**Parcial 2**

**Traducción**

**Utilice Snapshots para restaurar cambios en todo el proyecto**

Las instantáneas son archivos que incluyen el estado actual de toda la configuración del proyecto y del espacio de trabajo y todos los archivos de documento en el proyecto. Las instantáneas proveen una forma fácil de realizar una copia de seguridad de la versión actual de tu proyecto o espacio de trabajo. Si algo sale mal por un cambio en el código que hayas hecho, una instantánea hace fácil restaurar tu proyecto entero, incluso un proyecto borrado, a su estado previo. Xcode automáticamente crea una instantánea del proyecto o espacio de trabajo antes de que tú realices alguna operación de edición masiva, tal como re factorizar su código o ejecutar una operación Búsqueda y Reemplazar. También puedes configurar Xcode para que automáticamente cree una instantánea en otras circunstancias, como antes de una compilación, seleccionando Xcode > Preferencias, seleccionando Comportamientos, y seleccionando la opción Crear Instantánea. También puedes crear una instantánea manualmente seleccionando Archivo > Crear Instantánea.

**DESARROLLO DE APLICACIONES NATIVAS PARA ANDROID**

**Herramientas de Desarrollo**

*Android SDK*

Android SDK provee las librerías API y las herramientas de desarrollo necesarias para crear, probar, y depurar aplicaciones para Android. El paquete de herramientas de desarrollo de Android (ADT) incluye los componentes esenciales del SDK de Android y una versión del eclipse IDE con herramientas de desarrollo de Android para optimizar las aplicaciones de Android. Un nuevo ambiente de desarrollo de Android llamado Android Studio, basado en IntelliJ IDEA, está ahora disponible como un avance de acceso anticipado.

*Setting up the ADT Bundle*

Descargar la última versión del paquete ADT en <http://developer.android.com/sdk/index.html>. Installar el SDK y el Eclipse Ide. Desempaquetar el archivo ZIP (llamado adt-bundle-<os\_platform>.zip) y guardarlo a una ubicación apropiada, como un directorio de desarrollo en su directorio de inicio. Abrir el directorio de adt-bundle-<os\_platform>/eclipse/ y ejecutar eclipse.

**Codificación**

*Definir la interfaz de usuario*

Para empezar con la codificación el primer paso debe ser definir la interfaz de usuario. La UI para una aplicación es generalmente definida como una serie de archivos de diseño. Estos son archivos basados de XML que describen los controles en una pantalla y la relación de sus diseños entre sí. Estos se colocan en el proyecto en una estructura de directorio definida por el Android SDK. Durante el tiempo de ejecución, Android dinámicamente carga contenido desde esta estructura de directorio. Para obtener el código Java de la interfaz debe escribirse para responder a varios eventos que ocurren desde los controles en una determinada pantalla y desde los cambios en el ciclo de vida de una aplicación. El código Java también es responsable de la carga de diseño y los archivos de menú asociados en cada pantalla. La SDK de Android fue diseñada desde el comienzo para permitir una amplia variedad de capacidades del dispositivo. Mucho se ha escrito sobre la fragmentación de dispositivos en la plataforma Android. Los atributos del dispositivo incluyen el tamaño de la pantalla, densidad de pixel, y las versiones de Android API. Diseños de pantalla, recursos de imagen (llamados dibujables), estilos y otros archivos de configuración se encuentran todos incorporados en una serie de subdirectorios debajo de un directorio maestro de “recursos”. En el proceso de definir los diseños, debe asociar un valor de ID con cada control (o widget de UI) al que desea referenciar desde el código. Esto pude hacerse especificando un valor de ID en el inspector de propiedades en el editor de diseño. En iOS, cargamos la lógica específica a una pantalla determinada en un ViewController. En Android, estas pantallas separadas son tratadas como “actividades” separadas. Tal como en un UIViewController de iOS, hay un ciclo de vida definido para una actividad que gobierna cuando la actividad comienza, pausa, continua, y se detiene. Depende del desarrollador colocar código en los diversos métodos de la actividad para responder a los diversos estados del ciclo de vida.

*Actividad*

La actividad es la forma más visible y prominente de una aplicación Android. Una actividad presenta la interfaz de usuario a una aplicación con la asistencia de una clase conocida como vista. La clase vista es implementada como varios elementos UI, como las cajas de texto, etiquetas, botones, y otras UI típicas de las plataformas informáticas móviles. Una aplicación puede contener una o más actividades. Tienen típicamente una relación de uno a uno con las pantallas encontradas en una aplicación. Una aplicación se mueve de una actividad a otra llamando a un método conocido como startActivity() o startSubActivity(). El método anterior es usado cuando la aplicación desea simplemente “cambiar” a la nueva actividad. Este último es usado cuando se desea un paradigma de llamada/respuesta sincrónica. En ambos casos, la intención es pasado como un argumento al método.

*Filtros de Intención*

Los componentes centrales de una aplicación (sus actividades, servicios, y receptores de difusión) son activados por intenciones. Una intensión es un conjunto de información (un objeto Intent) que describe una acción deseada – incluyendo los datos sobre los que se actuará, en la categoría de componentes que debe realizar la acción, y otras instrucciones pertinentes. Android localiza un componente apropiado para responder a la intención, lanza una nueva instancia del componente si uno es necesario, y lo pasa al objeto Intent. Los componentes anuncian sus capacidades – las clases de intenciones que pueden responder – a través de filtros de intención. Dado que el sistema Android debe aprender que intenciones puede manejar antes de que el componente se lance, los filtros de intenciones son especificados en el manifiesto como elementos <intent-filter>. Un componente puede tener cualquier número de filtros, cada uno describiendo una capacidad diferente. El siguiente fragmento viene de una aplicación Android usando el filtro de intención que responde a los mensajes SMS (texto) entrantes:

\*\* IMAGEN \*\*

**Figura 1: una fracción de código del IntentFilter**

*Proveedor de Contenido*

El Proveedor de Contenido es el mecanismo de Android para la abstracción del almacén de datos. Por ejemplo, el libro de direcciones contiene todos los contactos y números de teléfonos que una persona puede requerir al utilizar un teléfono móvil. El Proveedor de Contenido es un mecanismo para abstraer el acceso a un almacén de datos particular. De muchas maneras, el Proveedor de Contenido actúa en el rol de un servidor de base de datos. Las operaciones para leer y escribir contenido en un almacén de datos particular deben ser pasadas a través de un Proveedor de Contenido apropiado, en lugar de acceder a un archivo o base de datos directamente.

*Servicios*

Al igual que otros entornos informáticos de multitarea, hay aplicaciones que se ejecutan “en segundo plano” que realizan varias funciones. Android llama a estos tipos de aplicaciones “servicios”. El servicio es una aplicación de Android que no tiene Interfaz de Usuario.

*AndroidManifest.xml*

Todas las aplicaciones deben tener un archivo AndroidManifest.xml (con precisamente ese nombre) en su directorio raíz. El archivo de manifiesto presenta información esencial sobre tu aplicación al sistema Android, información que el sistema debe tener antes de que pueda ejecutar cualquiera de los códigos de la aplicación.