Dialogos y fragmentos

Los diálogos son ventanas que aparecen en la pantalla de tu dispositivo espontáneamente para hacer al usuario alguna pregunta sobre alguna decisión a tomar para poder proceder con el funcionamiento normal de tu App.

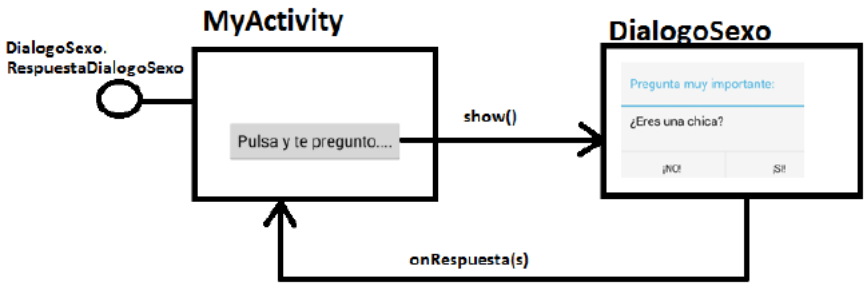
Hay muchos tipos:

- Diálogos con listas de elementos
- Diálogos con botones de tipo Radio o checkboxes
- incluso puedes usar un fichero de recursos XML para definir un Layout en que diseñes tu diálogo personalizado. (Para cómo personalizar tus diálogos https://developer.android.com/guide/topics/ui/dialogs#CustomLayout)

Android recomienda usar la clase DialogFragment para contener diálogos. Un fragmento de una actividad es una parte de tu App que controla parte de la interface de usuario de la Activity. Los fragmentos tienen su propio ciclo de vida, reciben sus propios eventos y se pueden eliminar o añadir a la Actividad mientras esta se ejecuta (ver ejemplos de Fragment en la plataforma). Un DialogFragment es un tipo especial de fragmento usado para crear diálogos.

Expliquemos esto mediante un ejemplo de un escenario que en base a una actuación de un usuario necesitas enseñarle un cuadro de diálogo para hacerle una pregunta, y en base a la respuesta, la aplicación tomará un curso de acción u otro. Si lo haces con las recomendaciones de Android y programas el diálogo como un DialogFrament, además te servirá para otras ocasiones. Por ejemplo, tienes una Actividad con un botón, al pulsar el botón, aparece un cuadro preguntándote tu sexo. Al terminar el diálogo, este retornará devolviendo a la actividad el resultado mediante un callback.

Observa el esquema siguiente:



Nuestra App tendrá dos clases, una con nuestra actividad principal “MyActivity” y otra con una clase llamada DialogoSexo que extiende de la clase DialogFragment

El código de la actividad principal será:

```
public class MyActivity extends Activity implements
DialogSexo.RespuestaDialogoSexo {

    @Override
    public void onRespuesta(String s) {
      Toast.makeText(getApplicationContext(),
s.Toast.LENGTH_LONG).show();
    }
    public void click(View v){
       DialogoSexo ds = new DialogoSexo();
       ds.show(getFragmentManager(),"Mi diálogo");
    }

    @Override
    protected void onCreate(Bundle savedInstanceState){
       . . .
    }
}
```

Propagamos una función que responda al clic del botón y creamos un objeto de la clase DialogoSexo, invocando al método **show** de un **Dialog,** que recibe como parámetros un objeto llamado FragmentManager que se obtiene con la llamada al método *getFragmentManager()* de la Activity, y una etiqueta con el texto identificador del diálogo.

Además hay que recibir la respuesta que nos envía el dialogo, esto lo haremos registrando una función de callback llamada **onRespuesta()**, método que implementamos a través de la interfaz llamada *RespuestaDialogo* y que programamos en la clase DialogoSexo. Este método onRespuesta crea un pequeño mensaje a modo de notificación que le aparece al usuario en el dispositivo a través de la clase Toast, cuya documentación puedes consultar en http://developer.android.com/guide/topics/ui/notifiers/toasts.html

Por otro lado la clase DialogSexo, tiene el siguiente código:

```
public class DialogoSexo extends DialogFragment{
   RespuestaDialogoSexo respuesta;
   /* Este método es llamado al hacer el show()* de la clase DialogFragment()*/
   @Override
   public Dialog onCreateDialog(Bundle savedInstanceState){
      //Usamos la clase Builder para construir el diálogo
```

```
        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
        //escribimos el título
        builder.setTitle("Pregunta muy importante:");
        //Escribimos la pregunta
        builder.setMessage("¿Eres una chica?");
        //Añadimos el botón de si y su acción asociada
        builder.setPositiveButton("SI!",new DialogInterface.onClickListener(){
            public void onClick(DialogInterface dialog, int id){
                respuesta.onRespuesta("Es una chica!");
            }
        });
        //Añadimos el botón de no y su acción asociada
        builder.setPositiveButton("NO!",new DialogInterface.onClickListener(){
            public void onClick(DialogInterface dialog, int id){
                respuesta.onRespuesta("Es un chico!");
            }
        });
        //Crear el AlertDialog y devolverlo
        return builder.create();
    }

    //interfaz para la comunicación entre la Actividad y el Fragmento
    public interface RespuestaDialogSexo{
        public void onRespuesta(String s);
    }

    //Se invoca cuando el fragmento se añade a la actividad
    @Override
    public void onAttach (Activity activity){
        super.onAttach(activity);
        respuesta = (RespuestaDialogoSexo)activity;
    }
}
```

Veamos paso a paso:

Se crean los diálogos mediante la clase AlertDialog.Builder, estableciendo título con *setTitle().*, el mensaje con *setMessage()*, y estableciendo los botones de si o no con su correspondiente respuesta a través de una función de callback de la interfaz *DialogInterface.OnClickListener*. Todo esto se hace reprogramando el método *onCreateDialog* de la clase DialogFragment, método que se invoca cuando la actividad principal ejecuta el show de nuestra clase DialogoSexo.

Cuando se pulsa uno de los botones del diálogo (si o no) se hace la llamada a la función *callback* onRespuesta que se ha programado en la actividad principal. Para ello:

- Cuando el diálogo se crea, el fragmento se une "Attach" a la actividad principal, el método *onAttach()* se ejecuta y podemos quedarnos con una referencia a la actividad. respuesta=(RespuestaDialogSexo)activity;
- Con esa referencia podemos invocar al método onRespuesta de la interfaz que obligamos a la actividad principal a implementar

```
builder.setPositiveButton("NO!",new DialogInterface.onClickListener(){
   public void onClick(DialogInterface dialog, int id){
      respuesta.onRespuesta("Es un chico!");
   }
});
```

La clase Builder de AlertDialog tiene muchos métodos interesantes y muy útiles, como por ejemplo:

- setItems(): coge de un array una lista de elementos a seleccionar
- setSingleChoiceItems() si quieres mostrar una lista con RadioButton
- setMultiChoiceItems() si quieres mostrar una lista con CheckBoxes

Además siempre puedes construir tu propio recurso de layout personalizado e inflar el diálogo a partir de este recurso personalizado.