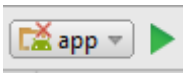
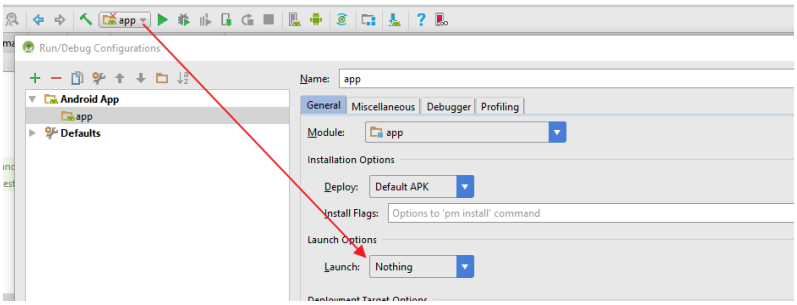
Ejercicio 1: Widget actualizable

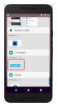
1. Crea un proyecto llamado P_84_Widget_2
2. En el ejemplo de App Widget desarrollado en la teoría hemos visto que la MainActivity solo se utilizaba para la instalación, podemos borrarla así como sus layouts predeterminados. **Es importante por tanto, borrar del AndroidManifest toda referencia a ella (si seleccionas clase java y layouts asociados a la vez el propio AS lo hace!).**

   Pero entonces AS parece que no nos deja probar la aplicación. Solución: hay que cambiar la configuración de lanzamiento:





3. Utilizando el asistente crea un Widget llamado MiWidget.
4. La imagen previa del widget es fea. Cámbiala por una de tu gusto (es el fichero /res/drawable/example_appwidget_preview.png) o directamente borra en el fichero mi_widget_info.xml la línea



```
android:previewImage="@drawable/example_appwidget_preview"
```

   y así pondrá nuestro conocido icono lanzador

5. El layout del widget también es feo!. Te aconsejo que utilices como background el fichero fondo.png que se ha obtenido de las antiguas "templates" recomendadas (está en la carpeta imágenes en recursos, optimízalo para las distintas densidades con nuestra herramienta Batch Drawable Importer). Puedes hacer otras modificaciones "estéticas" que estimes oportunas.
6. Modifica la frecuencia de actualización en el fichero mi_widget_info.xml para que pase a ser 30 minutos (es el valor mínimo permitido; si pones otro menor, ni caso, media hora; se supone que es para ahorro de batería).
7. Implementa la funcionalidad del widget (que es mostrar la hora) en la clase MiWidget modificando el método :

```
static void updateAppWidget(Context context, AppWidgetManager appWidgetManager,
                            int appWidgetId) {

    DateFormat hora = new SimpleDateFormat("hh:mm:ss");
    CharSequence widgetText = hora.format(new Date());
    // Construct the RemoteViews object
    RemoteViews views = new RemoteViews(context.getPackageName(),
R.layout.mi_widget);
    views.setTextViewText(R.id.appwidget_text, widgetText);

    // Instruct the widget manager to update the widget
    appWidgetManager.updateAppWidget(appWidgetId, views);
}
```

Es aquí donde habrá que obtener los datos que queremos visualizar, en nuestro ejemplo new Date(); generalmente será leer BD o servicio web, etc.

8. Prueba el widget en el emulador y si tienes suficiente paciencia comprueba que se actualiza la hora pasado el tiempo fijado.

Ejercicio 2: Widget actualizable con click sobre él

1. Deseamos modificar el widget anterior para que haciendo click sobre él se actualice la hora sin necesidad de tener que esperar.
2. Implementa la funcionalidad del widget cambiando el código indicado (intenta entenderlo, está explicado por comentarios):

```
static void updateAppWidget(Context context, AppWidgetManager appWidgetManager,
                            int appWidgetId) {

    DateFormat hora = new SimpleDateFormat("hh:mm:ss");
    CharSequence widgetText = hora.format(new Date());
    // Construct the RemoteViews object
    RemoteViews views = new RemoteViews(context.getPackageName(), R.layout.mi_widget);
    views.setTextViewText(R.id.appwidget_text, widgetText);

    //Configuramos el PendingIntent asociado al click sobre el textView
    Intent intent = new Intent(context, MiWidget.class);
    intent.setAction("ACCION_DEL_CLICK");
    intent.putExtra(AppWidgetManager.EXTRA_APPWIDGET_ID, appWidgetId);
    PendingIntent pendingIntent = PendingIntent.getBroadcast(context, 0,intent, 0);
    views.setOnClickPendingIntent(R.id.appwidget_text, pendingIntent);

    // Instruct the widget manager to update the widget (ESTE TIPO DE WIDGET)
    ComponentName esteWidget = new ComponentName(context, MiWidget.class);
    appWidgetManager.updateAppWidget(esteWidget, views);
}

…
@Override
public void onReceive(Context context, Intent intent) {
    if (intent.getAction().equals("ACCION_DEL_CLICK")) {
        Bundle extras = intent.getExtras();
        if (extras != null) {
            // Obtenemos el ID del widget a actualizar
            int widgetId = extras.getInt(AppWidgetManager.EXTRA_APPWIDGET_ID,
AppWidgetManager.INVALID_APPWIDGET_ID);
            // Comprobamos que el Id del widget es correcto
            if (widgetId != AppWidgetManager.INVALID_APPWIDGET_ID) {
                // Obtenemos el widget manager de nuestro contexto
                AppWidgetManager widgetManager = AppWidgetManager.getInstance(context);
                // Actualizamos el widget
                updateAppWidget(context, widgetManager, widgetId);
            }
        }
    } else
        super.onReceive(context, intent);
}
```

3. Prueba el widget en el emulador y comprueba su actualización por click.
4. Añade otro widget y comprueba que si actualizamos uno se actualiza también el otro. (En el siguiente ejercicio lo arreglaremos!).

Nota: Es necesario el método onReceive porque el propio widget debe actualizarse con el click a través de la llamada

```
Intent intent = new Intent(context, MiWidget.class);
intent.setAction("ACCION_DEL_CLICK");
intent.putExtra(AppWidgetManager.EXTRA_APPWIDGET_ID, appWidgetId);
PendingIntent pendingIntent = PendingIntent.getBroadcast(context, 0,intent, 0);
```

si lo que deseásemos es que se abra una actividad bastaría con

```
Intent intent = new Intent(context, ExampleActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(context, 0, intent, 0);
```

Ejercicio 3: Widget actualizable independientemente con click sobre él

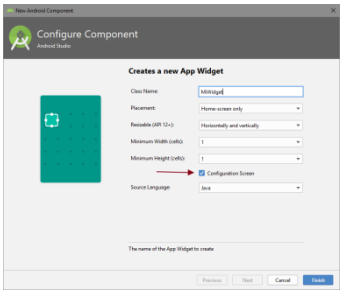
1. Implementa la funcionalidad del widget cambiando el código del método (señaladas las modificaciones con respecto al ejercicio anterior):

```
static void updateAppWidget(Context context, AppWidgetManager appWidgetManager,
                            int appWidgetId) {

    DateFormat hora = new SimpleDateFormat("hh:mm:ss");
    CharSequence widgetText = hora.format(new Date());
    // Construct the RemoteViews object
    RemoteViews views = new RemoteViews(context.getPackageName(),
R.layout.mi_widget);
    views.setTextViewText(R.id.appwidget_text, widgetText);

    //Configuramos el PendingIntent asociado al click sobre el textView
    Intent intent = new Intent(context, MiWidget.class);
    intent.setAction("ACCION_DEL_CLICK");
    intent.putExtra(AppWidgetManager.EXTRA_APPWIDGET_ID, appWidgetId);
    PendingIntent pendingIntent = PendingIntent.getBroadcast(context,
appWidgetId,intent, 0);
    views.setOnClickPendingIntent(R.id.appwidget_text, pendingIntent);

    // Instruct the widget manager to update the widget
    //ComponentName esteWidget = new ComponentName(context, MiWidget.class);
    appWidgetManager.updateAppWidget(appWidgetId, views);
}
```

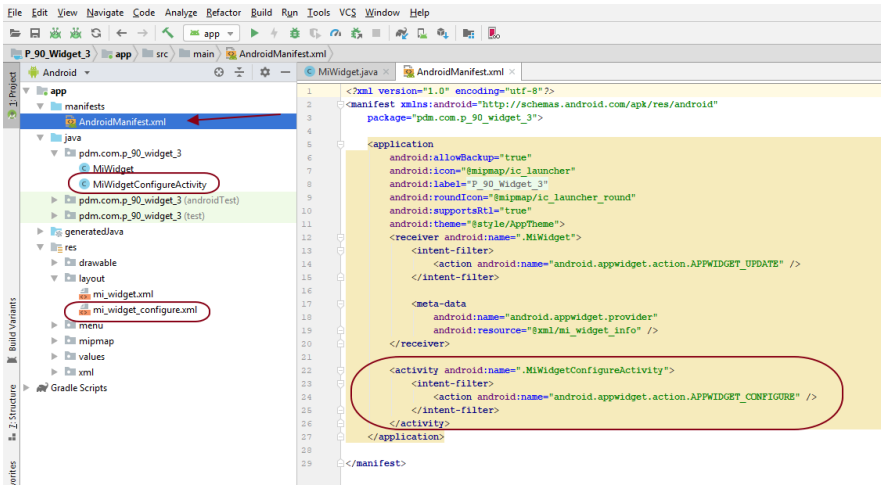
2. Añade varios widget en el emulador y comprueba sus actualizaciones por click independientes.

Ejercicio 4: Widget configurable en la instalación

1. Crea un proyecto llamado P_85_Widget_3
2. Borra (si quieres) la MainActivity y su layout y cambia el modo de lanzamiento.
3. Crea un widget llamado MiWidget que tenga actividad de configuración



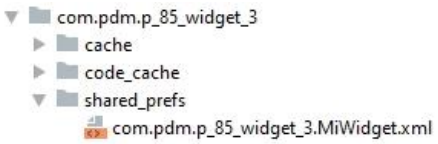
4. Observa todo el nuevo trabajo realizado por el asistente



5. Ejecuta para comprobar que en el widget aparece el texto que se ha tecleado cuando se añade el widget
6. Observa el código de la clase MiWidgetConfigureActivity, utilizan nuestra ya conocida clase SharedPreferences para guardar los datos introducidos.

a) El nombre del fichero de preferencias se fija con la constante

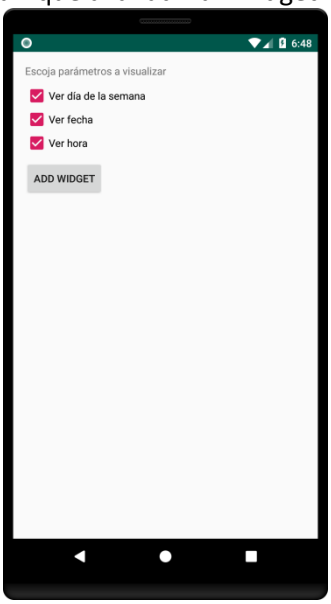
```
private static final String PREFS_NAME = "pdm.com.p_85_widget_3.MiWidget";
```



b) Como es posible que se añadan varios widgets, la clave para cada uno de los textos se fija con un prefijo seguido del identificador del widget que se trate:

```
private static final String PREF_PREFIX_KEY = "appwidget_";
…

prefs.putString(PREF_PREFIX_KEY + appWidgetId, text);
```

c) Estudia el código que está debidamente comentado
d) Realiza las modificaciones oportunas (seguro que piensas menos "liosamente" que el asistente de AS!) para conseguir que al añadir un widget nos solicite



Y podamos obtener widgets como los de la imagen