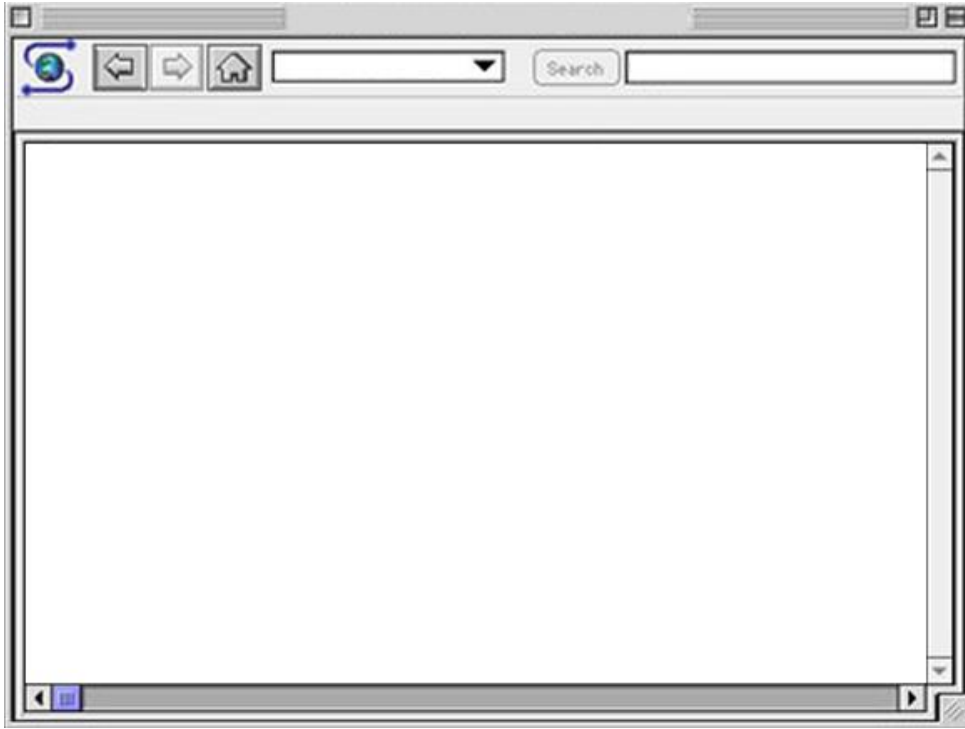


SYNERGY • 20
13

DataFlex Web Application Symposium – Part 4

Understanding the Web Framework

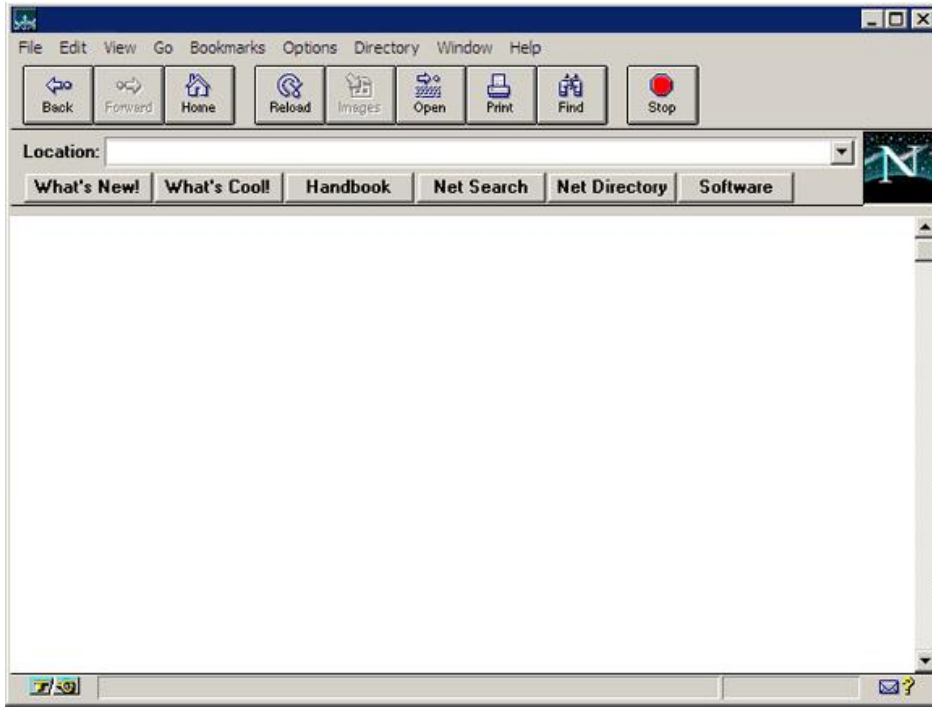
Harm Wibier
Development Team
www.dataaccess.com



1993

MOSAIC 0.1

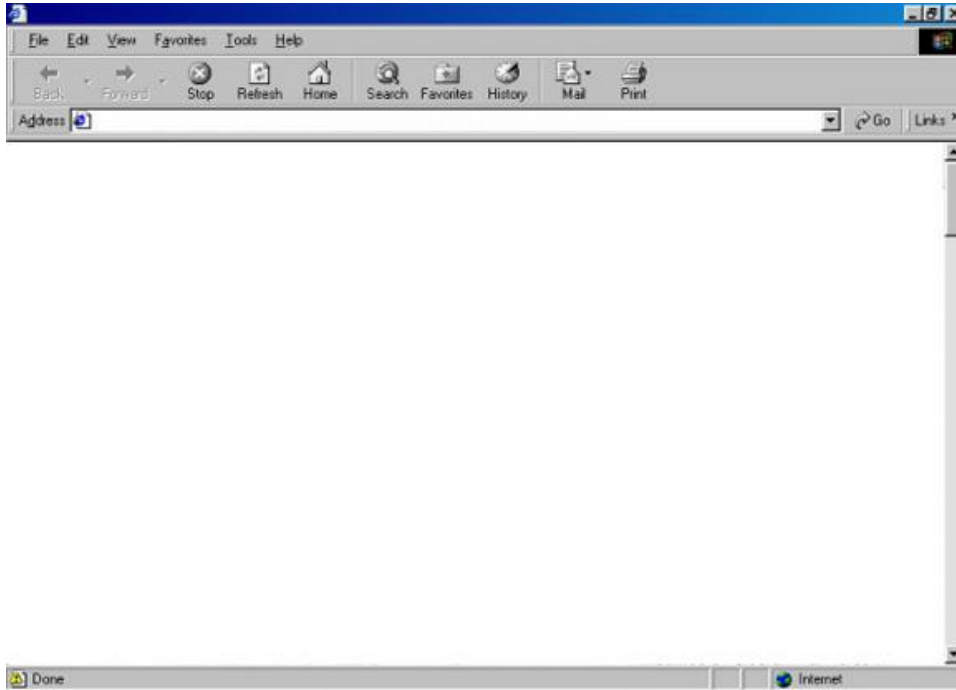
- HTTP
- Hyperlinks
- HTML
- Images



1996

Netscape 2.0

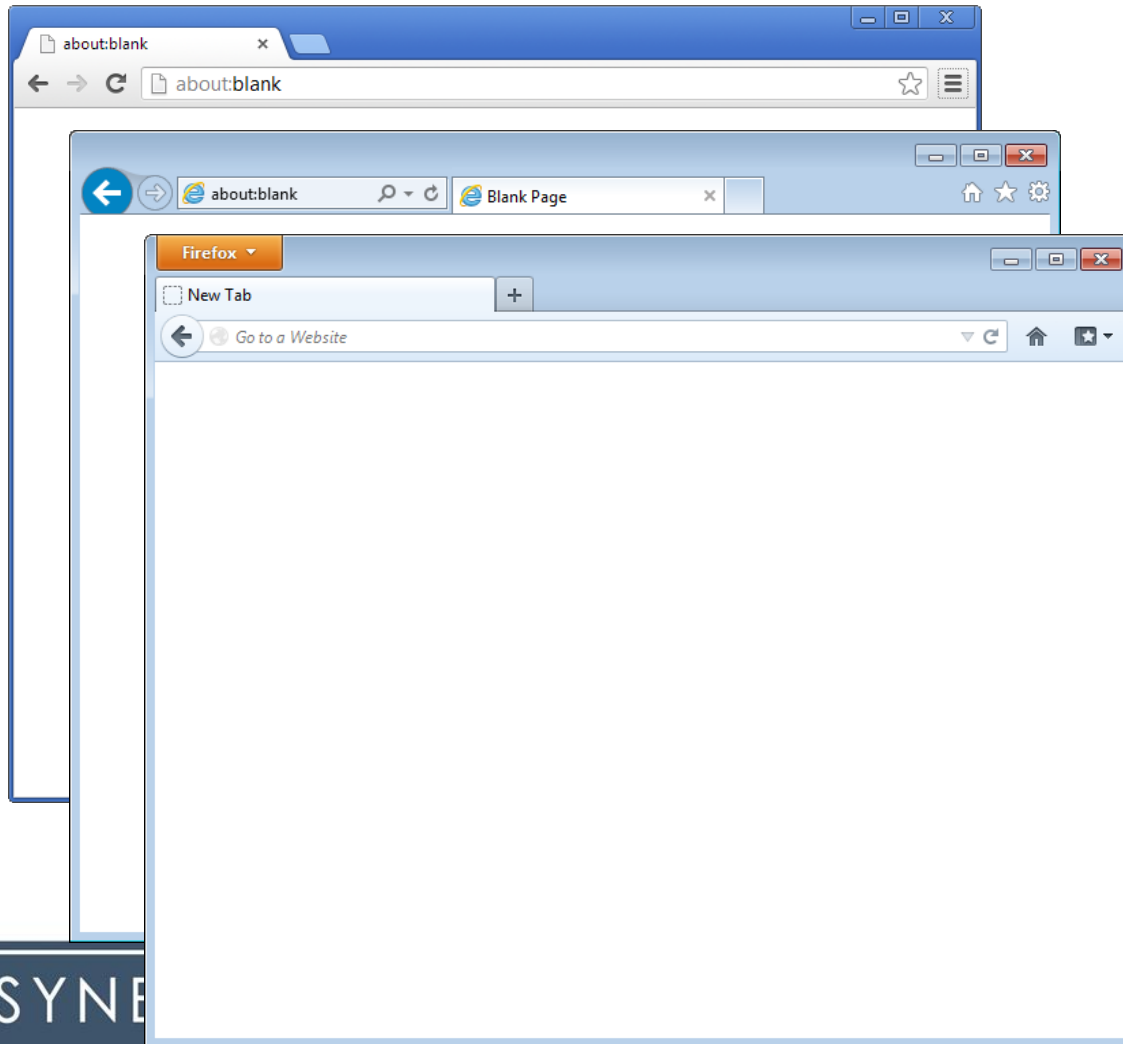
- Forms
- Cookies
- JavaScript



1999

Internet Explorer 5

- CSS
- AJAX (XmlHttpRequest)



2013
Chrome
FireFox
Internet Explorer 10

- HTML5
- CSS3

How the web works...

WEB APPLICATIONS

Static pages

- Request is sent to load a page
 - HTML is returned
 - HTTP GET Request
 - Images are loaded
 - HTTP GET Request
- HTML is rendered
 - Includes clickable links to other pages

Browser



IIS

AppHtml

Page.html



GET

`http://www.newwebsite.com/Page.html`

`<html>`

`<head>`

`<title>My Page</title>`

`...`

`</html>`

Classic Web Applications

- Pages become dynamic
 - Server-side scripts
 - Generate HTML
 - Parameters are passed in URL's
 - HTTP GET Request

[customer.asp?customerid=13](#)

- Data is passed using form posts
 - HTTP POST Request

```
<form action="demo_form.asp" method="POST">
```

Web Application Server

- ASP <-> DataFlex

- Implement Web Business Processes
- Call published functions from ASP

```
<%=oCustomer.call("Get_CustomerDetails",  
    Request("customerid") %>
```

- GenerateHTML

Browser

IIS

AppHtml

Customer.asp

ASP

WebApp.exe

GET

http://localhost/Customer.asp?customer=12



WebApp.exe

```
<table>
```

```
<tr>
```

```
<td>Customer Name:</td>
```

```
<td>Data Access Europe</td>
```

```
<%=oCustomer.call("Get_CustomerDetails",  
Request("customerid")) %>
```

AJAX

- JavaScript makes pages interactive
 - Calls load extra data from server
 - HTTP POST / GET
 - Displayed page is manipulated
 - Document Object Model

AJAX Library

- JavaScript widgets added to HTML
 - Calls DataFlex Web Service
 - HTTP POST
 - Find / Save / Delete Records
 - Emulated DDO's on the client
 - Still uses page reloads
 - Still uses server-side scripts

Browser

POST

http://localhost/WebService.wso

AppHtml

Customer.asp

<soap>

<soap>

<field>customer.name</field>

<value>Data Access Europe</value>

..

WebApp.exe



WebApp.exe



Web 2.0

- More and more client processing
 - Application in a page
 - No more page reload
 - Complete Widget Libraries
 - Client-side storage
 - AJAX, JSON, HTML5, CSS3, ...

Web Application Framework

- Application written in Visual DataFlex
 - JavaScript Engine
 - Widget implementation
 - Calls DataFlex Web Service
 - HTTP POST
 - JSON
 - Logic in DataFlex
 - No more ASP

Browser

POST

```
{ http://localhost/WebService.wso/CallAction/JSON
  "aSyncProps": [{
    "s0" : "oCustomer.oCustomerName"
    "aProps": [{
      "sP": "sValue",
      "sV": "Data Access Europe"
    }
  ]
}
```

<html> ...

<head>

<script src="DfEngine/df-include.is"></script>

</html>

o.exe

WebApp.exe

HTTP

- Stateless
 - No connection!
- Calls from the client to the Server
 - GET
 - HTML / CSS / Images
 - POST
 - JSON / XML
 - Web Service calls

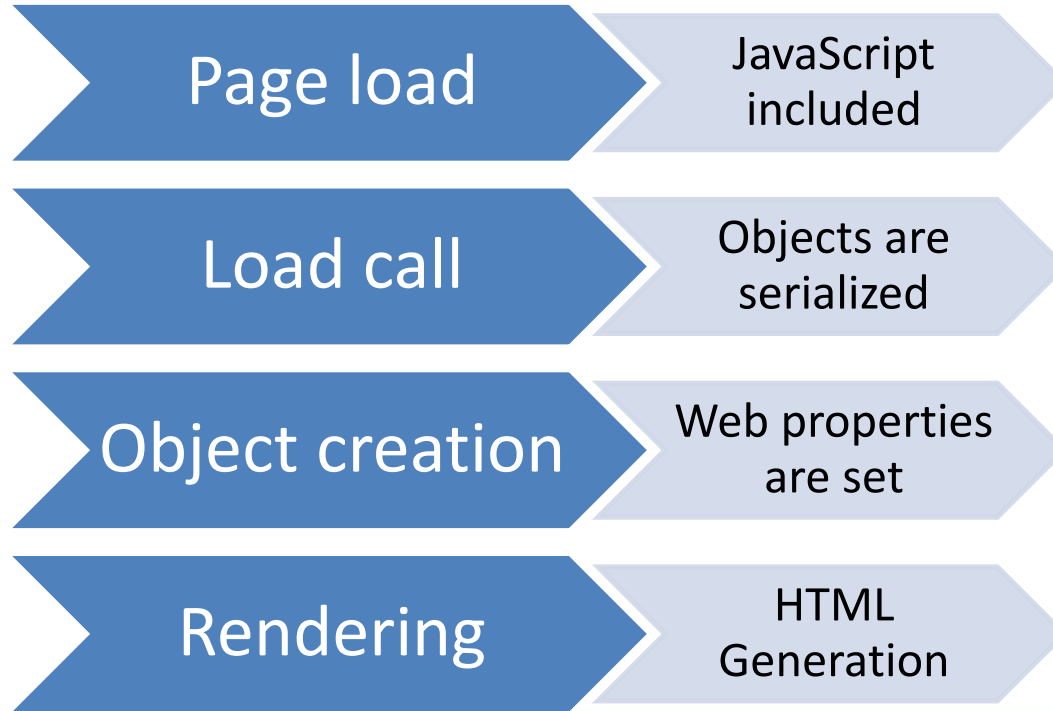
Main concepts of the web framework...

WEB FRAMEWORK CONCEPTS

JavaScript Engine

- Set of classes
 - Classes match DataFlex classes
- Objects are created for web objects
 - Object created for each view, form, menu item, button
- Web properties are implemented
- HTML is generated to render

Initialization



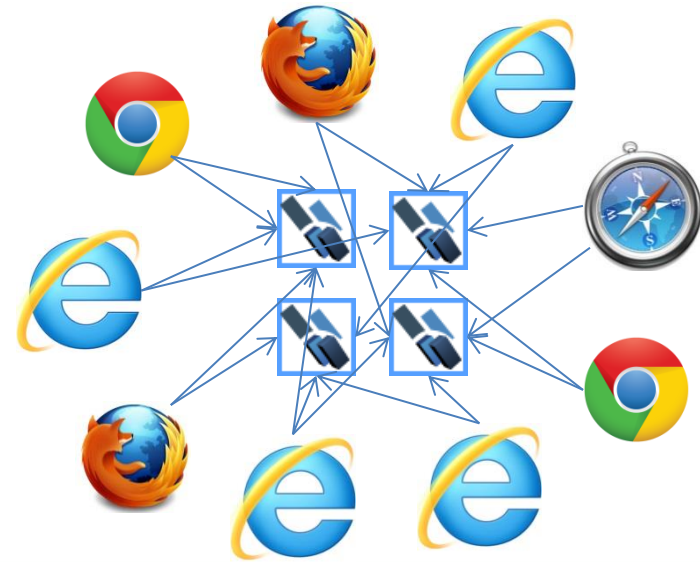
Object serialization

- Object definitions are sent to the client
 - Object name
 - Web properties
 - Class name to be used
 - Nested objects
- Recursive structure
- Implemented in cWebObject

```
{  
  "sName": "oCustomerName",  
  "sType": "df.WebForm",  
  "aProps": [{  
    "sN": "pbEnabled",  
    "sV": "1"  
  }],  
  "a0bjs": []  
}
```


Persistency

- No persistency on the server
 - Process pooling
- Calls should be self-explanatory



Applications State

- Maintained on the client
 - Web property values
 - List of changed properties
 - DDO Snapshot
 - List of current rowid's per view
- Sent with each call

Events

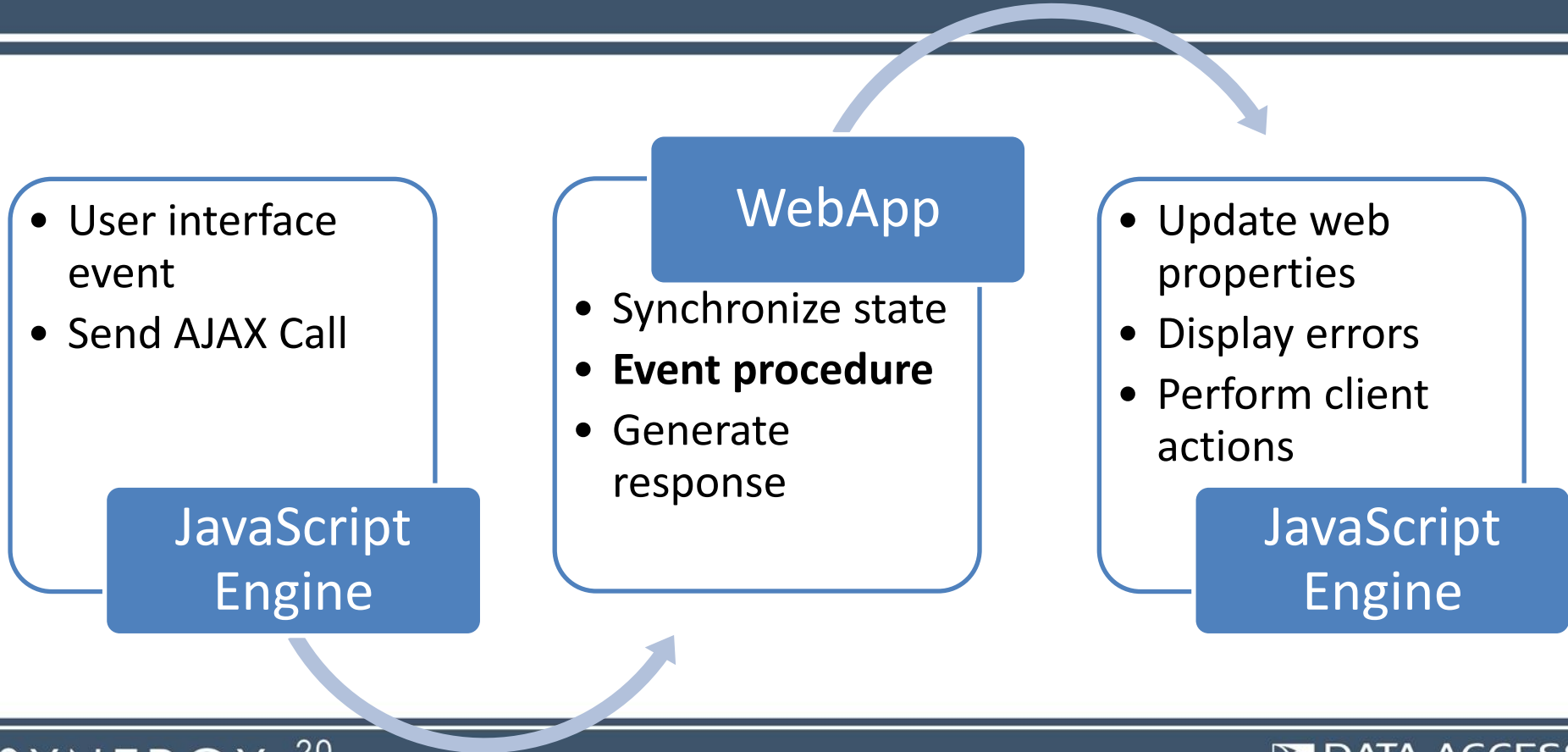
- Sent to the server
 - Current application state
 - List of changed web properties
 - List of rowid's
 - Server action(s)
- Processed on the server
- Returns
 - Changes on application state
 - Client actions



The screenshot shows a web browser's developer console with a POST request to `http://localhost/WebOrder_17_`. The response is a JSON object with the following structure:

```
{
  "Header": {
    "aDDODefs": [
      {
        "sView": "oOrderDtl_DD",
        "sRowID": "050000",
        "sName": "oCustomer"
      }
    ],
    "aSyncProps": [
      {
        "sO": "oOrder.",
        "sN": "pbChanged",
        "sV": "0"
      },
      {
        "sN": "psValue",
        "sV": "200"
      },
      {
        "sN": "pbEnabled",
        "sV": "0"
      }
    ],
    "sO": "oOrder.oWebM"
  }
}
```

Server call



John Tuohy will continue from a server-side perspective...

THE DATAFLEX PERSPECTIVE