# Tarea Investigación

***Dialog y DialogFragment.***

**¿Qué es un Dialog?**

La clase Android Dialog es la clase base para todos los tipos de controles dialogs que puede usar dentro de las clases Activity. Estos residen dentro del ciclo de vida de su activity. Surgen en el fondo, bloqueando la pantalla de su Activity, para captar la atención de los usuarios para que realicen alguna verificación o introducción de datos requeridos.

**¿Para que se usan los Dialogs?**

Algunos de los usos de los Dialogs son:

- Para informar sobre algún evento, notificación o progreso (“¡Tiene un correo!”, “Descargar mensaje 1 de 20”).
- Para obligar al usuario a confirmar alguna acción (“¿Está seguro que quiere borrar el archivo?”)
- Para pedir al usuario información necesaria para continuar (“inicio de sesión”).

Algunos programadores utilizan los Toast para enviar las notificaciones o mensajes al usuario. La diferencia entre un Toast y un Dialog es que en el Dialog requiere que el usuario interactúe con el mensaje por lo es seguro que este leerá la notificación o mensaje. Por lo que si el mensaje que se muestra es información importante para el usuario se usará Dialog, pero si la información que aparece no es de importancia se usaría un Toast que desaparecerá automáticamente.

**Estructura de los Dialogs.**

Los dialogs están compuestos por diferentes componentes y la mayoría de ellos son opcionales. Los componentes que tiene un Dialogs básico son:

- Un título. El cuál se añade con setTitle().
- Un mensaje o contenido, que se añadiría con setMessage().
- Botones para indicar la respuesta del usuario. Existen tres tipos de botones básicos los cuales son:
    - Positivo, que se añadiría con setPositiveButtom() e indicaría la confirmación del usuario.
    - Negativo, que se añadirá con setNegativeButtom() e indica la negación del usuario.
    - Neutral, que se añade con setNeutralButtom() e indica una opción neutral que se podrá establecer como “Recordar mas tarde” por ejemplo.

Se puede añadir un máximo de tres botones y no se puede repetir ninguno de los anteriores. A los botones se le añade un texto y un evento OnClick para controlarlo.

También existen otros tipos de diálogos como:

- DataPickerDialog. Es un diálogo que permite seleccionar fecha.
- TimePickerDialog. Es un diálogo que permite seleccionar una hora.

Existe también otra llamada ProgressDialog que muestra un diálogo con una barra de progreso, pero es recomendable utilizarla ya que para mostrar una barra de carga o progreso se utiliza ProgressBar.

## DialogFragment.

Un DialogFragment es un fragmento que flota en el centro de nuestra actividad. Son útiles a la hora de necesitar que el usuario responda a algo para continuar con la ejecución. Para crear un DialogFragment tenemos que heredar de la clase DialogFragment.

**Creación de un proyecto:**

Cuando creamos nuestra Activity le decimos que hereda de DialogFragment.

- onCreateDialog: Despues de extender de DialogFragment instanciamos el método onCreateDialog.
- AlertDialog.Builder: Una vez instaciado el onCreateDialog continuamos creando un AlertDialog con "AlertDialog.Builder".

A continuación podríamos asignarle un título al dialogo, un mensaje y los botones tal y como hemos visto anteriormente en Dialog.
Támbien tenemos la posibilidad de crear listas en el diálogo. Estas listas pueden ser de selección multiple (con checkbox), de selección única (con radio buttoms) o también podemos crear una lista en la que se seleccione un elemento como si fuese un botón. Para crear estas listas se utilizan los siguientes atributos:

- .setMultiChoiceItems: Este permite seleccionar más de una opción. A esto se le pasa un array con los datos que se desean mostrar y creado con anterioridad, también le pasamos los elementos que queremos que estén seleccionados por defecto (si es que queremos alguno) y por ultimo le asignaríamos un evento cuando se haga click en algún item "new DialogInterface.OnMultiChoiceClickListener(). Tambíen se le puede asignar botones y mensajes o titulo como hemos visto anteriormente.

- .setItems: Este muestra una lista de elementos a los que se le puede pulsar como si fuse un botón. A este también se le pasaría un array de datos y se le añadiría el evento OnClickListener.

- .setSingleChoiceItems: Este permite solo la selección de una opción de la lista. También se le pasaría un array de datos creado con anterioridad, después como segundo argumento se le añade cual quieres que esté

seleccionado por defecto, que en caso de no querer ninguno se establecería en -1 (ya que cuenta la posición del array) y por último el OnClickListener.

Por último una vez que hemos creado todos los botones y mensajes o listas que queramos antes de cerrar el onCreateDialog tenemos un return en el cual pondríamos el builder.create();

Con DialogFragment podemos crear diálogos personalizados. La forma de crear diálogos personalizados es la siguiente:

- Comenzamos extendiendo de DialogFragment como lo hemos hecho antes e instanciando el onCreateDialog.
- Continuamos construllendo el AlertDialog.Builder.
- A continuación creamos un LayoutInflater con getActivity y getLayoutInflater.
- Después creamos un View y lo igualamos al inflater creado antes y como parámetros le metemos el layout que tenemos que tener creado y personalizado a nuestro gusto, y como segundo parámetro le pasamos null.
- Ahora cogemos el nombre del alertDialog y le hacemos un setView(pasándole aquí el nombre del view creado anteriormente.), por ejemplo: builder.setView(v);.
- Después instanciamos los botones que tengamos en nuestro layout y le agregamos un setOnClickListener para darle una función a los botones.
- Por último devolvemos con return el builder.create();, el builder hacer referencia al AlertDialog.Builder creado cuando empezamos.

## Comparacion de DialogFragment y Dialog.

DialogFragment es un fragmento por lo que como ya sabemos esta cualidad nos proporciona algunas ventajas frente al Activity. La ventaja son las devoluciones de llamada del ciclo de vida. Así que con DialogFragment, puede ser muy potente y hace que su código sea mucho más limpio.
Con esto se puede proporcionar más inteligencia al diálogo y hacer que haga un trabajo inteligente por su cuenta en el lugar de la Actividad diciéndole qué hacer.

Como aclaración cabe destacar que se considera mejor practica utilizar DialogFragment que Dialog ya que su uso es más limpio y reutilizable al ser un fragmento.