Data Representation

Lab 06.3: AJAX calls to REST

Lecturer: Andrew Beatty

In this lab we want to make the AJAX calls that will call the RESTful API

At http://andrewbeatty1.pythonanywhere.com

* Remember the server serves out a RESTfull API that performs CRUD operations on books.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Action | Method | URL | Sample params | Sample return |
| Get all | GET | /books | none | [  {  “title”:”blah”,  …  },  {…},{…}  ] |
| Find by id | GET | /books/id | none | {  "Title":"blah",  "Author":"blah",  "id":23,  "price":1250  } |
| Create | POST | /books | {  "Title":"blah",  "Author":"blah",  "price":1250  } | {  "Title":"blah",  "Author":"blah",  "id":23,  "price":1250  } |
| Update | PUT | /books/*id* | {  "price":1300  } | {  "Title":"blah",  "Author":"blah",  "id":23,  "price":1300  } |
| delete | DELETE | /books/*id* | none | {  "done":true  } |

1. Test that we can call the server from the web page, by creating a webpage called testgetall.html.

<html>

    <head>

        <title>

            test get all

        </title>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>

    </head>

    <body>

    <script>

        function getAll(){

            $.ajax({

                "url": " http://andrewbeatty1.pythonanywhere.com/books",

                "method":"GET",

                "data":"",

                "dataType": "JSON",

                "success":function(result){

                    console.log(result);

                },

                "error":function(xhr,status,error){

                    console.log("error: "+status+" msg:"+error);

                }

            });

        }

getAll()

    </script>

</html>

1. If nothing happens in the console, check that you called the function getall()
2. Later we will be opening the page through a local server <http://127.0.0.1:5000/testgetall.html>

but for now you will be opening the html page through the file system, so you will have something like this as your URL. <file:///C:/morespace/Dropbox/gmit/dataRepresentation/code/topic06-REST/testgetall.html>

1. Add a callback function to the parameters and call it in the success. This way we can have the functionality of what we do with the data separate from retrieving the data.

<script>

        function getAll(callback){

            $.ajax({

                "url": "http://andrewbeatty1.pythonanywhere.com/books",

                "method":"GET",

                "data":"",

                "dataType": "JSON",

                "success":function(result){

                    //console.log(result);

                    callback(result)

                },

                "error":function(xhr,status,error){

                    console.log("error: "+status+" msg:"+error);

                }

            });

        }

        function processGetAll(result){

            console.log("in process")

            console.log(result)

        }

        getAll(processGetAll)

# create

1. Make a web page that tests the create called testCreate.html (I have put this code into the same file as above in the sample code.

contentType

is the type of the data that is being passed to the server (in data)

function createBook(book, callback){

            //console.log(JSON.stringify(book));

            $.ajax({

                "url": "http://andrewbeatty1.pythonanywhere.com/books",

                "method":"POST",

                "data":JSON.stringify(book),

                "dataType": "JSON",

                contentType: "application/json; charset=utf-8",

                "success":function(result){

                    console.log(result);

                    //callback(result)

                },

                "error":function(xhr,status,error){

                    console.log("error: "+status+" msg:"+error);

                }

            });

python a\_simpleserver.py

1. Test it first without the callback function and then with it

Be aware, on the server that **T**itle, **A**uthor and **P**rice all start with capitals

Should have been small, but I made a mistake with the code I uploaded to the server

function processCreateresponse(result){

            console.log(result)

        }

book = {"Title":"javascript","Author":"andrew","Price":12}

createBook(book,processCreateresponse)

## update

1. Test the update

 function updateBook(book, callback){

            //console.log(JSON.stringify(book));

            $.ajax({

                "url": "http://andrewbeatty1.pythonanywhere.com/books/"+encodeURI(book.id),

                "method":"PUT",

                "data":JSON.stringify(book),

                "dataType": "JSON",

                contentType: "application/json; charset=utf-8",

                "success":function(result){

                    console.log(result);

                    //callback(result)

                },

                "error":function(xhr,status,error){

                    console.log("error: "+status+" msg:"+error);

                }

            });

        }

1. Firstly, without the call back function then with the callback

  function processUpdateResponse(result){

            console.log(result)

        }

   book = {id:155,"Price":999}

   updateBook(book,processUpdateResponse)

## delete

1. Test delete

function deleteBook(id, callback){

            $.ajax({

                "url": "http://andrewbeatty1.pythonanywhere.com/books/"+id,

                "method":"DELETE",

                "data":"",

                "dataType": "JSON",

                contentType: "application/json; charset=utf-8",

                "success":function(result){

                    console.log(result);

                    //callback(result)

                },

                "error":function(xhr,status,error){

                    console.log("error: "+status+" msg:"+error);

                }

            });

        }

1. Again, test withoug the callback then with it

 function processDeleteResponse(result){

            console.log("in pprocess delete")

            console.log(result)

        }

        deleteBook(155,processUpdateResponse)

## How would you integrate this with lab06.02

1. So we have sample code in the test section, comment out the function calls and you can include this file at the top of the lab06.02, then put the calls into the appropriate places.
2. I have made a html called lab06.3b-book-interface.html
3. You should call getAll at the bottom of the page to initialise the table
4. In the processGetAll use a for loop to get each book individually, you would use this with the add to table to add them to the table.

 function processGetAllResponse(result){

            console.log("in process")

            //console.log(result)

            for (book of result){

                //console.log(book)

                // issue the format of the book object is different from lab06.02

                // there are two solutions change the book object in lan06.02 to have capitals

                // or convert

                displayBook = {}

                displayBook.id = book.id

                displayBook.author = book.Author

                displayBook.title = book.Title

                displayBook.price = book.Price

                // you can now pass it to addBookToTable

                console.log(displayBook)

            }

        }

1. You will have do a similar conversion in the doCreate, do Update and doDelete.
2. Remember there may be multiple people working on this data.