Web - New Generation

Ideas and specifications

How the web can be better?

Author:
Ioannis Charalampidis

Version:
Very first Draft

July 29, 2012

1 Introduction

I guess the first question someone has after looking the title is... why? Why should we create again something that is already there? We have the new HTML5 and the awsome CSS3! Shouldnt́ be thankful? In my opinion... no.

HTML is a beautiful language, very intuitive and very flexible. I love it! Also CSS is an amazing technology that fits with HTML perfectly! What about Javascript? I think it has a lots of bad issues, but still, with HTML5 we have an amazing new world with hardware-accelerated 3D graphics and integration with the native platform. Shouldn't that be enough? In my opinion, again... no.

In my opinion HTML lost it's purpose and started to become something it was not designed to be: A cross platform *application* programming language. It is a lovely idea, I am completely in favor, but I think HTML is not the correct language for that. Take for example the following points:

- *Most of the web application developers want to obfuscate/encrypt their commercial sources.* However, the way the web is structured right now does not provide such option. By default if someone can see a webpage, all of it's resources are completely exposed. Should't that be thought of?

- *There are hundreds of different frameworks for client or server-side web development.* This tells us something. HTML is a *BAD* language of developing what we need right now! The existence of those different frameworks tells us that the language itself is not easy to use any more. The biggest example are the javascript libraries (MooTools, jQuery, etc.). They all try to simplify the complex (and some times browser-dependent) interface to some very basic and widely used functions, ranging from element selection till animation. Shouldn't be the web development language a lot simpler to use?

- *Everyone is trying to minimize the size of their web projects.* They are compressing the scripts, compressing the HTML, compressing the CSS, optimizing the quality of the images and so on... However the whole website is still transmitted as a plaintext (even if it's compressed). Moreover, websites that are sharing the same server-side framework are transferring the same resources again and again under different domain names! Those resources might be the theme of the website (that many other websites might also be using) or javascript libraries that support the UI. Why not share those repeating resources?

- *The logic of the web site is transferred as a source code.* That might not be bad, but think how faster you could download and run a script if it was in an application-independent byte-code?

- *Not everyone is a designer.* Yes, I have seen thousands of different websites. Few of them picked a theme and stuck with it, fewer of them are designed from scratch with amazingly good results, but most of them are either badly designed or started well but during their maintenance something went terribly bad and the styling went off. Wouldn't it be the life a lot easier if you let the designers do the job for you and you are only responsible of filling the placeholders with your data?

- *Not all the users like your design.* Period. There are websites that respect all the guidelines of a proper web design, but there are also websites with a rainbow-colored, 20px comic sans ms text all over them. And -who knows- our current standards of a 'good looking' website might change in a couple of years. Wouldn't it be awesome if the website re-styles itself to fit the current needs?

- *There are no HTML elements with the required appealing visual appearance.* Even though with CSS3 you can solve most of your problems, there are many cases where you have to nest elements in a weird order, perform some mambo-jumbos and wish the magic trick works in every browser. Wouldn't it be perfect if you could create your own native element in order to preserve the semantics of that part of the website?

2 Solution

All the questions in the previous sections are not really rhetorical. They all sum-up to the following bullets:

- We need a platform-independent, byte-code language.

- We need to separate the *View* from the *Data*

The first one is already there. There already a couple of platform-independent Virtual Machines, such as *Java*, *.NET* or *Parrot*. Java - especially - exists already in every Operating System for both desktop and mobile platforms.

The second one is also simple to do:

- Your *website* should be only your data and nothing more.

- The view is a fully customizable application that is fetched from somewhere and renders your data.

3 Technical details

The website is defined as a *YAML* document transfered over HTTP.

```
$view :  com.wavesoft.views.main
title : Welcome to my website!
navigation:
- title: Home
  url: home
- title: Link
  url: link
```