HCIM Summer Workshop

Guião Hands-on

Exemplo de Aplicação Facebook

1. Download do Software Development Kit (SDK)

O download do SDK do Facebook encontra-se disponível aqui.

Depois de fazer o download do ficheiro, extraia para uma pasta que seja fácilmente reacessível.

2. Importar para o IDE

O ficheiro contém uma directoria facebook, a qual contém o SDK em si. Para importar o SDK para o seu IDE (Eclipse, no caso), vá a 'File' > 'Import' e no menu escolha 'General' > 'Existing Projects into Workspace' e importe essa pasta:

	Import	>		
mport Projects				
some projects canno workspace	t be imported because they already exist in the			
Select root directory:	C:\Users\Gonçalo Gomes\Documents\faceboo	Browse		
) Select archive file:		Browse		
ojects:				
FacebookSDK (C	:\Users\Gonçalo Gomes\Documents\facebook-ar	Select All		
		Deselect All		
		Refresh		
¢	>			
<	orkspace			
< Copy projects into w Working sets	orkspace			
 Copy projects into w Working sets Add project to wor 	orkspace king sets			
Copy projects into w Working sets Add project to wor Working sets:	orkspace king sets	Select		
Copy projects into w Working sets	rorkspace king sets	Select		
Copy projects into w Working sets Add project to wor Working sets:	orkspace king sets	Select		
Copy projects into w Working sets	orkspace king sets	Select		

De seguida carregue em 'Finish' para dar o processo como concluído e verá como o SDK é adicionado como um novo "projecto" à sua biblioteca (lado direito do IDE).

3. Importar o exemplo

Para importar o exemplo de projecto que usa o SDK do Facebook, siga o exemplo análogo ao usado para importar o Facebook SDK no ponto anterior. Mais uma vez vá a 'File' > 'Import' e no menu escolha 'General' > 'Existing Projects into Workspace' e importe o projecto que se encontra neste link com o nome de Exemplo:

0	Import	- • ×
Import Projects Select a directory to s	search for existing Eclipse projects.	
Select root director	ry: \HCIM\Summer Workshop\Materials\Exemple	Browse
○ Select archive file:		Browse
Projects:		
Hackbook (C:	:\Users\Gonçalo Gomes\Dropbox\TIES\HCIM\Sum	m Select All
		Deselect All
		Refresh
<		>
<	o workspace	>
 Copy projects into Working sets Add project to w 	o workspace vorking sets	>
 Copy projects into Working sets Add project to w Working sets: 	o workspace vorking sets	Select
Copy projects into Working sets Add project to w Working sets:	o workspace vorking sets	Select
< Copy projects into Working sets Add project to w Working sets:	o workspace vorking sets	Select
< Copy projects into Working sets Add project to w Working sets:	o workspace vorking sets	Select

4. Pronto a correr!

Finalmente o exemplo está pronto a correr, com três funcionalidades básicas de experiência: Postar no Mural, Enviar Pedidos de Aplicação e Ver os Amigos.

A Sua Aplicação Facebook

1. Registar Aplicação no Facebook Developers

Clique para aceder ao Facebook Devs

Carregue em **Criar nova aplicação**. De seguida dê-lhe um **nome**, **namespace** e **categoria** à sua escolha:

Nome da aplicação: [?] WSTest	Valid			
Namespace da wstest_namespace aplicação: [?]	Disponível			
Categoria da Comunicação Escolhe uma subcateg	goria: 🗸			
Alojamento web: [?] Sim, gostaria de receber alojamento web grátis fornecido pela Heroku (Learn More)				
o prosseguires, estás a concordar com as Políticas da Plataforma do Facebook Continuar Cancelar				

Uma vez criada, note que a aplicação possui um ID associado, visível no campo App ID:



Este ID é único e que faz a associação entre a aplicação e a chave única gerada para cada um delas. Para obter tal chave, é necessário que seja instalado o **OpenSSL**, o qual pode ser obtido <u>aqui</u>. Após instalado o OpenSSL, deve-se abrir uma linha de comandos e correr o seguinte comando:

```
keytool -exportcert -alias <ALIAS_QUALQUER> -keystore
<CHAVE KEYSTORE> | openssl sha1 -binary | openssl base64
```

Nota importante: O comando keytool está tipicamente contido na pasta do **Java/jdk/bin/** ou **Java/jdk/bin/**, pelo que devem colocar o caminho antes do comando. O mesmo deve ser feito para o comando **OpenSSL**.

A chave da keystore é obtida em 'Window' > 'Preferences' > Tab 'Android' > Subtab 'Build' > Campo 'Default debug keystore':

0	Pre	ferences	_ 🗆 🗙
type filter text	Build		⇔ • ⇔ • •
 ▷ General Android Build DDMS Editors Launch Lint Error Checking ▷ LogCat NDK Usage Stats ▷ Ant ▷ C/C++ ▷ Help ▷ Install/Update ▷ Java ▷ Run/Debug ▷ Team Validation ▷ XML 	Build Settings: ✓ Automatically refresh I ✓ Force error when exter Skip packaging and der Build output Silent Normal ✓ Verbose Default debug keystore: MD5 fingerprint: SHA1 fingerprint: SHA1 fingerprint: SHA1 fingerprint:	Resources and Assets folder on build nal jars contain native libraries xing until export or launch. (Speeds up autor C:\Users\Gonçalo Gomes\.android\debug. B3:15:F8:CB:47:89:61:7A:FF:18:B2:BF:F6:F4:6 C0:38:48:04:20:12:5C:7E:9F:1E:58:9E:9D:74:4E	keystore 4:97 D:2B:B6:41:94:69 Browse faults Apply
?		ОК	Cancel

Após isto, o resultado de correr o comando na linha de comandos deverá ser algo como:

C1	Linha de comandos	-	×	
ARNING: can't open config file: (Illegal option: Gomes\.android\ keytool -exportcert [OPTION]	C:\OpenSSL-Win64\bin\openssl.cfg lebug.keystore		^	•
Exports certificate				
Options: -rfc	output in RFC style			
-alias <alias> -file <filename> -keystore <keystore> -storepass <arg> -storetype <storetype> -providername <providername> -providerclass <providerclass> -providerarg <arg> -providerpath <pathlist> -v -protected</pathlist></arg></providerclass></providername></storetype></arg></keystore></filename></alias>	alias name of the entry to process output file name keystore name keystore password keystore type provider name provider class name provider argument provider classpath verbose output password through protected mechanism			
Use "keytool -help" for all avai. 2jmj715rSw0yUb/v1WAYkK/YBwk=	lable commands			
C:\Program Files\Java\jdk1.7.0_2	L\bin>		~	1

Como se pode ver, é gerada então uma chave com MD5, para que possamos certificar a nossa aplicação e garantir a integridade e unicidade da mesma.

De seguida voltamos ao *lobby* da aplicação recém-criada do Facebook Developers e colocamos a chave gerada no campo "**Key Hashes**" e activamos o campo que diz "**Início de sessão no Facebook**". Guardam-se a alterações efectuadas, posteriormente.

Site com início de sessão pelo Facebook	Log in to my website using Facebook.		
🔗 Aplicação no Facebook	Use my app inside Facebook.com.		
◇ Internet no telemóvel	Bookmark my web app on Facebook mobile.		
✓ Aplicação nativa para i05	Publish from my iOS app to Facebook.		
🤣 Aplicação nativa para Android	×		
Nome do pacote: [?]			
Nome da cadeira: [?]			
Key Hashes: [?]	2jmj7l5rSw0yVb/vlWAYkK/YBwk= ×		
Início de sessão no Facebook: [?]	Ativar O Desativar		
Deep Linking: [?]	O Ativar () Desativar		
🛷 Page Tab	Build a custom tab for Facebook Pages.		
	Guardar alterações		

Nota: Para além disso, será necessário voltar a aceder a este ecrã, no qual precisaremos de registar o nome do *package* principal da aplicação registada.

2. Criar uma aplicação que usa o Facebook Login

Agora é tempo de criarmos a aplicação que usará o Login do Facebook. Para tal, no Eclipse clicamos em "File" > "New" > "Project" > "New Android Application":

0	New Android Application	-	×	
New Android Application				
The application name for most apps begins with an uppercase letter				
Application Name:	facebook_test			
Project Name:	facebook_test			
Package Name:&	com.example.facebook_test			
Minimum Required SDK:	API 10: Android 2.3.3 (Gingerbread) V			
Target SDK:0	API 10: Android 2.3.3 (Gingerbread)			
Compile With:0	Google APIs (Google Inc.) (API 10) 🗸 🗸			
Theme:0	Theme: 6 None Y			
Choose the base theme to use for the application				
?	< Back Next > Finish		Cancel	

Nota: Tendo em conta que grande parte dos dispositivos que possuimos correm a versão do 2.3.3 do Android, configuraremos o SDK para correr com tais definições.

Siga as instuções até criar uma actividade simples com o nome que desejar:

File Edit Refactor Source Navigate Search Project Run Window Help
Image: Construction of the construc
If Package Explorer IX If Seckage Complexity main xml If MainActivity java IX If Backage Complexity If Backage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity If Dackage Complexity
▲ Bacebook_test A D 1 package com.example.facebook_test; 2 A Bacebook_test; A Bacebook_test; A Bacebook_test; A Bacebook_test; A Bacebook_test; A Bacebook_test; A Bacebook_test; A Bacebook_test;
a 🗁 src 2
a
J MainActivity.java 6
> 🤔 gen [Generated Java Files] 7 public class MainActivity extends Activity {
b mil Google APIs [Android 2.3.3]
> 🛋 Android Private Libraries 99 @ @Override
Android Dependencies
assets
bin 12 SetContentView(R.layout.activity_main);
b b libs
4 😓 res
b 🕞 dravable-hdpi
evering evering evering bools on creater on treater on the second s
b drawable mdpi
b drawable xhdpi 19 gret Menul nef later (), inflate (B. menu) agin, menu):
c drawable-xxhdpi 20 return true:
a activity main.xml 22
23)
⊳ values 24
> values-sw60dp
> values sw720dp-land
ic launcher-web.ong
proquard-project.txt
project.oroperties
≥ S≅ FacebookSDK
FragmentTester > <
🕆 Relinder
Teste
Android
Writable Smart Insert 1 : 1 1 6M of 433M

Agora sim, voltamos ao *lobby* da nossa actividade no Facebook Developers e **registamos a** aplicação, nome da classe principal (que corre o Login) e o nome do *package*:

🧇 Aplicação nativa para Android		×	
Nome do pacote: [?]	com.example.facebook_test		
Nome da cadeira: [?]	com.example.facebook_test.MainActivity		
Key Hashes: [?]	2jmj7l5rSw0yVb/vlWAYkK/YBwk= ×		
Início de sessão no Facebook: [?]	Ativar O Desativar		
Deep Linking: [?]	O Ativar () Desativar		
Site com início de sessão pelo Facebook	Log in to my website using Facebook.		
🔗 Aplicação no Facebook	Use my app inside Facebook.com.		
☆ Internet no telemóvel	Bookmark my web app on Facebook mobile.		
🔗 Aplicação nativa para i05	Publish from my iOS app to Facebook.		
🔗 Page Tab	Build a custom tab for Facebook Pages.		
	Guardar alterações		

Caso se precise de verificar qual o nome do *package* com mais precisão, pode-se ainda consultar o **Manifest.xml** da aplicação, e consultar a *tag* "package".

3. Usar o SDK como biblioteca da Aplicação

Temos então de ligar a aplicação ao SDK, fazendo uso deste último como biblioteca, podendo assim usar todas as chamadas de métodos do Facebook. Para tal, carregamos em "Project" > "Properties" > Tab "Android" > "Library" > "Add" e escolhemos a pasta do SDK usado no primeiro projecto:

0	Properties for facebook_test ×			
type filter text	Android		$\Leftrightarrow \bullet \Rightarrow \bullet \bullet \bullet$	
Resource Android	Project Build Target			
Android Lint Preferences Builders	Target Name Vend	or	Platfor AP	
Java Build Path	Android 2.3.3 Andr	oid Open Source Project	2.3.3 10	
Java Code Style	Google APIs Goog	ile Inc.	2.3.3 10	
Java Compiler Java Editor	Android 4.2.2 Andr	old Open Source Project	4.2.2 17	
Javadoc Location				
Project References				
Run/Debug Settings Task Tags	Library			
Validation	Is Library			
	Reference	Project	Add	
	 //facebook-androi 	FacebookSDK	Remove	
			Up	
			Down	
		Restore Defau	lts Apply	
?		ОК	Cancel	

Depois, vamos ao nosso ficheiro **res\strings** e adicionamos uma string que irá conter o ID da aplicação, para finalmente ligarmos com a chave gerada ainda há pouco no ponto anterior:

ⓓ *strings.xml ⊠				
🚔 Android Resources (default)				
Resources Elements (S) (C) (D) (S) (D) (S)	S I P Az	Attributes for String		
 (\$) app_name (String) (\$) action_settings (String) (\$) hello_world (String) (\$) String 	Add Remove Up Down	©Strings©, with optional simple formatting, can be stored and retrieved as resources. You can add formatting to your string by using three standard HTML tags: b, i, and u. If you use an apostrophe or a quote in your string, you must either escape it o enclose the whole string in the other kind of enclosing quotes. Name app_id Value*	or	

Depois disto, temos de dar acesso à Internet à nossa aplicação e para isso temos de definir no nosso **Manifest.xml** que vamos requisitar este serviço:

d *facebook_test Manifest ∞		□ □
🛱 Android Manifest Permissions		
Permissions	POPPAz	Attributes for Uses Permission
() Uses Permission	Add Remove Up Down	The tag requests a {@link #AndroidManifestPermission < permission>} that the containing package must be granted in order for it to operate correctly. Name android.permission.INTERNET v

Finalmente temos de criar *metadata* para o ID da aplicação, para início de sessão. Adicionalmente temos de declarar a classe **com.facebook.LoginActivity** no nosso **Manifest.xml**, portanto neste adicionamos o seguinte:

<activity android:name="com.facebook.LoginActivity" ></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activity></activi

```
e
```

```
<meta-data
android:name="com.facebook.sdk.ApplicationId"
android:value="@string/app_id" />
```

dentro da tag "application"

4. Implementar a aplicação

- 1. Começamos então a implementar a aplicação, colocando na mesma um objecto que sirva de *Label* (e.g. TextView). Esta *Label* servirá para "dar as boas vindas" ao utilizador logado.
- 2. Uma parte importante do SDK é a classe Session, a qual faz a gestão da autenticação e autorização dos utilizadores do Facebook. Tendo em conta que se quer fazer *login* e *logout* na aplicação, deve-se implementar um mecanismo que permita fazer isso e para tal, adiciona-se o seguinte código à classe:

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    Session.getActiveSession().onActivityResult(this, requestCode, resultCode,
    data);
}
```

3. De seguida, precisamos de fazer com que o utilizador seja logado quando a actividade se inicía. Para isso, precisamos de implementar o método **openActiveSession quando a actividade começa (onCreate())**:

```
// start Facebook Login
Session.openActiveSession(this, true, new Session.StatusCallback() {
    // callback when session changes state
    @Override
    public void call(Session session, SessionState state, Exception
    exception) {
    }
});
```

4. Como este método irá estar sempre a ser chamado (porque funciona com *callback*), nós só queremos que aconteça isso enquanto não houver uma sessão activa, então adicionamos a seguinte condição:

```
if (session.isOpened()) {
}
```

5. Finalmente, queremos fazer um pedido ao Facebook que nos retorne o nosso nome. Para que tal aconteça, fazemos um pedido assíncrono (a actividade continua a correr) ao Facebook, para que nos retorne informação sobre nós e filtramos o que queremos, no caso, o nosso nome:

```
// make request to the /me API
Request.executeMeRequestAsync(session, new Request.GraphUserCallback() {
    // callback after Graph API response with user object
    @Override
    public void onCompleted(GraphUser user, Response response) {
    }
});
```

Nota: O objecto GraphUser tem encapsulada informação relevante acerca do utilizador.

Desafio 1:

Apresentar a data de nascimento do utilizador, bem como o primeiro e último nomes (separados)

Desafio 2:

Apresentar a localização actual do utilizador e a sua terra natal

Desafio 3:

Obter o sitio onde o utilizador trabalha