Tema 1.2.- Desarrollo de la aplicación

2.1. La arquitectura de una app

Una aplicación móvil para Android se construye a partir de cuatro elementos principales: las *activities* y su ciclo de vida, los *intents*, el *document manifest* y los *resources*. Antes de empezar a desarrollar una aplicación, hay que tener muy claros estos conceptos.

1. *Activity*: es una ventana que contiene la interfaz de usuario de nuestra aplicación. Una aplicación debe estar formada por una o más *activities*. Las *activities* tienen un ciclo de vida definido por diferentes métodos. Es muy importante entender el ciclo de vida de las *activities* para programar bien una aplicación. Estos son los métodos que definen el ciclo de vida de una *activity*.
    - onCreate(): se llama cuando se crea la *activity*.
    - onStart(): se llama cuando la *activity* es visible para el usuario.
    - onPause(): se llama cuando se pausa la *activity*. Luego se puede volver a llamar.
    - onStop(): se llama cuando la *activity* deja de ser visible para el usuario.
    - onDestroy(): se llama justo antes de que la *activity* sea destruida por el sistema.
    - onRestart(): se llama cuando la *activity* se ha detenido y se vuelve a reiniciar.

    Desde el punto de vista del desarrollador tenemos que tener claro lo siguiente:

    - onStart() y onRestart(): En general no es necesario usarlo. Podemos usar onResume directamente.
    - onStop() y onDestroy(): No es aconsejable escribir código aquí. Las apps pueden ser detenidas por el sistema de forma silenciosa y puede que no se ejecute este evento. Es mejor usar onPause().
    - onCreate(): Se llama cuando se crea la *activity*. Aquí podemos crear Ventanas, elementos UI, inputs, etc.
    - onResume(): Iniciamos o reiniciamos el hilo.
    - onPause(): Detenemos el *loop* Main y salvamos a disco el estado (en la tarjeta SD). Luego se puede volver a llamar. En este evento podemos comprobar si nuestra *Activity* ha sido eliminada llamando a la función *isFinishing()*.
2. *Intents*: los *intents* ofrecen un servicio de paso de mensajes para interconectar las *activities*.
3. *Manifest*: el *manifest* es un fichero XML único en cada aplicación. Este fichero define varias características de la aplicación: la API con la que trabaja la aplicación, los permisos de acceso al sistema, cuál es la *activity* inicial con la que empieza la aplicación, si la aplicación puede trabajar en modo vertical y horizontal, etc
4. *Resources:* los *resources* son los ficheros adicionales y el contenido estático que se usa en la aplicación.

2.2. Creación de la primera app

Los pasos que hay que seguir para crear la primera aplicación, que será una aplicación sencilla que solo mostrará un texto, se pueden dividir en tres.

1. Crear un proyecto Android en Android Studio. Se abre el Android Studio y nos vamos al menú para crear un nuevo proyecto. Hay que definir unas características iniciales: el nombre de la aplicación, el nombre del proyecto, el nombre del paquete, la API mínima que hay que tener instalada en el dispositivo para ejecutar la aplicación, la API que se usará para el desarrollo y algunas otras opciones de diseño. Android Studio ya crea de manera automática todo el sistema de ficheros necesarios para la aplicación.. Además, cuando se genera el proyecto también se crea automáticamente la *activity* principal con la que se abrirá la aplicación.
2. Escribir el código. Una vez creado el proyecto Android hay que localizar la clase Java que se ha creado de manera automática. Esta es la clase que contiene la *activity*

principal y la que se abre cuando se ejecuta la aplicación. En esta clase, escribimos el código de la primera aplicación.

3. Ejecutar la aplicación. En la interfaz de Android Studio hay que hacer clic en el botón Run (triángulo verde en la barra de herramientas). El IDE ya compila y ejecuta la aplicación en el mismo paso. El proceso de la ejecución de la aplicación variará según si se hace en un dispositivo virtual (emulador) o un dispositivo físico (*smartphone* o tableta).

2.3. Diseño

El primer paso para crear una app es pensar y plantear el diseño de la aplicación. Un buen diseño de una aplicación puede ser igual o más importante que la propia implementación.

Cuando una aplicación es simple, el diseño de la aplicación es fácil. Sin embargo, cuando las aplicaciones son más complejas nos surge la duda de ¿cómo se decide cuál es el mejor componente para navegar entre pantallas?

Android ofrece multitud de estructuras y componentes gráficos para el diseño de aplicaciones, y con el tiempo se van añadiendo más.

El siguientes enlace permite acceder a la documentación de Google con algunos ejemplos de código y algunos con explicaciones de ejemplos que se pueden descargar directamente: https://developer.android.com/guide/?hl=es Un vez en el enlace, en el lateral izquierdo accede al apartado Core Topics y dentro de este ve navegando.

2.3.1. Ficheros XML

El entorno gráfico de las aplicaciones se define con ficheros XML. En cada fichero XML se crea el diseño de una pantalla de la aplicación, lo que se denomina una vista (*view*). Una vista es el elemento básico de la interfaz de usuario. Una vista ocupa un área rectangular de la pantalla y es la responsable de la elaboración y gestión de eventos. **Es la clase base para crear los componentes de la interfaz de usuario**.

La vista se declara en la clase Java, pero XML define cada uno de los componentes gráficos de la aplicación. Hay elementos gráficos que también se pueden definir en la clase Java, pero es preferible usar XML, ya que así queda separada la estructura del código.

2.3.2. *Layouts*

Un *layout* es un contenedor de una o más vistas que gestiona su comportamiento. El *layout* desciende de la clase*View* y puede contener múltiples *layouts*. Dentro del *layout* se definen los elementos gráficos de la vista. Según la distribución de estos elementos, encontramos distintos *layouts*. Los más comunes son los siguientes:

- *Linear layout*: alinea todos los elementos de la vista en una misma dirección.
- *Relative layout*: dispone los elementos en relación con el padre u otro elemento.
- *Web view*: muestra páginas web.

Algunos *layouts* cambian de manera dinámica, ya que su vista depende del contenido:

- *List view*: muestra una lista con *scroll*.
- *Grid view*: muestra una cuadrícula con *scroll*.

2.3.3. Menús

Los menús son un elemento de la interfaz que se utiliza para presentar diferentes opciones. Hay tres tipos de menús.

Menú de opciones y *action bar*: este menú es el principal de la *activity* e incluye las opciones importantes de la misma. En las versiones Android 2.3.x o inferiores, el menú aparece en la parte inferior de la pantalla. En versiones 3.x o superiores el menú aparece en la *action bar*.

- Menú contextual: se utiliza para acceder a opciones de un solo elemento.
- Menú *popup*: es un menú desplegable anclado a una vista.

2.3.4. Tabs

Las *tabs* se usan para acceder a diferentes vistas. Las *tabs* pueden ser fijas, cuando muestran todos los elementos y solo hay que presionar para cambiar de vista, o con *scroll*, cuando solo se muestran algunos elementos y hay que hacer *scroll* para cambiar de vista.

Las *tabs* también se pueden usar para crear *swipe views*, navegación lateral entre actividades.

2.3.5. Controles UI

Los elementos de la interfaz de usuario muestran la información por pantalla. Los más usados son los siguientes.

- *TextView*: subvista que muestra texto en la pantalla.
- *ImageView*: subvista que muestra una imagen o icono.
- *ProgressBar* e indicadores de actividad: indicador visual del progreso de una operación.

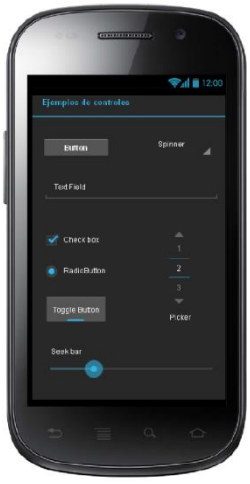
2.3.6. Controles y eventos de entrada

Los controles y eventos de entrada permiten introducir información por parte del usuario. Es la parte clave en la interacción entre dispositivo y usuario.

2.3.6.1. Controles

Los controles son la parte interactiva de la interfaz de usuario: permiten que el usuario introduzca información en la aplicación, seleccione opciones o fechas, etc. El catálogo de controles disponibles es el siguiente.

- *Button*: elemento que muestra un texto o imagen que conecta a una acción cuando el usuario hace clic en el botón.
- *Text field*: elemento que permite introducir texto en la aplicación.
- *Spinner*: desplegable que permite seleccionar un ítem entre varios.
- *Checkbox*, *RadioButton*, *ToggleButton*: elementos que permiten seleccionar una o más opciones de un conjunto. En el caso de *ToggleButton*, permite cambiar un elemento entre dos estados.
- *Picker*: control que permite al usuario elegir una fecha o un tiempo.
- *Seek bar*: elemento deslizante que permite seleccionar un valor de un rango continuo o discreto.



En la siguiente de la derecha, podemos ver un ejemplo de un diálogo de una aplicación Android donde se muestran algunos de estos controles. A la hora de diseñar una aplicación, es importante elegir el tipo de control más apropiado para el tipo de información que deba introducir el usuario, por ejemplo un *check box* para elegir entre valores sí/no o un *seek bar* para desplazarse por un vídeo.

2.3.6.2. Eventos

Los eventos en Android suelen ser generados en respuesta a una acción externa, normalmente en una interacción con la pantalla táctil. Estos eventos se denominan eventos de entrada. Android ofrece varias maneras de interceptar los eventos de entrada.

Encontramos los siguientes.

- *Event listeners*: objetos que reciben la notificación cuando un evento pasa.

- *Event handlers*: objeto que maneja el evento. Hay diferentes tipos de *event handlers*: *OnClick*, *OnLongClick*, *OnFocusChange*, *OnKey*, *OnTouch*, *OnMenuItemClick*. Cada uno de estos *event handlers* tiene un *event listener* que recibe la acción:
  - *OnClickListener*: se llama cuando el usuario hace clic en un botón, texto o imagen.
  - *OnLongClickListener*: se llama cuando el usuario hace clic en un botón, texto o imagen durante más de un segundo.
  - *OnFocusChangeListener*: se llama cuando un elemento pierde su enfoque.
  - *OnKeyListener*: se llama cuando el usuario pulsa o libera una tecla de hardware, como puede ser el teclado.
  - *OnTouchListener*: se llama cuando el usuario pulsa o libera una tecla de hardware o toca la pantalla.
  - *OnMenuItemclickListener*: se llama cuando el usuario pulsa una opción de un menú.

Los eventos no son triviales de implementar, ya que hay que definir el *event handler* y el *event listener* junto con el elemento gráfico sobre el que pasa el evento.

2.3.7. Mensajes de la aplicación

La interacción entre la aplicación y el usuario es constante. Por tanto, a veces es necesario que la aplicación lance mensajes de notificación al usuario, ya sea para confirmar una acción o para avisarle de una acción en concreto. Para esto, Android dispone de tres componentes.

- Notificaciones: son mensajes que se pueden enviar al usuario fuera de la interfaz de usuario de la aplicación.
- Diálogos: son pequeñas ventanas que muestran información extra al usuario o le piden que introduzca información.
- *Toast*: son pequeños *popups* que muestran información al usuario.

La imagen de la derecha muestra ejemplos de notificaciones, diálogos y *toast* en una aplicación Android. Nuevamente, para llevar a cabo un buen diseño de la aplicación Android es necesario elegir el tipo de mensaje más apropiado para el contenido que queremos comunicar. Es importante tener en cuenta el volumen de información que hay que comunicar, si esta información puede comunicarse en segundo plano o si, por el contrario, requiere de una respuesta inmediata por parte del usuario.