



Asynchronous JavaScript and XML (AJAX)

Αθανάσιος Ανδρούτσος



Web-based Εφαρμογές (1)

AJAX

- Οι web εφαρμογές αποτελούνται από δύο βασικά μέρη (End Points)
 - Το **Front-End** στην πλευρά του χρήστη
 - Το **Back-End** που τυπικά βρίσκεται πίσω από ένα Web Server
- Η επικοινωνία γίνεται μέσω του πρωτοκόλλου HTTP (Hypertext Transfer Protocol)





Web-based Εφαρμογές (2)

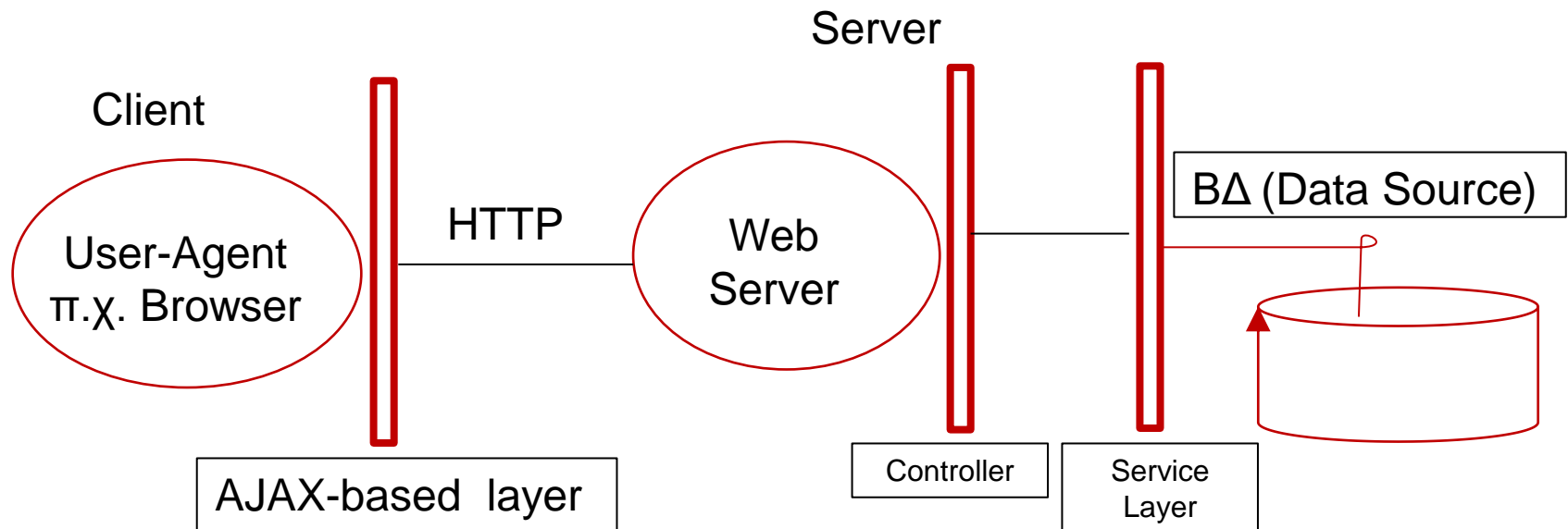
AJAX

- Στο front-end η JavaScript μας παρέχει ένα API (βιβλιοθήκη) που υλοποιεί τη δυνατότητα επικοινωνίας με τον Server
- Δηλαδή ένα API για να μπορούμε να κάνουμε ασύγχρονα calls προς τον server και να αποστέλλουμε JSON strings με δεδομένα καθώς και να λαμβάνουμε πίσω από τον Server πάλι JSON strings με τα αποτελέσματα



AJAX

AJAX



- Ο AJAX (Asynchronous JavaScript and XML), είναι ένας μηχανισμός (βιβλιοθήκη) για να στέλνουμε asynchronous requests σε web servers και να λαμβάνουμε **data** χωρίς να χρειάζεται να κάνουμε reload όλη τη web σελίδα



XMLHttpRequest

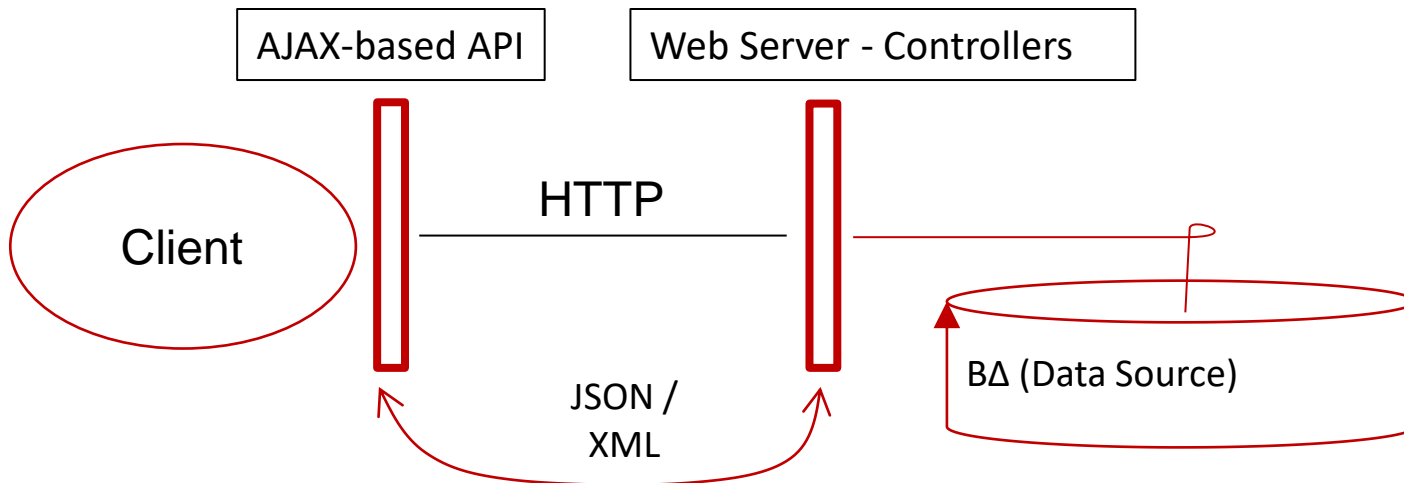
AJAX

- Το βασικό object που μας παρέχει η AJAX είναι το **XMLHttpRequest** που δίνει τη δυνατότητα στον client να κάνει **AJAX requests** προς τον server
- Καθώς επίσης και να λάβει πίσω από τον Server το **response**
- Το οποίο στη συνέχεια μπορεί να το επεξεργαστεί



JSON ή XML Objects

- Μέσω του **XMLHttpRequest** μπορούμε να αποστείλουμε και να λάβουμε δεδομένα σε μορφή **XML** ή/και κυρίως **JSON** (JavaScript Object Notation)
- Τα JSON Objects ή τα XML Objects χρησιμοποιούνται για τη μεταφορά πληροφοριών από τον χρήστη στον Web Server και το αντίστροφο





HTTP (1)

- Το βασικό πρωτόκολλο επικοινωνίας ανάμεσα στον Web Server και τον User-Agent (π.χ. Browser) είναι το HTTP (Hypertext Transfer Protocol)
- Γιαυτό οι Web Servers είναι τεχνικά HTTP Servers
- Το HTTP ορίζει τρόπους επικοινωνίας ανάμεσα στον Web Server και τον User-Agent μέσω της παροχής μεθόδων από τον HTTP Server, όπως *GET*, *POST*, *PUT*, *DELETE*
- Για να επικοινωνήσει ο client με τον HTTP Server χρειάζεται να γνωρίζει το URI (URL) του πόρου στον Web Server



URL Paths (1)

- Για να επικοινωνήσει ένας user-agent με ένα web server θα πρέπει να αναφερθεί με συγκεκριμένο URL Path
- Η μορφή του URL path είναι η παρακάτω:
http://host:port/path?query
- Για παράδειγμα:
`http://www.aueb.gr/login?user=than&pass=than123`



URL Paths (2)

- Για παράδειγμα, ο client μπορεί να καλέσει το παρακάτω URL:

`http://www.aueb.gr/login?user=than&pass=than123`

- Το `http://www.aueb.gr` είναι το URL του Web Server
- Το ***/login*** είναι το path ακολουθούμενο από το Query string που αποτελείται από παραμέτρους στη μορφή `key=value`, όπως το `user=than` και το `pass=than123` (το `&` σημαίνει ΚΑΙ). Το `?` είναι διαχωριστικό
- Τόσο το path όσο και το query string πρέπει να είναι URL encoded (βλ. percent encoding) ώστε να μην υπάρχει conflict με ειδικούς χαρακτήρες (`?`, `/`, `#`, κλπ.)



HTTP Request Header (1)

AJAX

- Κάθε HTTP Request περιέχει τον HTTP Header (που περιέχει HTTP metadata) και το body (που περιέχει δεδομένα - payload)
- Οι headers μέσα στο HTTP Header είναι μεταδεδομένα ανάλογα αν είναι HTTP Request ή HTTP Response
- Στο HTTP Request μπορεί να έχουμε το Request-Line, καθώς και τους παρακάτω headers: User-Agent, Accept, Cookie, Connection, Host, Referer, Authorization,



HTTP Request Header (2)

AJAX

```
GET /index.html HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:90.0) Gecko/20100101 Firefox/90.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

- Η 1^η γραμμή είναι το Request Line και περιλαμβάνει το HTTP method: **GET**, το URI: **/index.html** και HTTP version: **HTTP/1.1**
- Στη συνέχεια έχουμε τα headers: **Host**: www.example.com, **User-Agent**: από που έγινε το request (Firefox 90.0 running on Windows 10), **Accept**: ποιους MIME types (τύπους δεδομένων) ο client μπορεί να χειριστεί, **Accept-Language**: γλώσσα που καταλαβαίνει ο client, **Accept-Encoding**: το encoding που μπορεί να χειριστεί ο client, **Connection**: ορίζει αν το connection θα μείνει ανοικτό (στο HTTP 1.1)



Request-Line

AJAX

- Για παράδειγμα ένα Request-Line θα μπορούσε να είναι το παρακάτω:
- GET /hello.html HTTP/1.1
Host: www.aueb.gr
- Σημαίνει να επιστραφεί στον client η σελίδα <http://www.aueb.gr/hello.html>
- Η μέθοδος GET είναι HTTP method που σημαίνει την επιστροφή από τον server ενός πόρου, π.χ. ενός αρχείου, όπως εδώ το hello.html
- Το HTTP/1.1 είναι μία έκδοση (version) του πρωτοκόλλου HTTP



HTTP Methods

AJAX

- Οι βασικές μέθοδοι που υποστηρίζονται από το HTTP είναι οι παρακάτω:
 - **PUT** – Κάνει update με βάση τα δεδομένα που προσδιορίζονται στο σώμα του Request
 - **GET** – Ζητάει τον πόρο που προσδιορίζεται από το HTTP URI. Μπορεί να παρέχει και query string
 - **POST** – Κάνει εισαγωγή με βάση τα δεδομένα που αποστέλλονται μέσα στο HTTP Packet (και όχι στο URL μετά το ?)
 - **DELETE** – Διαγράφει τον πόρο



XML / JSON Μορφή

AJAX

- Τα δεδομένα του request και του response είναι XML ή JSON. Αν για παράδειγμα ζητήσουμε από ένα Web Service (service του back-end που παρέχεται over HTTP) τα στοιχεία ενός Καθηγητή με id = 1, τότε αυτά θα επιστρέφονταν ως εξής

XML

```
<teacher>
  <id>1</id>
  <name>Ath. Androutsos</name>
</teacher>
```

JSON

```
{
  "id": 1,
  "name": "Ath. Androutsos"
}
```



XML vs JSON

AJAX

- Σχετικά με τις δύο μορφές αναπαράστασης δεδομένων XML και JSON για τη μεταφορά τους μέσω δικτύου είναι προτιμότερη η μορφή JSON γιατί:
 - είναι πιο σύντομη
 - Το συντακτικό είναι παρόμοιο με της JavaScript
 - Μπορεί να επεξεργαστεί ευκολότερα με JavaScript



AJAX (1)

- Asynchronous JavaScript and XML (AJAX)
- Χρησιμοποιείται για τη μεταφορά δεδομένων από τον client στον server και το αντίστροφο με *ασύγχρονο τρόπο*, ταυτόχρονα με το κύριο πρόγραμμα (δηλαδή δεν διακόπτεται το πρόγραμμα στον client περιμένοντας να ολοκληρωθεί το request, αλλά το πρόγραμμα συνεχίζει κανονικά και μόλις ολοκληρωθεί η μεταφορά των δεδομένων τα λαμβάνει)



AJAX (2)

- Το πλεονέκτημα της τεχνολογίας AJAX είναι ότι στον client (front-end) δεν γίνεται reload όλη η HTML σελίδα, αλλά αφού λαμβάνουμε δεδομένα μπορούμε να διατηρήσουμε την HTML σελίδα και να αλλάξει μόνο το μέρος της σελίδας που αφορά το AJAX Request
- Σκεφτείτε για παράδειγμα κάποιες σελίδες κοινωνικών δικτύων. Όταν κάνουμε μία ενέργεια δεν αλλάζει όλη η σελίδα αλλά μέρη της σελίδας



AJAX Request / Response

AJAX

- Ένα AJAX request αιτείται την μεταφορά από τον server ενός πόρου δηλαδή δεδομένα over HTTP και το response από τον Server μπορεί να είναι XML ή JSON
- Χρησιμοποιεί το πρωτόκολλο HTTP (HyperText Transfer Protocol) για να επικοινωνεί με URL endpoints
- Με AJAX δεν ζητάμε HTML σελίδες αλλά δεδομένα
- Οι HTML σελίδες είναι human-readable το JSON string ή το XML είναι machine-readable



Apache Tomcat (1)

AJAX

The screenshot shows the Apache Tomcat 10 Software Downloads page. The browser address bar shows `tomcat.apache.org/download-10.cgi`. The page features the Apache Tomcat logo (a yellow cat) and the Apache Software Foundation logo. A search bar is present on the left. The main content area is titled "Tomcat 10 Software Downloads" and contains the following text:

Welcome to the Apache Tomcat® 10.x software download page. This page provides download links for obtaining the latest version of Tomcat 10.1.x software, as well as links to the archives of older releases.

Unsure which version you need? Specification versions implemented, minimum Java version required and lots more useful information may be found on the ['which version?'](#) page.

Users of Tomcat 10 onwards should be aware that, as a result of the move from Java EE to Jakarta EE as part of the transfer of Java EE to the Eclipse Foundation, the primary package for all implemented APIs has changed from `javax.*` to `jakarta.*`. This will almost certainly require code changes to enable applications to migrate from Tomcat 9 and earlier to Tomcat 10 and later. A [migration tool](#) has been developed to aid this process.

Quick Navigation

[KEYS](#) | [10.1.18](#) | [Browse](#) | [Archives](#)

Release Integrity

You **must** [verify](#) the integrity of the downloaded files. We provide OpenPGP signatures for every release file. This signature should be matched against the [KEYS](#) file which contains the OpenPGP keys of Tomcat's Release Managers. We also provide [SHA-512](#) checksums for every release file. After you download the file, you should calculate a checksum for your download, and make sure it is the same as ours.

On the left sidebar, under "Download", the "Tomcat 10" link is circled in red.

- Θα εγκαταστήσουμε ένα Web Server στον υπολογιστή μας, τον Apache Tomcat – Έκδοση 10 <http://tomcat.apache.org/>



Apache Tomcat (2)

AJAX

The screenshot shows the Apache Tomcat download page for version 10.1.18. The browser address bar shows the URL tomcat.apache.org/download-10.cgi. The page has a left sidebar with links for Documentation, Problems?, and other resources. The main content area shows the version 10.1.18 and a list of Binary Distributions. The link for the 32-bit/64-bit Windows Service Installer is circled in red.

Documentation

- Tomcat 11.0 (alpha)
- Tomcat 10.1
- Tomcat 9.0
- Tomcat 8.5
- Upgrading
- Tomcat Connectors
- Tomcat Native 2
- Tomcat Native 1.2
- Wiki
- Migration Guide
- Presentations
- Specifications

Problems?

- Security Reports
- Find help
- FAQ
- Mailing Lists
- Bug Database
- IRC

10.1.18

Please see the [README](#) file for packaging information. It explains what you need to do to get the binaries.

Binary Distributions

- Core:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [32-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [64-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [32-bit/64-bit Windows Service Installer](#) ([pgp](#), [sha512](#))
- Full documentation:
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Deployer:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Embedded:
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [zip](#) ([pgp](#), [sha512](#))

- <https://tomcat.apache.org/download-90.cgi>
- Για Windows κατεβάζετε τον installer και εγκαθιστάτε
- Για άλλα λειτουργικά ακολουθείτε τις οδηγίες



Apache Tomcat (3)

AJAX

- Εγκαταστήστε τον Apache ως Windows Service
- Καλύτερα να επιλέξετε να μην ξεκινάει αυτόματα το Windows Service όταν ξεκινάει ο υπολογιστής σας (όχι auto startup) γιατί καταναλώνει πόρους από τον υπολογιστή σας, εκτός αν θέλετε να ξεκινάει αυτόματα



Apache Tomcat (4)

AJAX

- Στην Εγκατάσταση πατήστε next

Apache Tomcat Setup: Configuration Options

Configuration
Tomcat basic configuration.

Server Shutdown Port: 81

HTTP/1.1 Connector Port: 8080

Windows Service Name: Tomcat10

Create shortcuts for all users: ☐

Tomcat Administrator Login (optional)

User Name:

Password:

Roles: manager-gui

Nullsoft Install System v3.09

< Back **Next >** Cancel



Apache Tomcat (5)

AJAX

Προγράμματα και δυνατότητες

Κέντρο φορητότητας

Επιλογές λειτουργίας

Προβολή συμβάντων

Σύστημα

Διαχείριση Συσκευών

Συνδέσεις δικτύου

Διαχείριση δίσκων

Διαχείριση υπολογιστή

Γραμμή εντολών

Γραμμή εντολών (Διαχειριστής)

Διαχείριση Εργασιών

Πίνακας Ελέγχου

Εξερεύνηση αρχείων

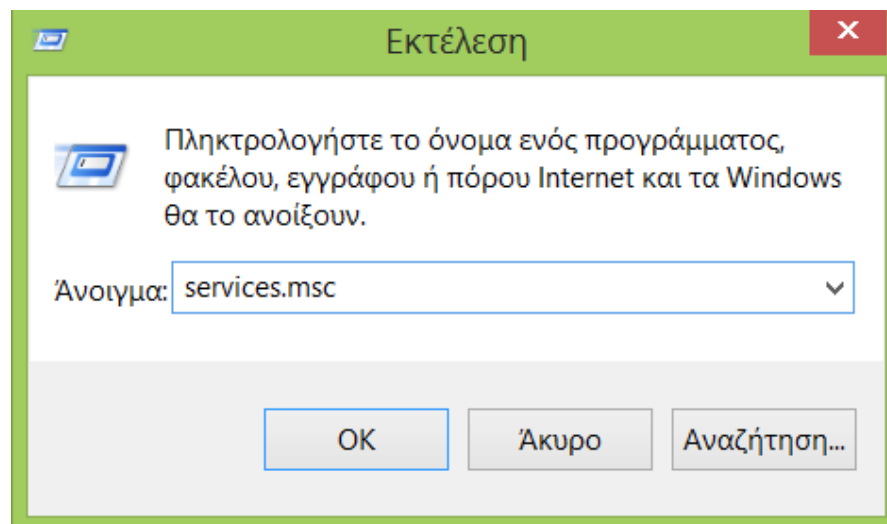
Αναζήτηση

Εκτέλεση

Τερματισμός λειτουργίας ή έξοδος

Επιφάνεια εργασίας

- Στη συνέχεια όποτε θέλετε να ξεκινάτε ή να σταματάτε τον Web Server μπορείτε με δεξί κλικ στο start menu και Εκτέλεση (Run)
- Στο παράθυρο που ανοίγει εισάγετε *services.msc* και enter





Apache Tomcat (6)

AJAX

Υπηρεσίες (τοπικές)					
Apache Tomcat 10.1 Tomcat10					
Διακοπή της υπηρεσίας Επανεκκίνηση της υπηρεσίας					
Όνομα	Περιγραφή	Κατάσταση	Τύπος εκκίνησης	Σύνδεση ως	
SQL Server (SQLEXPRESS)	Provides sto...	Σε λειτουργία	Αυτόματα	NT Service\M...	
SQL Server CEIP service (SQLEXPRESS)	CEIP service ...	Σε λειτουργία	Αυτόματα	NT Service\S...	
Apache Tomcat 10.1 Tomcat10	Apache Tom...	Σε λειτουργία	Μη αυτόματα	Τοπική υπηρ...	

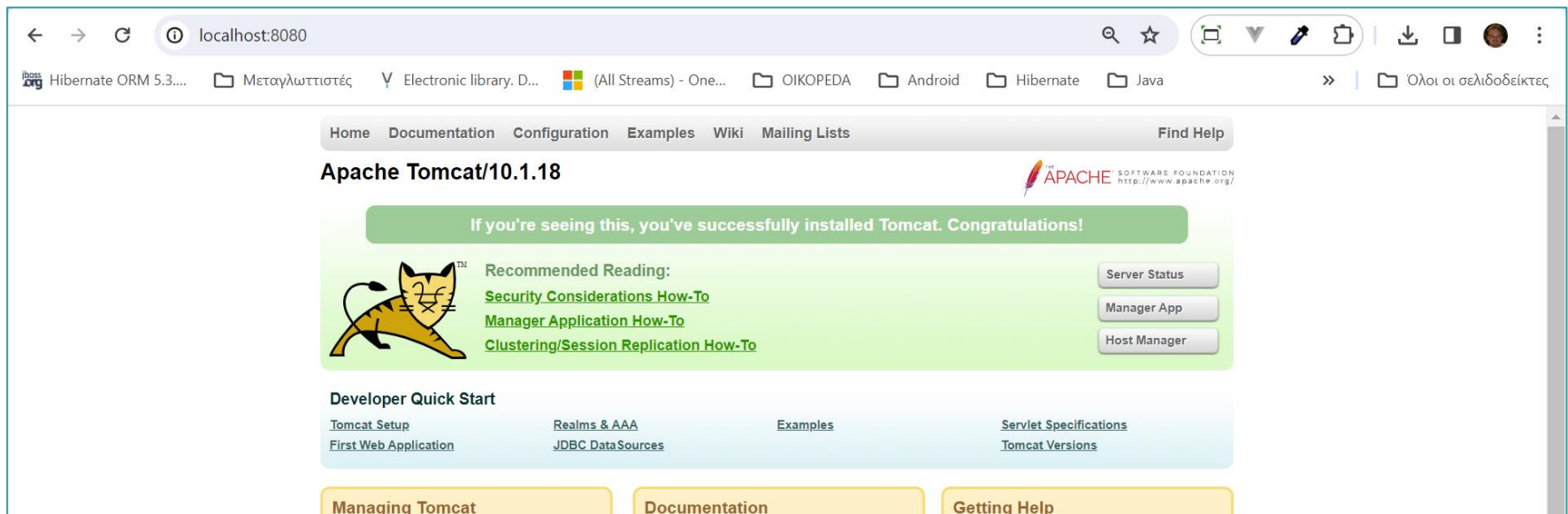
- Από τη λίστα των services, επιλέγετε την υπηρεσία Apache Tomcat 10.1 και κάνετε δεξί κλικ και Εκκίνηση και αν θέλετε να την σταματήσετε, δεξί κλικ πάνω στην επιλογή και Διακοπή



Apache Tomcat (7)

AJAX

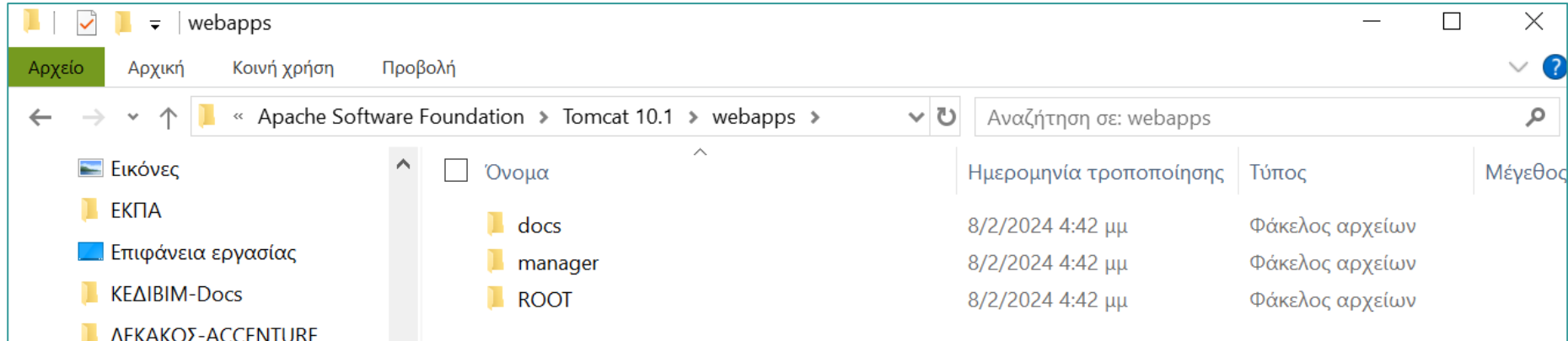
- Για να τεστάρουμε ότι λειτουργεί σωστά ο Apache Server μας, ανοίγουμε τον Chrome και δίνουμε
- <http://localhost:8080/>
- Πρέπει να εμφανιστεί η παρακάτω σελίδα:





Apache Tomcat projects folder

AJAX



- Ο Apache Tomcat συνήθως εγκαθίσταται στο C:\Program Files\Apache Software Foundation\Tomcat 10.*
- Μέσα στον παραπάνω φάκελο υπάρχει υποφάκελος **webapps**, μέσα στον οποίο μπορούμε να **δημιουργούμε (υπό)φакέλους με τα Web project μας όταν χρησιμοποιούμε AJAX**



Client - Server

AJAX

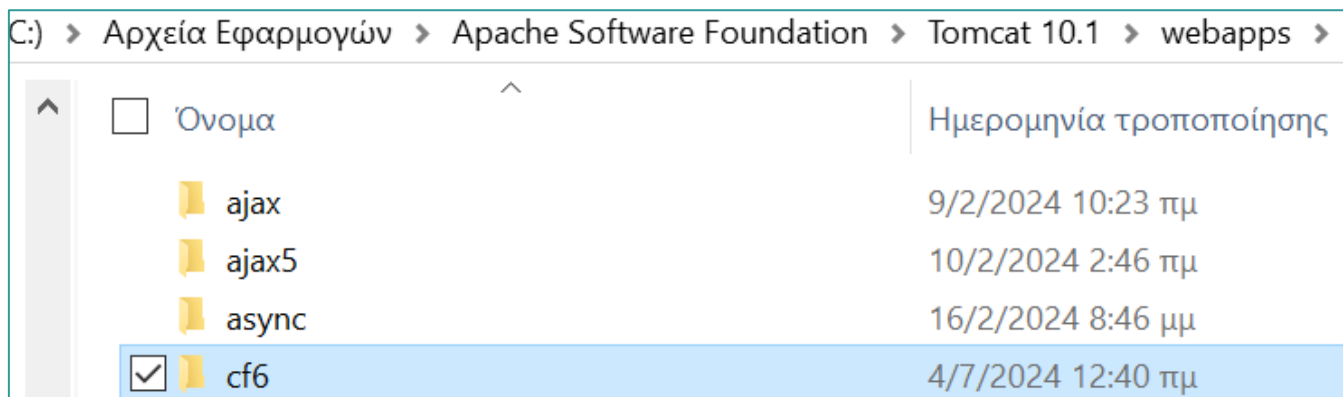
- Ο λόγος που εγκαταστήσαμε τον Web Server είναι γιατί όπως είπαμε για τις σελίδες που χρησιμοποιούν AJAX και αιτούνται πόρων, τόσο οι html σελίδες όσο και οι πόροι (XML, JSON) πρέπει να βρίσκονται σε ένα Web Server
- Δεν έχει σημασία που ο Web Server μας είναι εγκατεστημένος τοπικά, αυτό δεν γίνεται αντιληπτό από το σύστημα, το που δηλαδή βρίσκεται ο Web Server



Παράδειγμα

AJAX

- Μέσα στον **Webapps**, δημιουργούμε φάκελο με οποιοδήποτε όνομα της επιλογής σας (έστω cf6)
- Ανοίγουμε το Visual Studio Code και επιλέγουμε αυτόν τον φάκελο. Επομένως όλα τα αρχεία του project μας (.html, .js κλπ) θα βρίσκονται μέσα στον φάκελο
- Για να δουλέψει η τεχνική AJAX θα πρέπει η σελίδα να φορτώνει από Web Server και το αρχείο να βρίσκεται στον Web Server αλλιώς δημιουργείται **CORS error** (Cross-origin resource sharing error)
- Οπότε αν ο φάκελος του project μας είναι cf6 και το αρχείο μας helloajax.html θα πρέπει να δώσουμε στον browser: `http://localhost:8080/cf6/helloajax.html`





HTML

AJAX

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Ajax</title>
8  </head>
9  <body>
10     <div class="outer">
11         <div class="center">
12             <div class="cf-text">
13
14             </div>
15             <button type="button" class="btn">Show Text File</button>
16         </div>
17     </div>
18
19     <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.3/jquery.min.js"
20         integrity="sha512-STof4xm1wgkfm7heWqFJVn58Hm3EtS31XFaagaa8VMReCXAkQnJZ+jEy8PCC/">
21 </body>
22 </html>
```

Θέλουμε όταν πατάμε το button να επικοινωνεί ο client (η html σελίδα μας) με τον server (web server μας) και με τη χρήση XMLHttpRequest να εμφανίζει το περιεχόμενο ενός αρχείου txt, το οποίο βρίσκεται στον server



CSS

- Προσθέτουμε λίγο CSS για styling

```
1  .outer {  
2      width: 800px;  
3      margin-left: auto;  
4      margin-right: auto;  
5      display: flex;  
6      flex-direction: column;  
7      justify-content: center;  
8      align-items: center;  
9  }  
10  
11  .center {  
12      text-align: center;  
13  }  
14  
15  .cf-text {  
16      padding-top: 10px;  
17      width: 400px;  
18      min-height: 300px;  
19      border: 1px solid black;  
20  }  
21  
22  .btn {  
23      margin-top: 30px;  
24      width: 150px;
```



JavaScript – AJAX (1)

AJAX

```
1  ✓ $(function() {  
2  ✓      $(' .btn').on('click', function() {  
3      |         fetchData()  
4      |     })  
5  ✓ })
```

- Ο Handler του event στο on click event καλεί την fetchData η οποία υλοποιεί το XMLHttpRequest που είναι το βασικό αντικείμενο που υλοποιεί τη σύνδεση με τον WebServer
- Το σενάριο είναι δηλαδή ότι 1) κατεβαίνει η σελίδα HTML μαζί με το CSS και το JS αρχείο από τον WebServer στον client και 2) ο client επικοινωνεί με τον Web Server και του ζητάει ένα resource που στην περίπτωσή μας είναι ένα απλό αρχείο txt



XMLHttpRequest (1)

AJAX

```
8  function fetchData() {
9      let ajaxRequest = new XMLHttpRequest()
10
11      ajaxRequest.open("GET", "./hello.txt", true) // default is true, for asynchronous
12
13      ajaxRequest.onreadystatechange = function() {
14
15          if (ajaxRequest.readyState === 4) // response received successfully
16          {
17              if (ajaxRequest.status === 200) { // HTML packet status code OK
18                  handleResults(ajaxRequest.responseText)
19              }
20              else {
21                  showError()
22              }
23          }
24      }
25
26      ajaxRequest.send()
27  }
```

- Το βασικό αντικείμενο που χρησιμοποιεί η τεχνολογία AJAX είναι το XMLHttpRequest. Η μέθοδος **open** αρχικοποιεί το request
- Ορίζει τον τύπο του HTTP request (GET) και το URL ή ένα αρχείο, όπως εδώ το hello.txt
- Το true στη συνέχεια είναι προαιρετική παράμετρος (το default είναι true) και ορίζει ότι η διαδικασία θα εκτελεστεί ασύγχρονα



XMLHttpRequest (4)

AJAX

- Το *onreadystatechange* είναι ένα event property που ενεργοποιείται όταν αλλάζει η κατάσταση του request. Το **.readyState === 4** ισχύει όταν ολοκληρωθεί επιτυχώς το request και το response είναι ready
- Το **.status === 200** αφορά ότι το resource βρέθηκε και είναι OK (success). Άλλοι κωδικοί όπως π.χ. 404 σημαίνουν failure
- Αν είναι OK εμφανίζουμε το response με την `handleResults()` αλλιώς εμφανίζουμε error
- Το `ajaxRequest.responseText` είναι το περιεχόμενο του Response



XMLHttpRequest (2)

AJAX

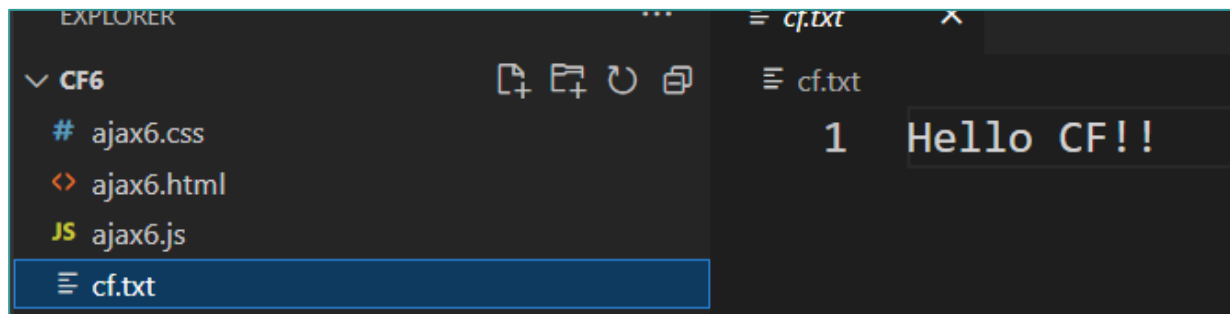
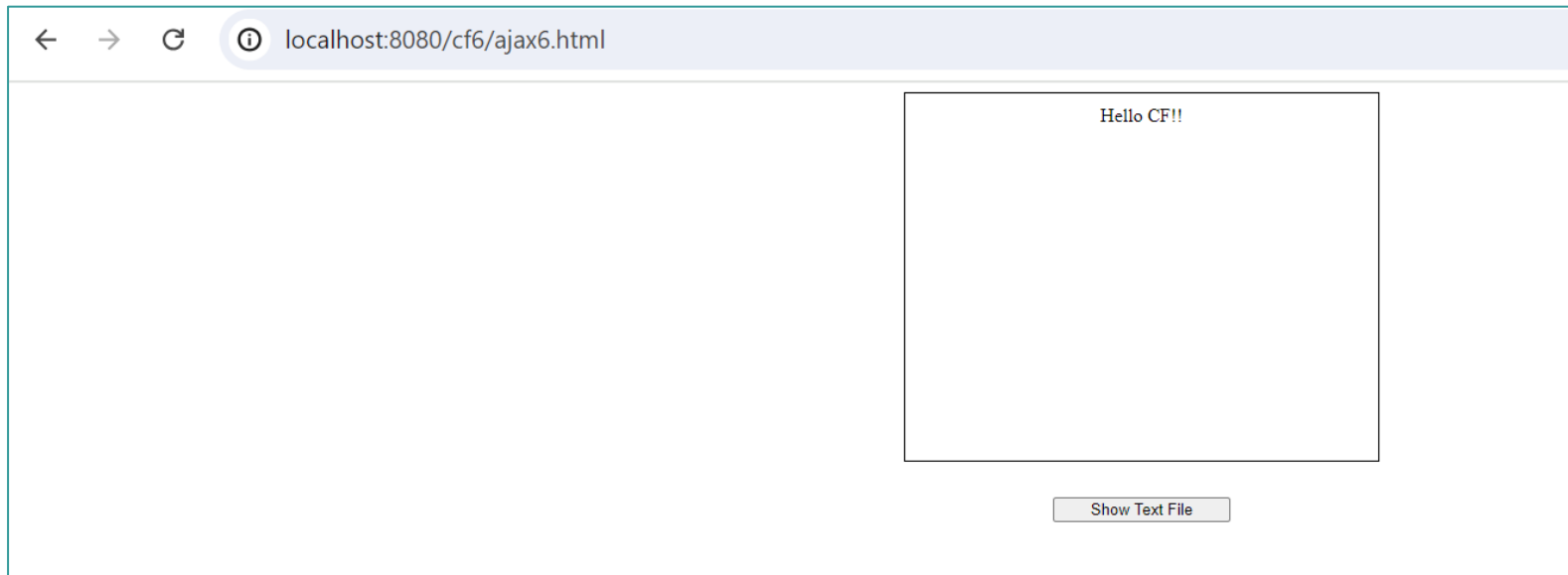
```
29  function handleResults(results) {  
30      |    $('cf-text').text(results)  
31  }  
32  
33  function showError() {  
34      |    console.log('API Error')  
35  }
```

- Handler των received data. Αντιστοιχεί σε UI Elements
- Error Handler



Αποτέλεσμα

AJAX



- Αν έχουμε το αρχείο αριστερά, το αποτέλεσμα είναι ορθό



Ready state / status

AJAX

0: request not initialized
1: server connection established
2: request received
3: processing request
4: request finished and response is ready

200: "OK"
403: "Forbidden"
404: "Not Found"

- Η ιδιότητα `readyState` μπορεί να πάρει όλες τις πάνω αριστερά τιμές. Μόνο όταν ολοκληρωθεί το `request-response` η τιμή είναι 4
- Το `response` θα πρέπει να ελεγχθεί αν είναι έγκυρο, αν δηλαδή έχει βρεθεί το `resource` που αιτηθήκαμε. Τότε η ιδιότητα `status` είναι 200 σύμφωνα με το πρωτόκολλο HTTP βλ. <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>



Επεξεργασία XML

AJAX

```
books.xml
1  <library>
2    <book>
3      <title>Java Fundamentals</title>
4      <author>A. Androutsos</author>
5    </book>
6    <book>
7      <title>The Art of Programming</title>
8      <author>D. Knuth</author>
9    </book>
10   <book>
11     <title>Computer Networks</title>
12     <author>A. Tanenbaum</author>
13   </book>
14 </library>
```

- Έστω το αρχείο XML αριστερά
- Θα προσπαθήσουμε να το διαβάσουμε και να το εμφανίσουμε σε μια web σελίδα σε μορφή πίνακα

Όπως είπαμε δύο είναι οι βασικές μορφές διακίνησης της πληροφορίας στο Web: XML και JSON. Θα ξεκινήσουμε να δούμε την μορφή XML



HTML

AJAX

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="
6   <meta name="viewport" content="width=device
7   <link rel="stylesheet" href="./ajaxxmljson.
8   <script src="./js/jquery.min.js"></script>
9   <script src="./ajaxxmljson.js"></script>
10
11   <title>List of Books</title>
12 </head>
13 <body>
14   <div class="outer">
15     <h1>List of Books</h1>
16     <table class="books-list"></table>
17   </div>
18
19   <div class="error hidden">
20     <span>No books found</span>
21   </div>
22
23 </body>
24 </html>
```

- Δημιουργούμε μία σελίδα HTML που περιέχει ένα `<h1>` και ένα `<table>`
- Συνδέουμε με το `ajaxxmljson.js` JavaScript αρχείο που θα φτιάξουμε
- Μέσα στο `table` θα εμφανίσουμε τα περιεχόμενα του XML



CSS

- Styling του εγγράφου

```
1  .outer {
2      margin: 0 auto;
3      width: 50%;
4      display: flex;
5      flex-direction: column;
6
7      align-items: center;
8  }
9
10 table {
11     border: 1px solid black;
12     border-collapse: collapse;
13 }
14
15 th, td {
16     text-align: center;
17     border: 1px solid black;
18     padding: 5px;
19 }
20
21 .hidden {
22     display: none;
23 }
24
```



XML – XMLHttpRequest

AJAX

- Αρχικά λαμβάνουμε το responseXML, που είναι το XML αρχείο και το κάνουμε handle

```
1 $(document).ready(function() {
2     fetchBooks()
3 })
4
5 function fetchBooks() {
6     beforeSend()
7     let xhr = new XMLHttpRequest()
8     xhr.open('GET', './books.xml', true)
9     xhr.onreadystatechange = function() {
10         if (xhr.readyState === 4) {
11             if (xhr.status === 200) {
12                 handleResults(xhr.responseXML)
13             }
14             else {
15                 onAPIError()
16             }
17         }
18     }
19     xhr.send()
20 }
```

```
1 <library>
2   <book>
3     <title>Java Fundamentals</title>
4     <author>A. Androutsos</author>
5   </book>
6   <book>
7     <title>The art of programming</title>
8     <author>D. Knuth</author>
9   </book>
10  <book>
11    <title>Computer Networks</title>
12    <author>A. Tanenbaum</author>
13  </book>
14 </library>
```




XML - handleResults

AJAX

```
22 function onBeforeSend() {  
23     hideError()  
24 }  
25  
26 function handleResults(response) {  
27     if (!response) {  
28         showError()  
29         return  
30     }  
31  
32     let books = $(response).find('book')  
33     handleBooks(books)  
34 }
```

- Η `handleResults` ελέγχει και καλεί την `handleBooks` για να τα εμφανίσει



XML - buildBooks

AJAX

```
36 function handleBooks(books) {
37     let output = `<tr>
38         <th>Title</th><th>Author</th>
39     </tr>`
40
41     for (const book of books) {
42         let title = $(book).find('title').text()
43         let author = $(book).find('author').text()
44         output += `<tr>
45             <td>${title}</td><td>${author}</td>
46         </tr>`
47     }
48
49     $('.books').html(output)
50 }
```

- Δημιουργούμε την επικεφαλίδα του πίνακα με το string **output** στο οποίο στη συνέχεια προσθέτουμε νέες γραμμές πίνακα μέσα στην for που διατρέχει τα books
- Μέσα στη for για κάθε book παίρνουμε το title και τον author με find



XML – show errors

AJAX

```
52 function onAPIError() {  
53     console.log('Error on API')  
54 }  
55  
56 function showError() {  
57     console.log($(' .error.hidden').text())  
58     $(' .error.hidden').clone().removeClass('hidden').appendTo($(' .outer'))  
59 }  
60  
61 function hideError() {  
62     $(' .outer').find(' .error').remove()  
63 }
```

- Error Handlers



Αποτέλεσμα

AJAX

← → ↻ ⓘ localhost:8080/ajax/ajax-xml/ 🔍 ☆ 🖨️ ▼ ✎ 📁 | 📱 👤 ⋮

List of Books

Title	Author
Java Fundamentals	A. Androutsos
The art of programming	D. Knuth
Computer Networks	A. Tanenbaum



JSON – fetch books (1)

AJAX

```
1  $(document).ready(function() {
2      fetchBooks()
3  })
4
5  function fetchBooks() {
6      beforeSend()
7      let xhr = new XMLHttpRequest()
8      xhr.open('GET', './books.json', true)
9      xhr.onreadystatechange = function() {
10         if (xhr.readyState === 4) {
11             if (xhr.status === 200) {
12                 handleResults(JSON.parse(xhr.responseText))
13             }
14             else {
15                 onAPIError()
16             }
17         }
18     }
19     xhr.send()
20 }
21
```

- Το JSON είναι text οπότε επιστρέφεται με την ιδιότητα *responseText*
- Η *JSON.parse* μετατρέπει από JSON σε JavaScript Object ώστε μετά να επεξεργαστούμε με την *handleResults()*



JSON String

AJAX

books.json

```
{
  "books": [
    {
      "title": "Java Fundamentals",
      "author": "Athan. Androutsos"
    },
    {
      "title": "The Art of programming",
      "author": "D. knuth"
    },
    {
      "title": "Computer Networks",
      "author": "A. Tanenbaum"
    }
  ]
}
```

- Μέσα στο αρχείο θεωρείται String
- Δεν χρειάζεται ' ή " στην αρχή και το τέλος όπως στην JavaScript



JSON – fetch books (2)

AJAX

```
22 function onBeforeSend() {  
23     hideError()  
24 }  
25  
26 function handleResultsJSON(response) {  
27     if (!response) {  
28         showError()  
29     }  
30  
31  
32     let books = response.books  
33     buildBooksJSON(books)  
34 }
```

- Η `handleResults` λαμβάνει JS object
- Λαμβάνει τα `books` που είναι πίνακας με `response.books`
- Και κάνει build το output



JSON - handleResults

AJAX

```
36 function handleBooks(books) {  
37     let output = `<tr>  
38         <th>Title</th><th>Author</th>  
39     </tr>`  
40  
41     for (const book of books) {  
42         let title = book.title  
43         let author = book.author  
44         output += `<tr>  
45             <td>${title}</td><td>${author}</td>  
46         </tr>`  
47     }  
48  
49     $(' .books').html(output)  
50 }
```

- Η πρόσβαση στα πεδία γίνεται με τον τελεστή τελεία αφού πρόκειται για JavaScript objects



JSON

```
{ } books.json > ...
```

```
1  {
2      "books": [
3          {
4              "title": "Java",
5              "author": "Androutsos"
6          },
7          {
8              "title": "Databases",
9              "author": "Kapetis"
10         },
11         {
12             "title": "Networking",
13             "author": "Fragoudakis"
14         }
15     ]
16 }
```

- Η μορφή JSON είναι πιο εύχρηστη από την XML γιατί είναι πιο κοντά στα objects της JavaScript
- Ένα JSON object περιέχει ιδιότητες δηλ. ένα ζεύγος *μεταβλητή: τιμή*, όπου οι ιδιότητες είναι μέσα σε "" και οι τιμές μπορούν να είναι αριθμοί, αλφαριθμητικά (μέσα σε ""), πίνακες ή άλλα JSON objects
- Αριστερά έχουμε την JSON μορφή του αντικειμένου του προηγούμενου παραδείγματος. Υπάρχουν free online converters από XML σε JSON που κάνουν αυτόματα τη μετατροπή



JSON – show errors

AJAX

```
52 function onAPIError() {  
53     console.log('Error on API')  
54 }  
55  
56 function showError() {  
57     console.log($('.error.hidden').text())  
58     $('.error.hidden').clone().removeClass('hidden').appendTo($('.outer'))  
59 }  
60  
61 function hideError() {  
62     $('.outer').find('.error').remove()  
63 }
```

- Εμφανίζουμε failures (errors) και αποκρύπτουμε για το επόμενο request



Αποτελέσματα

AJAX

localhost:8080/testajax/ajaxxmljson.html

List of Books

Title	Author
Java Fundamentals	Athan. Androutsos
The art of programming	D. Knuth
Computer Networks	A. Tanenbaum

- Τα αποτελέσματα. Εμφανίζουμε ένα HTML Element (τον πίνακα) με data



OpenAPIs (1)

AJAX

- Στη συνέχεια θα δούμε κάποια OpenAPIs δηλαδή κάποιους Servers που παρέχουν Web Services (δηλαδή JSON data), τα οποία είναι διαθέσιμα μέσω της διαδικασίας που αναφερθήκαμε
- Καλούμε την υπηρεσία μέσω ενός URL δίνοντας και κάποιες παραμέτρους αν χρειάζεται



OpenAPIs (2)

AJAX

- Λέγονται OpenAPIs (Open Application Programming Interfaces) γιατί είναι αφενός μεν **διαθέσιμα προς όλους (Open)** και αφετέρου παρέχουν υπηρεσίες (APIs, που μπορούμε να καλέσουμε μέσω URL)
- Αυτές οι υπηρεσίες μπορούν να κληθούν προγραμματιστικά (με το XMLHttpRequest object της JavaScript)



OMDb API

AJAX

- OpenAPI για ταινίες – Χρειαζόμαστε API Key

← → ↻ ⓘ Μη ασφαλής | omdbapi.com


OMDb API Usage Parameters Examples Change Log **API Key** Become a Patron Contact

OMDb API

The Open Movie Database

The OMDb API is a RESTful web service to obtain movie information, all content and images on the site are contributed and maintained by our users.

If you find this service useful, please consider making a [one-time donation](#) or [become a patron](#).



Poster API

The Poster API is only available to patrons.

Currently over 280,000 posters, updated daily with resolutions up to 2000x3000.

- Μας δίνει πληροφορίες για ταινίες
- Για να καλέσουμε τις διαθέσιμες υπηρεσίες χρειάζεται να λάβουμε ένα API Key όπως λέγεται δηλαδή ένα κωδικό που θα αποστέλλουμε σε κάθε request



API Key (1)

AJAX

OMDb API

[Become a Patron](#) [Donate](#) [Contact](#)

API Key

10/05/22 Email Delays! If your requested key doesn't show up within an hour, please contact me directly. ✕

Generate API Key

Account Type ☒ Patreon ☐ FREE! (1,000 daily limit)

Email

Enter the email address that you used with Patreon.

- Επιλέγουμε FREE, δίνουμε τα στοιχεία μας και λαμβάνουμε στο mail ένα API Key που μας επιτρέπει να κάνουμε αναζητήσεις

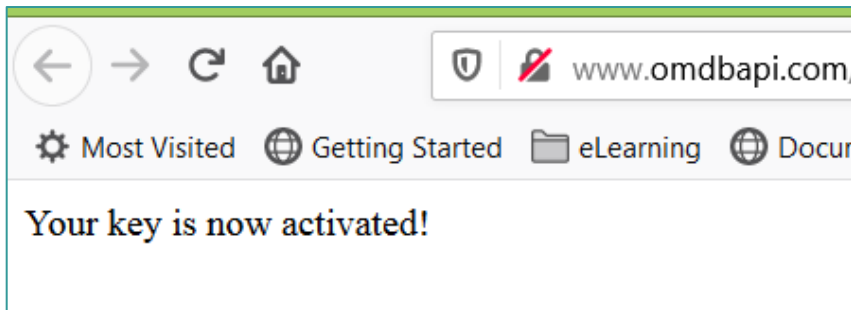


API Key (2)

AJAX

A verification link to activate your key was sent to: a8anassis@gmail.com

Click the following URL to activate your key: [http://www.omdbapi.com/apikey.aspx?VERIFYKEY=\[redacted\]](http://www.omdbapi.com/apikey.aspx?VERIFYKEY=[redacted])
If you did not make this request, please disregard this email.



- Αφού επιβεβαιώσουμε το e-mail το API Key ενεργοποιείται



Πως καλούμε τις υπηρεσίες

AJAX

OMDb API - The Open Movie Database

Μη ασφαλής | omdbapi.com

OMDb API Usage **Parameters** Examples Change Log API Key Become a Patron Contact

By ID or Title

Parameter	Required	Valid Options	Default Value	Description
i	Optional*		<empty>	A valid IMDb ID (e.g. tt1285016)
t	Optional*		<empty>	Movie title to search for.
type	No	movie, series, episode	<empty>	Type of result to return.
y	No		<empty>	Year of release.
plot	No	short, full	short	Return short or full plot.
r	No	json, xml	json	The data type to return.
callback	No		<empty>	JSONP callback name.
v	No		1	API version (reserved for future use).

*Please note while both "i" and "t" are optional at least one argument is required.

- Μπορούμε να πάμε στο Μενού Parameters για να δούμε πως καλούμε
- Παρατηρούμε πως με την παράμετρο **t** μπορούμε να αναζητήσουμε με βάση τον τίτλο της ταινίας



Example

AJAX

The screenshot shows the OMDb API website. The browser address bar displays 'omdbapi.com'. The navigation bar includes links for 'OMDb API', 'Usage', 'Parameters', 'Examples', 'Change Log', and 'API Key', along with a 'Become a Patron' button. The main heading is 'Examples'. Under the 'By Title' section, there is a search form with the following fields: 'Title' (containing 'Casablanca'), 'Year' (empty), 'Plot' (a dropdown menu set to 'Short'), and 'Response' (a dropdown menu set to 'JSON'). There are 'Search' and 'Reset' buttons. Below the form, the 'Request:' section shows the generated URL: <http://www.omdbapi.com/?t=Casablanca>.

- Κάνουμε αναζήτηση για Casablanca



Usage

AJAX

Usage

Send all data requests to:

```
http://www.omdbapi.com/?apikey=[yourkey]&
```

Poster API requests:

```
http://img.omdbapi.com/?apikey=[yourkey]&
```



Παράδειγμα (1)

AJAX

```
{
  "Title": "Casablanca",
  "Year": "1942",
  "Rated": "PG",
  "Released": "23 Jan 1943",
  "Runtime": "102 min",
  "Genre": "Drama, Romance, War",
  "Director": "Michael Curtiz",
  "Writer": "Julius J. Epstein (screenplay), Philip G. Epstein (screenplay), Howard Koch (screenplay), Murray Burnett (play), Joan Alison (play)",
  "Actors": "Humphrey Bogart, Ingrid Bergman, Paul Henreid, Claude Rains",
  "Plot": "A cynical American expatriate struggles to decide whether or not he should help his former lover and her fugitive husband escape French Morocco.",
  "Language": "English, French, German, Italian",
  "Country": "USA",
  "Awards": "Won 3 Oscars. Another 5 wins & 9 nominations.",
  "Poster": "https://m.media-amazon.com/images/M/MV5BY2IzZGY2YmEtYzljNS00NTM5LTgwMzUtMzM1NjQ4NGI0OTk0XkEyXkFqcGdeQXVyNDYyMDk5MTU@._V1_SX300.jpg",
  "Ratings": [
    {
      "Source": "Internet Movie Database",
      "Value": "8.5/10"
    },
    {
      "Source": "Rotten Tomatoes",
      "Value": "99%"
    },
    {
      "Source": "Metacritic",
      "Value": "100/100"
    }
  ],
  "Metascore": "100",
  "imdbRating": "8.5",
  "imdbVotes": "513,662",
  "imdbID": "tt0034583",
  "Type": "movie",
  "DVD": "N/A",
  "BoxOffice": "N/A",
  "Production": "Warner Brothers",
  "Website": "N/A",
  "Response": "True"
}
```

- Μπορούμε να εισάγουμε στο URL την παράμετρο **t** και τον τίτλο της ταινίας (Casablanca) ενώ ακολουθεί το **apikey** που έχουμε λάβει (το & συμβολίζει το λογικό ΚΑΙ)



Παράδειγμα (2)

AJAX

```
{
  "Title": "Casablanca",
  "Year": "1942",
  "Rated": "PG",
  "Released": "23 Jan 1943",
  "Runtime": "102 min",
  "Genre": "Drama, Romance, War",
  "Director": "Michael Curtiz",
  "Writer": "Julius J. Epstein (screenplay), Philip G. Epstein (screenplay), Howard Koch (screenplay), Murray Burnett (play), Joan Alison (play)",
  "Actors": "Humphrey Bogart, Ingrid Bergman, Paul Henreid, Claude Rains",
  "Plot": "A cynical American expatriate struggles to decide whether or not he should help his former lover and her fugitive husband escape French Morocco.",
  "Language": "English, French, German, Italian",
  "Country": "USA",
  "Awards": "Won 3 Oscars. Another 5 wins & 9 nominations.",
  "Poster": "https://m.media-amazon.com/images/M/MV5BY2IzZGY2YmEtYzljNS00NTM5LTgwMzUtMzM1NjQ4NGI0OTk0XkEyXkFqcGdeQXVyNDYyMDk5MTU@._V1_SX300.jpg",
  "Ratings": [
    {
      "Source": "Internet Movie Database",
      "Value": "8.5/10"
    },
    {
      "Source": "Rotten Tomatoes",
      "Value": "99%"
    },
    {
      "Source": "Metacritic",
      "Value": "100/100"
    }
  ],
  "Metascore": "100",
  "imdbRating": "8.5",
  "imdbVotes": "513,662",
  "imdbID": "tt0034583",
  "Type": "movie",
  "DVD": "N/A",
  "BoxOffice": "N/A",
  "Production": "Warner Brothers",
  "Website": "N/A",
  "Response": "True"
}
```

- Η απάντηση (το response) είναι σε μορφή JSON και μπορούμε στη συνέχεια να το επεξεργαστούμε



Παράδειγμα OMDb(1)

AJAX

```
{
  "Title": "Casablanca",
  "Year": "1942",
  "Rated": "PG",
  "Released": "23 Jan 1943",
  "Runtime": "102 min",
  "Genre": "Drama, Romance, War",
  "Director": "Michael Curtiz",
  "Writer": "Julius J. Epstein (screenplay), Philip G. Epstein (screenplay), Howard Koch (screenplay), Murray Burnett (play), Joan Alison (play)",
  "Actors": "Humphrey Bogart, Ingrid Bergman, Paul Henreid, Claude Rains",
  "Plot": "A cynical American expatriate struggles to decide whether or not he should help his former lover and her fugitive husband escape French Morocco.",
  "Language": "English, French, German, Italian",
  "Country": "USA",
  "Awards": "Won 3 Oscars. Another 5 wins & 9 nominations.",
  "Poster": "https://m.media-amazon.com/images/M/MV5BY2IzZGY2YmEtYzljNS00NTM5LTgwMzUtMzM1NjQ4NGI0OTk0XkEyXkFqcGdeQXVyNDYyMDk5MTU@._V1_SX300.jpg",
  "Ratings": [
    { "Source": "Internet Movie Database", "Value": "8.5/10" },
    { "Source": "Rotten Tomatoes", "Value": "99%" },
    { "Source": "Metacritic", "Value": "100/100" }
  ],
  "Metascore": "100",
  "imdbRating": "8.5",
  "imdbVotes": "513,662",
  "imdbID": "tt0034583",
  "Type": "movie",
  "DVD": "N/A",
  "BoxOffice": "N/A",
  "Production": "Warner Brothers",
  "Website": "N/A",
  "Response": "True"
}
```

Παρατηρούμε τη μορφή του JSON αρχείου για την ταινία Casablanca. Υπάρχει ένα πεδίο **Poster** που αναπαριστά την Poster (διαφημιστική εικόνα της ταινίας) καθώς και ένα πεδίο στο τέλος του JSON, το **Response** που είναι true αν υπάρχει η ταινία που αναζητούμε, αλλιώς είναι False



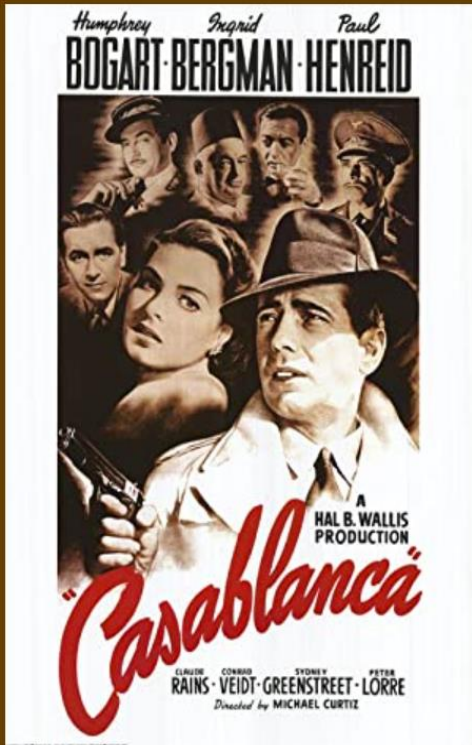
Επιθυμητό Αποτέλεσμα

AJAX

localhost:8080/cf3/testbed/movies.html#

Αναζήτηση Ταινιών

Casablanca



Casablanca

1942 102 min Drama, Romance, War

IMDb 8.5 / 10

A cynical expatriate American cafe owner struggles to decide whether or not to help his former lover and her fugitive husband escape the Nazis in French Morocco.

Σκηνοθεσία: Michael Curtiz

Ηθοποιοί: Humphrey Bogart, Ingrid Bergman, Paul Henreid

Παραγωγή: -

Box Office: \$4,219,709

Γλώσσα: English, French, German, Italian

Καταλληλότητα: PG

Περισσότερα



- Θα αναπτύξουμε ένα project που να υλοποιεί αναζήτηση ταινιών από το omdbapi.com και να εμφανίζει τα αποτελέσματα



HTML (1)

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Αναζήτηση Ταινιών</title>
6      <link rel="stylesheet" href="./movies.css">
7
8  </head>
9
10 <body>
11     <div class="center">
12         <div class="bot-gap">
13             <span class="title">Αναζήτηση Ταινιών</span>
14         </div>
15         <div class="bot-gap">
16             <form>
17                 <input id="searchInput" type="text" class="search rounded"
18                     placeholder="Πληκτρολογήστε έναν τίτλο" autofocus/>
19                 </img>
20             </form>
21         </div>
```




HTML (2)

```
<div class="movie hidden">
  <div class="movie-poster">
    <img id="image" class="valign-middle" alt='Μη διαθέσιμη εικόνα'></img>
  </div>
  <div class="movie-info">
    <div class="basic">
      <div class="block-text text-left">
        <a id="favoriteLink" href="#">
          </img>
        </a>
      </div>
      <div id="title" class="block-text movie-title text-left">
        Τίτλος Ταινίας
      </div>
      <div class="inline text-left">
        <span id="year" class="movie-text">Έτος παραγωγής</span>
        <span id="runtime" class="movie-text">Διάρκεια</span>
        <span id="genre" class="movie-text">Είδος</span>
      </div>
    </div>
  </div>
</div>
```



HTML (3)

```
<div class="inline text-left">
  <a id="imdbId" target="blank"></img></a>
  <span id="imdbRating" class="movie-text rating" data-scale="10">Βαθμολογία</span>
</div>
<div class="block-text movie-text text-left">
  <p id="plot" class="collapsible"></p>
</div>
</div>
<div class="extended text-left hidden">
  <div id="director" class="movie-text"><b>Σκηνοθεσία: </b><span></span></div>
  <div id="actors" class="movie-text"><b>Ηθοποιοί: </b><span></span></div>
  <div id="production" class="movie-text"><b>Παραγωγή: </b><span></span></div>
  <div id="boxOffice" class="movie-text"><b>Box Office: </b><span></span></div>
  <div id="language" class="movie-text"><b>Γλώσσα: </b><span></span></div>
  <div id="rated" class="movie-text"><b>Καταλληλότητα: </b><span></span></div>
</div>
<a id="showMore" class="show-more" href="#">Περισσότερα
  </img>
</a>
</div>
</div>
</div>
```



HTML (4)

AJAX

```
<div class="not-found hidden">
|   <span>Δε βρέθηκαν ταινίες σύμφωνα με τα κριτήρια αναζήτησης.</span>
</div>
<div class="error hidden">
|   <span>Παρουσιάστηκε σφάλμα κατά την κλήση της υπηρεσίας.</span>
</div>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.3/jquery.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/lodash.js/4.17.21/lodash.min.js"></script>
<script type="module" src="./movies.js"></script>
</body>
</html>
```



CSS (1)

```
1  body {
2      background: #6a4104;
3      margin: 0;
4  }
5
6  .center {
7      text-align: center;
8  }
9
10 .bot-gap {
11     margin-bottom: 20px;
12 }
13
14 .title {
15     color: #fff;
16     font-size: 28px;
17     text-shadow: 0 2px 4px rgba(0, 0, 0, 0.4);
18 }
19
```



CSS (2)

```
20  input[type=text].search {
21      width: 200px;
22      height: 50px;
23      background-image: url('../img/search.png');
24      background-position: left;
25      background-repeat: no-repeat;
26      background-size: 45px 45px;
27      padding-left: 45px;
28  }
29
30  input[type=text].rounded {
31      border-radius: 18px;
32      border: 1.5px solid black;
33  }
34
35  /* Remove focus border added by browser */
36  input.search:focus {
37      outline: none;
38  }
```



CSS (3)

```
44  ✓ .movie {
45      display: inline-flex;
46      flex-wrap: wrap;
47  }
48
49  ✓ .movie-poster {
50      width: 300px;
51      height: 450px;
52      margin-right: 15px;
53  }
54
55  ✓ .movie-info {
56      width: 450px;
57      height: 450px;
58  }
59
60  ✓ .basic > div {
61      margin-bottom: 15px;
62  }
63
64  ✓ .block-text {
65      display: block;
66  }
```

```
68  .text-left {
69      text-align: left;
70  }
71
72  .movie-title {
73      color: ■ #fff;
74      font-size: 28px;
75      font-family: monospace;
76      margin-bottom: 15px;
77  }
78
79  .movie-text {
80      color: ■ #fff;
81      text-align: left;
82      font-family: monospace;
83  }
84
85  .inline > span {
86      margin-right: 15px;
87  }
88
89  .rating:after {
90      content: " / " attr(data-scale);
91  }
```



CSS (4)

AJAX

```
94  ∨ p.collapsible.expanded {
95      max-height: 500px;
96      transition: max-height 2.25s ease-in;
97      white-space: normal;
98  }
99
100  ∨ p.collapsible {
101      max-height: 1.25em;
102      transition: max-height 2.15s ease-out;
103      overflow: hidden;
104      white-space: nowrap;
105      text-overflow: ellipsis;
106  }
107
108  ∨ .extended {
109      margin-bottom: 10px;
110  }
111
112  ∨ .show-more {
113      border: 1px solid grey;
114      padding: 15px 10px;
115      text-decoration: none;
116      color: wheat;
117  }
```

```
119  .show-more:hover {
120      text-decoration: underline;
121  }
122
123  .icon-small {
124      width: 30px;
125      height: 30px;
126      margin-left: 8px;
127  }
128
129  .hidden {
130      display: none;
131  }
132
133
134  .valign-middle {
135      vertical-align: middle;
136  }
137
138  .valign-bottom {
139      vertical-align: bottom;
140  }
```



JS (1)

AJAX

```
1  $(document).ready(function() {
2      var debounceTimeout = null
3      $("#searchInput").on('input', function() {
4          clearTimeout(debounceTimeout)
5          debounceTimeout = setTimeout(() => getMovie(this.value.trim()), 1500)
6      })
7
8      $('#showMore').on('click', function() {
9          onShowMoreClicked()
10     })
11 })
12
13 /**
14  * Uses the movie title provided by the user to search and show the corresponding movie.
15  */
16 function getMovie(title) {
17     if (!title) {
18         return
19     }
20     onBeforeSend()
21     fetchMovieFromApi(title)
22 }
```




JS (2)

AJAX

```
24  /**
25   * Fetches a movie from the Movies API.
26   * This function defines handling for both successful and failed(movie not found, api unavailable etc.) responses
27   */
28  function fetchMovieFromApi(title) {
29      let ajaxRequest = new XMLHttpRequest()
30      ajaxRequest.open("GET", `http://www.omdbapi.com/?t=${title}&apikey=${API_KEY}`, true)
31      ajaxRequest.timeout = 5000 //timeout after 5 seconds
32      ajaxRequest.ontimeout = (e) => onApiError()
33      ajaxRequest.onreadystatechange = function() {
34          if (ajaxRequest.readyState == 4)
35          {
36              if(ajaxRequest.status === 200) {
37                  handleResults(JSON.parse(ajaxRequest.responseText))
38              }
39              else {
40                  onApiError()
41              }
42          }
43      }
44      ajaxRequest.send()
45  }
```



JS (3)

AJAX

```
49  /**
50   * Determines if the API found a movie or not.
51   * If the movie is found, the API response is transformed and then shown.
52   * Otherwise, show a not found message.
53   */
54  function handleResults(result) {
55      if (result.Response === 'True') {
56          let transformed = transformResponse(result)
57          buildMovie(transformed)
58      } else if (result.Response === 'False') {
59          hideComponent('#waiting')
60          showNotFound()
61      }
62  }
63
```

- Η `handleResults` έχει δύο μέρη: 1) **να κάνει transform** το αρχικό response (να κάνει τα keys camel-case, τα πεδία που είναι κενά να τα κάνει «-»), να φτιάξει ένα link προς το imdb), και 2) να εμφανίζει στο UI με την `buildMovie`



JS (4)

AJAX

```
64  /**
65   * Assigns transformed API response to the corresponding UI elements.
66   */
67  function buildMovie(apiResponse) {
68    if (apiResponse.poster) { //show a poster, if it is available
69      $('#image').attr('src', apiResponse.poster).on('load', function() { //wait for the poster to load
70        buildMovieMetadata(apiResponse, $(this))
71      })
72    } else { //show every other detail of the movie
73      buildMovieMetadata(apiResponse)
74    }
75  }
```

- Η buildMovie, κάνει δύο πράγματα: 1) εισάγει στο UI το poster αν υπάρχει αντικαθιστώντας το προηγούμενο, 2) εισάγει τα metadata της ταινίας στο UI
- Αν το poster δεν υπάρχει, απλά καταργείται το παλιό



JS (5)

AJAX

```
77  /**
78   * Actions to take before the search query is sent, like hiding any previous information about a movie.
79   */
80  function onBeforeSend() {
81      showComponent('#waiting')
82      hideComponent('.movie')
83      hideNotFound()
84      //resetFavorite()
85      hideError()
86      collapsePlot()
87      hideExtras()
88  }
```

- Όταν ξεκινήσει η αναζήτηση εμφανίζεται ένα waiting gif, και αποκρύπτονται προηγούμενα elements



JS (6)

```
90  /**
91   * Actions to take if the Movie API fails to respond.
92   */
93  function onApiError() {
94      hideComponent('#waiting')
95      showError()
96  }
97
98  /**
99   *Adds the metadata of the movie to the appropriate fields.
100  */
101  function buildMovieMetadata(apiResponse, imageTag) {
102      hideComponent('#waiting')
103      handleImage(imageTag)
104      handleLiterals(apiResponse)
105      showComponent('.movie')
106  }
```



JS (7)

```
108  /**
109   * Shows the movie poster if any, otherwise just hide.
110   */
111  function handleImage(imageTag) {
112      imageTag ? $('#image').replaceWith(imageTag) : $('#image').removeAttr('src')
113  }
114
115  /**
116   * Fills the values of the corresponding HTML elements using the API Response.
117   */
118  function handleLiterals(apiResponse) {
119      $('.movie').find('[id]').each((index, item) => { //find all items in div with class movie that have an id
120          if ($(item).is('a')) { //if it's a link, then update the href
121              $(item).attr('href', apiResponse[item.id])
122          } else { //for every other element just update the text
123              let valueElement = $(item).children('span');
124              let metadataValue = apiResponse[item.id] ? apiResponse[item.id] : '- '
125              valueElement.length ? valueElement.text(metadataValue) : $(item).text(metadataValue)
126          }
127      })
128  }
```

- Όπως αναφέρθηκε το image αν υπάρχει, αντικαθιστά το προηγούμενο, αλλιώς αν δεν υπάρχει το νέο src είναι κενό
- Επίσης, τα metadata εισάγονται στο UI με την handleLiterals



JS (8)

AJAX

```
130  ∨ /**
131    * Transforms API response (ie N/A values with empty string, build the imdb url based on imdb id).
132    */
133  ∨ function transformResponse(apiResponse) {
134    let camelCaseKeysResponse = camelCaseKeys(apiResponse)
135    clearNotAvailableInformation(camelCaseKeysResponse)
136    buildImdbLink(camelCaseKeysResponse)
137    return camelCaseKeysResponse
138  }
139
140  ∨ /**
141    * Transforms the object keys of the API Response to camel case.
142    */
143  ∨ function camelCaseKeys(apiResponse) {
144    return _.mapKeys(apiResponse, (v, k) => _.camelCase(k))
145  }
146
```

- Η `_mapKeys` είναι συνάρτηση της Lodash



JS (9)

AJAX

```
147  /**
148   * Transforms the imdb id given by the API Response to the corresponding imdb url.
149   */
150  function buildImdbLink(apiResponse) {
151      if (apiResponse.imdbId && apiResponse.imdbId !== 'N/A') {
152          apiResponse.imdbId = `https://www.imdb.com/title/${apiResponse.imdbId}`
153      }
154  }
155
156  /**
157   * Replaces the API Response from N/A (= Not Available) to empty string values.
158   */
159  function clearNotAvailableInformation(apiResponse) {
160      for (var key in apiResponse) {
161          if (apiResponse.hasOwnProperty(key) && apiResponse[key] === 'N/A') {
162              apiResponse[key] = ''
163          }
164      }
165  }
166
```




JS (10)

```
167  ✓ function onShowMoreClicked() {
168      $('#plot').toggleClass('expanded')
169  ✓   if($('#.extended').is(':visible')) {
170
171      |       $('#.extended').hide(700)
172      |   }
173  ✓   else {
174      |       $('#.extended').show(700)
175      |   }
176  }
177
178  ✓ /**
179      * Hides a component identified by the provided jquery selector.
180      * The component is returned for further chained calls.
181      */
182  ✓ function hideComponent(jQuerySelector) {
183      |   return $(jQuerySelector).addClass('hidden')
184      |   }
185  }
```



JS (11)

```
185
186  ✓ /**
187     * Shows a component identified by the provided jquery selector.
188     * The component is returned for further chained calls.
189     */
190  ✓ function showComponent(jQuerySelector) {
191      return $(jQuerySelector).removeClass('hidden')
192  }
193
194  ✓ function showNotFound() {
195      $('.not-found').clone().removeClass('hidden').appendTo($('.center'))
196  }
197
198  ✓ function hideNotFound() {
199      $('.center').find('.not-found').remove()
200  }
201
202  ✓ function showError() {
203      $('.error').clone().removeClass('hidden').appendTo($('.center'))
204  }
205
```



JS (12)

```
206  ✓ function hideError() {  
207      |     $(' .center').find(' .error').remove()  
208      | }  
209  
210  ✓ function hideExtras() {  
211      |     $(' .extended').hide()  
212      | }  
213  
214  ✓ function collapsePlot() {  
215      |     | $('#plot').removeClass('expanded')  
216      | }  
217
```



- Διερευνήστε το Open weather-map API <https://openweathermap.org/api> και προσπαθήστε να αναπτύξετε μία εφαρμογή πρόγνωσης του καιρού όπου ο χρήστης θα αναζητά μία πόλη (π.χ. Athens) και θα εμφανίζει την πρόγνωση για αυτή την πόλη