Tabs, menús por pestañas

Este es otro tipo de menú. Elegiremos el menú adecuado en función de las necesidades de nuestra aplicación y de las secciones que este tenga.

En este caso vamos a usar un Button Navigation Activity, que nos permite navegar a través de las opciones que aparecen por defecto abajo del dispositivo.

No es normal usar este tipo de navegación y el anterior en la misma pantalla, ya que pueden llevar a confusión dos menús, aunque en alguna ocasión puede darse.

Planteamos las mismas opciones que en el ejemplo anterior, para ello veamos los cambios que hemos de realizar.

Creamos una aplicación con un Button Navigation Activity.

En el activity_main.xml, vemos que tiene un FrameLayout, donde tenemos que tener un id/content, y que carga un texto, pero nosotros podemos usarlo como un contenedor para cargar fragmentos. La parte del TextView lo borramos porque no nos hace falta.

Debajo tenemos un BottomNavigationView con un @menu/navigation, es decir un menú de navegación con las opciones que podemos modificar.

Dejamos solo dos opciones y las modificamos, como en el ejemplo anterior por averías y talleres.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/navigation_averias"
        android:icon="@drawable/ic_dashboard_black_24dp"
        android:title="Averías" />

    <item
        android:id="@+id/navigation_talleres"
        android:icon="@drawable/ic_notifications_black_24dp"
        android:title="Talleres" />

</menu>
```

Para los id lo que hacemos para que se actualicen los nombres que usamos en el código de mi aplicación, es marcarlos y dar le a la opción refactor. De esta forma en el MainActivity veremos que se han actualizado.

En el método *OnNavigationItemSelectedListener* borramos la primera opción, ya que solo tenemos dos, y ahora, creamos un fragmento a null y en cada opción del switch haríamos que se cargase el fragmento adecuado.

Debajo del switch, si se ha pulsado alguno que lo cambie por el adecuado.

Nota que los return del switch los borramos y ponemos después de actualizar el fragmento.

Para que esto funcione hemos de crear previamente dos fragmentos con fragmento blank (desmarcar opciones)

```
private BottomNavigationView.OnNavigationItemSelectedListener mOnNavigationIt
        = new BottomNavigationView.OnNavigationItemSelectedListener() {

    @Override
    public boolean onNavigationItemSelected(@NonNull MenuItem item) {
        Fragment f = null;

        switch (item.getItemId()) {
            case R.id.navigation_averias:
                f = new AveriasFragment();
                break;

            case R.id.navigation_talleres:
                f = new TalleresFragment();
                break;
        }

        if(f!=null) {
            getSupportFragmentManager()
                    .beginTransaction()
                    .replace(R.id.content,f)
                    .commit();

            return true;
        }
```

Además en el onCreate vamos a hacer que se cargue por defecto uno de los dos fragmentos como hemos venido haciendo en los ejemplos anteriores

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    BottomNavigationView navigation = (BottomNavigationView) findViewById(R.id.na
    navigation.setOnNavigationItemSelectedListener(mOnNavigationItemSelectedListe

    getSupportFragmentManager()
            .beginTransaction()
            .add(R.id.content,new AveriasFragment())
            .commit();
}
```

Ya sólo quedaría cambiar los iconos.

En este caso hemos usado solo dos iconos. En este tipo de menús como mucho se usan tres opciones, si queremos usar más, lo más correcto sería usar un menú como el del ejemplo anterior, un Navigation Drawer