



PROGRAMOWANIE URZĄDZEŃ MOBILNYCH

WYKŁAD 5 Jetpack Compose

- Fundamenty
- Komponenty
- @Composable
- MutableState


Jetpack Compose

MainActivity.kt

```
@Composable
fun TextSample() {
    Card(shape = RoundedCornerShape(4.dp)) {
        Row(verticalGravity = Alignment.CenterVertically) {
            Image(
                vectorResource(R.drawable.ic_jetpack_compose_image)
            )
            Column(modifier.preferredSizeIn(minWidth = 360.dp)) {
                Text(
                    text = "Welcome to Jetpack",
                    style = MaterialTheme.typography.h5,
                    modifier = Modifier.padding(start=16.dp)
                )
            }
        }
    }
}

@Preview
@Composable
fun DefaultPreview() {
    ComposeAppTheme(darkTheme = false) {
        TextSample()
    }
}
```

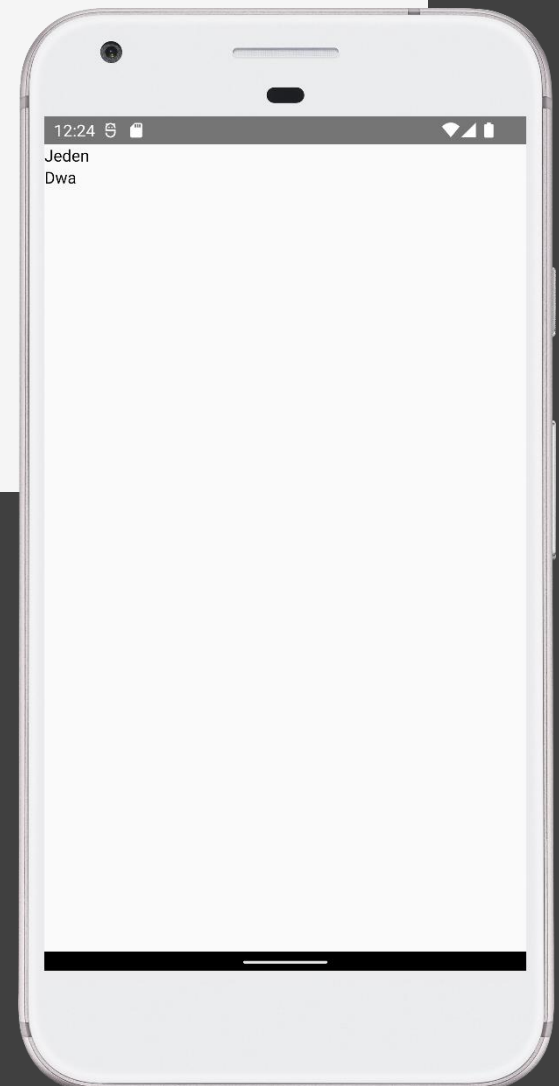
DefaultPreview



Welcome to Jetpack

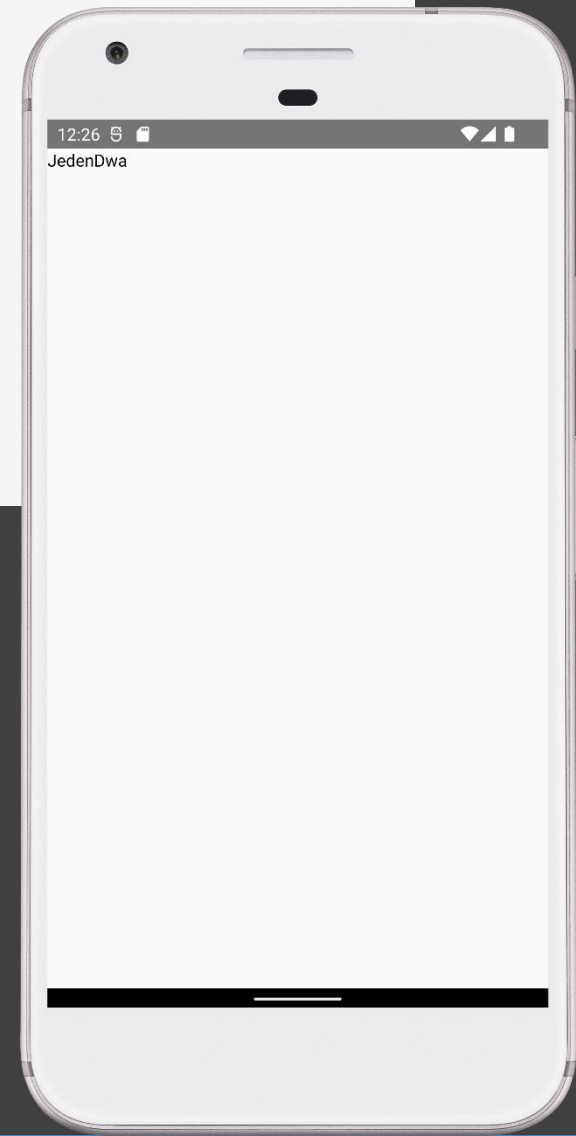
Jetpack Compose

```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContent {  
        Column {  
            Text(text = "Jeden")  
            Text(text = "Dwa")  
        }  
    }  
}
```



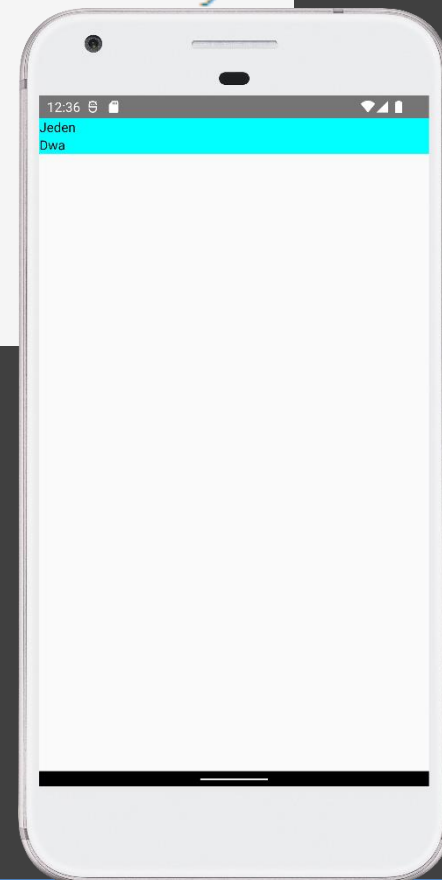
Jetpack Compose

```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContent {  
        Row {  
            Text(text = "Jeden")  
            Text(text = "Dwa")  
        }  
    }  
}
```



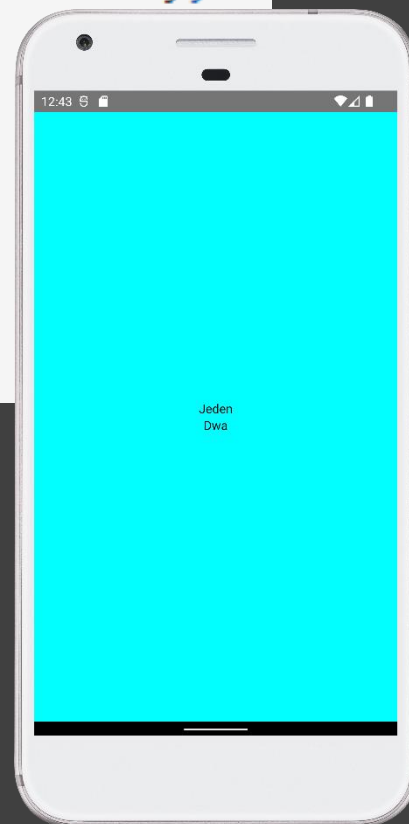
Jetpack Compose

```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContent {  
        Column(  
            modifier = Modifier  
                .fillMaxWidth() // szerokość kolumny  
                .background(Color.Cyan) // kolor tła kolumny  
        ) {  
            Text(text = "Jeden")  
            Text(text = "Dwa")  
        }  
    }  
}
```



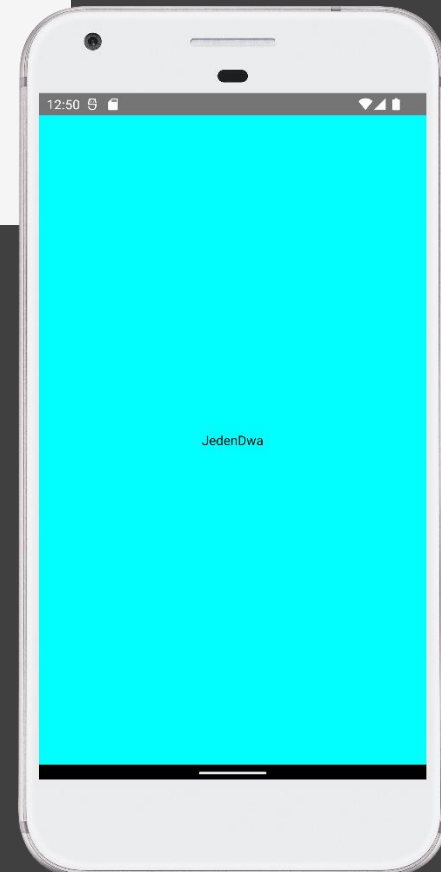
Jetpack Compose

```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContent {  
        Column(  
            modifier = Modifier  
                .fillMaxSize()  
                .background(Color.Cyan),  
            horizontalAlignment = Alignment.CenterHorizontally,  
            verticalArrangement = Arrangement.Center  
        ) {  
            Text(text = "Jeden")  
            Text(text = "Dwa")  
        }  
    }  
}
```



Jetpack Compose

```
Row(  
    modifier = Modifier  
        .fillMaxSize()  
        .background(Color.Cyan),  
    horizontalArrangement = Arrangement.Center,  
    verticalAlignment = Alignment.CenterVertically  
) {  
    Text(text = "Jeden")  
    Text(text = "Dwa")  
}
```



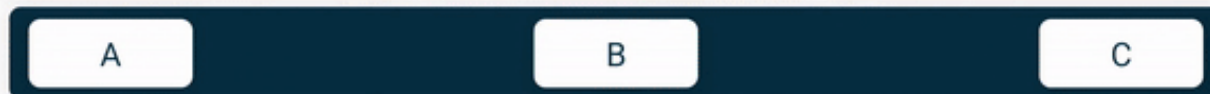
```
Column(  
    modifier = Modifier.fillMaxSize(),  
    verticalArrangement = Arrangement.spacedBy(8.dp)  
) {  
    Text(  
        text = "Element 1",  
        modifier = Modifier.size(50.dp)  
    )  
    Text(  
        text = "Element 2",  
        modifier = Modifier.size(80.dp)  
    )  
    Text(  
        text = "Element 3",  
        modifier = Modifier.size(100.dp)  
    )  
}
```


Jetpack Compose

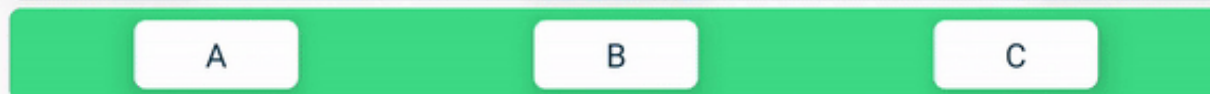
Equal Weight



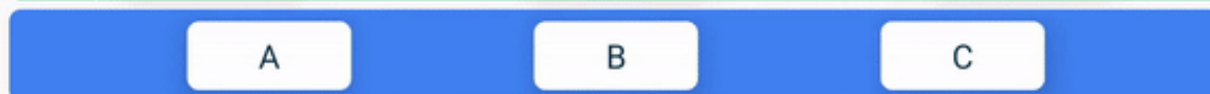
Space Between



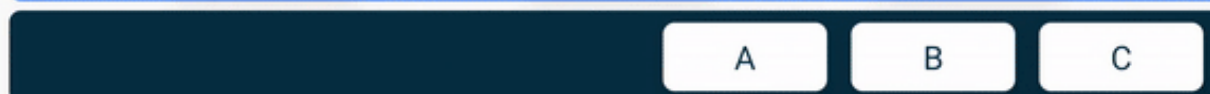
Space Around



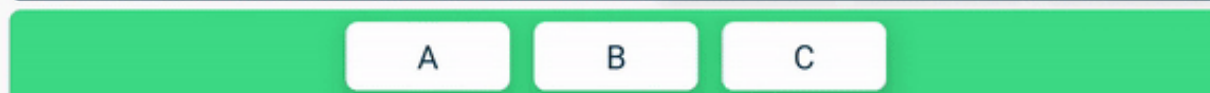
Space Evenly



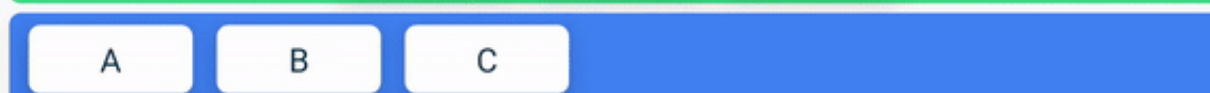
End (LTR)



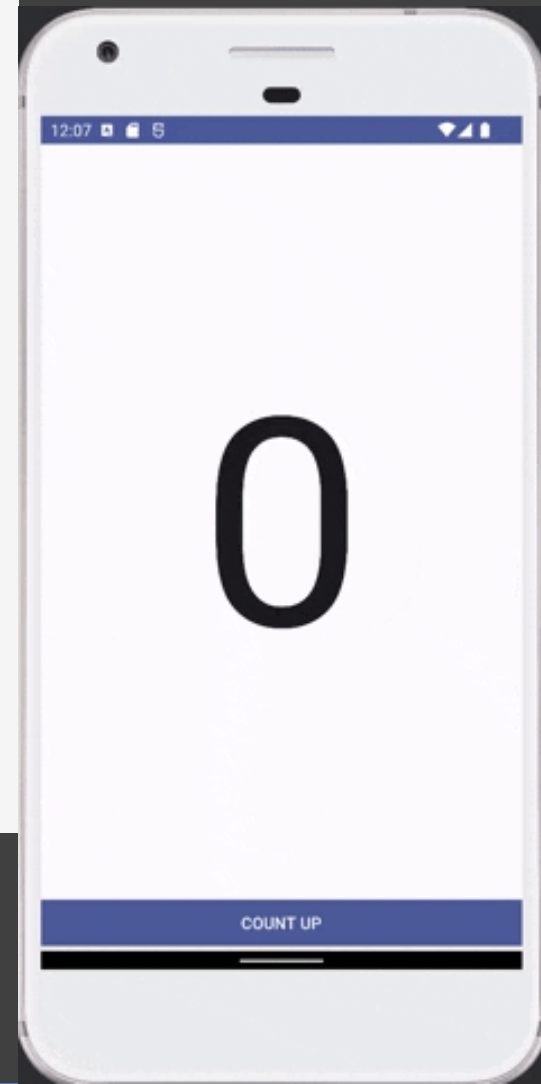
Center



Start (LTR)



```
@Composable
fun CounterExample() {
    Column(
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally,
        modifier = Modifier.fillMaxSize()
    ) {
        Spacer(modifier = Modifier.weight(0.3f))
        Text(
            text = "test",
            fontSize = 250.sp,
            textAlign = TextAlign.Center,
            modifier = Modifier.weight(1f),
        )
        Button(
            modifier = Modifier.fillMaxWidth(),
            shape = RectangleShape
        ) {
            Text(text = "COUNT UP")
        }
    }
}
```



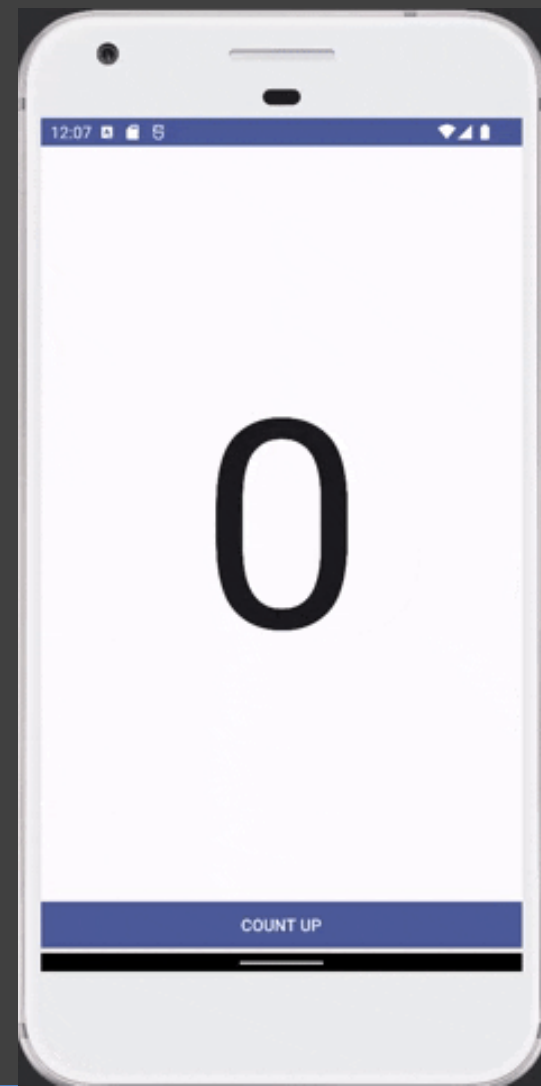
```
@Composable
fun CounterExample() {
    Column(
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally,
        modifier = Modifier.fillMaxSize()
    ) {

        val count: MutableState<Int> = remember {
            mutableStateOf(0)
        }

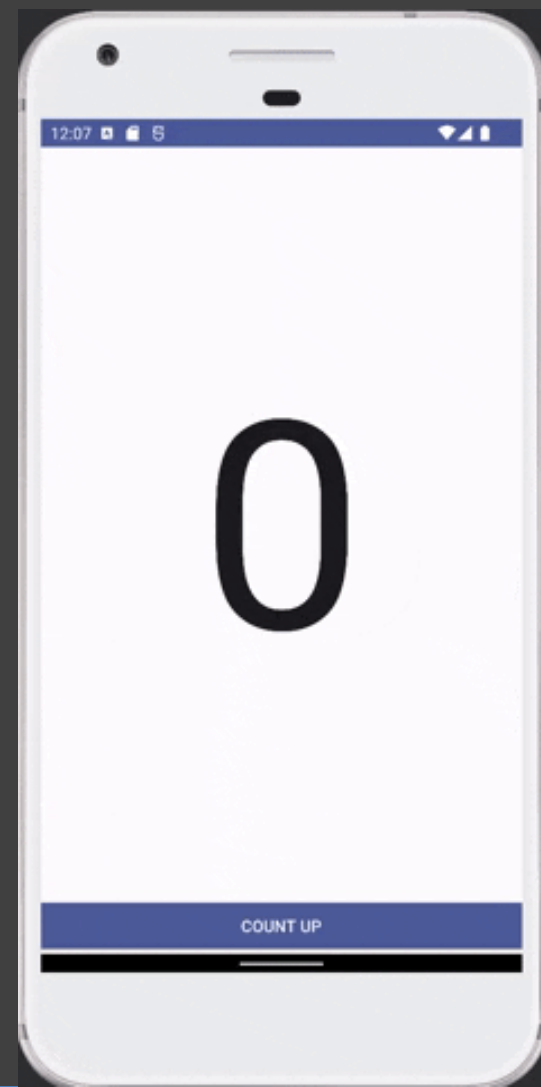
        Spacer(modifier = Modifier.weight(0.3f))

        Text(
            text = "${count.value}",
            fontSize = 250.sp,
            textAlign = TextAlign.Center,
            modifier = Modifier.weight(1f),
        )

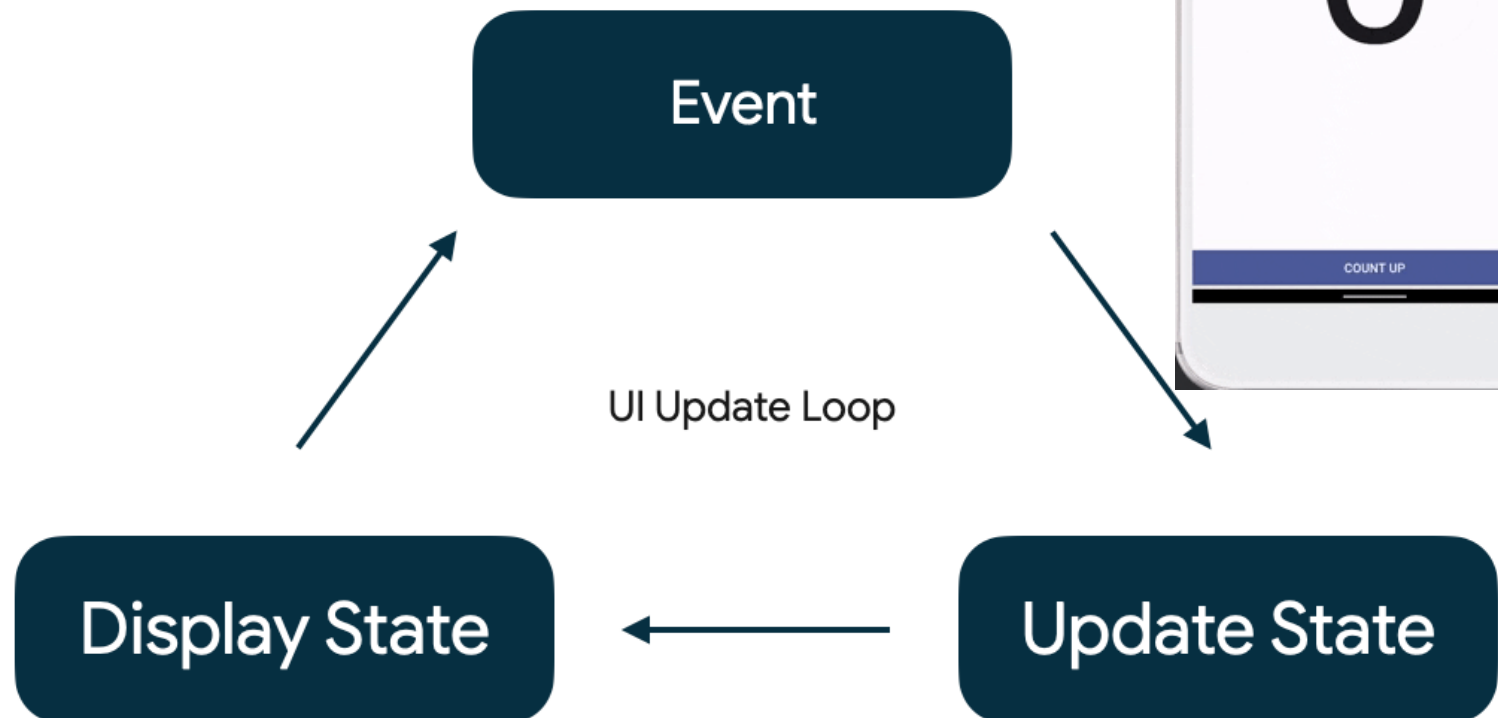
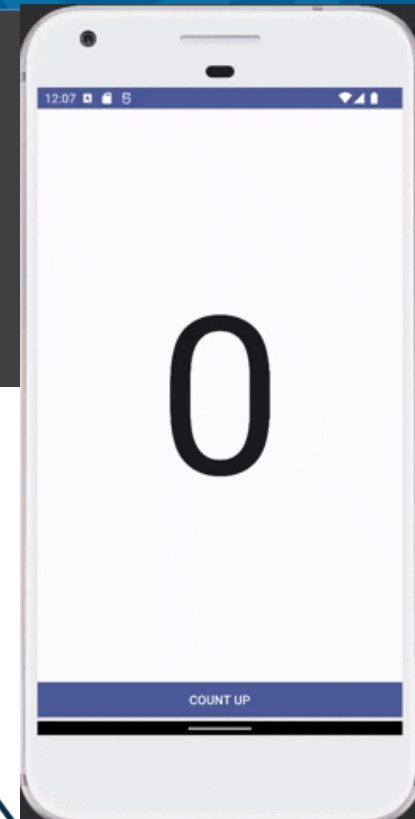
        Button(
            onClick = { count.value++ }, // onclick
            modifier = Modifier.fillMaxWidth(),
            shape = RectangleShape
        ) {
            Text(text = "COUNT UP")
        }
    }
}
```



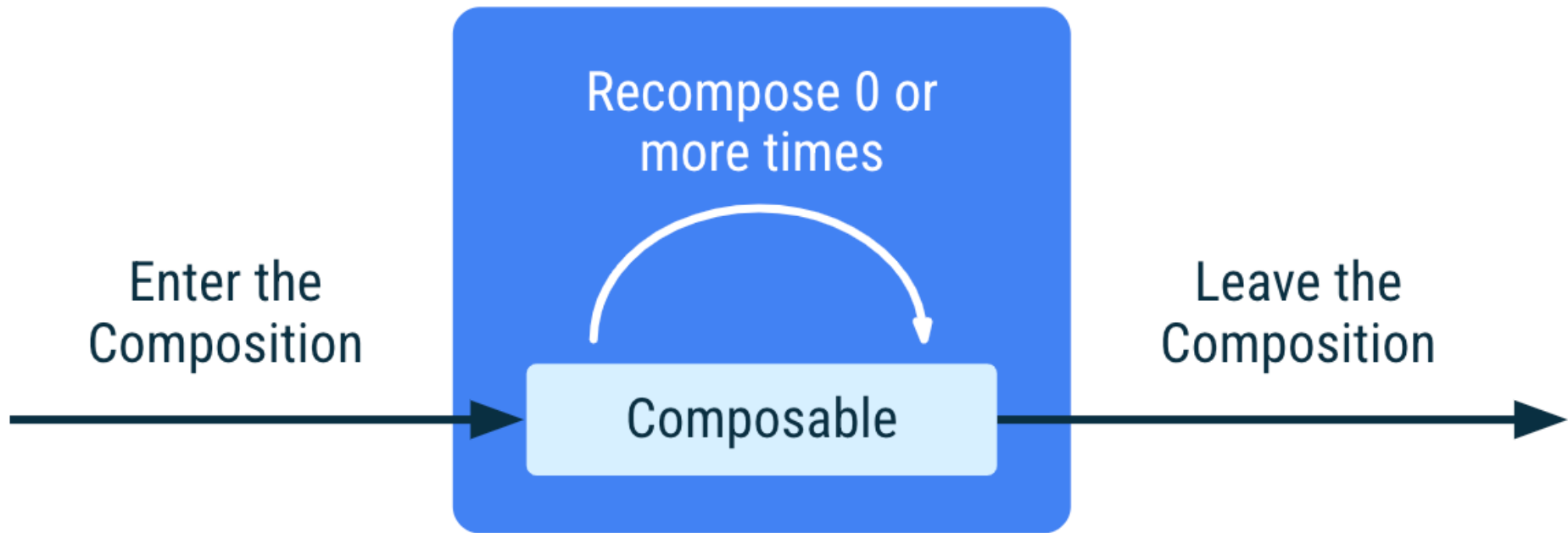
```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContent {  
        JetpackComposeStateBasicsTheme {  
            Surface(  
                modifier = Modifier.fillMaxSize(),  
                color = MaterialTheme.colorScheme.background  
            ) {  
                CounterExample()  
            }  
        }  
    }  
}
```



```
val count: MutableState<Int> = remember {  
    mutableStateOf(0)  
}
```

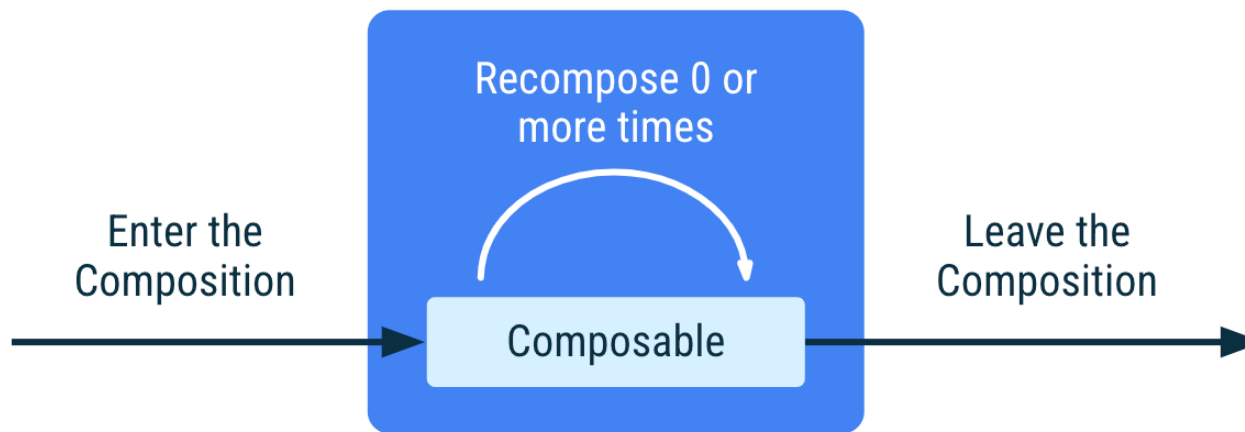


Composition





Composition



```
@Composable
fun MyComposable() {
    Column {
        Text("Hello")
        Text("World")
    }
}
```

MyComposable

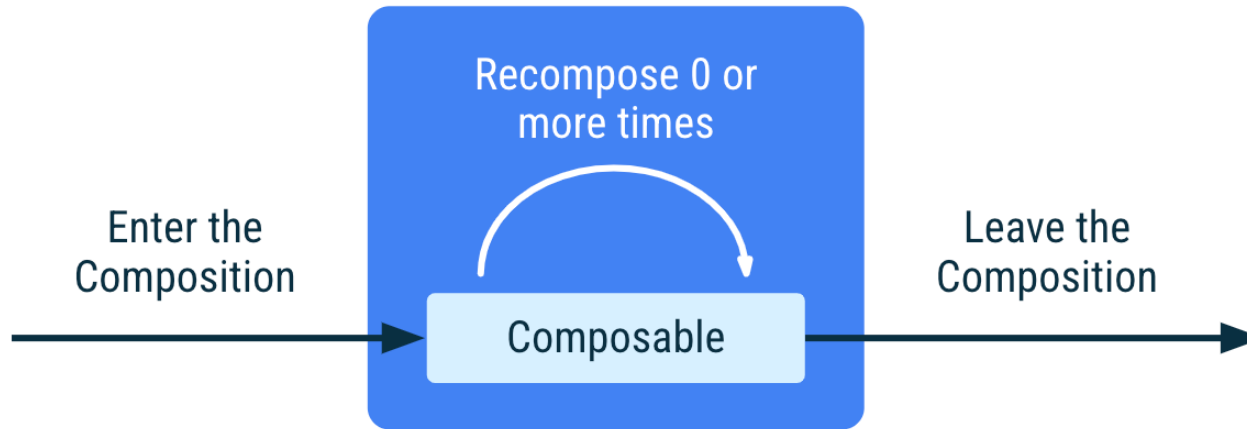


Column

Text

Text

Composition



MoviesScreen

Column

MovieOverview

MovieOverview

Recomposition
(new element added to
the bottom of the list)

MoviesScreen

Column

MovieOverview

MovieOverview

MovieOverview