**Clase 1**

Aplicación: Mostrar un texto que cambie de tamaño al pulsar un botón

**Clase 2**

Cosas a utilizar: editText, Button, Intent, TextView

Aplicación: Desarrollar una app que tenga dos activities:

* Activity 1: “Log in” debe contener una entrada de usuario y pass, y un botón de enter. Además se debe pasar datos a la Activity 2.
* Activity 2: Un texto, 3 botones (uno de “C” y otros “+” y “-“). Recibir el dato e imprimirlo. El texto debe cambiar de tamaño con los botones + y – y de color con C.

Botones: usa el concepto de listening que es un pedazo de código que se queda esperando un evento (similar a una interrupción).

\*Con alt + Enter importo la librería que falta

EditText: cuando en este objeto no se ingresa nada y se hace uso de getText se crashea el programa.

Para pasar de una activity a la otra se utiliza intents.

Para pasar datos entre activities se debe tener en cuenta la cantidad de datos y cuantas activities lo van a leer.

* Pocos datos y se pasa a una sola activity: utilizo el intent directamente
* Pocos datos y se pasa a muchas activities:
* Muchos datos: Se utiliza base de datos

**Clase 3**

Temas de hoy: Spinners, ListView, ArrayAdapter

\* Apartir de ahora se va a usar ConstrainLayout

Spinners: Es una lista desplegable, utiliza MVC (Model View Controller) que es un paradigma de interactuar modelos con vistas. La idea es que los model no vean view y todo la información de comunicación para por el controller.

\* En android los controller son adapters que permite sincronizar modelos con vistas

ListView: es una vista scroleable donde cada celda es un item. Se parece a un Spinner. Actualmente se usa ReceiclerView.

Codigo: Se va a personalizar cada item -> Arrastrar un ListView. Crear una clase .java; Con la clase y el item se puede armar el adaptador. El item como es personalizado se debe crear un xml con dos textView.

Dentro de getView se hacer la sincronizacion

setOnItemClickListener() devuelve la posicion y la vista

\*\*\*VER EJEMPLO SGOlIVER

Mientras las vistas sean de mostrar informacion el ListView no tiene inconvenientes pero poner cosas como botones trae inconvenientes por el metodo setOnClickListener(). Con ReceiclerView se tiene la ventaja de tener un listener por cada item, y ademas solo carga en memoria los item desplegados y no todo como en ListView

SplashCream: hay dos tipos, las inutiles y las utilies. El primero mientras informacion y cosas asi, las otras descargan cosas. Los Splash tienen un timer con un scheduler. En los splash de deben hacer finish() ya que no quiero que me vuelva a mostrar la presentacion por ejemplo.

Para que empieze desde splash hay que agregar a la acitvity correspondiente el intent-filter (splash\_activity)

\*\*\* Ver documentacion compartida

Cosas a agregar al proyecto:

Splash debe ejecutarse primero y luego abrir la activity LogIn. Despues al loguearse ir a una activity que tenga un ListView con objetos propios donde al hacer click en cualquier item abrir una activity y pasar el dato con el detalle. En dicha activity poner un spinner y mostrar el detalle.

Usar constrainLayout en la activity LogIn y Detail

ConstrainLayout:

HorizontalGradle: pone una linea de margen, usar porcentual

Siempre hay que enganchar los cuatro bordes. En general se utiliza mucho wrap\_content o match\_constrain

\* AndroidManifest.xml tiene todas las activities