Ejecutar una aplicación (1)

Android permite utilizar otras aplicaciones disponibles en el dispositivo para realizar una tarea determinada.

Por ejemplo podemos invocar la aplicación de ajustes de Android para que el usuario habilite algún elemento requerido por nuestra aplicación: Bluetooth, GPS, etc.

El inconveniente de este mecanismo es que requiere conocer el nombre del paquete de la aplicación correspondiente.

38

Ejecutar una aplicación (2)

En primer lugar obtendremos el gestor de paquetes de Android (*PackageManager*) mediante *getPackageManager.*

A continuación crearemos un *Intent* y llamaremos al método *getLaunchForPackage* pasándole como parámetro el nombre del paquete de la aplicación que queremos lanzar.

Por último comprobamos el valor del *Intent* obtenido, ya que si es nulo significa que la aplicación no está disponible.

39

Ejecutar una aplicación (3)

```
Button botSettings =(Button)findViewById(R.id.botSettings);
botSettings.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View view) {
    PackageManager pm = getPackageManager();
    Intent intent = pm.getLaunchIntentForPackage("com.android.settings");
    if (intent != null) {
      startActivity(intent);
    } else {
      int duracion = Toast.LENGTH_SHORT;
      String mensa = "La aplicación Settings no está disponible";
      Toast alerta = Toast.makeText(MainActivity.this,mensa,duracion);
      alerta.show();
    }
  }
});
```

40

Ejecutar una acción (1)

Invocar a una aplicación presenta dos problemas:

1) Se necesita conocer el paquete de la aplicación.
2) Alguno de los dispositivos en que se ejecute nuestra app puede no tener instalada esa aplicación concreta.

Resulta mucho más conveniente establecer la acción a realizar y que sea el sistema el que determine la aplicación que se encargará de llevarla a cabo.

41

Ejecutar una acción (2)

Para invocar una acción deberemos crear un *Intent* cuyo contenido dependerá de esa acción, siendo los principales:

- *action*: establece la acción a realizar: como mostrar (*ACTION_VIEW*), enviar (*ACTION_SEND*, *ACTION_SENDTO*), llamar (*ACTION_DIAL*), etc.
- *data*: establece los datos sobre los que realizar la operación expresados como URI: ruta del archivo a mostrar, contacto, número de teléfono, etc.

42

Ejecutar una acción (3)

Ejemplo: abrir el marcador telefónico

```
Intent intent = new Intent(Intent.ACTION_DIAL);
```

Ejemplo: realizar una llamada

```
Intent intent = new Intent(Intent.ACTION_DIAL);
intent.setData(Uri.parse("tel:600123456"));
```

Ejemplo: mostrar una web

```
Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.google.com"));
```

43

Ejecutar una acción (4)

En ocasiones tendremos que añadir algún atributo secundario para realizar la acción:

- *category*: información adicional de la acción a realizar: *CATEGORY_LAUNCHER*, *CATEGORY_ALTERNATIVE*.
- *extra*: permite añadir información adicional en forma de parejas clave-valor mediante *putExtra*, por ejemplo el asunto o el contenido de un mensaje.

44

Ejecutar una acción (5)

- *type*: normalmente el tipo de dato se obtiene directamente de los datos, pero este atributo permite fijar un tipo MIME concreto: *application/pdf, message/rfc822*, *text/plain*, *etc.*
- *component*: permite establecer de forma explícita el componente que se encargará de realizar la acción en lugar de dejar al sistema que elija uno o muestre las opciones disponibles.

45

Ejecutar una acción (6)

Ejemplo: enviar SMS

```
Intent intent = new Intent(Intent.ACTION_SENDTO);
intent.setData(Uri.parse("smsto:600123123"));
intent.putExtra("sms_body", "Texto del mensaje");
```

Ejemplo: enviar eMail

```
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setType("message/rfc822");
intent.putExtra(Intent.EXTRA_EMAIL, new
String[]{"mail@sanandres.com"});
intent.putExtra(Intent.EXTRA_SUBJECT, "Asunto");
intent.putExtra(Intent.EXTRA_TEXT, "Contenido del mensaje");
```

46

Ejecutar una acción (7)

Tras crear el *Intent* estableciendo todos sus atributos, tendremos que comprobar si hay alguna aplicación capaz de realizar la acción. Para ello tenemos dos alternativas:

- Utilizar el método *resolveActivity*, que devolverá *null* cuando no haya ningún componente que pueda realizar la acción.
- Iniciar el *Intent* dentro de un bloque *try* y capturar la excepción que se produce cuando no hay un componente disponible.

47

Ejecutar una acción (8)

Una vez realizada la comprobación podemos iniciar el *Intent* directamente o permitir que el usuario pueda elegir la aplicación utilizando el método *createChooser*.

48

Ejecutar una acción (9)

Método 1: *resolveActivity*

```
PackageManager pm = getPackageManager();
ComponentName component = intent.resolveActivity(pm);
if (component != null) {
  startActivity(Intent.createChooser(intent, "Accion"));
} else {
  String mensaje = "No hay ninguna aplicación disponible";
  int duracion = Toast.LENGTH_SHORT;
  Toast alerta = Toast.makeText(getApplication(), mensaje, duracion);
  alerta.show();
}
```

49

Ejecutar una acción (10)

Método 2: *Excepción*

```
try {
  startActivity(Intent.createChooser(intent, "Accion"));
} catch (android.content.ActivityNotFoundException ex) {
  String mensaje = "No hay ninguna aplicación disponible";
  int duracion = Toast.LENGTH_SHORT;
  Toast alerta = Toast.makeText(getApplication(), mensaje, duracion);
  alerta.show();
}
```

50