Internet Applications Design and Implementation 2017/2018 (Lab 2: JQuery & AJAX)

MIEI - Integrated Master in Computer Science and Informatics
Specialization block

João Leitão (jc.leitao@fct.unl.pt)
João Costa Seco (joao.seco@fct.unl.pt)

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Lab2 Goal:

- Learn the very basics of JQuery.
- Learn the basic usage of AJAX requests (GET and PUT) to interact with a REST service.

Introduction on JQuery:

- JQuery is a “default” javascript library that is commonly used to write client-side web applications.
  - Reduces the amount of text to interact with HTML elements.
  - Allows for the dynamic manipulation of the HTML DOM.
    - And this manipulation can be performed over multiple HTML elements at the same time.
  - Allows for CSS manipulation.
  - Handling and control over HTML events.
  - Provides a nice (more clean) interface to AJAX.

Introduction on JQuery:

- Using JQuery in your web application:
- Two methods:

1. Download and Link the JQuery library into your application:
   - jQuery.com
   - <script src="jquery-3.2.1.min.js"></script>
2. Link the JQuery library from the Google CDN
   - <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

Introduction on JQuery:

- Using JQuery in your web application:
- Two methods:

1. Download and Link the JQuery library into your application:
   - jQuery.com
   - <script src="jquery-3.2.1.min.js"></script>
2. **Link the JQuery library from the Google CDN**
   - **<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>**

Introduction on JQuery:

- Basics of Using JQuery:
- Basic syntax is: **$(*selector*).*action*()**
  - A $ sign to define/access jQuery
  - A (*selector*) to "query (or find)" HTML elements, operates in a similar fashion to CSS selectors.
  - A jQuery *action*() to be performed on the element(s)

Introduction on JQuery:

- Basics of Using JQuery:
- Basic syntax is: **$(*selector*).*action*()**
  - A $ sign to define/access jQuery
  - A (*selector*) to "query (or find)" HTML elements, operates in a similar fashion to CSS selectors.
  - A jQuery *action*() to be performed on the element(s)

Notice that the *selector* actually selects a **set of HTML elements**, applying na action to the result of a selector, will apply the action to **all** selected elements.

Introduction on JQuery:

- Basics of Using JQuery:
- Basic syntax is: **$(*selector*).*action*()**
  - A $ sign to define/access jQuery
  - A (*selector*) to "query (or find)" HTML elements, operates in a similar fashion to CSS selectors.
  - A jQuery *action*() to be performed on the element(s)

$(this).hide() - hides the current element.

$("p").hide() - hides all <p> elements.

$(".test").hide() - hides all elements with class="test".

$("#test").hide() - hides the element with id="test".

Introduction on JQuery:

- Capturing the **Document Ready Event:**
- As in many aspects of web programming, there are multiple ways to do this, in particular there are two in this case:
- $(document).ready(function(){
  // Your code goes here
  });
- $(function(){
  // Your code goes here
  });

Introduction on JQuery:

- Capturing the **Document Ready Event:**
- As in many aspects of web programming, there are multiple ways to do this, in particular there are two in this case.
- Here you can do things like:
  - Hide HTML elements that are not yet being used.
  - Associate functions with user actions (pressing buttons, scrolling, mouse over events).
  - Trigger the loading of dynamic content to the static HTML structure.

Introduction on JQuery:

- Capturing the **Document Ready Event:**
- As in many aspects of web programming, there are multiple ways to do this, in particular there are two in this case.
- Here you can do things like:
  - Hide HTML elements that are not yet being used.
  - Associate functions with user actions (pressing buttons, scrolling, mouse over events).
  - Trigger the loading of dynamic content to the static HTML structure.

Introduction on JQuery:

- Associating functions with HTML elements events**:**
- Two ways to do that:
  - $(*selector*).**event**(function(){
    //Your code goes here;
    });
  - $(selector).on(”**eventname**", function()
    //Your code goes here;
    });

Introduction on JQuery:

- Associating functions with HTML elements events**:**
- Two ways to do that:
  - $(**.submit**).**click**(function(){
    //Your code goes here;
    });
  - $(**.submit**).on(”**click**", function()
    //Your code goes here;
    });

Introduction on JQuery:

- Manipulation of the DOM**:**
- Appending a (child) HTML element to na existing HTML element:

$(*selector*).append(

$(**ELEMENT TYPE**).

attr(“*atributeName*”,”*AtributeValue*”)

);

Introduction on JQuery:

- Manipulation of the DOM (with chaining)**:**
- Appending a (child) HTML element to na existing HTML element:

$(*selector*).append(

$(**ELEMENT TYPE**).

attr(“*atributeName*”,”*AtributeValue*”).

attr(“*atributeName*”,”*AtributeValue*”).

append($(**ELEMENT TYPE**).attr(...)

).append(

$(**ELEMENT TYPE**).attr(“*atributeName*”,”*Value*”)

);

Introduction on JQuery:

- Manipulation of the DOM (with chaining)**:**
- Appending a (child) HTML element to na existing HTML element:

```
$(“.list”).append(
        $(“<li>”).
                attr(“id”,”1”).
                attr(“name”,”myFirstItem”).
                append( $(“<div>”).attr(...)
).append(
        $(“<li>”).attr(“id”,”2”)
);
```

Introduction on JQuery:

- Manipulation of the DOM (with chaining)**:**
- Appending a (child) HTML element to na existing HTML element:

```
$(“.list”).append(
        $(“<li>”).
                attr(“id”,”1”).
                attr(“name”,”myFirstItem”).
                append( $(“<div>”).attr(...)
).append(
        $(“<li>”).attr(“id”,”2”).append(“Visible text”)
);
```

Introduction on JQuery:

- Many high level and useful functions (Go explore on your own)
  - https://www.w3schools.com/jquery/default.asp
  - https://api.jquery.com

- Example here:
  - Imagine that you want to perform some special action on each element of na (object) list

```
$.map(objectArray, function(element) {
        //your code goes here
});
```

Introduction on JQuery:

- Many high level and useful functions (Go explore on your own)
  - https://www.w3schools.com/jquery/default.asp
  - https://api.jquery.com

- Example here:
  - You can even transform each object in the list and return it to create a new array of (transformed objects):

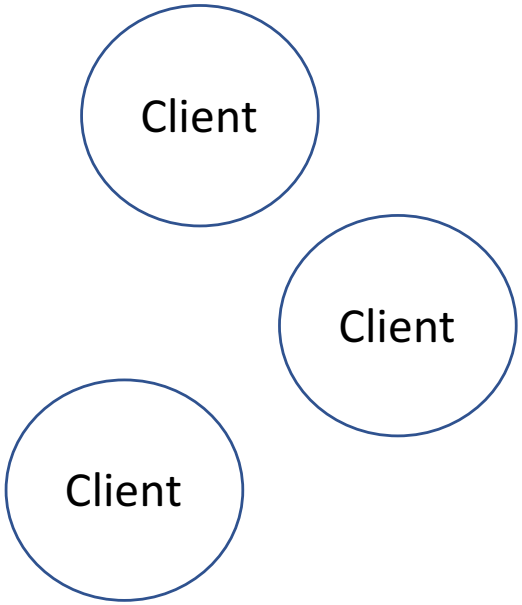
```
var arr = $.map(objectArray, function(element) {
        //your code goes here
        return element;
});
```

Introduction to AJAX: (With JQuery)

- Why would anyone need to use AJAX?
  - Update a web page without reloading the page.
  - Request data from a server - after the page has loaded.
  - Receive data from a server - after the page has loaded.
  - Send data to a server - in the background.

- It’s the javascript interface to perform REST requests (GET, POST, PUT, DELETE)

Introduction to AJAX: (With JQuery)

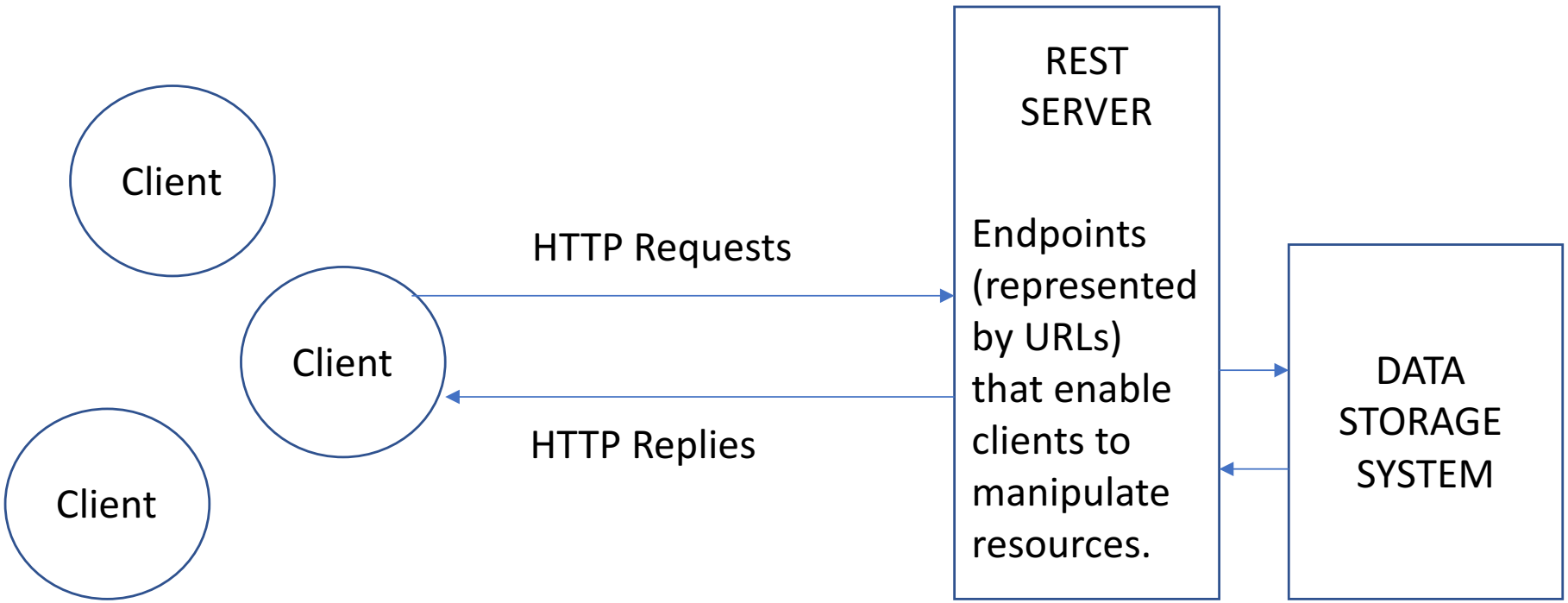


REST SERVER

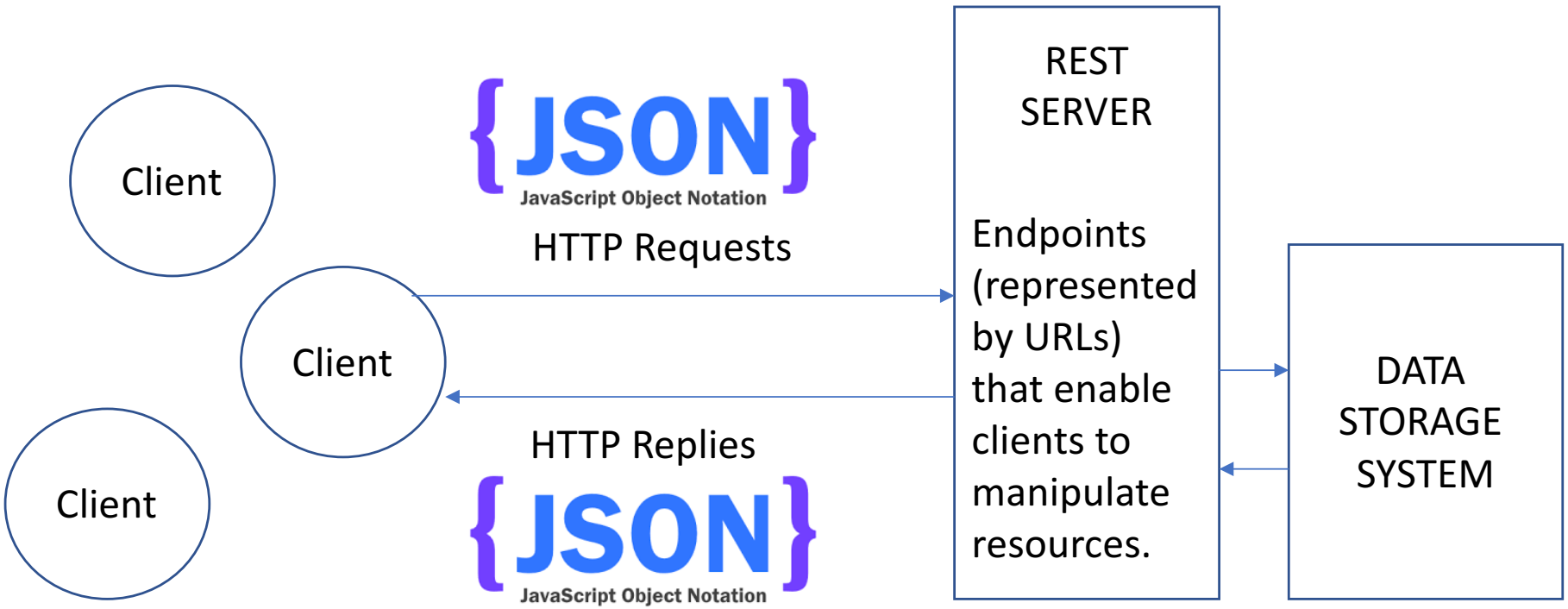
Endpoints (represented by URLs) that enable clients to manipulate resources.

DATA STORAGE SYSTEM

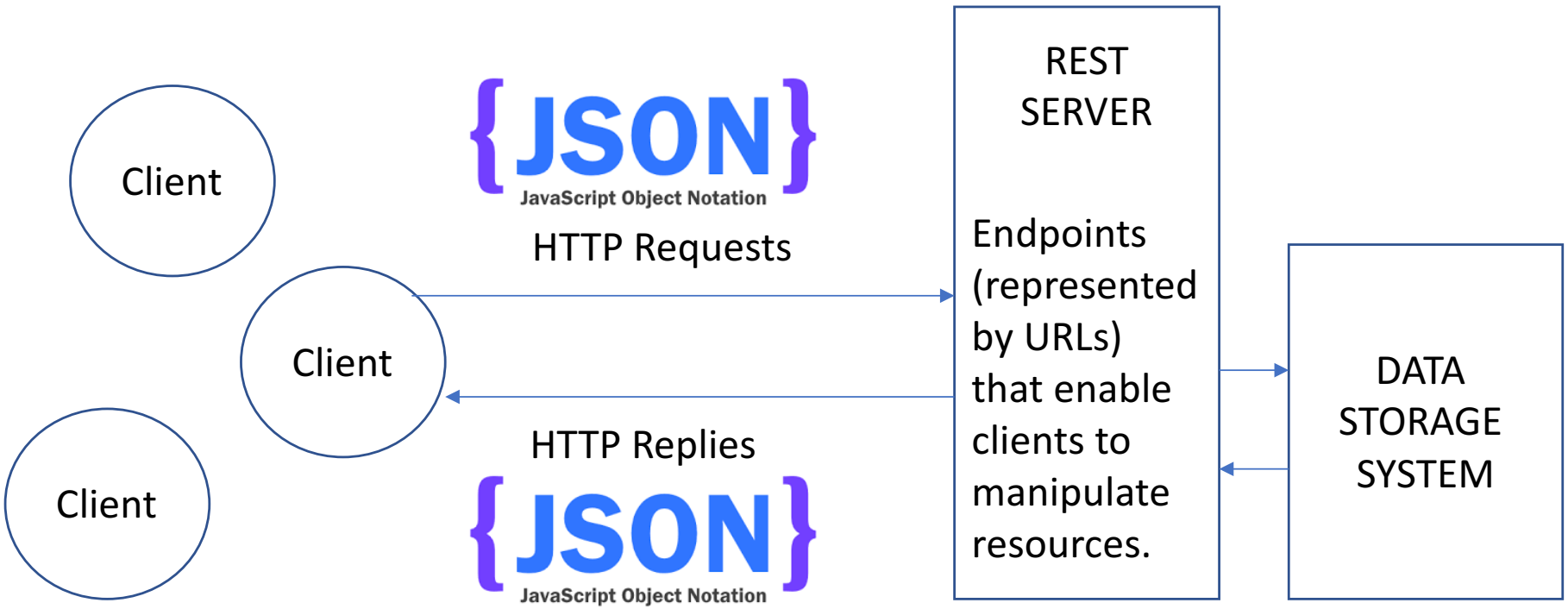
Introduction to AJAX: (With JQuery)



Introduction to AJAX: (With JQuery)



Introduction to AJAX: (With JQuery)



In AJAX these interactions are asynchronous by default.

Introduction to AJAX: (With JQuery)

- Multiple types of requests:
  - GET: Get some resources (information).
  - POST: Create some resource (with information).
  - PUT: Modify some resource (i.e, update resource).
  - DELETE: Delete some resource.

Today we will be focusing on the GET and POST operations.

Introduction to AJAX: (With JQuery)

- You can execute na AJAX request natively in Javascript.

```
var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
     document.getElementById("demo").innerHTML = this.responseText;
    }
  };
  xhttp.open("GET", "ajax_info.txt", true);
  xhttp.send();
```

Introduction to AJAX: (With JQuery)

- You can execute na AJAX request nati
  Javascript.

```
var xhttp = new XMLHt
  xhttp.onread
    if (
  x
  xh
```

NOT THE MOST ELEGANT OR CONCISE SEMANTICS…

Don’t worry JQuery can help you with this too…

Introduction to AJAX: (With JQuery)

- Performing a AJAX Get request using JQuery:

- Structure of the call: $.get(*Endpoint URL,callback function*);

- Simple example:

```
$("button").click(function(){
    $.get("demo_test.asp", function(data, status){
        alert("Data: " + data + "\nStatus: " + status);
    });
});
```

Introduction to AJAX: (With JQuery)

- Performing a AJAX Get request using JQuery:

- Structure of the call: $.get(*Endpoint URL,callback function*);

- Simple example:

```
$("button").click(function(){
    $.get("demo_test.asp", function(data, status){
        alert("Data: " + data + "\nStatus: " + status);
    });
});
```

Introduction to AJAX: (With JQuery)

- Performing a AJAX POST request using JQuery:

- Structure of the call: $.post(*Endpoint URL,data,callback function,data type of the reply*);
- Simple example:

```
$("button").click(function(){
    $.post("demo_test_post.asp",
    {
        name: "Donald Duck",
        city: "Duckburg"
    },
    function(data, status){
        alert("Data: " + data + "\nStatus: " + status);
    });
});
```

Introduction to AJAX: (With JQuery)

- Sometimes you might need more control over what happens in your AJAX request.
- For instance, ensure that data is passed encoded in JSON on your Post Request.
- In these cases, you can always resort to the general ajax call provided by JQuery.

Introduction to AJAX: (With JQuery)

- An example of a general POST operation with the generic JQuery AJAX interface:

```
var data = {'id': "1",
            'name': "John"};
$.ajax({
        type: "POST",
        url: "http://peopledatabase.nop/rest/person",
        processData: false,
        contentType: 'application/json; charset=utf-8',
        data: JSON.stringify(data),
        success: function() {
                console.log("Post executed with sucess."); },
        error: function(status) {
                console.log("Failed to Post: " + status); }
        });
```

Todays Exercise...

- Lets pick up the example from last week, and improve some aspects:
  - No javascript code within the HTML file.
  - Lets add a button for reloading the messages in the page.
  - Lets add another user imput for the person to provide a username when they post something...
  - Oh... And remove the list itself from the HTML (build that with JQuery, as well as all elements within that list).
  - Lets ensure that the contenst shown come from a service.

Todays Exercise…

- I have setup a service online that you can use:
- Endpoint: http://ciai2017-180918.appspot.com/rest/
- Has two operations:
  - GET Operation: Provides you with a JSON array of message entries.
  - POST Operation: Receives a JSON object with the structure of a message and stores it.

- Go Ahead and use your favorite browser to see the output of the GET operation.

Todays Exercise…

- The output of the GET Operation:
- [{"title":"First Message","summary":"This is an example of a message","imageURL":"","username":"jleitao","timestamp":1506418498824},{"title":"Second Message","summary":"This is another example, this time with pictrue","imageURL":"https://memegenerator.net/img/images/600x600/14834216/cat-on-drugs.jpg","username":"jleitao","timestamp":1506418698824},{"title":"DI / FCT / UNL","summary":"Melhor departamento de informÃ¡tica do mundo.","imageURL":"https://www.di.fct.unl.pt/sites/www.di.fct.unl.pt/themes/di_fct_unl_pt_2017/images/logo.png","username":"jleitao","timestamp":1506418798824},{"title":"Lecture of CIAI","summary":"There will be no Lecture of CIAI this week","imageURL":"https://static.rappad.co/api/file/9MyNQVqfR6OxgO2Et0yq","username":"jcs","timestamp":1506418898824}]

Todays Exercise…

- The output of the GET Operation:
- [{"title":"First Message","summary":"This is an example of a message","imageURL":"","username":"jleitao","timestamp":1506418498824},{"title":"Second Message","summary":"This is another example, this time with pictrue","imageURL":"https://memegenerator.net/img/images/600x600/14834216/cat-on-drugs.jpg","username":"jleitao","timestamp":1506418698824},{"title":"DI / FCT / UNL","summary":"Melhor departamento de informÃ¡tica do mundo.","imageURL":"https://www.di.fct.unl.pt/sites/www.di.fct.unl.pt/themes/di_fct_unl_pt_2017/images/logo.png","username":"jleitao","timestamp":1506418798824},{"title":"Lecture of CIAI","summary":"There will be no Lecture of CIAI this week","imageURL":"https://static.rappad.co/api/file/9MyNQVqfR6OxgO2Et0yq","username":"jcs","timestamp":1506418898824}]

Todays Exercise…

- The output of the GET Operation:
- **[**{"title":"First Message","summary":"This is an example of a message","imageURL":"","username":"jleitao","timestamp":1506418498824},**{"title":"Second Message","summary":"This is another example, this time with pictrue","imageURL":"https://memegenerator.net/img/images/600x600/14834216/cat-on-drugs.jpg","username":"jleitao","timestamp":1506418698824}**,{"title":"DI / FCT / UNL","summary":"Melhor departamento de informÃ¡tica do mundo.","imageURL":"https://www.di.fct.unl.pt/sites/www.di.fct.unl.pt/themes/di_fct_unl_pt_2017/images/logo.png","username":"jleitao","timestamp":1506418798824},{"title":"Lecture of CIAI","summary":"There will be no Lecture of CIAI this week","imageURL":"https://static.rappad.co/api/file/9MyNQVqfR6OxgO2Et0yq","username":"jcs","timestamp":1506418898824}**]**

Todays Exercise…

- A message JSON representation:

{ "title":"Second Message",

"summary":"This is another example, this time with pictrue",

"imageURL":"https://memegenerator.net/img/images/600x600/14834216/cat-on-drugs.jpg",

"username":"jleitao",

"timestamp":1506418698824}

Todays Exercise…

- A message JSON representation:

{ "title":"Second Message",

"summary":"This is another example, this time with pictrue",

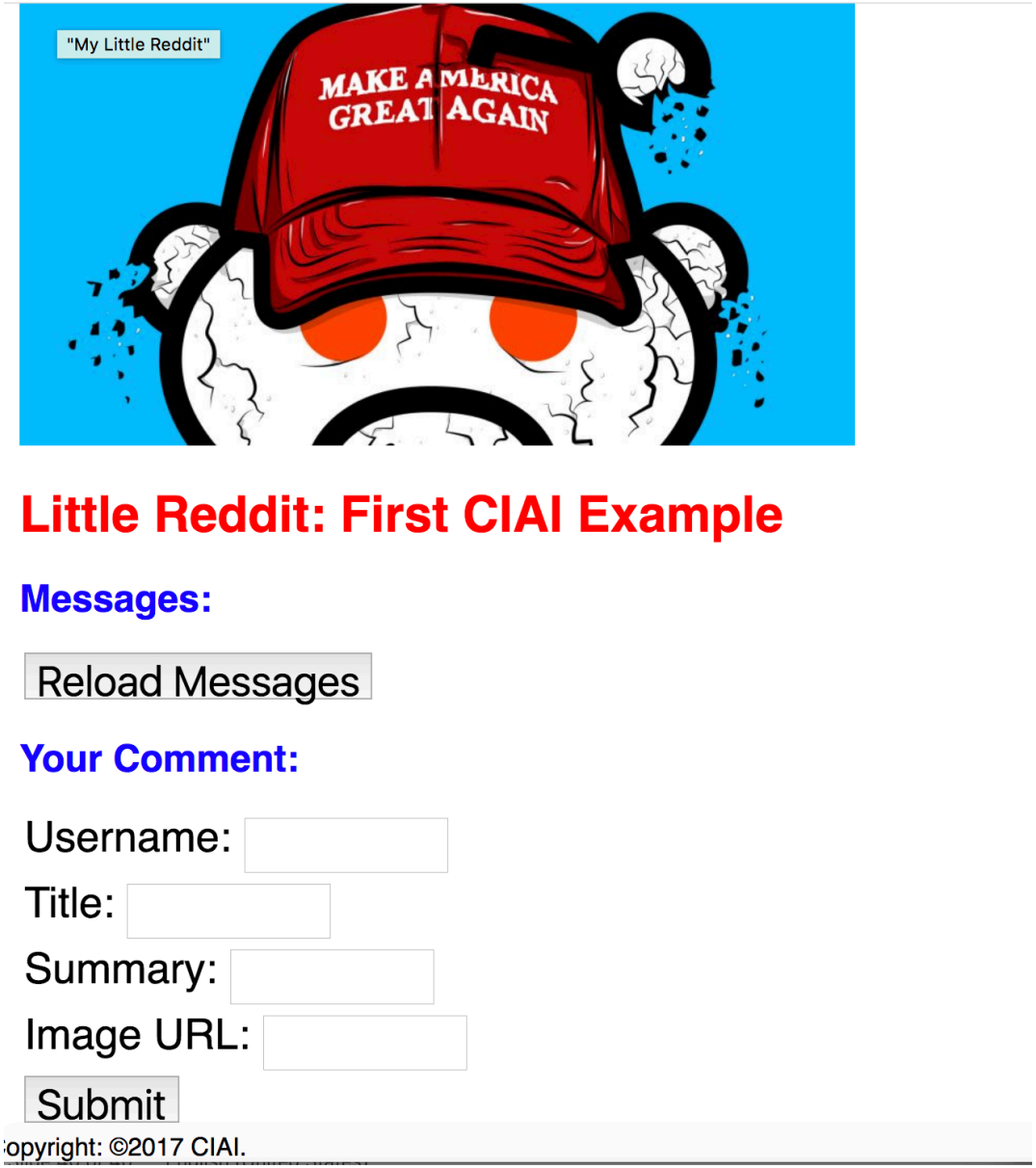
"imageURL":"https://memegenerator.net/img/images/600x600/14834216/cat-on-drugs.jpg",

"username":"jleitao",

"timestamp":1506418698824}

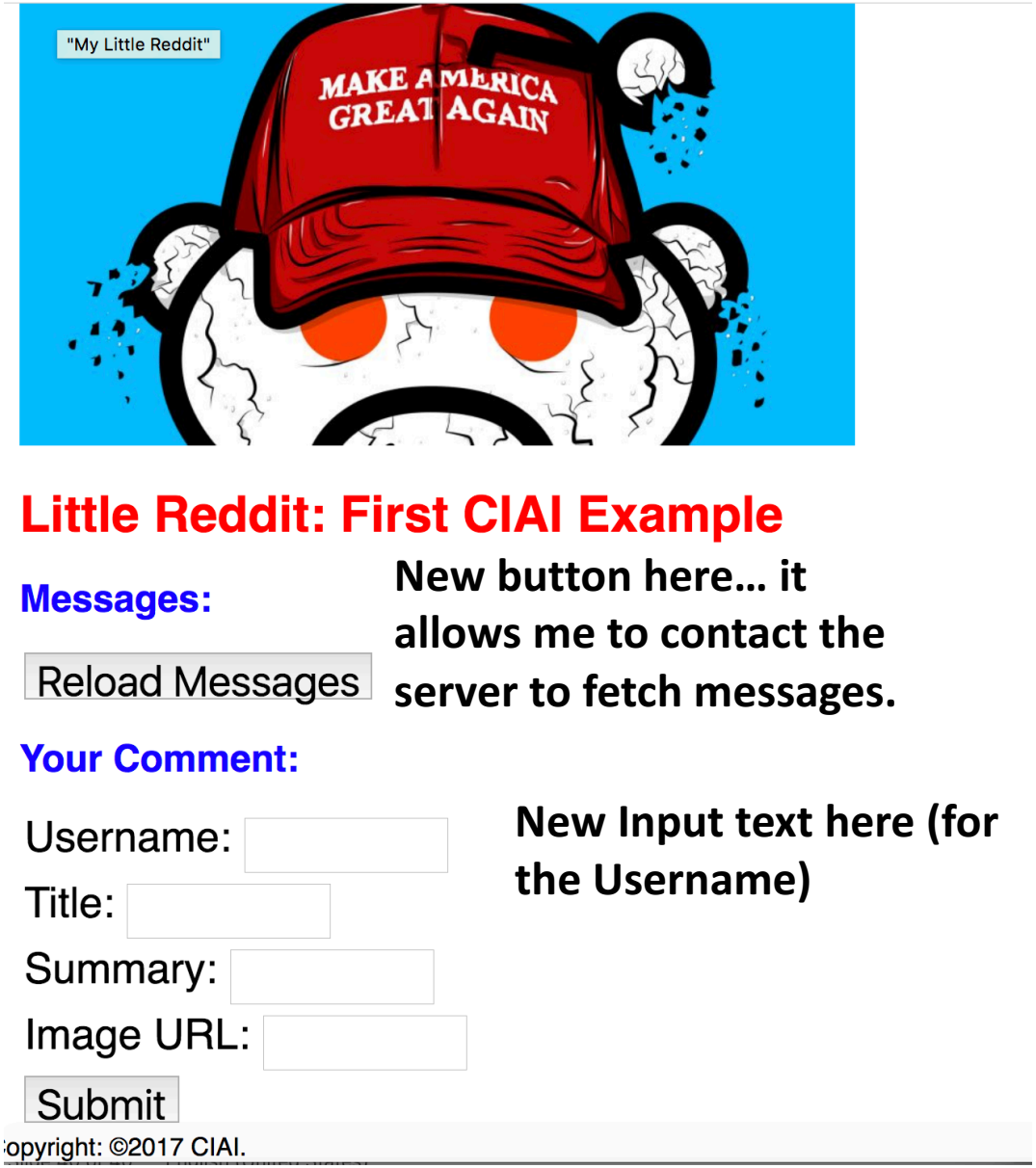
Use an empty string for notifying the use of the default image.

Starting Point:



Starting Point:

There is no List here any longer (but there is an empty div)



What should you do…

- Make your page similar to that one (do not delete the structure of the messages, you will need it eventually).
- Import the Jquery Library
- Associate actions to the buttons: you can test with *alert(TEXT)*
- Upon clicking the reload button, get messages from server and display them.
- Upon clicking on the submit button, add a new message to the ones stored in the server.

Solutions: Change Page Structure

```
<section>
  <h2>Messages:</h2>

  <div id="board">
  <!--<ol id="messagelist">
     <li>
         <div class="image"><img src="img/default.png" alt="the default image for a post"></div>
         <div class="mcontent">
          <div class="title">First message.</div>
          <div class="summary">This is the summary of the first message.</div>
          <div class="info">by: username, posted: a long time ago.</div>
         </div>
      </li>
      <li>
         <div class="image"><img src="img/default.png" alt="the default image for a post"></div>
         <div class="mcontent">
          <div class="title">Second message.</div>
          <div class="summary">This is the summary of the second message.</div>
          <div class="info">by: username, posted: less time ago.</div>
         </div>
      </li>
  </ol>-->
  </div>
  <div class="userinput"><button id="load">Reload Messages</button></div>

  <h2>Your Comment:</h2>

  <div class="userinput">Username: <input type="text" id="username"></div>
  <div class="userinput">Title: <input type="text" id="title"></div>
  <div class="userinput">Summary: <input type="text" id="summary"></div>
  <div class="userinput">Image URL: <input type="text" id="imgurl"></div>
  <div class="userinput"><button name="submit">Submit</button></div>

</section>
```

Solutions: Change Page Structure

```
<section>
  <h2>Messages:</h2>

  <div id="board">
  <!--<ol id="messagelist">
     <li>
        <div class="image"><img src="img/default.png" alt="the default image for a post"></div>
        <div class="mcontent">
         <div class="title">First message.</div>
         <div class="summary">This is the summary of the first message.</div>
         <div class="info">by: username, posted: a long time ago.</div>
        </div>
      </li>
      <li>
        <div class="image"><img src="img/default.png" alt="the default image for a post"></div>
        <div class="mcontent">
         <div class="title">Second message.</div>
         <div class="summary">This is the summary of the second message.</div>
         <div class="info">by: username, posted: less time ago.</div>
        </div>
      </li>
  </ol>-->
  </div>
  <div class="userinput"><button id="load">Reload Messages</button></div>

  <h2>Your Comment:</h2>

  <div class="userinput">Username: <input type="text" id="username"></div>
  <div class="userinput">Title: <input type="text" id="title"></div>
  <div class="userinput">Summary: <input type="text" id="summary"></div>
  <div class="userinput">Image URL: <input type="text" id="imgurl"></div>
  <div class="userinput"><button name="submit">Submit</button></div>

</section>
```

Solutions: Associate Functions with Buttons

```
1  $(document).ready(function() {
2      $("#load").on("click", function() {
3          alert("Triggering Reload of Messages.");
4      });
5
6      $("[name='submit']").on("click", function() {
7          alert("Triggering Submit of a new Message.");
8      });
9  });
10
11
```

Solutions: Get Messages from the Server

```
$(document).ready(function() {
    $("#load").on("click", function() {
        reloadMessages();
    });

    $("[name='submit']").on("click", function() {
        alert("Triggering Submit of a new Message.");
    });
});
```

Solutions: Get Messages from the Server

```
function reloadMessages() {
   $.get("http://ciai2017-180918.appspot.com/rest/",
           function(data, status) {
                if(status == "success") {
                    console.log("data: " + data);
                    $("#board").empty();
                    $("#board").append( $('<ol>').attr('id','messageList') );
                    $.map(data, function(el) {
                        $("#messageList").append(
                            $('<li>').append(
                                $('<div>').attr('class','image').append(
                                    $('<img>').attr('src',
                                    (el.imageURL==""?"img/default.png":el.imageURL)).attr('alt','Post Image')
                                )
                            ).append(
                                $('<div>').attr('class','mcontent').append(
                                   $('<div>').attr('class','title').append(el.title)
                                ).append(
                                    $('<div>').attr('class','summary').append(el.summary)
                                ).append(
                                    $('<div>').attr('class','info').append("Posted by: " + el.username + " at
                                    " + new Date(el.timestamp))
                                )
                            )
                        )
                    });
                } else {
                    console.log("Failed to load content.");
                }

           }
       );
}
```

Solutions: Post a Message to the Server

```
$(document).ready(function() {
    $("#load").on("click", function() {
        reloadMessages();
    });

    $("[name='submit']").on("click", function() {
        postMessage();
    });
});
```

Solutions: Post a Message to the Server

```
function postMessage() {
    var data = {'title': $("#title").val(),
                'summary': $("#summary").val(),
                'imageURL': $("#imgurl").val(),
                'username': $("#username").val(),
                'timestamp': new Date().getMilliseconds()};

    $.ajax({
        type: "POST",
        url: "http://ciai2017-180918.appspot.com/rest/",
        processData: false,
        contentType: 'application/json; charset=utf-8',
        data: JSON.stringify(data),
        success: function() {
            console.log("Post executed with sucess.");
        },
        error: function(status) {
            console.log("Failed to Post: " + status);
        }
    });
}
```

Solutions: Just to make this good...

- After the load of the document, and associating actions with the buttons, trigger the load of messages from the server.
- After successfuly posting a message to the server, trigger the load of messages from the server.