02

Preparando Cordova

Transcrição

Agora iremos preparar o nosso ambiente local para trabalhar com o Cordova, assim não precisaremos trabalhar com o *PhoneGap Build*. O primeiro passo será ter o ***Node*** instalado no nosso computador. Entraremos no site do Node (https://nodejs.org), faremos o download e depois a instalação na sua plataforma. Em seguida, quando abrirmos o terminal - independente do sistema operacional -, já poderemos usar o *Node*. Usarei o comando `- + v`, para ver qual é a versão:

```
~ $ node -v
v4.2.3
~ $
```

A minha versão é `4.2.3`, mas a sua poderá ser uma mais recente.

Eu já possuo o *Node* instalado corretamente. O passo seguinte é a instalação do *Cordova* - que faz parte do pacote do *Node*. Quando usamos o pacote do Node, usamos o gerenciador `npm` para a instalação. Iremos incluir isto no código.

```
~ $ node -v
v4.2.3
~ $ npm install cordova
```

Precisamos cuidar de mais alguns detalhes antes de começarmos a instalação. Vamos incluir o comando `- + g` que significa que queremos a instalação do Cordova **global**, assim poderemos usar no computador com qualquer usuário. Se estivermos utilizando um Mac ou Linux, precisarei adicionar o `sudo`, porque precisaremos ter primeiro a permissão de administrador. Ele irá solicitar a senha, após digitá-la, ele começará a instalação.

```
~ $ node -v
v4.2.3
~ $ sudo npm install -g cordova
Password:
```

Como já fiz anteriormente a instalação, posso usar o comando `cordova -v` e ele irá mostrar qual a minha versão instalada.

```
~ $ cordova -v
5.4.1
~ $
```

A minha versão é a `5.4.1`. Após o teste, sei que o Cordova está instalado e posso começar o projeto.

Vamos criar o primeiro projeto com o Cordova local, que será também para o restaurante **Só de Cenoura**. Desta vez, trabalharemos com um projeto voltado para as garçonetes. O objetivo é que elas consigam anotar o pedido e conseguir direcioná-lo para a cozinha de forma automática.

O projeto do app será batizado de **Garçonapp**, para criá-lo, usaremos o comando `cordova create garconapp`. Nós receberemos primeiro o nome da pasta que queremos criar, ou seja, ele irá abrir uma pasta nova com este nome.

```
~ $ cordova create garconapp
```

Em seguida, iremos incluir o nome do pacote. Já havíamos utilizado no projeto anterior o `id` do arquivo `.xml`, no caso é o meu site escrito ao contrário, juntamente com o subdomínio `garconapp`.

```
~ $ cordova create garconapp org.sergiolopes.garconapp
```

Você precisará ter um domínio próprio, porque usando o meu, não será possível publicar o projeto na loja e outras ações.

Também iremos adicionar um título que será visualizado no ícone da Home: "Garçonete So de Cenoura".

```
~ $ cordova create garconapp org.sergiolopes.garconapp "Garçonete So de Cenoura"
```

Damos um *Enter* e ele criará um novo projeto.

```
~ $ cordova create garconapp org.sergiolopes.garconapp "Garçonete So de Cenoura"
Creating a new cordova project
~ $
```

O novo projeto criou a pasta `garconapp`. Nós iremos entrar nesta pasta, pelo código.

```
~ $ cordova create garconapp org.sergiolopes.garconapp "Garçonete So de Cenoura"
Creating a new cordova project.
~ $ cd garconapp/
garconapp $
```

Agora estamos dentro do projeto Cordova. Com isto, podemos fazer várias coisas. Quando criamos um projeto Cordova novo, ele não suporta inicialmente Android, iOS, Windows Phone... Por padrão, ele vem "pelado" de plataformas. Precisaremos adicioná-las.

Por exemplo, se quisermos trabalhar com Android iremos incluir no código `cordova platform add android`. Se quisermos trabalhar com IOS, a linha ficará semelhante: `cordova platform add ios`. No código ficará assim:

```
~ $ cordova create garconapp org.sergiolopes.garconapp "Garçonete So de Cenoura"
Creating a new cordova project.
~ $ cd garconapp/
garconapp $ cordova platform add ios
```

Temos várias plataformas possíveis e podemos instalar as que forem necessárias. O projeto é multiplataforma.

Para começarmos, antes de definir uma plataforma no código, vamos adicionar `browser`. Ela é uma plataforma de teste do Cordova, onde consigo rodar o meu projeto no próprio browser do computador (no nosso caso, o Google Chrome).

Desta forma não precisamos da infraestrutura do Android, nem do IOS, e facilita o processo.

Então, iremos adicionar o `browser` no nosso código.

```
~ $ cd garconapp/
garconapp $ cordova platform add browser
```

Ele irá baixar e depois fazer a instalação. Irão aparecer algumas informações no terminal.

```
 Creating a new cordova project.
 ~ $ cd garconapp/
 garconapp $ cordova platform add browser
 Adding browser project...
 Running command: /USers/alura/.cordova/lib/npm_cache/cordova-browser/4.0.0/package/bin/create ,
Creating Browser project. Path: platforms/browser
Discovered plugin "cordova-plugin-whitelist@1" via npm
Installing "cordova-plugin-whitelist" for browser

This plugin is only applicabe for versions of cordova-android greater than 4.0. If you have a pı

garconapp $
```

Em seguida iremos inserir no código `cordova run` e o nome da plataforma que utilizaremos `browser`.

```
garconapp $ cordova run browser
```

Damos *Enter* e ele irá abrir a plataforma, numa nova janela do Chrome, com algumas configurações especiais do Cordova. Ele irá rodar o projeto nesta janela. Vamos abrir no modo de emulação de um aparelho de celular. Se selecionar um celular o Google Nexus 5, por exemplo, ele apresentará a tela com as mesmas dimensões. Como nós acabamos de criar o app, ele irá nos apresentar o que chamamos de "HelloWorld!", que gera um ícone, com o nome "Apache Cordova" e a frase "Device is Ready".

Nós podemos mudar estes elementos. Vamos abrir o editor de texto Sublime, depois entraremos na pasta recém criada `garconapp`, ao abri-la veremos que ela tem elementos semelhantes as outras, como por exemplo, `config.xml` - bastante semelhante ao arquivo `config.xml` enviado anteriormente para o *PhoneGap Build*. Ele possui as *tags* `name`, `description` e `author`.

```
<name>Garçonete So de Cenoura</name>
<description>
    A sample Apache Cordova application that responds to the deviceready event.
</description>
<author email="dev@cordova.apache.org" href="http://cordova.io">
    Apache Cordova Team
</author>
<content src="index.html" />
<plugin name="cordova-plugin-whitelist" spec="1" />
```

Ele também gerou alguns itens, que podemos editar. Por exemplo, podemos alterar o nome do `author` e o email.

```
<author email="dev@cordova.apache.org" href="http://cordova.io">
   Sérgio Lopes
</author>
```

Ele também disponibiliza o `id="org.sergiolopes.garconapp"` , que é importante por ser único.

Temos outra pasta disponível bastante importante é: `www` . Nela encontraremos todo o conteúdo CSS e JS, integrantes do nosso projeto. É o mesmo que nós criamos no *PhoneGap* sem muita estrutura de pastas. No Cordova, iremos criar usando a pasta `www` . Quando a abrimos, haverá um HTML gerado bastante longo, com várias informações, incluindo o título padrão "Hello World". Encontraremos as *string*s que vimos no navegador. Ele nos mostrava "Apache Cordova". Vamos alterar no código para `Oi Cordova` .

```
    <title>Hello World</title>
\\...

<body>
    <div class="app">
        <h1>Oi Cordova</h1>
        <div id="deviceready" class="blink">
            <p class="event listening">Connecting to Device</p>
            <p class="event received">Device is Ready</p>
        </div>
        </div>
        <script type="text/javascript" src="cordova.js"></script>
        <script type="text/javascript" src="js/index.js"></script>
    </body>
 </html>
```

Após salvar o arquivo HTML, vamos para o terminal e vou rodar de novo.

```
garconapp $ cordova run browser
```

Ele irá abrir o navegador novamente e já veremos as alterações aparecerem: Juntamente com o ícone padrão, veremos a mensagem "OI CORDOVA". Nós podemos alterar o HTML, JS e CSS, à medida que desenvolvermos o app. Em seguida, veremos como rodar o aplicativo no Android e IOS.