


Open · Opened 3 weeks ago by  **Rubén Montero**

Clicar un botón

Resumen

- Veremos cómo responder a un *click* de un usuario en un botón
- Hablaremos brevemente de interfaces Java
- Mostraremos un *Toast* si el usuario clicla en el botón superior

Descripción

Responder al *click* de un usuario en un componente es muy fácil.

Basta con usar un objeto `OnClickListener` :

[OnClickListener](#)

Es una interfaz Java que declara un único método:

```
public void onClick(View v)
```

Recuerda que las interfaces, en Java, son *firmas de métodos*. Si una clase *tiene* esos métodos, se dice que *conforma* ó *implementa* esa interfaz Java.

Por ejemplo, una interfaz Java *válida* sería:

```
public interface InterfazEjemploNumero {  
  
    public int devolverUnNumero(boolean positivo);  
}
```

Como ves, el método acaba en punto y coma (`;`) en vez tener un cuerpo entre llaves (`{ }`). Eso es porque en la interfaz sólo está presente la *firma* del método. Son las *clases* quienes implementan las *interfaces* especificando `implements` , por ejemplo:

```
public class ClaseImplementadora implements InterfazEjemploNumero {  
  
    @Override  
    public int devolverUnNumero(boolean positivo) {  
        if (positivo) {  
            return 10;  
        } else {  
            return -10;  
        }  
    }  
}
```

La anotación `@Override` sirve para indicar sobrescritura de un método de *superclase* o de una *interfaz*. Este doble uso puede despistar a veces, pero no le demos mucha importancia.

La tarea

Añade una nueva clase. Para ello, **haz click** derecho en `com.afundacion.fp.library` > **New** > **Java Class**. **Dale** el nombre `ClickHandler` .

Añade el siguiente contenido a la clase (te resultará fácil de entender si ya eres familiar con los [POJOs](#)):

```
package com.afundacion.fp.library;  
  
import android.content.Context;  
  
public class ClickHandler {  
    private Context context;  
  
    public ClickHandler(Context context) {  
        this.context = context;  
    }  
}
```

```
}
}
```

Ahora, en `MainActivity.java`, **crea** una nueva instancia de `ClickHandler` después del `Toast` de la tarea anterior. **Así:**

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Toast.makeText(this, "¡Primera tostada del día!", Toast.LENGTH_SHORT).show();
    ClickHandler myHandler = new ClickHandler(this); // Instanciamos un nuevo ClickHandler
}
```

(Una `Activity` es un `Context`, por eso, podemos pasar `this` en el constructor)

Ahora vamos a identificar y asociar nuestro botón al `ClickHandler`.

En `activity_main.xml`, el `<Button>` es una etiqueta.

En Java, será una *variable* de tipo `Button` que no *inicializamos* usando `new`, sino usando `findViewById`.

Adelante, **añade** la siguiente línea:

```
Button myButton = findViewById(R.id.toastButton);
```

(Como puedes ver, `R.id.toastButton` es el mismo `android:id` que habíamos indicado en el XML)

El penúltimo paso

Vamos a unir nuestro `myHandler` al botón. Eso se hace *invocando* `setOnClickListener` sobre el botón.

Hazlo así:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Toast.makeText(this, "¡Primera tostada del día!", Toast.LENGTH_SHORT).show();
    ClickHandler myHandler = new ClickHandler(this);
    Button myButton = findViewById(R.id.toastButton);
    myButton.setOnClickListener(myHandler);
}
```

Hemos terminado con `MainActivity.java`

No te preocupes por la alerta que aparece en la última línea:

```
RequiredType: OnClickListener
ProvidedType: ClickHandler
```

Java está indicando: "Eh, quería que pasases un `OnClickListener` a este método y me has pasado un `ClickHandler` en su lugar" ¡Lleva razón!

Vamos a...

...hacer que `ClickHandler` sea también `OnClickListener`

En `ClickHandler.java`, **añade** `implements View.OnClickListener`:

```
import android.content.Context;
import android.view.View;

public class ClickHandler implements View.OnClickListener{
    private Context context;

    public ClickHandler(Context context) {
        this.context = context;
    }
}
```

Verás que la primera línea aparece subrayada en rojo.

Class 'ClickHandler' must (...) implement abstract method 'onClick(View)'

¡Claro! Tras declarar que se *implementa* una interfaz... ¡Debemos implementarla!

Añade el método `onClick`:

```
public class ClickHandler implements View.OnClickListener{
    private Context context;

    public ClickHandler(Context context) {
        this.context = context;
    }

    @Override
    public void onClick(View view) {

    }
}
```

Y, para terminar, en el método `onClick`, **haz** que se lance un `Toast`. **Así**:

```
@Override
public void onClick(View view) {
    Toast.makeText(this.context, "¡Rápido, coge un plato!", Toast.LENGTH_LONG).show();
}
```

¡Enhorabuena! Has conectado tu primer botón en Android.

Ya puedes lanzar `app` y hacer *click* en "Ver tostada", ¿qué ves?

Por último

Sube tus cambios al repositorio en un nuevo *commit*.



Rubén Montero @ruben.montero changed milestone to [%Sprint 1](#) 3 weeks ago



Ania Blanco @ania.blanco mentioned in commit [35e7638f](#) 1 week ago