



AJAX and jQuery AJAX

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

XMLHttpRequest



Au Mau Duong
Technical Trainers

Table of Contents

- 1. AJAX Concepts
- 2. XMLHttpRequest

3. jQuery AJAX

- **\$.load**
- **\$.get() / \$.post()**
- **\$.ajax()**
- AJAX – Examples
 - Accessing Firebase with AJAX

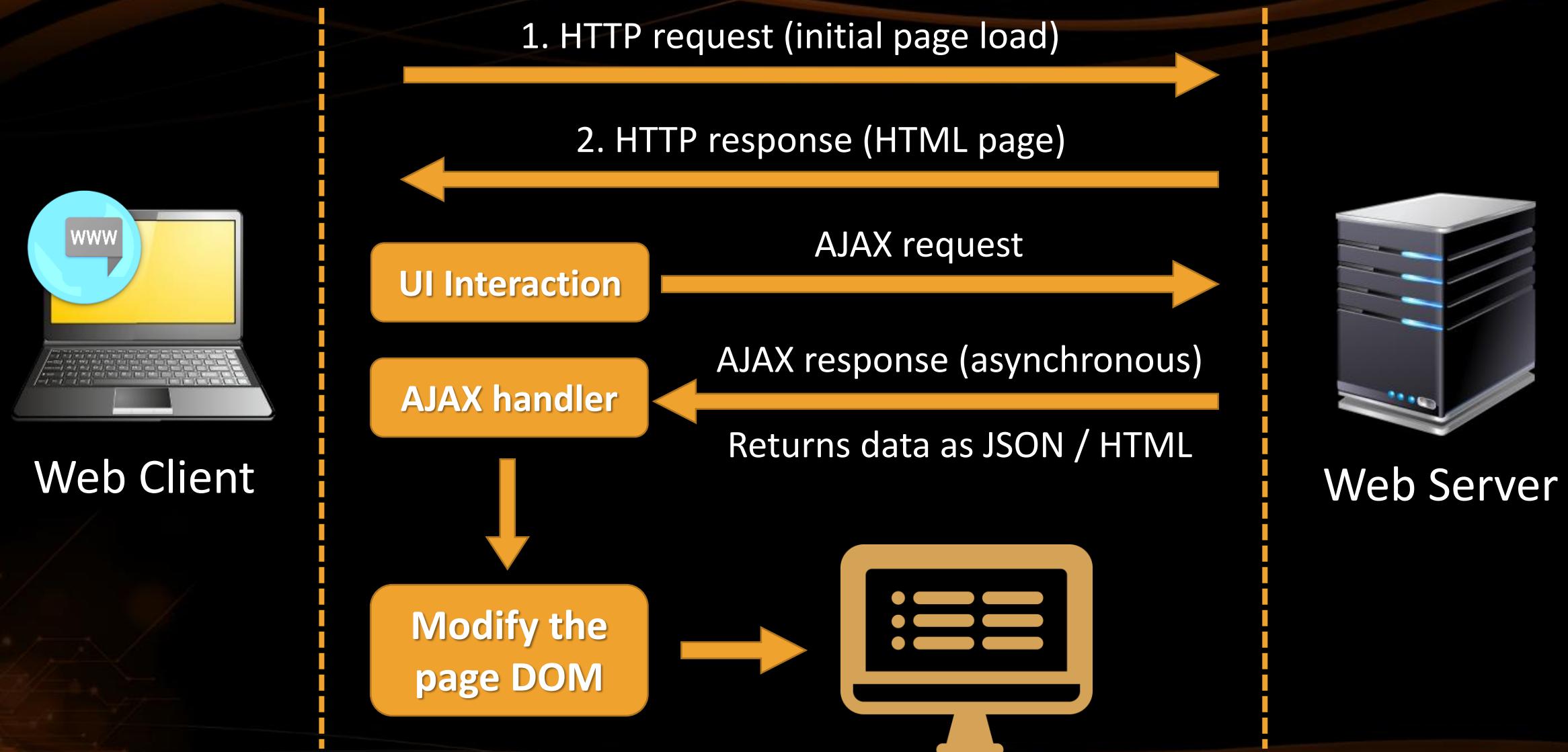


What is AJAX?

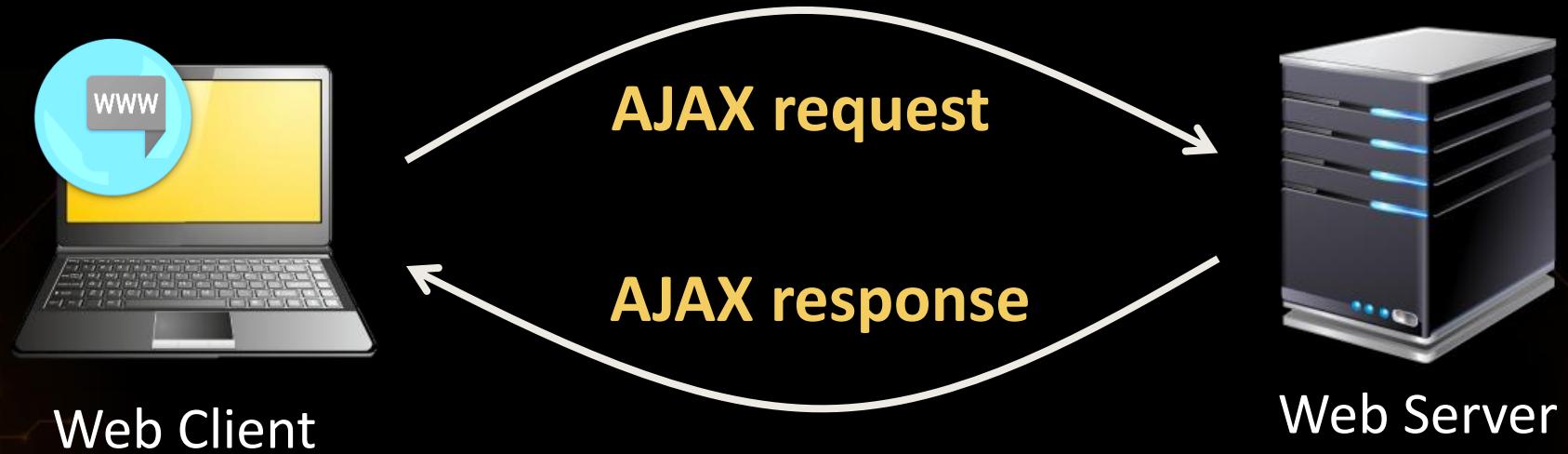
- **AJAX == Asynchronous JavaScript And XML**
 - Technique for background loading of dynamic content / data
 - Web pages / apps load data from the Web server and render it
- Two types of AJAX
 - **Partial page rendering**
 - Load HTML fragment + show it in a `<div>`
 - **JSON service**
 - Load JSON object and display it with JavaScript / jQuery



AJAX: Workflow



XMLHttpRequest



Using the XMLHttpRequest Object

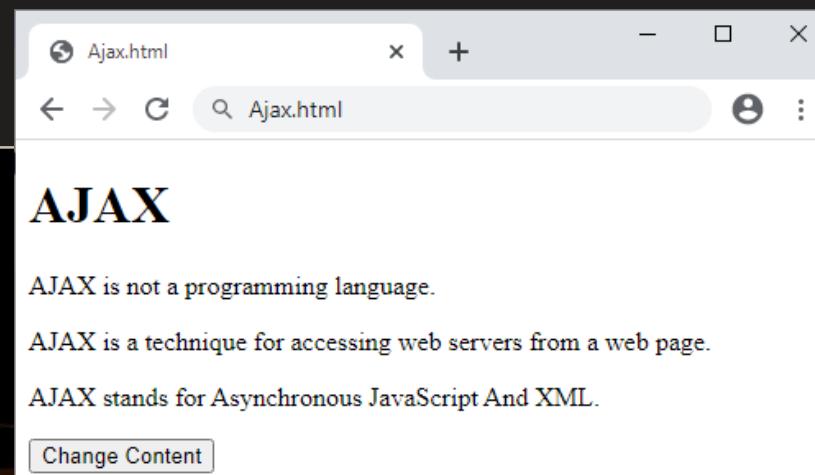
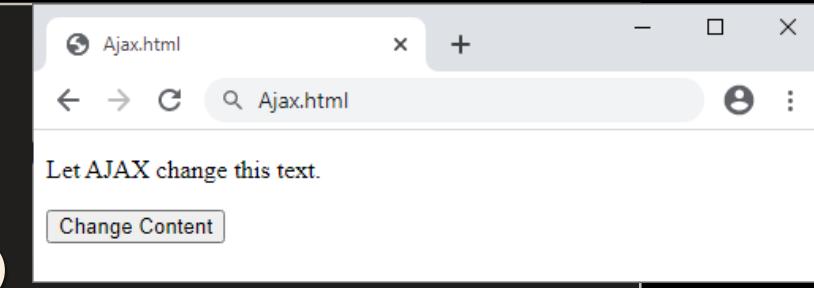
XMLHttpRequest – Standard API for AJAX

```
<p id="demo">Let AJAX change this text.</p>
<button type="button" onclick="loadDoc()">Change Content</button>

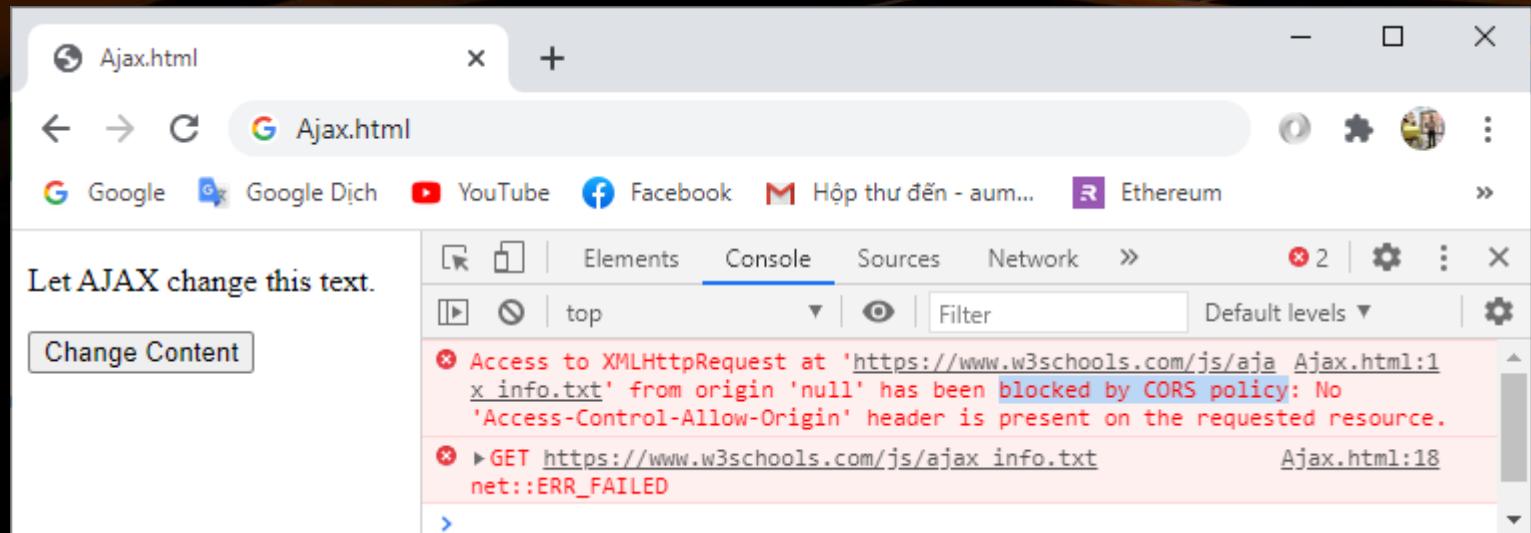
function loadDoc() {
    let req = new XMLHttpRequest();
    req.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200)
            document.getElementById("demo").innerHTML =
                this.responseText;
    }
    req.open("GET","https://www.w3schools.com/js/ajax_info.txt", true);
    req.send();
}
```

executed when the
readyState changes

Server Response



XMLHttpRequest – blocked by CORS policy



How to fix?

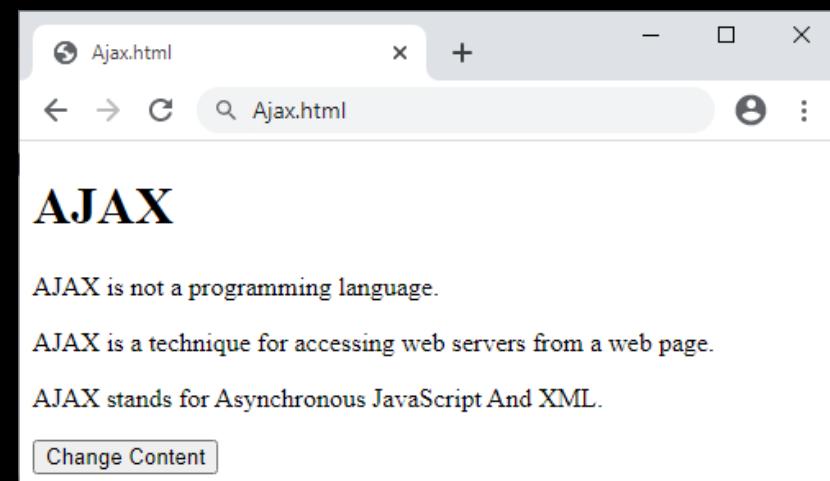
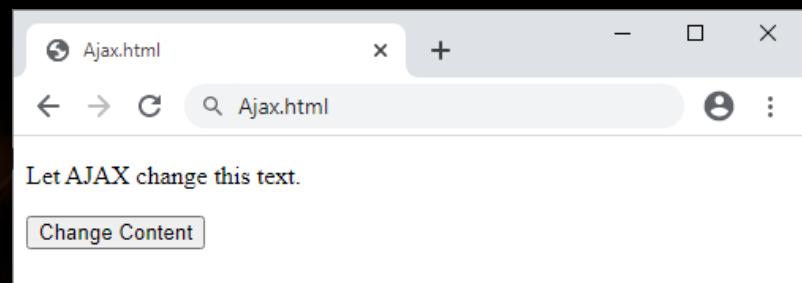
```
Window: "C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" --  
disable-web-security --disable-gpu --user-data-dir=~/chromeTemp
```

```
OSX: open -n -a /Applications/Google\ Chrome.app/Contents/MacOS/Google\  
Chrome --args --user-data-dir="/tmp/chrome_dev_test" --disable-web-  
security
```

XMLHttpRequest – Synchronous Request

```
<p id="demo">Let AJAX change this text.</p>
<button type="button" onclick="loadDoc()">Change Content</button>
```

```
function loadDoc() {
  let req = new XMLHttpRequest();
  req.open("GET", "https://www.w3schools.com/js/ajax_info.txt", false);
  req.send();
}
```



AJAX XML Example (1)

```
function loadDoc() {  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        if (this.readyState == 4 && this.status == 200) {  
            myFunction(this);  
        }  
    };  
    xhttp.open("GET", "https://www.w3schools.com/js/cd_catalog.xml", true);  
    xhttp.send();  
}
```

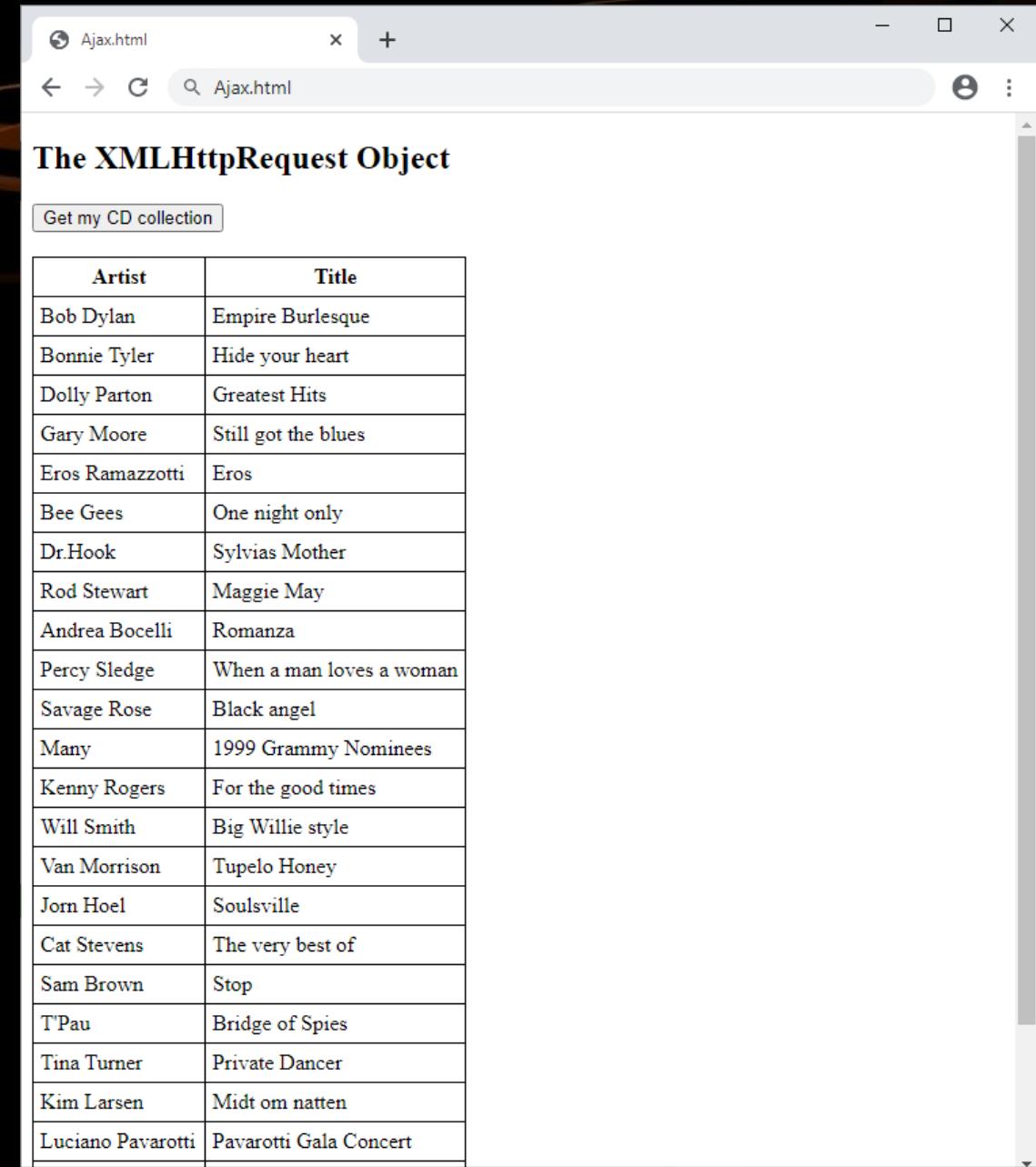
AJAX XML Example (2)

```
function myFunction(xml) {  
    var i;  
    var xmlDoc = xml.responseXML;  
    var table=<tr><th>Artist</th><th>Title</th></tr>;  
    var x = xmlDoc.getElementsByTagName("CD");  
    for (i = 0; i <x.length; i++) {  
        table += "<tr><td>" +  
            x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +  
            "</td><td>" +  
            x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +  
            "</td></tr>";  
    }  
    document.getElementById("demo").innerHTML = table;  
}
```

AJAX XML Example (3)

```
<h2>The XMLHttpRequest Object</h2>
<button type="button"
onclick="loadDoc()">Get my CD
collection</button><br><br>
<table id="demo"></table>
```

```
<style>
table,th,td {
    border : 1px solid black;
    border-collapse: collapse;
}
th,td {
    padding: 5px;
}
</style>
```

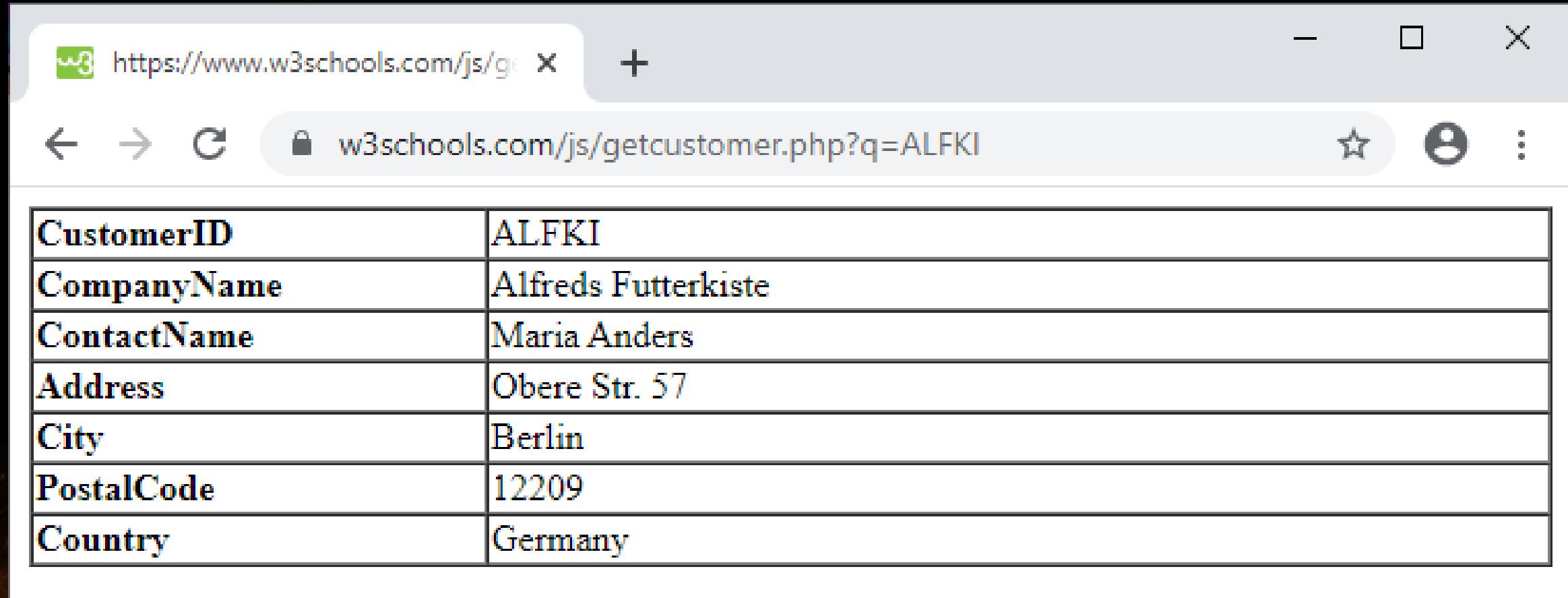


The screenshot shows a web browser window titled "Ajax.html" with the URL "Ajax.html". The page content is titled "The XMLHttpRequest Object" and contains a button labeled "Get my CD collection". Below the button is a table with two columns: "Artist" and "Title". The table lists various artists and their album titles. The browser interface includes standard controls like back, forward, and search.

Artist	Title
Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr.Hook	Sylvias Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey
Jorn Hoel	Soulsville
Cat Stevens	The very best of
Sam Brown	Stop
T'Pau	Bridge of Spies
Tina Turner	Private Dancer
Kim Larsen	Midt om natten
Luciano Pavarotti	Pavarotti Gala Concert

AJAX Database Example (1)

Link: <https://www.w3schools.com/js/getcustomer.php?q=ALFKI>



The screenshot shows a web browser window with the URL [w3schools.com/js/getcustomer.php?q=ALFKI](https://www.w3schools.com/js/getcustomer.php?q=ALFKI) in the address bar. The page displays a table with customer information:

CustomerID	ALFKI
CompanyName	Alfreds Futterkiste
ContactName	Maria Anders
Address	Obere Str. 57
City	Berlin
PostalCode	12209
Country	Germany

AJAX Database Example (2)

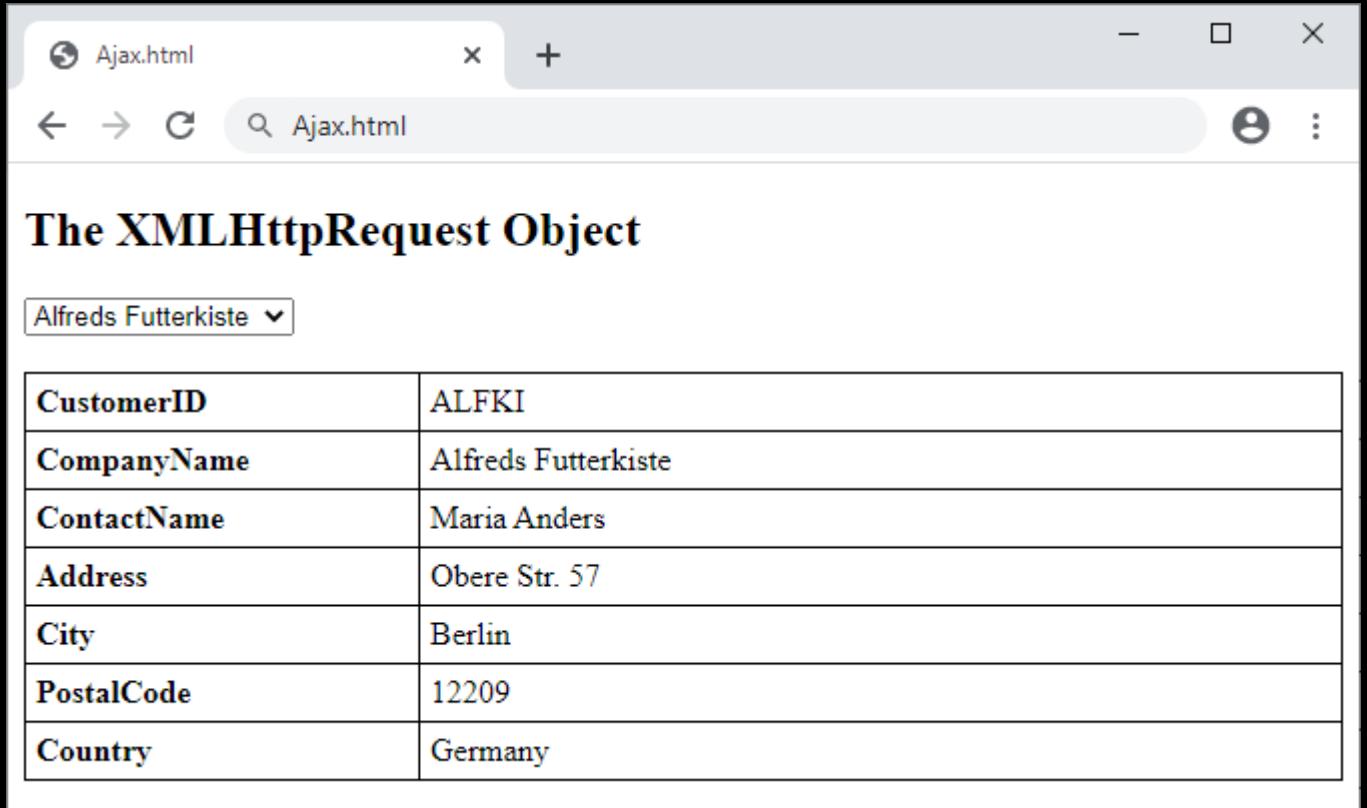
```
function showCustomer(str) {  
    var xhttp;  
    if (str == "") {  
        document.getElementById("txtHint").innerHTML = "";  
        return;  
    }  
    xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function() {  
        if (this.readyState == 4 && this.status == 200) {  
            document.getElementById("txtHint").innerHTML = this.responseText;  
        }  
    };  
    xhttp.open("GET", "https://www.w3schools.com/js/getcustomer.php?q="+str,  
true);  
    xhttp.send();  
}
```

AJAX Database Example (3)

```
<h2>The XMLHttpRequest Object</h2>
<form action="">
  <select name="customers" onchange="showCustomer(this.value)">
    <option value="">Select a customer:</option>
    <option value="ALFKI">Alfreds Futterkiste</option>
    <option value="NORTS">North/South</option>
    <option value="WOLZA">Wolski Zajazd</option>
  </select>
</form>
<br>
<div id="txtHint">Customer info will be listed here...</div>
```

AJAX Database Example (4)

```
<style>
table,th,td {
    border : 1px solid black;
    border-collapse: collapse;
}
th,td {
    padding: 5px;
}
</style>
```



The screenshot shows a web browser window titled "Ajax.html". The page content is titled "The XMLHttpRequest Object". A dropdown menu is open, showing the value "Alfreds Futterkiste". Below the dropdown is a table with the following data:

CustomerID	ALFKI
CompanyName	Alfreds Futterkiste
ContactName	Maria Anders
Address	Obere Str. 57
City	Berlin
PostalCode	12209
Country	Germany

JQUERY AJAX Database Example (1)

Link: <https://www.w3schools.com/js/getcustomer.php?q=ALFKI>



The screenshot shows a web browser window with the URL [w3schools.com/js/getcustomer.php?q=ALFKI](https://www.w3schools.com/js/getcustomer.php?q=ALFKI) in the address bar. The page displays a table with customer information:

CustomerID	ALFKI
CompanyName	Alfreds Futterkiste
ContactName	Maria Anders
Address	Obere Str. 57
City	Berlin
PostalCode	12209
Country	Germany

AJAX Database Example (2)

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></s
cript>
<script>
$(document).ready(function(){
    $("select").change(function(){
        console.log($(this).text())
        $.get("https://www.w3schools.com/js/getcustomer.php?q="+$(this).val(),
function(data, status){
            document.getElementById("txtHint").innerHTML = data;
        });
    });
});
</script>
```

AJAX Database Example (3)

```
<h2>The XMLHttpRequest Object</h2>
<form action="">
  <select name="customers" onchange="showCustomer(this.value)">
    <option value="">Select a customer:</option>
    <option value="ALFKI">Alfreds Futterkiste</option>
    <option value="NORTS">North/South</option>
    <option value="WOLZA">Wolski Zajazd</option>
  </select>
</form>
<br>
<div id="txtHint">Customer info will be listed here...</div>
```

AJAX Database Example (4)

```
<style>
table,th,td {
    border : 1px solid black;
    border-collapse: collapse;
}
th,td {
    padding: 5px;
}
</style>
```



The screenshot shows a web browser window titled "Ajax.html". The page content is titled "The XMLHttpRequest Object". A dropdown menu is open, showing the value "Alfreds Futterkiste". Below the dropdown is a table with the following data:

CustomerID	ALFKI
CompanyName	Alfreds Futterkiste
ContactName	Maria Anders
Address	Obere Str. 57
City	Berlin
PostalCode	12209
Country	Germany

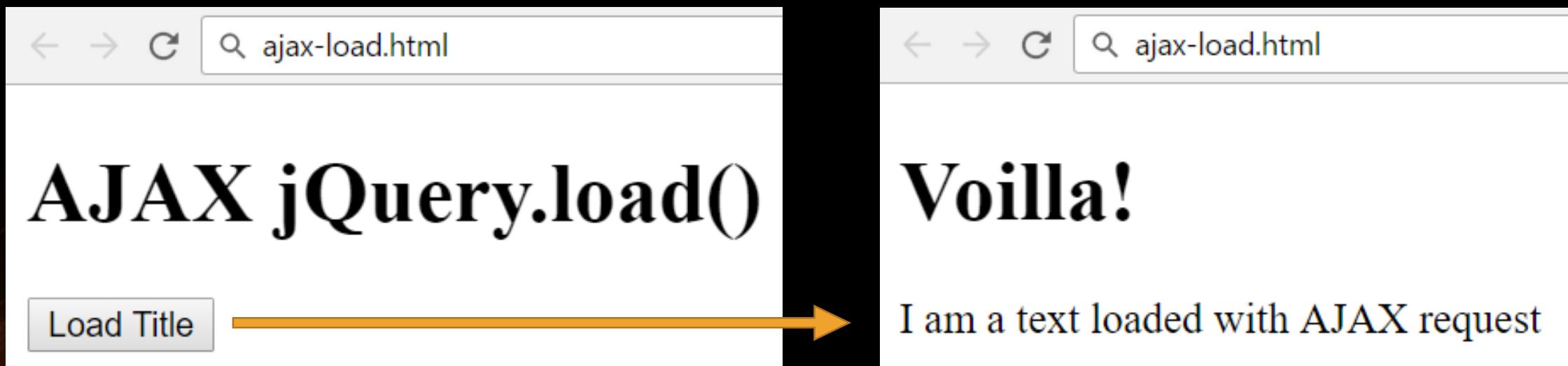


jQuery AJAX

Simplified AJAX Calls with jQuery

jQuery AJAX

- jQuery dramatically simplifies how developer make AJAX calls
- Problem: create a page holding a button
 - Clicking the button should load a html fragment and display it inside a **div**



Solution: jQuery.load()

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>
```

Handling Errors

- 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("http://dir.bg");  
}
```

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

Examples



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"

The screenshot shows a web browser window with the URL 'phonebook-firebase.html' in the address bar. The page itself has a dark background with orange and yellow circuit board patterns.

Phonebook

- Kiril: +359 27474332 [\[Delete\]](#)
- Maria: +359 234737343 [\[Delete\]](#)
- Todor: +359 2344733234 [\[Delete\]](#)

Create Contact

Person:

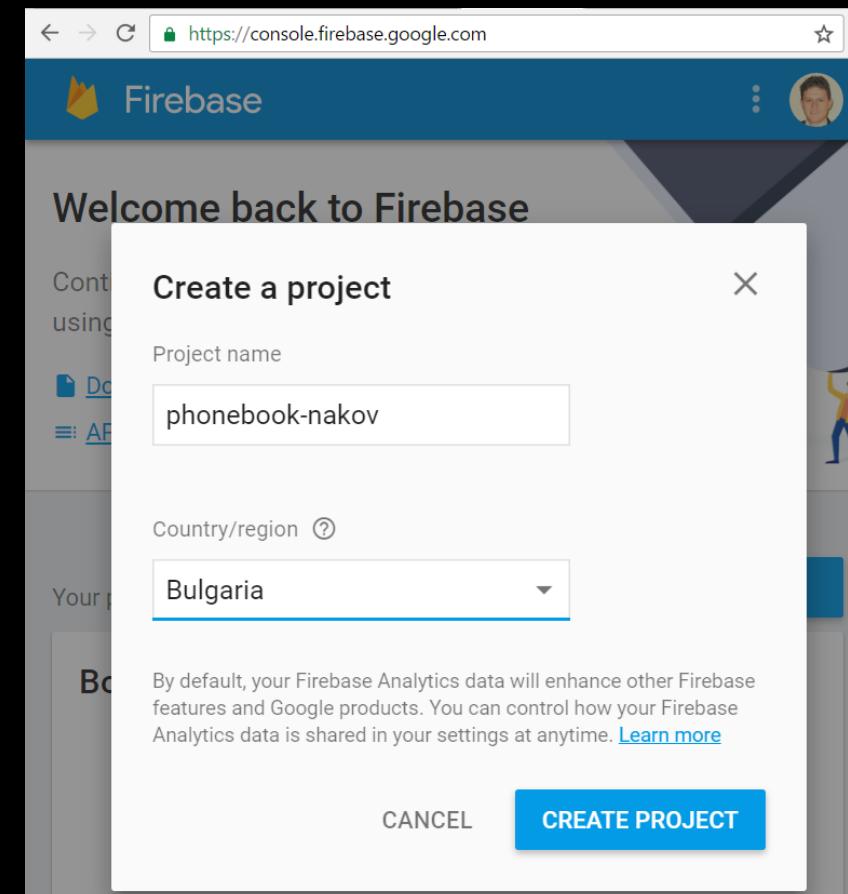
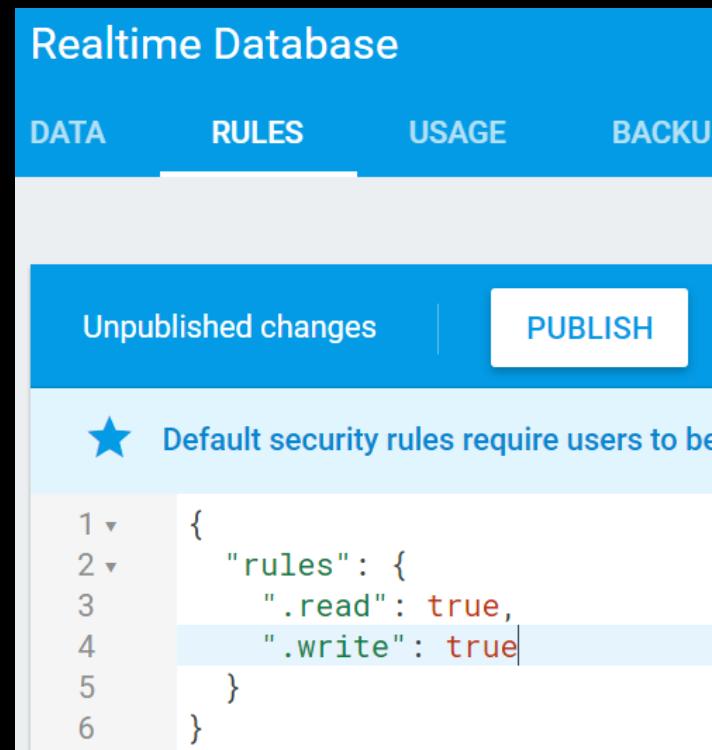
Phone:

Solution: Setup a Firebase DB

- Open <https://console.firebaseio.google.com/>

- ## ■ Create a new project

- Enable public read / write access



Solution: Add Sample Data in Firebase

The screenshot shows the Firebase Realtime Database console interface. The URL in the browser is <https://console.firebaseio.google.com/project/phonebook-nakov/database/data/>. The project name is "phonebook-nakov". The left sidebar includes links for Analytics, Authentication, Database (which is selected), Storage, Hosting, Test Lab, Crash Reporting, Notifications, Remote Config, Spark (Free \$0/month), and an UPGRADE button. The main area is titled "Realtime Database" with tabs for DATA, RULES, USAGE, and BACKUPS. A preview URL <https://phonebook-nakov.firebaseio.com/> is shown. The data structure under "phonebook-nakov" is being edited:

- Root level: "phonebook-nakov": null
- Child "phonebook":
 - Child "first":
 - Child "person": Name "Maria", Value "Maria"
 - Child "phone": Name "+359 234737343", Value "+359 234737343"
 - Child "second":
 - Child "person": Name "Todor", Value "Todor"
 - Child "phone": Name "+359 2344733234", Value "+359 2344733234"

At the bottom, there are "CANCEL" and "ADD" buttons.

Solution: Test Your REST Service

The screenshot shows the Postman application interface. At the top, there are tabs for 'Runner' and 'Import'. Below that is a search bar with the URL 'https://phonebook-nakov.firebaseio.com/phonebook.json'. To the right of the URL are buttons for 'Sync Off' and notifications. The main area shows a 'Builder' tab selected. Underneath it, there's a 'Send' button and a 'Save' button. Below these are tabs for 'Authorization', 'Headers' (which is selected), 'Body', 'Pre-request Script', 'Tests', and 'Code'. The 'Headers' section has a table with columns 'key' and 'value'. The 'Body' section shows the response from a GET request to the specified URL. The response status is '200 OK' and the time taken is '635 ms'. The body content is displayed in 'Pretty' format, showing a JSON object with two entries: 'first' and 'second', each containing a person's name and phone number.

key	value

Body

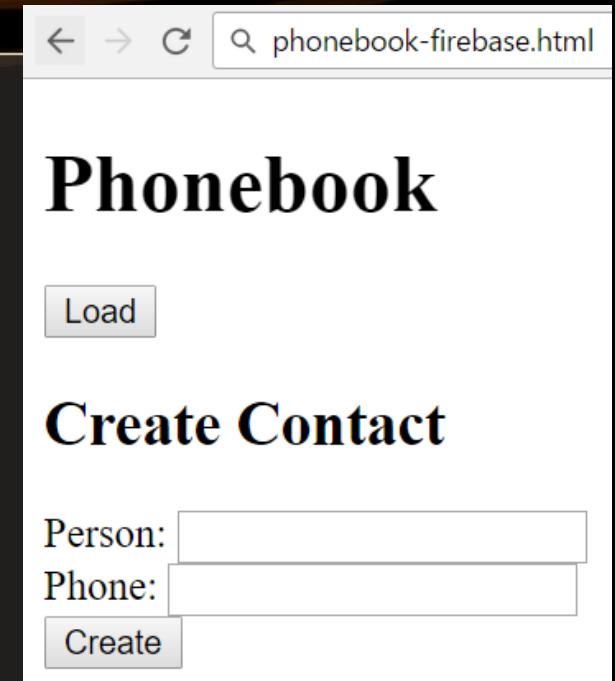
Pretty Raw Preview JSON

```
1 {  
2   "first": {  
3     "person": "Maria",  
4     "phone": "+359 234737343"  
5   },  
6   "second": {  
7     "person": "Todor",  
8     "phone": "+359 2344733234"  
9   }  
10 }
```

Solution: Phonebook in Firebase – 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

```
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)

Summary

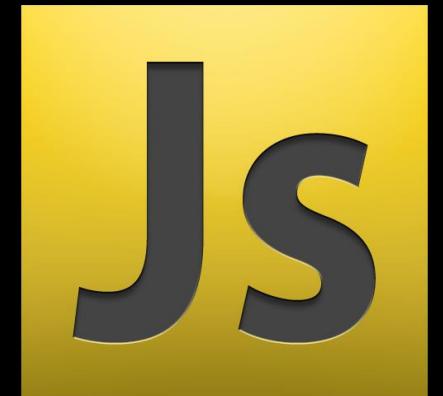
- AJAX sends asynchronous HTTP requests from JS
- jQuery simplifies how developers use AJAX

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

function displayRepos(respos) { ... }

function displayError(err) { ... }

}
```



AJAX and jQuery AJAX



Questions?

