netmind

WeKnowIT

JJD310 – Desarrollo de Apps para Android 5 Lollipop

Material práctico

Práctica 1 Desarrollo de Apps para Android 5 Lollipop

El entorno de desarrollo

En este caso práctico se trabajarán los conceptos aprendidos en el tema 3 "El entorno de desarrollo: Android Studio" del material didáctico proporcionado por *net**mind***.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android desde 0. Durante el proceso, se examinarán las ventanas y comandos más relevantes del entorno de desarrollo Android Studio.

Objetivo

Examinar y conocer con profundidad el entorno de desarrollo Android Studio, así como la organización de ficheros, recursos y carpetas de un proyecto Android. Además, se introduce el emulador Genymotion, recomendado para el desarrollo de aplicaciones.

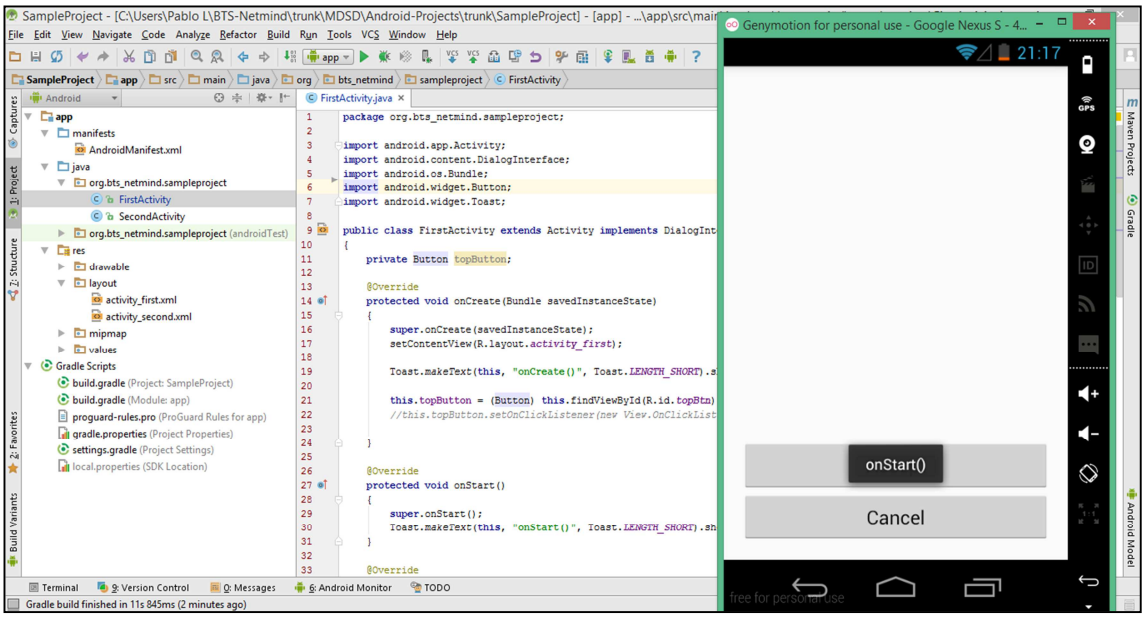
Contenidos

- Presentación del *IDE Android Studio*
- Creación de un proyecto Android
- Ejecución en el emulador *Genymotion*

Planteamiento del caso

La aplicación a implementar es tremendamente sencilla, y su creación no requerirá escribir ni una sola línea de código. El layout de su única Activity albergará un TextView con el mensaje "Hello world!". La idea no es programar una aplicación, sino examinar y entender las herramientas que un desarrollador tiene a su disposición para esta labor.

Resultado final



Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "SampleProject" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/SampleProject

Al finalizar la práctica has aprendido…

En este caso práctico se ha introducido, examinado y descrito el entorno de desarrollo (Android Studio IDE). Se ha realizado un análisis detallado de la estructura de un proyecto Android, con especial atención hacia algunos elementos como el *Android Manifest* y los scripts de compilación de *Gradle*.

Práctica 2 Desarrollo Avanzado de Apps para Android 5 Lollipop

Activity

En este caso práctico se trabajarán los conceptos aprendidos en el tema 4 "Componentes de una App I - Activity" del material didáctico proporcionado por *netmind*.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que haga uso de varios elementos Activity que interactúen entre ellas, proporcionando una correcta experiencia de usuario.

Objetivo

Conocer, estudiar y emplear con soltura los elementos Activity que conforman una aplicación. El buen entendimiento del ciclo de vida de un objeto Activity, así como de la gestión de su estado son fundamentales para proporcionar una correcta experiencia de usuario. Por último, se presta una especial atención a la implementación de la comunicación entre elementos Activity, y entre estas y otros componentes típicos del desarrollo Android (Intents, BroadcastReceivers, etc.).

Contenidos

- El ciclo de vida de una Activity y el almacenamiento de su estado
- Gestión de una Activity ante cambios de configuración
- Procedimiento para lanzar y finalizar una Activity
- Comunicación entre Activity/s

Planteamiento del caso

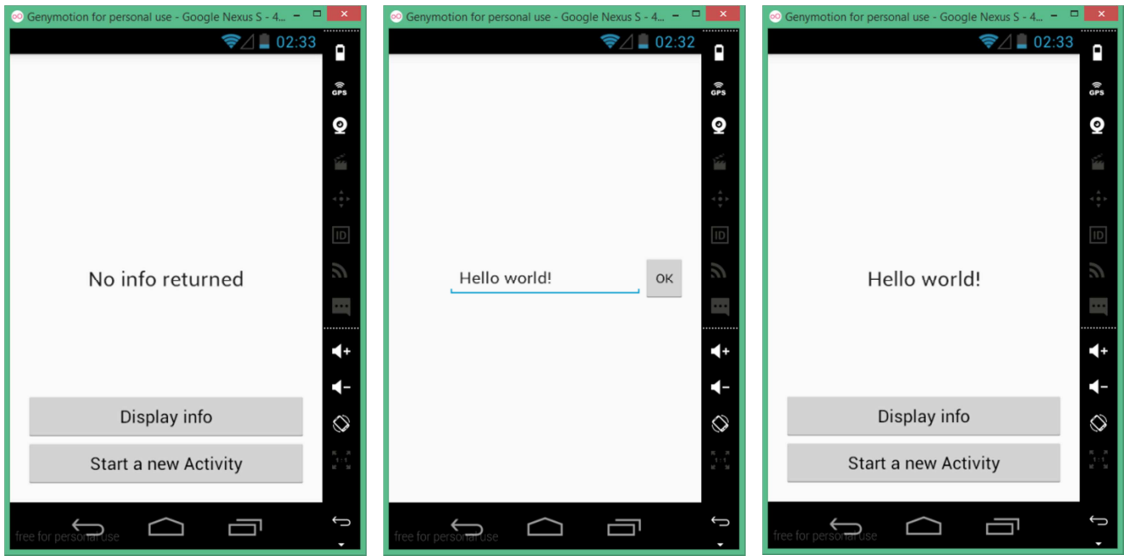
La aplicación a implementar consta de dos objetos Activity. El layout de la primera incluye un TextView y dos botones, mientras que el de la segunda aloja un EditText y un botón. El objetivo es programar la correcta comunicación entre ambos elementos, de forma que al presionar el botón de la segunda, se actualiza el TextView de la primera con la información introducida en el EditText. Además, se presta especial atención a la gestión del estado de la Activity, de forma que un cambio de configuración del terminal (rotación de pantalla, por ejemplo) no suponga la pérdida de la posible información introducida por el usuario (texto del TextView, por ejemplo).

Por último, es interesante remarcar que es conveniente, a título ilustrativo, solapar (*override*) todos los métodos de los objetos Activity (o al menos uno) a fin de mostrar un mensaje por pantalla cuando se acceda a ellos. De esta forma el alumno podrá entender más fácilmente el ciclo de vida de una Activity.

Ej.:

```
@Override
protected void onStop()
{
    super.onStop();
    Toast.makeText(this, "Main Act onStop()", Toast.LENGTH_SHORT).show();
}
```

Resultado final



Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "ActivityManager" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/ActivityManager

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los fundamentos de uno de los principales componentes de aplicación de la plataforma Android, la Activity. Este elemento se asocia habitualmente como una pantalla de la aplicación; es decir, cada interfaz que se presenta al usuario tiene asociada una Activity que es responsable de implementar su funcionalidad.

Práctica 3 Desarrollo Avanzado de Apps para Android 5 Lollipop

Diseño básico GUI

En este caso práctico se trabajarán los conceptos aprendidos en el tema 5 "Diseño básico GUI y gestión de layouts" del material didáctico proporcionado por *net**mind***.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que integre múltiples elementos gráficos típicos de la plataforma Android. No se pretende, por tanto, conseguir una aplicación funcional y coherente, sino dar una pincelada de las diferentes alternativas con las que se cuenta a la hora del diseño de interfaces gráficas.

Objetivo

Conocer, estudiar e implantar los principios de diseño de interfaz gráfica en Android. La idea es presentar los elementos más relevantes a incluir en la aplicación, así como la gestión de estilos (*styles* y *themes*) para dar un aspecto visual atrayente a ciertas partes del layout.

Contenidos

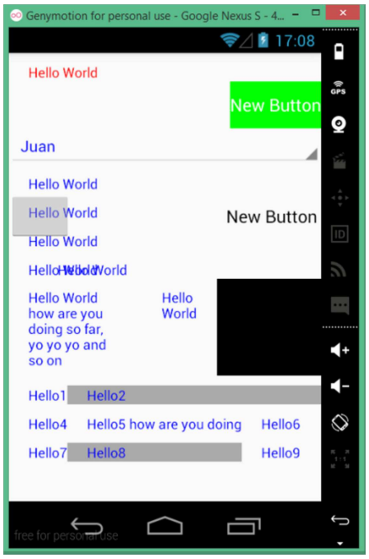
- Layouts y diferentes alternativas
- Containers y listas de elementos
- Widgets de especial relevancia: Button y Spinner
- Styles y Themes

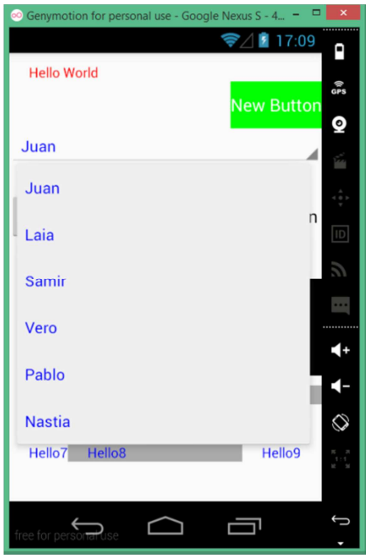
Planteamiento del caso

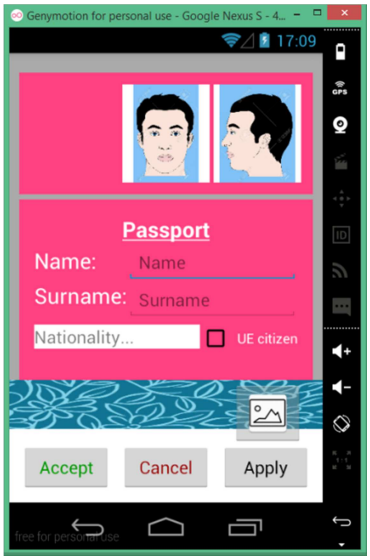
La aplicación a implementar albergará numerosos elementos gráficos típicamente utilizados a la hora de diseñar una interfaz de usuario en Android. Además, se propondrá un ejercicio para que el alumno reproduzca fehacientemente un diseño de interfaz por su cuenta.

Resultado final

Las dos primeras capturas corresponden al ejercicio realizado conjuntamente con el formador durante la sesión práctica. La tercera imagen se refiere al ejercicio propuesto al alumnado.







Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "GUIDesign" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/GUIDesign

El ejercicio propuesto puede encontrarse en la carpeta "GUIDesignProposal" del mismo repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/GUIDesignProposal

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los fundamentos del diseño de la interfaz gráfica (UI) en Android. En esta plataforma es primordial separar la funcionalidad de la aplicación (en ficheros Java) de su aspecto (en ficheros XML).

Práctica 4 Desarrollo Avanzado de Apps para Android 5 Lollipop

Localización en Android

En este caso práctico se trabajarán los conceptos aprendidos en el tema 6 "Gestión de recursos y localización" del material didáctico proporcionado por *net**mind***.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que integre recursos (*resources*) que el sistema pueda utilizar en función del hardware del terminal y de la configuración del sistema operativo.

Objetivo

Conocer, estudiar e implantar los principios de localización en Android. La idea es presentar los procedimientos por los que una aplicación puede disponer de varios grupos de recursos que puedan ser utilizados automáticamente por el sistema en función de su configuración.

Contenidos

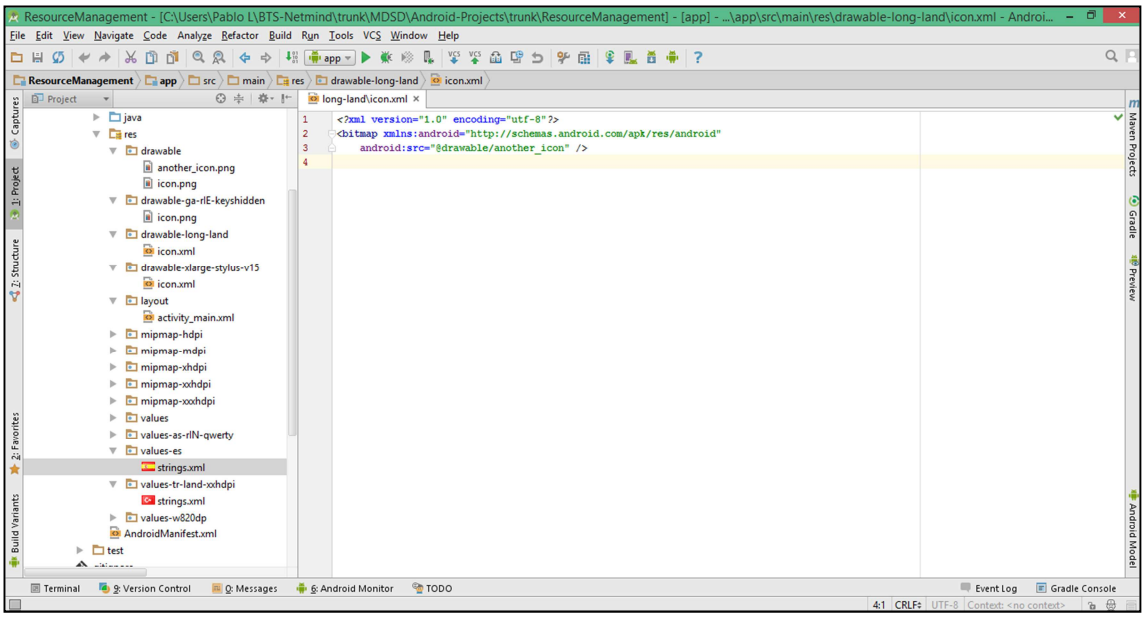
- Cómo proporcionar recursos a una aplicación
- Cómo acceder y utilizar recursos
- Cómo gestionar los cambios en tiempo de ejecución
- Cómo manejar recursos para múltiples configuraciones: *Locale*

Planteamiento del caso

La aplicación a implementar incluye varios grupos de recursos, almacenados en diferentes carpetas de acuerdo a distintos modificadores (*qualifiers*) que determinarán el aspecto de la misma.

Por tanto, es conveniente definir varios AVD en el emulador para apreciar así la forma en la que se visualiza la aplicación en función de las características hardware del dispositivo.

Resultado final



Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "ResourceManagement" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/ResourceManagement

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los conceptos relativos a recursos y su gestión en diferentes versiones de Android. Desde su nacimiento, esta plataforma se ha caracterizado por ofrecer una amplia variedad de dispositivos de distintos fabricantes, lo que ha hecho especialmente difícil la compatibilidad entre el sistema operativo y el hardware.

Práctica 5 Desarrollo Avanzado de Apps para Android 5 Lollipop

Intents y Notifications

En este caso práctico se trabajarán los conceptos aprendidos en el tema 7 "Intents, intent-filters y notificaciones" del material didáctico proporcionado por *netmind*.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que incluya notificaciones implementadas de diferentes formas. Este tipo de mensajes, que aparecen en la barra de estado del dispositivo, sirven para informar al usuario que una tarea se está llevando a cabo..

Objetivo

Conocer, estudiar y emplear con soltura las alternativas que ofrece la plataforma Android a la hora de crear y representar notificaciones. Concretamente se diferenciará entre notificaciones estándar y notificaciones con layout personalizado por el desarrollador. Además, se mostrará el uso del servicio de alarmas (AlarmManager) mediante Pending Intent.

Contenidos

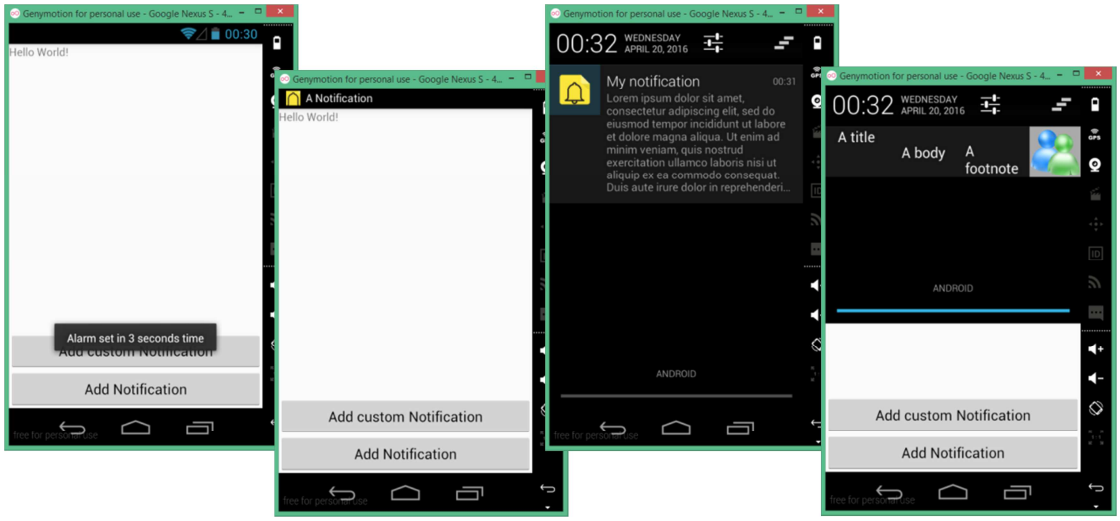
- Tipos de Intents
- Cómo recibir Intent implícito (intent-filter)
- Cómo crear una notificación
- Mostrando Notificaciones con Layout propio

Planteamiento del caso

La aplicación a implementar incluirá un layout con dos botones. Al pulsar el inferior, se mostrará una notificación con los elementos por defecto (icono, título y cuerpo). El botón superior lanzará una notificación con un layout previamente definido por el desarrollador.

Además, a los tres segundos de arrancar la aplicación se mostrará un mensaje (Toast) indicando que se ha disparado una alarma. Esta última característica puede parecer anecdótica, pero el uso de Pending Intents es de especial importancia cuando se quiere asociar una acción al pulsar sobre una notificación.

Resultado final



Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "NotificationManager" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/NotificationManager

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los conceptos relativos a Intents y Notificaciones.

Un Intent es un paquete de información, un elemento que expresa la «intención» de hacer algo, concretamente lanzar un componente de aplicación (Activity, Service o Broadcast Receiver).

Una notificación no es más que un breve mensaje que se muestra en la parte superior de la pantalla del terminal, en la barra de estado. Su función es informar al usuario de alguna eventualidad está ocurriendo durante la ejecución de la aplicación.

Práctica 6 Desarrollo Avanzado de Apps para Android 5 Lollipop

Service

En este caso práctico se trabajarán los conceptos aprendidos en el tema 8 "Componentes de una App II - Service" del material didáctico proporcionado por *net**mind***.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que incluya Services para realizar labores de procesamiento en background. La idea es ilustrar las diferentes posibilidades que ofrece la plataforma Android a la hora de acometer tareas "pesadas", de forma que la experiencia de usuario no se vea mermada.

Objetivo

Conocer, estudiar y aprender a implementar con soltura Services en Android. Analizar por qué los Services son necesarios y cuáles son los distintos tipos existentes.

Contenidos

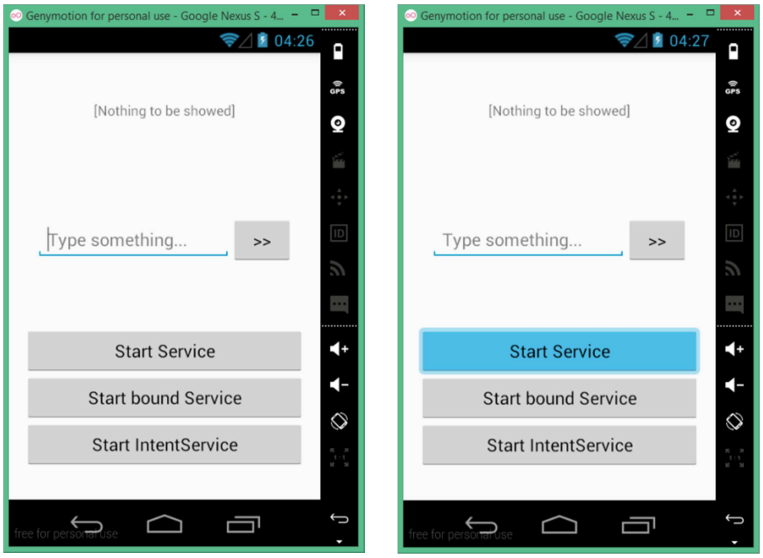
- Introducción a Service y tipos de Services
- IntentService: un Service independiente
- Comunicación entre Services

Planteamiento del caso

La aplicación a implementar consta de un layout con varios botones, un TextView y un EditText. Cuando el botón superior es pulsado, se arranca un Service en el que se simula una tarea "pesada" (en realidad se manda al hilo de procesamiento en cuestión a "dormir"). Sin embargo, durante este lapso de tiempo, la aplicación no responde a la interacción con el usuario, de forma que el EditText, por ejemplo, no puede utilizarse para actualizar el TextView.

Los dos botones siguientes lanzan también un Service pero en otro hilo de ejecución (*worker thread*). Un Intent Service se ejecuta en otro hilo por defecto, mientras que un Bound Service es un Service estándar al que la Activity en ejecución se puede "enganchar" (*bound*). Será labor del desarrollador que este Service se ejecute en otro hilo (usando un Thread, por ejemplo).

Resultado final



Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "ServiceManager" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/ServiceManager

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los conceptos relativos a un componente de aplicación, el Service. Este elemento permite realizar tareas en background en el sentido de no estar integradas en la Activity con la que el usuario está interactuando. Sin embargo, tanto una como otra son generalmente realizadas por el hilo principal de la aplicación (*UI thread*).

Práctica 7 Desarrollo Avanzado de Apps para Android 5 Lollipop

Broadcast Receiver

En este caso práctico se trabajarán los conceptos aprendidos en el tema 9 "Componentes de una App III – Broadcast Receiver & Content Provider" del material didáctico proporcionado por *netmind*.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que incluya varios Broadcast Receivers asociados a diferentes mensajes de difusión.

Objetivo

Conocer, estudiar y emplear con soltura los Broadcast Receivers, que son el tipo de elemento de aplicación que permiten "escuchar" mensajes de difusión, tanto del sistema como personalizados, y actuar en consecuencia. Analizar las situaciones en las que es conveniente usar este tipo de procedimientos y examinar la idoneidad de combinarlos con Services para agregar funcionalidades adicionales a la aplicación.

Contenidos

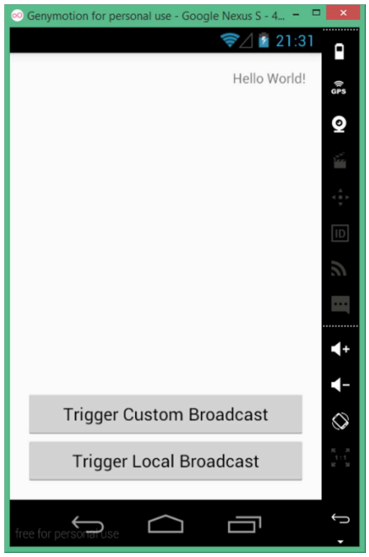
- Broadcasts del sistema
- Crear y registrar un Broadcast Receiver
- La clase *LocalBroadcastManager*
- Modificando intent-filters en el *Android Manifest*

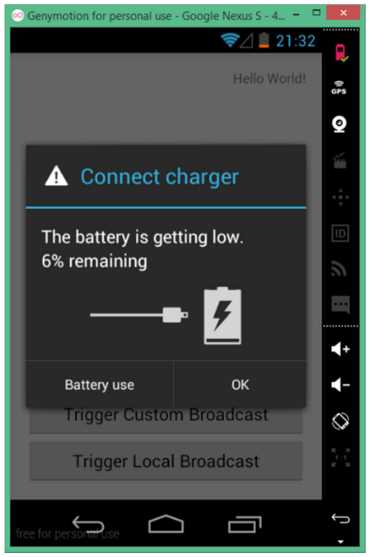
Planteamiento del caso

La aplicación a implementar incluye una única Activity y dos Broadcast Receivers. El primero de los Broadcast Receivers "escucha" al evento del sistema que se dispara cuando la batería es baja, mientras que el segundo está asociado a una acción propia definida por el desarrollador.

El layout de la aplicación incluye únicamente dos botones que disparan el evento personalizado, cuyo identificador es totalmente arbitrario (por ejemplo `"es.mi_android_broadcast.CUSTOM_INTENT"`), de dos formas distintas. El evento del sistema asociado a "batería baja" no necesita ser disparado. El identificador de la acción a indicar en el Manifest es `"android.intent.action.ACTION_POWER_CONNECTED"`.

Resultado final





Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "BroadcastReceiverManager" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/BroadcastReceiverManager

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los conceptos relativos a uno de los principales componentes de aplicación como es el Broadcast Receiver. La principal característica de este elemento es aportar funcionalidades adicionales a las aplicaciones al permitir su comunicación con el «resto del mundo».

Práctica 8 Desarrollo Avanzado de Apps para Android 5 Lollipop

Menús en Android

En este caso práctico se trabajarán los conceptos aprendidos en el tema 10 "Menús y Application Bar" del material didáctico proporcionado por *net**mind***.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que incluya varios de las alternativas que Android ofrece a la hora de presentar menús de opciones.

Objetivo

Conocer, estudiar y emplear con soltura los menús y la App Bar en Android. Analizar las situaciones en las que es conveniente usar este tipo de elementos y entender los procedimientos y buenas prácticas que permiten proporcionar una correcta experiencia de usuario en aplicaciones que incluyen menús de opciones.

Contenidos

- Definición de un menú en XML
- Menú de opciones, menús contextuales y menús pop-up
- La App Bar <Toolbar>

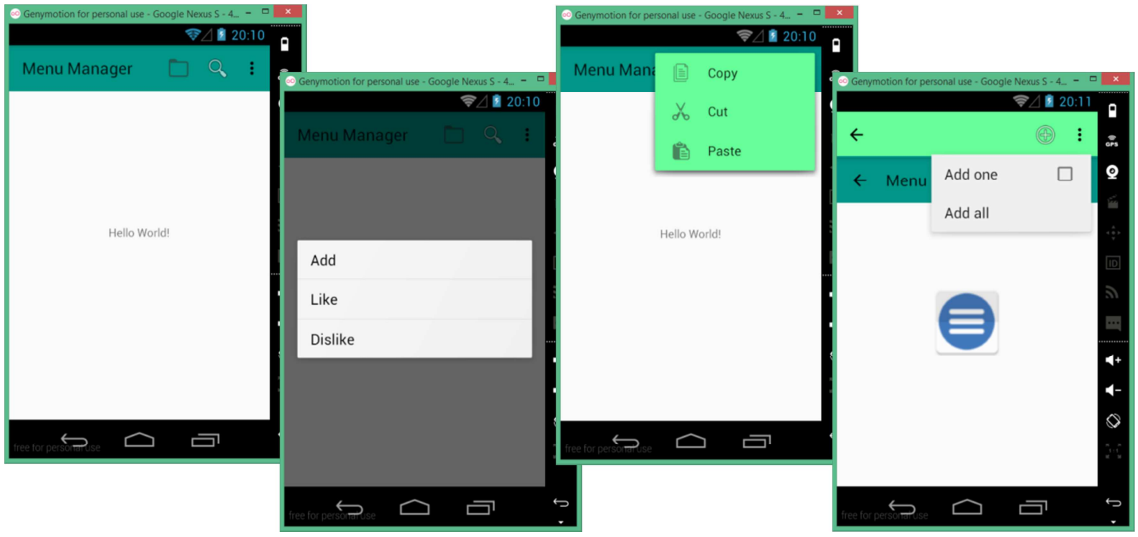
Planteamiento del caso

La aplicación a implementar incluye dos elementos Activity con layouts muy simples, aunque diferentes. En el primer caso la App Bar incluye la tradicional Action Bar, mientras que en el segundo alberga la novedosa Toolbar, más flexible y robusta.

Además, en lo referente a menús contextuales, el primer layout incluye un clásico menú contextual flotante, mientras que el segundo implementa lo que se conoce como *Contextual Action Mode*, y que convive con la Toolbar.

Por último, a una de las opciones del segundo layout se le ha agregado un pop-up menú, mostrando así una de las clásicas alternativas que hay en Android para presentar información adicional al usuario.

Resultado final



Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "MenuManager" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/MenuManager

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los conceptos relativos a los menús de opciones y acciones de una aplicación.

Es habitual en cualquier aplicación que el usuario disponga de una serie de elementos que le permitan acceder a menús de opciones y acciones para realizar operaciones en la aplicación. Esta práctica ha puesto de manifiesto las muchas alternativas que ofrece la plataforma Android a estos efectos.

Además, con la llegada de *Material Design*, nuevas «reglas de estilo» han aflorado, cambiando sensiblemente los patrones de diseño de los menús de aplicación.

Práctica 9 Desarrollo Avanzado de Apps para Android 5 Lollipop

Datos en Android

En este caso práctico se trabajarán los conceptos aprendidos en el tema 11 "Almacenamiento de datos" del material didáctico proporcionado por *netmind*.

La metodología a seguir consistirá en realizar, conjuntamente con el formador, una aplicación Android que incluya varios procedimientos para almacenar datos de usuario y de la propia configuración de la aplicación. El almacenamiento de datos en sin duda uno de los aspectos más importantes a considerar por el desarrollador; una correcta y sencilla gestión conllevará una grata experiencia de usuario y un código coherente y fácil de leer.

Objetivo

Conocer, estudiar y emplear con soltura los frameworks que ofrece Android para el almacenamiento de datos. La idea es explorar e implementar, conjuntamente en una aplicación, todas las alternativas existentes en la plataforma.

Contenidos

- Shared Preferences
- Almacenamiento interno
- Almacenamiento externo y permisos
- Bases de datos SQL

Planteamiento del caso

La aplicación a implementar incluye un layout bastante sencillo, con una serie de botones, un EditText y un TextView.

La sección central de la pantalla sirve para mostrar el uso de las Shared Preferences. Al introducir un texto en el EditText y pulsar el botón anexo, se modifica el TextView superior (conviene usar expresiones regulares para evitar que el texto esté en blanco o solo con espacios). Si la aplicación se cierra completamente (las Shared Preferences se guardan en el método onStop() del ciclo de vida de la Activity) y se vuelve a iniciar, el TextView mostrará el último mensaje con que fue actualizado.

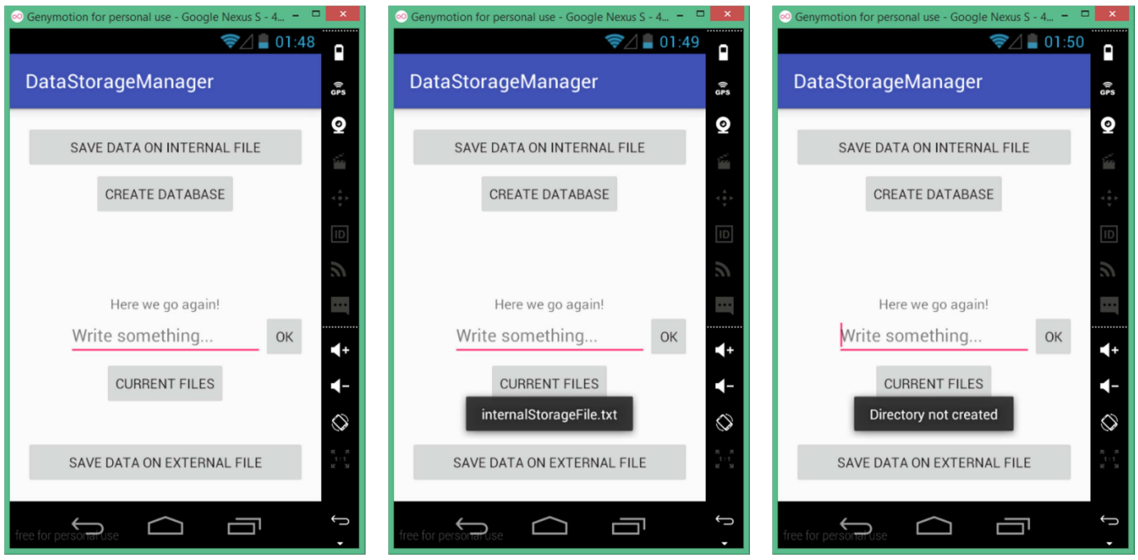
Si se rellena el EditText (sin ni si quiera actualizar el TextView) y se pulsa el botón superior del layout, el texto se almacena en un fichero de la memoria interna del dispositivo, concretamente en *internalStorageFile.txt*.

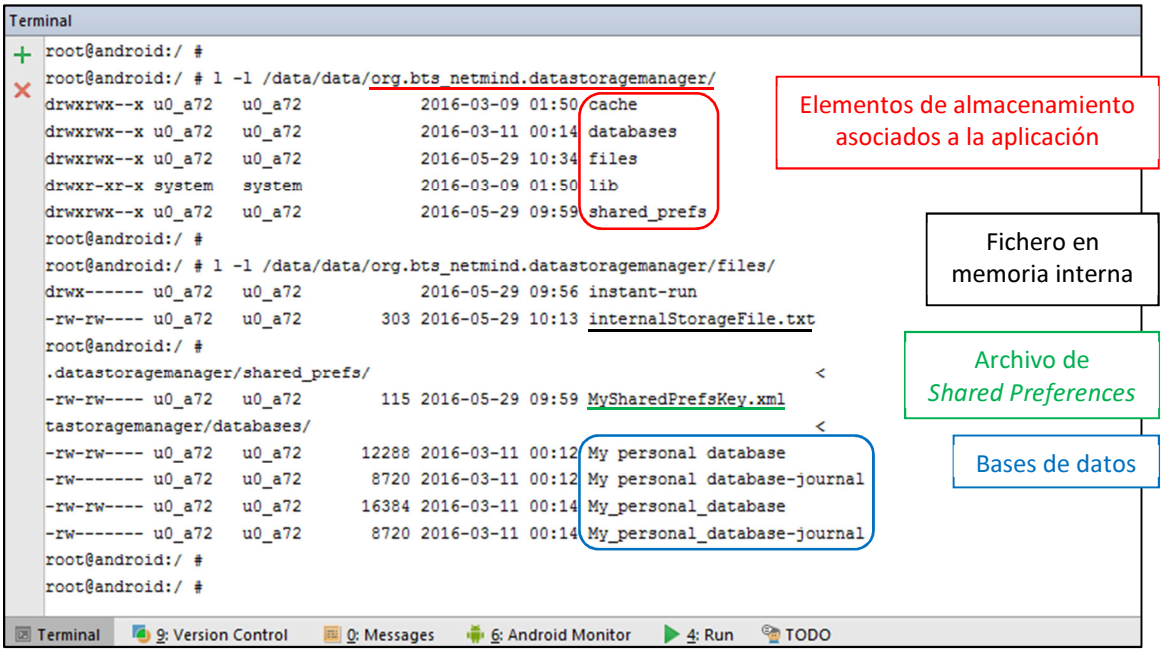
Un procedimiento similar al anterior se sigue cuando se pulsa sobre el botón inferior, pero esta vez el fichero se emplaza en la memoria externa del dispositivo, concretamente en el directorio público "Downloads".

El segundo botón simplemente crea una base de datos en caso de que no exista, pero no guarda ningún registro en ella. Es recomendable usar el "Terminal" de Android Studio para comprobar que la base de datos se crea al pulsar este botón.

Finalmente, el botón "CURRENT FILES" muestra por pantalla (Toast) una lista de los ficheros asociados a la aplicación (concretamente la lista asociada al Context de la aplicación y devuelta por el método *fileList()*).

Resultado final





Soluciones

El proyecto completo para Android Studio puede encontrarse en la carpeta "DataStorageManager" o en el siguiente enlace de repositorio de GitHub:

https://github.com/pablodeafsapps/Netmind-JJD310_AndroidProjects/tree/master/DataStorageManager

Al finalizar la práctica has aprendido…

En este caso práctico se han presentado y explicado los conceptos relativos al almacenamiento de datos en una aplicación Android. La plataforma ofrece muchas posibilidades a la hora de salvaguardar información de interés para el usuario en función del nivel de privacidad que se les quiera otorgar.

Control de cambios del documento

Versión	Fecha	Descripción	Autor
V.1.0	31/05/2016	Contenidos originales	Pablo L. Sordo Martínez