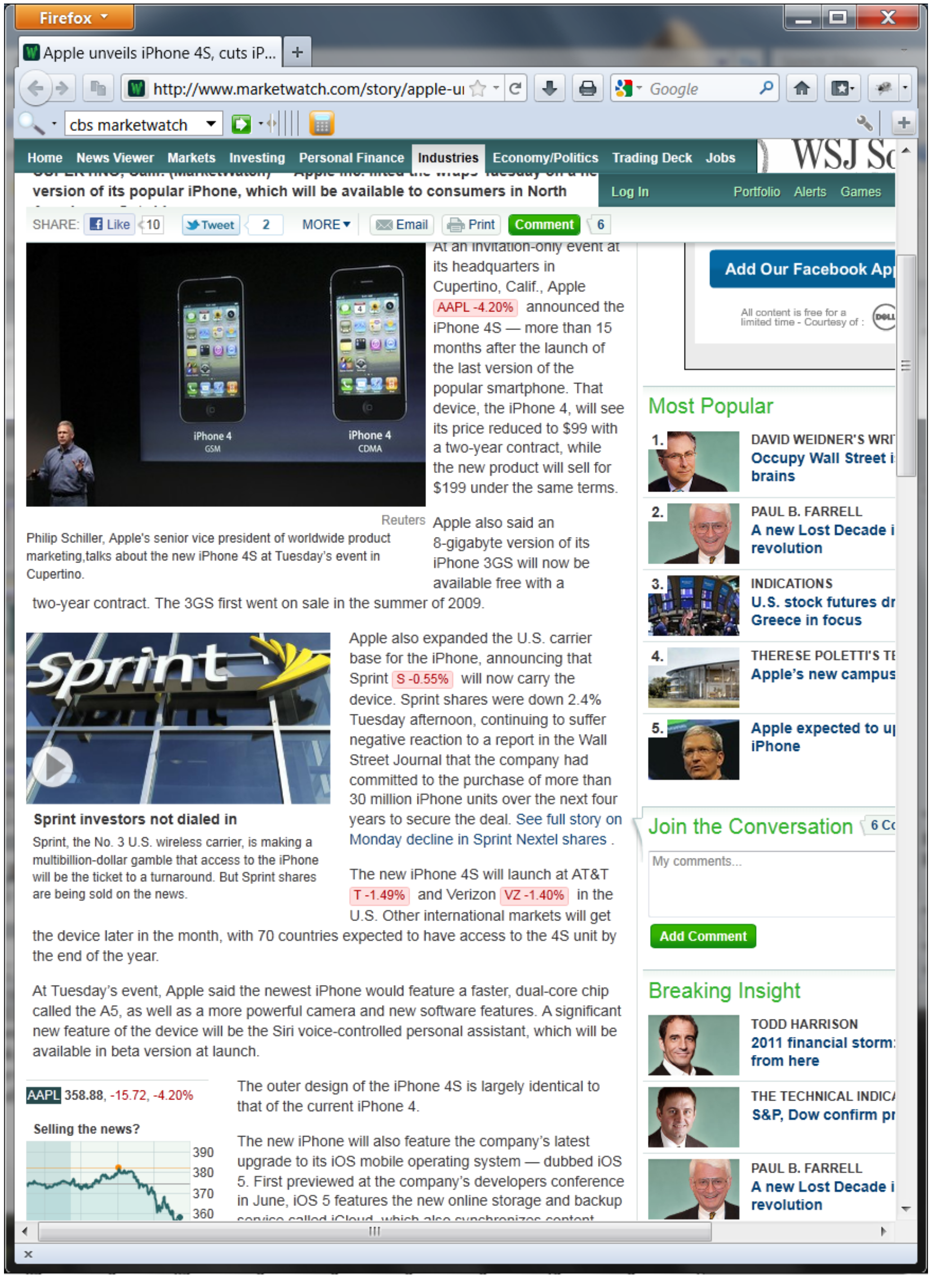
E Apple Inc (AAPL), Google Inc (GOOG), Amazon (AMZN): This Hedge Fund is ...
Insider Monkey - 8 hours ago

All news for Apple Inc. » Subscribe

Advertisement

RBC Royal Bank

Ajax example: live document updates



CUPERTINO, Calif. (MarketWatch) — Apple Inc. lifted the wraps Tuesday on a ne
version of its popular iPhone, which will be available to consumers in North

At an invitation-only event at its headquarters in Cupertino, Calif., Apple AAPL -4.20% announced the iPhone 4S — more than 15 months after the launch of the last version of the popular smartphone. That device, the iPhone 4, will see its price reduced to $99 with a two-year contract, while the new product will sell for $199 under the same terms.

Reuters

Philip Schiller, Apple's senior vice president of worldwide product marketing,talks about the new iPhone 4S at Tuesday's event in Cupertino.

Apple also said an 8-gigabyte version of its iPhone 3GS will now be available free with a two-year contract. The 3GS first went on sale in the summer of 2009.

Sprint investors not dialed in

Sprint, the No. 3 U.S. wireless carrier, is making a multibillion-dollar gamble that access to the iPhone will be the ticket to a turnaround. But Sprint shares are being sold on the news.

Apple also expanded the U.S. carrier base for the iPhone, announcing that Sprint S -0.55% will now carry the device. Sprint shares were down 2.4% Tuesday afternoon, continuing to suffer negative reaction to a report in the Wall Street Journal that the company had committed to the purchase of more than 30 million iPhone units over the next four years to secure the deal. See full story on Monday decline in Sprint Nextel shares .

The new iPhone 4S will launch at AT&T T -1.49% and Verizon VZ -1.40% in the U.S. Other international markets will get the device later in the month, with 70 countries expected to have access to the 4S unit by the end of the year.

At Tuesday's event, Apple said the newest iPhone would feature a faster, dual-core chip called the A5, as well as a more powerful camera and new software features. A significant new feature of the device will be the Siri voice-controlled personal assistant, which will be available in beta version at launch.

AAPL 358.88, -15.72, -4.20%

Selling the news?

The outer design of the iPhone 4S is largely identical to that of the current iPhone 4.

The new iPhone will also feature the company's latest upgrade to its iOS mobile operating system — dubbed iOS 5. First previewed at the company's developers conference in June, iOS 5 features the new online storage and backup

Most Popular

1. DAVID WEIDNER'S WRI
Occupy Wall Street i
brains

2. PAUL B. FARRELL
A new Lost Decade i
revolution

3. INDICATIONS
U.S. stock futures dr
Greece in focus

4. THERESE POLETTI'S TE
Apple's new campus

5. Apple expected to u
iPhone

Join the Conversation 6 Co

My comments...

Add Comment

Breaking Insight

TODD HARRISON
2011 financial storm:
from here

THE TECHNICAL INDIC
S&P, Dow confirm pr

PAUL B. FARRELL
A new Lost Decade i
revolution

Ajax Interaction

What's different with Ajax?

Traditional web

- user-initiated HTTP requests
  - typing on navigation window, or clicking on form or hyperlink
- response from server replaces existing page
- low request rate, and random amount of time between requests

Ajax application

- new type of request that does not trigger page reload
- not initiated by user (at least not directly)
- requests are typically small, but more frequent

Ajax and JSON

We will only use a few specific *jQuery Ajax* methods related to managing JSON data.

What is JSON data?

- JSON, JavaScript Object Notation, is a *string-based* representation of JavaScript array and dictionary objects
- A popular way to communicate structured data to (input) and from (output) Web services

JSON Data

- A JSON dictionary is written:

```
{
    "field0": "value0",
    "field1": "value1",
    ...
}
```

A JSON array is written:

```
[ value1, value2, … ]
```

- Values can be primitive types such as numbers and strings, and can also be arrays and dictionaries nested arbitrarily

JSON Data

- For example:

```
{
     "id": "cd1",
     "title": "Sgt. Pepper",
     "band": "The Beatles",
     "price": 22.00
}
```

- Note that JSON is strict about the presence of quotes around keys and values that are strings.
- A malformed JSON string will not be interpreted by JavaScript and will fail silently

JSON Data

JSON array of dictionaries:

```
   Music = [{"id": "cd1”, "title": "Sgt. Pepper”,
        "band": "The Beatles”, "price": 22.00},
        {"id": "cd2”, "title": ”Hey Jude”, "band":
        "The Beatles”, "price": 20.00},
        {"id": "cd3”, "title": ”Greatest Hits”,
        "band": "Neil Young”, "price": 21.00}]
```

Accessing the JSON data:

```
// Both ways return “Hey Jude”
Music[1].title


Music[1][“title”]
```

When do we use JSON Data?

When our webpage interacts with a database, we call a php file to read from the database.

The resulting data is returned as JSON data.

How do we do this?

PHP provides global functions for converting JSON string representation to/from a PHP (associative) array object:

```
json_decode(string)   # returns PHP object
json_encode(object)   # returns JSON string value
```

Outputting JSON Data

For example, suppose the php needs to return the following:

```
<?php
$json = array(
      "title" => "Sgt. Pepper",
      "band" => "The Beatles", "price" => 22.00);

header("Content-type: application/json");
print json_encode($json);
?>
```

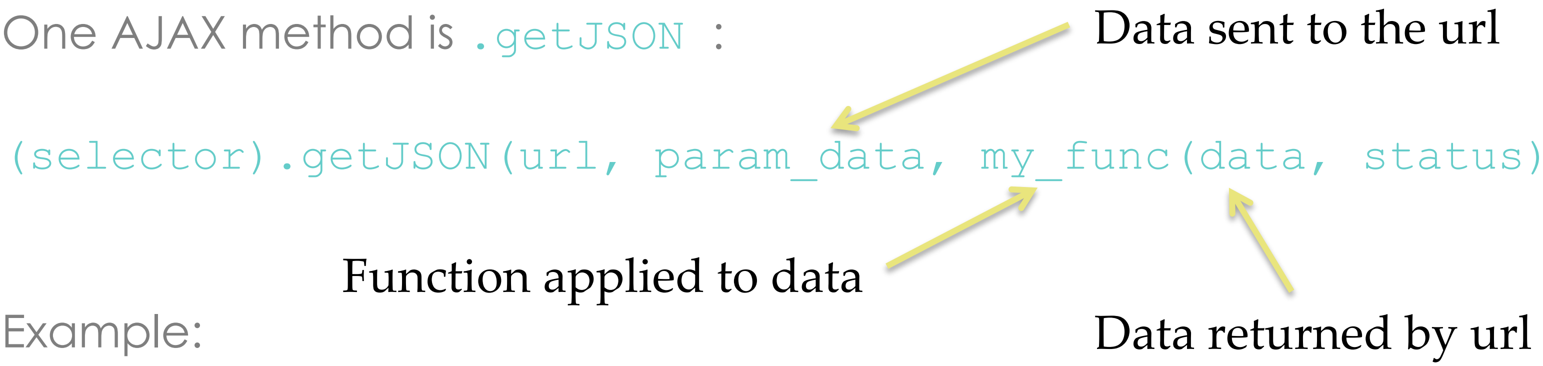
The `print` statement returns the JSON data.

In our case, the calling file will be a jQuery script.

jQuery AJAX

We will only use a few specific *jQuery Ajax* methods related to managing JSON data.

One AJAX method is `.getJSON` :



```
(selector).getJSON(url, param_data, my_func(data, status))
```

Data sent to the url

Function applied to data

Data returned by url

Example:

```
$(document).ready(function(){
    $("button").click(function(){
        $.getJSON("demo_ajax_json.js", function(result){
            $.each(result, function(i, field){
                $("div").append(field + " ");
            });
        });
    });
});
```

jQuery AJAX

What is `$.each`?

```
$(document).ready(function(){
    $("button").click(function(){
        $.getJSON("demo_ajax_json.js”,function(result){
            $.each(result, function(i, field){
                $("div").append(field + " ");
            });
        });
    });
});
```

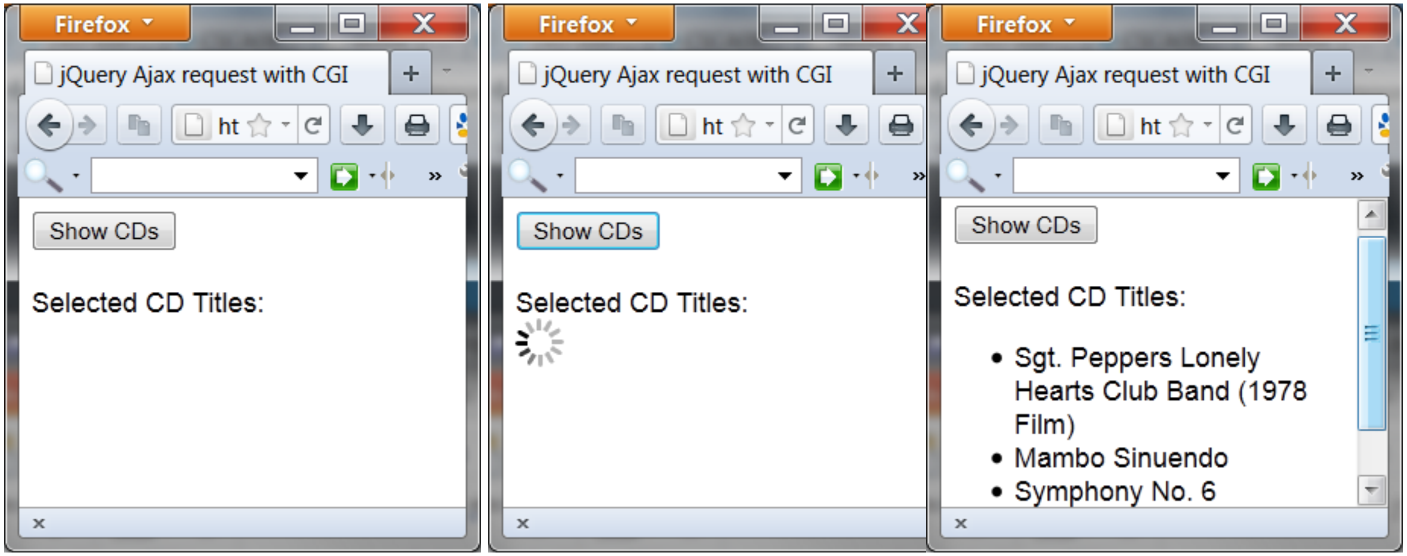
The `$.each(result, func())` method specifies a function to run for *each* element in `result`.

Another variation is:

```
        $(selector).each(function(index, element))
```

Where `function` is applied to each element matching the `selector`.

jQuery : events


Firefox
jQuery Ajax request with CGI
Show CDs
Firefox
jQuery Ajax request with CGI
Show CDs
Firefox
jQuery Ajax request with CGI
Show CDs

Selected CD Titles:

Selected CD Titles:

Selected CD Titles:

- Sgt. Peppers Lonely Hearts Club Band (1978 Film)
- Mambo Sinuendo
- Symphony No. 6

- Another task, possibly useful for assignment 3: provide the user with a button, that when clicked will initiate an Ajax request to a server-side PHP service to retrieve JSON data
- We use jQuery's getJSON() method to call the server, retrieve the JSON result, and define a "callback" function to handle this JSON.

jQuery Ajax callbacks

- A typical jQuery Ajax callback function binds the Ajax result value to the callback parameter, as in:

```
$.getJSON("… URL …", function(data) { … });
```

  - Within the anonymous callback function, `data` refers to the value returned by the getJSON() Ajax call.

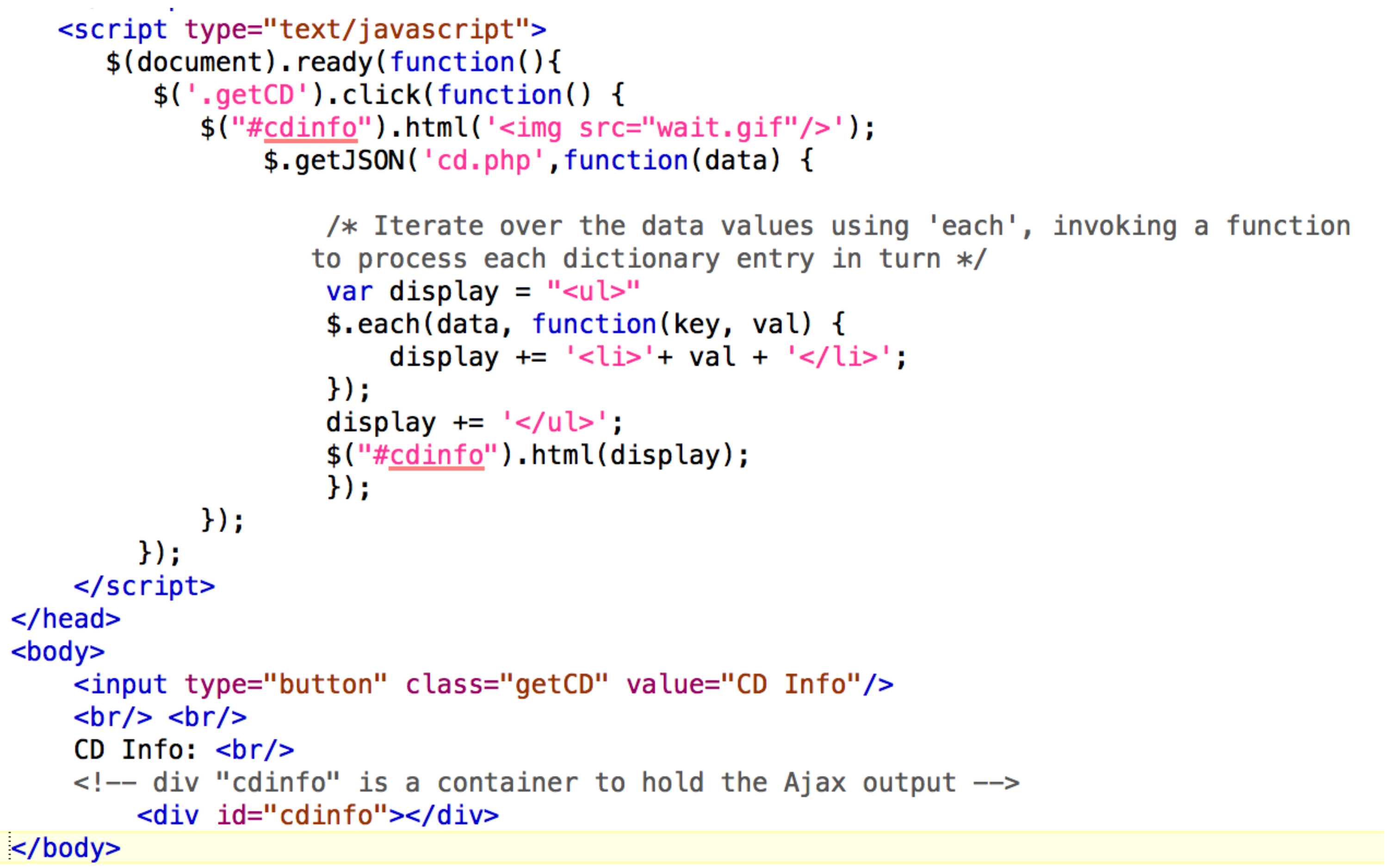
- If the Ajax result is a dictionary or array, you can iterate over the items in this dictionary/array using $.each():

```
$.each(data, function(key, value) { … });  // dict
$.each(data, function(index, value) { … });  // array
```

- Putting the parts together:

```
$.getJSON("… URL …", function(data) { ...
    $.each(data, function(index, value) {
    ... // index refers to current array index });
  ...
});
```

Using the JSON data



```
   <script type="text/javascript">
      $(document).ready(function(){
         $('.getCD').click(function() {
            $("#cdinfo").html('<img src="wait.gif"/>');
                $.getJSON('cd.php',function(data) {

                    /* Iterate over the data values using 'each', invoking a function
                   to process each dictionary entry in turn */
                    var display = "<ul>"
                    $.each(data, function(key, val) {
                        display += '<li>'+ val + '</li>';
                    });
                    display += '</ul>';
                    $("#cdinfo").html(display);
                    });
            });
        });
    </script>
</head>
<body>
    <input type="button" class="getCD" value="CD Info"/>
    <br/> <br/>
    CD Info: <br/>
    <!-- div "cdinfo" is a container to hold the Ajax output -->
        <div id="cdinfo"></div>
</body>
```

Example: DB Results as JSON

Using a database query to produce JSON results:

```
<?php
    $conn = mysqli_connect($dbhost, $dbname,
                           $dbuser, $dbpass);
    $stmt = mysqli_query($conn, "SELECT * FROM
                  Cities WHERE population > 9000000;");
   $cities = array();
   // create an array of all the results
   while ($row = mysqli_fetch_assoc($stmt)) {
        $cities[] = $row;

   $json = json_encode($cities);
    print $json;
?>
```

```
[ {"id":"206","name":Sao Paulo, "country_code":"BRA",
"district":null,"population":"9968485"},
{"id":"939","name":"Jakarta","country_code":"IDN","district
":"Jakarta Raya", "population":"9604900"}, …  ]
```

The End