





Schuster Stefan

Advanced JavaScript in FAT Clients



Don't panic!

No JSF

No Java

Topics

- └• **What/Why?**
- └• **Examples**
- └• **Dojo**
- └• **How-to**
 - **Start**
 - **Structure & Organization**
 - **Implementation**

What/Why?

What/Why?

- **What are we talking about?**
 - JavaScript - Pure JavaScript
- **Why do we talk about JavaScript?**
 - Initially: Ajax
 - In the meantime: JS indispensable

What/Why?

- └ **Areas of application:**
 - **Validation**
 - **UI enhancements (show/hide)**
 - **AJAX**
 - **RIA/SPA**
 - **Rich Internet Application**
 - **Single Page Application**

What/Why?

– What is an RIA/SPA?

- *Rich Internet applications (RIA) are web applications that have the features and functionality of traditional desktop applications. RIAs typically transfer the processing necessary for the user interface to the web client...*

http://en.wikipedia.org/wiki/Rich_Internet_application - 15.02.2008

What/Why?

- **Advantages: Usability**
 - **Direct feedback**
 - **More responsive**
 - **Desktop-like widgets and usage possible**

What/Why?

- Advantages: Application types
 - Some applications impossible within page-based architecture
 - WYSIWYG
 - Rich text formatting
 - Interactive graphics
 - Collaborative applications

What/Why?

- **Advantages: Server load**
 - Server only has to provide the data
 - Rendering gets done on the client
 - Fewer requests to handle

What/Why?

– Advantages: Offline

- Server only needed for data
- Browser will have an integrated database:
 - Google Gears
 - HTML 5
- Only data source has to be changed ...
 - ... in disregard of synchronisation

Example: Google Docs

The screenshot shows the Google Docs interface for a document titled "About RIA". The document content is a Wikipedia-style entry for "Rich Internet application". The interface includes a top navigation bar with the Google Docs logo, user information (st.schuster@gmail.com), and links to Docs Home, Help, and Sign out. Below the navigation bar are buttons for Save, Save & close, and Discard changes. A menu bar contains File, Edit, Insert, Revisions, and Edit HTML. A toolbar with various editing icons is visible. The document content includes a title "Rich Internet application", a subtitle "From Wikipedia, the free encyclopedia", and a paragraph of text with several hyperlinks. A "Check spelling" button is located at the bottom right of the document area.

Google Docs BETA

st.schuster@gmail.com | [Docs Home](#) | [Help](#) | [Sign out](#)

About RIA
saved on February 15, 2008 5:57 PM by St

File Edit Insert Revisions Edit HTML Preview Print Email Share Publish

Rich Internet application

From Wikipedia, the free encyclopedia

Jump to: [navigation](#), [search](#)
For the geographical term, see [ria](#).

Rich Internet applications (RIA) are [web applications](#) that have the features and functionality of traditional [desktop applications](#). RIAs typically transfer the processing necessary for the [user interface](#) to the [web client](#) but keep the bulk of the data (i.e., maintaining the [state](#) of the program, the data etc) back on the [application server](#).

RIAs typically:

- run in a [web browser](#), or do not require [software installation](#)
- run locally in a secure environment called a [sandbox](#)

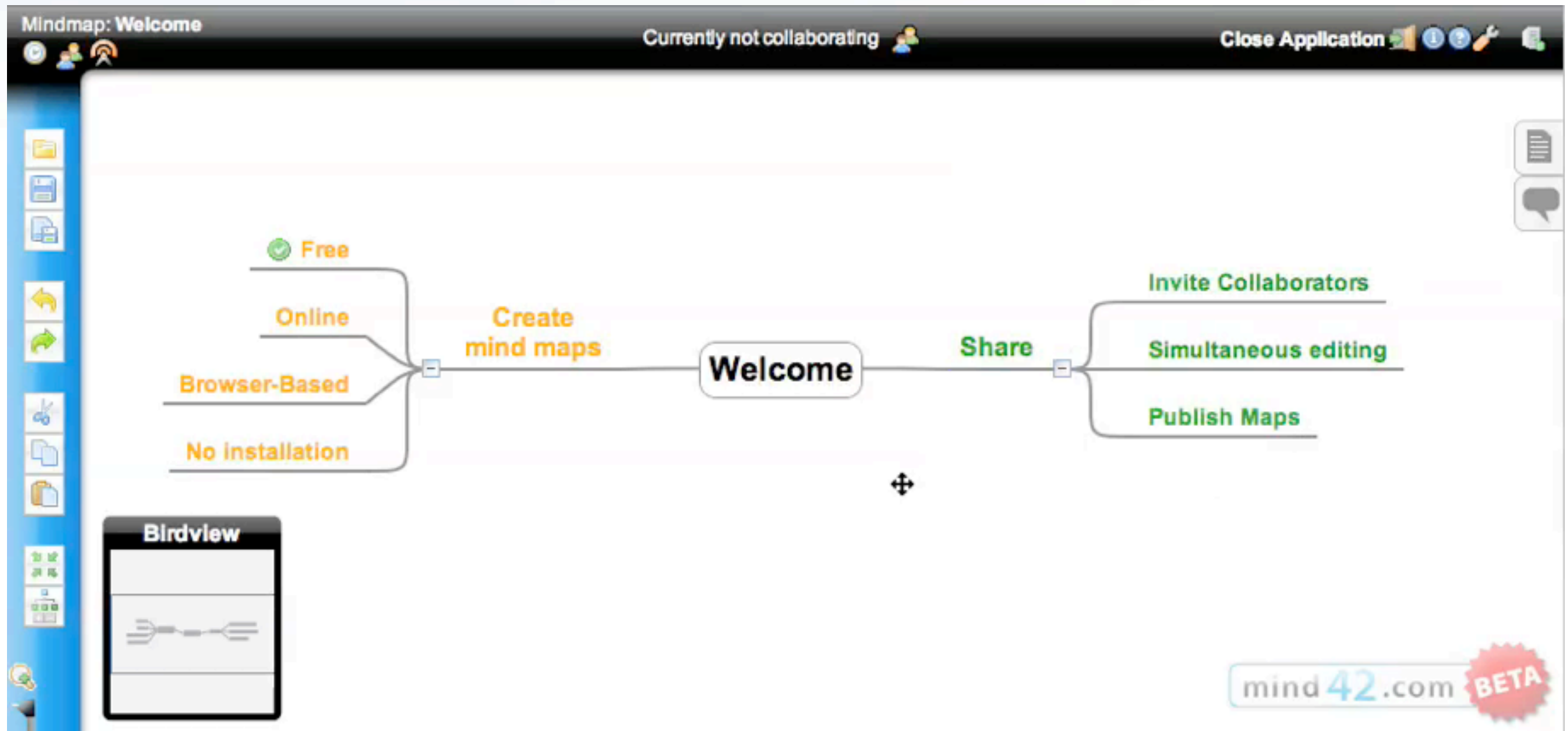
History

The term "Rich Internet Application" was introduced in a [white paper](#) of [March 2002](#) by [Macromedia](#), though the concept had existed for a number of years earlier under names such as:

- [Remote Scripting](#), by [Microsoft](#), circa [1998](#)

Check spelling

Example: Mind42



What/Why?

— Counter-arguments

- Security
- Complexity
 - Know-How
 - Additional work (bad tools)
- Type of applications
 - SPA for some simple forms? NO!

Dojo

JS Frameworks - General

└• Abstraction areas

- Cross Browser Help
 - Event handling
 - JS API (Arrays, window object, ...)
- DOM manipulation
- Widgets
- Effects
- ...

Dojo

– Many Libraries

- **Prototype/Scriptaculous**
- **jQuery**
- **YUI**
- **Ext JS**
- **MooTools**
- **Qooxdoo**
- ...

Dojo

- Why I prefer Dojo:
 - Module based
 - `dojo.require`
 - Build System

Dojo

—• Module based

- Small & fast set of core features
- Separate modules for every use-case

| | | | |
|-----------|----------|------------|---------|
| Events | Ajax | DOM | Effects |
| String | Date | Colour | Math |
| I18N | DnD | Widgets | Offline |
| Crypto | Charting | GFX | Comet |
| Templates | SQL | Validation | CSS |

Dojo

- **dojo.require:**
 - **Namespacing**
 - **dojo.date.locale**
 - **AJAX based on demand loading**
 - **dojo/date/locale.js**
 - **Ideal for development**

Dojo

└ Build System:

- Based on dojo.require
 - recursive resolution of requires
 - creation of single JS file of resolved requires
 - Minification / Obfuscation of code possible
- Single obfuscated file means less HTTP requests and smaller file size

Dojo

- **Additional goodies:**
 - **dojo.declare** for OO object definition
 - **dojo.connect**
 - Normalized DOM connection
 - Connections to every other method

How-to

How-to: Start

- If the client should work with the data, we need a:

Data model

How-to: Start

– JavaScript models basically consist of:

- Strings
- Numbers
- Booleans
- Arrays
- Objects

How-to: Start

- No good Dojo-specialized tooling available
 - Model classes hand-coded
 - Getters/Setter hand-coded
 - But generators could be written pretty easily

How-to: Implementation

Example: `irian.myApp.model.User`

```
dojo.provide("irian.myApp.model.User");

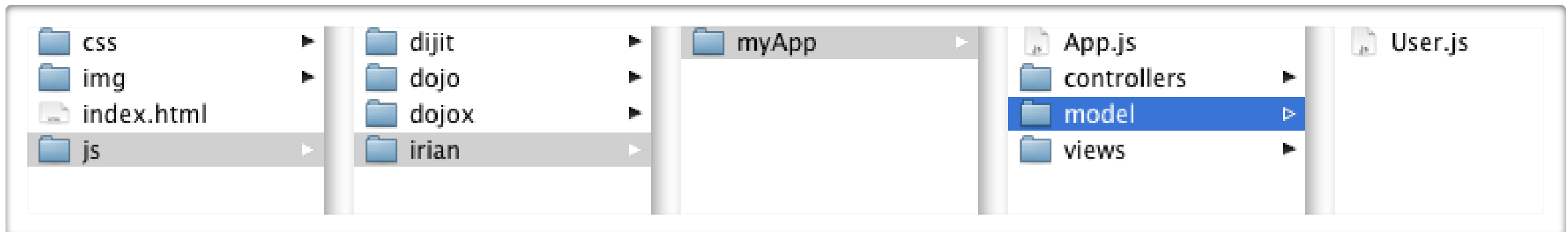
dojo.declare("irian.myApp.model.User", null, {
  constructor: function(id, username) {
    this._id = id || 0;
    this._username = username || "";
  },
  getId: function() {
    return this._id;
  },
  setId: function(id) {
    this._id = id;
  },
  ...
});
```

How-to: Start

- Based on the model you could implement:
 - Needed business logic in JavaScript
 - Communication
 - Format e.g. JSON
 - Method-based
 - Data-based

How-to: Structure & Organization

- Based on dojo.require
 - One file per “class”
 - Folders for namespaces



How-to: Structure & Organization

- **Model**
 - Business Logic
- **View**
 - DOM manipulation
 - Templating
 - Widget initialization
- **Controller**
 - Event handling

How-to: Structure & Organization

└ Controllers

- Inits the view
- Implements event handlers
- Initiates all the real work

How-to: Implementation

Controller: Example

```
dojo.provide("irian.myApp.controllers.Login");  
dojo.require("irian.myApp.views.Login");
```

```
dojo.declare("irian.myApp.controllers.Login", null,  
{  
    constructor: function() {  
        this.view = new irian.myApp.views.Login(this);  
    },  
    show: function(div) {  
        this.view.show(div);  
    },  
    onLogin: function() {  
        //Do something  
    }  
});
```

How-to: Structure & Organization

– Views

- Know their controller (for event setup)
- Finally show the mask
 - innerHTML
 - templating
 - DOM manipulation

How-to: Structure & Organization

– Views:

- **innerHTML faster than DOM access**
 - `http://www.quirksmode.org/dom/innerHTML.html`
- **JS Templating to separate presentation from JS**
 - **dojox.dtl (Django Template Language)**
 - **Trimpath JavaScript Templates**
 - `http://code.google.com/p/trimpath/wiki/JavaScriptTemplates`

How-to: Implementation

– View: Example

```
dojo.provide("irian.myApp.views.Login");

dojo.declare("irian.myApp.views.Login", null, {
  constructor: function(controller) {
    this.controller = controller;
    this.div = null;
  },
  show: function(div) {
    this.div = div;
    //Fill the div
    dojo.connect(dojo.byId("loginButton"), "onclick",
      this.controller, "onLogin");
  }
});
```

Example

- A simple working example:
 - Pure JavaScript Proof Of Concept Mini Blog

[Add article](#)

Test Article

Huhu!

Posted 25. Februar 2008 10:48:33 GMT+01:00

[1 comments](#)

Example

– JS Blog

- Simplified structure
 - Model
 - View-Controller
- No templating (inner-html)
- No server
- No persistence

Example

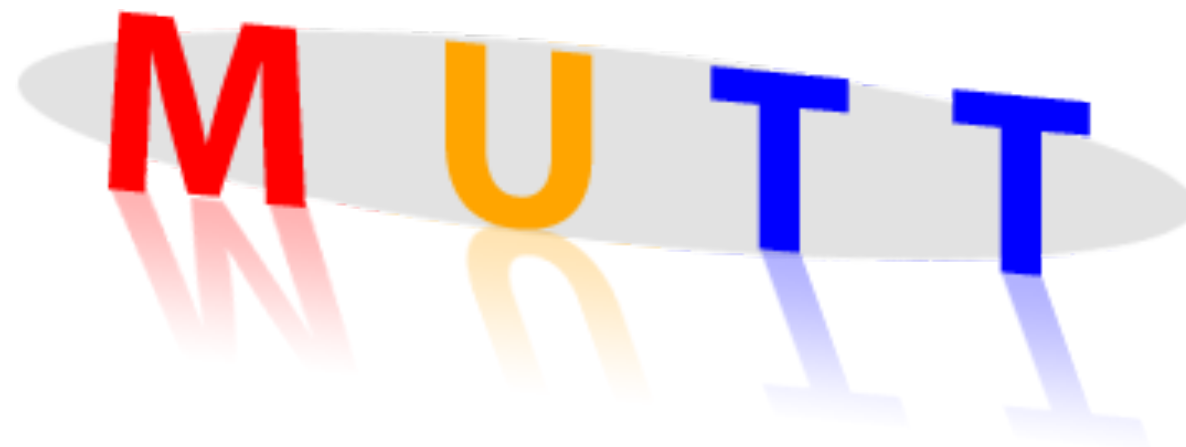
DEMO

Download:

[http://www.sschuster.net/presentations/
barcamp/200709/bcBlog.zip](http://www.sschuster.net/presentations/barcamp/200709/bcBlog.zip)

Example

- More advanced example application with Offline support



<http://www.sschuster.net/mutt>

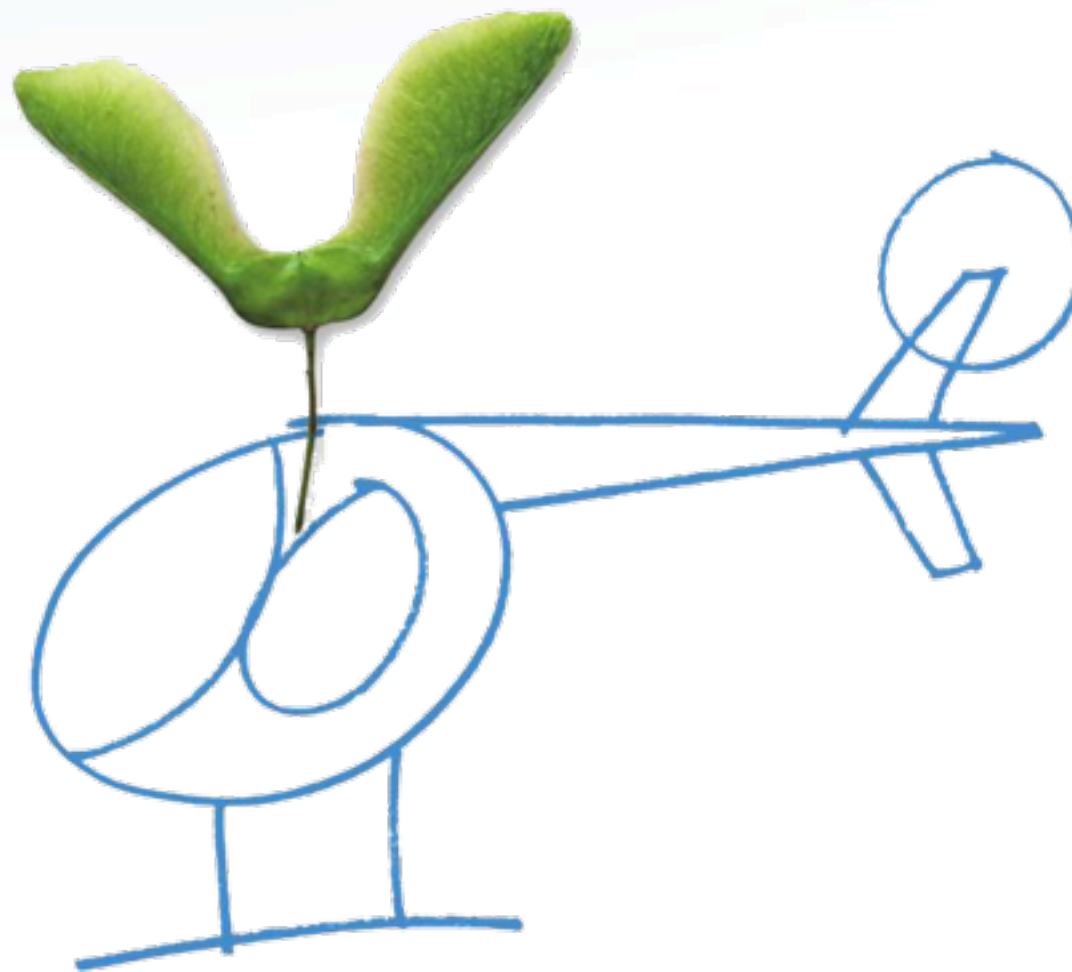
Conclusion

- JavaScript is a full programming language
 - design patterns applicable
 - business logic realizable
- “Endless” possibilities for client programming
- Examples + hands on code could give and idea

Ressources

- <http://javascript.crockford.com>
- <http://developer.yahoo.com/yui/theater/>

Thank you for your attention



There is always an idea in the beginning.