Component TextField

Component TextField

- Un TextField és un component que s'utilitza perquè l'usuari pugui introduir un text.
- Per afegir un TextField a la nostra vista podem fer:

```
var myText by remember{ mutableStateOf("") }
TextField(value = myText, onValueChange = { myText = it
})
```

- Com pots veure al codi anterior, per utilitzar un TextField fem ús dels estats.

Component TextField

- La variable "`myText`" contindrà el valor del TextField (en l'exemple el posem inicialment buit).
- El constructor del TextField té dos paràmetres:
  - "`value`" que serà el valor del TextField.
  - "`onValueChanged`" on crearem una funció lambda amb el que volem fer quan l'usuari canviï el valor del TextField. El nou contingut estarà a la variable "`it`". A l'exemple, assignem el valor de "`it`" a la nostra variable "`myText`".

Component TextField

- És útil, i ajuda a l'usuari, indicar per a què serveix el nostre TextField. Podem afegir una etiqueta per fer-ho:

```
TextField(
    value = myText,
    onValueChange = { myText = it },
    label = { Text(text = "Enter your name") }
)
```

Component TextField

- De vegades tindrem un TextField que ens interessa que només admeti números. En aquest cas podem utilitzar la variable "*keyboardOptions*" i especificar el valor de "*keyboardType*". D'aquesta manera apareixerà un teclat específic com, per exemple, el teclat numéric:

```
TextField(
    value = myText,
    onValueChange = { myText = it },
    label = { Text(text = "Enter your year of birth") },
    keyboardOptions = KeyboardOptions(keyboardType =
KeyboardType.Number)
)
```

Component TextField

- També podem afegir filtres al nostre TextField, de manera que puguem controlar el seu comportament segons el valor introduït per l'usuari:

```
val pattern = remember { Regex("^\\d+\$") }
TextField(
    value = myText,
    onValueChange = {
        if (it.isEmpty() || it.matches(pattern)) myText = it
    },
    label = { Text(text = "Enter your year of birth") },
    keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Number)
)
```

Component TextField: exemple password

- Observa aquest exemple:

```
@Composable
fun PasswordField() {
    var password by remember { mutableStateOf("") }
    var passwordVisibility by remember { mutableStateOf(false) }
    TextField(
        value = password,
        onValueChange = { password = it },
        label = { Text("Enter your password") },
        maxLines = 1,
        singleLine = true,
        ...
}
```

Component TextField: exemple password

- Continuació:

```
TextField(
        ...
        keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Password),
        trailingIcon = {
            val image = if (passwordVisibility) { Icons.Filled.VisibilityOff }
                else { Icons.Filled.Visibility }
            IconButton(onClick = { passwordVisibility = !passwordVisibility }) {
                Icon(imageVector = image, contentDescription = "Password
visibility")
            }
        },
        visualTransformation = if (passwordVisibility) { VisualTransformation.None
}
            else { PasswordVisualTransformation() }
    )
```

Component OutlinedTextField

- El component OutlinedTextField funciona exactament igual que el TextField, però afegeix una vora al voltant del TextField:

```
var myText by remember { mutableStateOf("") }
OutlinedTextField(
    value = myText,
    onValueChange = { myText = it },
    label = { Text("Enter your name") })
```

Component OutlinedTextField

- Prova el següent codi:

```
OutlinedTextField(
    value = myText,
    onValueChange = { myText = it },
    label = { Text("Enter your name") },
    colors = TextFieldDefaults.outlinedTextFieldColors(
        focusedBorderColor = Color.Green,
        unfocusedBorderColor = Color.Black
    ))
```

- Amb la propietat "`colors`" podem especificar colors diferents per quan l'OutlinedTextField tingui el focus o no.