

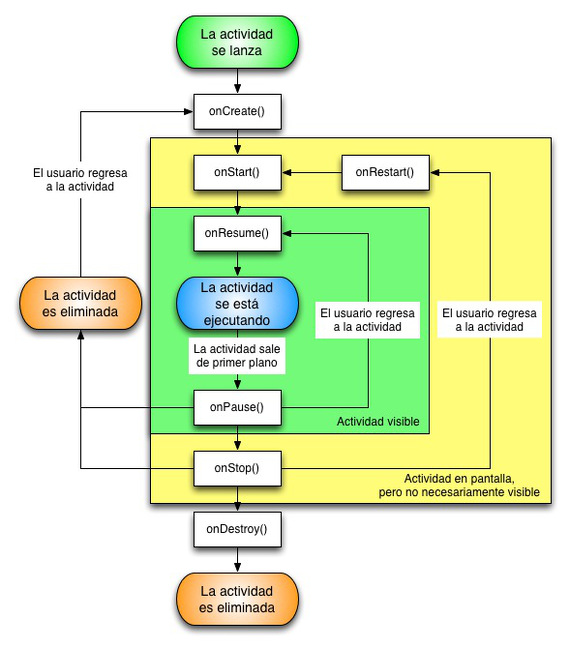
|  |
| --- |
| http://www.becas.sep.gob.mx/images/logo.pngEnsayo de Articulo  TECNOLÓGICO NACIONAL DE MÉXICO  Instituto Tecnológico de Chihuahua II |
|  |
| Ciclo de Vida |
|  |
|  |
| **26/08/2016**  Unidad 1  Ulises Manuel De la Rosa Contreras  13550362  Sistemas Operativos Para Dispositivos Móviles |

DESCRIPION DEL CICLO DE VIDA

Cuando se empieza a programar en un lenguaje como C++ o Java, lo primero que se enseña es el método main, el punto al que llamará el sistema operativo cuando vayamos a arrancar nuestra aplicación.

En Android no existe un método main como tal, pero sí existen varios métodos de nuestra actividad que serán llamados por el SSOO cuando ocurran eventos importantes. En este capítulo estudiaremos a fondo cuáles son esos eventos, y como funciona el ciclo completo de una actividad de Android. La documentación oficial ofrece una explicación extensa de este tema, aquí estudiaremos los elementos más importantes junto con algunos errores comunes a la hora de manejarlos.

El ciclo de vida de Android sigue este esquema:

Los eventos del ciclo de vida

onCreate(Bundle)

Representa el momento en el que la actividad se crea. Este método normalmente lo generará el asistente al crear una nueva actividad en Android, y es donde crearemos todo lo que vaya a necesitar la actividad. Si antes hemos salvado los datos de la actividad en un objeto Bundle, podremos utilizarlo para regenerarla. Normalmente no lo usaremos.

onStart()

La actividad va a pasar a estar en pantalla, aunque no necesariamente visible. Si venimos de una parada, pasaremos antes por onRestart().

onRestart()

Anterior a onStart() cuando procedemos de una llamada a onStop().

onResume()

La actividad va a empezar a responder a la interacción del usuario.

onPause()

La actividad va a dejar de responder a la interacción del usuario.

onStop()

La actividad ha pasado completamente a segundo plano.

onDestroy()

La actividad va a ser destruida y sus recursos liberados.

Cuando necesitemos implementar uno de estos métodos, lo haremos añadiendo a nuestra actividad con estos perfiles:

public class MiActividad extends Activity {

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

...

}

protected void onStart() {

super.onStart();

...

}

protected void onRestart() {

super.onRestart();

...

}

protected void onResume() {

super.onResume();

...

}

protected void onPause() {

...

super.onPause();

}

protected void onStop() {

...

onStop();

}

protected void onDestroy() {

...

super.onDestroy();

}

}

Es importante mantener la llamada al método de la superclase para no encontrarnos con sorpresas. Las tareas de cada evento que están por encima de nuestra actividad deben mantenerse. Esta llamada irá al principio de los eventos de entrada, y al final de los de salida. De esa forma nos evitaremos sorpresas, ya que los elementos de la actividad que necesitemos pero que no estarán bajo nuestro control estarán creados antes de usarlos, y se destruirán después.

No tenemos que añadir todos los eventos, los que no necesitemos usarán la implementación por defecto. Los métodos que a menudo utilizaremos -y no se recomienda tocar los demás- son onCreate, onPause, y onRestart.

El significado de onCreate es claro: es el lugar donde cargaremos los recursos que necesitemos, las vistas, y cualquier otra cosa que necesitemos. Para la salida, el único método en el que nos centraremos es onPause. La razón de evitar onStop y onDestroy es que no tenemos ningún control sobre ellas. onPause se ejecutará siempre que la aplicación salga de primer plano, mientras que los otros dos los ejecutará el sistema operativo en función de sus necesidades. ¡Puede que nunca lleguen a ejecutarse! Esto se hace para evitar el coste que tiene crear la actividad una y otra vez si pasamos de la actividad al escritorio y viceversa, y el sistema operativo sólo liberará los recursos utilizados si los necesita, y no tiene para atender todos los procesos abiertos.

Eso implica que asumiremos que la aplicación morirá después de ejecutar onPause, y que es nuestra última oportunidad de salvar los datos que necesitemos salvar, y de parar servicios que estemos utilizando, como la geolocalización. Si hemos parado servicios, el sitio adecuado para reiniciarlos es onRestart.

Los demás métodos no necesitaremos usarlos mucho. Un caso habitual en el que sí nos hará falta será cuando integremos librerías de terceros, como Facebook o Flurry. En estos casos, se nos pedirá que unamos los métodos de nuestra actividad a su código. Por ejemplo, para registrar una sesión de Flurry se nos pedirá que la sesión comience en el método onStart.

Algunas ideas útiles

Recuerda para qué sirve cada evento. Siempre necesitarás onCreate, y bastante a menudo necesitarás onPause y onResume para parar y relanzar los servicios que más consuman.

No toques los demás eventos si no lo necesitas expresamente.

No confíes en onStop y onDestroy, podrían no llamarse nunca. Salva todo lo que necesites en onPause.

Evita usar variables estáticas no finales. La aplicación puede seguir cargada cuando regreses, y conservarán los valores con los que se quedaron. Si no tienes más remedio que usarlas, asegúrate de reiniciar sus valores al volver al primer plano.