Instituto Tecnológico Superior de Jerez.

Jerez de García Salinas a 03 de Mayo del 2019.

Cristofer Casas Murillo.

cristofer32513@gmail.com

S17070157.

INGENIERÍA EN SISTEMAS COMPUTACIONALES.

Tópicos Avanzados de Programación.

4to. SEMESTRE.

Mapa Conceptual (Aplicaciones Android).

ISC. Salvador Acevedo Sandoval.

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

Un paquete de Android, es un archivo de almacenamiento con el sufijo **.apk** que 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?

- **Actividades:** Representa una pantalla con interfaz de usuario.
  Una actividad se implementa como una subclase de Activity.
- **Servicios:** 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 y se implementa como una subclase de Service.
- **Proveedores de contenido:** Administra un conjunto compartido de datos de la app. A través de este, otras aplicaciones pueden consultar o incluso modificar los datos (si el proveedor de contenido lo permite). También son útiles para leer y escribir datos privados de tu aplicación y que no se comparten.
  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:** Es un componente que responde a los anuncios de mensajes en todo el sistema. Si bien los receptores de mensajes no exhiben una interfaz de usuario, pueden crear una notificación de la barra de estado para alertar al usuario cuando se produzca un evento de mensaje.
  Comúnmente, un receptor de mensajes es simplemente una "puerta de enlace" a otros componentes y está destinado a realizar una cantidad mínima de trabajo.
  Un receptor de mensajes se implementa como una subclase de BroadcastReceiver y cada receptor de mensajes se proporciona como un objeto Intent.

3. ¿Cómo 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 son como mensajeros que solicitan una acción de otros componentes ya sea que el componente le pertenezca o no a tu aplicación.

Para ***actividades y servicios***, una intent define la acción a realizar y puede especificar el URI de los datos en los que debe actuar.

Para los ***receptores de mensajes***, la intent simplemente define el anuncio que se está transmitiendo.

El otro tipo de componente, ***proveedor de contenido***, no se activa mediante intents, sino a través de solicitudes de un ***ContentResolver***. El solucionador de contenido aborda todas las transacciones directas con el proveedor de contenido, de modo que el componente que realiza las transacciones con el proveedor no deba hacerlo y, en su lugar, llame a los métodos del objeto ContentResolver.

Existen métodos independientes para activar cada tipo de componente:

- Puedes iniciar una ***actividad*** al pasar una Intent a ***startActivity()*** o ***startActivityForResult()***.
- Puedes iniciar un ***servicio*** 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. La aplicación debe declarar todos sus componentes en este archivo, y este debe encontrase en la raíz del directorio de proyectos de la aplicación.

Además de declarar los componentes de la aplicación, el archivo MANIFEST sirve para:

- Identificar los permisos de usuario que requiere la aplicación.
- Declarar el nivel de API mínimo requerido por 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.

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.
- **Detenida (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.

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.

- **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.
- **onPause():** Indica que la actividad está a punto de ser lanzada a segundo plano, normalmente porque otra actividad es lanzada.
- **onStop():** La actividad ya no va a ser visible para el usuario, en algunos casos, si hay muy poca memoria, es posible que la actividad se destruya sin llamar a este método.
- **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(), en algunos casos, si hay muy poca memoria, es posible que la actividad se destruya sin llamar a este método.

7. ¿Qué es y para qué sirve MATERIAL DESIGN?

Es una guía integral para el diseño visual, de movimientos y de interacción en distintas plataformas y dispositivos.

8. ¿Cuáles son las 6 grandes pautas que especifica MATERIAL DESIGN para un buen diseño de apps?

- **Material es la metáfora:** Es la teoría que unifica un espacio racionalizado y un sistema de movimiento.
- **Superficies que se basan en la realidad:** El uso de atributos táctiles ayuda a los usuarios a comprender rápidamente la interfaz.
- **Elementos gráficos e intencionales:** El uso adecuado de colores, imágenes y espacios sumergen al usuario en una mejor experiencia y crean jerarquía, significado y enfoque.
- **El movimiento proporciona significado:** El usuario es el motor principal, el que inicia el movimiento, transformando el diseño.
- **Toda acción tiene lugar en un solo entorno:** Los objetos son presentados al usuario sin romper la continuidad de la experiencia a medida que se transforma y reorganizan.

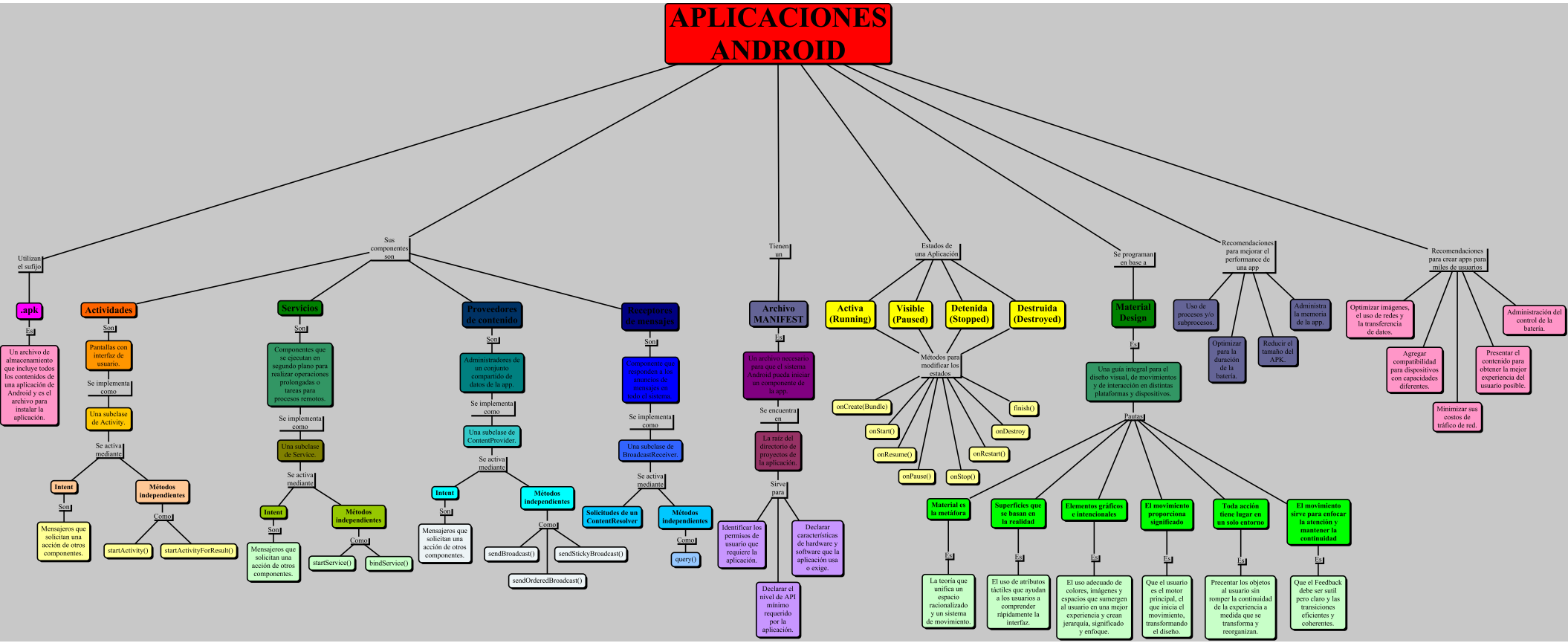
- **El movimiento sirve para enfocar la atención y mantener la continuidad:** El Feedback debe ser sutil pero claro y las transiciones eficientes y coherentes.

9. Menciona 5 "mejores prácticas" indicadas por Google para el "desempeño" (performance) de la aplicación.

- Uso de procesos y/o subprocesos.
- 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".

- Optimizar imágenes, el uso de redes y la transferencia de datos.
- 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.
- Minimizar sus costos de tráfico de red reduciendo el tamaño de las apps y ofreciendo ajustes de red configurables.
- Administración del control de la batería a fin de asegurarte de que tu app no agote la carga de forma innecesaria.
- 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 ANDROID
Utilizan el sufijo
.apk
Es
Un archivo de almacenamiento que incluye todos los contenidos de una aplicación de Android y es el archivo para instalar la aplicación.
Sus componentes son
Actividades
Son
Pantallas con interfaz de usuario.
Se implementa como
Una subclase de Activity.
Se activa mediante
Intent
Son
Mensajeros que solicitan una acción de otros componentes.
Métodos independientes
Como
startActivity()
startActivityForResult()
Servicios
Son
Componentes que se ejecutan en segundo plano para realizar operaciones prolongadas o tareas para procesos remotos.
Se implementa como
Una subclase de Service.
Se activa mediante
Intent
Son
Mensajeros que solicitan una acción de otros componentes.
Métodos independientes
Como
startService()
bindService()
Proveedores de contenido
Son
Administradores de un conjunto compartido de datos de la app.
Se implementa como
Una subclase de ContentProvider.
Se activa mediante
Intent
Son
Mensajeros que solicitan una acción de otros componentes.
Métodos independientes
Como
sendBroadcast()
sendStickyBroadcast()
sendOrderedBroadcast()
Receptores de mensajes
Son
Componente que responden a los anuncios de mensajes en todo el sistema.
Se implementa como
Una subclase de BroadcastReceiver.
Se activa mediante
Solicitudes de un ContentResolver
Métodos independientes
Como
query()
Tienen un
Archivo MANIFEST
Es
Un archivo necesario para que el sistema Android pueda iniciar un componente de la app.
Se encuentra en
La raíz del directorio de proyectos de la aplicación.
Sirve para
Identificar los permisos de usuario que requiere la aplicación.
Declarar características de hardware y software que la aplicación usa o exige.
Declarar el nivel de API mínimo requerido por la aplicación.
Estados de una Aplicación
Activa (Running)
Visible (Paused)
Detenida (Stopped)
Destruida (Destroyed)
Métodos para modificar los estados
onCreate(Bundle)
onStart()
onResume()
onPause()
onStop()
onRestart()
onDestroy
finish()
Se programan en base a
Material Design
Es
Una guía integral para el diseño visual, de movimientos y de interacción en distintas plataformas y dispositivos.
Pautas
Material es la metáfora
Es
La teoría que unifica un espacio racionalizado y un sistema de movimiento.
Superficies que se basan en la realidad
Es
El uso de atributos táctiles que ayudan a los usuarios a comprender rápidamente la interfaz.
Elementos gráficos e intencionales
Es
El uso adecuado de colores, imágenes y espacios que sumergen al usuario en una mejor experiencia y crean jerarquía, significado y enfoque.
El movimiento proporciona significado
Es
Que el usuario es el motor principal, el que inicia el movimiento, transformando el diseño.
Toda acción tiene lugar en un solo entorno
Es
Precentar los objetos al usuario sin romper la continuidad de la experiencia a medida que se transforma y reorganizan.
El movimiento sirve para enfocar la atención y mantener la continuidad
Es
Que el Feedback debe ser sutil pero claro y las transiciones eficientes y coherentes.
Recomendaciones para mejorar el performance de una app
Uso de procesos y/o subprocesos.
Optimizar para la duración de la batería.
Reducir el tamaño del APK.
Administra la memoria de la app.
Recomendaciones para crear apps para miles de usuarios
Optimizar imágenes, el uso de redes y la transferencia de datos.
Agregar compatibilidad para dispositivos con capacidades diferentes.
Minimizar sus costos de tráfico de red.
Presentar el contenido para obtener la mejor experiencia del usuario posible.
Administración del control de la batería.

Referencias.

- NE. (NE). Aspectos fundamentales de la aplicación. 2019, de developer.android.com Sitio web: https://developer.android.com/guide/components/fundamentals?hl=es-419

- NE. (NE). Material Design para Android. 2019, de developer.android.com Sitio web: https://developer.android.com/design/material?hl=es-419

- Sonia Ruiz Cayuela. (NE). Qué es Material Design y cómo se aplica a un sitio web. 2019, de webappdesign.es Sitio web: https://webappdesign.es/que-es-material-design/

- NE. (NE). Ciclo de vida de una actividad. 2019, de androidcurso.com Sitio web: http://www.androidcurso.com/index.php/tutoriales-android/37-unidad-6-multimedia-y-ciclo-de-vida/158-ciclo-de-vida-de-una-actividad