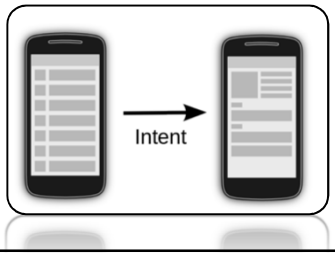
Navegación entre pantallas (1)

Las aplicaciones Android están compuestas de varias actividades (pantallas), y la navegación entre ellas es posible gracias a los *Intent* que se encargarán de invocar a la nueva actividad, suministrarle los datos necesarios y obtener la respuesta una vez que termine.



16

Navegación entre pantallas (2)

A la hora de invocar a otra actividad podemos encontrarnos con dos escenarios que dan lugar a dos tipos de *Intent*:

- ***Intent* explícito**: conocemos la actividad que se debe ejecutar.
- ***Intent* implícito**: no conocemos la actividad que se debe ejecutar pero sí el tipo de acción que se debe realizar.

17

Navegación entre pantallas (3)

Iniciar una nueva actividad

Para iniciar una nueva actividad hemos de crear un objeto *Intent*, cuyo constructor recibe el contexto y la clase de la actividad que deseamos iniciar:

```
Intent (Context packageContext, Class<?> activityToLaunch)
```

A continuación llamaríamos al método *startActivity* pasándole como parámetro el intent que hemos creado.

18

Navegación entre pantallas (4)

Iniciar una nueva actividad

La llamada quedaría así:

```
Intent intento = new Intent(this, NuevaActividad.class);
startActivity(intento);
```

No debemos olvidar agregar la nueva actividad al manifiesto de la aplicación, dentro del bloque "*application*":

```
<application
   <!-- ... -->
   <activity android:name=".DetailActivity" />
   <!-- ... -->
</application>
```

19

Navegación entre pantallas (5)

Paso de datos entre actividades

Podemos suministrar datos a la nueva actividad utilizando el método *putExtra* con una pareja clave/valor sobre nuestro *Intent*.

```
Intent intent = new Intent(this, NuevaActividad.class);
intent.putExtra("nombre", "Juan");
intent.putExtra("edad", 25);
startActivity(intent);
```

Este mecanismos sirve para cualquier tipo básico de Java: *int*, *float*, *double*, *string*…

20

Navegación entre pantallas (6)

Recoger los datos

- Método 1: obtenemos la referencia al *Intent* con *getIntent*, y a partir de él utilizamos *hasExtra* para comprobar si existe una clave y un método *get* acorde al tipo para recogerlo: *getIntExtra*, *getStringExtra*, etc.

```
Intent intent = getIntent();
if (intent != null) {
    if (intent.hasExtra("nombre")) {
        String user = intent.getStringExtra("nombre");
    }
    if (intent.hasExtra("edad")) {
        // En valores numéricos y booleanos tendremos que establecer
        // un valor por defecto como segundo parámetro
        int edad = intent.getIntExtra("edad",0);
    }
}
```

21

Navegación entre pantallas (7)

Recoger los datos

- Método 2: creamos un objeto *Bundle* en el que recogemos todos los extras mediante *getExtras*, y a partir de él obtenemos el dato deseado utilizando los métodos *get*: *getInt*, *getString*, etc.

```
Bundle extras = getIntent().getExtras();
if (extras != null) {
    String nombre = extras.getString("nombre");
    int edad = extras.getInt("edad");
}
```

22

Navegación entre pantallas (8)

Obtener un resultado

Una actividad puede iniciar otra para realizar una operación y después recoger el resultado obtenido en esa operación.

Para ello, debemos iniciar la actividad mediante *startActivityForResult*, que además del *Intent* recibe como segundo parámetro un entero que permite identificar el resultado obtenido. Si este valor es negativo la llamada equivale a *startActivity*.

```
startActivityForResult (Intent intent, int requestCode);
```

23

Navegación entre pantallas (9)

Obtener un resultado

```
private static final int INTENT_RESULT = 123;

@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);

  /* ... */

  Intent nuevoIntent = new Intent (Activity1.this, Activity2.class);
  startActivityForResult(nuevoIntent, INTENT_RESULT);

  /* ... */

}
```

24

Navegación entre pantallas (10)

Obtener un resultado

A continuación, hemos de sobrecargar el método *onActivityResult*:

```
onActivityResult (int requestCode,
     int resultCode, Intent data);
```

- *requestCode*: permite identificar el origen del resultado (identificador de *startActivityForResult*).
- *resulCode*: tipo de resultado establecido en la actividad invocada mediante *setResult*. Por defecto es *RESULT_OK* o *RESULT_CANCELED*, aunque podemos crear nuestros propios valores.
- *data*: *Intent* que contiene los datos del resultado.

25

Navegación entre pantallas (11)

Obtener un resultado

```
@Override
protected void onActivityResult (int requestCode, int resultCode,
Intent data) {
  super.onActivityResult(requestCode, resultCode, data);
  if (requestCode == INTENT_RESULT) {
    if (resultCode == RESULT_OK) {
      String nombre = data.getStringExtra("nombre");
      /* ... */
    }
  }
}
```

26

Navegación entre pantallas (12)

Obtener un resultado

Por último, en la actividad invocada tras realizar la operación se deberá establecer el resultado mediante *setResult* y después llamar al método *finish* para finalizar la actividad:

```
setResult(RESULT_OK);
finish();
```

Pulsar el botón atrás del móvil en la segunda actividad se corresponde con una llamada a *setResult (RESULT_CANCELED)*.

27

Navegación entre pantallas (13)

Obtener un resultado

También podemos devolver datos adicionales a la actividad inicial.

Para ello, en la actividad invocada crearemos un *Intent*, almacenaremos en él los datos que queramos devolver mediante *putExtra* y devolveremos el *Intent* en *setResult*.

```
Intent resultIntent = new Intent ();
resultIntent.putExtra("nombre", nombre);
setResult(RESULT_OK, resultIntent);
finish();
```

28

Navegación entre pantallas (14)

Paso de objetos: Parcelable

Como hemos visto se pueden pasar datos entre actividades mediante **Bundle**, pero este método tiene la limitación de que sólo se puede utilizar con los tipos básicos (*int*, *float*, *String*, etc).

Cuando queremos pasar tipos más complejos (objetos), Java ofrece la interfaz *Serialize.*

Sin embargo este método resulta muy lento en Android y por eso desarrolló ***Parcelable***.

29

Navegación entre pantallas (15)

Paso de objetos: Parcelable

En primer lugar debemos implementar la interfaz *parcelable* en la clase de aquellos objetos que deseamos que se puedan pasar entre actividades.

```
public class User {
  private String nombre;
  private String pass;
  public User (String nombre, String pass) {
    this.nombre = nombre;
    this.pass = pass;
  }
  public String getNombre() { return nombre; }
  public String getPass()   { return pass;   }
}
```

30

Navegación entre pantallas (16)

Paso de objetos: Parcelable

Esta interfaz agregará dos métodos:

- *describeContents*: sirve para describir el contenido del *Parcel* si contiene objetos especiales como descriptores de archivo. Como no es el caso, devolvemos 0.
- *writeToParcel*: sirve para escribir el contenido del objeto en el *Parcel*.

https://developer.android.com/reference/android/os/Parcelable.html

31

Navegación entre pantallas (17)

Paso de objetos: Parcelable

```
public class User implements Parcelable {
  private String nombre;
  private String pass;

  public User (String nombre, String pass) {
    this.nombre = nombre;
    this.pass = pass;
  }

  public String getNombre() { return nombre; }

  public String getPass()   { return pass;   }

  @Override
  public int describeContents() { return 0; }

  @Override
  public void writeToParcel(Parcel parcel, int i) {
    parcel.writeString(nombre);
    parcel.writeString(pass);
  }
}
```

32

Navegación entre pantallas (18)

Paso de objetos: Parcelable

A continuación creamos un objeto **CREATOR** que nos permitirá crear nuestro objeto a partir de un *Parcel*.

- Debe ser *public static*, llamarse *CREATOR* y su tipo se especifica a partir del nombre de la clase que implementa Parcelable.
- *createFromParcel*: permite crear un objeto desde un *Parcel*.
- *newArray*: permite crear un array de objetos.

33

Navegación entre pantallas (19)

Paso de objetos: Parcelable

```
public class User implements Parcelable {
  ...
  public static final Parcelable.Creator<User> CREATOR =
  new Parcelable.Creator<User>() {
    @Override
    public User createFromParcel (Parcel source) {
      return new User (source);
    }

    @Override
    public User[] newArray (int size) {
      return new User[size];
    }
  };
}
```

34

Navegación entre pantallas (20)

Paso de objetos: Parcelable

Por último añadimos un constructor en nuestra clase que reciba un *Parcel* y asigne los valores a cada uno de los atributos en el mismo orden en que se escribieron en el método *writeToParcel*.

```
public User (Parcel source) {
    this.nombre = source.readString();
    this.pass = source.readString();
}
```

35

Navegación entre pantallas (21)

Paso de objetos: Parcelable

Una vez implementada la interfaz *Parcelable* en nuestra clase, ya podemos enviar objetos de esa clase con el método *putExtra*.

```
String user =
  ((EditText)findViewById(R.id.editUser)).getText().toString().trim();
String pass =
  ((EditText)findViewById(R.id.editPass)).getText().toString().trim();
User usuario = new User (user, pass);
Intent intent = new Intent (MainActivity.this, Activity2.class);
intent.putExtra("usuario", usuario);
startActivity(intent);
```

36

Navegación entre pantallas (22)

Paso de objetos: Parcelable

En la actividad de destino se recogerá el objeto mediante *getParcelableExtra*.

```
Intent intent = getIntent();
User usuario = intent.getParcelableExtra("usuario");
TextView txtUser = (TextView)findViewById(R.id.txtUser2);
txtUser.setText(usuario.getNombre());
TextView txtPass = (TextView)findViewById(R.id.txtPass2);
txtPass.setText(usuario.getPass());
```

37