05

Acesso no dispositivo real

Transcrição

Testamos o nosso app com o Ajax no navegador, agora, faremos o mesmo no dispositivo móvel. No Terminal, vamos usar o comando `cordova run android`. Ele começará a instalar o app e faremos o mesmo teste, marcando um pedido. Clicaremos no botão de finalização, porém depois, veremos que não abrirá a janela de resumo do pedido. No navegador, o mural visualizado pelos cozinheiros não receberá também os pedidos.

Se fizermos um novo pedido através do navegador, veremos que o pedido será efetuado. Mas no dispositivo ainda não funciona. Temos ainda alguns problemas no projeto: para ter acesso ao Ajax que está em um servidor remoto, você precisa autorizar a ação, que recebe o nome **CORS** (*Cross-Origin Resource Sharing*). O nosso servidor `cozinhapp.sergiolopes.org` precisa permitir o acesso externo. No navegador, após abrir o `Inspecionar`, vamos clicar na aba `Network`, onde conseguimos ver as chamadas de rede. Faremos um novo teste com o app, na janela de resumo, vou digitar que a mesa é a 43. Ele fará um Ajax.

No navegador, veremos quais são os cabeçalhos dessa chamada Ajax. Um bastante relevante é `acces-control-allow-origin`.

```
Response Headers
    access-control-allow-origin: *
    connection: keep-alive
    content-type:text/plain; charset=UTF-8
    date: Tue, 19 Jan 2016 18:05:34 GMT
    transfer-enconding: chumked
    vary: Accept-Encoding
```

Do que se trata? O `back-end` do Cozinhapp precisa adicionar o cabeçalho na resposta ("Responde Headers"), para permitir que ele possa acessar fora do servidor. Com `acces-control-allow-origin`, estamos dizendo que habilitamos outros domínios/origens a acessarem o Cozinhapp. O `*` (asterisco) indica que o mundo inteiro poderá acessá-lo. Podemos também colocar no mesmo campo o nome de domínio, como por exemplo, poderíamos permitir o acesso do `sodecenoura.com.br`. Porém, neste caso, apenas o Só de Cenoura terá acesso. Mas nós queremos que todos os aplicativos Cordova tenham acesso, e o dispositivo de cada usuário não tem uma origem, um *hostname*. Se verificarmos, o HTML que está rodando, funciona como se tivesse `file:`. Logo, para conseguirmos ter acesso aos dispositivos do mundo inteiro, usaremos o "*". Esta não é uma brecha de segurança. Mas se acharmos necessário, podemos adicionar autenticação no mecanisto. Porém, se já existia essa configuração habilitando todos os celulares, porque o aplicatico não funcionou quando fizemos o pedido. Porque existe mais um detalhe. Por padrão, os apps Cordova bloqueiam os acessos externos. É necessário abrir explicitamente o acesso aos domínios desejados. Ou seja, precisaremos permitir o acesso da URL `cozinapp.sergiolopes.org` dentro do app Cordova. Nós faremos isto com um plugin.

Vamos no Terminal e usaremos o comando `cordova plugin add cordova-plugin-whitelist`.

```
garconapp $ cordova plugin add cordova-plugin-whitelist
```

Geralmente, ele já vem instalado quando criamos um app novo. Talvez, você já o tenha instalado, mas nós pediremos para instalá-lo, por precaução.

Este plugin irá habilitar uma nova configuração no `config.xml` - que inclusive, já havia aparecido no nosso arquivo antes, junto com diversas preferências. Mas nós a apagamos, porque ainda não tinha utilidade.

Agora teremos a *tag* `acces` que permite configurar quais origens, queremos ter acesso. Iremos incluir o domínio desejado nela. Precisaremos fazer isto para cada `origen` que quisermos acessar.

```
<access origin="http://cozinhapp.sergiolopes.org"/>
```

O nosso código ficou assim:

```
<author email="qualquer@sergiolopes.org" href="http.//sergiolopes.org">
    Sérgio Lopes
</author>

<access origin="http://cozinhapp.sergiolopes.org"/>

<preference name="Orientation" value="portrait" />
<preference name="BrackgroundColor" value="0xF2F2F2FF" />
/...
```

Se você estiver em uma situação semelhante, em que sabemos qual servidor irá acessar, podemos especificá-lo no código. Mas evite usar `origen= *` , se de fato não necessário.

Adicionada a configuração, vamos rodar o app novamente e ver se ele funciona corretamente.

Vamos abrir o celular, ele fará a compilação do app com o plugin e fará a instalação da última versão. Quando o "Só de Cenoura" já estiver rodando no dispositivo, faremos um novo pedido. Para testar, marcaremos dois bolos e o número da mesa, "51". Clicaremos em "Pedir" e processo é finalizado com sucesso. No navegador, veremos que o pedido da "Mesa 51" chegará no nosso *back-end*. Ele funcionou, porque liberamos o acesso ao `origen` . O Android irá aceitar o plugin e no iOS, está configuração também será aceita - a partir dela, o iOS criará outro arquivo, que também precisará habilitar o acesso. Então, o `access origin` servirá tanto para Android, como para iOS.