Programación Móvil.

Nombre: Luis Ismael López Urbina

Docente: Luis Guido.

Fecha: 17 de Mayo de 2024.

Índice

Componentes Clave del Proyecto 3

Módulos 3

Archivos del Proyecto 3

Configuración de la Estructura del Proyecto 3

Crear un Proyecto en Android Studio 4

Crear un Nuevo Proyecto 4

Importar un Proyecto Existente 5

Migrar a Android Studio 5

Aspectos Básicos de Android Studio 5

Migrar desde IntelliJ 6

Configurar Android Studio 8

Archivos de configuración 8

Actualizar el IDE y las herramientas SDK en Android Studio 11

Eliminar directorios no utilizados 12

Actualizar las herramientas SDK con el SDK Manager 12

Paquetes Requeridos 12

Paquetes Recomendados 13

Editar o añadir sitios de herramientas SDK 13

Descargar automáticamente paquetes faltantes con Gradle 13

Actualizar las herramientas con la línea de comandos 14

Componentes Clave del Proyecto

Módulos

Un módulo es una colección de archivos fuente y configuraciones de construcción que permiten dividir el proyecto en unidades funcionales discretas. Los módulos pueden ser independientes y pueden utilizarse como dependencias unos de otros. Existen diferentes tipos de módulos:

- Android app module: Contiene el código fuente, archivos de recursos y configuraciones de la aplicación. Hay varios tipos según el dispositivo: Phone & Tablet, Automotive, Wear OS, etc.

- Feature module: Representa una característica modularizada que puede entregarse bajo demanda a través de Google Play.

- Library module: Contiene código reutilizable que puede usarse como dependencia en otros módulos o proyectos. Puede ser una Android Library, Android Native Library o Java/Kotlin Library.

Archivos del Proyecto

Por defecto, Android Studio muestra los archivos del proyecto en la vista "Android", que organiza los archivos por módulos y tipos de archivo, simplificando la navegación. La estructura real de archivos puede verse seleccionando la vista "Project".

- manifests: Contiene el archivo AndroidManifest.xml.

- java: Contiene archivos de código fuente Java y Kotlin.

- res: Contiene recursos no codificados como cadenas de UI e imágenes.

- assets: Contiene archivos que se compilan tal cual en el APK.

- build.gradle: Define las configuraciones de construcción específicas del módulo y del proyecto.

Configuración de la Estructura del Proyecto

Puedes cambiar varias configuraciones del proyecto desde el diálogo Project Structure:

- Project: Establece la versión de Gradle y el plugin de Gradle de Android.

- SDK Location: Define la ubicación del JDK, Android SDK y NDK.

- Variables: Permite editar variables utilizadas en los scripts de construcción.

- Modules: Configura las propiedades específicas de cada módulo, incluyendo la SDK objetivo y mínima, firma de la aplicación y dependencias.

- Build Variants: Configura diferentes sabores y tipos de construcción del proyecto. Cada sabor puede especificar configuraciones como la versión mínima y objetivo de la SDK, y cada tipo de construcción puede definir configuraciones para debug y release, entre otros.

Crear un Proyecto en Android Studio

Crear un Nuevo Proyecto

1. Iniciar un Nuevo Proyecto:

- Sin un proyecto abierto: Hacer clic en "Start a new Android Studio project" en la pantalla de bienvenida.

- Con un proyecto abierto: Seleccionar `File > New > New Project` en el menú principal.

2. Elegir el Tipo de Proyecto:

- En la pantalla "New Project", seleccionar el tipo de proyecto que se desea crear desde las categorías de dispositivos mostradas en el panel de plantillas.

- Android Studio incluirá código de ejemplo y recursos en el proyecto para ayudarnos a comenzar.

3. Configurar el Proyecto:

- Nombre del Proyecto: Especificar el nombre del proyecto.

- Nombre del Paquete: Este nombre será el espacio de nombres del proyecto y el ID de la aplicación para su publicación.

- Ubicación de Guardado: Seleccionar dónde se desea almacenar el proyecto localmente.

- Lenguaje: Elegir Kotlin o Java. Se pueden usar ambos lenguajes.

- Nivel Mínimo de API: Seleccionar el nivel de API mínimo que la aplicación soportará. Un nivel de API más bajo permite que más dispositivos ejecuten la aplicación, pero limita el uso de APIs modernas.

- Bibliotecas AndroidX: El proyecto usará las bibliotecas AndroidX por defecto. Para usar las bibliotecas de soporte heredadas, seleccionar "Use legacy android.support libraries" .

4. Finalizar la Configuración:

- Cuando todo este completado, hacer clic en "Finish".
- Android Studio creará el nuevo proyecto con código y recursos básicos para empezar.

Importar un Proyecto Existente

Para importar un proyecto local existente en Android Studio:

1. Hacer clic en `File > New > Import Project`.
2. Navegar al directorio raíz del proyecto que se desea importar.
3. Hacer clic en "OK".
   - Android Studio abrirá el proyecto en una nueva ventana del IDE e indexará su contenido.

Para importar un proyecto desde control de versiones:

1. Seleccionar `File > New > Project from Version Control`.

Migrar a Android Studio

Aspectos Básicos de Android Studio

Organización de proyectos y módulos:

Android Studio se basa en IntelliJ IDEA IDE y organiza el código en proyectos que contienen todo lo necesario para definir una aplicación Android, desde el código fuente hasta las configuraciones de

compilación y pruebas. Los proyectos se abren en ventanas separadas de Android Studio y pueden contener uno o más módulos, que permiten dividir el proyecto en unidades funcionales independientes.

Sistema de compilación basado en Gradle:

El sistema de compilación de Android Studio se basa en Gradle y utiliza archivos de configuración de compilación escritos en Groovy o Kotlin. Algunas características clave son:

- Soporte para bibliotecas binarias (AARs).

- Soporte para variantes de compilación.

- Configuración y personalización fáciles de la compilación.

- Gradle se puede usar desde el IDE, la línea de comandos y servidores de integración continua como Jenkins.

Dependencias:

Las dependencias de bibliotecas en Android Studio usan declaraciones de dependencias de Gradle y dependencias de Maven para bibliotecas locales y binarias conocidas.

Migrar desde IntelliJ

Importar un proyecto de IntelliJ basado en Gradle:

Si el proyecto de IntelliJ ya utiliza Gradle:

1. Hacer clic en `File > New > Import Project`.

2. Seleccionar el directorio del proyecto IntelliJ y hacer clic en OK.

Importar un proyecto de IntelliJ no basado en Gradle:

Si tu proyecto de IntelliJ no usa Gradle, tienes dos opciones:

1. Crear un nuevo proyecto vacío:

   - Abrir Android Studio y hacer clic en `File > New > New Project`.

   - Ingresar un nombre y especifica la ubicación para el proyecto, luego haz clic en Next.

- Seleccionar los factores de forma de tu aplicación y haz clic en Next.

- Hacer clic en `Add No Activity` y luego en Finish.

- En la ventana de proyecto, seleccionar la vista "Project" para ver la estructura del nuevo proyecto.

- Copiar el código, pruebas unitarias, pruebas de instrumentación y recursos de tu proyecto antiguo a las ubicaciones correctas en la nueva estructura de proyecto.

- Hacer clic en `File > Project Structure` para abrir el diálogo de estructura del proyecto y asegúrate de que el módulo de tu aplicación esté seleccionado.

- Modificar las propiedades necesarias, agrega dependencias como dependencias de Gradle y haz clic en OK para guardar los cambios.

- Hacer clic en `Build > Make Project` para probar la compilación del proyecto y resolver cualquier error.

2. Crear un archivo de compilación Gradle personalizado:

- Respaldar los archivos de proyecto en una ubicación separada.

- Crear un archivo `build.gradle` (o `build.gradle.kts` si usas Kotlin) en el directorio de proyecto.

- Organizar el proyecto según la estructura predeterminada y configura los repositorios en `settings.gradle` o `settings.gradle.kts`.

- Agrega declaraciones de dependencias en el bloque `dependencies{}` de tu archivo de compilación.

- Eliminar el directorio `.idea` y los archivos IML del proyecto.

- Abrir Android Studio y hacer clic en `File > New > Import Project`.

- Seleccionar el archivo `build.gradle` o `build.gradle.kts` que se creó y hacer clic en OK para importar el proyecto.

- Hacer clic en `Build > Make Project` para probar la compilación y resolver cualquier error.

Próximos Pasos

Configuración del control de versiones

- Desde el menú VCS, hacer clic en `Enable Version Control Integration` y seleccionar el sistema de control de versiones adecuado.

- Si la aplicación no está bajo control de versiones, configurar esto después de importar la aplicación a Android Studio.

Firma de la aplicación:

- Android Studio detectará un certificado de depuración o utilizará uno generado por el propio Android Studio.

- Agregar la configuración de firma para la versión de lanzamiento en el archivo `build.gradle`.

Ajustar el tamaño máximo del heap de Android Studio:

- Incrementar el tamaño máximo del heap para mejorar el rendimiento en proyectos grandes.

Actualizaciones de software:

- Android Studio se actualiza independientemente del plugin de Gradle, las herramientas de compilación y las herramientas del SDK.

- Puedes especificar las versiones que deseas usar y elegir recibir versiones preliminares o beta.

Configurar Android Studio

Archivos de configuración

Android Studio ofrece acceso a dos archivos de configuración a través del menú de Ayuda:

1. studio.vmoptions: Permite personalizar opciones para la Máquina Virtual de Java (JVM) de Android Studio, como el tamaño del heap y la memoria caché.

2. dea.properties: Permite personalizar propiedades de Android Studio, como la ruta de la carpeta de plugins o el tamaño máximo de archivo soportado.

Personalización de las opciones de la JVM

Para mejorar el rendimiento de Android Studio, se recomienda ajustar el tamaño máximo del heap, especialmente si trabaja en proyectos grandes o si su sistema tiene mucha RAM.

Pasos para editar el archivo `studio.vmoptions:

1. Ir a `Ayuda > Editar opciones de VM personalizadas`.

2. Si nunca se ha editado estas opciones, el IDE le pedirá crear un nuevo archivo `studio.vmoptions`. Hacer clic en `Crear`.

3. El archivo se abrirá en el editor de Android Studio. Editar el archivo para agregar las opciones personalizadas de JVM.

Para ajustar el tamaño del heap:

1. Ir a `Archivo > Configuración (Android Studio > Preferencias en macOS)`.

2. Seleccionar `Apariencia y comportamiento > Configuración del sistema > Configuración de memoria`.

3. Ajustar el tamaño del heap según sea necesario y hacer clic en `Aplicar`.

Exportar e importar configuraciones del IDE

Se puede exportar un archivo `Settings.jar` que contiene sus configuraciones preferidas del IDE y luego importarlo en otros proyectos.

Personalización de las propiedades del IDE

El archivo `idea.properties` permite personalizar varias propiedades del IDE. Para crear o editar este archivo:

1. Ir a `Ayuda > Editar propiedades personalizadas`.

2. Si es la primera vez, Android Studio pedirá crear un nuevo archivo `idea.properties`. Hacer clic en `Sí` para crear el archivo.

3. Editar el archivo para agregar las propiedades personalizadas.

Ejemplos de configuraciones comunes en `idea.properties` incluyen:

- Ruta de la carpeta de plugins: `# idea.plugins.path=${idea.config.path}/plugins`

- Tamaño máximo de archivo para asistencia de código: `idea.max.intellisense.filesize=2500`

- Tamaño del buffer cíclico de la consola: `idea.cycle.buffer.size=1024`

Configuración del proxy HTTP

Para configurar un proxy HTTP en Android Studio:

1. Ir a `Archivo > Configuración (Android Studio > Preferencias en macOS)`.

2. Seleccionar `Apariencia y comportamiento > Configuración del sistema > Proxy HTTP`.

3. Elijir `Detección automática de configuraciones de proxy` o `Configuración manual del proxy` e ingresar los detalles necesarios.

4. Hacer clic en `Aplicar` o `Aceptar`.

Optimizar el rendimiento en Windows

Para mejorar el rendimiento de Android Studio en Windows:

- Excluir los directorios de Android Studio de la exploración en tiempo real de su software antivirus, como `%USERPROFILE%\.gradle`, `%USERPROFILE%\AndroidStudioProjects`, y `%USERPROFILE%\AppData\Local\Android\SDK`.

- Minimizar el tamaño máximo del heap para Android Studio y Gradle si se tiene menos memoria RAM disponible.

- Desactivar la compilación paralela de módulos independientes y otras configuraciones intensivas en memoria.

Configuración de dispositivos virtuales y físicos

- Crear y gestionar dispositivos virtuales

- Ejecutar aplicaciones en un dispositivo de hardware

- Instalar controladores USB OEM

Actualizar el IDE y las herramientas SDK en Android Studio

Actualizar el IDE

Usando JetBrains Toolbox

Si se instaló Android Studio usando JetBrains Toolbox, esta herramienta se encarga de gestionar las actualizaciones por ti. Toolbox permite la instalación de versiones canary, beta y estables en paralelo, y también posibilita volver a versiones anteriores si es necesario. Las actualizaciones disponibles se mostrarán en Toolbox.

Comprobando actualizaciones manualmente

Si se instaló Android Studio manualmente, este notificará con un pequeño diálogo cuando haya una actualización disponible. Para comprobar las actualizaciones manualmente:

1. Hacer clic en: File > Settings > Appearance & Behavior > System Settings > Updates (en macOS, Android Studio > Check for Updates).

2. Elegir entre los siguientes canales:

- Canal Canary: Actualizaciones semanales con las últimas características, pero más propensas a errores. No se recomienda para desarrollo en producción.

- Canal Beta: Candidatos a versión estable, basados en las versiones canary estables.

- Canal Estable: Versiones oficiales y estables de Android Studio.

Eliminar directorios no utilizados

Cuando ejecutas una nueva versión mayor de Android Studio por primera vez, puede que se te solicite eliminar directorios de versiones anteriores que ya no se utilizan. Esto ayuda a liberar espacio en el disco.

Actualizar las herramientas SDK con el SDK Manager

El SDK Manager te ayuda a descargar las herramientas SDK, plataformas y otros componentes que necesitas para desarrollar tus aplicaciones. Una vez descargados, puedes encontrar cada paquete en el directorio indicado como la **Android SDK Location**.

Para abrir el SDK Manager desde Android Studio:

1. Hacer clic en : Tools > SDK Manager o en el icono del SDK Manager en la barra de herramientas.

2. Cuando haya una actualización disponible para un paquete que ya tienes, aparecerá un guion (-) en la casilla junto al paquete.

3. Para actualizar un elemento o instalar uno nuevo, seleccionar la casilla.

4. Para desinstalar un paquete, desmarcar la casilla.

5. Las actualizaciones pendientes se indican en la columna izquierda con un icono de descarga, y las eliminaciones pendientes con una X roja.

6. Para actualizar los paquetes seleccionados, hacer clic en **Apply** o **OK** y aceptar los acuerdos de licencia.

Paquetes Requeridos

En la pestaña SDK Tools se pueden encontrar las siguientes herramientas:

- Android SDK Build Tools: Incluye herramientas para compilar aplicaciones Android.

- Android SDK Platform Tools: Incluye herramientas necesarias para la plataforma Android, como el comando adb.

- Android SDK Command-Line Tools: Incluye herramientas esenciales como ProGuard.

En la pestaña SDK Platforms:

- Debes instalar al menos una versión de la plataforma Android para poder compilar tu aplicación. Usa la última versión como tu objetivo de compilación para proporcionar la mejor experiencia de usuario en los dispositivos más recientes.

Paquetes Recomendados

- Android Emulator: Una herramienta de emulación basada en QEMU para depurar y probar aplicaciones en un entorno de tiempo de ejecución Android real.

- Imágenes del sistema Intel o ARM: Necesarias para ejecutar el emulador de Android.

- Google Play services: Incluye bibliotecas, documentación y ejemplos para ayudar a construir tu aplicación.

Editar o añadir sitios de herramientas SDK

En la pestaña SDK Update Sites, puedes añadir y gestionar otros sitios que alojan sus propias herramientas y luego descargar los paquetes de esos sitios.

Para añadir un sitio de herramientas SDK:

1. Hacer clic en la pestaña SDK Update Sites.

2. Hacer clic en Add en la parte superior de la ventana.

3. Introducir el nombre y la URL del sitio de terceros y haz clic en OK.

4. Asegurarse de que la casilla esté seleccionada en la columna Enabled.

5. Hacer clic en Apply u OK.

Descargar automáticamente paquetes faltantes con Gradle

Gradle puede descargar automáticamente los paquetes SDK faltantes en los que depende un proyecto, siempre y cuando ya se hayan aceptado los acuerdos de licencia correspondientes en el SDK Manager.

Para copiar las licencias a otra máquina

1. En una máquina con Android Studio instalado, hacer clic en Tools > SDK Manager y anota la ubicación del Android SDK Location.

2. Navegar a ese directorio y localizar el directorio licenses/.

3. Copiar el directorio licenses/ completo y pegarlo en el directorio principal del SDK de Android en la otra máquina.

Actualizar las herramientas con la línea de comandos

En sistemas sin interfaz gráfica, como los servidores CI, usa la herramienta de línea de comandos `sdkmanager` para instalar y actualizar las herramientas y plataformas SDK.

Para aceptar las licencias faltantes:

```bash
$ sdkmanager --licenses
```

Este comando escanea todas las herramientas y plataformas SDK instaladas y muestra las licencias que no han sido aceptadas. Se te pedirá aceptar cada licencia.