



# AJAX and JSON with jQuery

---

Chen-Hsiang (Jones) Yu  
chyu@mit.edu

<http://chyu.scripts.mit.edu/ajax-json-jquery-examples.zip>

## The goal of this lecture

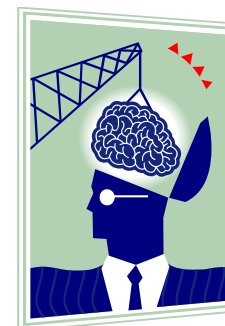
---

- Understand and know how to use AJAX
- Understand JSON
- Understand and know how to use jQuery
- Understand the usage of AJAX and JSON with jQuery

jQuery

JSON

AJAX





# Outline

---

- AJAX
- JSON
- jQuery



## Prerequisites

---

- MIT SIPB Web Script Service
  - <http://scripts.mit.edu/web/>
  - Instructions:
    - Using any terminal to connect to Athena
    - athena% add scripts
    - athena% signup-web
    - Your web space will be at [http://username.\*\*scripts\*\*.mit.edu/](http://username.<b>scripts</b>.mit.edu/)
    - You can use any FTP client to upload your files and scripts
- FTP client:
  - Filezilla (Windows/Linux/Mac)
    - <http://filezilla-project.org/download.php>
  - Alternative: OpenAFS - <http://www.openafs.org/>

# AJAX

---



# AJAX - 1

---

- It stands for “Asynchronous JavaScript and XML”
- It was first mentioned by Jesse James Garrett in 2005. [1]
- It is not a new technology. It is several technologies. [1]
  - standards-based presentation: [XHTML](#) and [CSS](#)
  - dynamic display and interaction: [DOM](#)
  - data interchange and manipulation: [XML](#) and [XSLT](#)
  - asynchronous data retrieval: [XMLHttpRequest](#)
  - binding everything together: [JavaScript](#)

## AJAX - 2

---

- Intuition: Static web site vs. AJAX site
- AJAX engine: XMLHttpRequest object
  - It allows for asynchronous communication
  - Instead of freezing up until the completeness, the browser can communicate with server and continue as normal.

# AJAX - 3

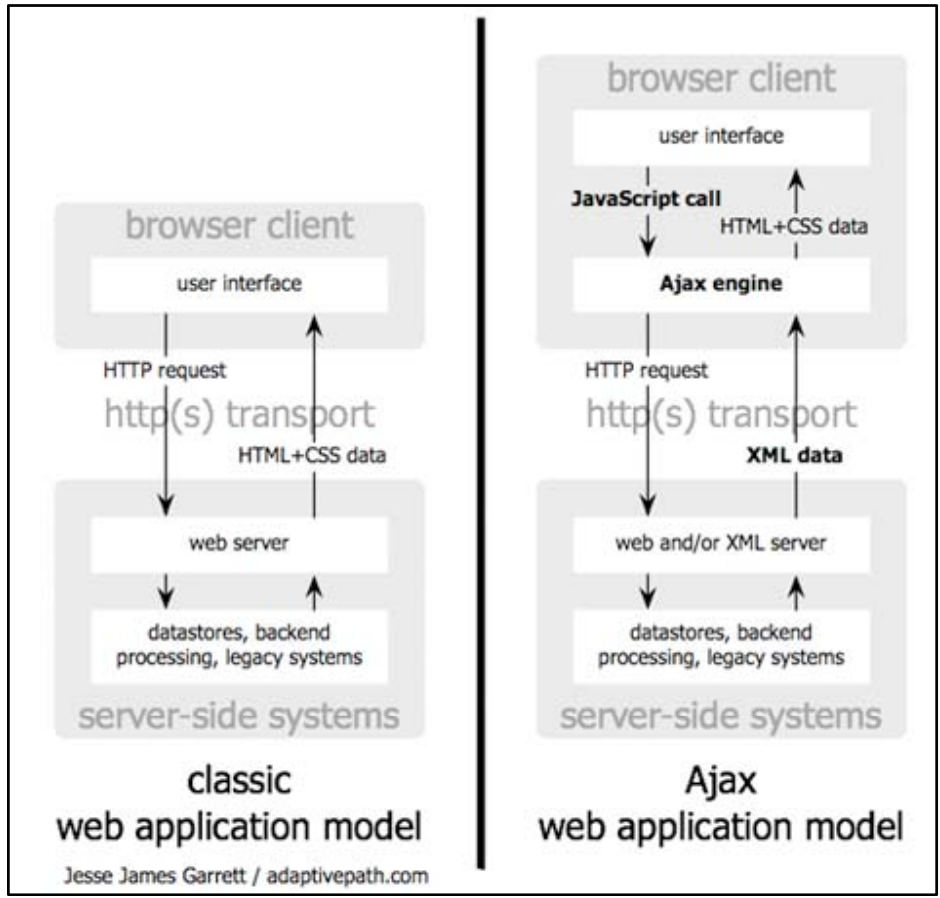


Figure: Traditional model vs. AJAX model [1]

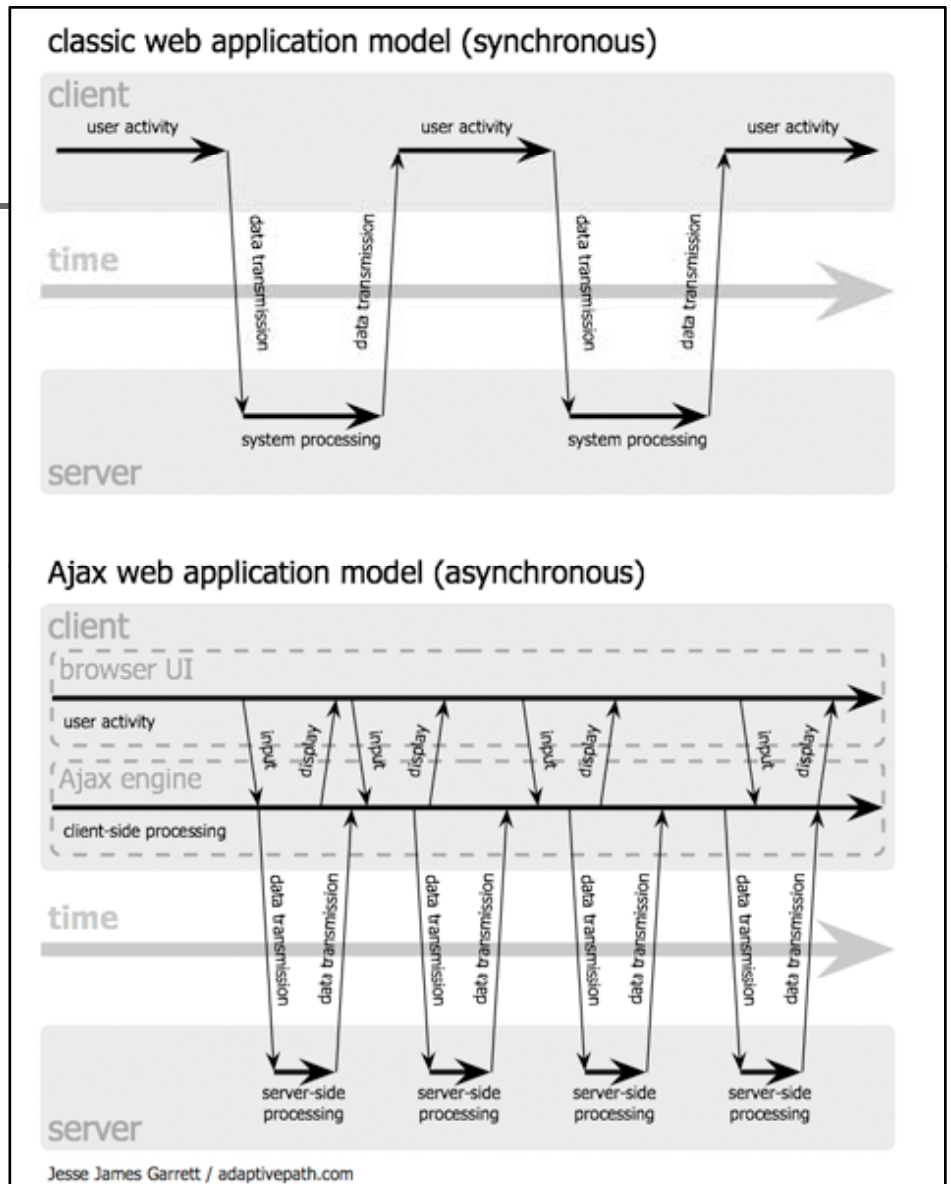


Figure: Synchronous interaction vs. Asynchronous interaction [1]



# AJAX - 4

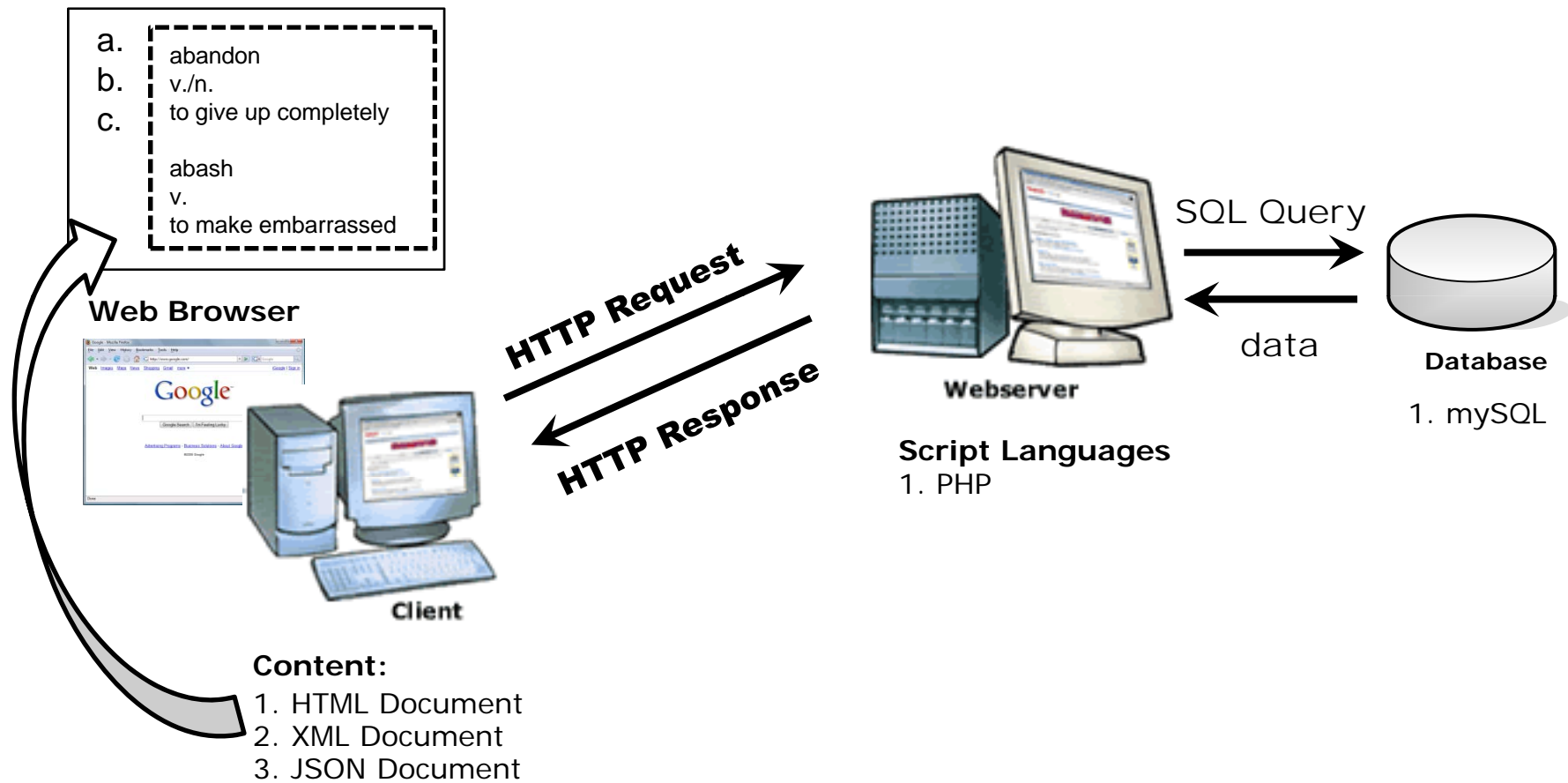


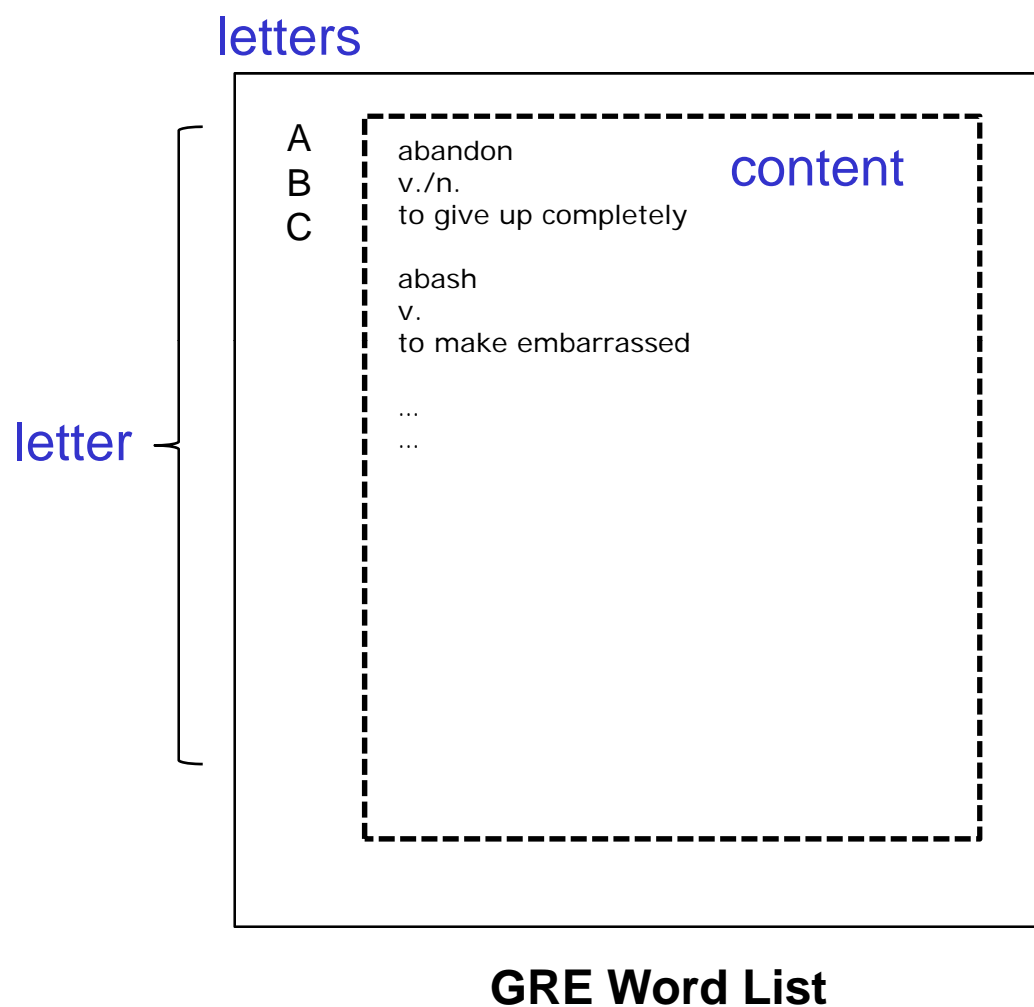
Figure: The process of AJAX call

## AJAX - 5

---

- POST and GET calls in AJAX [2]
  - GET places arguments in the query string, but POST doesn't.
  - No noticeable difference in AJAX - AJAX request does not appear in the address bar.
  - GET call in AJAX still has the size limitation on the amount of data that can be passed.
  - General principle:
    - GET method: it is used to retrieve data to display in the page and the data is not expected to be changed on the server.
    - POST method: it is used to update information on the server.

# Exercise: Example 1



```
<?php
print '<div class="entry">
    <h3 class="term">ABANDON</h3>
    <div class="part">v./n.</div>
    <div class="definition">
        to give up completely
    </div>
</div>

    <div class="entry">
    <h3 class="term">ABASH</h3>
    <div class="part">v.</div>
    <div class="definition">
        to make embarrassed
    </div>
    </div>
</div>
';
?>
```

example1.php

<http://chyu.scripts.mit.edu/example1.php>

```

<html>
<head>
<link rel="stylesheet" href="example1.css" type="text/css"/>
<script type="text/javascript">

var ajax;

function ajaxCreate(){
  if(window.XMLHttpRequest){
    //For IE7+, Firefox, Chrome, Opera, Safari
    return new XMLHttpRequest();
  } else {
    //For IE6, IE5
    return new ActiveXObject("Microsoft.XMLHTTP");
  }
}

function receiveData(){
  if(ajax.readyState == 4){
    if(ajax.status == 200){
      var content = document.getElementById('content');
      content.innerHTML = ajax.responseText;
    }else{
      alert("Server process error");
    }
  }
}
}

```

```

function sendRequest(url){
  ajax = ajaxCreate();

  if(!ajax){
    alert("Browser is not compatible with XMLHttpRequest");
    return 0;
  }

  ajax.onreadystatechange = receiveData;
  ajax.open("GET", url, true);
  ajax.send(null);
}

</script>
</head>

<body>
  <div class="letters">
    <div class="letter" id="letter-a">
      <h3>
        <a href="javascript:sendRequest('example1.php')">A</a>
      </h3>
    </div>
  </div>
  <div id="content"></div>
</body>
</html>

```

example1.html

http://chyu.scripts.mit.edu/example1.html

# AJAX - 5



| Properties         | Description [3]  |   |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
|--------------------|--|---|-------|-------------|---|---------------|---------------------------------|---|---------|---------------------------------|---|--------|---|---|-------------|---|---|-----------|----------------------------|
| onreadystatechange | A JavaScript function object that is called whenever the readyState attribute changes. The callback is called from the user interface thread.  |   |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| readyState         | Returns values that indicate the current state of the object. <table border="1"><thead><tr><th>Value</th><th>State</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>UNINITIALIZED</td><td>open() has not been called yet.</td></tr><tr><td>1</td><td>LOADING</td><td>send() has not been called yet.</td></tr><tr><td>2</td><td>LOADED</td><td>send() has been called, and headers and status are available.</td></tr><tr><td>3</td><td>INTERACTIVE</td><td>Downloading; responseText holds partial data.</td></tr><tr><td>4</td><td>COMPLETED</td><td>The operation is complete.</td></tr></tbody></table> | Value   | State | Description | 0 | UNINITIALIZED | open() has not been called yet. | 1 | LOADING | send() has not been called yet. | 2 | LOADED | send() has been called, and headers and status are available. | 3 | INTERACTIVE | Downloading; responseText holds partial data. | 4 | COMPLETED | The operation is complete. |
| Value              | State  | Description   |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| 0                  | UNINITIALIZED  | open() has not been called yet.                               |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| 1                  | LOADING  | send() has not been called yet.                               |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| 2                  | LOADED   | send() has been called, and headers and status are available. |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| 3                  | INTERACTIVE  | Downloading; responseText holds partial data.                 |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| 4                  | COMPLETED  | The operation is complete.                                    |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| responseText       | The response to the request as text, or null if the request was unsuccessful or has not yet been sent.   |   |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| responseXML        | The response to the request as a DOM Document object, or null if the request was unsuccessful, has not yet been sent, or cannot be parsed as XML. The response is parsed as if it were a text/xml stream.  |   |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| status             | The status of the response to the request. This is the HTTP result code (for example, status is 200 for a successful request).   |   |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |
| statusText         | The response string returned by the HTTP server. Unlike status, this includes the entire text of the response message ("200 OK", for example).   |   |       |             |   |               |                                 |   |         |                                 |   |        |   |   |             |   |   |           |                            |

# AJAX - 6



| Methods   | Description [3]   |
|---|---|
| <code>Abort()</code>  | Aborts the request if it has already been sent.   |
| <code>getAllResponseHeaders()</code>  | Returns all the response headers as a string..  |
| <code>getResponseHeader("headerLabel")</code>                                       | Returns the text of a specified header.   |
| <code>open("method", "URL"[,<br/>asyncFlag[, "userName"[,<br/>"password"]]])</code> | Initializes a request. This method is to be used from JavaScript code; to initialize a request from native code, use <code>openRequest()</code> instead.  |
| <code>send(content)</code>  | Sends the request. If the request is asynchronous (which is the default), this method returns as soon as the request is sent. If the request is synchronous, this method doesn't return until the response has arrived. |
| <code>setRequestHeader("label", "value")</code>                                     | Sets the value of an HTTP request header.   |

A

## GRE Word List

### **ABANDON**

v./n.  
to give up completely

---

### **ABASH**

v.  
to make embarrassed

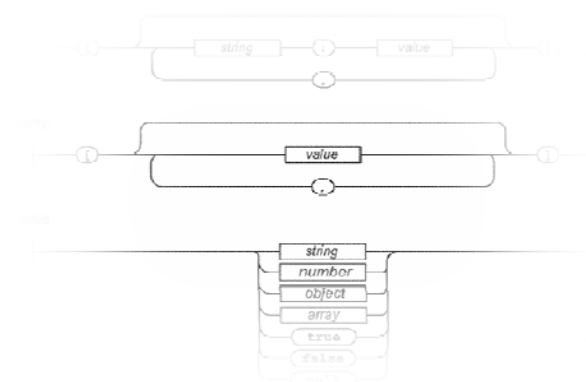
---

Organized by TIM.



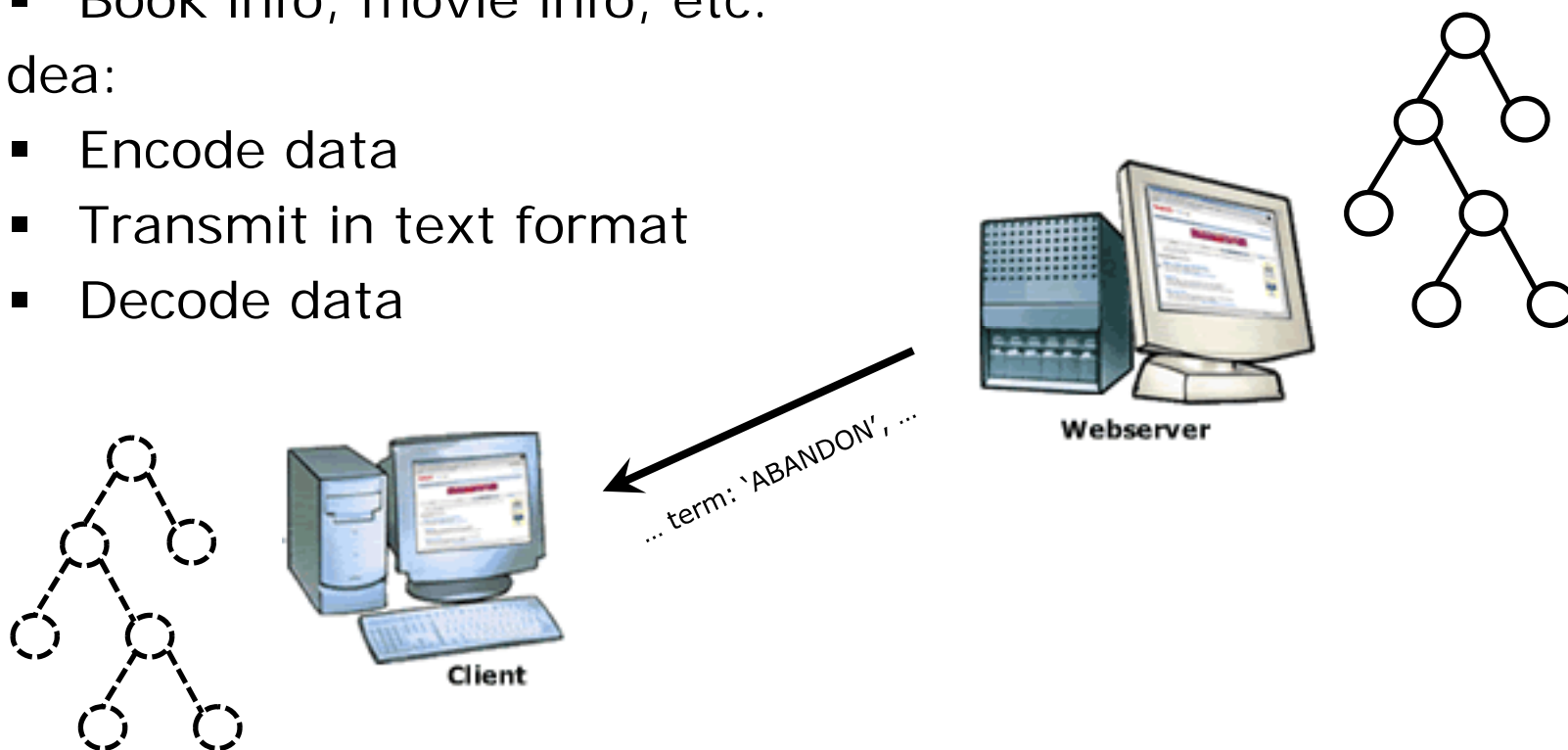
# JSON

---



# JSON - 1

- Text is a data transmission format on the Internet
- A need to transmit complicated (structured) data
  - Book info, movie info, etc.
- Idea:
  - Encode data
  - Transmit in text format
  - Decode data



## JSON - 2

---

- It stands for “JavaScript Object Notation” and it is a text-based data interchange format. [4]
- JSON is built on two structures:
  - Object: A collection of name and value pairs
  - Array: An ordered list of values

Figure: The diagrams of the syntax of JSON [3]

| Explanation                                       | Examples  |
|---|---|
| object with one member                            | <pre>{   "course-name": "6.470" }</pre>   |
| object with two members<br>(separated by a comma) | <pre>{   "course-name": "6.470",   "units": 6 }</pre>   |
| array with 3 elements                             | <pre>["foo", "bar", 12345]</pre>  |
| object with an array as a value                   | <pre>{   "course-name": "6.470",   "times": ["12-4", "12-4", "12-4"] }</pre>  |
| objects with objects and arrays                   | <pre>{   "John":     {       "label": "John",       "data": [[1880,0.081536],[1881,0.08098],[1882,0.078308]]     },   "James":     {       "label": "James",       "data": [[1880,0.050053],[1881,0.05025],[1882,0.048278]]     } }</pre> |

### Concrete Examples [5]

# JSON - 3

- Examples:

```
{
  "course-name" : "6.470"
}
```

```
["foo", "bar", 12345]
```

```
{
  "course-name" : "6.470",
  "times": ["12-4", "12-4", "12-4"]
}
```

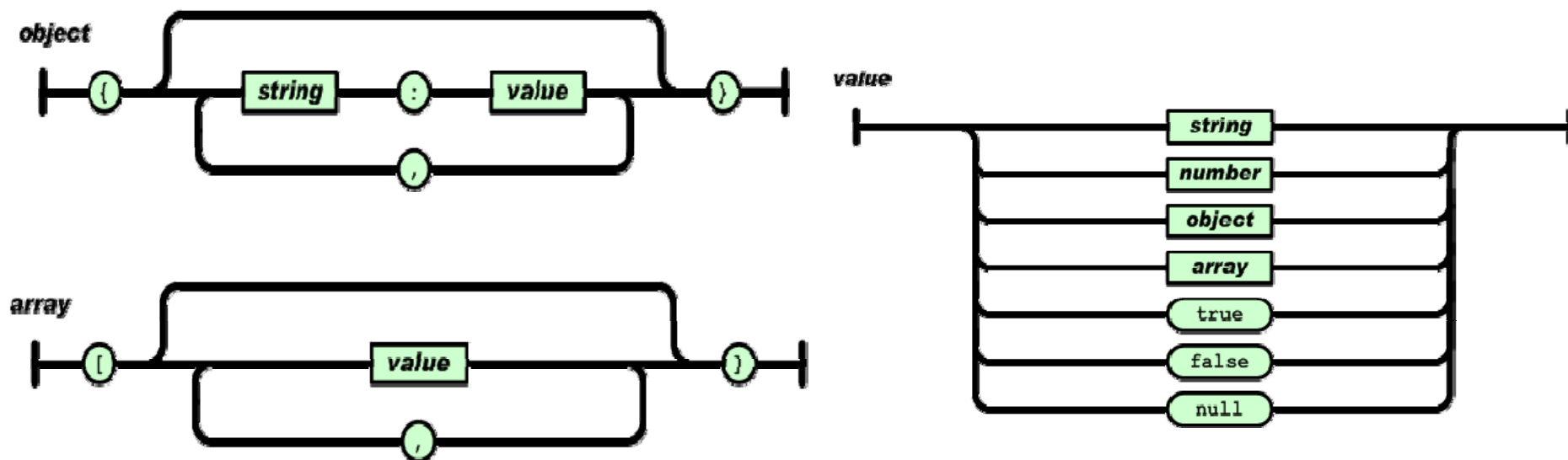


Figure: The diagrams of the syntax of JSON [4]

# JSON - 4

"data": [[1880,0.081536],[1881,0.08098],[1882,0.078308]]

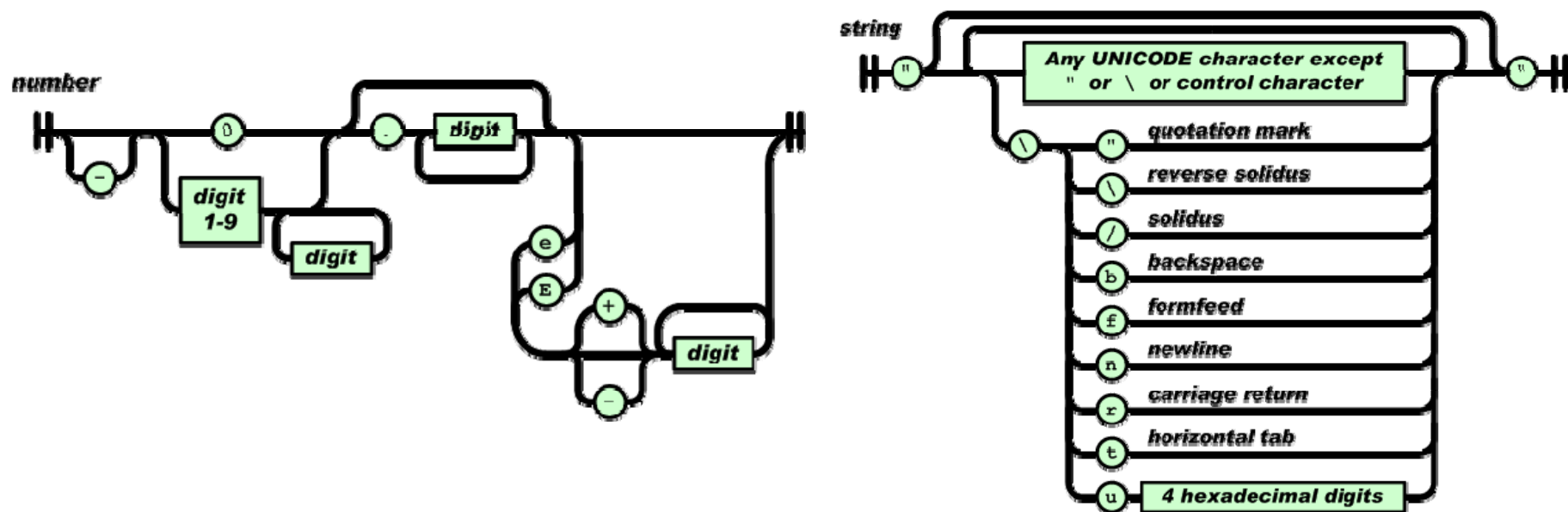


Figure: The diagrams of the syntax of JSON [4]

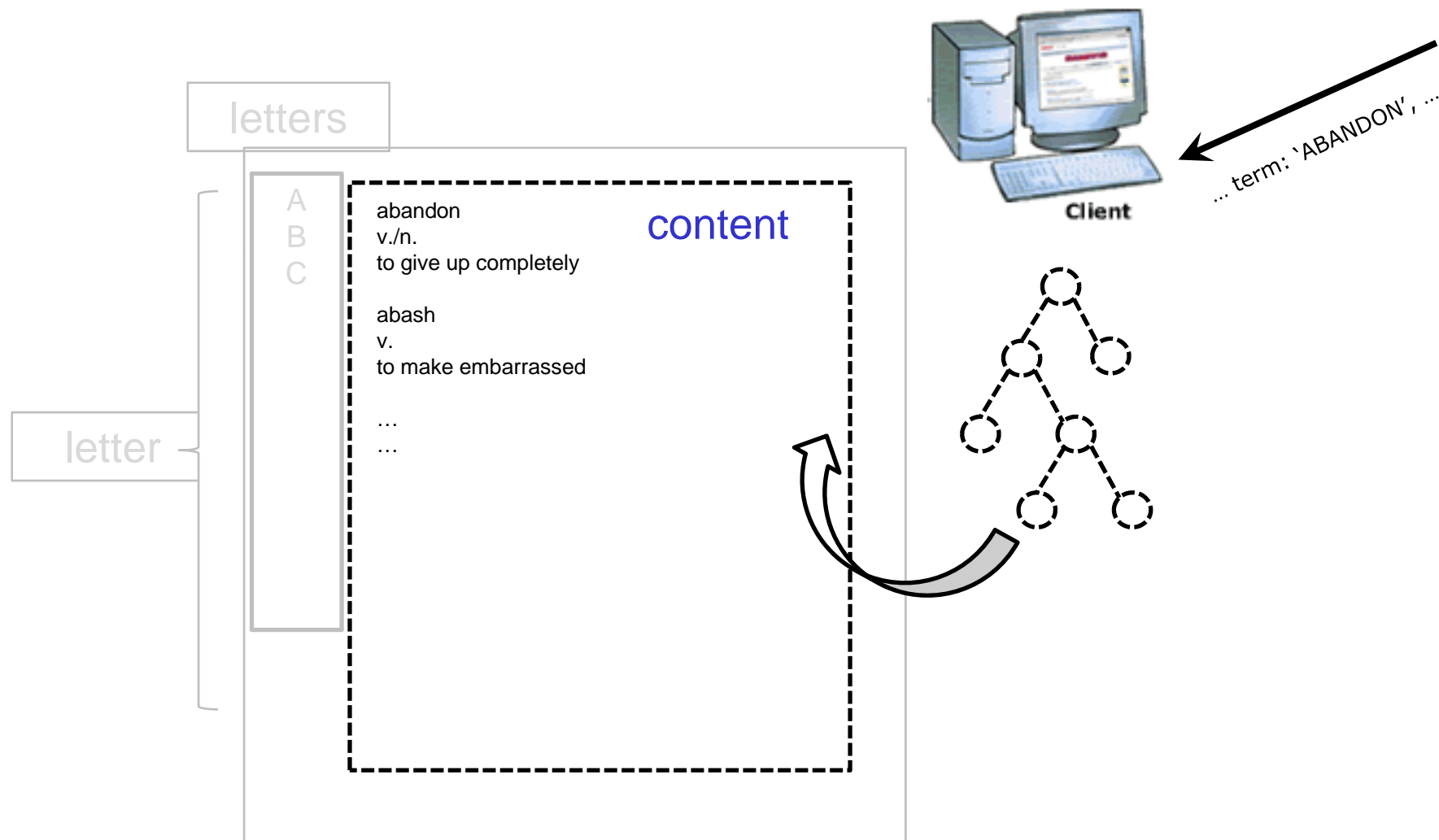
## JSON - 5

- Why JSON format is easier to be evaluated?

```
var a = {  
  "John":  
    {  
      "label": "John",  
      "data": [[1880, 0.081536], [1881, 0.08098], [1882, 0.078308]]  
    },  
  "James":  
    {  
      "label": "James",  
      "data": [[1880, 0.050053], [1881, 0.05025], [1882, 0.048278]]  
    }  
}
```

- To get the value 1880 marked with red color:
  - We can use `a.John.data[0][0]`

# Exercise: Example 2





## Example 2: Extending Example 1

```

<script type="text/javascript">
...
function receiveData(){
  if(ajax.readyState == 4){
    if(ajax.status == 200){
      var data = [
        { "word": "ABANDON",
          "function": "v./n.",
          "definition": "to give up completely"},
        { "word": "ABASH",
          "function": "v.",
          "definition": "to make embarrassed"}
      ];

      var content = document.getElementById('content');
      content.innerHTML = data;
    }else{
      alert("Server process error");
    }
  }
}
...
</script>

```

# What is the problem?



example2.html

<http://chyu.scripts.mit.edu/example2.html>

[A](#)

## GRE Word List

[object Object],[object Object]

Organized by TIM.

# What is the problem?



```
<script type="text/javascript">
...
function receiveData(){
  if(ajax.readyState == 4){
    if(ajax.status == 200){
      var data = [
        { "word": "ABANDON",
          "function": "v./n.",
          "definition": "to give up completely"},
        { "word": "ABASH",
          "function": "v.",
          "definition": "to make embarrassed"}
      ];

      var output = "";
      for(each in data){
        output += '<div class="entry">';
        output += '<h3 class="word">' + data[each].word + '</h3>';
        output += '<div class="function">' + data[each].function + '</div>';
        output += '<div class="definition">' + data[each].definition + '</div>';
        output += '</div>';
      }

      var content = document.getElementById('content');
      content.innerHTML = output;
    }else{
      alert("Server process error");
    }
  }
}
}
}
...
</script>
```

example2-sol.html

<http://chyu.scripts.mit.edu/example2-sol.html>

A

## GRE Word List

### **ABANDON**

v./n.  
to give up completely

---

### **ABASH**

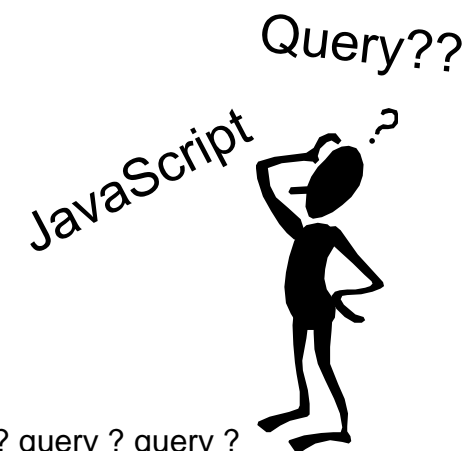
v.  
to make embarrassed

---

Organized by TIM.

# jQuery

---



query ? query ? query ? query ? query ? query ? query ? query ? query ? query ? query ? query ? query ?

# jQuery - 1

---

- JavaScript libraries and frameworks
  - Prototype, jQuery, Dojo, Echo, GWT, MooTools, Rico, YUI, etc.
- Advantages
  - Providing common utilities
  - Handling cross-platform issues
  - (Might) provide extendibility - plugins
  - Faster Web development





## jQuery - 2

---

- jQuery: A JavaScript library was released by John Resig in 2006
- Idea: Putting together a set of functions
  - Navigate a document
  - Event handling
  - Select DOM elements
  - Create animations
  - Ajax development
  - Extendibility: plugins
- It is used at 21% of the 10000 most popular websites. [6]



## jQuery - 3

---

- Usage:
  - Step 1: Download the latest jQuery from [jquery.com](http://jquery.com)
  - Step 2: Link the downloaded .js file
    - `<script type="text/javascript" src="jquery-1.3.2.js" ></script>`
    - Note: It adds jQuery() function in the global namespace and the function is also aliased as \$().
  - Step 3: Write your code



## jQuery - 4

---

- APIs:
  - jQuery core
  - Selectors
  - Attributes
  - Traversing
  - Manipulation
  - CSS
  - Events
  - Effects
  - Ajax
  - Utilities
  - jQuery UI

## jQuery - 5

---

- APIs:
  - jQuery core
  - Selectors
  - Attributes
  - Traversing
  - Manipulation
  - CSS
  - Events
  - Effects
  - Ajax
  - Utilities
  - jQuery UI

## jQuery - 6

---

- Core:
  - jQuery functions:
    - `$(expression, [context])`
      - ❑ `$('input:radio', document.form[0])`
    - `$(html)`
      - ❑  `$('<div id="loading">Loading...</div>')`
    - `$(elements)`
      - ❑ `$(document.body).css('background', 'red');`
    - `$(callback)`                      The same as `$(document).ready()`
      - ❑ `$(function(){alert("test");})`

# jQuery - 7

---

- Selectors:
  - No matter which type of selectors, we start with `$()`.
  - jQuery supports nearly all selectors in CSS 1 through 3.
  - Types:
    - Basic:
      - ❑ element, #id, class, .classA.classB
      - ❑ Ex: `$('p')`, `$('#id')`, `$('.class')`, `$('.classA.classB')`
    - Hierarchy:
      - ❑ ancestor decendent, parent > child, prev + next
      - ❑ Ex: `$('form input')`, `$('#main > *')`, `$('label + input')`

# jQuery - 8

- o Form:

| Selectors   | Matched Elements  |
|---|---|
| :input  | input, select, textarea and button elements                           |
| :text, :radio, :checkbox, :image, :submit, :reset, :password, :file | input element with attribute that is equal to the specified selectors |
| :button   | button element, input element with type is "button"                   |

- o Basic filters:

- ❑ :first , :last , :not(selector) , :even , :odd , :eq(index) , :gt(index) , :lt(index) , :header , :animated

- o Attribute filters:

- ❑ [attribute] , [attribute!=value] , [attribute^=value] , [attribute\$=value] , [attribute\*=value] , [filter1][filter2]

## jQuery - 9

---

- Attributes:
  - Attr:
    - `attr(name)`, `attr(properties)` `attr(key,value)`, `removeAttr(name)`
  - Class:
    - `addClass(class)`, `removeClass(class)`, `toggleClass(class)`
  - HTML:
    - `html()`, `html(value)`
  - Text:
    - `text()`, `text(value)`
  - Value:
    - `val()`, `val(value)`

## jQuery - 10

---

- Events:
  - Page load:
    - `ready(fn)`
  - Event handling:
    - `bind(type, fn)`, `unbind(type, fn)`, `trigger(event)`
  - Event helpers:
    - `click()`, `click(fn)`, `mousedown(fn)`, `mouseout(fn)`, ...
- Effects:
  - Basics:
    - `show()`, `show(speed)`, `hide()`, `toggle()`, `toggle(speed)`
  - Fading:
    - `fadeIn(speed)`, `fadeOut(speed)`, `fadeTo(speed, opacity)`

## Exercise: Example 3

content

A SCANDAL IN BOHEMIA                      F F F

I.

To Sherlock Holmes she is always THE woman. I have seldom heard him mention her under any other name. In his eyes she eclipses and predominates the whole of her sex. It was not that he felt any emotion akin to love for Irene Adler. All emotions, and that one particularly, were abhorrent to his cold.....

II.

At three o'clock precisely I was at Baker Street, but Holmes had not yet returned. The landlady informed me that he had left the house shortly after eight o'clock in the morning. ..

- Please use `example3.html`
- Requirement:
  - Using jQuery selectors, attributes, events
  - When clicking font symbols (small, medium, large), the content of paragraphs can change properly.
  - Set default font size of the content of paragraphs as “medium”.

`example3.html`

<http://chyu.scripts.mit.edu/example3.html>





## Exercise: Example 3 - reference solution

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

$(document).ready(function(){
  $('#large').bind('click', function(){
    $('p').removeClass().addClass('large');
  });

  $('#medium').bind('click', function(){
    $('p').removeClass().addClass('medium');
  });

  $('#small').bind('click', function(){
    $('p').removeClass().addClass('small');
  });

  // Set default font size as medium
  $('#medium').trigger('click');
});

</script>
```

example3-sol.html

<http://chyu.scripts.mit.edu/example3-sol.html>

## Exercise: Example 3

F F F

### A SCANDAL IN BOHEMIA

#### I.

To Sherlock Holmes she is always THE woman. I have seldom heard him mention her under any other name. In his eyes she eclipses and predominates the whole of her sex. It was not that he felt any emotion akin to love for Irene Adler. All emotions, and that one particularly, were abhorrent to his cold, precise but admirably balanced mind. He was, I take it, the most perfect reasoning and observing machine that the world has seen, but as a lover he would have placed himself in a false position. He never spoke of the softer passions, save with a gibe and a sneer....

#### II.

At three o'clock precisely I was at Baker Street, but Holmes had not yet returned. The landlady informed me that he had left the house shortly after eight o'clock in the morning. I sat down beside the fire, however, with the intention of awaiting him, however long he might be. I was already deeply interested in his inquiry, for, though it was surrounded by none of the grim and strange features which were associated with the two crimes which I have already recorded, still, the nature of the case and the exalted station of his client gave it a character of its own....

The content was from [7].

F F F

### A SCANDAL IN BOHEMIA

#### I.

To Sherlock Holmes she is always THE woman. I have seldom heard him mention her under any other name. In his eyes she eclipses and predominates the whole of her sex. It was not that he felt any emotion akin to love for Irene Adler. All emotions, and that one particularly, were abhorrent to his cold, precise but admirably balanced mind. He was, I take it, the most perfect reasoning and observing machine that the world has seen, but as a lover he would have placed himself in a false position. He never spoke of the softer passions, save with a gibe and a sneer....

#### II.

At three o'clock precisely I was at Baker Street, but Holmes had not yet returned. The landlady informed me that he had left the house shortly after eight o'clock in the morning. I sat down beside the fire, however, with the intention of awaiting him, however long he might be. I was already deeply interested in his inquiry, for, though it was surrounded by none of the grim and strange features which were associated with the two crimes which I have already recorded, still, the nature of the case and the exalted station of his client gave it a character of its own....

## jQuery - 11

- Ajax:
  - Ajax Request:
    - `$.ajax(options)` => options: a set of key/value pairs
    - `$.get(url, [data], [callback], [type])`
    - `$.post(url, [data], [callback], [type])`
    - `$.getJSON(url, [data], [callback], [type])`
    - `$.getScript(url, [callback])`

```
$.ajax(  
  {  
    url: "process.php",  
    type: "POST",  
    data: "class=6470&name=Tim",  
    success: function(msg){  
      alert("Data:" + msg );  
    }  
  }  
);
```

```
$.post(  
  "test.php",  
  {  
    func: "getNameAndTime"  
  },  
  function(data){  
    alert(data.name);  
    alert(data.time);  
  },  
  "json"  
);
```



## jQuery - 12

---

```
$.getJSON(
  "http://api.flickr.com/services/feeds/photos_public.gne?tags=doggy&format=json&jsoncallback=?",
  function(data){
    $.each(data.items, function(i,item){
      $("<img/>").attr("src", item.media.m).appendTo("#images");
      if ( i == 2 ) return false;
    }
  );
});
```

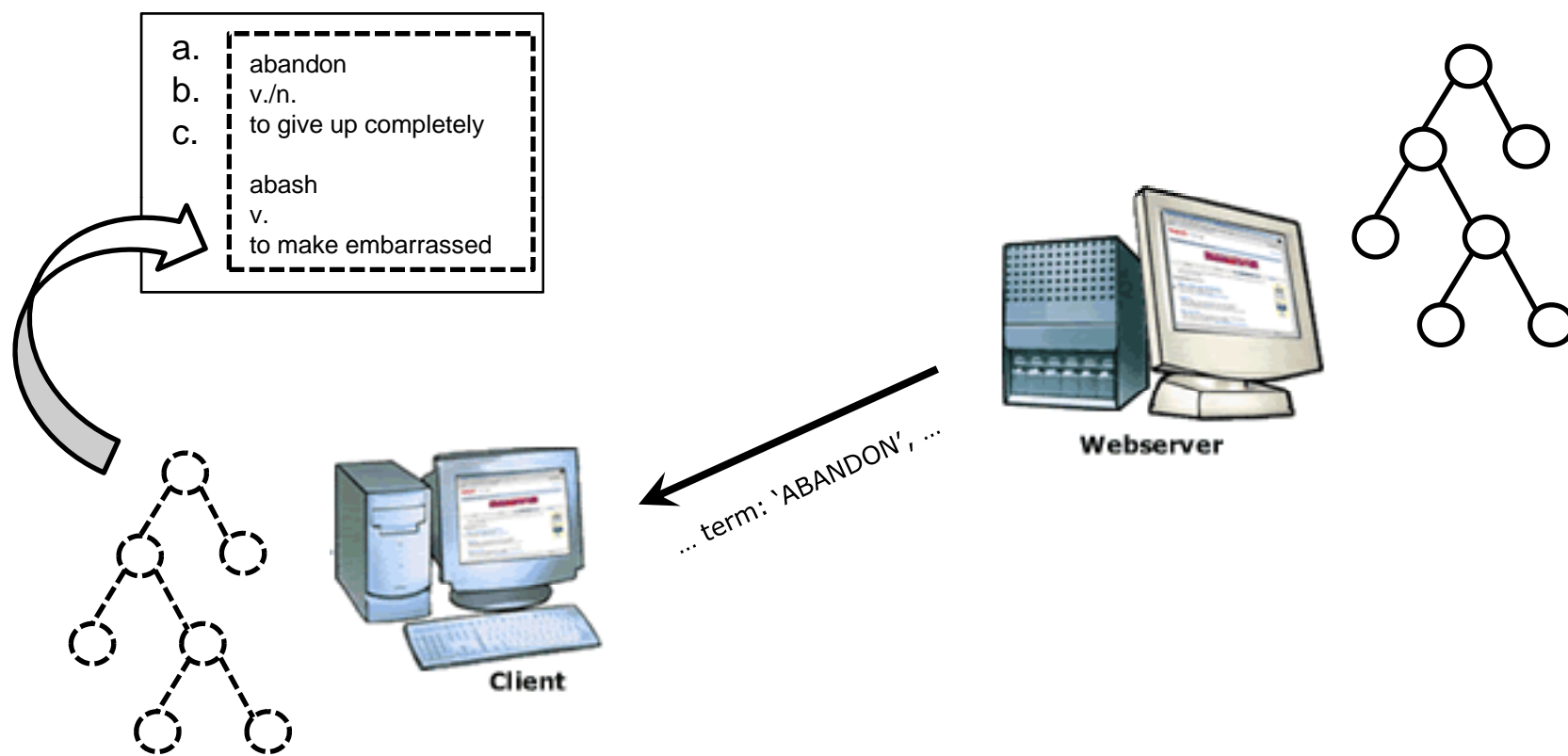
## jQuery - 13

---

- Ajax Events:
  - `ajaxComplete(callback)`, `ajaxStart(callback)`, `ajaxStop(callback)`, `ajaxSend(callback)`, `ajaxError(callback)`, `ajaxSuccess(callback)`
  - Ex:

```
$('#<div id="loading">Loading...</div>')  
  .insertBefore("#images")  
  .ajaxStart(  
    function(){  
      $(this).show();  
    }  
  ).ajaxStop(  
    function(){  
      $(this).hide();  
    }  
  );
```

# Exercise: Example 4



## Exercise: Example 4

```
<?php
$data = '
[
  {
    "word": "ABANDON",
    "function": "v./n.",
    "definition": "to give up completely"
  },
  {
    "word": "ABASH",
    "function": "v.",
    "definition": "to make embarrassed"
  },
  {
    "word": "ABATE",
    "function": "v.",
    "definition": "to make less in amount; wane"
  },
  {
    "word": "ABBREVIATE",
    "function": "v.",
    "definition": "to make shorter; to shorten a word or phrase"
  }
]
';
print($_GET['callback'].'(.'.$data.')');
?>
```

example4.php

<http://chyu.scripts.mit.edu/example4.php>

# Example 4: Using jQuery for example 1

Part 1

```

...
<script type="text/javascript">
  $(document).ready(function(){
    var url="example4.php";

    $('#letter-a').click(function(){

      $.getJSON(
        url + '?callback=?',
        function(data){
          $('#content').empty();
          $.each(data, function(index, entry){
            var html = '<div class="entry">';
            html += '<h3 class="word">' + entry['word'] + '</h3>';
            html += '<div class="function">' + entry['function'] + '</div>';
            html += '<div class="definition">' + entry['definition'] + '</div>';
            html += '</div>';
            $('#content').fadeOut().append(html);
          });
        });

      return false;
    });
  });
</script>
...

```

1<sup>st</sup> parameter ←

2<sup>nd</sup> parameter {

example4.html

<http://chyu.scripts.mit.edu/example4.html>



## Example 4: Using jQuery for example 1

```
...
<script type="text/javascript">
  $(document).ready(function(){
    ...

    // Slide up or down the function and definition of the word
    $('.word').live('click', function(){
      $(this).siblings('.function').slideToggle().siblings('.definition').slideToggle();
    });

    $('#showDetails').click(function(){
      $('.word').siblings('.function').show('slow').siblings('.definition').show('slow');
    });

    $('#hideDetails').click(function(){
      $('.word').siblings('.function').hide('slow').siblings('.definition').hide('slow');
    });
  });
</script>
...
```

**Part 2**

**Chaining the effects**

example4.html

<http://chyu.scripts.mit.edu/example4.html>

## Example 4: Using jQuery for example 1

### Complete Version

```

...
<script type="text/javascript">
  $(document).ready(function(){
    var url="example4.php";
    $('#letter-a').click(function(){
      $.getJSON(url + '?callback=?', function(data){
        $('#content').empty();
        $.each(data, function(index, entry){
          var html = '<div class="entry">';
          html += '<h3 class="word">' + entry['word'] + '</h3>';
          html += '<div class="function">' + entry['function'] + '</div>';
          html += '<div class="definition">' + entry['definition'] + '</div>';
          html += '</div>';
          $('#content').fadeIn().append(html);
        });
      });
      return false;
    });

    // Slide up or down the function and definition of the word
    $('.word').live('click', function(){
      $(this).siblings('.function').slideToggle().siblings('.definition').slideToggle();
    });

    $('#showDetails').click(function(){
      $('.word').siblings('.function').show('slow').siblings('.definition').show('slow');
    });

    $('#hideDetails').click(function(){
      $('.word').siblings('.function').hide('slow').siblings('.definition').hide('slow');
    });
  });
</script>
...

```

example4.html

<http://chyu.scripts.mit.edu/example4.html>

**A** GRE Word List

**B**

**C** **ABANDON**  
v./n.  
to give up completely

---

**E** **ABASH**  
v.  
to make embarrassed

---


**F** **ABATE**  
v.  
to make less in amount; wane

---

**ABBREVIATE**  
v.  
to make shorter; to shorten a word or phrase

---

Organized by TIM.



**A** GRE Word List

**B**

**C** **ABANDON**

---

**D** **ABASH**

---


**E** **ABATE**

---

**F** **ABBREVIATE**

---

Organized by TIM.





## References

---

- [1] Ajax: A New Approach to Web Applications.  
<http://www.adaptivepath.com/ideas/essays/archives/000385.php>
- [2] AJAX - Get or Post. <http://javascript.about.com/od/ajax/a/ajaxgp.htm>
- [3] Mozilla XMLHttpRequest. <https://developer.mozilla.org/en/XMLHttpRequest>
- [4] Introducing JSON. <http://www.json.org/index.html>
- [5] 6.470 2009 IAP - AJAX with jQuery. <http://6.470.scripts.mit.edu/2009/>
- [6] Websites using jQuery, ordered by popularity. <http://www.backendbattles.com/backend/jquery>
- [7] Project Gutenberg - The Adventures of Sherlock Holmes. <http://www.gutenberg.org/etext/1661>
- [8] Jonathan Chaffer and Karl Swedberg, *Learning jQuery 1.3*, PACKT publishing, 2009.

- Lecture Slides: <http://chyu.scripts.mit.edu/ajax-json-jquery-slides.pdf>
- Lecture Examples+Sol: <http://chyu.scripts.mit.edu/ajax-json-jquery-examples+sol.zip>