Ingeniería en Sistemas Computacionales

INSTITUTO TECNOLÓGICO SUPERIOR DE JEREZ
JEREZ DE GARCÍA SALINAS A 03 DE MAYO DEL 2019

NOMBRE:
GUADALUPE VÁZQUEZ DE LA TORRE

NUMERO DE CONTROL:
S17070158

CORREO:
guvadlt@Outlook.com

CARRERA:
INGENIERÍA EN SISTEMAS COMPUTACIONALES

NOMBRE DE LA MATERIA:
TOPICOS AVANZADOS DE PROGRAMACIÓN

CUARTO SEMESTRE

TEMA 5 - PROGRAMACIÓN DE DISPOSITIVOS MÓVILES

“ACTIVIDAD 2 -CUESTIONARIO Y MAPA CONCEPTUAL”

DOCENTE:
SALVADOR ACEVEDO SANDOVAL

1. **¿Cuál es el sufijo para las aplicaciones que se instalan en Android?**

Las herramientas de Android SDK compilan tu código, junto con los archivos de recursos y datos, en un APK: un paquete de Android, que es un archivo de almacenamiento con el sufijo .apk. Un archivo de APK incluye todos los contenidos de una aplicación de Android y es el archivo que usan los dispositivos con tecnología Android para instalar la aplicación.

2. **¿Cuáles son los 4 componentes que forman a una aplicación Android?**

   Los componentes de la aplicación son bloques de creación esenciales de una aplicación para Android. Cada componente es un punto diferente a través del cual el sistema puede ingresar a tu aplicación.

   Actividades: Una actividad representa una pantalla con interfaz de usuario, una actividad se implementa como una subclase de Activity.

   Servicios: Un servicio es un componente que se ejecuta en segundo plano para realizar operaciones prolongadas o tareas para procesos remotos. Un servicio no proporciona una interfaz de usuario. Un servicio se implementa como una subclase de Service.

   Proveedores de contenido: Un proveedor de contenido administra un conjunto compartido de datos de la app. Puedes almacenar los datos en el sistema de archivos. A través del proveedor de contenido, otras aplicaciones pueden consultar o incluso modificar los datos (si el proveedor de contenido lo permite). Un proveedor de contenido se implementa como una subclase de ContentProvider y debe implementar un conjunto estándar de API que permitan a otras aplicaciones realizar transacciones.

   Receptores de mensajes: Un receptor de mensajes es un componente que responde a los anuncios de mensajes en todo el sistema. Muchos mensajes son originados por el sistema. Un receptor de mensajes se implementa como una subclase de BroadcastReceiver y cada receptor de mensajes se proporciona como un objeto Intent.

3. **¿Como se "activan" dichos componentes?**

   Tres de los cuatro tipos de componentes (actividades, servicios y receptores de mensajes) se activan mediante un mensaje asincrónico llamado intent. Las intents enlazan componentes individuales en tiempo de ejecución ya sea que el componente le pertenezca a tu aplicación o a otra. Una intent se crea con un objeto Intent, que define un mensaje para activar un componente o un tipo específico de componente; una intent puede ser explícita o implícita, respectivamente.

   Existen métodos independientes para activar cada tipo de componente:

- Puedes iniciar una actividad (o asignarle algo nuevo para hacer) al pasar una Intent a startActivity() o startActivityForResult() (cuando quieras que la actividad devuelva un resultado).
- Puedes iniciar un servicio (o darle instrucciones nuevas a un servicio en curso) al pasar una Intent a startService(). O puedes establecer un enlace con el servicio al pasar una Intent a bindService().
- Puedes iniciar la transmisión de un mensaje pasando una Intent a métodos como sendBroadcast(), sendOrderedBroadcast(), o sendStickyBroadcast().
- Puedes realizar una consulta a un proveedor de contenido llamando a query() en un ContentResolver.

4. **¿Qué es el archivo MANIFEST y para qué sirve?**

Para que el sistema Android pueda iniciar un componente de la app, el sistema debe reconocer la existencia de ese componente leyendo el archivo AndroidManifest.xml de la app.

Sirva para:

- Identificar los permisos de usuario que requiere la aplicación
- Declarar el nivel de API mínimo requerido por la aplicación en función de las API que usa la aplicación.
- Declarar características de hardware y software que la aplicación usa o exige.
- Bibliotecas de la API a las que la aplicación necesita estar vinculada
- Etc.

5. **¿Cuáles son los estados en los que se puede encontrar una app?**

Una actividad en Android puede estar en uno de estos cuatro estados:

- Activa (Running): La actividad está encima de la pila, lo que quiere decir que es visible y tiene el foco.
- Visible (Paused): La actividad es visible pero no tiene el foco. Se alcanza este estado cuando pasa a activa otra actividad con alguna parte transparente o que no ocupa toda la pantalla. Cuando una actividad está tapada por completo, pasa a estar parada.
- Parada (Stopped): Cuando la actividad no es visible. El programador debe guardar el estado de la interfaz de usuario, preferencias, etc.
- Destruida (Destroyed): Cuando la actividad termina al invocarse el método finish(), o es matada por el sistema.

Cada vez que una actividad cambia de estado se van a generar eventos que podrán ser capturados por ciertos métodos de la actividad

6. **¿Cuáles son los métodos que permiten manipular dichos estados?**

- onCreate(Bundle): Se llama en la creación de la actividad. Se utiliza para realizar todo tipo de inicializaciones, como la creación de la interfaz de usuario o la inicialización de estructuras de datos. Puede

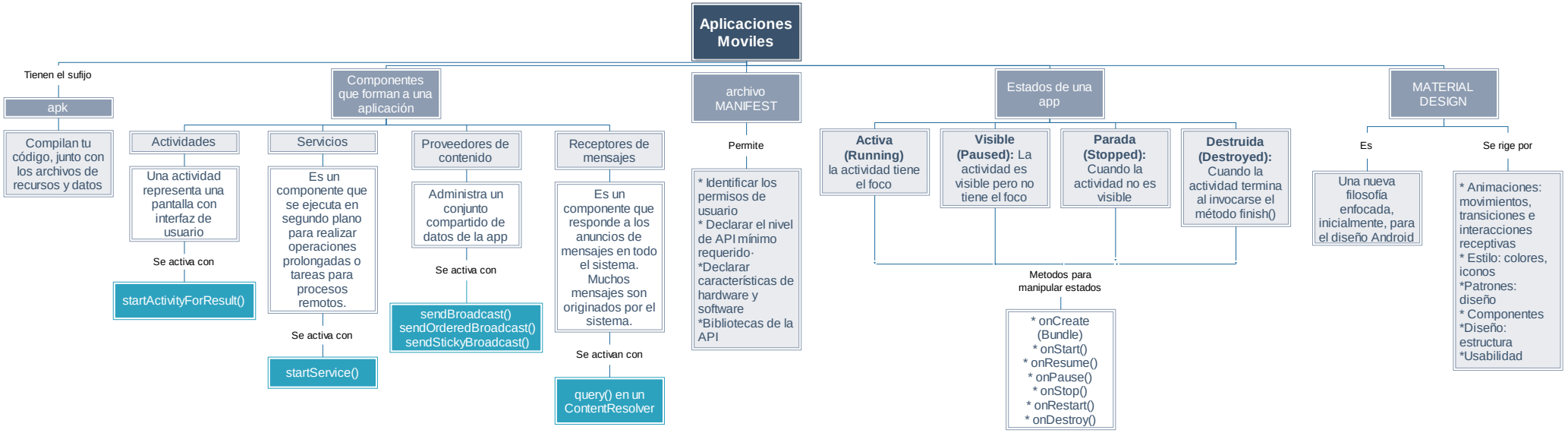
recibir información de estado de la actividad (en una instancia de la clase Bundle), por si se reanuda desde una actividad que ha sido destruida y vuelta a crear.

- onStart(): Nos indica que la actividad está a punto de ser mostrada al usuario.
- onResume(): Se llama cuando la actividad va a comenzar a interactuar con el usuario. Es un buen lugar para lanzar las animaciones y la música.
- onPause(): Indica que la actividad está a punto de ser lanzada a segundo plano, normalmente porque otra actividad es lanzada. Es el lugar adecuado para detener animaciones, música o almacenar los datos que estaban en edición.
- onStop(): La actividad ya no va a ser visible para el usuario
- onRestart(): Indica que la actividad va a volver a ser representada después de haber pasado por onStop().
- onDestroy(): Se llama antes de que la actividad sea totalmente destruida. Por ejemplo, cuando el usuario pulsa el botón de volver o cuando se llama al método finish().

7. **¿Qué es y para qué sirve MATERIAL DESIGN?**
   Es una nueva filosofía enfocada, inicialmente, para el diseño Android; sin embargo, su implementación en el mundo del desarrollo ha sido tal, que muy pronto comenzó a extenderse a toda la web. Es así como la encontramos, actualmente, no solo en aplicaciones Android, sino en páginas web y demás plataformas de software. Su nombre traduce “Diseño material”, y… ¿Qué significa? Esta filosofía de diseño basó su nombre en los objetos materiales. Según la definición oficial de Google: “es un lenguaje que combina los principios innovadores de la tecnología con las normas clásicas del diseño”.
8. **¿Cuáles son las 6 grandes pautas que especifica MATERIAL DESIGN para un buen diseño de apps?**
   - Animaciones: movimientos, transiciones e interacciones receptivas
   - Estilo: colores, iconos
   - Patrones: diseño
   - Componentes:
   - Diseño: estructura
   - Usabilidad:
9. **Menciona 5 "mejores prácticas" indicadas por Google para el "desempeño" (performance) de la aplicación**
   - Uso de procesos u subprocesos: La entrada de manifiesto de cada tipo de elemento de componente (<activity>, <service>, <receiver> y <provider>) admite un atributo android:process que puede especificar un proceso en el cual el componente debe ejecutarse.

- Optimizar para la duración de la batería.
- Reducir el tamaño del apk
- Administra la memoria de la aplicación
- Verificación del comportamiento de la app en el tiempo de ejecución de Android (ART)

10. **Menciona 5 "mejores prácticas" indicadas por Google para la "Crear apps para miles de usuarios"**
    - Conectividad: Descubre cómo brindar una mejor experiencia a los usuarios de redes más lentas. Enfócate en optimizar imágenes, el uso de redes y la transferencia de datos.
    - Capacidad del dispositivo: Descubre cómo agregar compatibilidad para dispositivos con capacidades diferentes de aquellos para los que sueles desarrollar contenido. Ten en cuenta los diferentes tamaños de pantalla, la compatibilidad con versiones anteriores y el uso eficiente de la memoria.
    - Costo de datos: Descubre cómo ayudar a los usuarios a minimizar sus costos de tráfico de red reduciendo el tamaño de las apps y ofreciendo ajustes de red configurables.
    - Consumo de batería: Descubre cómo tu app puede ayudar a extender la duración de la batería. Sigue las recomendaciones indicadas para la administración y el control de la batería a fin de asegurarte de que tu app no agote la carga de forma innecesaria.
    - Interfaz de usuario y contenido: Descubre cómo presentar el contenido para obtener la mejor experiencia del usuario posible. Entre las áreas principales, se incluyen la capacidad de respuesta de la interfaz, las recomendaciones para IU y la localización.


Aplicaciones Moviles
Tienen el sufijo
apk
Compilan tu código, junto con los archivos de recursos y datos
Componentes que forman a una aplicación
Actividades
Una actividad representa una pantalla con interfaz de usuario
Se activa con
startActivityForResult()
Servicios
Es un componente que se ejecuta en segundo plano para realizar operaciones prolongadas o tareas para procesos remotos.
Se activa con
startService()
Proveedores de contenido
Administra un conjunto compartido de datos de la app
Se activa con
sendBroadcast()
sendOrderedBroadcast()
sendStickyBroadcast()
Receptores de mensajes
Es un componente que responde a los anuncios de mensajes en todo el sistema. Muchos mensajes son originados por el sistema.
Se activan con
query() en un ContentResolver
archivo MANIFEST
Permite
* Identificar los permisos de usuario
* Declarar el nivel de API mínimo requerido·
*Declarar características de hardware y software
*Bibliotecas de la API
Estados de una app
Activa (Running) la actividad tiene el foco
Visible (Paused): La actividad es visible pero no tiene el foco
Parada (Stopped): Cuando la actividad no es visible
Destruida (Destroyed): Cuando la actividad termina al invocarse el método finish()
Metodos para manipular estados
* onCreate (Bundle)
* onStart()
* onResume()
* onPause()
* onStop()
* onRestart()
* onDestroy()
MATERIAL DESIGN
Es
Una nueva filosofía enfocada, inicialmente, para el diseño Android
Se rige por
* Animaciones: movimientos, transiciones e interacciones receptivas
* Estilo: colores, iconos
*Patrones: diseño
* Componentes
*Diseño: estructura
*Usabilidad

Bibliografía

- Android Developers (s.f), Aspectos fundamentales de la aplicación, recuperado de: https://developer.android.com/guide/components/fundamentals?hl=es-419
- Máster en Desarrollo de Aplicaciones Android, (2017), Ciclo de vida de una actividad, recuperado de: http://www.androidcurso.com/index.php/tutoriales-android/37-unidad-6-multimedia-y-ciclo-de-vida/158-ciclo-de-vida-de-una-actividad
- Material design: ¿por qué todos hablan de eso? (s.f), recuperado de: https://www.nextu.com/blog/material-design-que-es/
- Diseña para Android (s.f), recuperado de: https://developer.android.com/design?hl=es-419
- Performance and power (s.f), recuperado de: https://developer.android.com/topic/performance
- Cómo crear contenido para miles de millones de usuarios (s.f), recuperado de: https://developer.android.com/docs/quality-guidelines/building-for-billions?hl=es-419