Single Page Applications

Diego Cardozo

github.com/diegocard/SPA-Presentation

Agenda

1. Motivation
2. Introduction
3. Examples
4. Architecture
5. Advantages and disadvantages
6. Tools
7. Demo

Motivation (1)

One of the main aspects that influence user experience is load time.

- We use many techniquest to make it quicker:
  - Minimize scripts and CSS
  - Combine images in a single sprite
  - Delay JavaScript execution
  - Handle static files separately (CDN)
  - Resource cache

Motivation (2)

However, even with cache the browser must:

- Re-parse and execute CSS and JavaScript code.
- Download and parse the whole page HTML.
  - Even when only a little part changed.
- Rebuild the DOM tree.
- Render the UI.
- The user sees the page being constructed as he waits.

Motivation (3)

To sum up, SPA is an answer to the following questions:

- ¿How can we achieve a more efficient behavior?
- ¿Can we only load what's new or necessary?
- ¿How can we improve user experience?

Introduction (1)

What is a SPA?

- It is a new approach to building web applications.
- The whole source code is either loaded initially or afterwards dynamically, without reloading the page.
- Navigation is resolved on the client side.
- Server calls are done asynchronously.
- UI is built on the client side.

Introduction (2)

What is NOT a SPA

- Join all webpages for the site and load them statically.
- Black or white, hybrid approaches do exist.
- A silver bullet: it might not be a good idea for some projects.

Introduction (3)

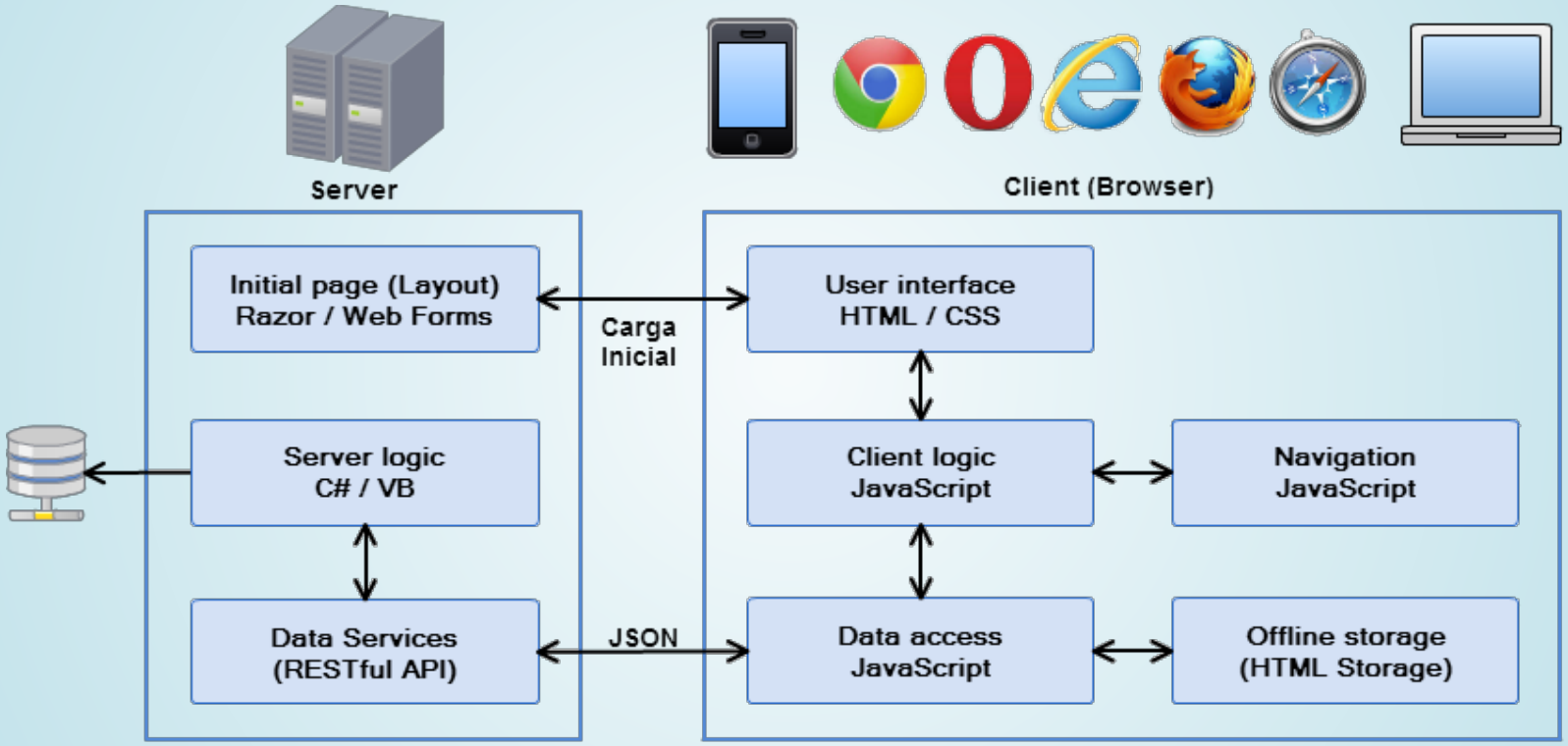
What can a SPA do?

- Show URL changes and navigate forward and backward.
- Manipulate the DOM on the client side.
- Wait for the view to load before showing it.
- Store previously loaded pages on the client.

Examples

- GMail
- **Stashy**
- **Nogginator**
- This presentation

Architecture



Advantages and disadvantages (1)

Advantages

- Faster and slicker UI.
- Easier maintenance.
- Better load distribution.
- The beginning of the development process is faster.
- UI is simply another client.
- Great for testing.
- Perfect approach to combine with mobile applications.

Advantages and disadvantages (2)

Disadvantages

- The initial load can be slow.
- SEO can become complex.
- Requires JavaScript to be enabled.
- Requires additional JavaScript knowledge.
- Breaks analytics, ads and widgets conventions.

Advantages and disadvantages (3)

Other aspects

- We move non-critic business logic to the client.
- Our code used to be90% C#/VB and 10% JS.
- Now it will be 50/50.

This isn't necessarily an advantage or disadvantage, but we will need a different set of tools to maintain good practices.

Tools (1)

Client side development

Frameworks

- Durandal
- Angular
- Ember

Routing

- Sammy
- History

Testing

- QUnit
- Jasmine

View management

- Knockout
- Backbone

Data management

- Breeze
- Amplify

DOM management

- jQuery
- Prototype
- Google closure

Tools (2)

Knockout.js

- Created by Steve Anderson
  - ASP.NET PM for Microsoft.
- Uses MVVM, a design pattern created by John Smith
  - Microsoft MVP for his work in WPF.
- It can be even used when we are not developing SPA.
- Included with MVC's default templates.

Tools (3)

Server side development

- En MVC, nuestros controladores pasan a ser ApiControllers para definir un API RESTful.
- Implementamos una sola vista (layout).
- No vamos a utilizar Razor para renderizar las vistas.
- Vamos a ver un template para Visual Studio que trae un proyecto pre-configurado para SPA (Hot Towel SPA).

Demo (1)

- Start from John Papa's Hot Towel SPA template
- Tour through at Durandal's main components
  - Analyzing each component in depth takes too long.
- We take a look at how the RESTful API is implemented
  - Exposes user data.
- We add a new functionality (sessions).
  - Tour through client code.
  - Mention the most important libraries.

Demo

github.com/diegocard/SPA-Demo

Resources

- learn.knockout.com
- singlepageappbook.com
- todomvc.com
- johnpapa.net
- slideshare.net/dcslides/spa-25806613

If you want to know more

Questions?