M7 - Estats

Definició d’estat

- “L’estat d’una app és qualsevol valor que pot canviar amb el pas del temps. [...] Abarca des d’una base de dades de Room fins a una variable d’una classe”. Definició de Google.

- La UI de la nostra app està associada a un estat. Cada vegada que un estat canvia, la vista canvia automàticament.

Definició de composició

- Les vistes amb que treballa Compose són **declaratives**. Què vol dir això?
- Si, per exemple, tenim un text, i volvem canviar el seu contingut, amb Compose no es modificaria un atribut del text i ja, es tornaria a crear l'objecte amb el nou valor.
- Cada vegada que modifiquem un atribut d'un element que admet composició, modifiquem l'estat, i es produeix una **recomposició**.

Exemple

- Observa el següent codi:

```
@Composable
fun MyStateExample(){
   var counter = 0
   Column(Modifier.fillMaxSize(),
       verticalArrangement = Arrangement.Center,
       horizontalAlignment = Alignment.CenterHorizontally) {
       Button(onClick = { counter += 1 }) {
           Text(text = "Push")
       }
       Text(text = "$counter times clicked")
   }
}
```

Exemple

- El codi anterior té per objectiu crear un comptador de “clicks”.
- Cada vegada que polsem sobre el botó, es modificarà el text incrementant el nombre de “clicks”.
- Si provem el codi veurem que no funciona.

Solució (I)

- Observa el següent codi:

```
@Composable
fun MyStateExample(){
   var counter = mutableStateOf(0)
   Column(Modifier.fillMaxSize(),
       verticalArrangement = Arrangement.Center,
       horizontalAlignment = Alignment.CenterHorizontally) {
       Button(onClick = { counter.value += 1 }) {
           Text(text = "Push")
       }
       Text(text = "${counter.value} times clicked")
   }
}
```

Solució (I)

- Hem creat la variable “`counter`” amb la funció “*`mutableStateOf`*”, que rep com a paràmetre el valor inicial que volem.
- Si proves el codi, veuràs que tampoc funciona. Per què?
- Si et fixes, cada vegada que recomposem l’element, estem creant la variable “`counter`” amb el valor 0. Hem de trobar alguna manera de que no es torni a crear la variable en cada recomposició i que es mantingui el valor de l’estat anterior.

Solució (definitiva)

- Observa el següent codi:

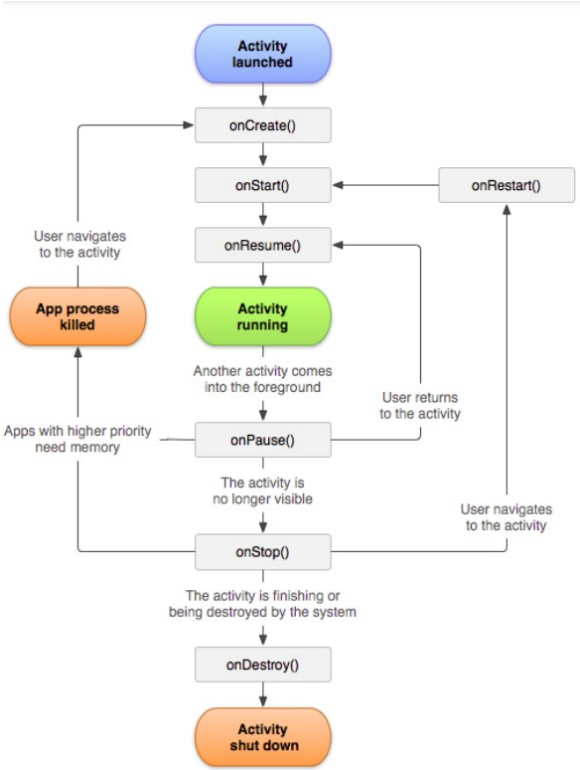
```
@Composable
fun MyStateExample(){
   var counter = remember {mutableStateOf(0)}
   Column(Modifier.fillMaxSize(),
       verticalArrangement = Arrangement.Center,
       horizontalAlignment = Alignment.CenterHorizontally) {
       Button(onClick = { counter.value += 1 }) {
           Text(text = "Push")
       }
       Text(text = "${counter.value} times clicked")
   }
}
```

Solució (definitiva)

- Amb la paraula *“`remember`”* fem que es guardi el valor de la variable *“counter”*. Així evitarem que es reinicii cada vegada que recomposem la vista.
- Prova el codi. Funciona! Però... què passa si rotem el dispositiu?

Solució (definitiva II)

- **Cicle de vida de les activities**:



- Quan rotem la pantalla, l'activity es destrueix i es torna a crear.

Solució (definitiva II)

- Observa el següent codi:

```
@Composable
fun MyStateExample(){
   var counter = rememberSaveable {mutableStateOf(0)}
   Column(Modifier.fillMaxSize(),
       verticalArrangement = Arrangement.Center,
       horizontalAlignment = Alignment.CenterHorizontally) {
       Button(onClick = { counter.value += 1 }) {
           Text(text = "Push")
       }
       Text(text = "${counter.value} times clicked")
   }
}
```

Solució (definitiva II)

- Amb *“`rememberSaveable`”* podem mantenir l’estat tot i que la vista es destrueixi.

Solució (definitiva III)

- Per finalitzar, pots utilitzar el següent codi:

```
@Composable
fun MyStateExample(){
   var counter by rememberSaveable {mutableStateOf(0)}
   Column(Modifier.fillMaxSize(),
       verticalArrangement = Arrangement.Center,
       horizontalAlignment = Alignment.CenterHorizontally) {
       Button(onClick = { counter += 1 }) {
           Text(text = "Push")
       }
       Text(text = "${counter} times clicked")
   }
}
```

Solució (definitiva III)

- Al codi anterior hem modificat la manera de crear la variable *“counter”*. En lloc de l’igual (=) posem l’operador *“by”*.
- Així podrem accedir i modificar el valor de la variable *“counter”* sense haver de posar *“.value”* (escriurem menys).
- Tingues en compte que hauràs de fer les següents importacions de forma manual:

```
import androidx.compose.runtime.getValue
import androidx.compose.runtime.setValue
```