Hybrid App development with Phonegap

Marc Edem @Devfest_13

Contents

What is Phonegap

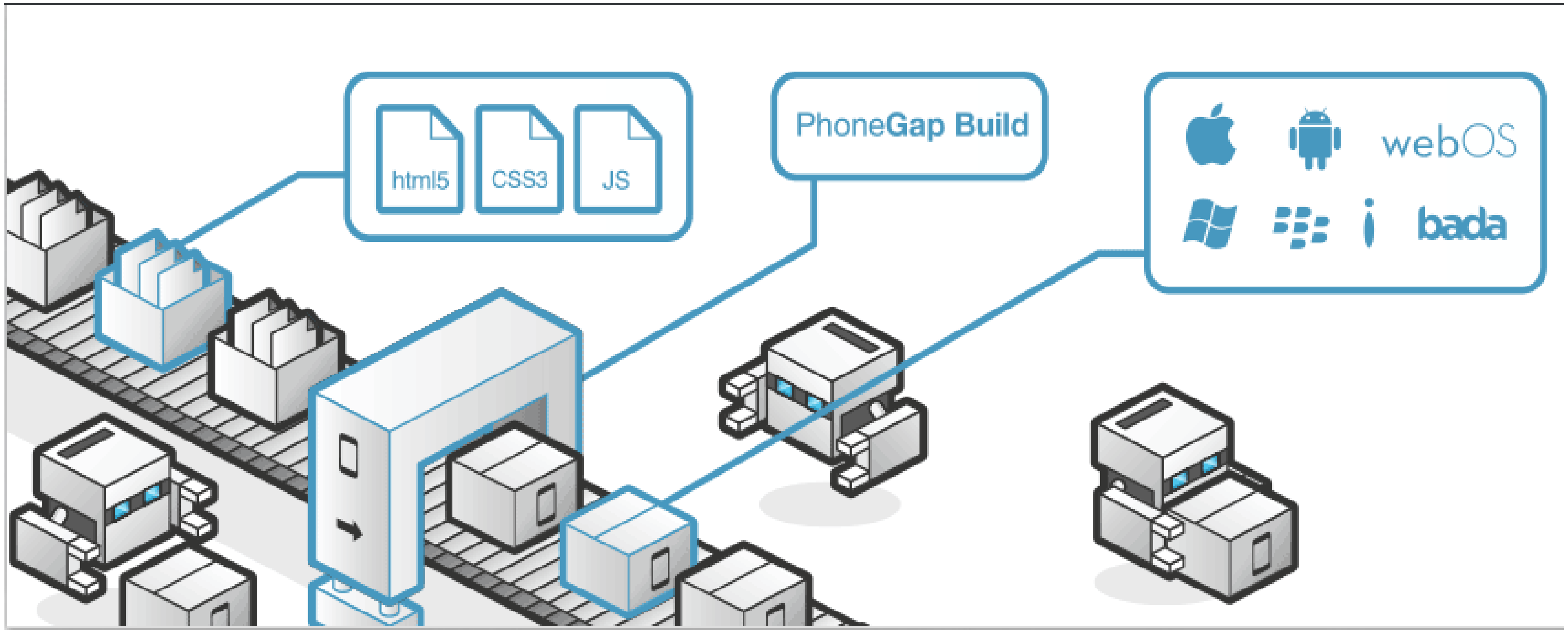
- Phonegap is :
  - Open-source mobile development framework
    - Developed by Nitobi Software and IBM
    - Bought by Adobe

- Enables building of applications for mobile devices using html5, Javascript and CSS.
- Is a wrapper for web app, and a bridge to different APIs or features of the devices such as camera, GPS, Contact etc ...

Why Phonegap?

- Complexity building apps that are crossplatform
- Advantage of using starndards-based web technologies to bridge web app and mobile devices
- More than 400000 developers are using phonegap.
- Cloud deployment support with Phonegap Build that allows to deploy without any SDKs, compilers and hardware.

PhoneGap Build



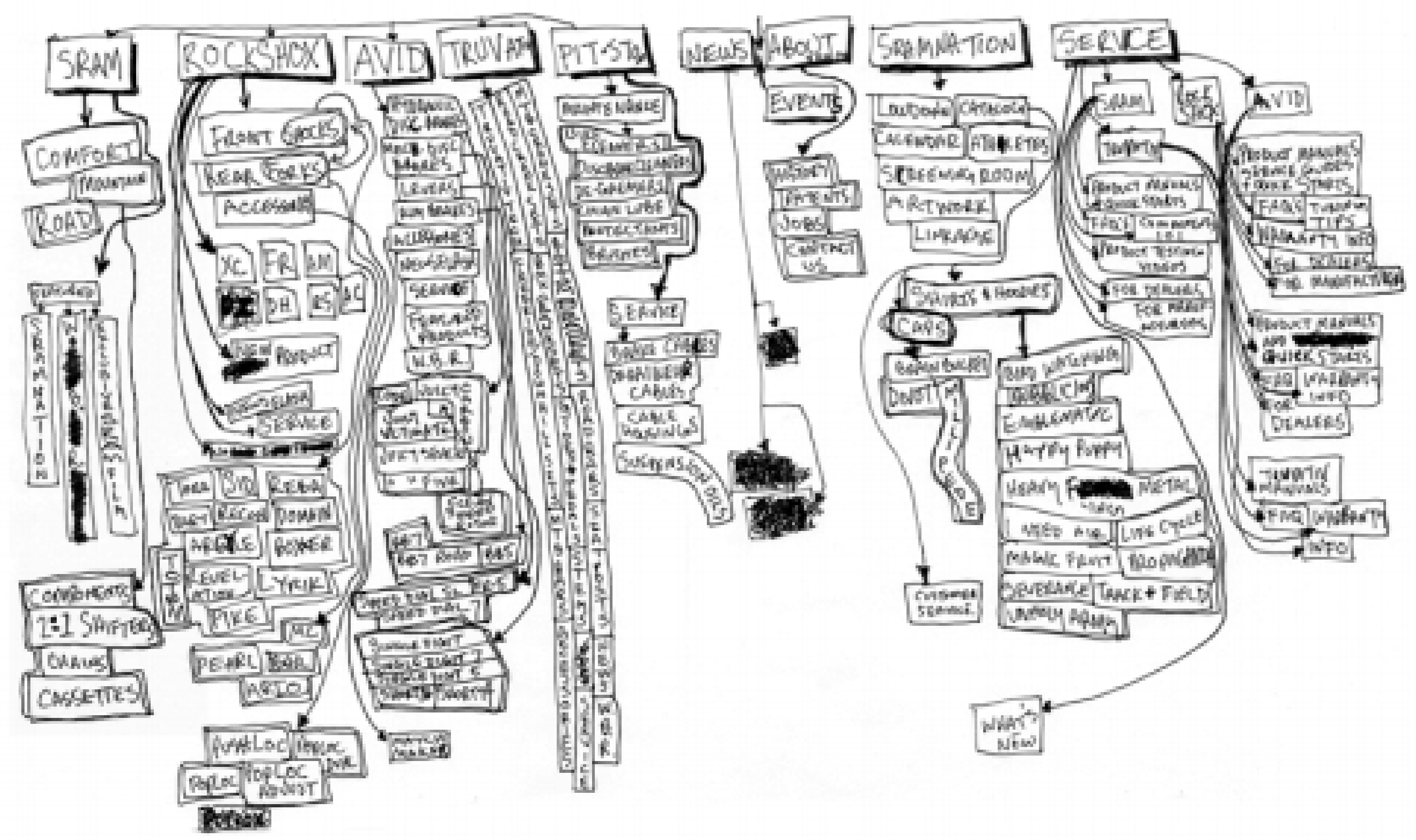
Phonegap and Apache Cordova

- October 2011: Phonegap donated to Apache Software Foundation (ASF) under the name Apache Cordova
- To maintain the development of Phonegap free and open source under the Apache License, Version 2.0.
- Apache Cordova: Platform for building native mobile application using HTML, CSS, Javascript.

Hybrid vs Native

	Hybrid	Native
Skills	HTML, JS, CSS	Obj C, Java, C/C++
Cross platform	Yes	No
Device APIs	Yes	Yes
Distribution	App Stores	App Stores
Updates	App Stores + instant	App Stores
Performance	Fast	Faster

Development Architecture



Development Architecture

- New School : One page

```
<html>
    <head>
        <title>Huge App</title>
        <script src="my-app.js"></script>
    </head>
    <body></body>
</html>
```

Development Architecture

	Multi-Page	Single Page
#Pages	Many	One
UI Generation Tier	Server	Client
Languages	Java, PHP, .Net, RoR, ...	Javascript
Offline Support	Limited	Yes
Page Transition	Browser	Developer
Performance	Laggy	Fast
App Assets Loaded	Many Time	One Time

Development Architecture : Templates

```
var html =
    '<div class="header">' +
          '<a href="#" class="button-left">List</a>' +'<h1>Employee</h1>’ +
   '</div>' +
'<div class="details">' +
      '<img src="pics/' + e.firstName + '_' + e.lastName + '.jpg"/>' + '<h1>' + e.firstName + ' ' + e.lastName + '</h1>' +
      '<h2>' + e.title + '</h2>' +
       '<ul class="list">' +
              '<li><a href="tel:' + e.officePhone + '">Call Office<br/>' + e.officePhone + '</a></li>' +
               '<li><a href="tel:' + e.cellPhone + '">Call Cell<br/>' + e.cellPhone + '</a></li>' +
              '<li><a href="sms:' + e.cellPhone + '">SMS<br/>' + e.cellPhone + '</a></li>' +
     '</ul>' +
'</div>';
```

Development Architecture : Templates

```
<div class="header">
   <a href="#" class="button-left">List</a>
   <h1>Employee</h1>
</div>
<div class="details">
   <img src="pics/{{firstName}}_{{lastName}}.jpg"/>
   <h1>{{firstName}} {{lastName}}</h1>
   <h2>{{title}}</h2>
   <ul class="list">
                  <li><a href="tel:{{officePhone}}">Call Office<br/>{{officePhone}}</a></li>
           <li><a href="tel:{{cellPhone}}">Call Cell<br/>{{cellPhone}}</a></li>
           <li><a href="sms:{{cellPhone}}">SMS<br/>{{cellPhone}}</a></li>
   </ul>
</div>
```

Development Architecture

- Use single page architecture
  - Benefits
    - Fast
    - Works offline
    - Control over experience
  - Caveats
    - More complex
    - Memory Management
    - Modular Strategy

- Templates
  - Benefits
    - Maintainable
    - Toolable
    - Separation of concerns
  - Examples
    - Mustache.js
    - Handelbars.js
    - Underscore.js

Development Architecture : MV*

- Providing Structure to the application using a MV* Architecture.
- Consider use of frameworks
  - Full stack
    - Sencha, Jquery mobile, Dojo Toolkit ...
  - Custom stack
    - Backbone.JS, AngularJS, Zepto.js, etc ...

Development Architecture

- Abstract Device Features

Interaction	Mouse	Touch
Notification	Javascript	Native
Storage	online	offline
Sensors	unavailable	available

Development Architecture : Performance

- Don't generate UI on the server
- Don't wait for data to display the UI
- Cache everything (data, selectors, precompiled templates, ...)
- Use Hardware acceleration
- Avoid click event's 300ms delay
- Use CSS sprite sheets
- Limit shadows and gradients
- Avoid reflows
- Do you need that framework?
- Test

Summary

- Use single page application
- Use templates
- Use MV* architecture
- Consider frameworks
- Abstract devices features
- Architect for performance

More architecture principles

- Abstract data access
- Keep application browser runnable
- Implement routing
- Hide HTMLish behavious

Questions ??

Thank you !!!