AJAX and jQuery AJAX

AJAX Concepts, XMLHttpRequest,
jQuery AJAX: $.ajax(), $.get(), $.post()

SoftUni Team

Technical Trainers

Software University
University

SoftUni
Foundation

Software University
http://softuni.bg

Table of Contents

SoftUni Foundation

Have a Question?

sli.do

#JSCORE

AJAX

Asynchronous JavaScript And XML

The conventional model

- Based on complete HTML pages
- Each user action required that a complete new page be loaded from the server
- Bad user experience
- All of the content had to be re-sent, even though only some of the information had changed
- Inefficient bandwidth use

XMLHTTP

- In 1996, the iframe tag was introduced by Internet Explorer
- In 1998, the Microsoft Outlook Web App team developed the concept behind the XMLHttpRequest scripting object. It appeared as XMLHTTP in the second version of the MSXML library, which shipped with Internet Explorer 5.0 in March 1999
- The functionality of the XMLHTTP ActiveX control in IE 5 was later implemented by Mozilla, Safari, Opera and other browsers as the XMLHttpRequest JavaScript object
- Microsoft adopted the native XMLHttpRequest model as of Internet Explorer 7

What is AJAX?

SoftUni Foundation

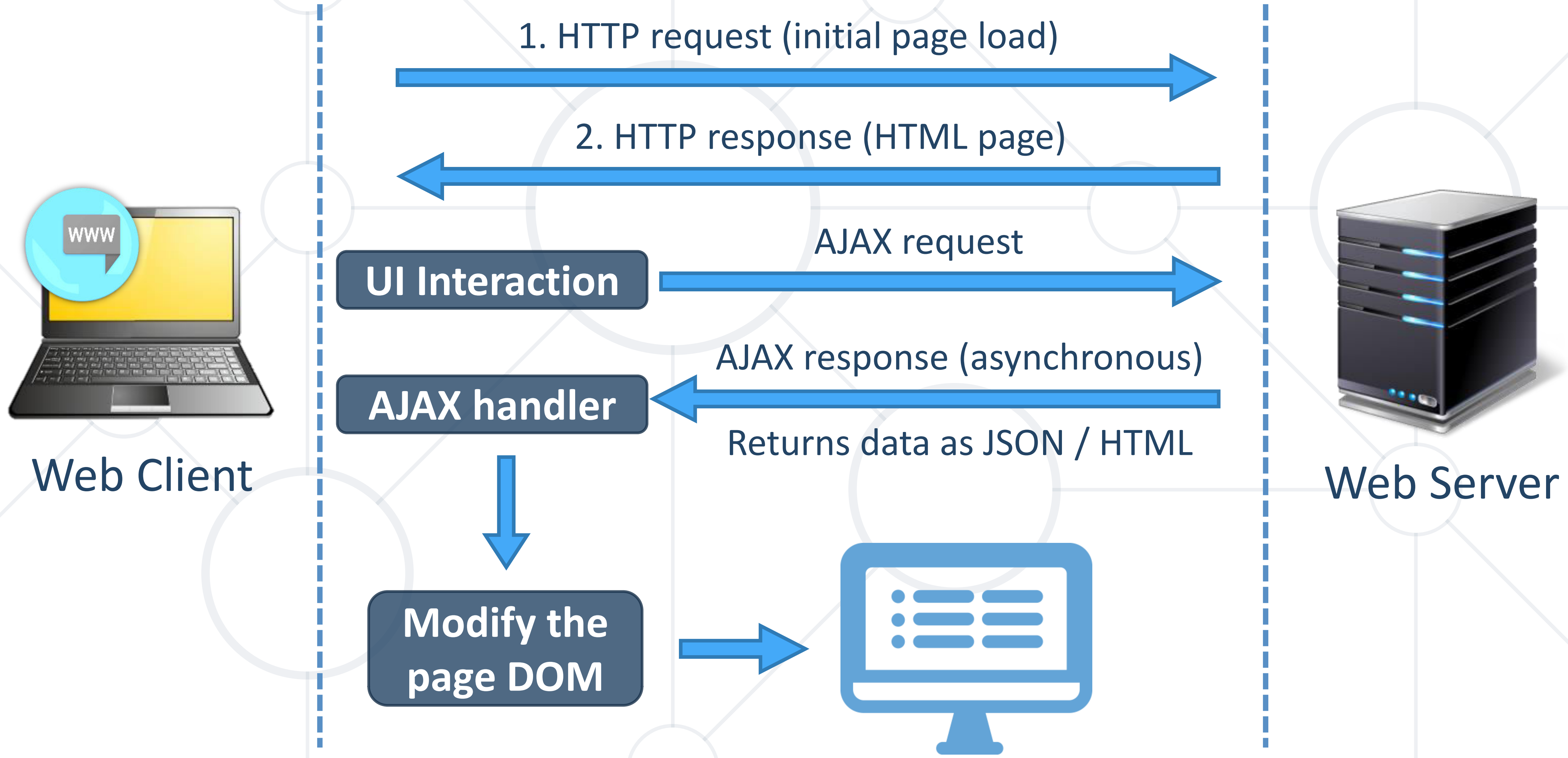
- AJAX == **Asynchronous JavaScript And XML**
  - A broad group of Web technologies that can be used to implement a Web application that communicates with a server in the background, without interfering with the current state of the page
    - HTML (or XHTML) and CSS for presentation
    - The Document Object Model (DOM) for dynamic display of and interaction with data
    - JSON or XML for the interchange of data, and XSLT for its manipulation
    - The XMLHttpRequest object for asynchronous communication
    - JavaScript to bring these technologies together

AJAX Use cases

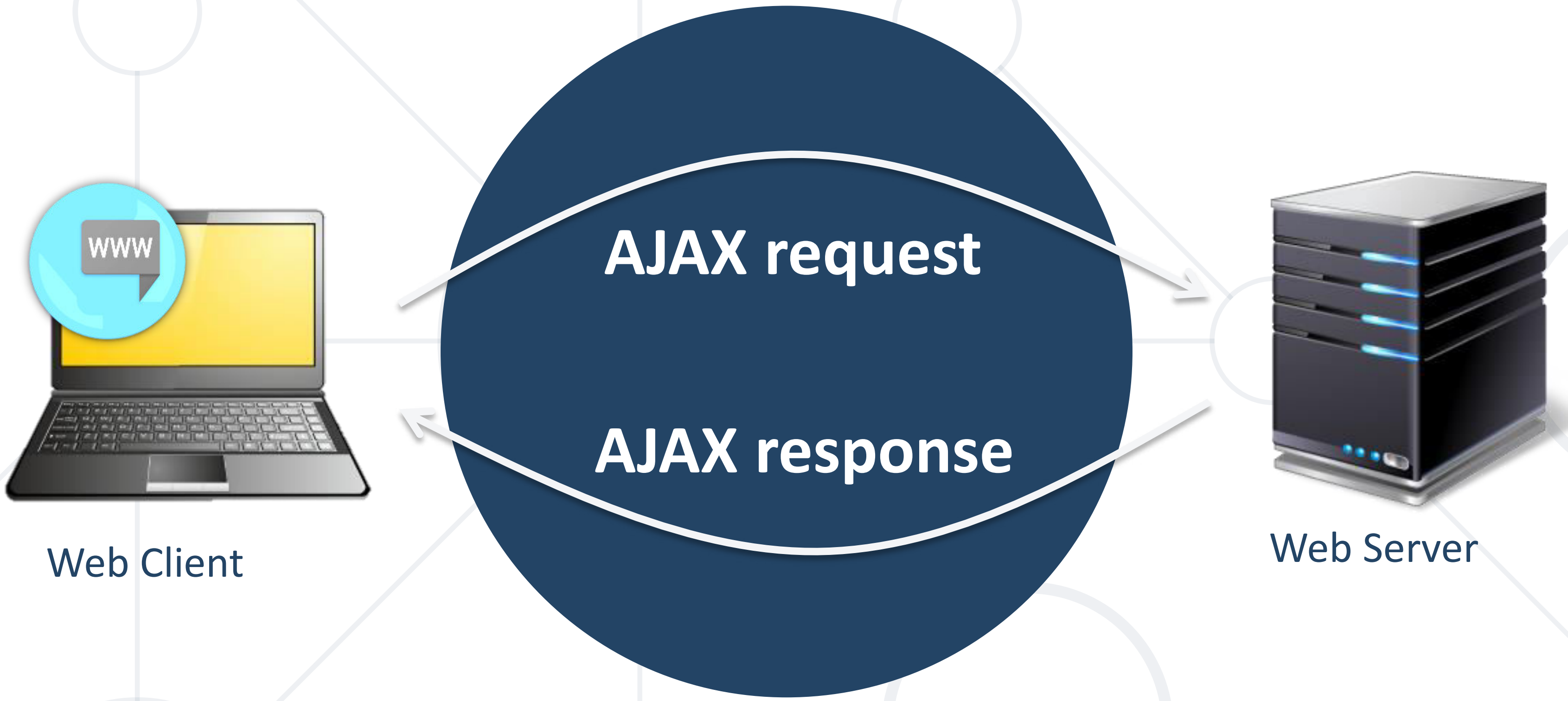
SoftUni Foundation

- **Partial page rendering**
  - Load HTML fragment and display it in a container
- **Web service**
  - Load data as JSON / XML object and visualize it with JavaScript / jQuery
- **Manipulate domain objects**
  - Using RESTful services

AJAX: Workflow



1. HTTP request (initial page load)
2. HTTP response (HTML page)
UI Interaction
AJAX request
AJAX response (asynchronous)
AJAX handler
Returns data as JSON / HTML
Web Client
Web Server
Modify the page DOM
www



Using the XMLHttpRequest Object

XMLHttpRequest – Standard API for AJAX

```
<button onclick="loadRepos()">Load Repos</button>
<div id="res"></div>
```

```
function loadRepos() {
  let req = new XMLHttpRequest();
  req.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200)
      document.getElementById("res").textContent =
        this.responseText;
  };
  req.open("GET",
    "https://api.github.com/users/testnakov/repos", true);
  req.send();
}
```

Check your code here: https://judge.softuni.bg/Contests/357

XMLHttpRequest – Standard API for AJAX

```
var xhr = new XMLHttpRequest();
xhr.open('GET', 'myservice/username?id=some-unique-id');
xhr.onload = function() {
    if (xhr.status === 200) {
        alert('User\'s name is ' + xhr.responseText);
    }
    else {
        alert('Request failed.  Returned status of ' + xhr.status);
    }
};
xhr.send();
```

jQuery AJAX

Simplified AJAX Calls with jQuery

jQuery AJAX

- jQuery simplifies how developers make AJAX calls
- Low-Level Interface
  - jQuery.ajax() - Perform an asynchronous HTTP (Ajax) request.
  - jQuery.ajaxPrefilter() - Handle custom Ajax options or modify existing options before each request is sent and before they are processed by $.ajax().
  - jQuery.ajaxSetup() - Set default values for future Ajax requests. Its use is not recommended.
  - jQuery.ajaxTransport() - Creates an object that handles the actual transmission of Ajax data.

jQuery AJAX

SoftUni Foundation

- Shorthand Methods
  - jQuery.get() - Load data from the server using a HTTP GET request.
  - jQuery.getJSON() - Load JSON-encoded data from the server using a GET HTTP request
  - jQuery.getScript() - Load a JavaScript file from the server using a GET HTTP request, then execute it.
  - jQuery.post() - Load data from the server using a HTTP POST request.
  - .load() - Load data from the server and place the returned HTML into the matched element.

jQuery AJAX

- Global Ajax Event Handlers
  - .ajaxComplete() - Register a handler to be called when Ajax requests complete
  - .ajaxError() - Register a handler to be called when Ajax requests complete with an error
  - .ajaxSend() - Attach a function to be executed before an Ajax request is sent
  - .ajaxStart() - Register a handler to be called when the first Ajax request begins
  - .ajaxStop() - Register a handler to be called when all Ajax requests have completed
  - .ajaxSuccess() - Attach a function to be executed whenever an Ajax request completes successfully

jQuery VS Native XMLHttpRequest - GET

SoftUni Foundation

```
$.ajax('myservice/username', {
    data: {
        id: 'some-unique-id'
    }
})
.then(
    function success(name) {
        alert('User\'s name is ' +
name);
    },

    function fail(data, status) {
        alert('Request failed.
Returned status of ' + status);
    }
);
```

```
var xhr = new XMLHttpRequest();
xhr.open('GET',
'myservice/username?id=some-unique-
id');
xhr.onload = function() {
    if (xhr.status === 200) {
        alert('User\'s name is ' +
xhr.responseText);
    }
    else {
        alert('Request failed.
Returned status of ' + xhr.status);
    }
};
xhr.send();
```

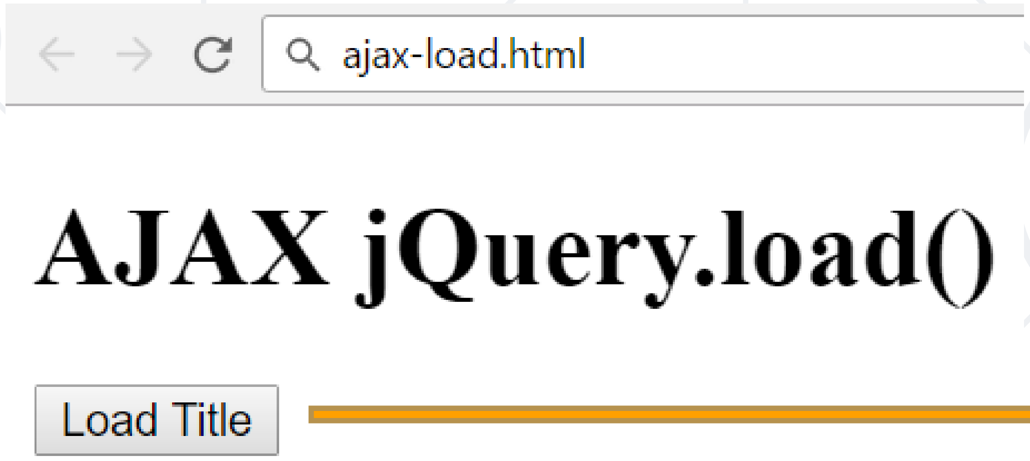
jQuery VS Native XMLHttpRequest - POST

```
var newName = 'John Smith';

$.ajax('myservice/username?' + $.param({id: 'some-
unique-id'}), {
    method: 'POST',
    data: {
        name: newName
    }
})
.then(
    function success(name) {
        if (name !== newName) {
            alert('Something went wrong.  Name is now
' + name);
        }
    },

    function fail(data, status) {
        alert('Request failed.  Returned status of '
+ status);
    }
);
```

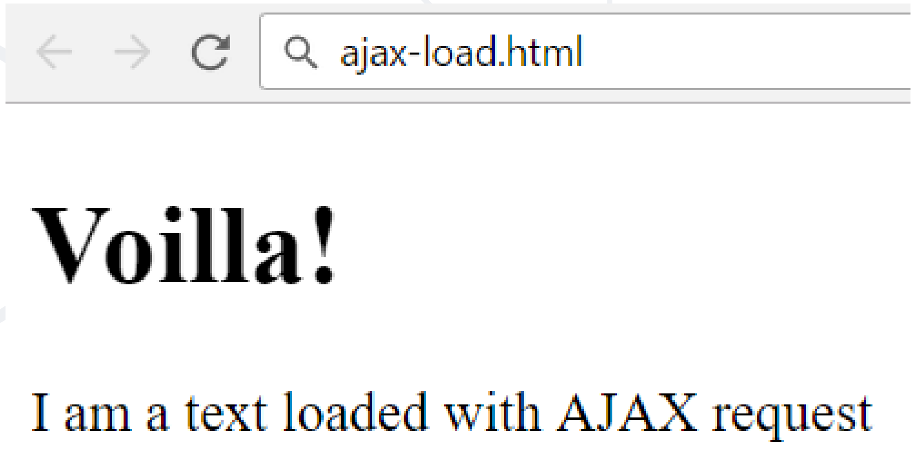
```
var newName = 'John Smith',
    xhr = new XMLHttpRequest();

xhr.open('POST', 'myservice/username?id=some-
unique-id');
xhr.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded');
xhr.onload = function() {
    if (xhr.status === 200 && xhr.responseText
!== newName) {
        alert('Something went wrong.  Name is
now ' + xhr.responseText);
    }
    else if (xhr.status !== 200) {
        alert('Request failed.  Returned status
of ' + xhr.status);
    }
};
xhr.send(encodeURI('name=' + newName));
```

jQuery AJAX

- Problem: create a **page** holding a **button**
  - Clicking the button should **load an html fragment** and display it inside a **div**


ajax-load.html

AJAX jQuery.load()


ajax-load.html

Voilla!

I am a text loaded with AJAX request

Solution: jQuery.load()

SoftUni Foundation

ajax-load.html

```
<div id="text">
  <h1>AJAX jQuery.load()</h1>
  <button onclick="loadTitle()">Load Title</button>
</div>
```

ajax-load.js

```
function loadTitle() {
  $('#text').load("text.html");
}
```

text.html

```
<h1>Voilla!</h1>
<p>I am a text loaded
with AJAX request</p>
```

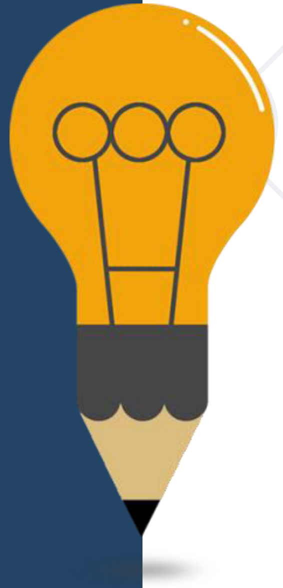
Check your solution here: https://judge.softuni.bg/Contests/357

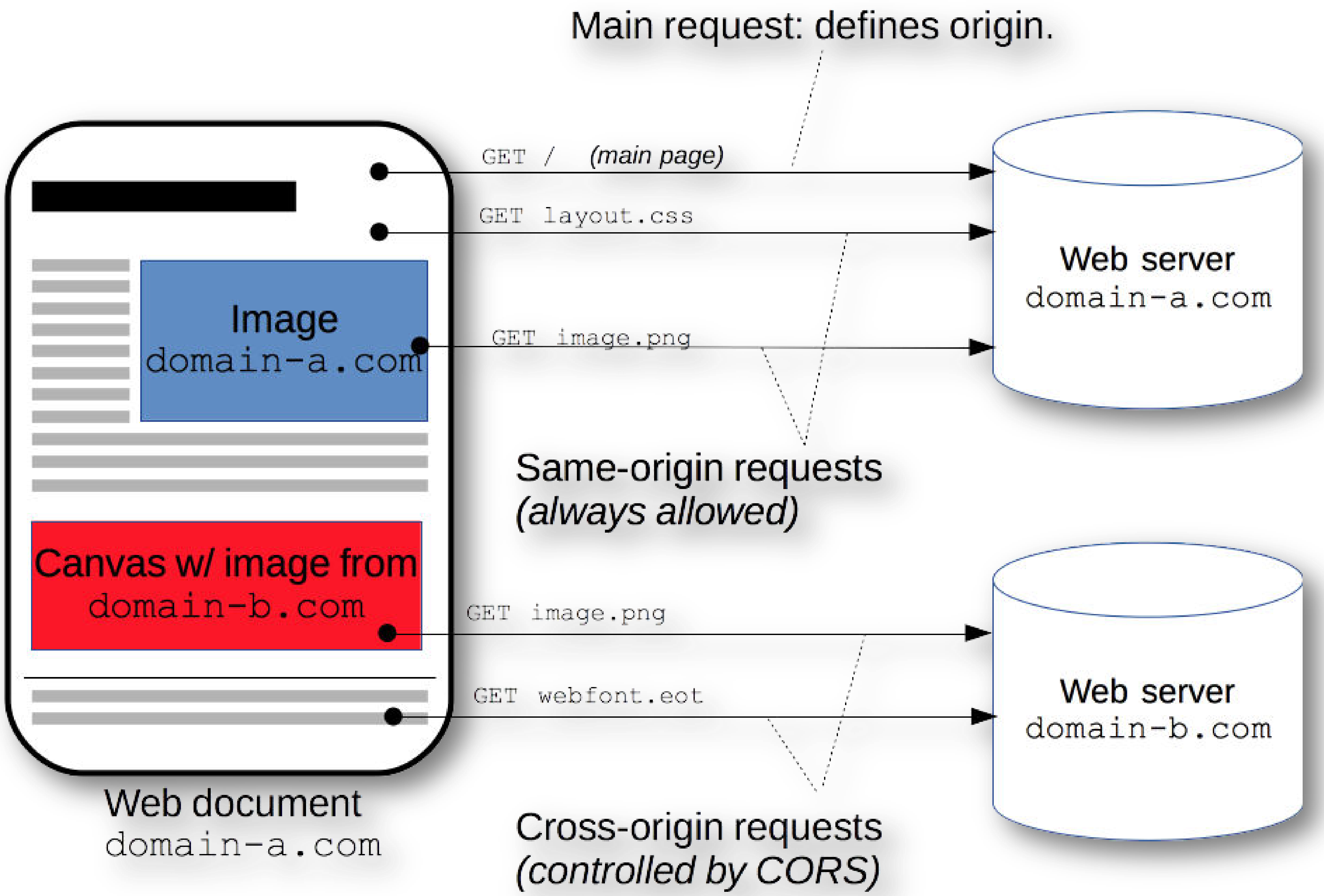
Submit in the judge the JS function **loadTitle()**

Same-origin policy / CORS

Same-origin policy

- The same-origin policy is a critical security mechanism that restricts how a document or script loaded from one origin can interact with a resource from another origin. It helps isolate potentially malicious documents, reducing possible attack vectors
- Two URLs have the same origin if the protocol, port (if specified), and host are the same for both
- Internet Explorer has two major exceptions to the same-origin policy
  - Trust Zones
  - Port

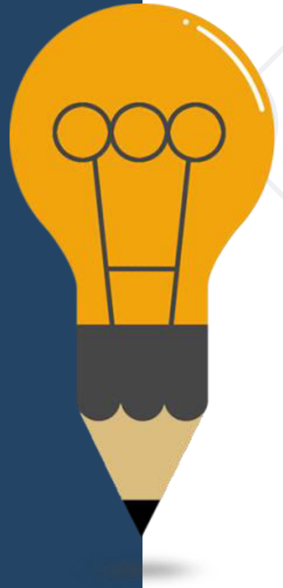



Main request: defines origin.
GET / (main page)
GET layout.css
GET image.png
Image
domain-a.com
Web server
domain-a.com
Same-origin requests
(always allowed)
Canvas w/ image from
domain-b.com
GET image.png
GET webfont.eot
Web server
domain-b.com
Web document
domain-a.com
Cross-origin requests
(controlled by CORS)

Cross-Origin Resource Sharing

SoftUni Foundation

- Cross-Origin Resource Sharing (CORS) is a mechanism that uses additional HTTP headers to tell a browser to let a web application running at one origin (domain) have permission to access selected resources from a server at a different origin
- Specification mandates that browsers "preflight" the request with an HTTP OPTIONS request method, and then, upon "approval" from the server, sending the actual request with the actual HTTP request method

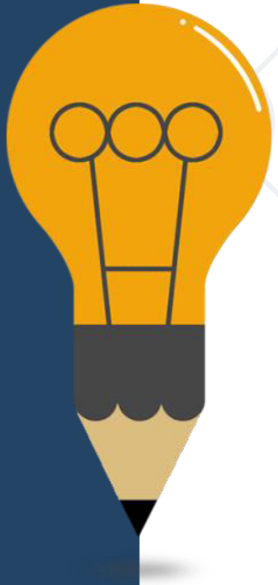


Internet Explorer

SoftUni Foundation

- Internet Explorer < 8 does not support cross domain ajax calls
- Internet Explorer 8 and 9 are using XDomainRequest object

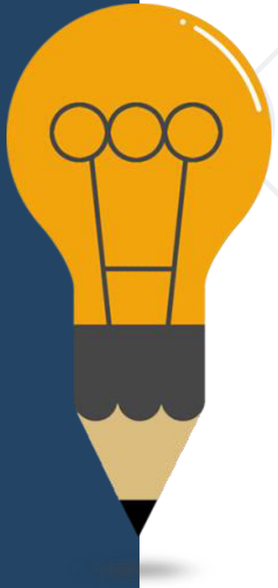
```
if(window.XDomainRequest){
  var xdr = new XDomainRequest();
  xdr.open("get", url);
  xdr.onprogress = function () { };
  xdr.ontimeout = function () { };
  xdr.onerror = function () { };
  xdr.onload = function() {
    console.log(xdr.responseText);
  };
  setTimeout(function () {xdr.send();}, 0);
}
```



Same-origin policy

- Scripts can execute AJAX requests
  - Either to their **origin** (same-origin policy)
  - Or when the remote server explicitly allows AJAX calls via CORS

Uses a special HTTP header: **Access-Control-Allow-Origin**

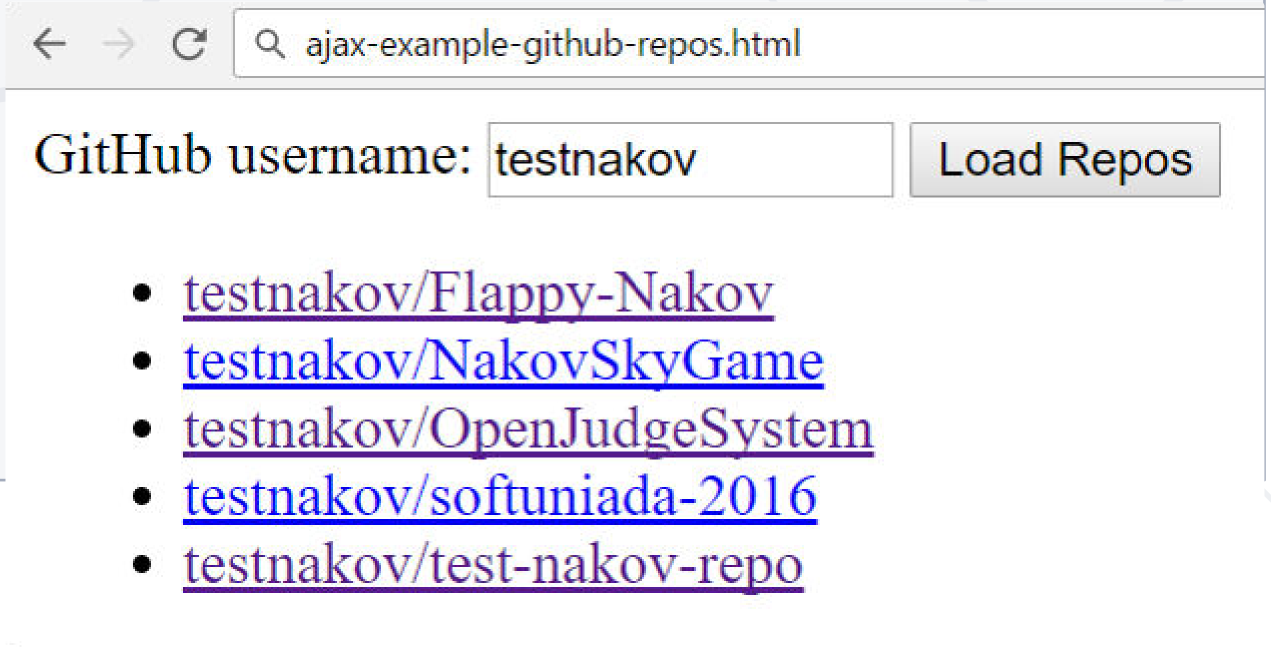


```
$(document).ajaxError(function(event, req, settings) {
  $('#text').text(`Error loading data: ${req.status}
    (${req.statusText})`);
});

function loadTitle() {
  $('#text').load("https://softuni.bg");
}
```

This cross-origin AJAX request will fail due to missing CORS headers

Problem: Load GitHub Repos with AJAX

```
GitHub username:
<input type="text" id="username" value="testnakov" />
<button onclick="loadRepos()">Load Repos</button>

<ul id="repos"></ul>

<script>
  function loadRepos() {
     // AJAX call …
  }
</script>
```


ajax-example-github-repos.html

GitHub username: testnakov Load Repos

- testnakov/Flappy-Nakov
- testnakov/NakovSkyGame
- testnakov/OpenJudgeSystem
- testnakov/softuniada-2016
- testnakov/test-nakov-repo

Solution: Load GitHub Repos with AJAX

SoftUni Foundation

```
function loadRepos() {
  $("#repos").empty();
  let url = "https://api.github.com/users/" +
    $("#username").val() + "/repos";
  $.ajax({ url,
    success: displayRepos,
    error: displayError
  });

  function displayRepos(respos) { … }

  function displayError(err) { … }
}
```

Solution: Load GitHub Repos with AJAX (2)

```
function displayRepos(respos) {
  for (let repo of respos) {
    let link = $('<a>').text(repo.full_name);
    link.attr('href', repo.html_url);
    $("#repos").append($('<li>').append(link));
  }
}

function displayError(err) {
  $("#repos").append($("<li>Error</li>"));
}
```

Check your solution here: https://judge.softuni.bg/Contests/357

jQuery
write less, do more.

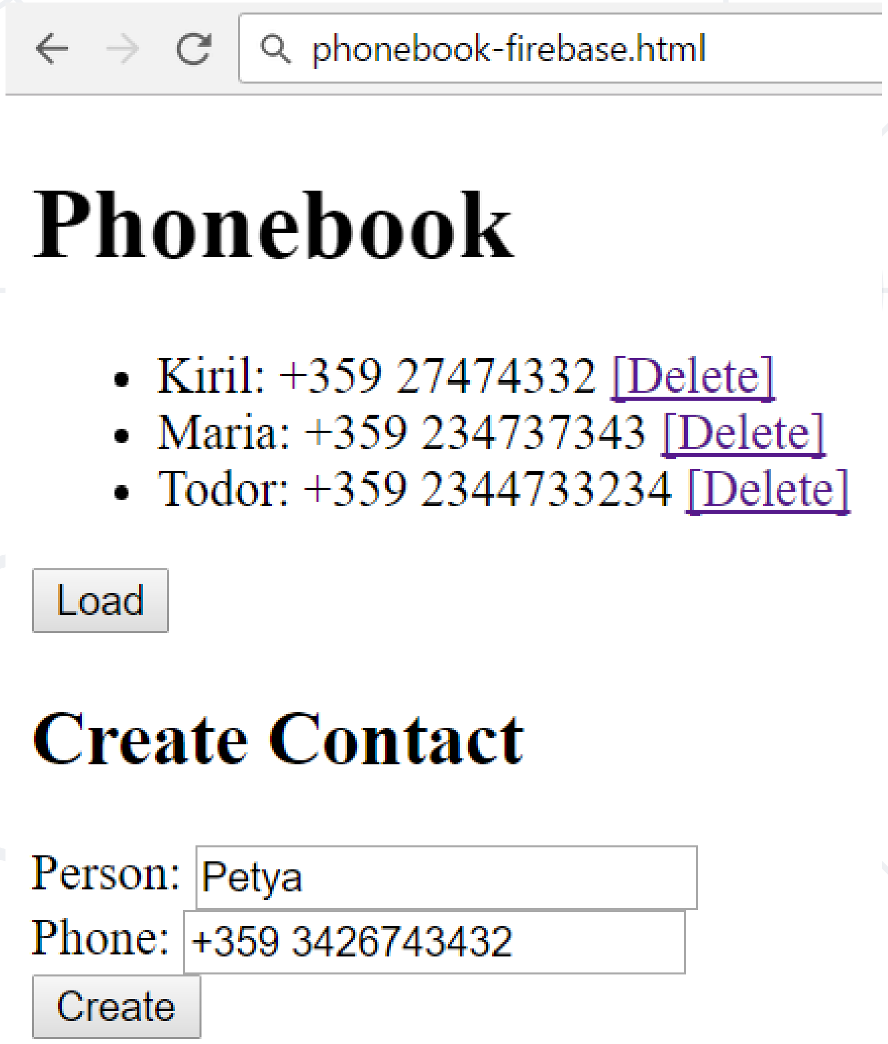
AJAX
Asynchronous Javascript And XML

AJAX – Examples

Using jQuery to Access REST APIs

Problem: Phonebook App in Firebase

- Create a mini **phonebook JS front-end app**
  - Hold your data in **Firebase**
    - Disable the authentication to simplify your work
  - Implement "**list phones**", "**add phone**", "**delete phone**"



phonebook-firebase.html

Phonebook

- Kiril: +359 27474332 [Delete]
- Maria: +359 234737343 [Delete]
- Todor: +359 2344733234 [Delete]

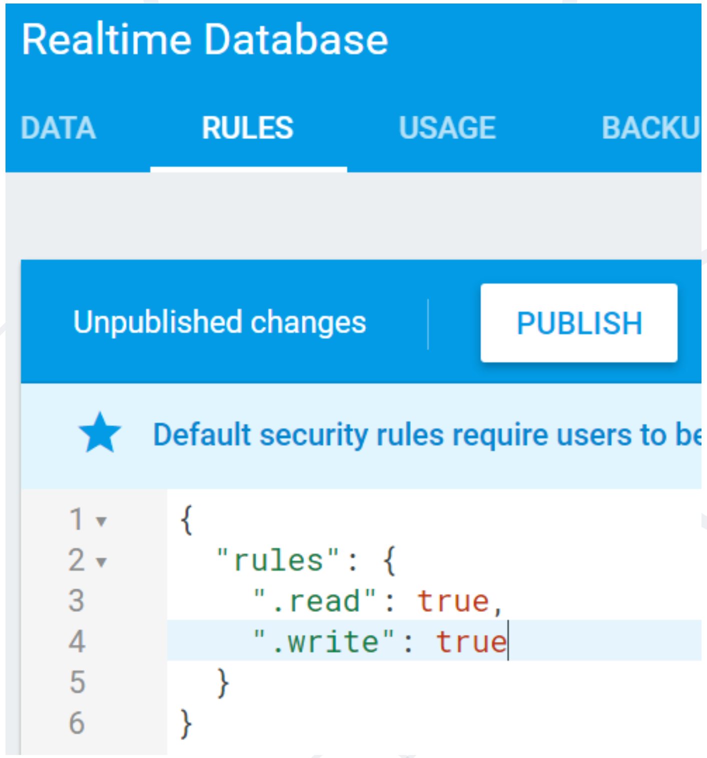
Load

Create Contact

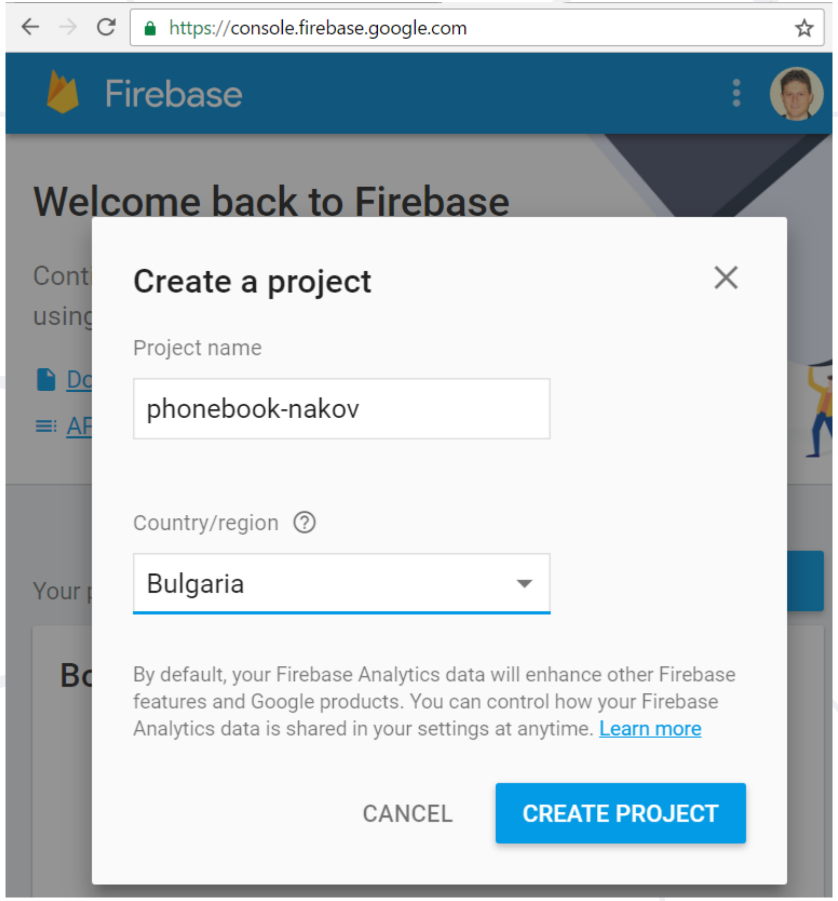
Person: Petya
Phone: +359 3426743432
Create

Solution: Setup a Firebase DB

- Open https://console.firebase.google.com/
- Create a new project
- Enable public read / write access



```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

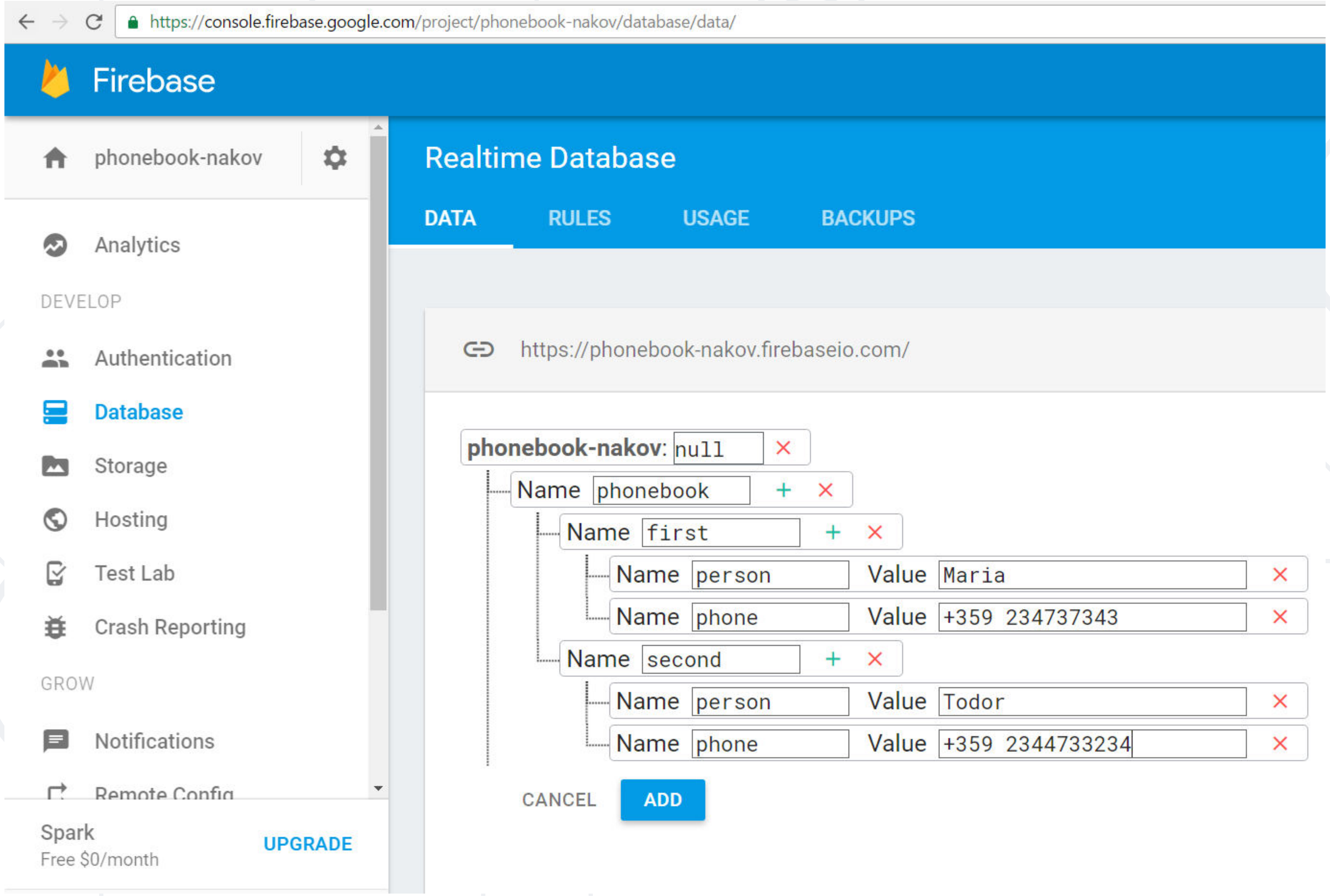


Project name: phonebook-nakov

Country/region: Bulgaria

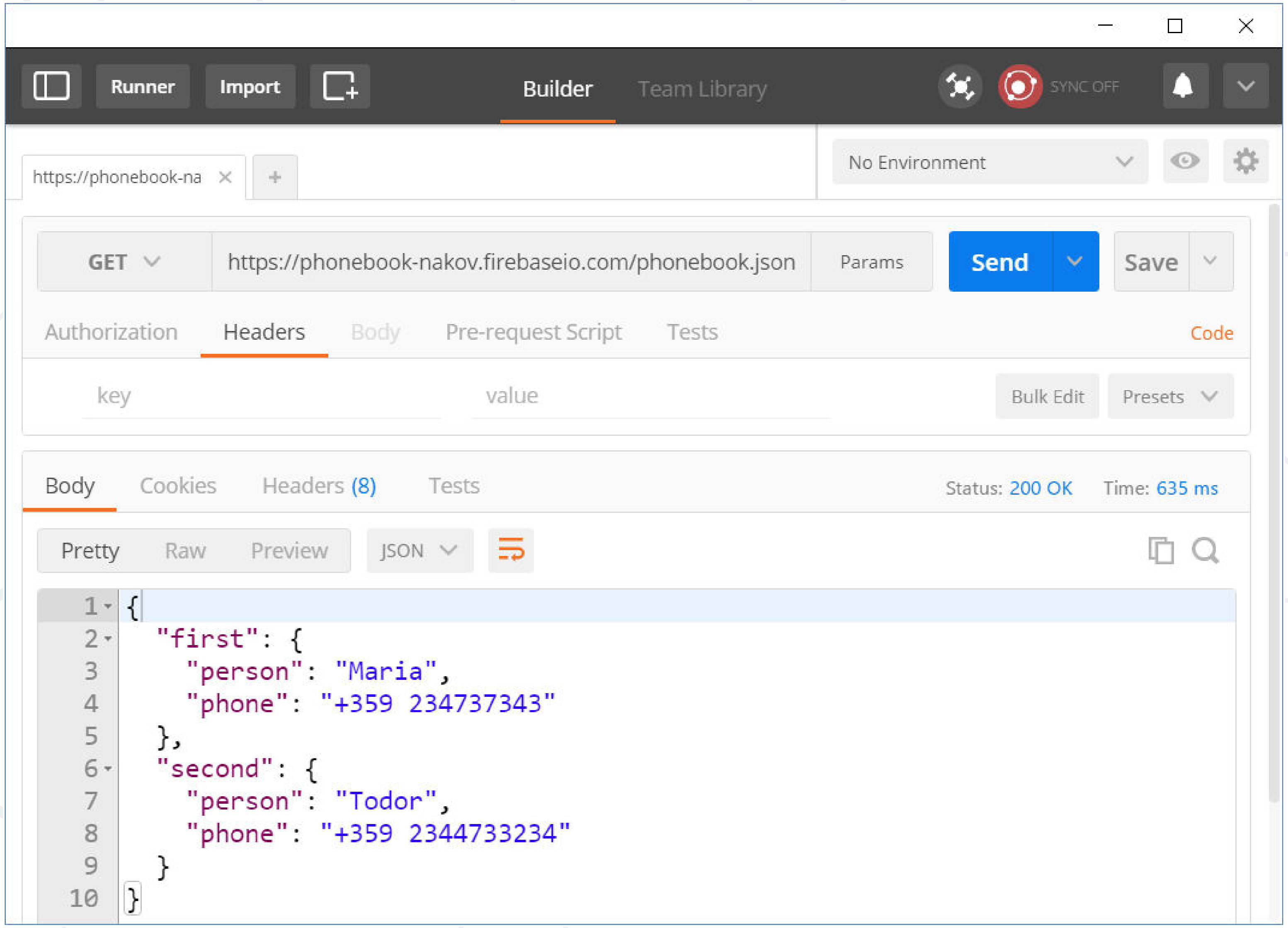
By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at anytime. Learn more

Solution: Add Sample Data in Firebase



```
phonebook-nakov: null
  Name phonebook
    Name first
      Name person   Value Maria
      Name phone    Value +359 234737343
    Name second
      Name person   Value Todor
      Name phone    Value +359 2344733234
```

Solution: Test Your REST Service



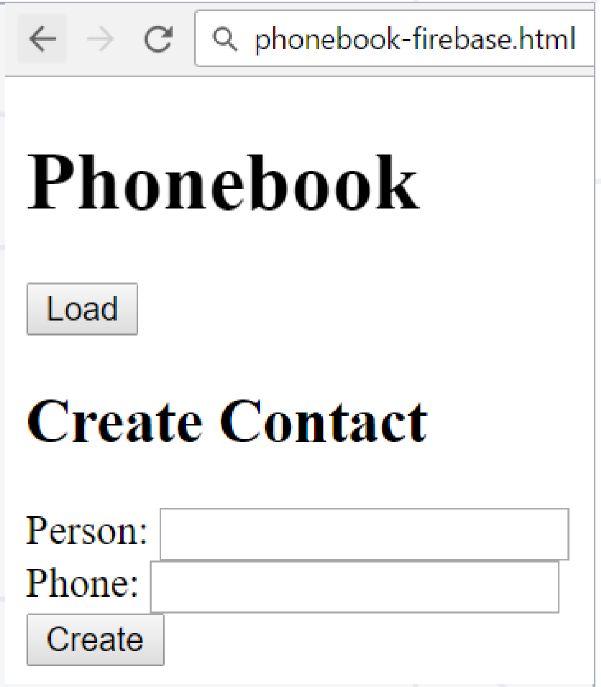
GET https://phonebook-nakov.firebaseio.com/phonebook.json

Status: 200 OK Time: 635 ms

```
{
  "first": {
    "person": "Maria",
    "phone": "+359 234737343"
  },
  "second": {
    "person": "Todor",
    "phone": "+359 2344733234"
  }
}
```

Solution: Phonebook in Firebase – HTML

```html
<h1>Phonebook</h1>
<ul id="phonebook"></ul>
<button id="btnLoad">Load</button>

<h2>Create Contact</h2>
Person: <input type="text" id="person" />
<br>
Phone: <input type="text" id="phone" />
<br>
<button id="btnCreate">Create</button>
```



Solution: Phonebook in Firebase – JS Code

```
$(function () {
  $('#btnLoad').click(loadContacts);
  $('#btnCreate').click(createContact);
  let baseServiceUrl =
    'https://phonebook-nakov.firebaseio.com/phonebook';
  function loadContacts() { … }
  function displayContacts(contacts) { … }
  function displayError(err) { … }
  function createContact() { … }
  function deleteContact(key) { … }
});
```

Solution: Phonebook in Firebase – JS Code

```
function loadContacts() {
  $("#phonebook").empty();
  $.get(baseServiceUrl + '.json')
    .then(displayContacts)
    .catch(displayError);
}

function displayError(err) {
  $("#phonebook").append($("<li>Error</li>"));
}
```

Solution: Phonebook in Firebase – JS Code

```
function displayContacts(contacts) {
  for (let key in contacts) {
    let person = contacts[key]['person'];
    let phone = contacts[key]['phone'];
    let li = $("<li>");
    li.text(person + ': ' + phone + ' ');
    $("#phonebook").append(li);
    li.append($("<button>Delete</button>")
      .click(deleteContact.bind(this, key)));
  }
}
```

Bind the event handler with the **current key**

Solution: Phonebook in Firebase – JS Code

SoftUni Foundation

```
function createContact() {
  let newContactJSON = JSON.stringify({
    person: $('#person').val(),
    phone: $('#phone').val()
  });
  $.post(baseServiceUrl + '.json', newContactJSON)
    .then(loadContacts)
    .catch(displayError);
  $('#person').val('');
  $('#phone').val('');
}
```

Solution: Phonebook in Firebase – JS Code

```
function deleteContact(key) {
  let request = {
    method: 'DELETE',
    url: baseServiceUrl + '/' + key + '.json'
  };
  $.ajax(request)
    .then(loadContacts)
    .catch(displayError);
}
```

The correct contact **key** will come as parameter (due to binding)

Check your solution here: https://judge.softuni.bg/Contests/357

Practice: jQuery AJAX

Live Exercises in Class (Lab)

Summary

- AJAX sends asynchronous HTTP requests from JS
- jQuery simplifies how developers use AJAX

```
$.ajax({ url, method: 'GET',
  success: displayRepos,
  error: displayError
});
function displayRepos(respos) { … }
function displayError(err) { … }
}
```

Questions?

SoftUni

Software University

SoftUni Svetlina

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

SoftUni Diamond Partners

SoftUni Foundation

XSsoftware

SBTech we know sports

telenor

SoftwareGroup doing it right

NETPEAK

SmartIT

Postbank Решения за твоето утре

SUPER HOSTING ®.BG

INDEAVR Serving the high achievers

INFRAGISTICS®

LIEBHERR

æternity

codexio

SoftUni Organizational Partners

SoftUni Foundation

ИНФОРМАЦИОННО ОБСЛУЖВАНЕ

OneBit SOFTWARE

WORLD OF MYTHS

Lukanet.com

codexio

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License