Adobe PhoneGap Build
para construção de aplicações móveis híbridas

Introdução

No artigo "Criar uma aplicação móvel com jQuery Mobile" [Programar #58] vimos como usar a *framework* jQuery Mobile para programar uma aplicação Web direcionada a dispositivos móveis. Neste artigo, vamos ver como usar o serviço PhoneGap Build da Adobe para gerar uma aplicação para Android.

Adobe PhoneGap

PhoneGap é uma *framework* de desenvolvimento de aplicações móveis híbridas. Aplicações híbridas são aplicações que combinam componentes nativos e componentes Web. Do ponto de vista do utilizador e da plataforma móvel, uma aplicação híbrida é indistinguível de uma aplicação nativa. No entanto, internamente, uma aplicação híbrida utiliza um componente *Web View* que contém a maioria do conteúdo e lógica da aplicação – ou seja, a aplicação é essencialmente programada como se de um aplicação web se tratasse. A framework PhoneGap, baseada em Apache Cordova, permite que as aplicações híbridas tenham acesso a funcionalidades nativas através de componentes específicos para cada plataforma móvel (e.g., Android, iOS, Windows) mas cuja interface é exposta em JavaScript (o programador não se preocupa com a plataforma).

PhoneGap Build

O website PhoneGap Build é uma interface web simplificada para o processo de build da *framework* PhoneGap, permitindo-nos "empacotar" uma aplicação web numa aplicação nativa para as várias plataformas móveis.

Uma das vantagens deste website é que nem sequer temos de conhecer muito sobre PhoneGap uma vez que o website esconde a maioria dos detalhes.

Figura 1 – Ecrã da aplicação. Código completo: https://goo.gl/P9dEHZ

Preparar a aplicação

A Figura 1 mostra um ecrã da aplicação construída no artigo da edição anterior. O link fornecido pode ser usado para descarregar o código da aplicação. A forma mais prática para descarregar o código será:

1. Seguir o link fornecido na Figura 1
2. Escolher a opção "Export" no canto inferior direiro
3. Escolher a opção "Export .zip"

Depois de descarregar o ficheiro Zip e descomprimi-lo, o resultado será um único ficheiro HTML que iremos de seguida modificar ligeiramente.

O código original inclui referências ao jQuery e ao jQuery Mobile com recurso a endereços CDN remotos (e.g., http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.css).

Para evitar ligações de rede desnecessárias na nossa aplicação vamos converter estas referências em referências a ficheiros locais. Para isso temos de:

1. Editar o ficheiro index.html
2. Procurar todas as referências a ficheiros externos (CSS, imagens, JavaScript)
3. Seguir essas referências e descarregar os ficheiros, gravando-os na mesma pasta do projecto
4. Alterar as referências no ficheiro index.html, mantendo apenas o nome do ficheiro.

Por exemplo, a referência ao ficheiro CSS do jQuery Mobile seria convertido de:

```
 <link rel='stylesheet prefetch' href='http://
code.jquery.com/mobile/1.4.5/
                        jquery.mobile-1.4.5.css'>
```

Para:

```
<link rel='stylesheet prefetch'
                    href=jquery.mobile-1.4.5.css'>
```

A estrutura final da pasta do projecto deverá ser algo como:

projecto-jquery-mobile

|-- index.html

|-- jquery.min.js

|-- jquery.mobile-1.4.5.css

|-- jquery.mobile-1.4.5.js

|-- space-cube.svg

Depois de fazermos e gravarmos estas alterações podemos voltar a comprimir a pasta do projecto num ficheiro Zip.

Gerar o ficheiro keystore

As aplicações móveis distribuídas através de uma loja (e.g., Android Play, ou App Store), precisam de ser assinadas. Para tal é necessário a geração de um certificado que identifica a entidade que criou a aplicação e uma chave privada e uma pública. A chave privada é usada para assinar digitalmente a aplicação. A emissão de um certificado é feita tipicamente por uma entidade reconhecida globalmente, mas, neste artigo, iremos gerar nós próprios este certificado. Para tal recorremos à ferramenta *keytool* do Java.

Para executar esta ferramenta, temos de usar uma linha de comando (Command Prompt no Windows, ou Terminal no Mac OS). Aqui vou exemplificar o comando em Windows (com fundo cinza o texto introduzido por mim, em resposta aos pedidos do *keytool*; [enter] representa o pressionar da tecla "Enter"):

```
C:\Users\jorge>"c:\Program Files\Java\jdk1.8.0_121
\bin\keytool.exe" -genkey -alias androidkey -v -
keystore android.keystore [enter]
Enter keystore password: supermegapass[enter]
Re-enter new password: supermegapass[enter]
What is your first and last name?
  [Unknown]:  Jorge Cardoso[enter]
What is the name of your organizational unit?
  [Unknown]: [enter]
What is the name of your organization?
  [Unknown]: [enter]
What is the name of your City or Locality?
  [Unknown]:  Porto[enter]
What is the name of your State or Province?
  [Unknown]: [enter]
What is the two-letter country code for this unit?
  [Unknown]:  PT[enter]
Is CN=Jorge Cardoso, OU=Unknown, O=Unknown,
L=Porto, ST=Unknown, C=PT correct?
  [no]:  yes[enter]

Generating 1,024 bit DSA key pair and self-signed
certificate (SHA1withDSA) with a validity of 90
days
        for: CN=Jorge Cardoso, OU=Unknown,
O=Unknown, L=Porto, ST=Unknown, C=PT
Enter key password for <androidkey>
        (RETURN if same as keystore password):
[enter]
[Storing android.keystore]
```

O ficheiro resultante, especificado no parâmetro "keystore" do *keytool*, é colocado na mesma pasta onde executamos o *keytool* (no meu caso em C:\Users\jorge\android.keystore). É este ficheiro que irá ser adicionado ao PhoneGap Build. É importante notar que o parâmetro "alias" do *keytool* terá de ser introduzido no site do PhoneGap Build (assim como as passwords que escolhemos) pelo que convém memorizar.

Website PhoneGap Build

De seguida, podemos registarmo-nos no site PhoneGap Build, escolhendo o plano grátis (ver Figura 2).

Adicionar o ficheiro keystore ao PhoneGap Build

Antes de configurarmos a nossa aplicação no PhoneGap Build, devemos fazer *upload* do ficheiro android.keystore, tal como indicado na Figuras 3 e 4. É necessário cuidado ao introduzir o "alias" uma vez que este deve ser exactamente igual ao parâmetro "alias" que usamos no comando *keytool*.

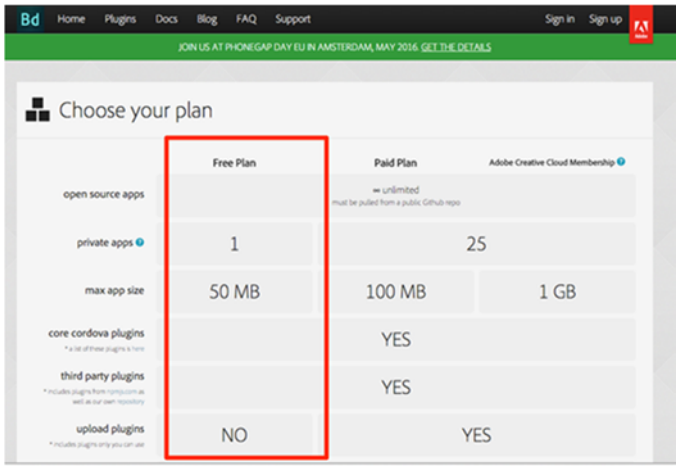


Figura 2 – Registo no site PhoneGap Build. http://build.phonegap.com.

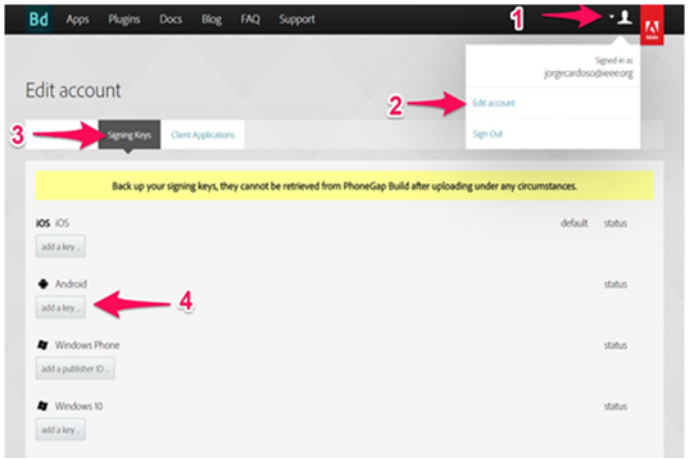
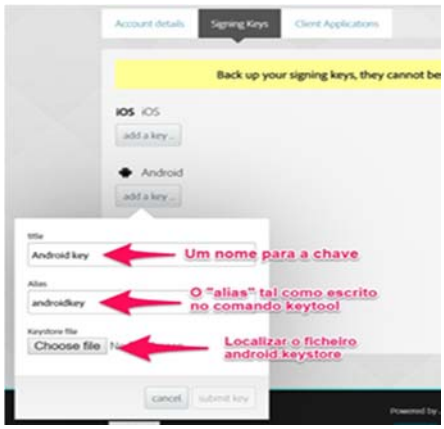


Figura 3 – Adição de chave Android.



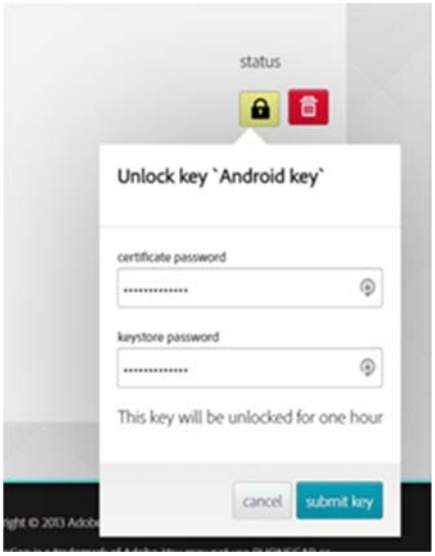
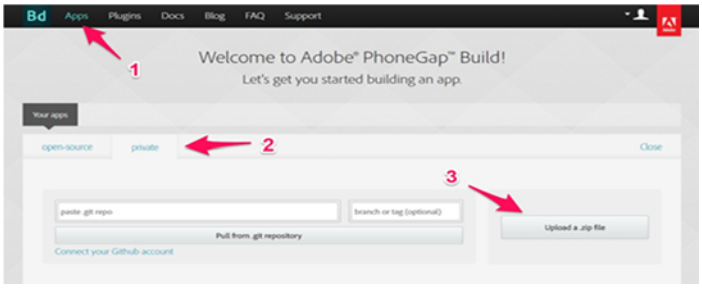


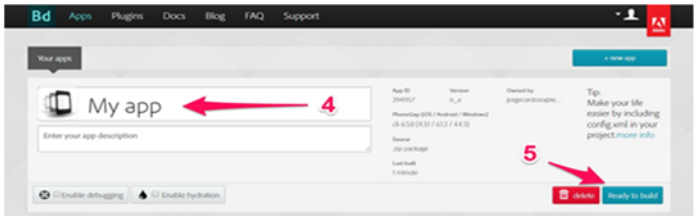
Figura 4 – Upload do ficheiro android.keystore e desbloqueamento da chave.

Depois de localizarmos o ficheiro android.keystore e clicarmos em "submit key", devemos desbloquear a keystore clicando no ícone do cadeado (do lado direito na mesma página) e introduzindo a password que escolhemos para a nossa keystore quando corremos o comando keytool (ver Figura 4, lado direito).

Criar a aplicação no PhoneGap Build

O passo seguinte será configurar uma aplicação no PhoneGap Build e carregar o nosso código fonte. Para tal, acedemos à secção "Apps" e fazemos upload do ficheiro zip que criamos no início.





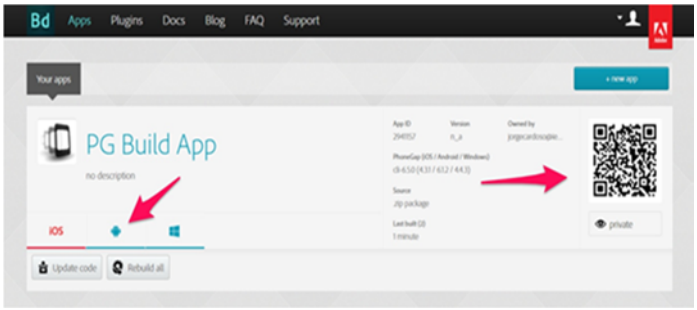


Figura 5 – Configurar aplicação, carregar código fonte e correr processo de build.

Depois do upload, o PhoneGap Build mostra-nos um ecrã em que podemos definir um nome e descrição para a nossa aplicação e iniciar o processo de build. A Figura 5 mostra os passos necessários.

Testar a aplicação

O último passo é instalar e testar a aplicação no nosso dispositivo móvel. Há duas formas de o fazer. Uma é descarregar manualmente o ficheiro APK (*Android Package file*), clicando no ícone relativo ao Android (Figura 5, em baixo); transferindo o ficheiro para o dispositivo móvel; e finalmente fazendo a instalação executando o APK. A outra forma é usando um leitor de QR Codes e apontando o dispositivo móvel para o QR Code que aparece no ecrã (Figura 5, em baixo).

O ficheiro APK que podemos descarregar depois do processo de *build* é também o ficheiro que precisamos para distribuir a nossa aplicação através da loja Google Play.

Conclusão

Os passos que vimos neste artigo são suficientes para, depois de termos uma aplicação Web programada para dispositivos móveis, conseguirmos gerar e distribuir uma aplicação como qualquer outra aplicação nativa.

A *framework* de desenvolvimento híbrido PhoneGap tem, obviamente, muitas funcionalidades que não exploramos aqui. No entanto, para quem está sobretudo habituado a programação Web e quer experimentar ou distribuir de forma rápida a sua aplicação, esta pode ser uma boa forma de o fazer.

AUTOR

Escrito por Jorge C. S. Cardoso
Professor auxiliar convidado no Departamento de Engenharia Informática da Universidade de Coimbra onde lecciona disciplinas relacionadas com tecnologias web, programação, e interacção humano-computador. É autor do livro "Java para Telemóveis" editado pela FEUP Edições. Licenciado em Engenharia Informática pela Universidade do Porto (FEUP), mestre em Sistemas Móveis e doutor em Tecnologias e Sistemas de Informação pela Universidade do Minho (Escola de Engenharia).