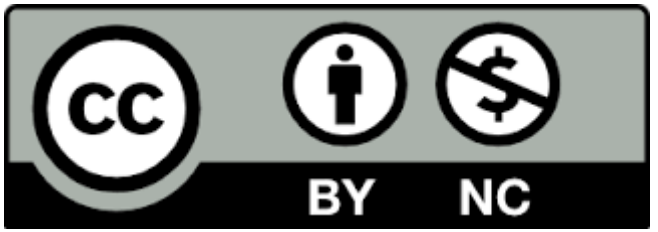
OFICINA SNCT

Alexandre Garcia Aguado
Mestre em Tecnologia e Inovação
Professor – IFSP – Capivari
Filho do Sr. Diogo e da Dona Maria
Marido da Gabriela

OFICINA SNCT



Partilhe!

Melhore!

Use!

Agenda

- Visão Macro sobre desenvolvimento mobile
- O que preciso saber pra começar?
- Preparação do ambiente!
- Nosso primeiro projeto!!!
- Vamos dificultar um pouco – Segundo Projeto
- Onde estudar mais?

Visão Macro sobre Desenvolvimento Mobile

Visão Macro sobre desenvolvimento Mobile

Visão Macro sobre PDM

Nativo

Web App

Híbrido

Existem três tipos de aplicativos Mobile ...

Visão Macro sobre PDM

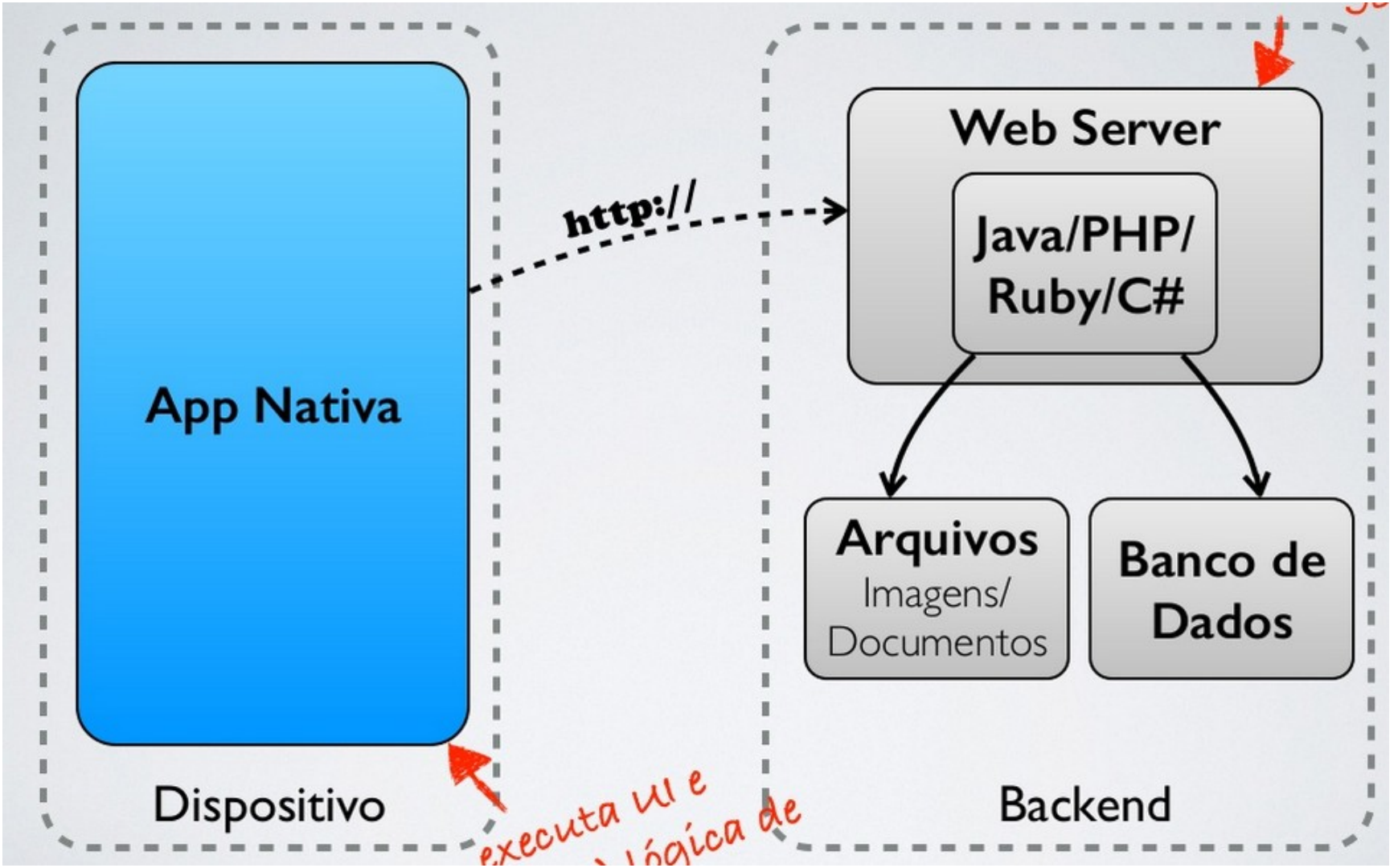
- Aplicações Nativas
  - São desenvolvidas diretamente na linguagem que o S.O do aparelho interpreta.




C++ Objective-C JAVA C#

Visão Macro sobre PDM

- Aplicações Nativas



Fonte: **www.loiane.com**

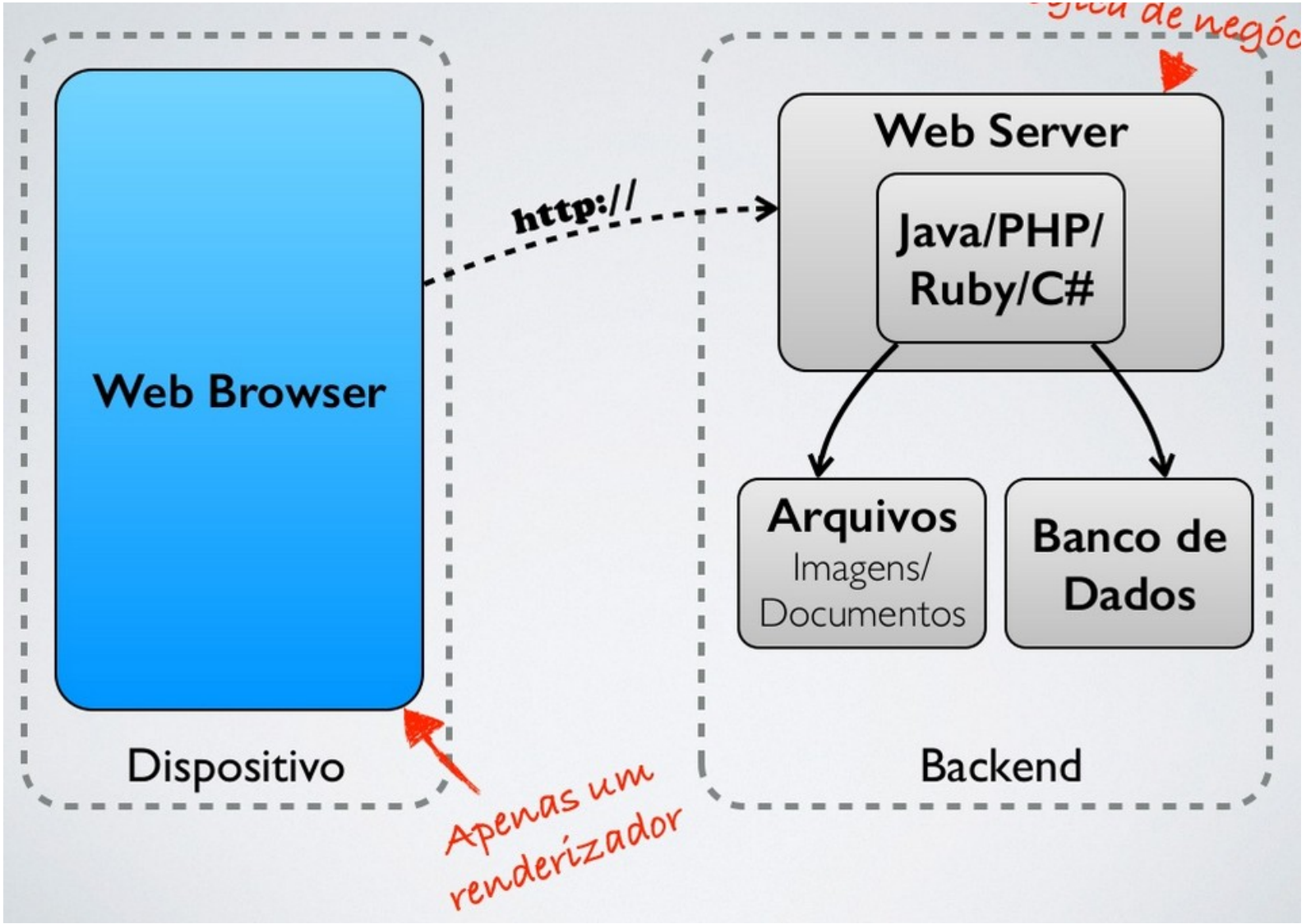
Visão Macro sobre PDM

- Web APP
  - São sites hospedados em servidores web, mas tem um visual muito adequado aos dispositivos móveis!

Rodam aqui!!!

Visão Macro sobre PDM

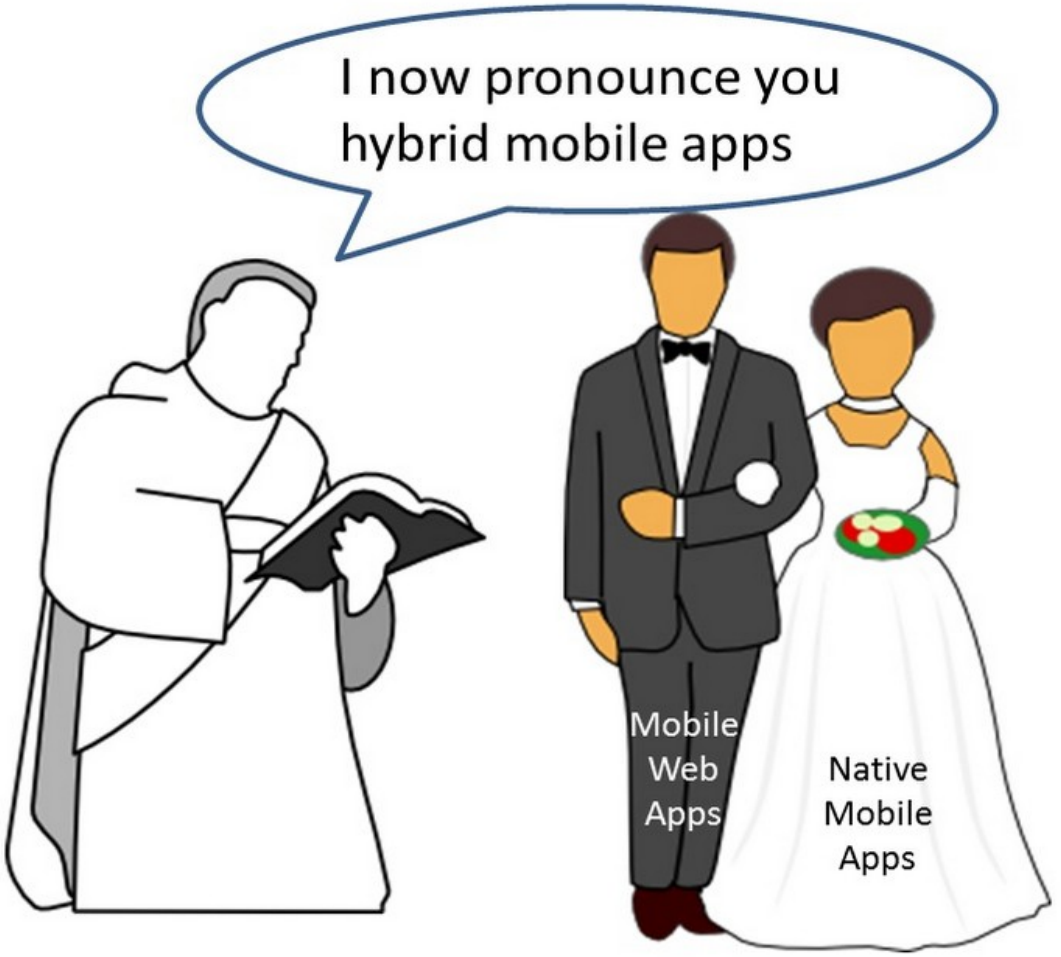
- Web APP



Fonte: **www.loiane.com**

Visão Macro sobre PDM

- Não dá pra juntar as duas coisas? SIMMM!!!



Fonte: **www.loiane.com**

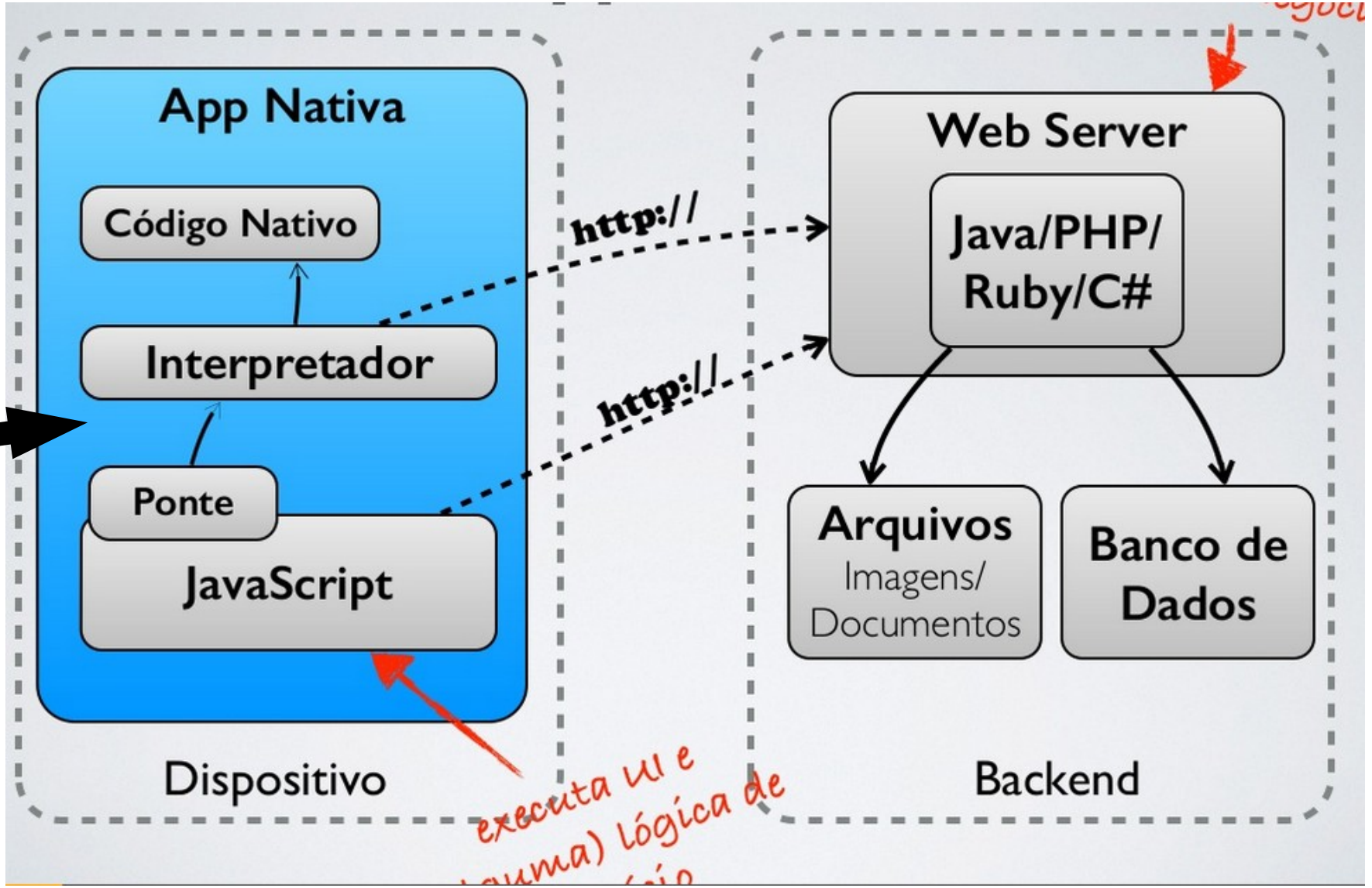
Visão Macro sobre PDM

- Não dá pra juntar as duas coisas? SIMMM!!!
  - Aplicativos Híbridos!
    - Os aplicativos híbridos são desenvolvidos utilizando HTML5 + CSS3 + JavaScript e um software interpretador, também conhecido como "**native bridge**" faz o acesso as operações nativas do celular!
    - Esse "native bridge" tem um nome: Cordova / Phonegap!

APACHE CORDOVA™

Visão Macro sobre PDM

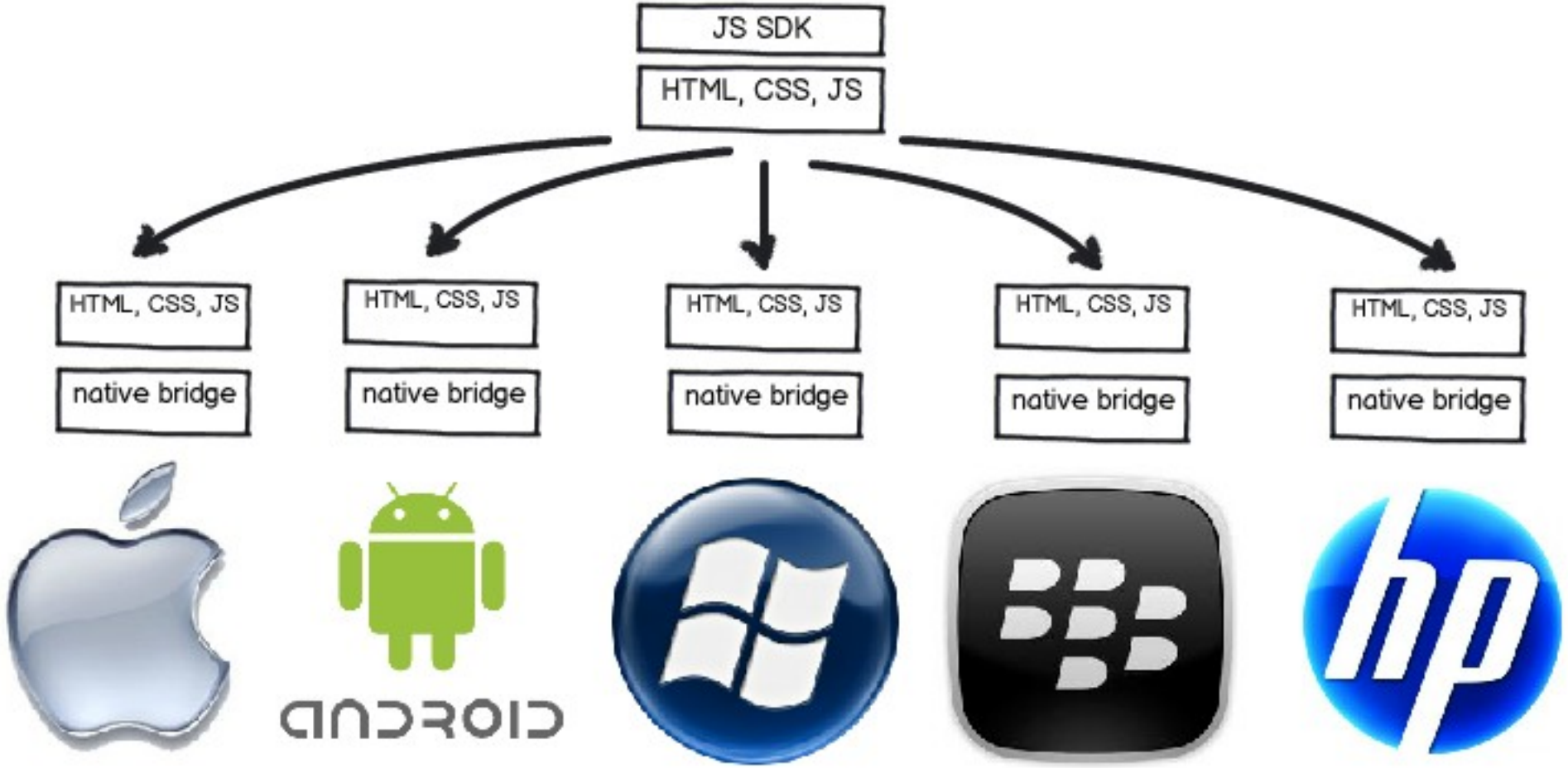
- Aplicativos Híbridos



Fonte: **www.loiane.com**

Visão Macro sobre PDM

- Aplicativos Híbridos



Visão Macro sobre PDM

Um breve comparativo ...

Visão Macro sobre PDM

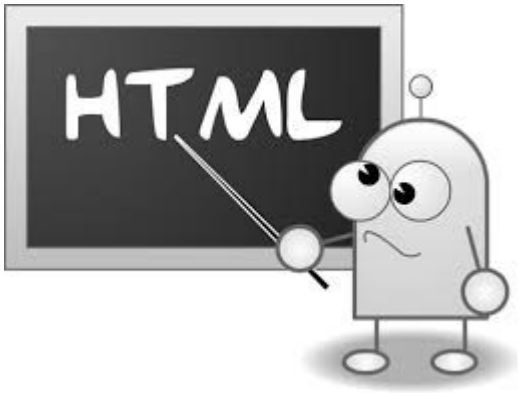
- Comparativo entre as três formas:

	Acesso Device	Velocidade	Tempo Desenvolvimento	App Store	Cross Platform
Nativo	Sim	Sim	Caro	Sim	Não
Web Apps	Parcial	?	Sussa	Não	Sim
Híbrido	Sim	?	Sussa*	Sim	Sim

Fonte: **www.loiane.com**

Vamos começar ….

… mas o que preciso saber para começar?



Preparação do Ambiente

Preparação do Ambiente

- O que vale para todos os S.O
  - O **Cordova** roda sobre o **nodejs**, ou seja, antes de instalar ele é necessário ter instalado o nodejs!
  - Após ter o nodejs para instalar o cordova basta um comando simples: **npm install -g cordova**
  - Para ser possível emular e passar para o celular os seus aplicativos você precisará do Android SDK (no caso do Android) ou do IOS SDK (no caso do IOS)!
  - Provavelmente você terá que configurar algumas variáveis de ambiente para conseguir trabalhar!

Preparação do Ambiente

- OBS: Não é minha praia, então recomendo alguns links que explicam MUITOOO melhor que eu essa parte:
  - **Opção 1:** https://www.youtube.com/watch?v=zSGySqUmPhY
  - **Opção 2:** http://fernandofreitasalves.com/guia-rapido-para-instalar-e-rodar-o-phonegap-no-windows/

Preparação do Ambiente

- **Passo 1**: Instalação do nodejs
  - Logue como root! `usuario@pc:~$ su`
  - Atualize seus repositórios: `usuario@pc:~$ apt-get update`
  - Vamos instalar o curl: `usuario@pc:~$ apt-get install curl`
  - Atualizamos: `usuario@pc:~$ curl -sL https://deb.nodesource.com/setup | bash -`
  - Instalamos o nodejs: `usuario@pc:~$ apt-get install nodejs`

Preparação do Ambiente

- **Passo 2**: Instalação do cordova
  - Para instalar o cordova:

```
usuario@pc:~$ npm -g install cordova
```

  - Vamos instalar também o ionic?

```
usuario@pc:~$ npm -g install ionic
```

Preparação do Ambiente

- **Passo 3**: Preparação do Android SDK
  - Faça o Download do SDK : **https://developer.android.com/studio/index.html**
  - Descompacte o arquivo dentro da pasta que preferir! Eu uso geralmente a pasta ***/opt*** !
  - Se seu sistema for 64 bits instale umas dependências:

```
usuario@pc:~$ apt-get install lib32z1 lib32ncurses5 lib32bz2-1.0 lib32stdc++6
```

Preparação do Ambiente

- **Passo 3**: Preparação do Android SDK
  - Existem algumas variáveis de ambiente que precisam ser criadas, então vamos editar um arquivo que irá fazer isso sempre quando o computador iniciar!
  - Saia do root:

```
usuario@pc:~$ exit
```

  - Abra o arquivo profile:

```
usuario@pc:~$ nano ~/.profile
```

  - Adicione essas 3 linhas nele:

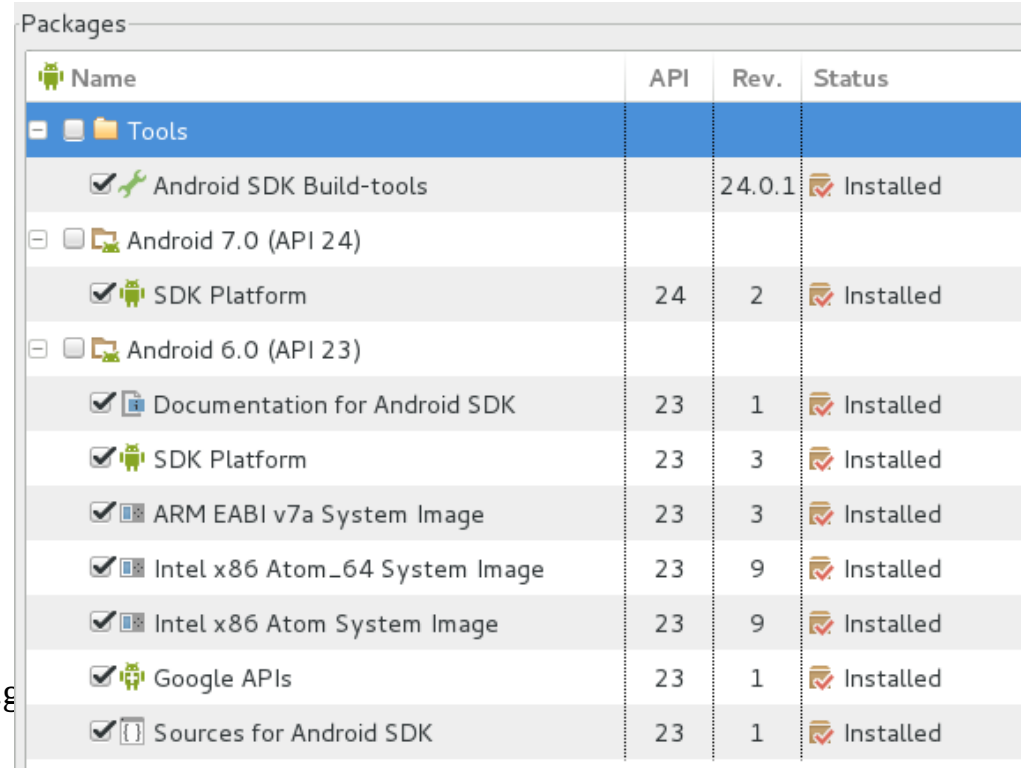
```
export ANDROID_HOME=/opt/android-sdk-linux
export PATH=${PATH}:$ANDROID_HOME/tools
export PATH=${PATH}:$ANDROID_HOME/platform-tools
```

  - Para salvar nesse editor
  - Aperte **Control X**, depois
  - **S** e dê enter!
  - Atualize os dados:

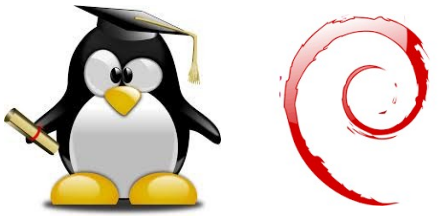
```
usuario@pc:~$ source ~/.profile
```

Preparação do Ambiente

- **Passo 3**: Preparação do Android SDK
  - Com as variáveis criadas, vamos baixar todas as bibliotecas necessárias para configurar nosso emulador.
  - Para abrir o configurador: `usuario@pc:~$ android`
  - Deixe selecionado para baixar os itens conforme figura e
  - depois clique em **Instalar**:



Preparação do Ambiente



- Se tudo deu certo, você chegara neste momento com seu ambiente preparado para programar com o Cordova!
- Se algum problema ocorreu, pesquise a solução! A comunidade tem produzido muito material, senão, me escreva! ale.garcia.aguado@gmail.com !
- AHHH! Você precisa ter o Java JDK instalado em seu computador!
- Para isso, pode seguir esse tutorial:
  - http://www.edivaldobrito.com.br/como-instalar-o-oracle-java-8-em-debian-via-repositorio/
- AHHH2 ! Eu costumo emular o que desenvolvo direto no meu celular! Para isso você precisa em seu aparelho habilitar o modo Desenvolvedor e permitir o DEBUG USB !

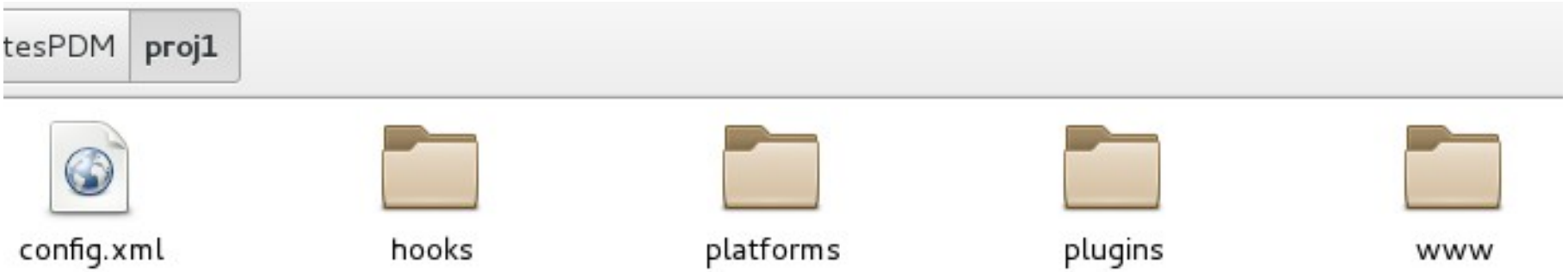
Nosso primeiro projeto

Nosso primeiro projeto

- Para criar o projeto e interagir com o Cordova vamos usar o Terminal!
- **Passo 1**: Crie o projeto

```
$ cordova create proj1 br.aguado.proj1 Proj1
```

Será criada uma pasta com o nome **proj1** e a seguinte estrutura!



- Nosso lugar de “trabalho” é a pasta **www** ! É nela que iremos criar nossa página Web que se transformará em nosso aplicativo mobile!!!
  - OBS: A página inicial de nosso aplicativo será o index.htmlda pasta www, ou seja, esse arquivo **precisa** existir com este nome!!

Nosso primeiro projeto

- **Passo 2:** Utilizando o editor html que preferir e seus conhecimentos de HTML, CSS e JavaScript, crie na pasta www “seu aplicativo”!
  - Eu particularmente uso o editor **geany !** Geralmente eu deleto todo conteúdo da pasta www e crio minha própria estrutura!
  - [Essa parte de criação web eu não vou detalhar porque imagino que você já conheça um básico de html, css e javascript!]
  - Após concluir seu trabalho na pasta www vamos seguir com o projeto!
- **Passo 3**: Vamos adicionar o Android (pode ser IOS ou outro S.O, mas estou usando Android como base aqui) como plataforma para nosso software! Esses comandos, após o create, precisam ser **executados a partir do diretório do projeto**!!!:

```
usuario@pc:~/proj1$  cordova platform add android
```

Nosso primeiro projeto

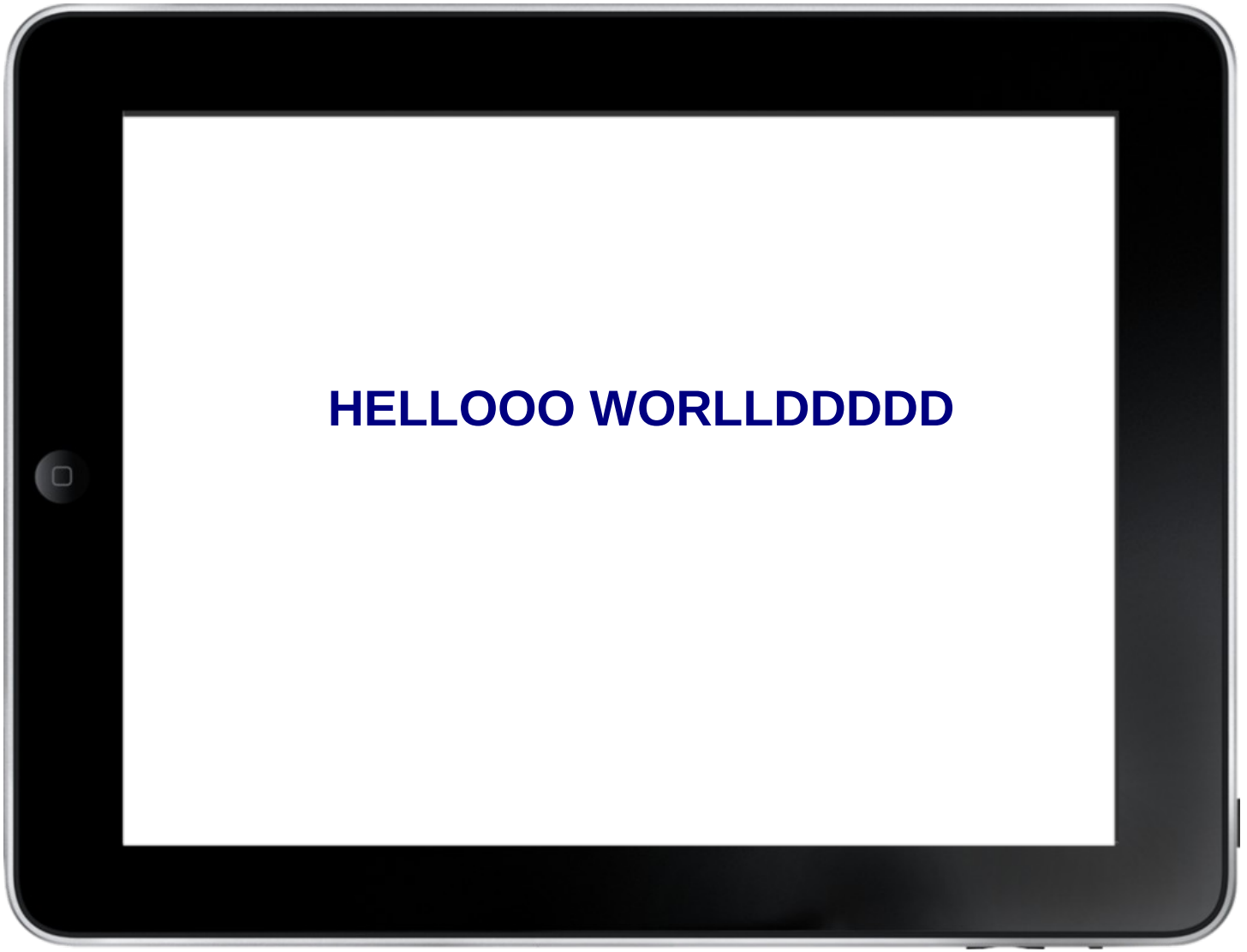
- **Passo 4:** Vamos agora construir nosso apk (é o nome do executável do celular … iremos “compilar” nosso aplicativo)

```
usuario@pc:~/proj1$  cordova build android
```

- ATENÇÃO! Aqui pode dar alguns erros relacionados a:
  - **Conexão de Internet:** Você precisa estar conectado!!!
  - **Android**: Tudo precisa estar bem configurado quanto ao Android SDK e as variáveis de ambiente!
  - **Java**: Você precisa ter o Java Instalado!!!!
- **Passo 5**: Com o celular ou tablet conectado no computador vamos colocar nosso projeto dentro dele e ver a mágica!!!

```
usuario@pc:~/proj1$  cordova run android
```

Nosso primeiro projeto



Mas a vida … essa sim é uma caixinha de surpresas ...

..mas a vida ...

- Nesse primeiro projeto fizemos bem pouco! Não utilizamos a principal função do CORDOVA que é servir de um **NATIVE BRIDGE**, ou seja, uma aplicação que possibilita acessarmos funções nativas do dispositivo móvel!!!
- O Cordova possibilita para nós uma série de PLUGINS que uma vez adicionados ao projeto nos permite o acesso a funções nativas!!!
- Para essa segunda parte de nossos trabalhos a principal fonte é essa:
  - http://cordova.apache.org/docs/en/latest/
- Nesse link teremos a documentação de quase todos os PLUGINS do Cordova com uma documentação RIQUÍSSIMA de exemplos para criarmos nossos aplicativos!!!
- Para demonstrar o uso dos PLUGINS vamos utilizar o **cordova-plugin-camera** que nos permitirá acessar a função de camera do dispositivo móvel!!!

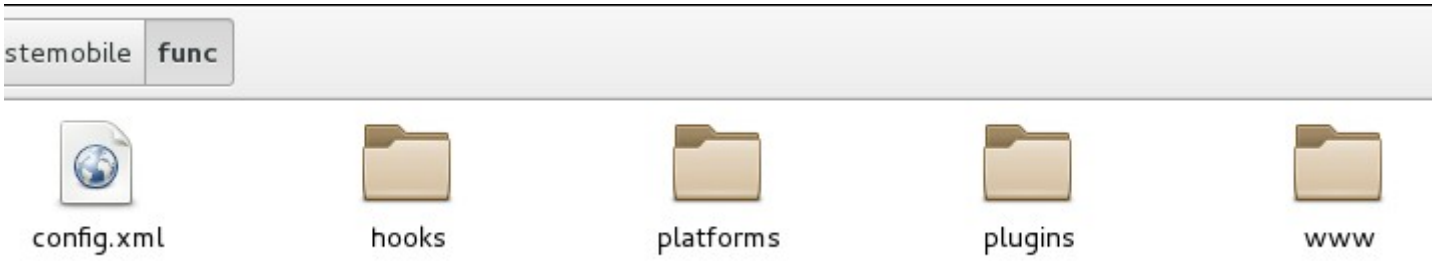
Segundo Projeto

- Como exemplo, vamos fazer juntos um aplicativo que chama a função “Tirar Foto” do celular e depois vamos pegar essa foto e colocar no nosso aplicativo como “FUNCIONÁRIO DO MÊS”

- **Passo 1**: Crie o projeto

```
$ cordova create func br.aguado.func Func
```

  Será criada uma pasta com o nome **func** e a seguinte estrutura!



- Acesse a pasta www e remova todo seu conteúdo! Vamos criar do zero nosso aplicativo!

Segundo Projeto

- **Passo 2**: Criando o aplicativo com css, html e javascript
  - Eu costumo organizar minha pasta **www** da seguinte forma:

  - Vamos agora programar!!!

Segundo Projeto

- Esse é o nosso index.html [Fique a vontade para mudar!]

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8">
5          <title>Funcionário do Mês</title>
6          <!-- Importando o cordova-->
7          <script type="text/javascript" charset="utf-8" src="cordova.js"></script>
8          <!-- Importando o js que é nosso principal arquivo!-->
9          <script type="text/javascript" charset="utf-8" src="js/camera.js"></script>
10         <!-- Importando um css que vamos construir..não vou me preocupar mto com estilo!-->
11         <link rel="stylesheet" href="css/estilo.css">
12     </head>
13     <body>
14         <!-- Ao clicar no botão chamará a função js capturarFoto()!-->
15         <button onclick="capturarFoto();"> Capturar imagem do funcionário do mês!!! </button>
16         <figure>
17             <!-- Essa tag de imagem irá receber a nova foto!-->
18             <img id="imagem" src="img/basica.png"/>
19             <figcaption>Funcionário do Mês</figcaption>
20         </figure>
21     </body>
22 </html>
```

Segundo Projeto

- Esse é o nosso estilo.css [Basicão ...]

```css
img{
    width: 90%;
    display: block;
    max-width: 200px;
    position: relative;
    left: 50%;
    margin-left: -45%;
}

figure{
    border: 10px double blue;
    max-width: 200px;

}

figure>figcaption{
    font-family: arial;
    font-size: 1.2em;
    text-align: center;
    color: white;
    background-color: blue;
}
```

Segundo Projeto

- Como é no JS que a mágica acontece, vamos por partes mostrando o código de nosso camera.js!
  - Geralmente adicionamos um *listener* que disparará uma função quando o dispositivo estiver pronto! Essa função pode ter qualquer nome e dentro dela podemos fazer qualquer coisa …

```
1   var pictureSource;   // guardará a origem da foto
2   var destinationType; // configura o tipo de destino
3
4   // Adicionamos um listener que será executado
5   // quando o dispositivo estiver pronto!
6   document.addEventListener("deviceready",onDeviceReady,false);
7
8   function onDeviceReady() {
9       pictureSource=navigator.camera.PictureSourceType;
10      destinationType=navigator.camera.DestinationType;
11  }
```

Segundo Projeto

- Como é no JS que a mágica acontece, vamos por partes!
  - No nosso caso, teremos um botão que chamará a função capturarFoto() !!! Essa função por sua vez é quem chamará o principal método do **plugin** da camera!!! Leia o comentário do código!

```
25    // Um botão chamará essa função!
26    function capturarFoto() {
27      // Chamamos aqui o método getPicture que é implementado pelo Plugin!
28      // Ao chamar um desses métodos dos Plugins precisamos passar 3 parametros
29      // básicos: Função de CallBack - será chamada de volta pelo plugin se tudo ser certo
30      //Função de erro: ... se algo der errado e um array com parametros de configuração,
31      //Que irão depender de qual plugin esta sendo usado
32      navigator.camera.getPicture(onPhotoDataSuccess, onFail, { quality: 50,
33        destinationType: destinationType.DATA_URL });
34    }
```

  - Dá uma olhadinha nesse link: http://cordova.apache.org/docs/en/latest/reference/cordova-plugin-camera/index.html

Segundo Projeto

- Como é no JS que a mágica acontece, vamos por partes!
  - Esse é o método de CallBack! Quem chama ele é a própria API (plugin) caso dê certo seu trabalho! Quando em nosso javaScript chamamos o método getPicture(), passamos o nome dessa função por parâmetro a que a API soubesse quem chamar se tudo desse certo!
  - É aqui que pegamos a imagem que é passada pela API e gravada em um lugar no dispositivo também pela API e colocamos na tag IMG que criamos em nosso HTML !

```
13      // Chama-se função de CallBack! É chamada de volta pelo
14      // plugin em caso de sucesso ao tirar a foto!
15      function onPhotoDataSuccess(imageData) {
16          //Jogamos em uma variável o elemento html que tem
17          //como id imagem!
18        var mypic = document.getElementById('imagem');
19          //Falamos que o src deste elemento passará a ser
20          //a imagem que foi tirada!
21        mypic.src = "" + imageData;
22      }
```

Segundo Projeto

- Como é no JS que a mágica acontece, vamos por partes!
  - É a mesma coisa, caso tudo dê errado!

```
36      // Se algo der errado ...
37      //
38      function onFail(message) {
39        alert('Deu ruim, porque: ' + message);
40      }
```

Segundo Projeto

- **Passo 3:** Uma vez o código estando pronto, vamos adicionar plataforma e depois o plugin de camera em nosso aplicativo! Voltamos ao terminal!

```
usuario@pc:~/func$  cordova platform add android
```

```
usuario@pc:~/func$  cordova plugin add cordova-plugin-camera
```

- OBS: Repare nas mensagens de retorno pra ver se não retornou erro para nenhum dos comandos!!!
- **Passo 4**: Vamos criar nosso **APK**

```
usuario@pc:~/func$  cordova build android
```

Segundo Projeto

- **Passo 5:** Vamos jogar para o dispositivo móvel e ver se tudo deu certo!!!

```
usuario@pc:~/func$  cordova run android
```

- No mundo perfeito, dará certo de primeira!!!
- … mas no mundo real pode ser que você tenha algum problema! Para isso é importante saber debugar e encontrar os erros! Algumas dicas quanto a isso!

Segundo Projeto

- **Dica 1:** Tenha certeza que seu plugin esta no projeto!

```
usuario@pc:~/func$  cordova plugin ls
```

- **Dica 2**: Tenha certeza que seu html esta “chamando” corretamente suas funções no JavaScript! Eu faço isso via alerta! Se não esta chamando é porque algo pode estar errado no seu javascript ou ao importar o arquivo em seu html.

```
// Um botão chamará essa função!
function capturarFoto() {
    alert("ESTOU SENDO CHAMADAAAAA!!!"); //para debug
```

- **Dica 3:** Pesquise! Busque ajuda! A comunidade de Software Livre disponibiliza muitos materiais interessantes! Se não encontrar nada em português, faça a busca em inglês!

Segundo Projeto

- **Dica 4:** Os plugins funcionam de forma muito parecida! Existe um padrão no fluxo de execução! Geralmente nosso aplicativo chama o método principal do PLUGIN passando os três parametros básicos (funcaoSeTudoDerCerto, funcaoSeTudoDerErrado, arrayOpções) … aí depois o PLUGIN faz uma chamada para uma de nossas funções, dependendo do sucesso ou insucesso de sua operação!
  - ATENÇÃO! Alguns PLUGINS possuem funções que você inicia uma verificação constante (watch) … esse “watch” funciona como se fosse um serviço que de tempos em tempos chama a função de callback! Você pode a qualquer instante parar esse “watch” ou reiniciá-lo!
    - **Exemplo … Geolocation ---------->**

Segundo Projeto

- Aqui você irá obter as coordenadas uma vez:

```
navigator.geolocation.getCurrentPosition(geolocationSuccess,
                                         [geolocationError],
                                         [geolocationOptions]);
```

- Aqui você irá iniciar uma escuta que irá chamar sua função de CallBack de tempos em tempos!

```
var watchId = navigator.geolocation.watchPosition(geolocationSuccess,
                                                  [geolocationError],
                                                  [geolocationOptions]);
```

- A qualquer momento você pode parar a escuta matando o watchId

```
navigator.geolocation.clearWatch(watchID);
```

Segundo Projeto

FICAMOS POR AQUI!!!

Ah! O segundo projeto completo, já com os plugins, pronto

para “dar o run” esta disponível em https://github.com/aleaguado/oficinaCordova !

OBS: Se for rodar ele em seu PC, talvez seja bom antes executar “ cordova platform remove android” e “cordova platform add android” antes de fazer o build e o run ! Isso é para o caso de nossos computadores estarem em versões diferentes!

Onde estudar mais?

Materiais essenciais

- Sei pouco sobre HTML, CSS e JavaScript! Onde estudar?

  No link: http://www.w3schools.com/

  Para praticar: https://www.codecademy.com/

- Quero estudar mais o Cordova. Onde encontrar bons materiais?
  - Faça o curso online da Loiane: http://www.loiane.com/2014/02/curso-online-phonegap-gratuito/
  - Use a documentação do Cordova!!!

    https://cordova.apache.org/docs/en/latest/

Tá muito fácil! Tem mais coisas legais?

Framework - IONIC

- O IONIC é um framework que trabalha com o Cordova! É muito bacana!!!!
  - https://github.com/IonicBrazil

Final

Dúvidas? Questionamentos?

ale.garcia.aguado@gmail.com