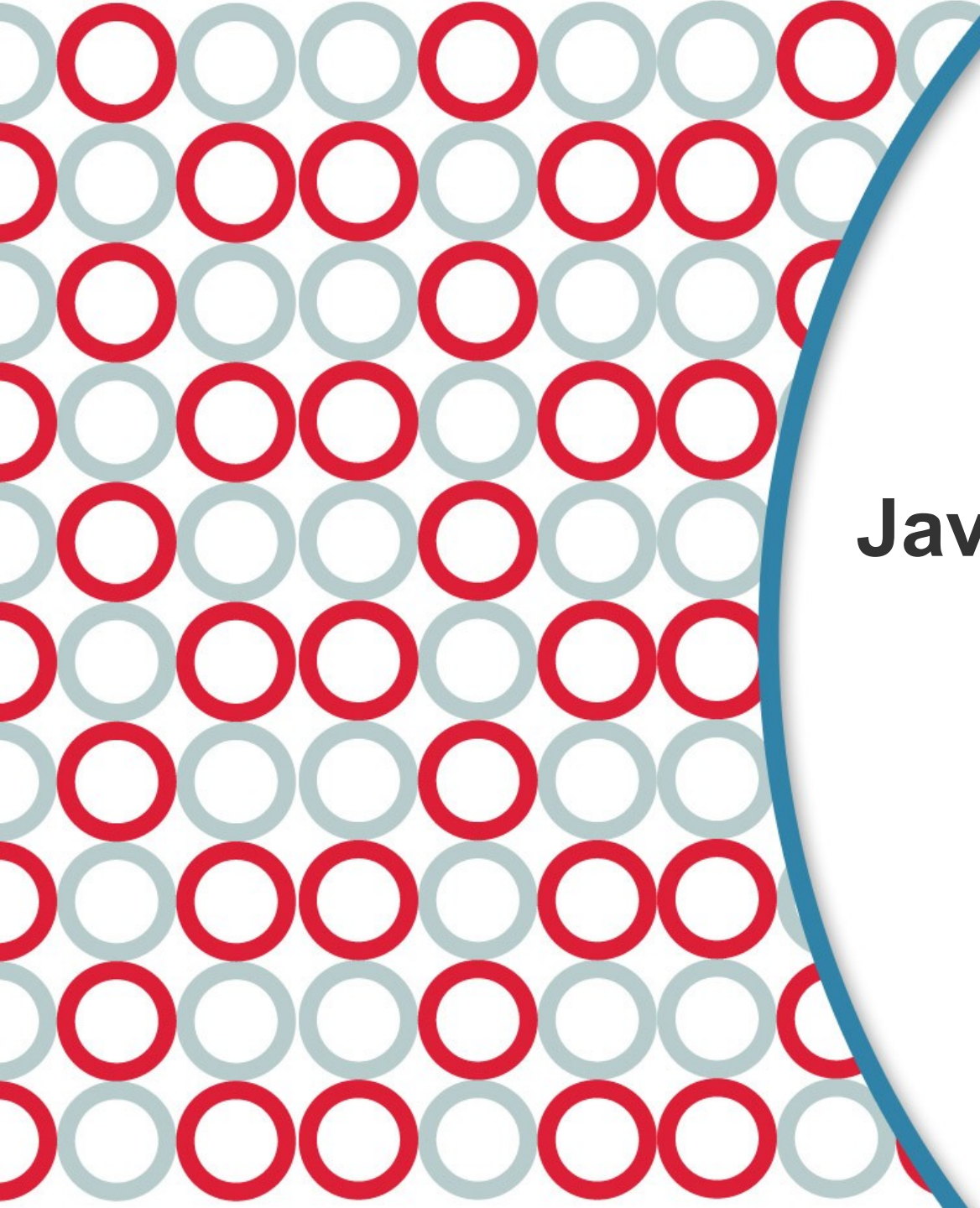


May 2019, IPT Course
Java Web Debelopment

Asyncronous JavaScript & XML (AJAX)

Trayan Iliev
tiliev@iproduct.org
http://iproduct.org

About me

Trayan Iliev

– CEO of IPT – Intellectual Products & Technologies

– Oracle® certified programmer 15+ Y

– end-to-end reactive fullstack apps with Java, ES6/7, TypeScript, Angular, React and Vue.js

– 12+ years IT trainer

– Voxxed Days, jPrime, jProfessionals, BGOUG, BGJUG, DEV.BG speaker

– Organizer RoboLearn hackathons and IoT enthusiast (http://robolearn.org)

Where to Find the Code?

Java Web Development projects and examples are available @ GitHub:

https://github.com/iproduct/course-java-web-development

Agenda for This Session

- Event Driven JS Programming – DOM Level 2
- Ajax
- Practical examples

Event Driven Programming in JavaScript™

- Main models:
  - DOM Level 0 (original Netscape model)

```
<a href="#" onclick= "alert('I\'m clicked!'); return false;" />
```

  - Traditional model (as properties)

```
anElem.onclick = function() { this.style.color = 'red'; }
```

- can register multiple event handlers:

```
var oldHandler = (anElem.onclick) ? anElem.onclick : function (){ };
anElem.onclick = function () {oldHandler(); this.style.color = 'red'; };
```

  - DOM Level 2 Event Handling Model
  - Microsoft Event Handling Model

W3C DOM Level 2 Event Handling Model

- Three phases in event handling life-cycle:
  - Capturing phase – from document to target element
  - At Target phase – processsing in the target element
  - Bubbling phase – returns back from target to document
- All events go through Capturing phase, but not all through Bubbling phase – only low level (raw) events
- event.stopPropagation() - stops further processing
- event.preventDefault() - prevents standards event processing
- Register/deregister event handlers:

anElement.addEventListener('click', eventListener, false)

anElement.removeEventListener('click', eventListener, false)

Microsoft Event Handling Model

- Register/deregister event handlers:

```
anElement.attachEvent('onclick', eventListener)
anElement.detachEvent('onclick', eventListener)
```

- Callback function *eventListener* does not receive *event* object:

```
function crossBrowserEventHandler(event) {
  if(!event) event = window.event;  ... // processing follows ... }
```

- No Capturing phase – every element has methods setCapture() and releaseCapture()
- from document towards target element
- window.event.cancelBubble = true; // stops bubbling -a
- window.event.returnValue=false; // prevents default action

W3C DOM Level 2 Events and APIs

Име на интерфейса	Събития
Event	abort, blur, change, error, focus, load, reset, resize, scroll, select, submit, unload
MouseEvent	click, mousedown, mousemove, mouseout, mouseover, mouseup
UIEvent	DOMActivate, DOMFocusIn, DOMFocusOut

Web 2.0

- Web 2.0 – Internet as an interaction platform for creating and sharing content – blogs, RSS/Atom, comments, pictures, audio, wideo (social media)
- Collaborative knowledge construction –Participatory, Decentralized, Linked, Emergent
- Network effect, comstant evolution
- Taksonomy -> Folksonomy
- Open standards free software
- Service-Oriented Architectures (SOA)
- Separation of data and presentation
- Rich interactive user interface (RIA)

Rich Internet Applications (RIA)

- Rich Internet Application (RIA) – feature rich web application:
  - Rich UI – drag-and-drop, animation effects, local processing, rich UI components – buttons, menus, tab panels, sliders, progress indicators
  - Reactive UI
  - Client-server ballance
  - Asynchronous communication
  - Network efficiency
- No need for installation, automatic update
- Universal accessibility independent of place and device

Asynchronous JavaScript & XML - AJAX

- Ajax – A New Approach to Web Applications, J. Garrett February, 2005

  http://www.adaptivepath.com/publications/essays/archives/000385.php
- Presentation based on standards HTML 5 / XHTML, CSS
- Dynamic visualisation and interaction using Document Object Model (DOM)
- Exchange and manipulation of data using XML and XSLT or JavaScript Object Notation (JSON)
- Asynchronous data fetch using **XMLHttpRequest**
- And JavaScript who wrapps everything above in one application

AJAX and Traditional Web Applications

Main difference:

- Ajax apps are based on processing of **events** and **data**
- Traditional web applications are based on presenting pages and hyperlink transitions between them

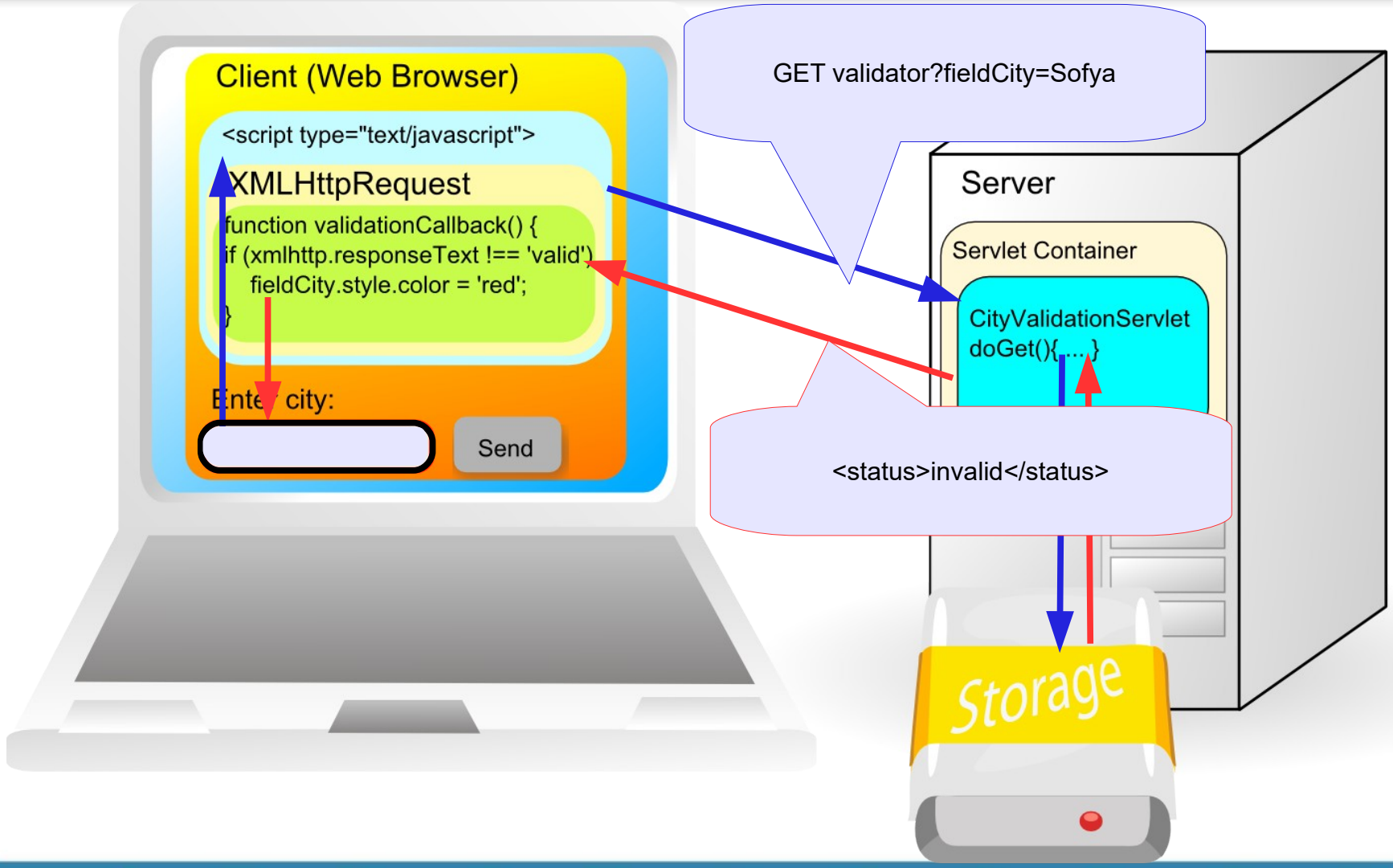
Problems connected with AJAX

- Sandboxing
- Scripting switched off
- Spped of client processing
- Time for script download
- Loosing integrity
- Search engine indexing
- Accessibility
- More complex development
- More complex profiling – 2 cycles
- Cross Domain AJAX

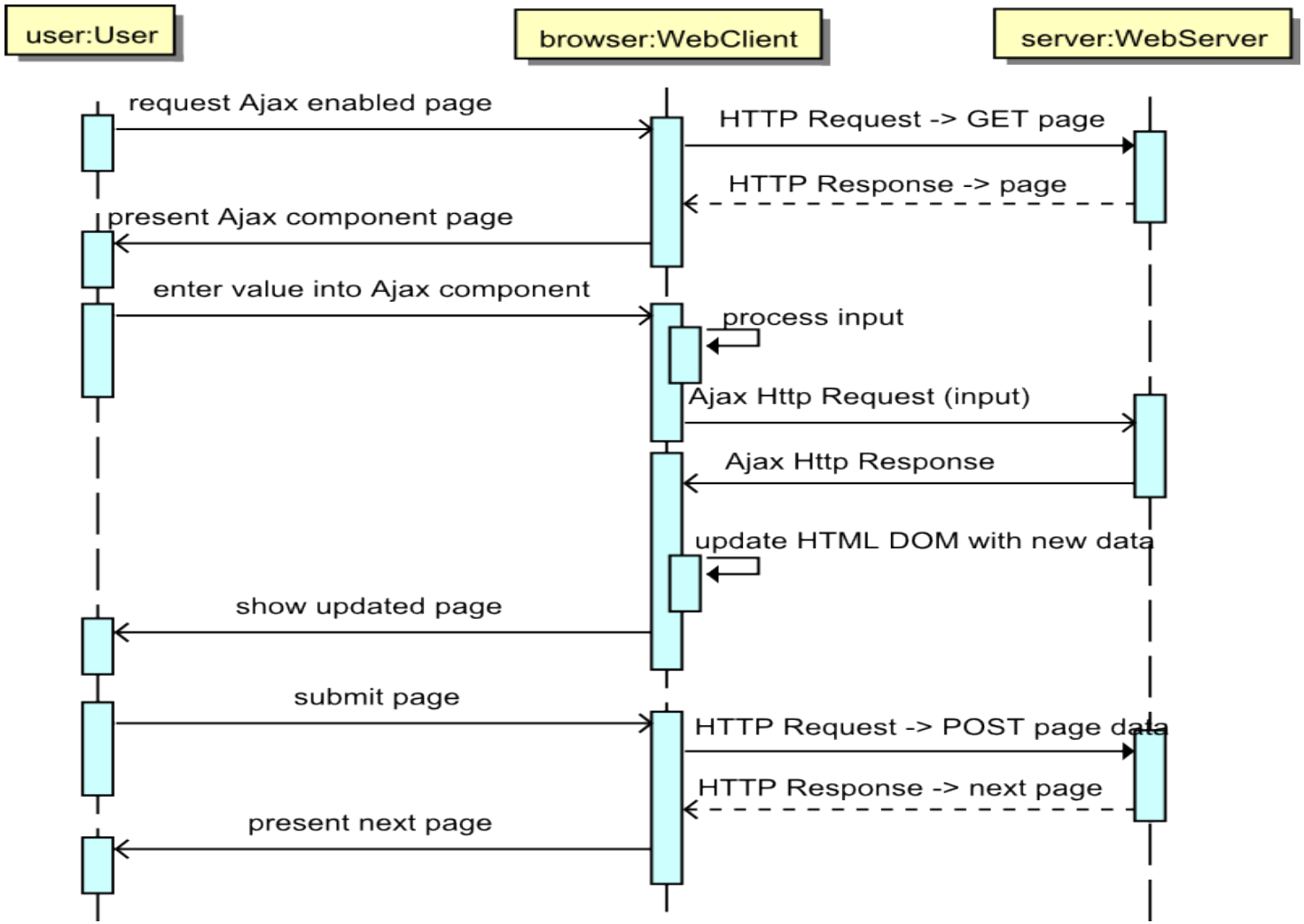
AJAX and Evolution of Web

- JavaScript
- Фреймове
- Скрити фреймове
- DHTML и DOM
- IFrames
- XMLHttpRequest
- New Fetch API

AJAX Interactions



AJAX Interactions Flowchart



AJAX Technologies

- HTML 5 & XHTML
- JavaScript, DHTML и DOM
- CSS
- XML и XSLT
- XMLHttpRequest
- SOAP, WSDL, UDDI, REST
- JSON, JSONP, ...

AJAX Examples

- Google Suggest
- Gmail
- Google Maps
- Google Docs
- Google Calender
- Product Search on Amazon – A9
- Blogger
- Yahoo! News
- and many more

Basic Structure of Synchronous AJAX Request

```
var method = “GET”;
var url = “resources/ajax_info.html”;

if (window.XMLHttpRequest)   {// IE7+, Firefox, Safari, Chrome,
Opera,
        xmlhttp=new XMLHttpRequest();
    }  else {// IE5, IE6
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
}
```


```
xmlhttp.open(method, url, false);
xmlhttp.send();
document.getElementById("results").innerHTML =
        xmlhttp.responseText;
```

AJAX Request with XML Processing and Authentication

```
if (window.XMLHttpRequest)   {// IE7+, Firefox, Safari, Chrome, Opera,
        xmlhttp=new XMLHttpRequest();
    }  else {// IE5, IE6
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
}
xmlhttp.open("GET", "protected/product_catalog.xml", false,
                                                    "trayan", "mypass");
xmlhttp.send();
if (xmlhttp.status == 200 &&
          xmlhttp.getResponseHeader("Content-Type") == "text/xml") {
        var xmlDoc = xmlhttp.responseXML;
        showBookCatalog(xmlDoc);  // Do something with xml document
    }
}
```

AJAX Request with XML Processing (2)

```
function showBookCatalog(xmlDoc){
    txt="<table><tr><th>Title</th><th>Artist</th></tr>";
  var x=xmlDoc.getElementsByTagName("TITLE");
  var y=xmlDoc.getElementsByTagName("AUTHOR");
  for (i=0;i<x.length;i++) {
    txt=txt +"<tr><td>"
      + x[i].firstChild.nodeValue
      + "</td><td>"+ y[i].firstChild.nodeValue
      + "</td></tr>";
  }
  txt += "</table>"
  document.getElementById("book_results").innerHTML=txt;
}
```

Basic Structure of Asynchronous AJAX Request

```
 if (window.XMLHttpRequest)   {// IE7+, Firefox, Safari, Chrome,
Opera,
        xmlhttp=new XMLHttpRequest();
 }  else {// IE5, IE6
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
 }
 xmlhttp.onreadystatechange = function(){
        if (xmlhttp.readyState==4 && xmlhttp.status==200){
            callback(xmlhttp);
        }
 }
xmlhttp.open(method, url, true);
xmlhttp.setRequestHeader("Content-type","application/x-www-
form-              urlencoded");
xmlhttp.send(paramStr);
```

Callback function

isAsynchronous = true

XMLHttpRequest.readyState

Код	Значение
1	след като XMLHttpRequest.open() е извикан успешно
2	заглавните части на отговора на HTTP заявката (HTTP response headers) са успешно получени
3	начало на зреждане на съдържанието на HTTP отговора (HTTP response content)
4	съдържанието на HTTP отговора е заредено успешно от браузъра

HTTP Request Headers

- В **HTTP 1.0** всички заглавни части са опционални
- В **HTTP 1.1** са опционални всички заглавни части без **Host**
- Необходимо е винаги да се проверява дали съответната заглавна част е различна от **null**

HTTP Requests Headers - RFC2616

- Accept
- Accept-Charset
- Accept-Encoding
- Accept-Language
- Accept-Language
- Authorization
- Connection
- Content-Length
- Cookie
- Host
- If-Modified-Since
- If-Unmodified-Since
- Referer
- User-Agent

HTTP Request Structure

GET */context/Servlet* **HTTP/1.1**

Host: *Client_Host_Name*

Header2: *Header2_Data*

...

HeaderN: *HeaderN_Data*

<Празен ред>

POST */context/Servlet* **HTTP/1.1**

Host: *Client_Host_Name*

Header2: *Header2_Data*

...

HeaderN: *HeaderN_Data*

<Празен ред>

POST_Data

HTTP Response Structure

HTTP/1.1 200 OK

Content-Type:
application/json

Header2: *Header2_Data*

…

HeaderN: *HeaderN_Data*

<Празен ред>

```
[{  "id":1,
    "name":"Novelties in Java EE 7 ...",
    "description":"The presentation
     is ...",
    "created":"2014-05-10T12:37:59",
    "modified":"2014-05-10T13:50:02",
 },
 {  "id":2,
    "name":"Mobile Apps with
     HTML5 ...",
    "description":"Building Mobile ...",
    "created":"2014-05-10T12:40:01",
    "modified":"2014-05-10T12:40:01",
 }]
```

Response Status Codes

- 100 Continue
- 101 Switching Protocols
- 200 OK
- 201 Created
- 202 Accepted
- 203 Non-Authoritative Information
- 204 No Content
- 205 Reset Content
- 301 Moved Permanently
- 302 Found
- 303 See Other
- 304 Not Modified
- 307 Temporary Redirect
- 400 Bad Request
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found

Response Status Codes

- 405 Method Not Allowed
- 415 Unsupported Media Type
- 417 Expectation Failed
- 500 Internal Server Error
- 501 Not Implemented
- 503 Service Unavailable
- 505 HTTP Version Not Supported

HTTP Response Headers

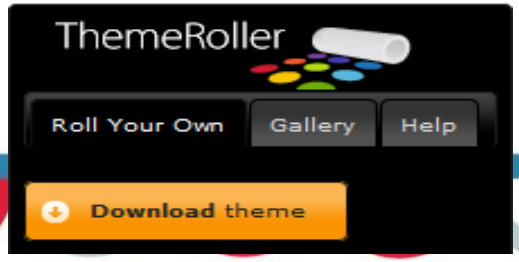
- Allow
- Cache-Control
- Pragma
- Connection
- Content-Disposition
- Content-Encoding
- Content-Language
- Content-Length
- Content-Type
- Expires
- Last-Modified
- Location
- Refresh
- Retry-After
- Set-Cookie
- WWW-Authenticate

jQuery

- jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development [http://jquery.com/]
- Lightweight Footprint - about 31KB in size (Minified and Gzipped)
- Easy-to-use but powerfull – Ajax, Attributes, Callbacks Object, Core, CSS, Data, Deferred Object, Dimensions, Effects, Events, Forms, Internals, Manipulation, Miscellaneous, Offset, Plugins, Properties, Selectors, Traversing, Utilities
- Widespread JS library with many third-party plugins

jQuery [2]

- Supports CSS 3 selectors and much more
- Cross-browser – IE 6.0+, FF 10+, Safari 5.0+, Opera, Chrome
- Supports own layout and presentation widgets – jQueryUI
  - Interactions – Draggable, Droppable, Resizable, Selectable, Sortable
  - Widgets – Accordion, Autocomplete, Button, Datepicker, Dialog, Menu, Progressbar, Slider, Spinner, Tabs, Tooltip
  - Effects – Add Class, Color Animation, Effect, Hide, Remove Class, Show, Switch Class, Toggle, Toggle Class
  - Utilities – Position, Widget Factory
- Supports custom themes (CSS)



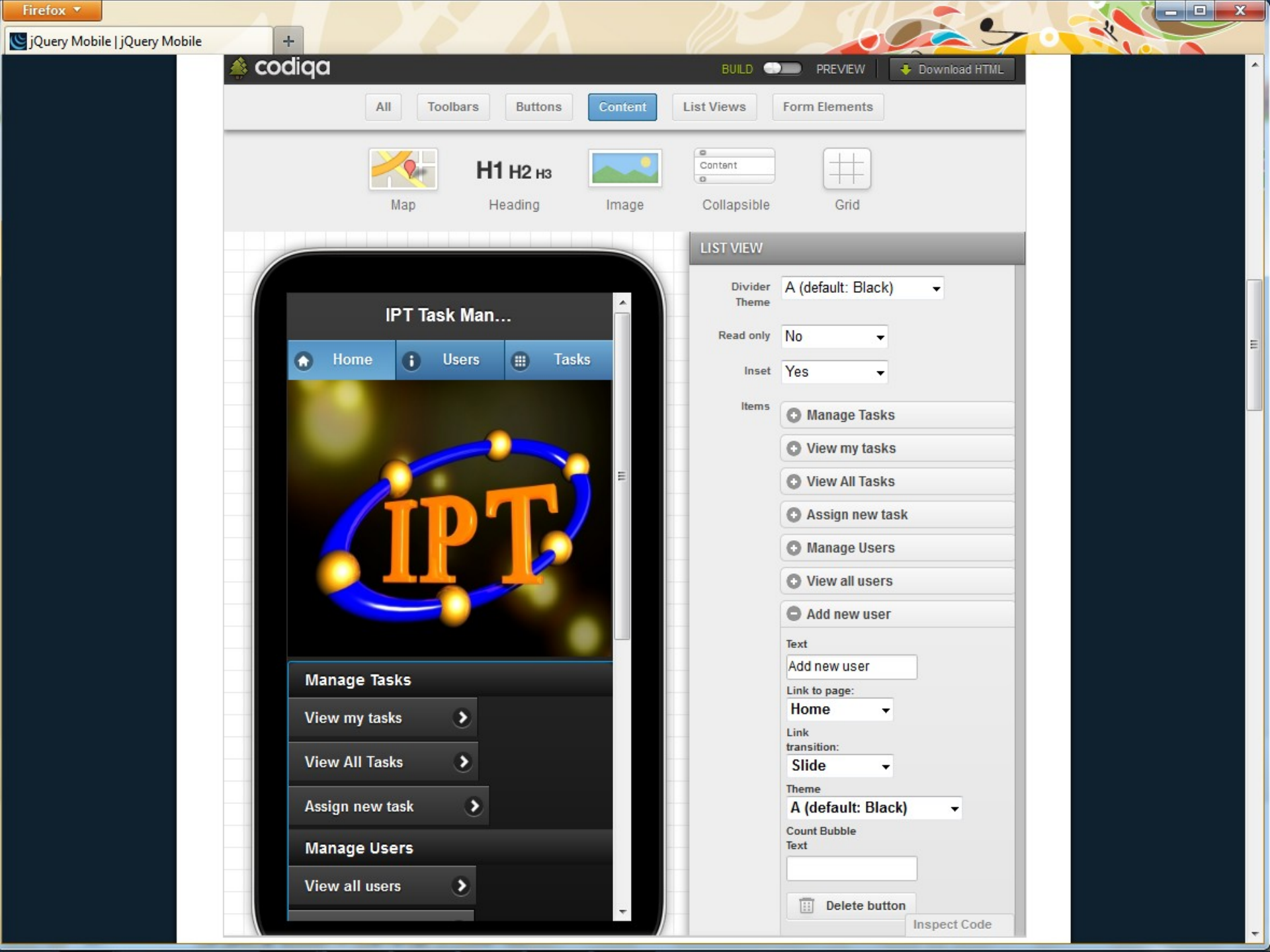
jQuery Mobile

- jQuery Mobile: Touch-Optimized Web Framework for Smartphones & Tablets – A unified, HTML5-based user interface system for all popular mobile device platforms, built on the rock-solid jQuery and jQuery UI foundation. Its lightweight code is built with progressive enhancement, and has a flexible, easily themeable design [http://jquerymobile.com/]

- jQuery Mobile is a separate project for building standard compliant (HTML 5, CSS 3, WAI-ARIA) mobile applications – now v1.2.0 Final

jQuery Mobile Main Features [1]

- HTML5 Markup-driven configuration
- Progressive Enhancement principles
- Scalable and adaptable for different screen sizes and devices
- Powerful Ajax-based navigation out-of-the-box – usability
- Accessibility – WAI-ARIA supp
- Touch and mouse events
- Unified UI widgets
- Custom CSS theming support – ThemeRoller
- Easy integration with PhoneGap for additional functionality


Firefox
jQuery Mobile | jQuery Mobile
codiqa
BUILD
PREVIEW
Download HTML
All
Toolbars
Buttons
Content
List Views
Form Elements
Map
H1 H2 H3
Heading
Image
Content
Collapsible
Grid
IPT Task Man...
Home
Users
Tasks
IPT
Manage Tasks
View my tasks
View All Tasks
Assign new task
Manage Users
View all users
LIST VIEW
Divider Theme
A (default: Black)
Read only
No
Inset
Yes
Items
Manage Tasks
View my tasks
View All Tasks
Assign new task
Manage Users
View all users
Add new user
Text
Add new user
Link to page:
Home
Link transition:
Slide
Theme
A (default: Black)
Count Bubble Text
Delete button
Inspect Code

Cross-Origin Resource Sharing (CORS)

- Aloows requests to services in domains different from the domain of the calling script. The server distinguishes between different requests, and decides wich to permit based on the **Origin** header, the method (GET, POST), the custom headers, etc.

- To make a cross-domain HTTP request, when the method is different from simple GET or HEAD, a preflight OPTIONS request is made by the browser. The server responds to this preflight request with information which methods and custom headers are allowed for the requested resource, based on the Origin of the caller script.

New CORS HTTP Headers

- HTTP GET request

GET /crossDomainResource/ HTTP/1.1

Referer: http://sample.com/crossDomainMashup/

Origin: http://sample.com

- HTTP GET response

Access-Control-Allow-Origin: http://sample.com

Content-Type: application/xml

New HTTP Headers when POST with CORS

- HTTP OPTIONS preflight request

```
OPTIONS /crossDomainPOSTResource/ HTTP/1.1
Origin: http://sample.com
Access-Control-Request-Method: POST
Access-Control-Request-Headers: MYHEADER
```

- HTTP response

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://sample.com
Access-Control-Allow-Methods: POST, GET, OPTIONS
Access-Control-Allow-Headers: MYHEADER
Access-Control-Max-Age: 864000
```

New Fetch API

[https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API]

- The **Fetch API** provides an interface for fetching resources like XMLHttpRequest, but more powerful and flexible feature set.
- Promise<Response> WorkerOrGlobalScope.fetch(input[, init])
  - input - resource that you wish to fetch – url string or Request
  - init - custom settings that you want to apply to the request: **method**: (e.g., GET, POST), **headers, body**(Blob, BufferSource, FormData, URLSearchParams, or USVString), **mode**: (cors, no-cors, or same-origin), **credentials**(omit, same-origin, or include. to automatically send cookies this option must be provided), **cache**: (default, no-store, reload, no-cache, force-cache, or only-if-cached), **redirect** (follow, error or manual), **referrer** (default is client), **referrerPolicy**: (no-referrer, no-referrer-when-downgrade, origin, origin-when-cross-origin, unsafe-url), **integrity** (subresource integrity value of request)

References

- jQuery JS library - http://jquery.com/
- Representational state transfer (REST) in Wikipedia – http://en.wikipedia.org/wiki/Representational_state_transfer
- JavaScript Object Notation (JSON) – http://www.json.org/
- Fielding's blog discussing REST – http://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven
- Representational state transfer (REST) in Wikipedia – http://en.wikipedia.org/wiki/Representational_state_transfer
- Hypermedia as the Engine of Application State (HATEOAS) in Wikipedia – http://en.wikipedia.org/wiki/HATEOAS
- JavaScript Object Notation (JSON) – http://www.json.org/

Thank's for Your Attention!

Trayan Iliev

CEO of IPT – Intellectual Products & Technologies

http://iproduct.org/

http://robolearn.org/

https://github.com/iproduct

https://twitter.com/trayaniliev

https://www.facebook.com/IPT.EACAD

https://plus.google.com/+IproductOrg