Taller
Electiva Desarrollo de proyectos de software para aplicaciones móviles.
Kathleen Samantha Vargas Avendaño.

Text
¿Para qué sirve?: Su función principal es mostrar cualquier tipo de texto en la pantalla, desde etiquetas cortas hasta párrafos largos. También permite la personalización del texto.
¿Cómo se crea?: Su forma más básica es Text(text = "Hola, mundo!") se le pueden agregar algunos modificadores cómo fontSize, color, fondStyle entre otras.
Biblioteca de donde se importa:androidx.compose.material3.*

Button
¿Para qué sirve?: Es un componente básico que permite crear interfaces de usuario de forma más concisa y expresiva además de permitir personalizar el botón usando modificadores.
¿Cómo se crea?: Su forma básica es Button(onClick = { /* Acción al hacer clic */ }) {
    Text(text = "Haz clic aquí")
} dónde button es el componente principal, OnClick define la acción que hará después de dar click, text muestra el texto dentro del botón.
Biblioteca de donde se importa:androidx.compose.material3.*

OutlinedButton
¿Para qué sirve?:es un tipo de botón que presenta un contorno en lugar de un relleno sólido.
¿Cómo se crea?: Un ejemplo es OutlinedButton(
    onClick = { /* Acción al hacer clic */ },
    border = BorderStroke(1.dp, Color.Gray),
    shape = RoundedCornerShape(8.dp)
) {
    Text(text = "Cancelar")
}
OutlinedButton es el componente principal, con Onclick se define la acción a ejecutar después de dar click, border configura el borde del botón, shape define la forma del botón y text muestra el texto dentro del botón.
Biblioteca de donde se importa: androidx.compose.material3.OutlinedButton

TextField
¿Para qué sirve?: Permite al usuario ingresar y editar texto.
¿Cómo se crea?: Un ejemplo es TextField(
    value = text,
    onValueChange = { newText ->
        text = newText
    },
    label = { Text("Nombre") }
)

Donde TextField es el componente principal de un campo de texto editable, value contiene el texto actual que se muestra en el campo, onValueChange actualiza el estado de la variable text cada vez que el usuario cambia el texto en el campo.
Biblioteca de donde se importa:

Image
¿Para qué sirve?: El elemento `image` se utiliza para mostrar imágenes en una interfaz de usuario.
¿Cómo se crea?: Un ejemplo es Image(

```
    painter = painterResource(id = R.drawable.my_image),
    contentDescription = "Mi imagen",
    modifier = Modifier
        .fillMaxWidth()
        .height(200.dp)
)
```

Donde image es el componente principal, painter especifica la fuente de la imagen, contentDescription proporciona una descripción de la imagen y modifier aplica mpodificaciones a la imagen.
Biblioteca de donde se importa: androidx.compose.ui.graphics.painter.Painter, androidx.compose.ui.res.painterResource

DropdownMenu
¿Para qué sirve?: se utiliza para crear menús desplegables que ofrecen al usuario una lista de opciones para seleccionar.
¿Cómo se crea?: Un ejemplo es

```
var expanded by remember { mutableStateOf(false) }
val items = listOf("Opción 1", "Opción 2", "Opción 3")

Box {
    Button(onClick = { expanded = !expanded }) {
        Text(text = "Seleccionar")
    }
    DropdownMenu(
        expanded = expanded,
        onDismissRequest = { expanded = false }
    ) {
        items.forEach {
            DropdownMenuItem(onClick = { /* Acción al seleccionar */ }) {
                Text(text = it)
            }
        }
    }
}
```

Donde expanded controla si el menu es abierto o cerrado, items es la lista de elementos que se mostrara en el menu, Button es el boton que al ser presionado abre o cierra el menu, DropDownMenu es el componente principal del menu desplegable y DropDownMenuItem Es cada elemento individual del menu

Biblioteca de donde se importa:androidx.compose.material3.DropdownMenu, androidx.compose.material3.DropdownMenuItem

Checkbox
¿Para qué sirve?: sirve para crear casillas de verificación que permiten al usuario seleccionar o deseleccionar una opción.
¿Cómo se crea?: Un ejemplo es var checkedState by remember { mutableStateOf(false) }

```
Checkbox(
    checked = checkedState,
    onCheckedChange = { checkedState = it },
    enabled = true
)
```

Donde heckbox Crepresenta una casilla de verificación, checked indica si la casilla esta marcada o no, onCheckedChange se ejecuta cuando el usuario marca o desmarca la casilla y checkedState almacena si la casilla está marcada o no
Biblioteca de donde se importa:androidx.compose.material3.*

Switch
¿Para qué sirve?: Es un componente de interfaz de usuario que te permite crear un control visual que se puede alternar entre dos estados: encendido (true) y apagado (false).
¿Cómo se crea?: Un ejemplo es @Composable

```
fun MyScreen() {
    var isChecked by remember { mutableStateOf(false) }

    Switch(
        checked = isChecked,
        onCheckedChange = { isChecked = it },
        colors = SwitchDefaults.colors(
            checkedThumbColor = Color.Green,
            uncheckedThumbColor = Color.Gray
        )
    )
}
```

Donde isChecked almacena el estado actual del switch, el componente switch se configura con el estado actual.
Biblioteca de donde se importa: androidx.compose.material.Switch

RadioButton
¿Para qué sirve?: Son elementos de la interfaz de usuario que permiten al usuario seleccionar una única opción de un conjunto de opciones mutuamente excluyentes.
¿Cómo se crea?: Un ejemplo es @Composable

```
fun MyScreen() {
    var selectedOption by remember { mutableStateOf("Opción 1") }

    RadioGroup(
```

```
        options = listOf("Opción 1", "Opción 2", "Opción 3"),
        selectedOption = selectedOption,
        onSelectedChange = { selectedOption = it }
    )
}
```

Donde RadioGroup agrupa los RadioButton para asegurar que solo se pueda seleccionar uno a la vez, Options es una lista con opciones diferentes, selectedOption almacena la opción actualmente seleccionada y onSelectedChange se ejecuta cuando el usuario selecciona una nueva opción.
Biblioteca de donde se importa: androidx.compose.material.RadioButton

Slider
¿Para qué sirve?: Es una herramienta muy útil para permitir que los usuarios seleccionen un valor dentro de un rango continuo.Por ejemplo control de volumen o brillo.
¿Cómo se crea?: Por ejemplo

```
@Composable
fun MySlider() {
    var sliderValue by remember { mutableStateOf(0f) } // Estado para almacenar el valor del Slider

    Column(
        modifier = Modifier.padding(16.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {
        Text(text = "Valor: ${sliderValue.toInt()}") // Muestra el valor actual del Slider

        Slider(
            value = sliderValue,
            onValueChange = { newValue -> sliderValue = newValue }, // Actualiza el valor
            valueRange = 0f..100f, // Rango de valores permitido
            steps = 9 // Divide el rango en pasos (opcional)
        )
    }
}
```

Donde value Controla el valor actual del slider.
onValueChange Callback que se ejecuta cada vez que el usuario mueve el slider.
valueRange Especifica el rango mínimo y máximo del slider.
steps Define los pasos discretos entre los valores (opcional; si no se especifica, el slider será continuo).
Biblioteca de donde se importa:androidx.compose.material.Slider

FloatingActionButton
¿Para qué sirve?: Se utiliza para destacar la acción más importante o frecuente que el usuario puede realizar en una pantalla.
¿Cómo se crea?:
Biblioteca de donde se importa: androidx.compose.material.FloatingActionButton

Row

¿Para qué sirve?: Alinea los elementos de izquierda a derecha, creando una disposición horizontal.
¿Cómo se crea?: Por ejemplo @Composable
```
fun MyScreen() {
    Row {
        Text(text = "Elemento 1")
        Text(text = "Elemento 2")
        Text(text = "Elemento 3")
    }
}
```
Biblioteca de donde se importa: androidx.compose.foundation.layout.Row

Column
¿Para qué sirve?: organiza los elementos de interfaz de usuario de forma vertical.
¿Cómo se crea?: Por ejemplo @Composable
```
fun MyScreen() {
    Column {
        Text(text = "Elemento 1")
        Text(text = "Elemento 2")
        Text(text = "Elemento 3")
    }
}
```
Biblioteca de donde se importa: androidx.compose.foundation.layout.*

LazyColumn
¿Para qué sirve?: Es un componente de diseñado para mostrar listas verticales de elementos de forma eficiente. Es ideal cuando se trabaja con grandes conjuntos de datos, ya que solo renderiza los elementos visibles en la pantalla en un momento dado
¿Cómo se crea?: Por ejemplo
```
LazyColumn {
    items(items = listOf("Item 1", "Item 2", "Item 3")) { item ->
        Text(text = item)
    }
}
```
Biblioteca de donde se importa: androidx.compose.foundation.lazy.LazyColumn

Card
¿Para qué sirve?: Permite crear elementos visuales que se asemejan a tarjetas físicas
¿Cómo se crea?: Por ejemplo
```
Card(
    modifier = Modifier
        .fillMaxWidth()
        .padding(16.dp),
    elevation = 4.dp
) {
    Text(text = "Esta es una tarjeta")
}
```
Biblioteca de donde se importa: androidx.compose.foundation.layout.Card

Scaffold
¿Para qué sirve?: Proporciona una estructura básica y consistente para pantallas
¿Cómo se crea?: Por ejemplo Scaffold(

```
    topBar = {
        TopAppBar(title = { Text(text = "Mi aplicación") })
    },
    floatingActionButton = {
        FloatingActionButton(onClick = { /* Acción a realizar */ }) {
            Icon(imageVector = Icons.Default.Add, contentDescription = "Agregar")
        }
    },
    content = {
        Text(text = "Contenido principal de la pantalla")
    }
)
```

Biblioteca de donde se importa: androidx.compose.material.Scaffold

Surface
¿Para qué sirve?: Representa un elemento de material design que simula una superficie física, como una hoja de papel o una pantalla.
¿Cómo se crea?: Por ejemplo Surface(

```
    color = MaterialTheme.colors.surface,
    shape = RoundedCornerShape(8.dp),
    elevation = 4.dp
) {
    // Contenido de la superficie
    Text(text = "Hola, mundo!")
}
```

Donde color define el color de fondo de la superficie, shape establece la forma de los bordes de la superficie, elevation Determina la elevación de la superficie, lo que afecta a la sombra y a la percepción de profundidad.
Biblioteca de donde se importa: androidx.compose.material.Surface