Developers

# PGR208 Android programmering

## Text, Button, Column, Row - **og stilsetting**

Rolando Gonzalez, 2024

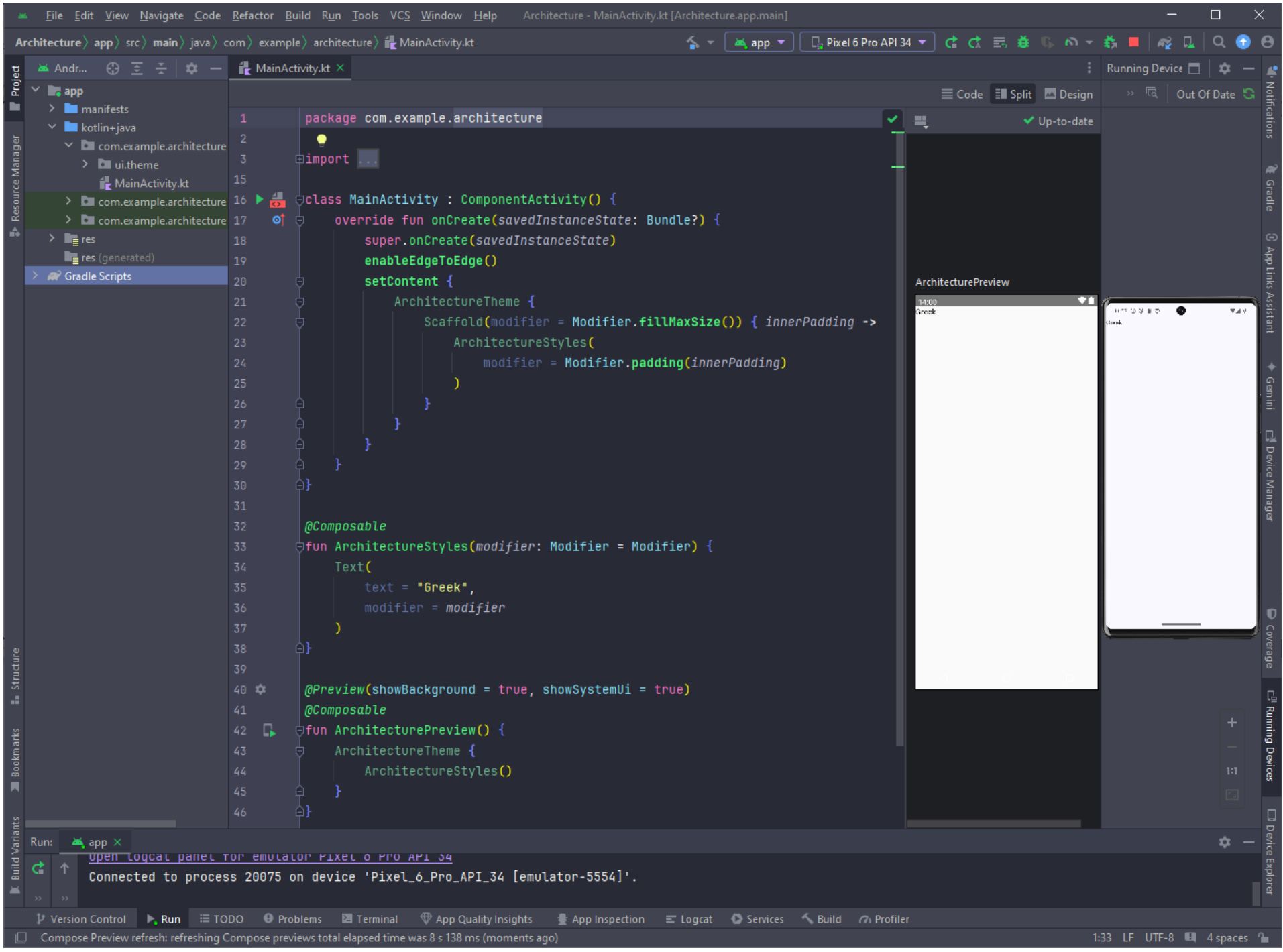
# Innhold

- Om denne slideserien og en liten intro
- Text()
- Column() og Row()
- Button()
  - Forskjellige knappekomponenter
  - Sette onClick på Button
- Column og rememberScrollState() og verticalScroll()

# Om denne slideserien

- Dette er en slideserie som gjennomgår flere teknikker og konsepter samtidig som komponentene beskrevet i forrige slide gås gjennom.
- Målet er å forstå en del detaljer i hvordan komponenter fungerer og hvordan de kan stilsette.

- Hvis du ønsker å gjøre som i slideserien:
- Har opprettet nytt prosjekt som kaller Architecture
- Har en @Composable ArchitectureStyles
- Kjører her både Preview og app (men trenger nok bare preview)



# Andre ting å merke seg

- Modifier som sendes inn i ArchitectureStyles fra onCreate-funksjonen og benyttes av Text gjør at det blir riktig avstand (innerPadding) fra toppen av appen, ellers vil det overlappe med tidsviseren og andre ting i statusbaren.
- Ved å tas inn i onCreate-funksjonen er ArchitectureStyles egentlig mer ment å være en «hub component»/moderkomponent enn en selvstendig komponent.
- showSystemUI i @Preview gjør at vi får opp hele appen i preview og gir mer realistisk presentasjon mtp. en mobiltelefon.

# Composables som konsept

- En app kan bestå av mange deler. For eksempel headerområde, hovedområde, footer/menu, produktbokser osv.
- En composable er en komponent som kan inneholde kode for en eller flere deler av en app.
- Composables er gjerne gjenbrukbare.
- Man ønsker ikke at en composable skal gjøre for mye.
- En composable skrives som en funksjon med UpperCamelCasing

# Text()

# Text()

- Text() er en komponent som viser en tekst. Tekst kan dreie seg om avsnitt eller overskrift og stilsettes etter behov.
- Hvis man peker på Text() så vil man få opp en god del properties man kan sette på text.

```
@Composable
public fun Text(
    text: String,
    modifier: Modifier = Modifier,
    color: Color = Color.Unspecified,
    fontSize: TextUnit = TextUnit.Unspecified,
    fontStyle: FontStyle? = null,
    fontWeight: FontWeight? = null,
    fontFamily: FontFamily? = null,
    letterSpacing: TextUnit = TextUnit.Unspecified,
    textDecoration: TextDecoration? = null,
    textAlign: TextAlign? = null,
    lineHeight: TextUnit = TextUnit.Unspecified,
    overflow: TextOverflow = TextOverflow.Clip,
    softWrap: Boolean = true,
    maxLines: Int = Int.MAX_VALUE,
    minLines: Int = 1,
    onTextLayout: ((TextLayoutResult) -> Unit)? = null,
    style: TextStyle = LocalTextStyle.current
): Unit
```

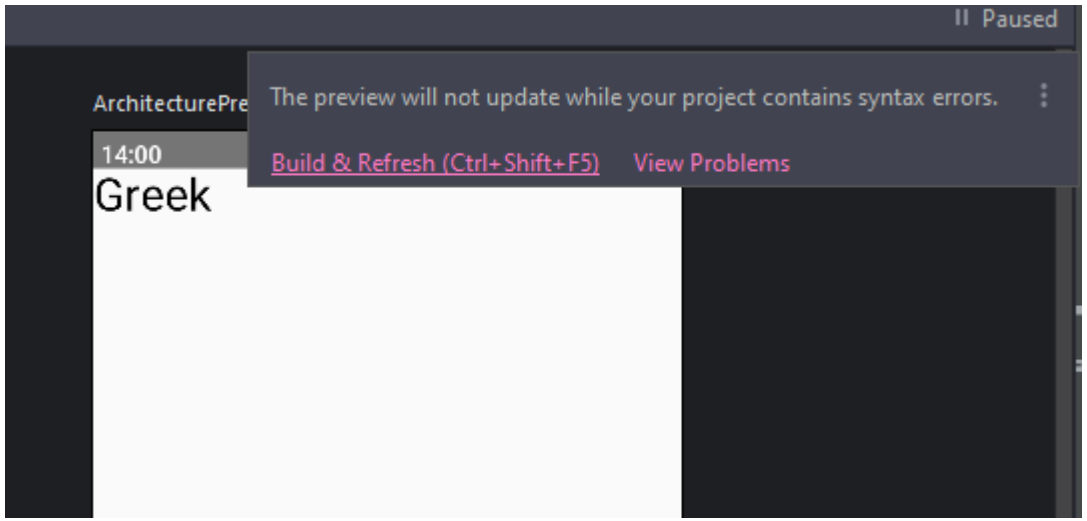
# fontSize

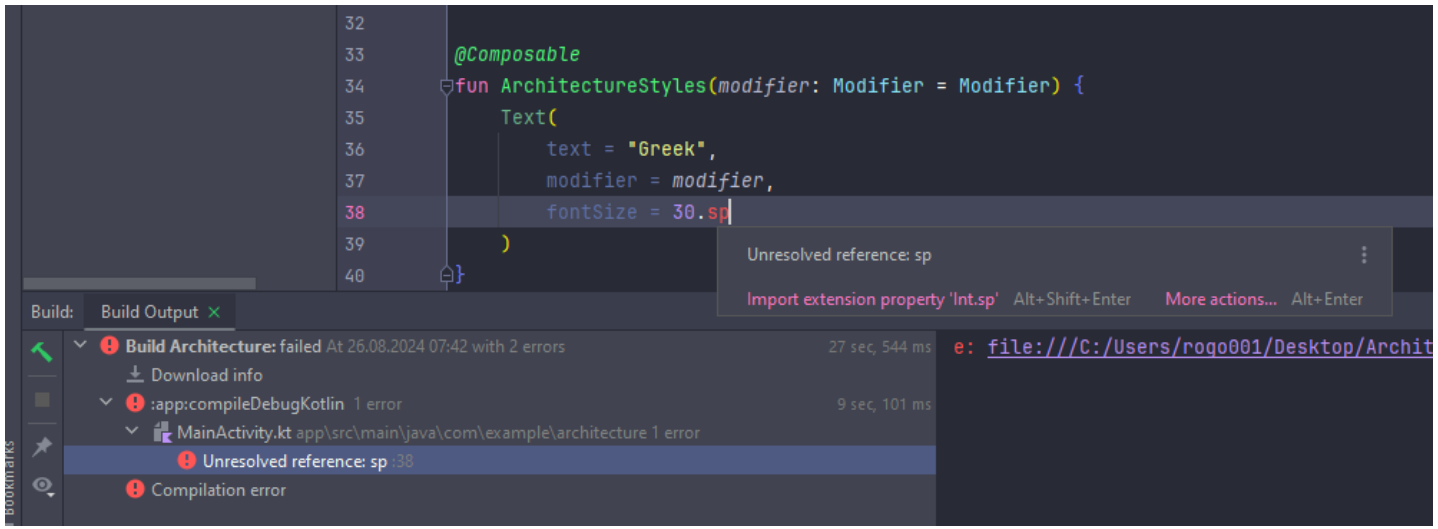
- fontSize brukes for å sette størrelse på tekst. Man bruker sp (Scale-Independent Pixels) som størrelsesenhet på tekst for å få en tekst som er dynamisk i forhold til preferanser til bruker.
- Når man bruker sp første gangen i filen ser man at man får feilmelding «unresolved reference: sp» - vi må importere sp. Klikk på «Import…» i meldingen og den legger seg til øverst i filen i importseksjonen.


```
import androidx.compose.ui.unit.sp
```

# Build Output ved feil

- Nederst i Android Studio kan man når man velger Build & Refresh se feilmeldinger i Build Output. Vi kan merke oss at det står hvilken fil problemet stammer fra, Compilation error, Unresolved reference: sp, og at det er på linje 38.



# color = Color.X

- Med color kan man sette tekstfargen på Text().
- Det finnes flere måter å gjøre dette på.
- Mest grunnleggende er å bruke Color-klassen som har et sett med predefinerte farger. Eller man kan bruke RGB-koder eller heksadesimale verdier.

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = "Greek",
        modifier = modifier,
        fontSize = 30.sp,
        color = Color.Red
    )
}
```

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = "Greek",
        modifier = modifier,
        fontSize = 30.sp,
        color = Color( red: 255,  green: 0,  blue: 255)
    )
}
```

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = "Greek",
        modifier = modifier,
        fontSize = 30.sp,
        color = Color( color: 0xFFC0CBFF)
    )
}
```

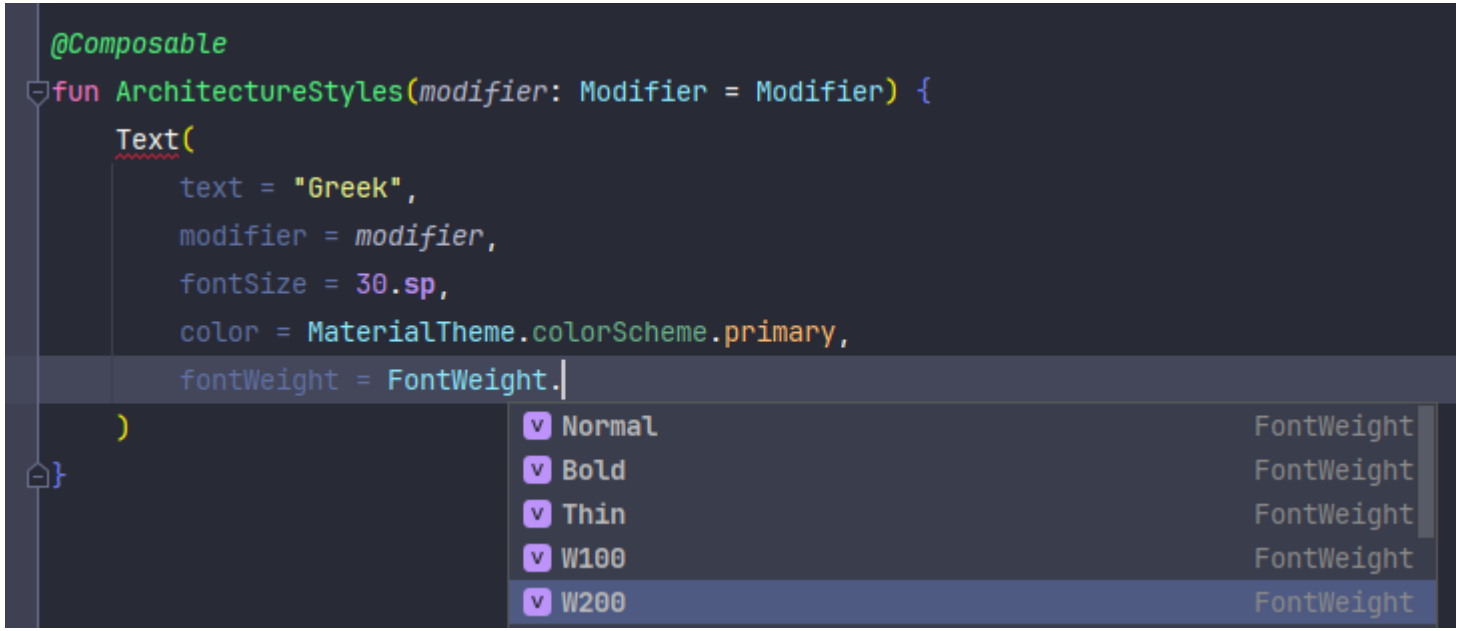
# Farge fra MaterialTheme

- Man har også mulighet til å sette farge fra MaterialTheme.colorScheme-pakken.

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = "Greek",
        modifier = modifier,
        fontSize = 30.sp,
        color = MaterialTheme.colorScheme.primary
    )
}
```

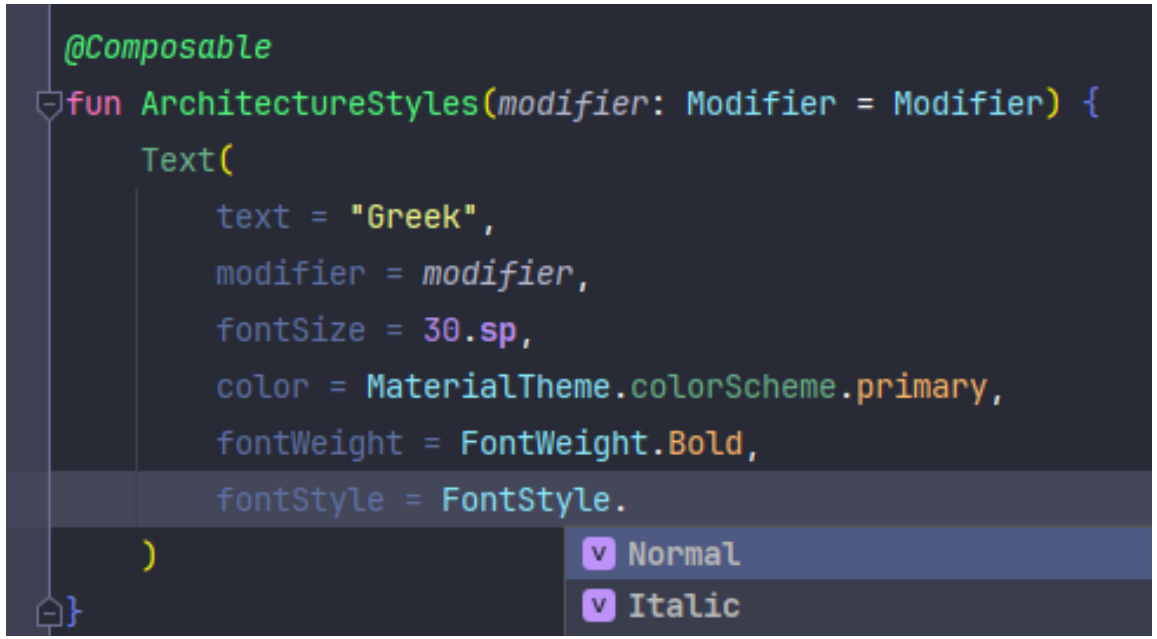
# fontWeight

- Med fontWeight får man justert tykkelsen på teksten.



# fontStyle

- Vanlig eller kursiv tekst



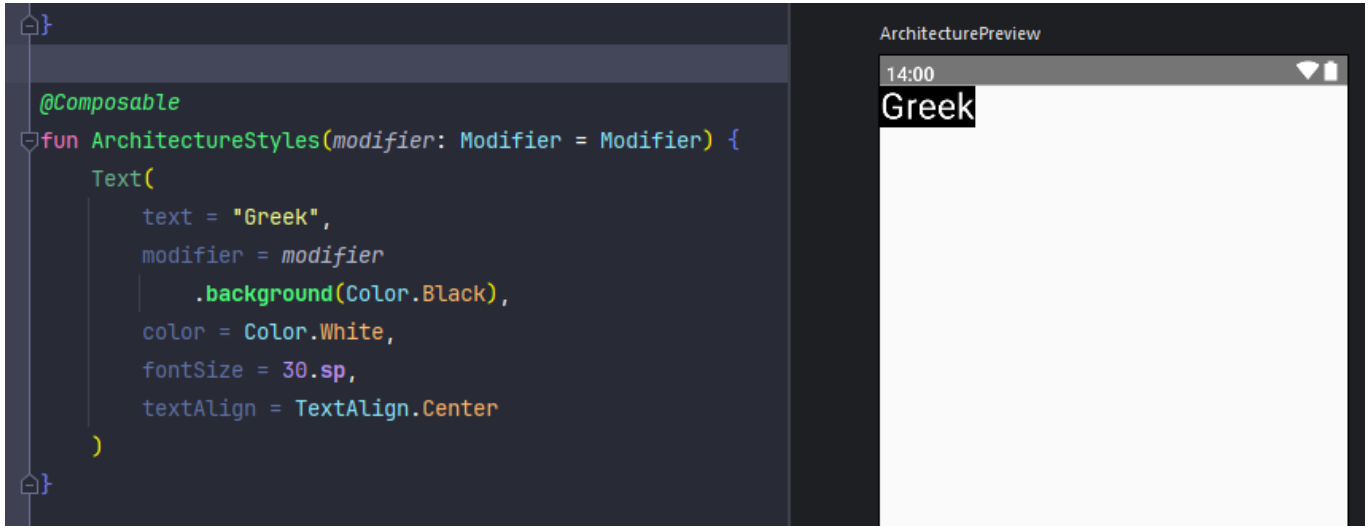
# textDecoration

- textDecoration benyttes blant annet for å lage understreking eller strek gjennom tekst.

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = "Greek",
        modifier = modifier,
        fontSize = 30.sp,
        color = MaterialTheme.colorScheme.primary,
        fontWeight = FontWeight.Bold,
        fontStyle = FontStyle.Normal,
        textDecoration = TextDecoration.LineThrough
    )
}
```

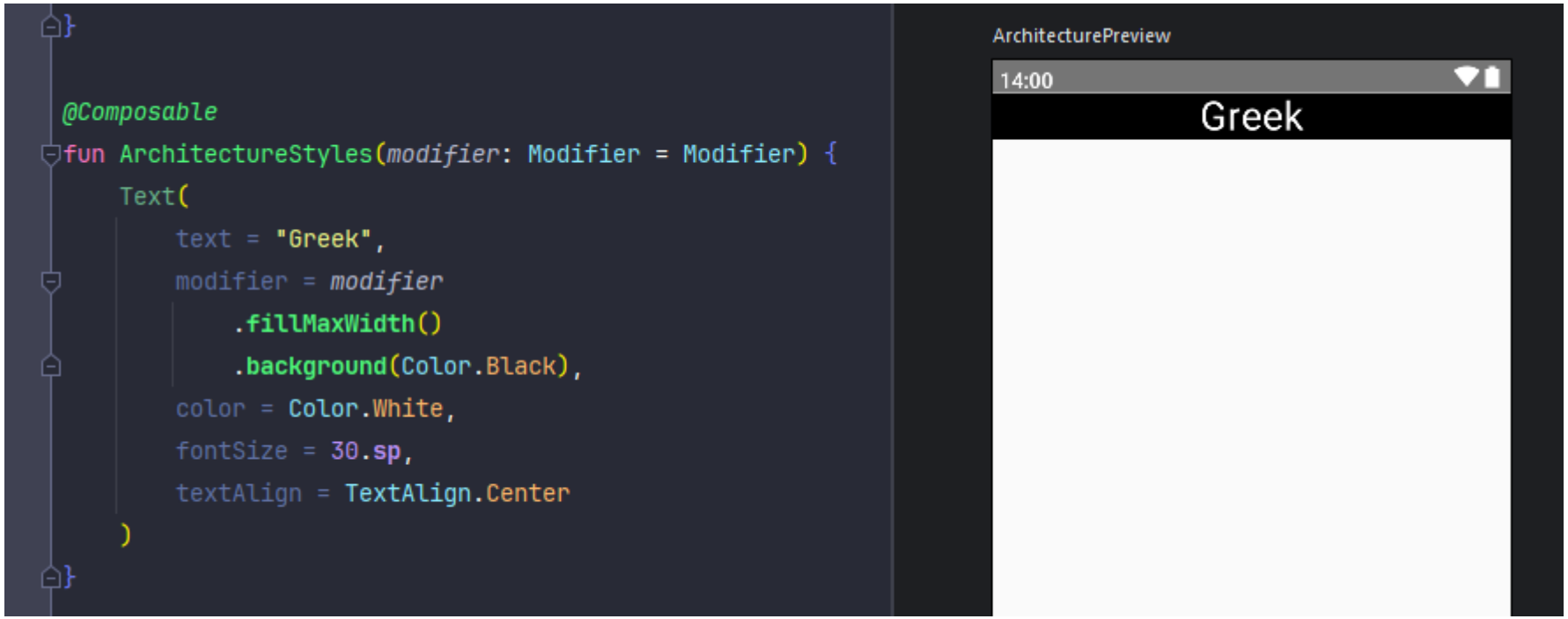
# Hvorfor virker ikke TextAlign.Center? 1 / 2

- TextAlign.Center er hva som kan brukes for å midtsentrere en tekst, men vi kan oppleve at den ikke midtstilles. La oss se på hvorfor.
- Hvis vi setter bakgrunnsfarge på Text() så ser vi at boksen som teksten ligger i ikke dekker bredden og teksten kan bare plasseres i forhold til boksen den ligger inni. Merk at vi her har brukt Modifier-klassen for å endre på background. Noen ting ligger i properties til komponent, men andre må hentes fra Modifier (gjelder alle komponenter).



# Hvorfor virker ikke TextAlign.Center? 2 / 2

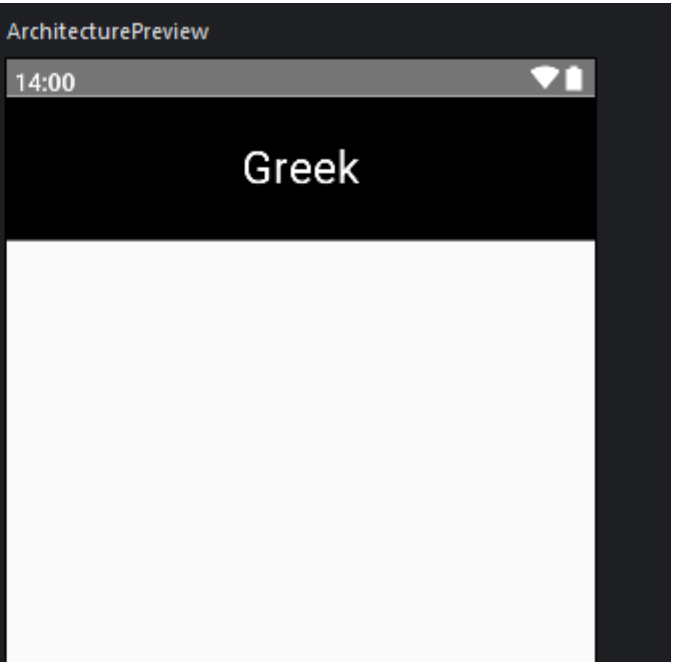
- Ved å gjøre at Text()-boksen dekker bredden  (.fillMaxWidth) så vil man kunne midtstille teksten.



# Padding – gjøre boksen rundt teksten større

- Man kan bruke padding på text for å utvide boksen - i dette tilfellet utvide boksen over og under teksten.

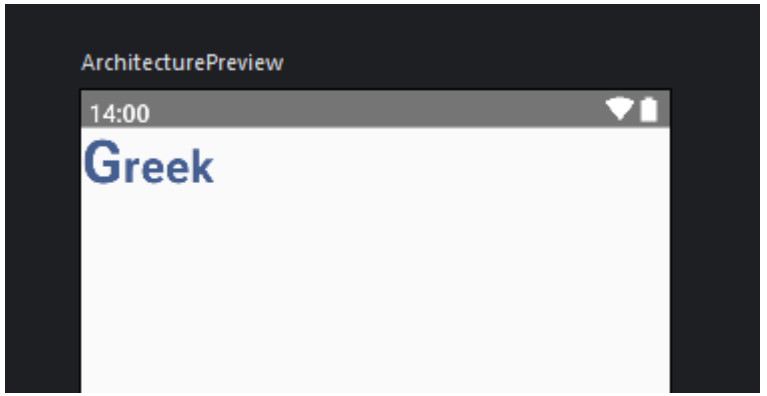
```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = "Greek",
        modifier = modifier
            .fillMaxWidth()
            .background(Color.Black)
            .padding(30.dp),
        color = Color.White,
        fontSize = 30.sp,
        textAlign = TextAlign.Center
    )
}
```



# buildAnnotatedString

- Med buildAnnotatedString kan vi gi deler av en tekst en annen stil.
- I kodeeksempelet ønsker vi at første bokstav skal være større enn resten av teksten.

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = buildAnnotatedString {
            withStyle(
                style = SpanStyle(
                    fontSize = 40.sp
                )
            ){
                append("G")
            }
            append("reek")
        },
        modifier = modifier,
        fontSize = 30.sp,
        color = MaterialTheme.colorScheme.primary,
```

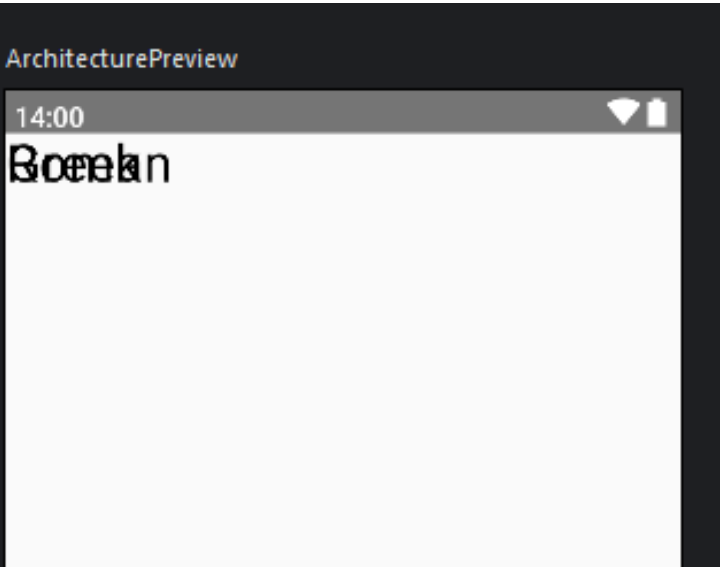


# Column() og Row()

# Column()

- Legg til 2 Text()-komponenter og du vil se at de legger seg over hverandre.
- Man må bruke teknikker som eksplisitt plasserer komponenter slik vi ønsker.

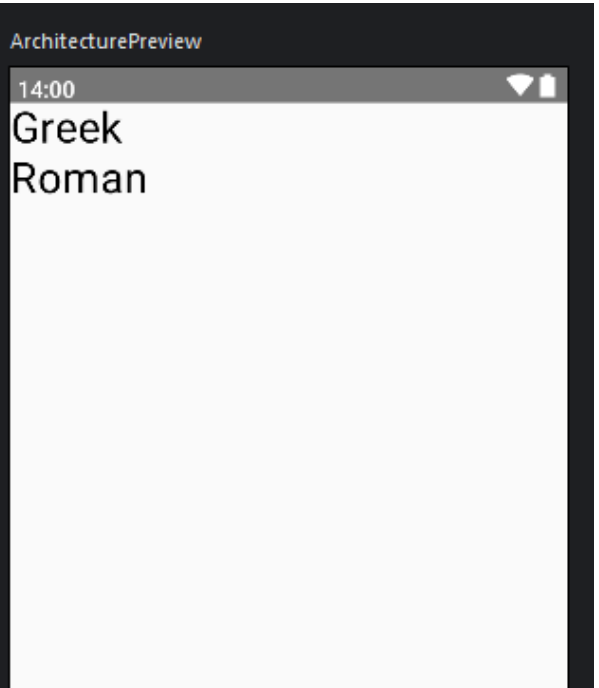
```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Text(
        text = "Greek",
        fontSize = 30.sp
    )
    Text(
        text = "Roman",
        fontSize = 30.sp
    )
)
```



# Column

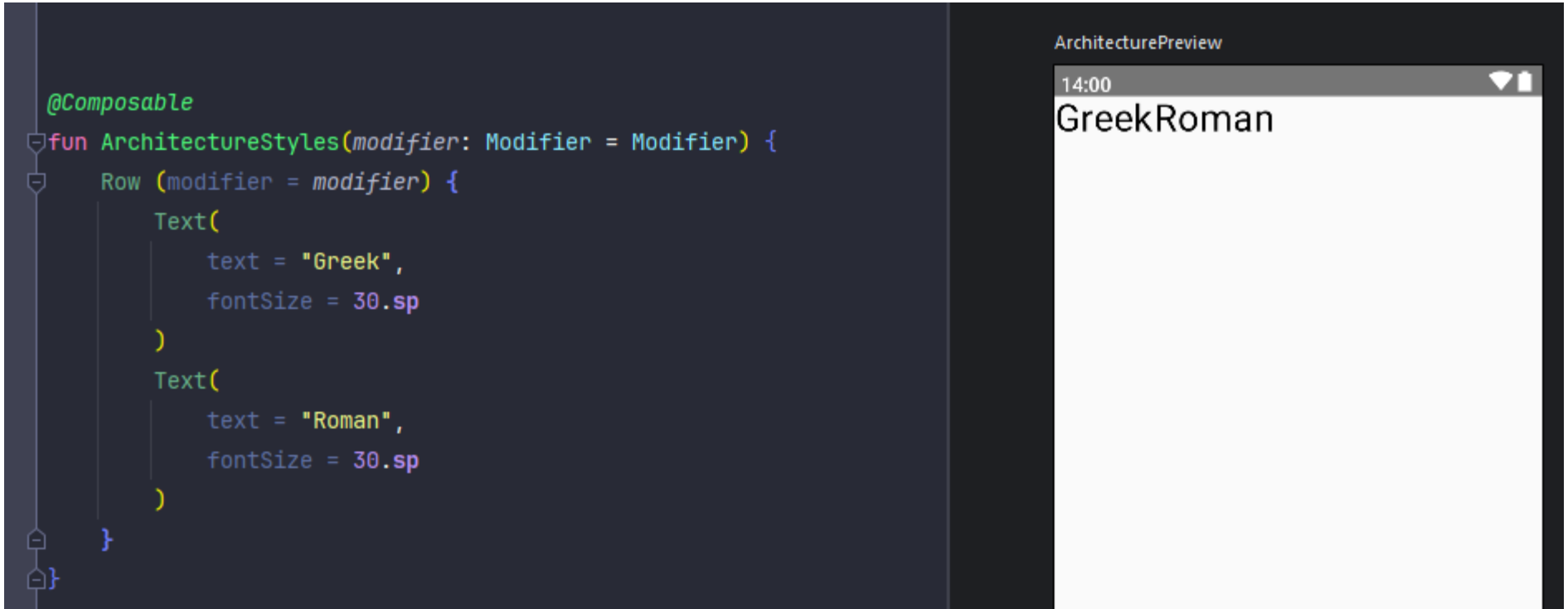
- Ved å wrappe Column rundt komponentene så vil de plasseres under hverandre vertikalt.
- **Annet:** merk at modifier er lagt til i Column for å få riktig avstand fra toppen på mobilen. Man kan ha både modifier i Column og modifier i Text.

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Column (modifier = modifier) {
        Text(
            text = "Greek",
            fontSize = 30.sp
        )
        Text(
            text = "Roman",
            fontSize = 30.sp
        )
    }
}
```



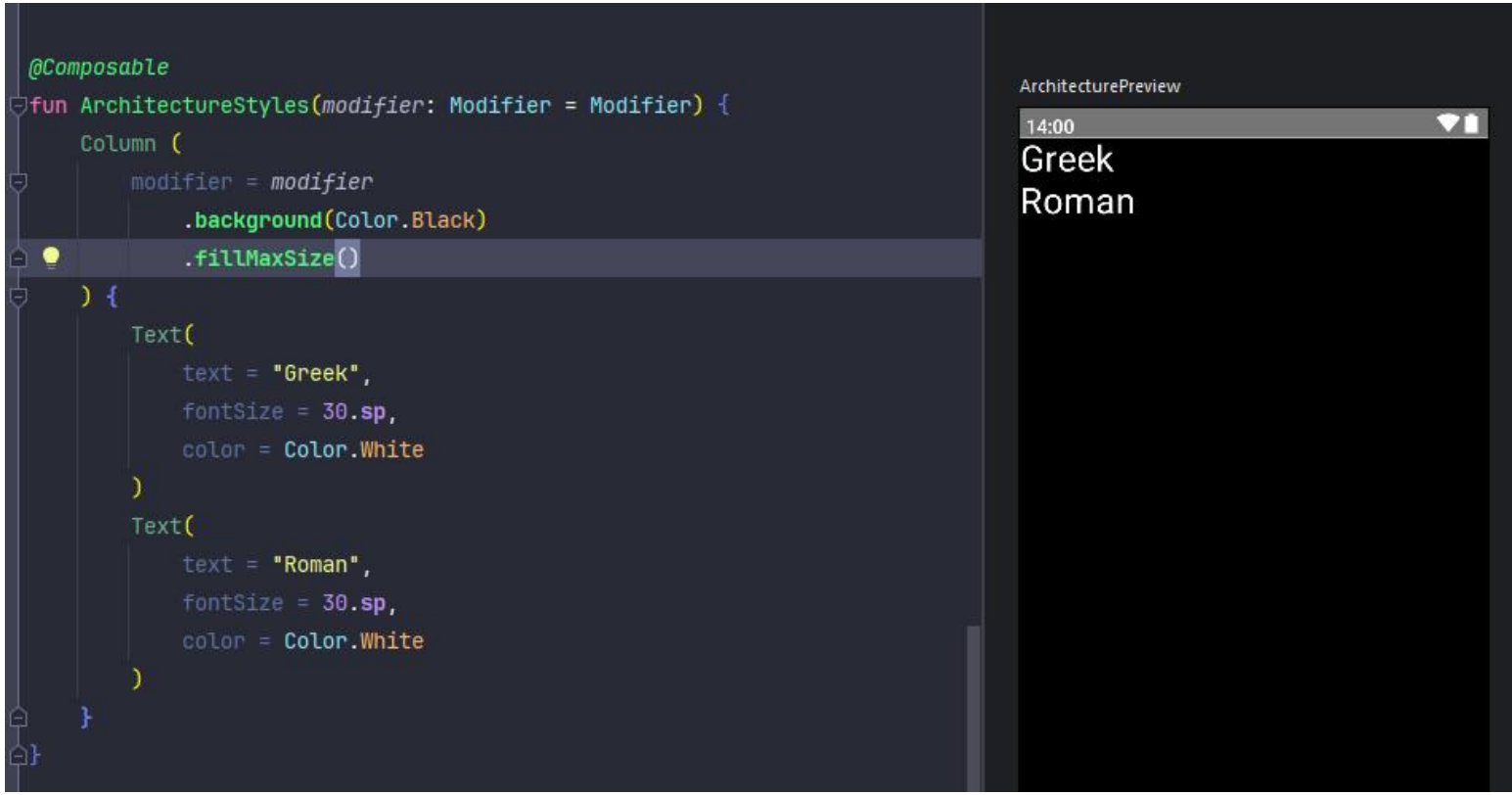
# Row

- Med Row kan man plassere ting ved siden av hverandre.
- PS! Husk at både Column og Row må importeres!



# Farge og størrelse på en moderkomponent 1 / 2

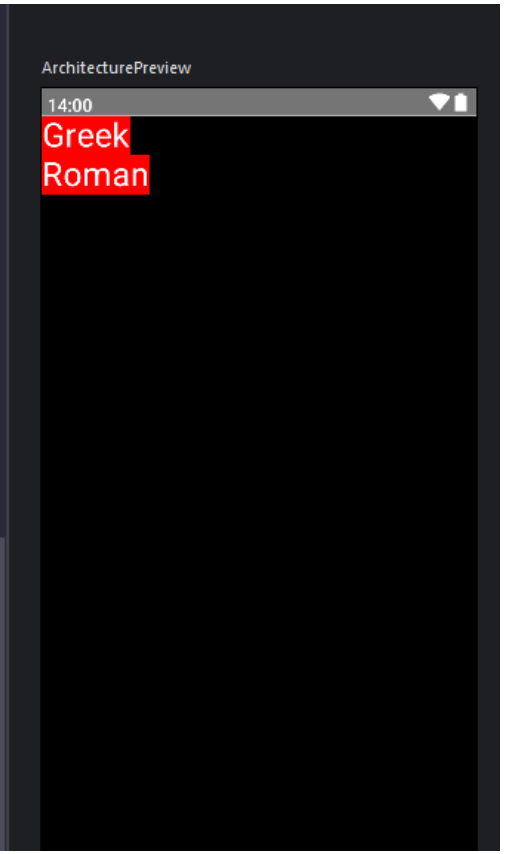
- I dette kodeeksempelet ser vi at vi har satt sort bakgrunnsfarge på Column og latt den ta hele skjermbredden med fillMaxSize (vi brukte i en tidligere slide fillMaxWidth)



# Farge og størrelse på en moderkomponent 2 / 2

- Selv om moderkomponenten har full skjermstørrelse betyr det ikke at barnekomponentene også har det – slik skjermbildet viser.
- Skjermbildet viser også hvordan vi legger til modifier per Text()

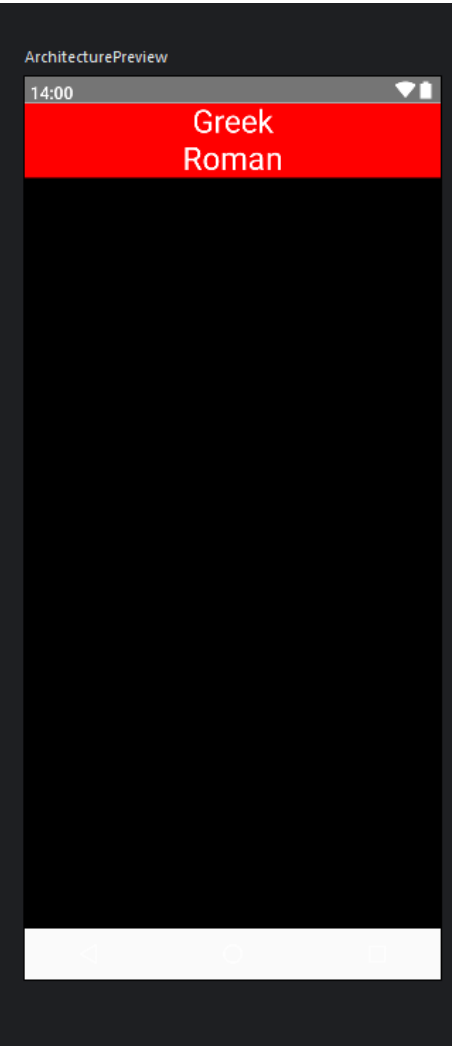
```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Column (
        modifier = modifier
            .background(Color.Black)
            .fillMaxSize()
    ) {
        Text(
            text = "Greek",
            fontSize = 30.sp,
            color = Color.White,
            modifier = Modifier
                .background(Color.Red)
        )
        Text(
            text = "Roman",
            fontSize = 30.sp,
            color = Color.White,
            modifier = Modifier
                .background(Color.Red)
        )
    }
}
```



# Gjenbruke stiler og modifier

- Kodeeksempelet viser hvordan man kan gjenbruke tekststiler og modifier slik at man slipper å definere per komponent.

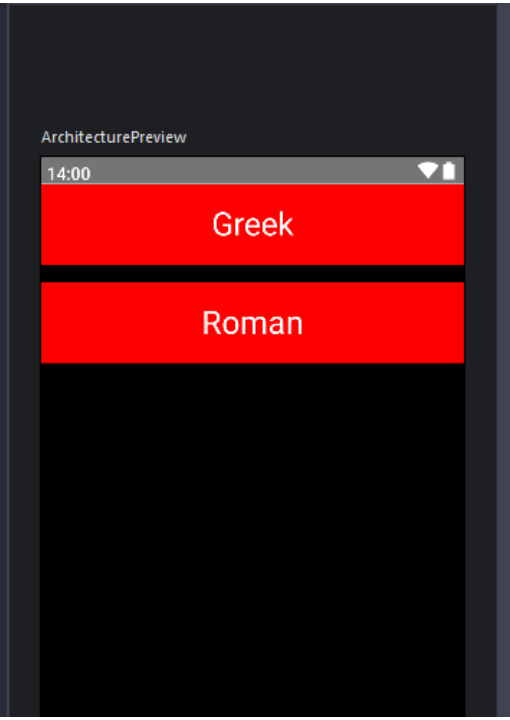
```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {

    val myTextStyle = TextStyle(
        fontSize = 30.sp,
        color = Color.White,
        textAlign = TextAlign.Center
    )

    val myTextModifier = Modifier
        .background(Color.Red)
        .fillMaxWidth()

    Column (
        modifier = modifier
            .background(Color.Black)
            .fillMaxSize()
    ) {
        Text(
            text = "Greek",
            style =  myTextStyle,
            modifier = myTextModifier
        )
        Text(
            text = "Roman",
            style = myTextStyle,
            modifier = myTextModifier
        )
    }
}
```



# Skape mellomrom mellom Text()

- Man kan bruke Spacer() for å skape mellomrom mellom komponenter

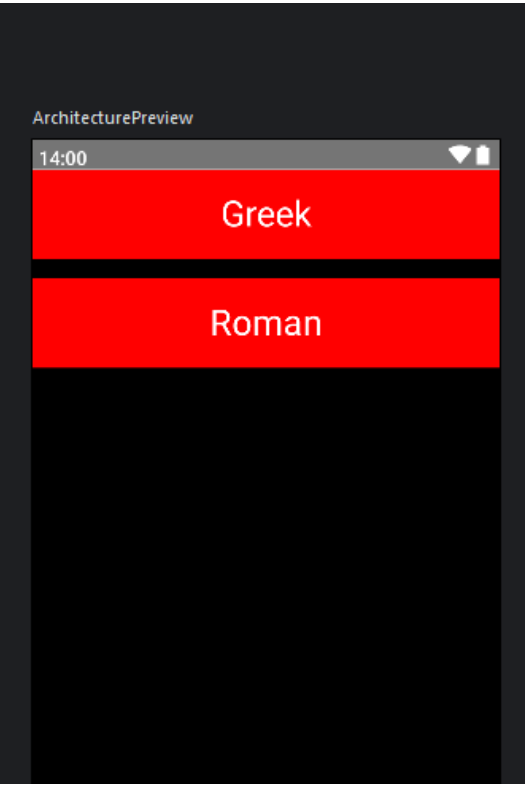
```
        .fillMaxWidth()
        .padding(0.dp, 20.dp)

    Column (
        modifier = modifier
            .background(Color.Black)
            .fillMaxSize()
    ) {
        Text(
            text = "Greek",
            style =  myTextStyle,
            modifier = myTextModifier
        )
        Spacer(modifier = Modifier.height(16.dp))
        Text(
            text = "Roman",
            style = myTextStyle,
            modifier = myTextModifier
        )
    }
}
```



# Skape mellomrom mellom Text() via Column()

- Man kan overlate ansvaret for plass mellom elementer til Column (eller Row) med verticalArrangement.

```
        .padding(0.dp, 20.dp)

    Column (
        modifier = modifier
            .background(Color.Black)
            .fillMaxSize(),
        verticalArrangement = Arrangement.spacedBy(16.dp)
    ) {
        Text(
            text = "Greek",
            style =  myTextStyle,
            modifier = myTextModifier
        )
        Text(
            text = "Roman",
            style = myTextStyle,
            modifier = myTextModifier
        )
    }
}
```



# Merknad om Column() og scrolling

- Merk at en Column() default ikke kan scrolles.
- Man må legge til litt ekstra kode for når man forventer at innholdet i Column() skal bli høyere enn høyden på mobiltelefonen.

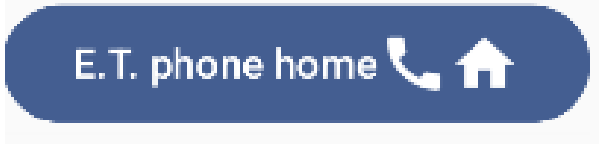
```
val scrollState = rememberScrollState()

Column (
        modifier = modifier
            .verticalScroll(scrollState)
)
{
    Text(
```

# Button()

# Button()

- En button er en komponent som man typisk putter en Text() i. Man kan vel å merke eksempelvis også putte en Icon() inni der også istedenfor eller eventuelt flere ting som f.eks. både Text() og Icon().
- Button tar seg av event og knappeboksen, mens Text() står for teksten og eventuelle stilrelaterte ting til den.

```
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
    Column(modifier = modifier){
        Button(onClick = { /*TODO*/ }) {
            Text(text = "Button 1")
        }
        Button(onClick = { /*TODO*/ }) {
            Text(text = "Button 2")
        }
    }
```




# Eksempel Button() med Icon()

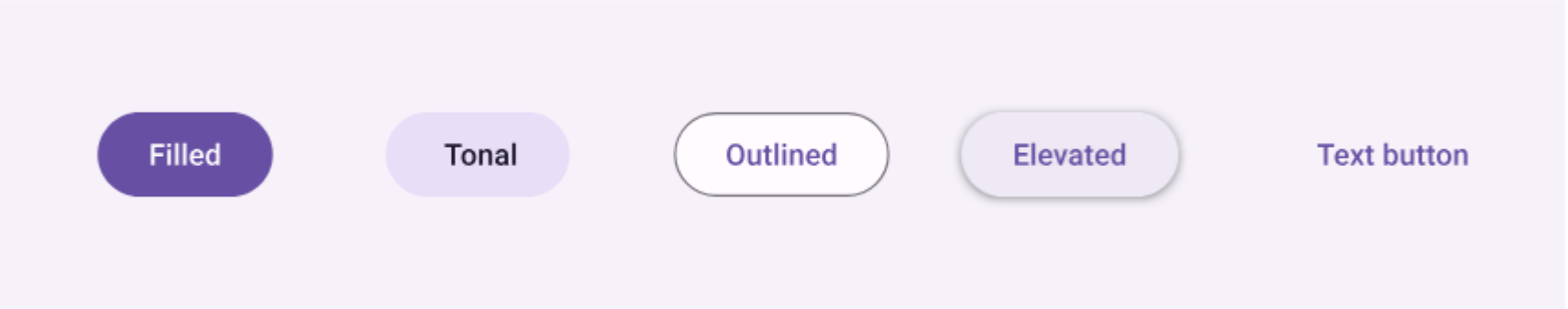
```
Button(onClick = { /*TODO*/ }) {
    Text( text: "E.T. phone home")
    Icon(imageVector = Icons.Default.Call, contentDescription = "Call")
    Icon(imageVector = Icons.Default.Home, contentDescription = "Home")
}
```

# Forskjellige knappekomponenter

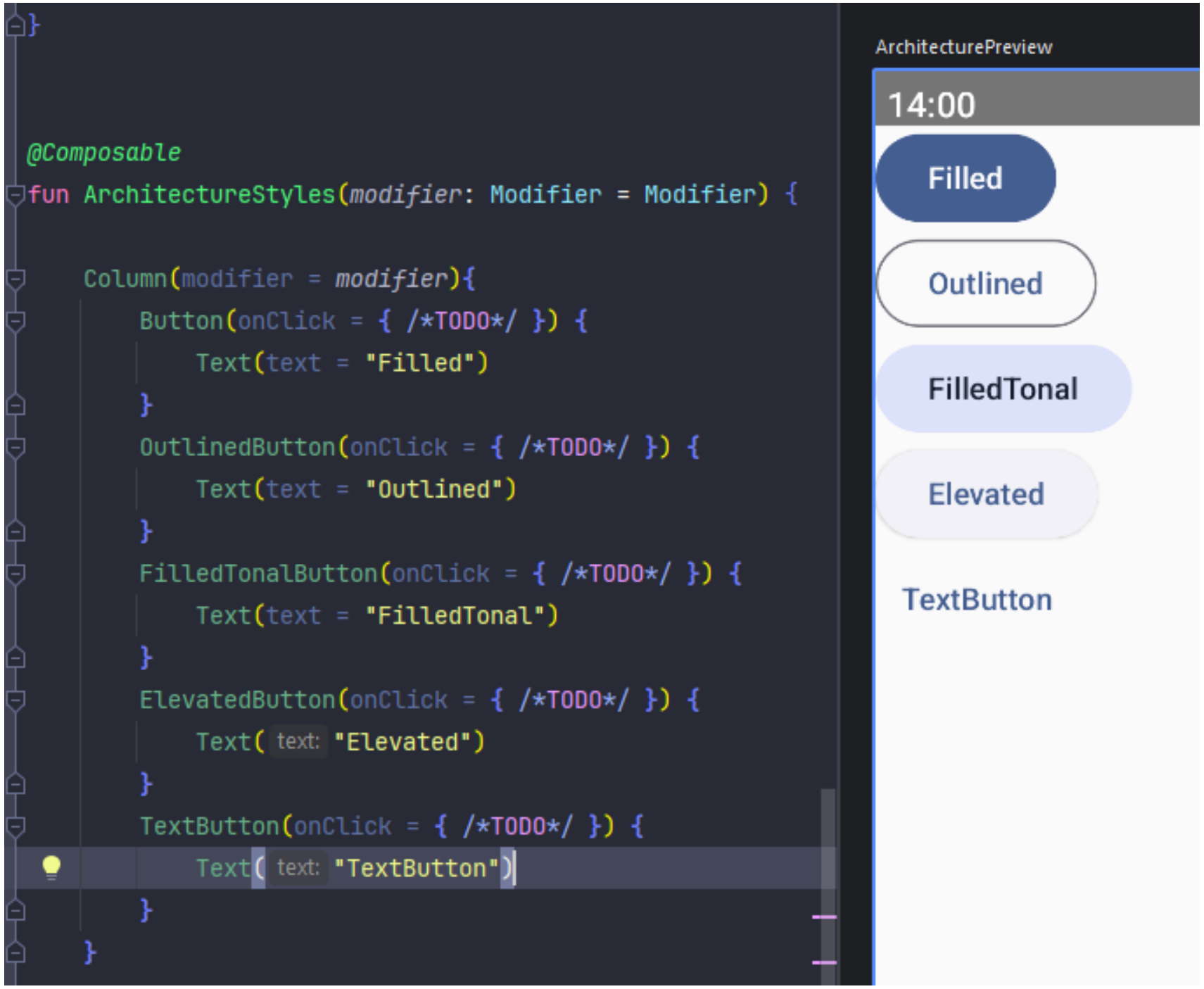
Button()

OutlinedButton()

...osv.



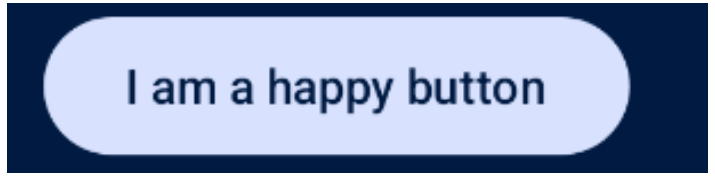

Les mer: https://developer.android.com/develop/ui/compose/components/button


}
@Composable
fun ArchitectureStyles(modifier: Modifier = Modifier) {
Column(modifier = modifier){
Button(onClick = { /*TODO*/ }) {
Text(text = "Filled")
}
OutlinedButton(onClick = { /*TODO*/ }) {
Text(text = "Outlined")
}
FilledTonalButton(onClick = { /*TODO*/ }) {
Text(text = "FilledTonal")
}
ElevatedButton(onClick = { /*TODO*/ }) {
Text( text: "Elevated")
}
TextButton(onClick = { /*TODO*/ }) {
Text( text: "TextButton")
}
}
ArchitecturePreview
14:00
Filled
Outlined
FilledTonal
Elevated
TextButton

# Endre Text() og Button(): tekst- og knappfarge

- Et eksempel på endring av bakgrunnen til knappen og teksten i Button sin farge.

```
Button(
    onClick = {/* TODO */ },
    colors = ButtonDefaults.buttonColors(
        containerColor = MaterialTheme.colorScheme.primaryContainer
    )
) {
    Text(
        color = MaterialTheme.colorScheme.onPrimaryContainer,
        text = "I am a happy button"
    )
}
```



# 4 måter å sette Button-farge

```
Button(
    onClick = {/* TODO */ },
    colors = ButtonDefaults.buttonColors(
        containerColor = MaterialTheme.colorScheme.tertiary
    )
) {
    Text(text = "I am a happy button")
}
```

**Alternativ 1: temafarge fra Theme.kt-filen i ditt prosjekt**

```
colors = ButtonDefaults.buttonColors(
            containerColor = MaterialTheme.colorScheme.tertiary
)
```

**Alternativ 2: egendefinert farge med rgb/rgba-kode**

```
colors = ButtonDefaults.buttonColors(
            containerColor =  Color(red = 50, green = 50, blue = 50) // kan også inkludere alpha (gjennomsiktighet)
)
```

**Alternativ 3: hovedfarger (core) via Color**

```
colors = ButtonDefaults.buttonColors(
            containerColor = Color.DarkGray
)
```

**Alternativ 4: egendefinert med hex-kode**

```
colors = ButtonDefaults.buttonColors(
            containerColor = Color(0xFFC0CBFF))
```