Pontificia Universidad Católica Madre y Maestra

Título:

Reporte #1

Objetivo:

Comprender los conceptos clave relacionados con el desarrollo de aplicaciones Android, incluyendo los entornos de desarrollo, el ciclo de vida de la aplicación,y la identificación del público objetivo y tipos de aplicaciones.

Integrante:

1014-3611 Starlin Frías Luna

Profesor:

Freddy A. Peña P

Santiago de los Caballeros, República Dominicana

Sábado 14 de Septiembre del 2024

El objetivo general de este reporte consiste en la investigación de los entornos Android que más utilizados están siendo de momento, todo con el propósito de lograr un mejor entendimiento de los mismos, como las funcionalidades, ventajas, características de cada uno.

Parte 1: Entornos de Desarrollo:

1. **Flutter:**

   Conozcamos Flutter y todo lo que nos permite este gran framework creado por Google, sus bases están orientadas al desarrollo de aplicaciones móviles, no solo se limita a un SO, sino que permite la creación tanto de aplicaciones Android como para IOS. La diversidad que permite a la hora de creación ha atraído un gran público permitiendo establecerse en el mercado. Es importante resaltar una de las características que lo hacen bastante famoso y es su Hot Reload, que básicamente permite visualizar los cambios que se realizan sin tener que reiniciar toda la aplicación.

   Cuando el objetivo sea utilizar un framework que facilite Widgets ricos y bastante personalizables, pues debemos inclinarnos por Flutter.

   Las mayores ventajas de este famoso framework es la capacidad de desarrollo rápido para las aplicaciones móviles, proporcionando un desarrollo bastante ágil, sin sacrificar lo que es el diseño atractivo, como la adaptabilidad.

   Algo que se puede ver como una desventaja es que no es tan cercano al rendimiento nativo como el que ofrece React Native debido a que este último tiene acceso a las API de Android.

No se puede olvidar que muchas aplicaciones famosas están desarrolladas con este hermoso framework, como lo son Alibaba, el cual todos conocemos que es una tienda bastante utilizada, también Hamilton.

2. **Xamarin:**

   También tenemos un entorno bastante conveniente para aquellos amantes al desarrollo con C#, este es Xamarin, el cual es un entorno creado por Microsoft. Este permite crear aplicaciones de forma nativa, proporcionando altas velocidades, pero no solo se limita a eso, además, permite la creación en multiplataforma (bastante conveniente). Si es cierto que no fue desarrollado para utilizar Java para Android, pero aun así no se puede dejar de lado, importante mencionar que viene integrado en IDE como lo son Visual Studio Code.

   En primer lugar las ventajas y funcionalidades que nos permite Xamarin es su gran familiaridad con las librerías .NET, como es importante saber que .NET cuenta con una cantidad vasta de librerías, pues con Xamarin se tienen acceso a todas estas. También permite el acceso a las API nativas tanto de IOS como de Android permitiendo una mejor sincronización con el hardware del equipo en cuestión.

   Entre las ventajas se pueden destacar el gran soporte por una empresa muy famosa como lo es Microsoft, y su gran integración con Visual Studio Code. Además, su gran facilidad permite compartir códigos entre las diferentes plataformas.

   Desventaja que podría tener es que no cuenta con la gran comunidad como lo es React Native, ni con la facilidad de desarrollo de aplicaciones

elegantes y atractivas sin implicar una gran curva de aprendizaje como lo brinda Flutter. Xamarin requiere de conocimientos de C# como de .NET.

Xamarin nos facilita la creación de aplicaciones para los famosos relojes de Android Watch y Apple.

3. **React Native:**

   El favorito de muchos, si, este es React Native el cual fue desarrollado por Facebook, permitiendo desarrollar utilizando JavaScript. Además, una de las ventajas y funcionalidad que tiene bastante útil es que cuenta con acceso a las API de Android lo que brinda velocidades bastante rápidas, logrando asimilarse a aplicaciones nativas.

   Este framework es tan utilizado y útil, que en su momento en el 2018 llegó a ser el #2 en contribuciones en los repositorios de Github. También permite visualizar los cambios inmediatamente estos son guardados debido a su conectividad con JS.

   Una de las ventajas que hace resaltar este framework por encima de muchos es la facilidad que permite para la reutilización de código desde Android a Apple, permitiendo un desarrollo mucho más eficiente desde una plataforma a otra. Además, lo mismo que se ha mencionado en varias ocasiones el rendimiento cercano al nativo, debido a las API.

   También cuenta con la recarga caliente (Hot Reload) que también lo tiene Flutter.

   Desventajas que podrían verse de este framework es que la curva de aprendizaje es un poco más inclinada que con Flutter, además, de que no

tienen los famosos widgets tan personalizables y atractivos con los que cuenta Flutter.

¿Nunca se han preguntado con qué está hecho Messenger o Instagram?, como es de saber, las dos aplicaciones mencionadas anteriormente son bastante utilizadas por los usuarios de todo el mundo sin importar el SO, pues ambas están desarrolladas con React Native.

Parte 2: Ciclo de Vida de la Aplicación Android

Para hablar del ciclo de vida de una aplicación Android primero se debe conocer la clase Activity la cual es la que realiza las llamadas de los diferentes estados. Los 6 diferentes estados que proporciona la clase Activity para gestionar las transacciones del ciclo de vida son **onCreate(), onStart(), onResume(), onPause(), onStop(), onDestroy().**

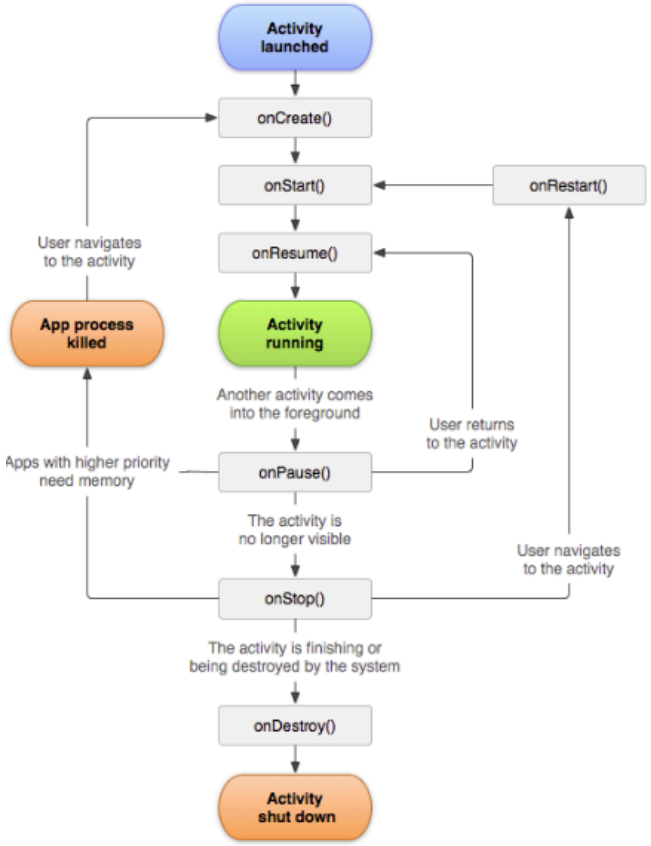

Figura 1: Ilustración simplificada del ciclo de vida de una actividad

- **onCreate():**

  Está básicamente es llamada cuando la actividad se crea por primera vez, su objetivo es la creación de componentes esenciales. Esta realiza la transacción cuando el sistema llama el **onStart().**

- **onStart():**

  Aquí es cuando la actividad que se ha creado se vuelve visible para el usuario en sí, permitiendo preparar los recursos que se mostrarán en la pantalla. La actividad pasa a primer plano y se prepara para interactuar con el usuario.

- **onPause():**

  Esta es necesaria y se llama siempre y cuando el sistema requiere la inicialización de otra actividad o reanudar alguna otra, pues se llama **onPause(),** y se pone en pausa la actividad actual. Es importante mencionar que se deben guardar aquellos cambios que no son persistentes. Esta tiene 2 transacciones, puede volver a **onResume()** o pasar a **onStop().**

- **onStop():**

  En esta etapa del ciclo de vida la actividad vuelve a no ser visible para el usuario, además, permite la liberación de los recursos del sistema. La transacción de esta etapa puede ser **onResume()** siempre y cuando se vuelva a reanudar, o **onDestroy()** si se destruye.

- **onResume():**

  Esta etapa se llama en el ciclo siempre y cuando una actividad en **onStop()** vuelve a reanudarse, ahí pasa a **onResume(),** básicamente lo

que permite restaurar el estado previo de la actividad. La transacción siempre será seguida por **onStart().**

- **onDestroy():**

  La última etapa del ciclo de vida es **onDestroy(),** es llamado antes de que la actividad sea destruida completamente, aquí es donde se liberan todos los recursos relacionados con la actividad en cuestión.

Son muchos los escenarios donde se requiere un entendimiento decente sobre las etapas descritas anteriormente, debido a que un mal entendimiento de las misma puede llevar a errores críticos como lo que veremos ahora:

-Persistencia de datos: En onPause() o onStop() se debe guardar los datos que no son persistentes, para así lograr reanudar la actividad en el estado previo.

-Multitarea: Es necesario entender el onRestart() y onStop() para lograr el funcionamiento correcto cuando hacemos un cambio de aplicación de una a otra.

Así como los ejemplos mencionados anteriormente hay muchísimos más que podrían desencadenarse de la mala gestión de estas llamadas (etapas).

Parte 3: Tipos de Aplicaciones y Público Objetivo:

- **Aplicaciones redes sociales:**

  En esta categoría hay bastantes como lo son Instagram, Facebook, Twitter, etc… Estas van para todo público, personas desde 12 años hasta adultos de 60, 65 o más. Estas van dirigidas a personas que le gusta mantenerse informada de acontecimientos que suceden alrededor del mundo, como también para mantenerse al tanto de amigos y familiares.

- **Aplicaciones de productividad:**

  Estas van más dirigidas a estudiantes y profesionales que requieren el uso de herramientas que faciliten el trabajo tanto individual como en equipo. Estas aplicaciones permiten una mejor organización y gestión de los elementos en cuestión. Una de estas aplicaciones bastante utilizada por diferentes entornos son Trello, Jira, etc.

- **Juegos:**

  El objetivo principal de estas aplicaciones es brindar entretenimiento por encima del de las redes sociales, permite todo tipo de público debido a su diversidad. Una de estas aplicaciones bastante utilizada y jugada por todo tipo de personas es Candy Crush.

Bibliografía:

- Badal, H. (2024, 9 julio). *Los mejores entornos para crear aplicaciones Android [sin Java]*. Yeeply. https://www.yeeply.com/blog/desarrollo-de-apps/entornos-programacion-desarrollar-apps-android/
- De Imagina, E. (2024, 8 septiembre). *Flutter, React Native o Xamarin ¿Cuál es el Mejor Framework?* Imagina Formación. https://imaginaformacion.com/tutoriales/flutter-react-native-o-xamarin-cual-es-el-mejor-framework-para-desarrollar-una-aplicacion-movil
- *Ciclo de vida de la actividad*. (s. f.). Android Developers. https://developer.android.com/guide/components/activities/activity-lifecycle?hl=es-419