Leibniz Universität Hannover

Android Compose

Layout

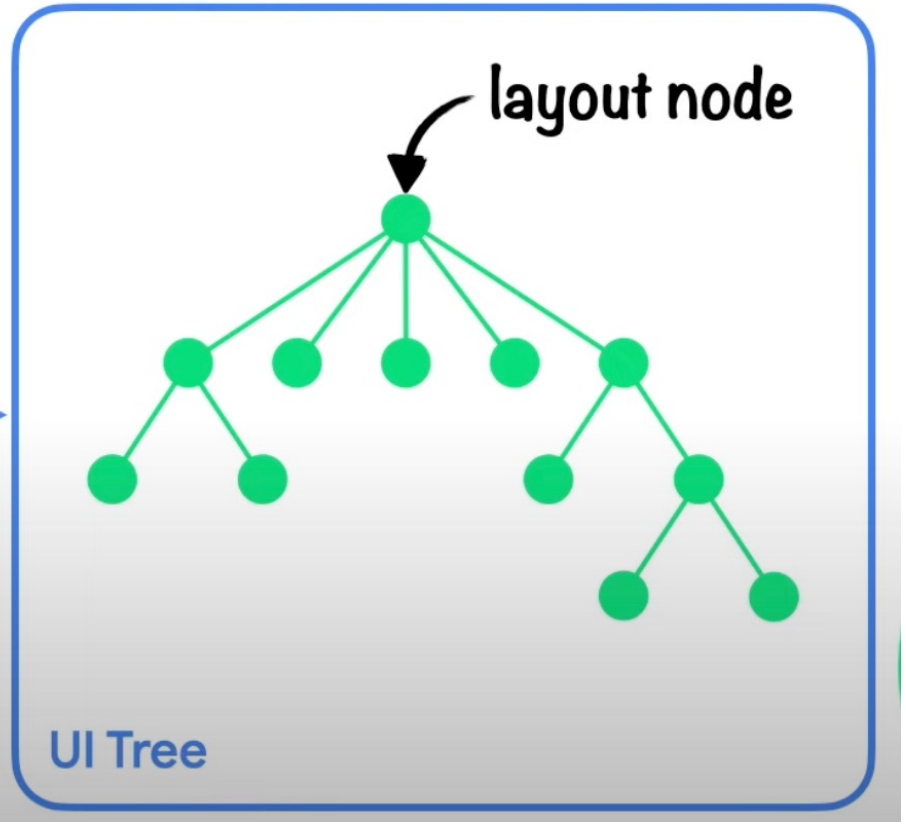
Human-Computer Interaction Group

Prof. Dr. Michael Rohs
michael.rohs@hci.uni-hannover.de

Layout

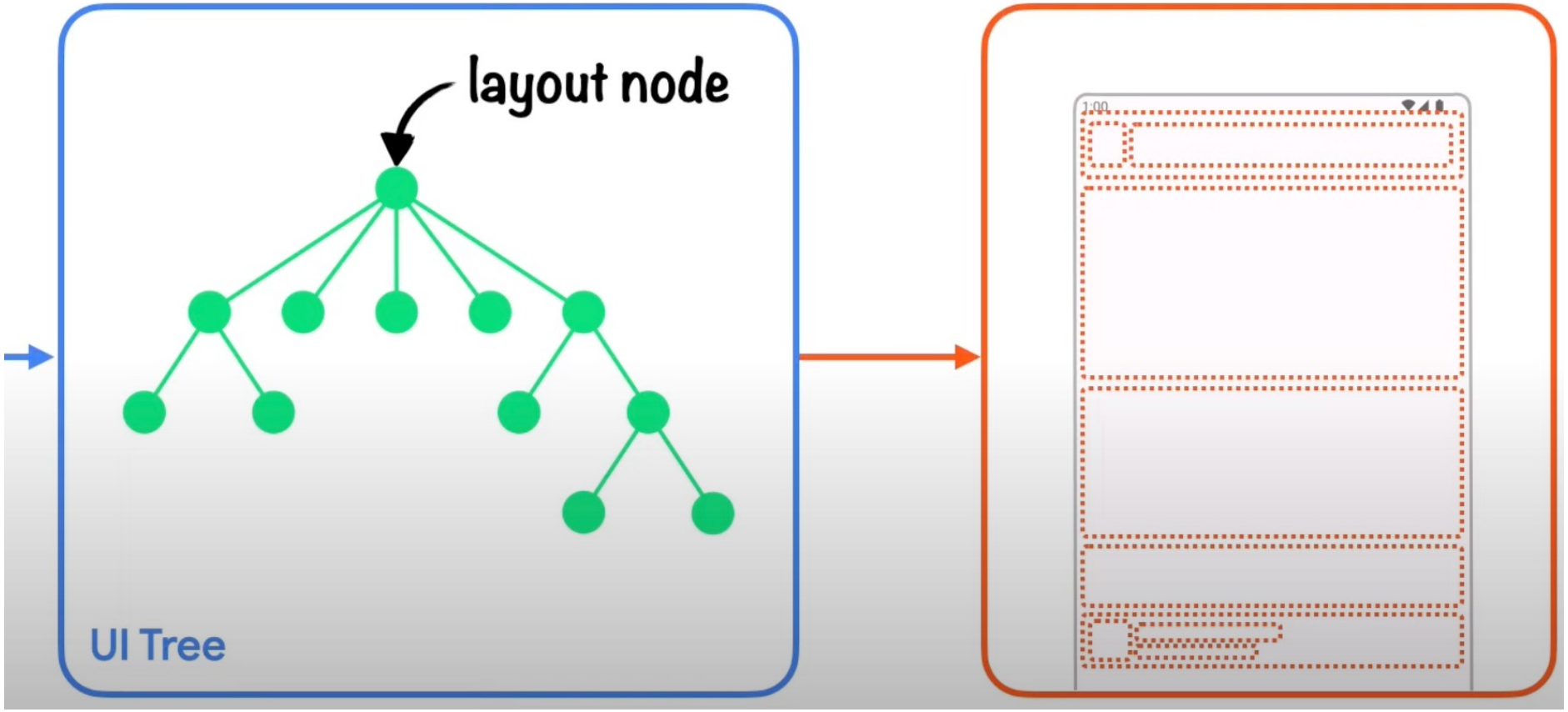
- Arrange UI elements on the screen
- Here: How to achieve a desired layout with,
  rather than abstract principles for layouts on small screens
- Layout phases


```
Column {
    Row {
        Icon(..)
        Title(..)
    }
    HeaderImage(..)
    Title(..)
    Subtitle(..)
    Row {
        Image(..)
        Column {
            Text(..)
            Text(..)
        }
    }
}
```



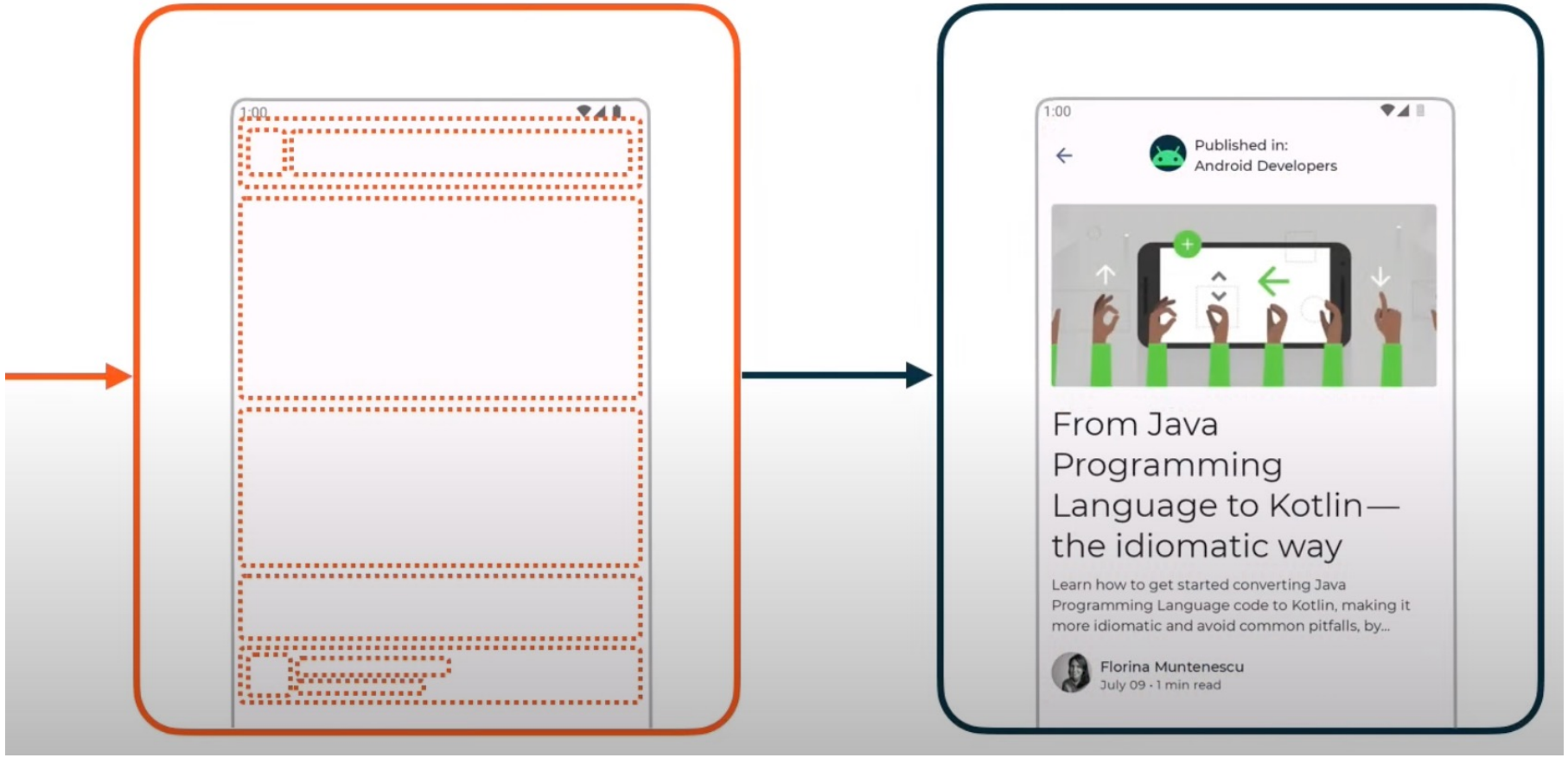
https://youtu.be/0yK7KoruhSM?si=723ewDTuy4BYzVsP

State → Composition → Layout → Drawing → UI



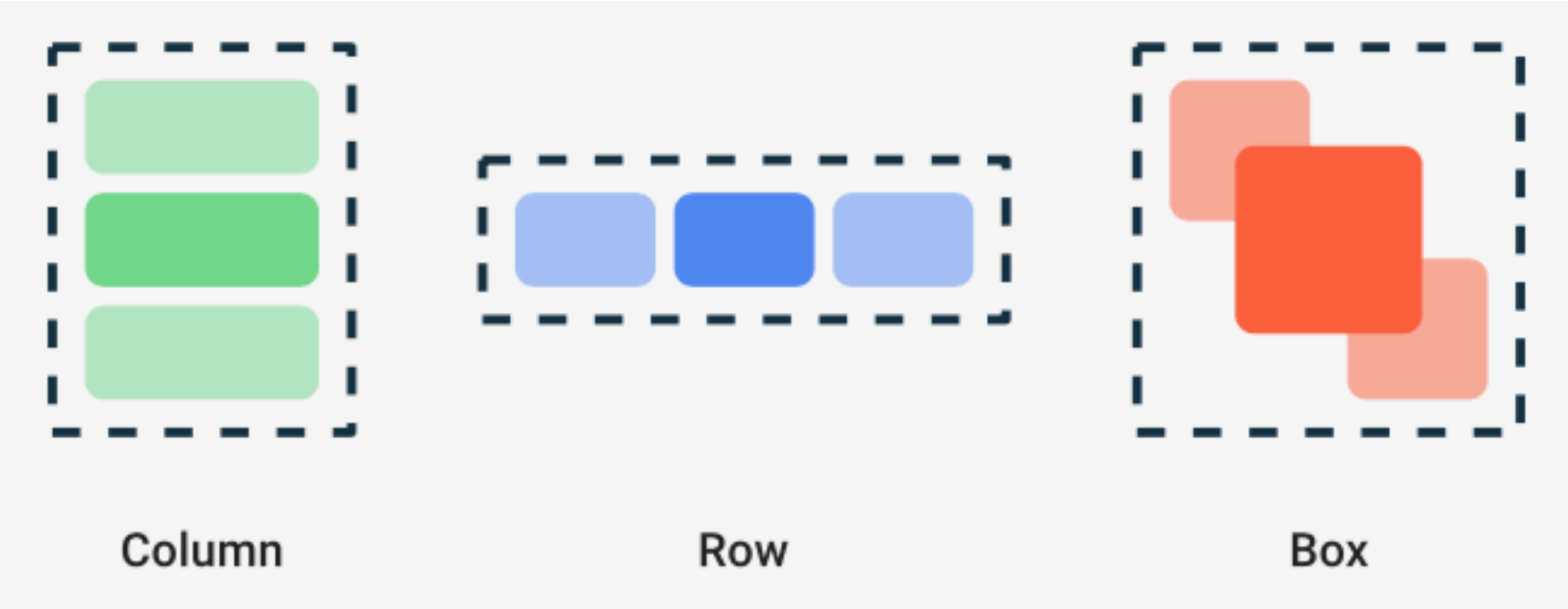
https://youtu.be/0yK7KoruhSM?si=723ewDTuy4BYzVsP

State → Composition → Layout → Drawing → UI



https://youtu.be/0yK7KoruhSM?si=723ewDTuy4BYzVsP

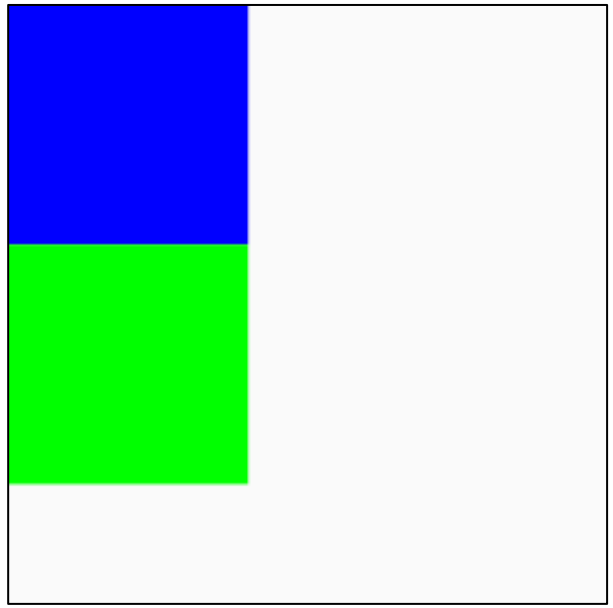
Standard Layouts: Column, Row, Box



https://developer.android.com/develop/ui/compose/layouts/basics

Standard Layouts: Arrangement and Alignment

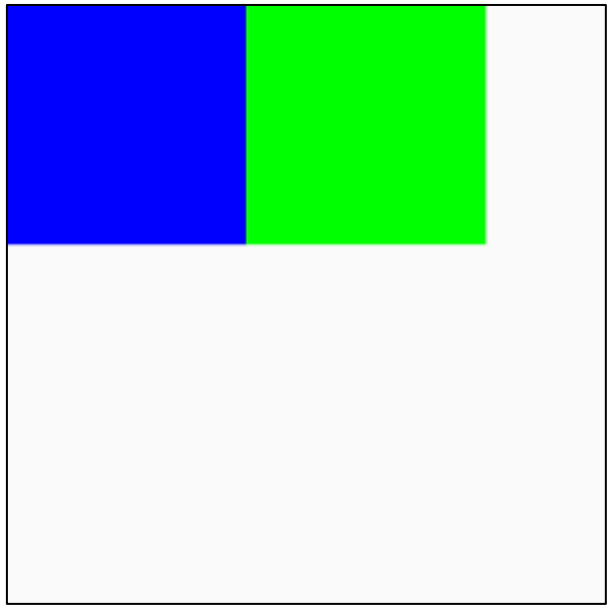
- Arrangement: primary axis
- Alignment: secondary axis



```
@Preview(showBackground = true)
@Composable
fun View() {
    Column(
        modifier = Modifier.size(100.dp),
        verticalArrangement = Arrangement.Top,
        horizontalAlignment = Alignment.Start,
    ) {
        Surface(
            modifier = Modifier.size(40.dp),
            color = Color.Blue
        ) {}
        Surface(
            modifier = Modifier.size(40.dp),
            color = Color.Green
        ) {}
    }
}
```

Standard Layouts: Arrangement and Alignment

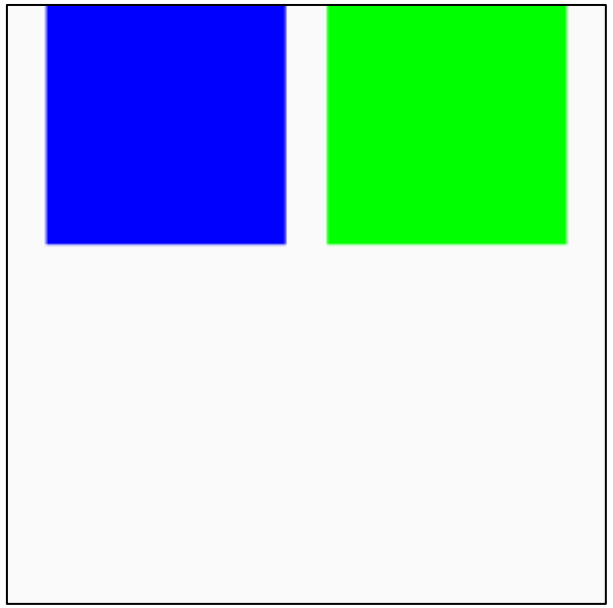
- Arrangement: primary axis
- Alignment: secondary axis



```
@Preview(showBackground = true)
@Composable
fun View() {
    Row(
        modifier = Modifier.size(100.dp),
        horizontalArrangement = Arrangement.Start,
        verticalAlignment = Alignment.Top,
    ) {
        Surface(
            modifier = Modifier.size(40.dp),
            color = Color.Blue
        ) {}
        Surface(
            modifier = Modifier.size(40.dp),
            color = Color.Green
        ) {}
    }
}
```

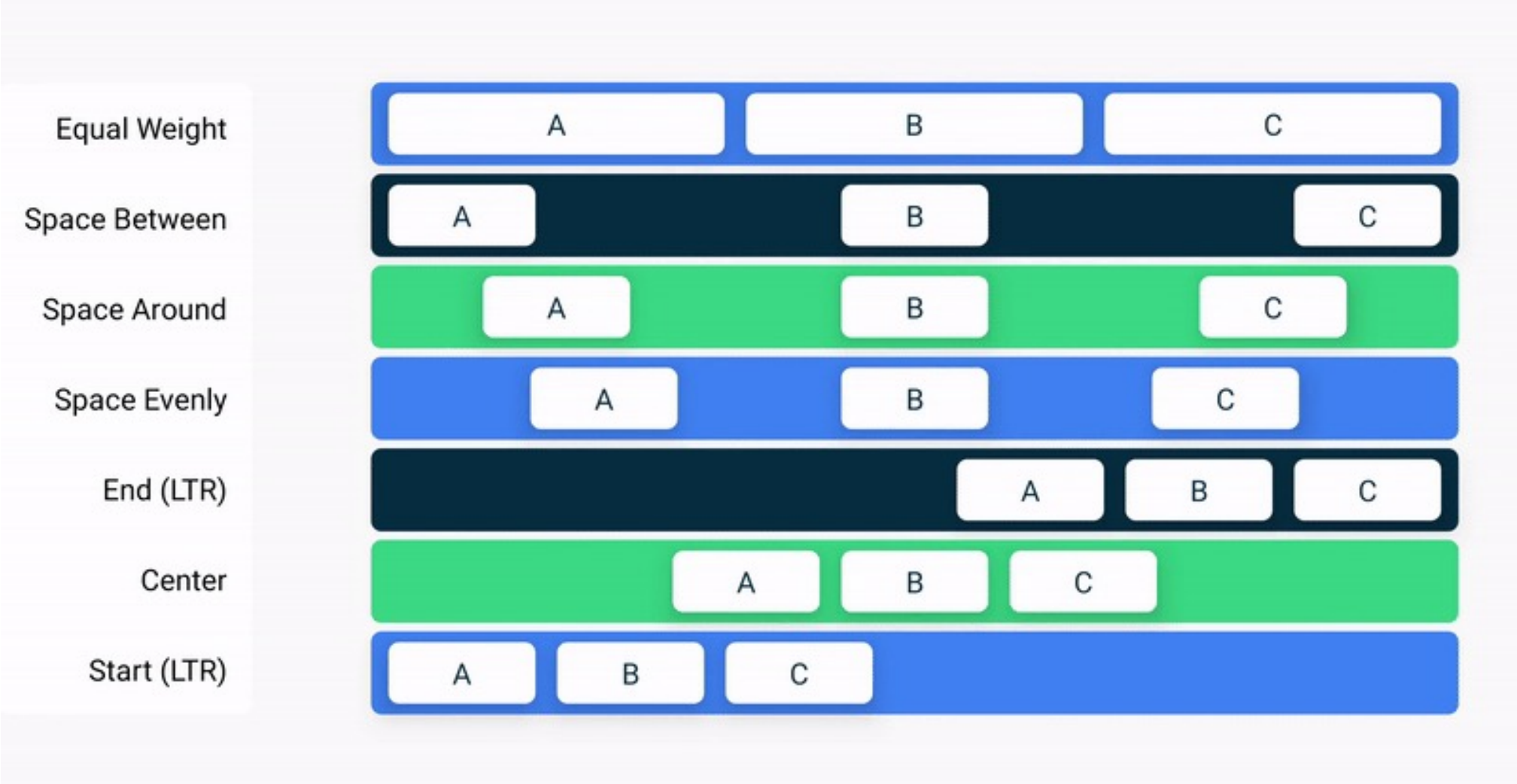
Standard Layouts: Arrangement and Alignment

- Arrangement: primary axis
- Alignment: secondary axis



```
@Preview(showBackground = true)
@Composable
fun View() {
    Row(
        modifier = Modifier.size(100.dp),
  horizontalArrangement = Arrangement.SpaceEvenly,
        verticalAlignment = Alignment.Top,
    ) {
        Surface(
            modifier = Modifier.size(40.dp),
            color = Color.Blue
        ) {}
        Surface(
            modifier = Modifier.size(40.dp),
            color = Color.Green
        ) {}
    }
}
```

Standard Layouts: Arrangement and Alignment



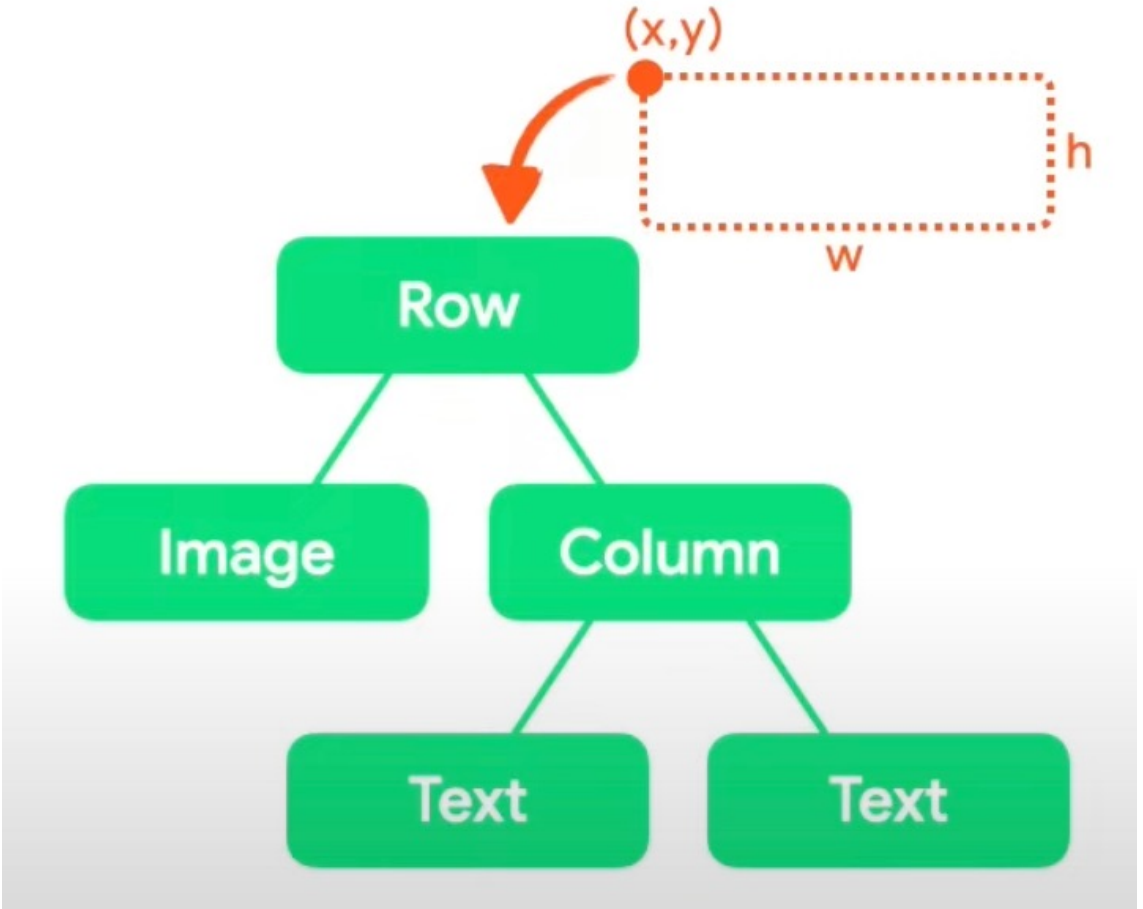
https://developer.android.com/reference/kotlin/androidx/compose/foundation/layout/Arrangement

The Layout Process

Each node in tree

1. Measure children
2. Compute own size
3. Place children

Layout nodes can layout multiple children



https://youtu.be/0yK7KoruhSM?si=723ewDTuy4BYzVsP

Modifiers

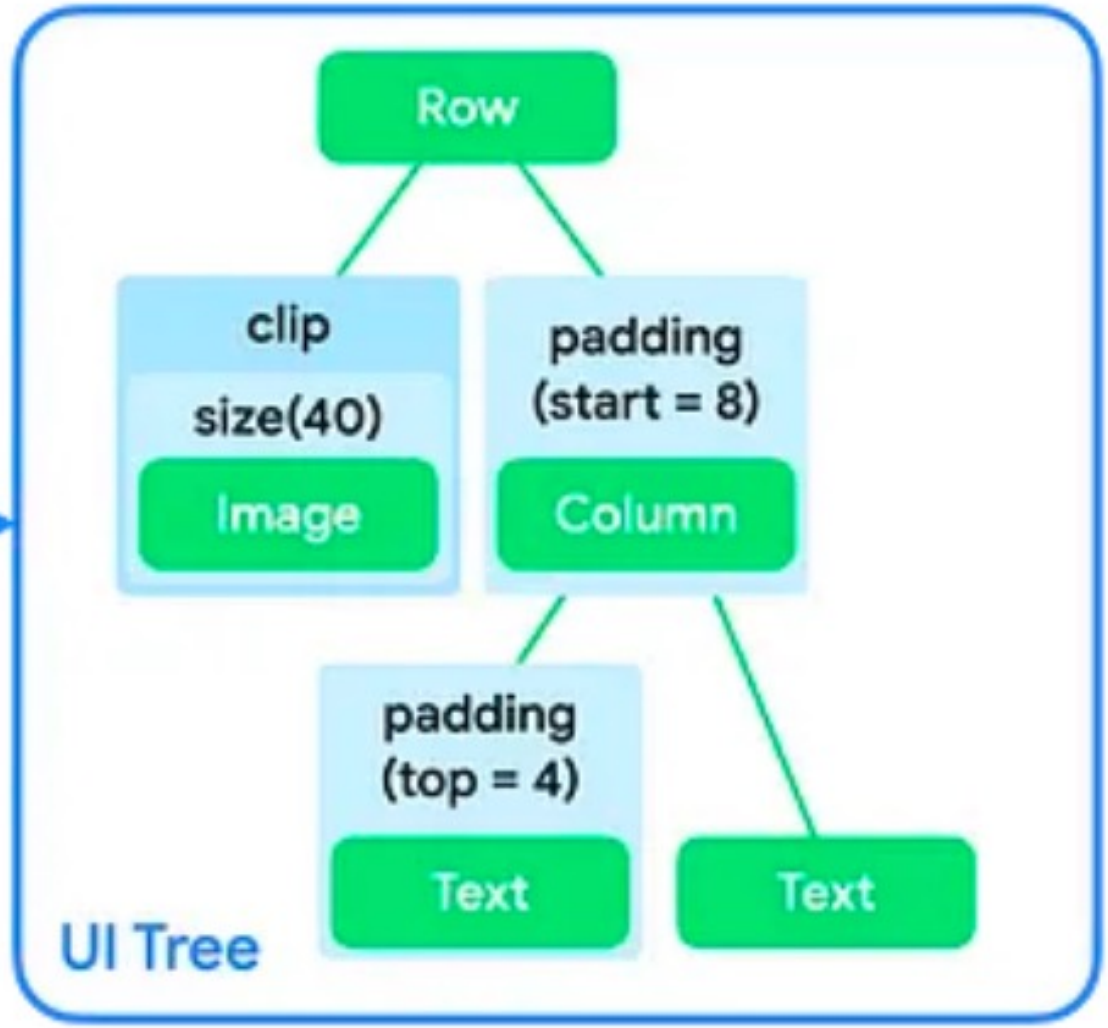
- Decorate or augment a composable

- Change size, layout, behavior, and appearance
- Add accessibility labels
- Process user input
- Add interactions: clickable, scrollable, draggable, zoomable

- Modifiers are Kotlin objects
- Modifiers can be chained: linked list of modifier elements
  - Example: Set padding, make clickable, set background color

A modifier wraps another modifier or a layout node

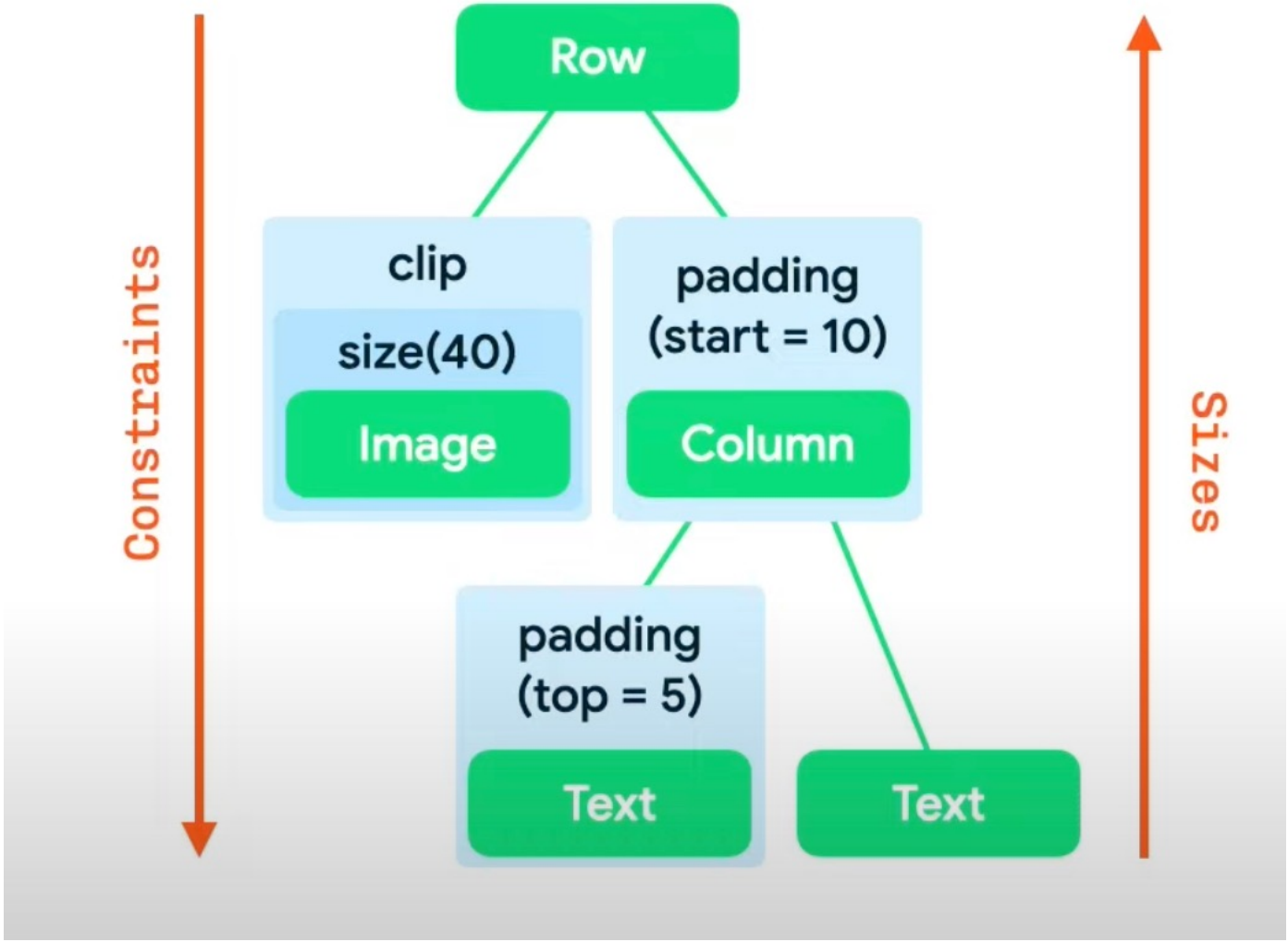
```
Row {
  Image(
    ..,
    Modifier
      .clip(CircleShape)
      .size(40.dp)
  )
  Column(
    Modifier.padding(start = 8.dp)
  ) {
    Text(
      ..,
      Modifier.padding(top = 4.dp)
    )
    Text(..)
  }
}
```



https://developer.android.com/develop/ui/compose/layouts/constraints-modifiers

Constraints

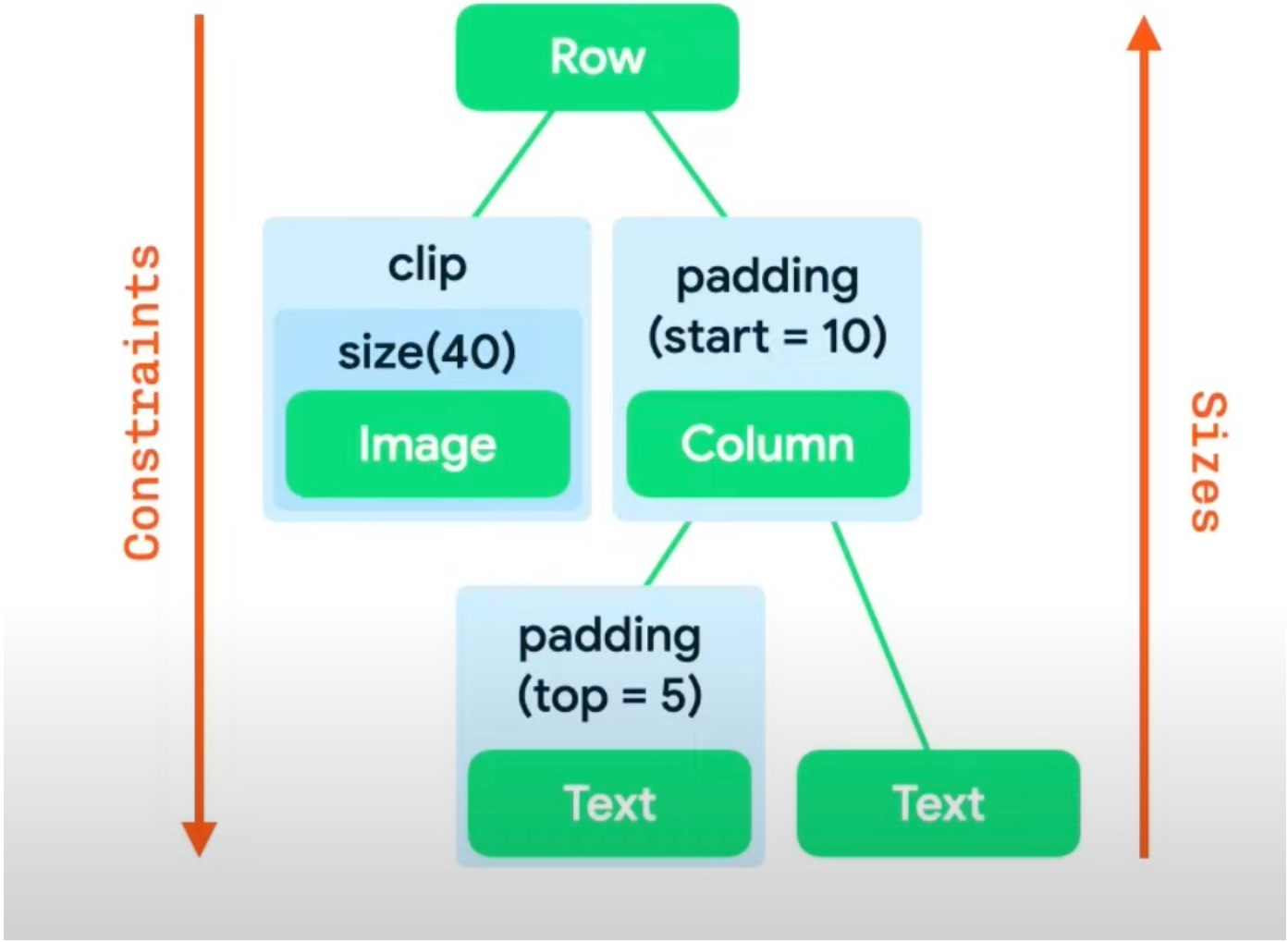
- Min/max width/height of a node
- Passed down from parent to children during layout phase
- A modifier can change the constraints



https://www.youtube.com/watch?v=0eC5jMV342A

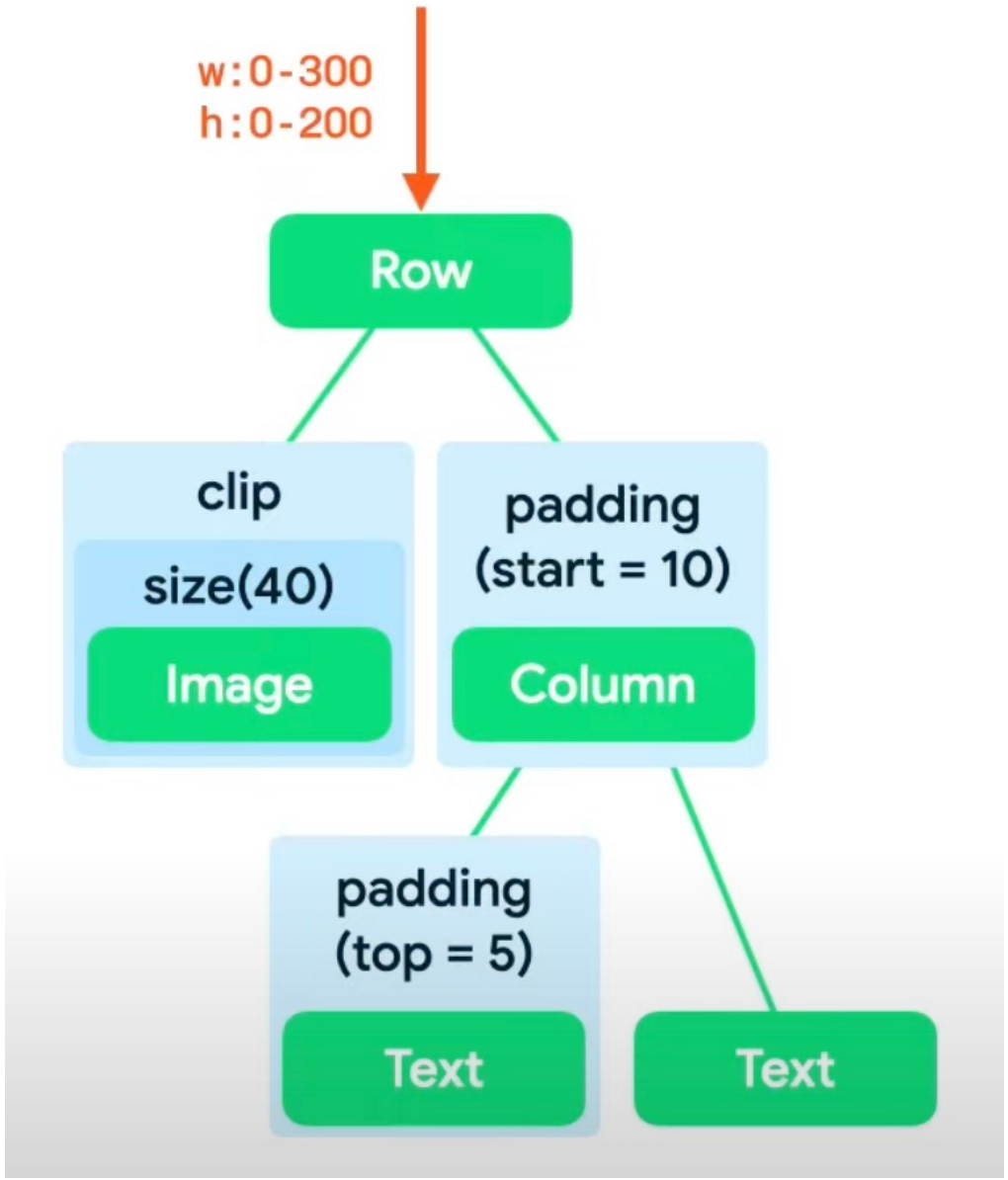
Constraints

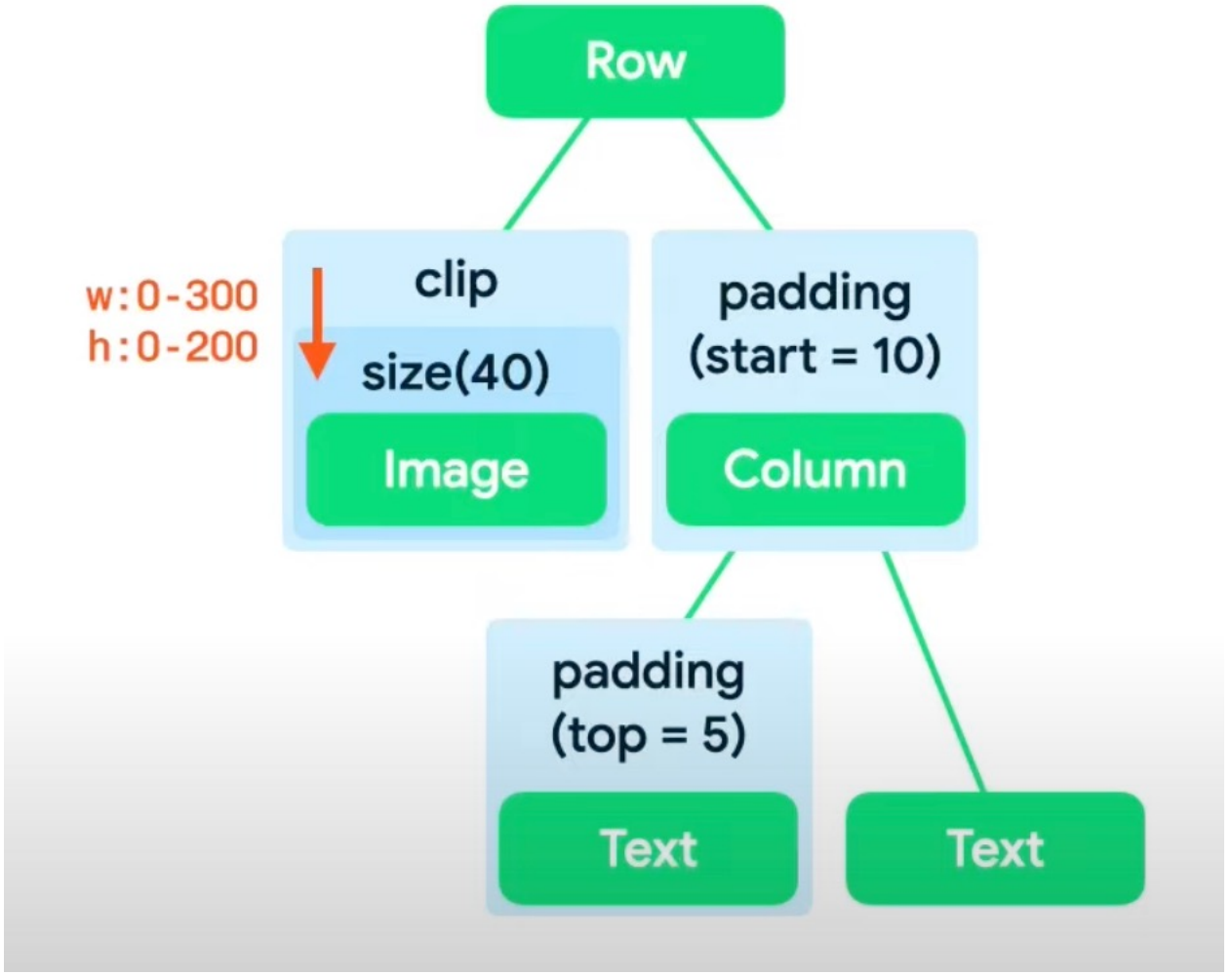
- Depth-first traversal of layout tree, passing down constraints
- A modifier may adjust the constraints
- A leaf computes its size based on the incoming constraints
- When all children of a node are measured, node computes its own size, returns that to its parent



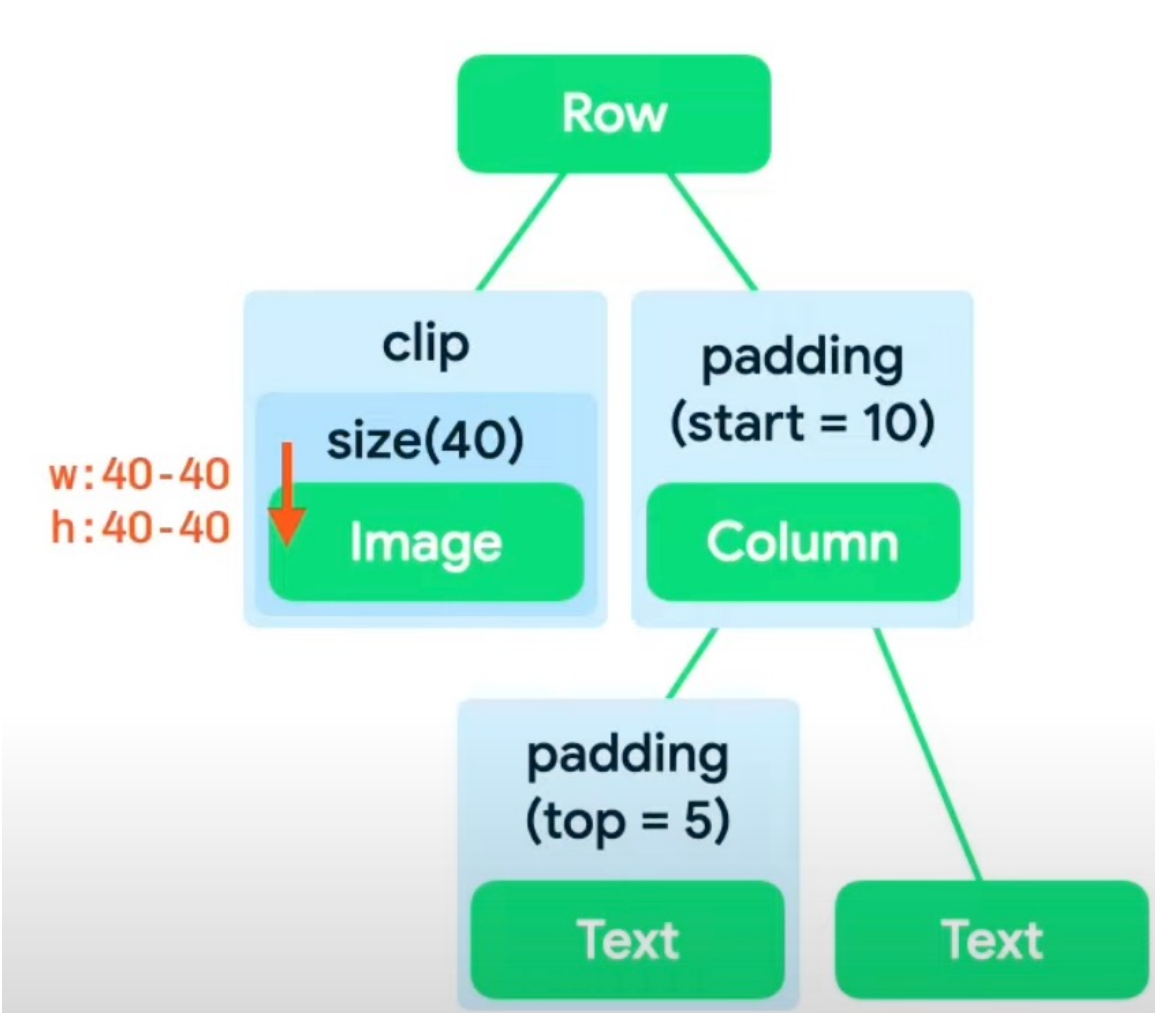
https://www.youtube.com/watch?v=0eC5jMV342A

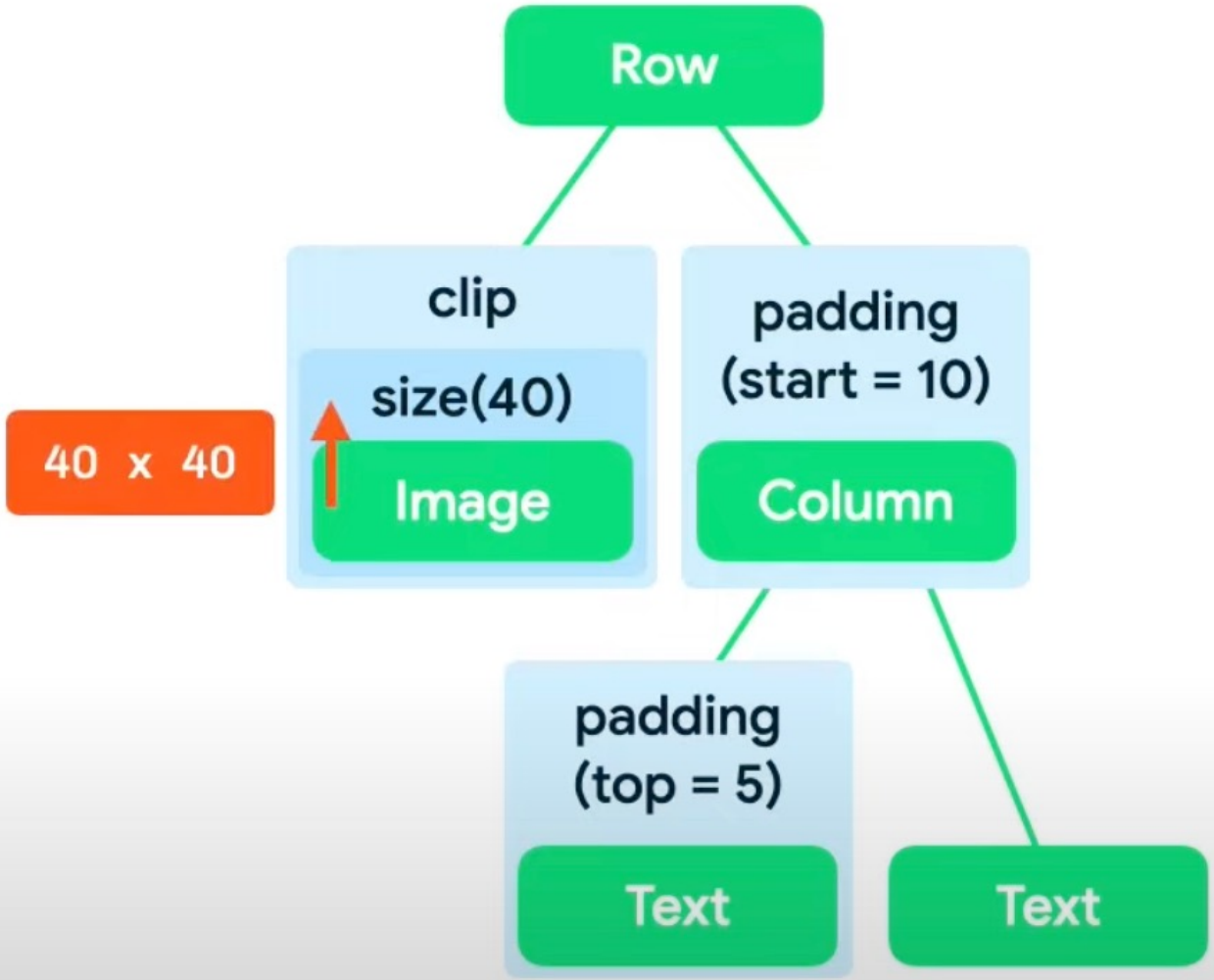
https://www.youtube.com/watch?v=0eC5jMV342A



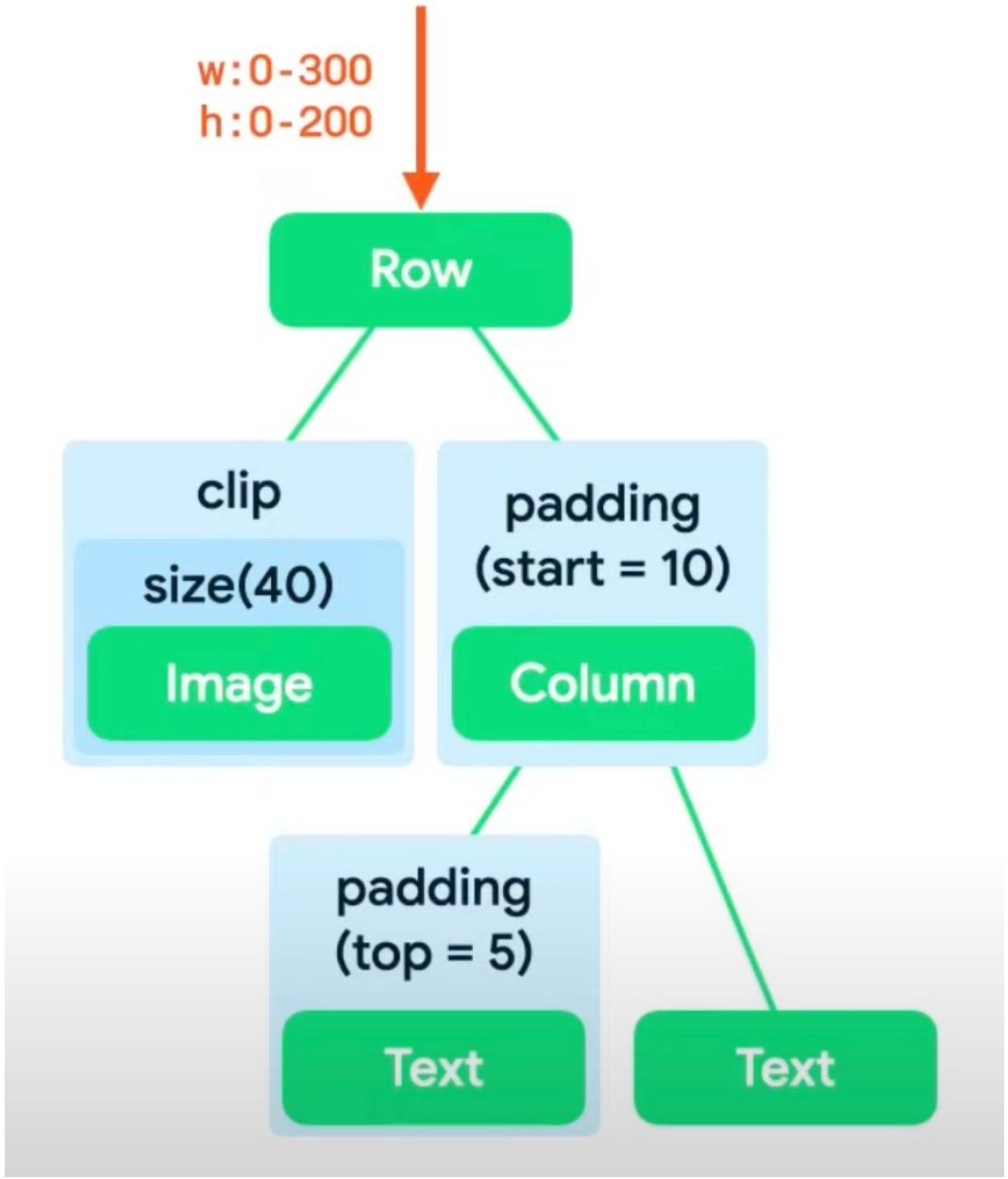


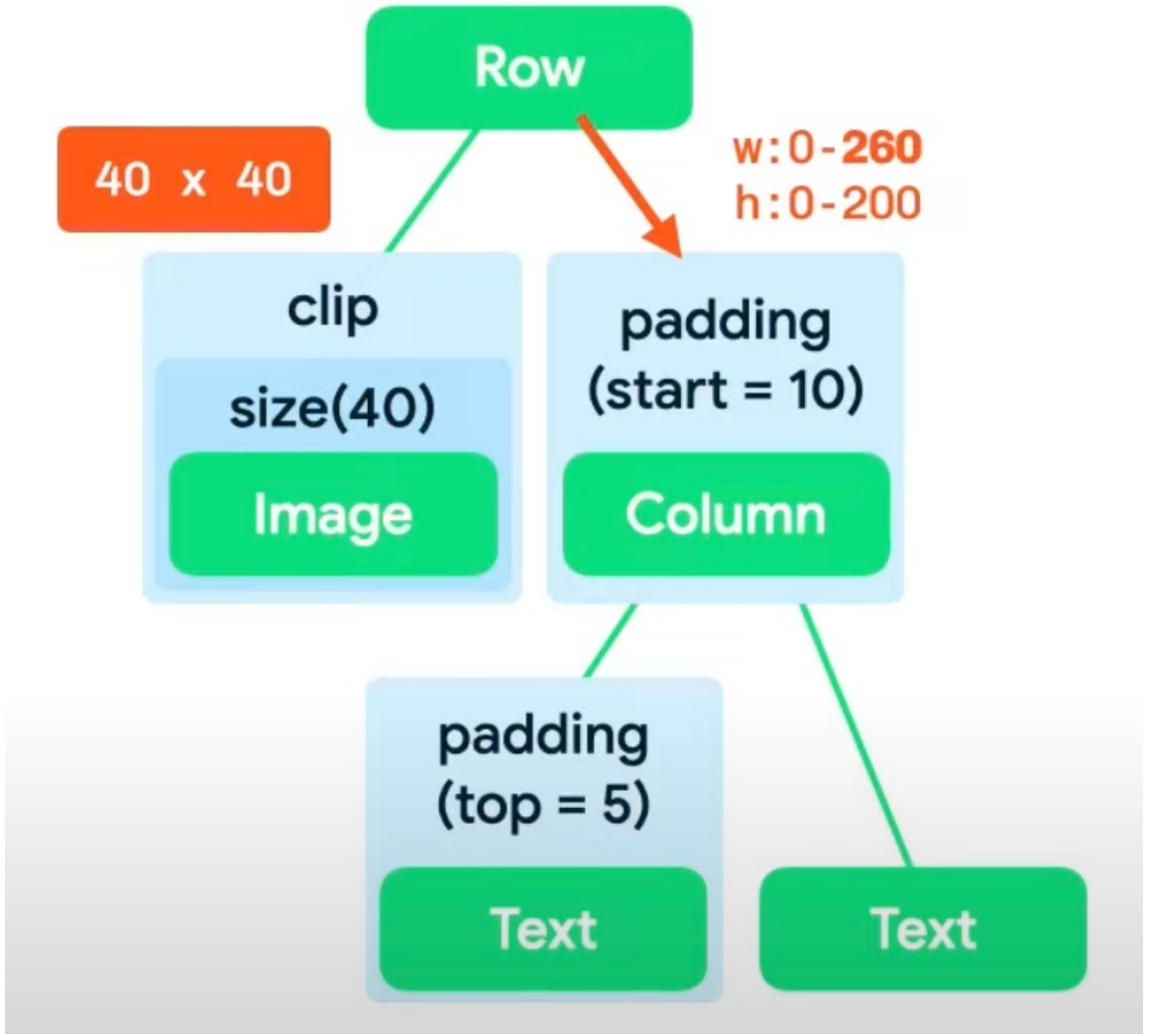
https://www.youtube.com/watch?v=0eC5jMV342A



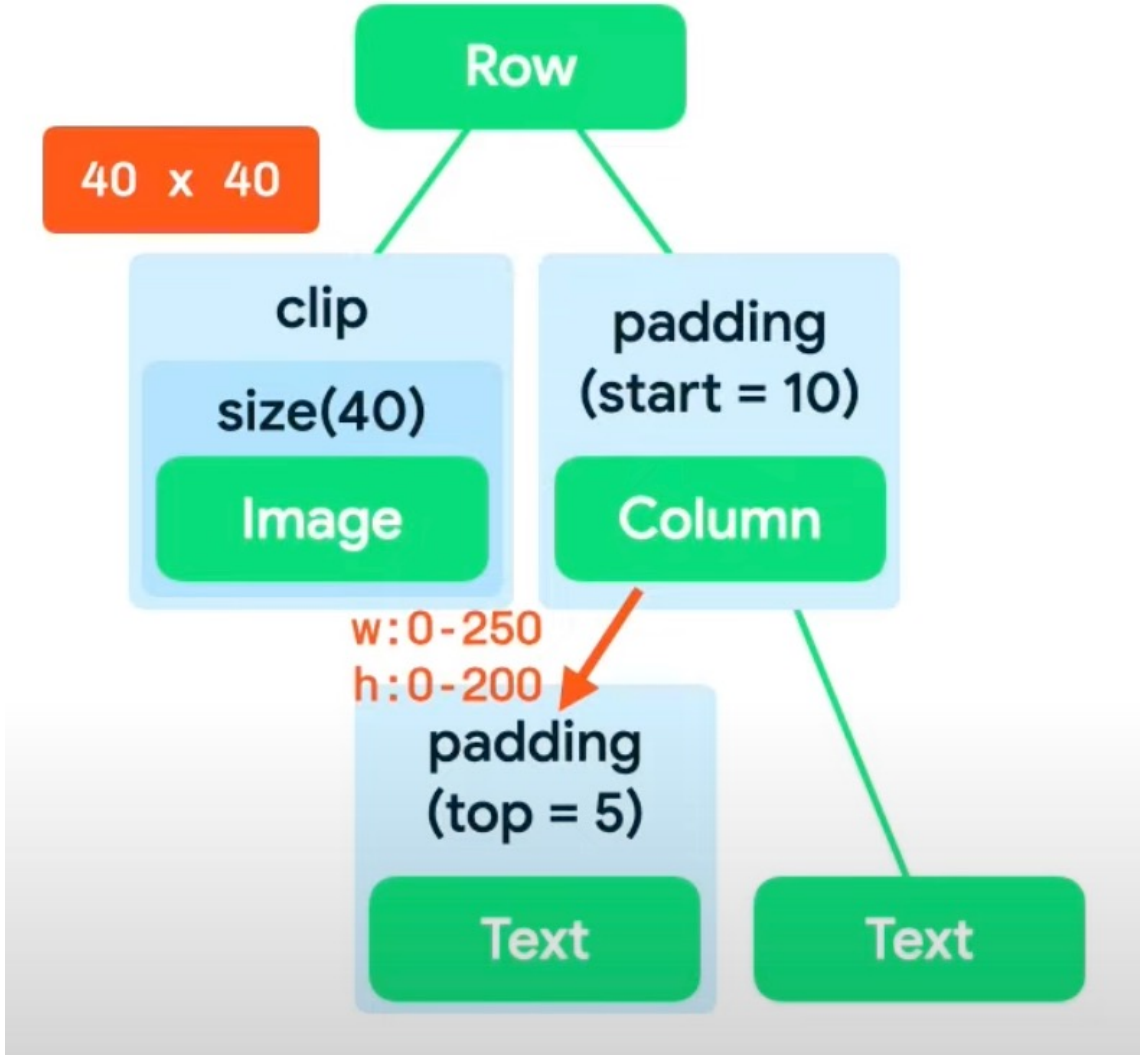


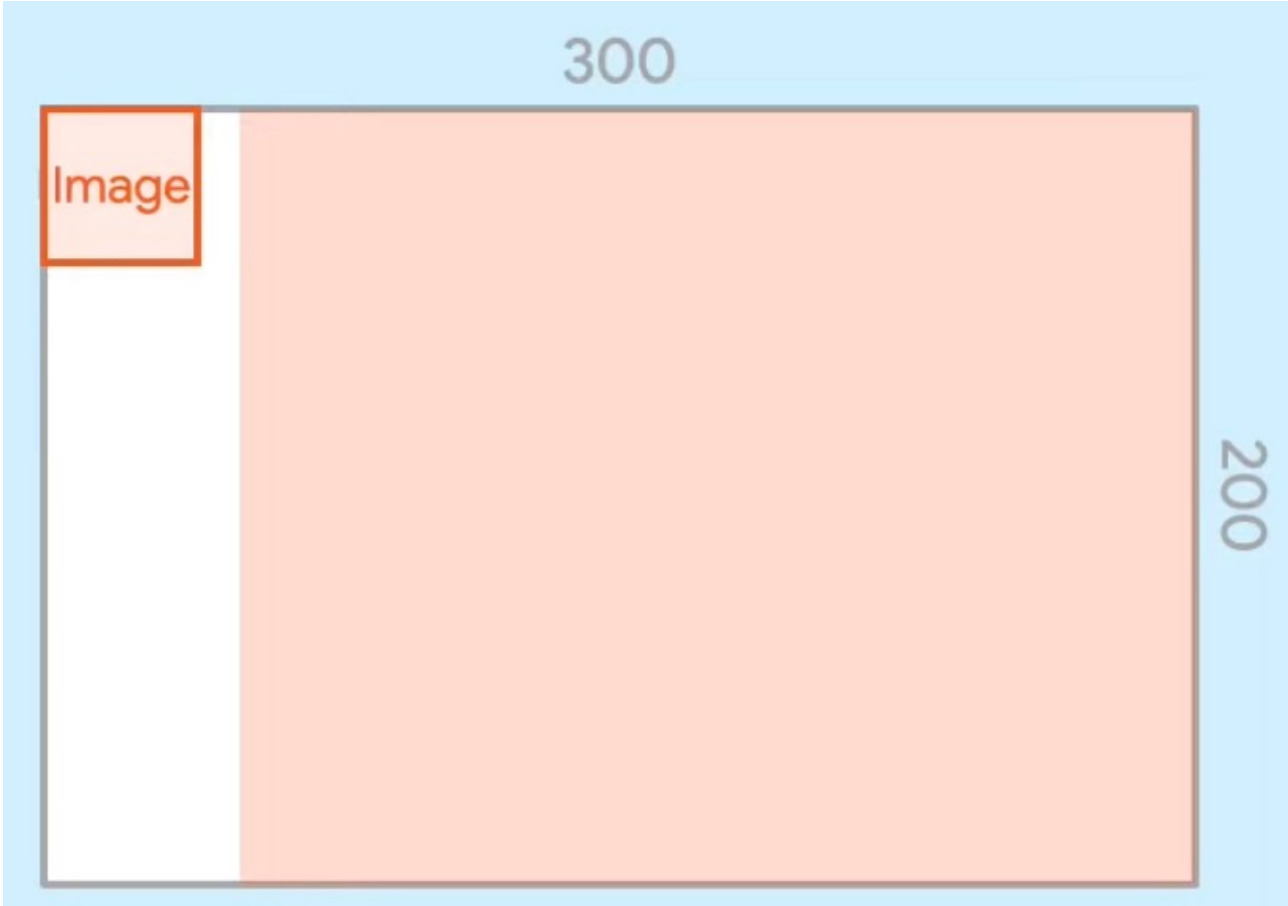
https://www.youtube.com/watch?v=0eC5jMV342A



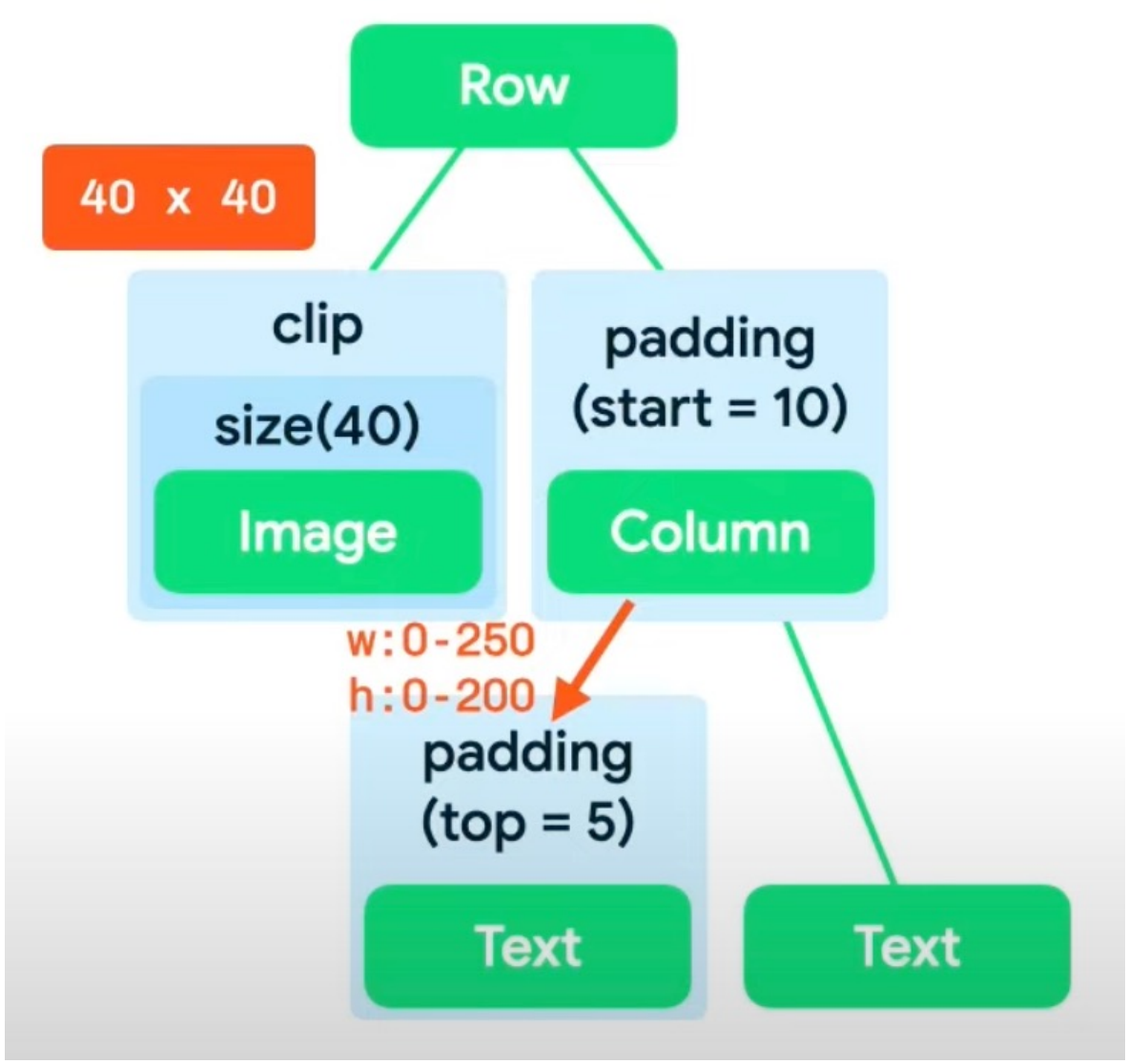


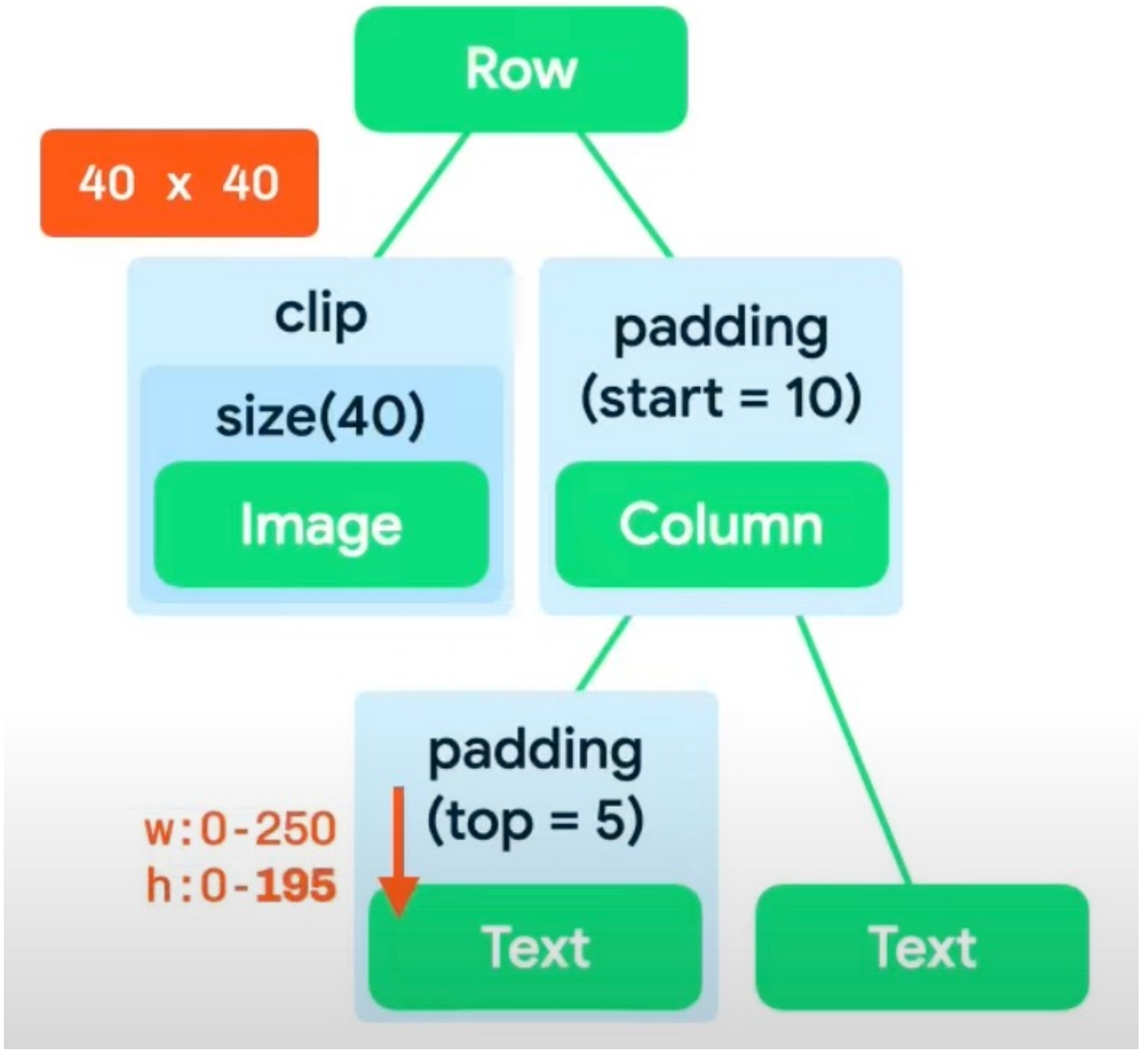
https://www.youtube.com/watch?v=0eC5jMV342A



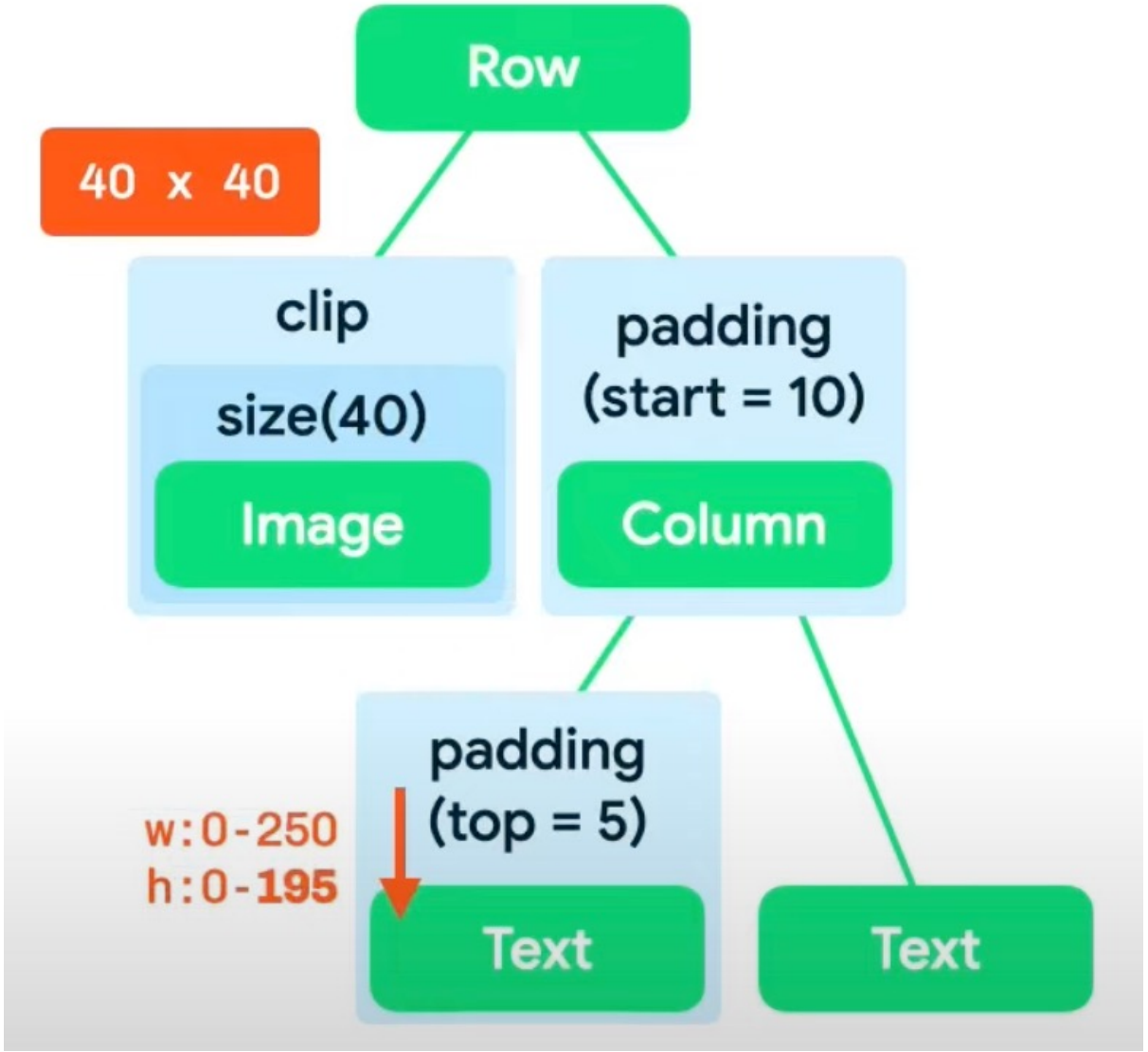


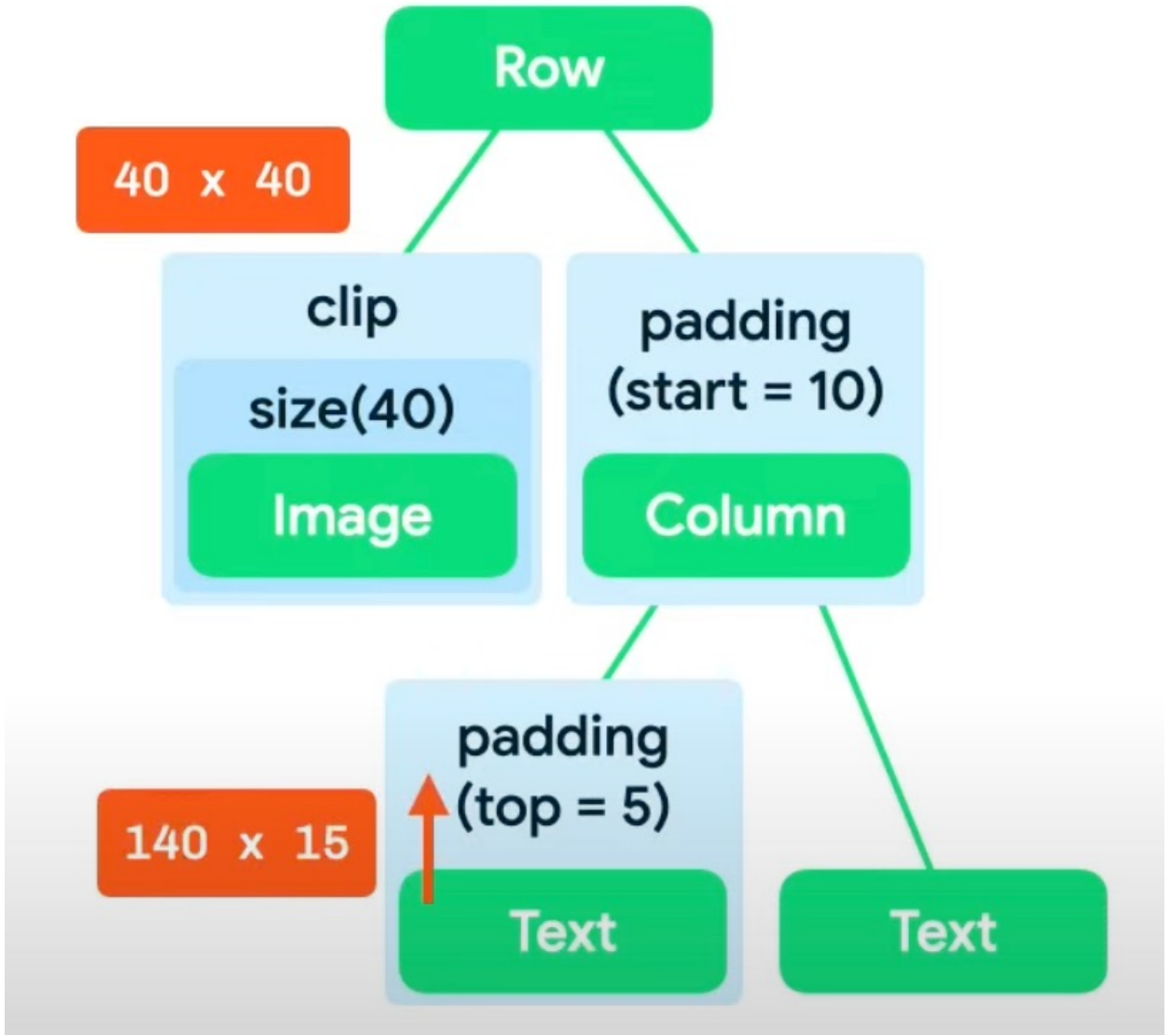
https://www.youtube.com/watch?v=0eC5jMV342A



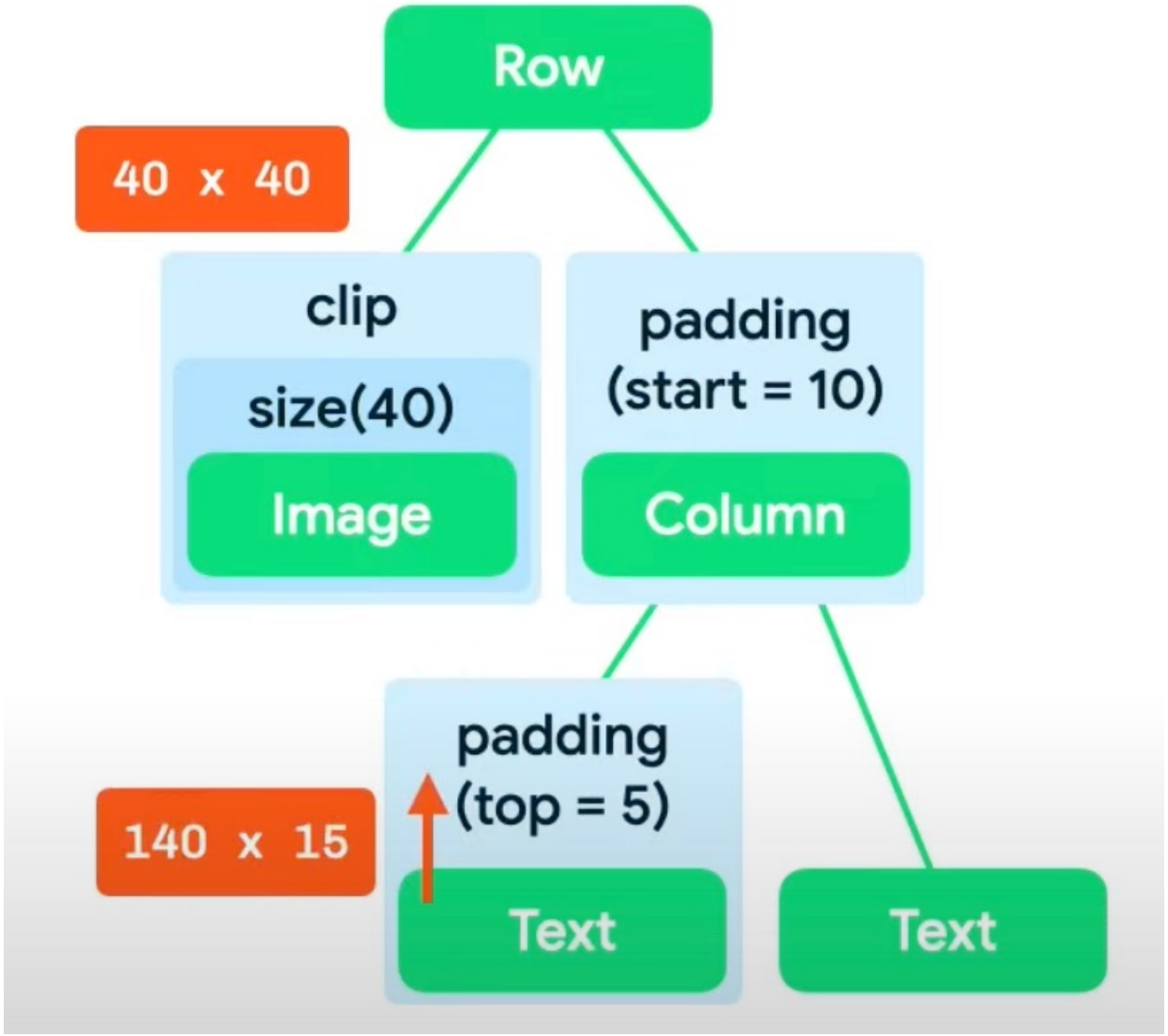


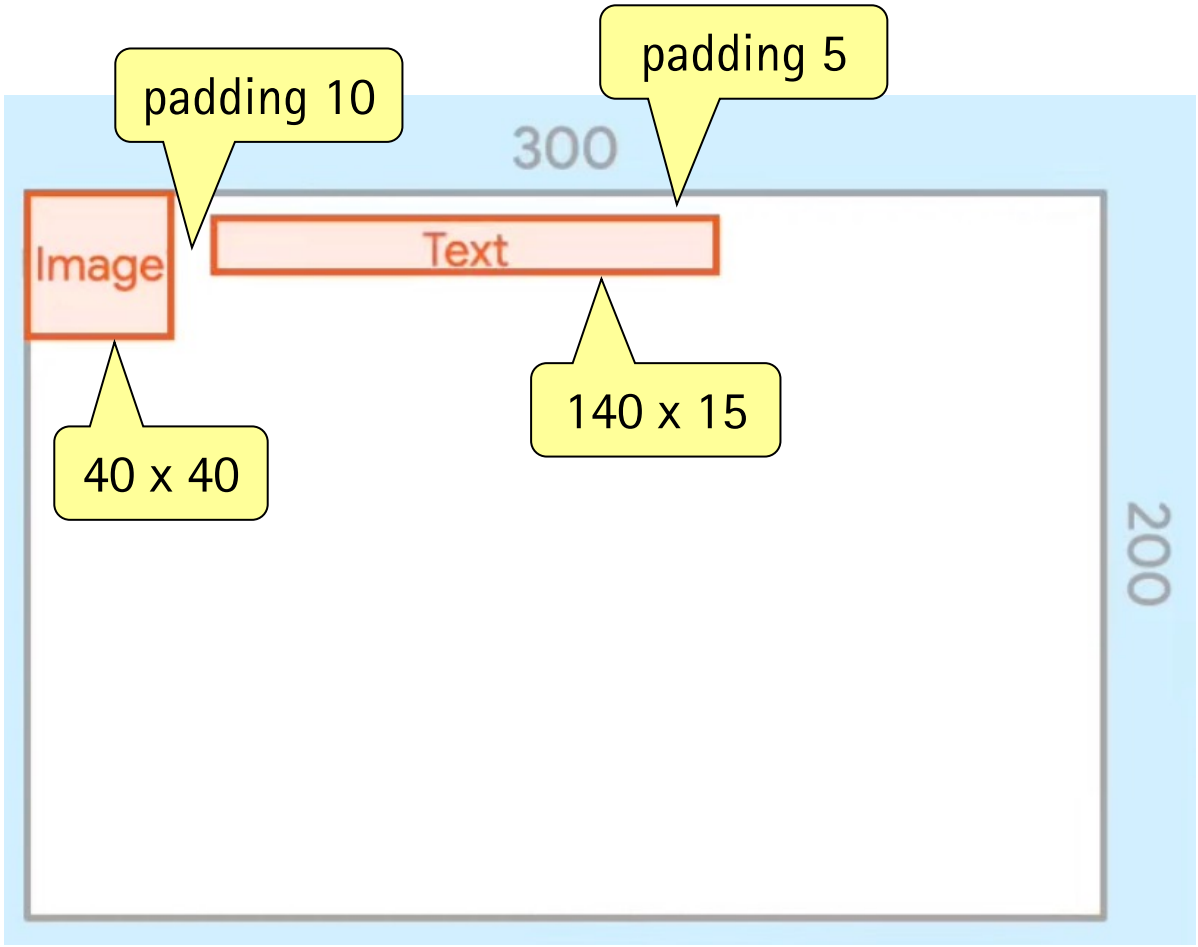
https://www.youtube.com/watch?v=0eC5jMV342A



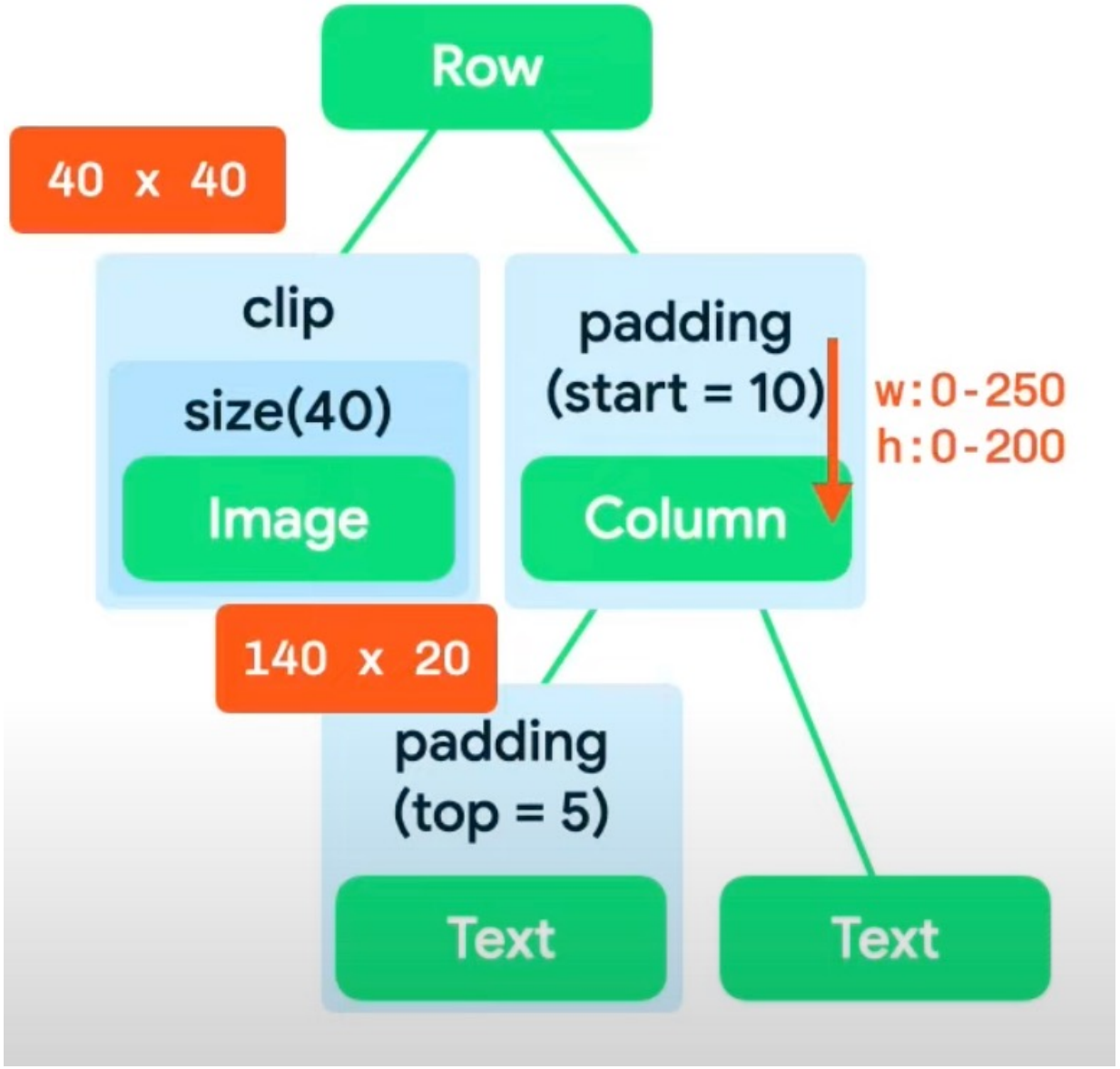


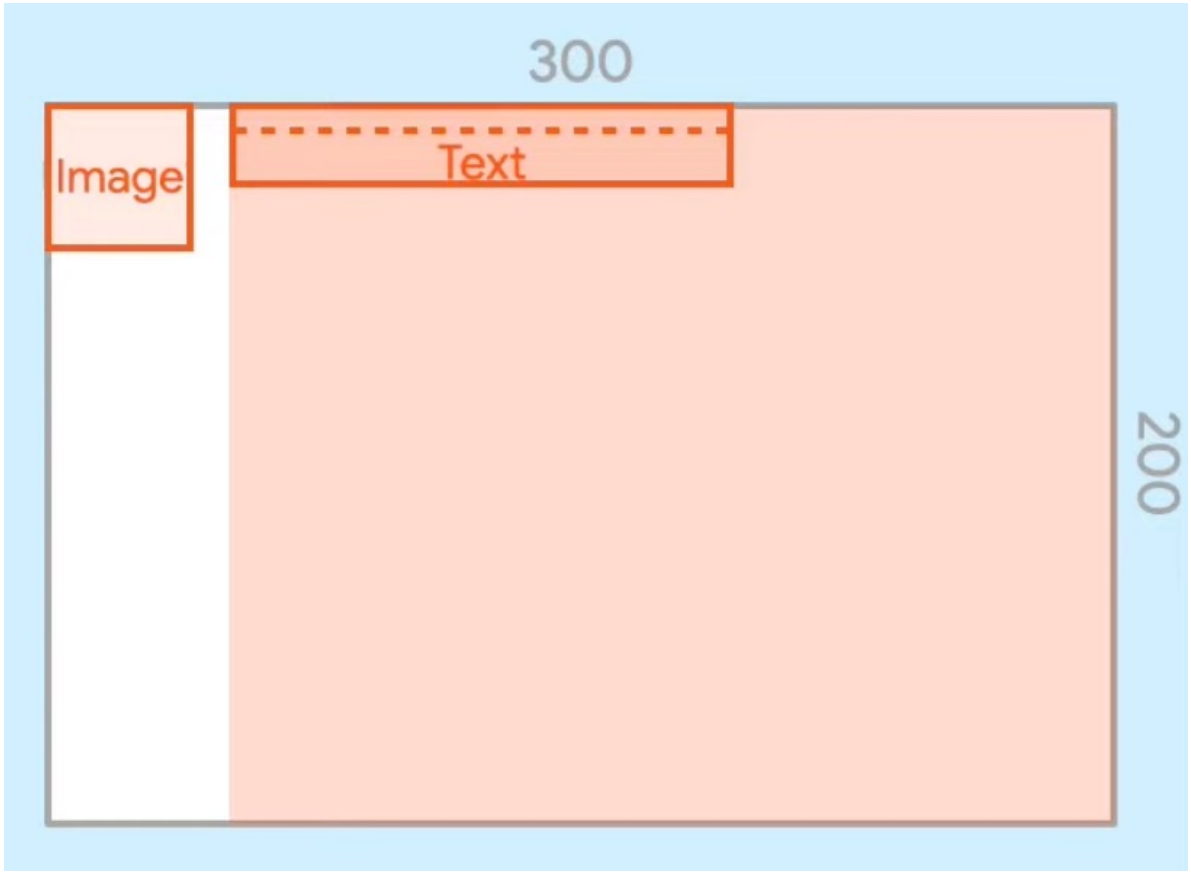
https://www.youtube.com/watch?v=0eC5jMV342A



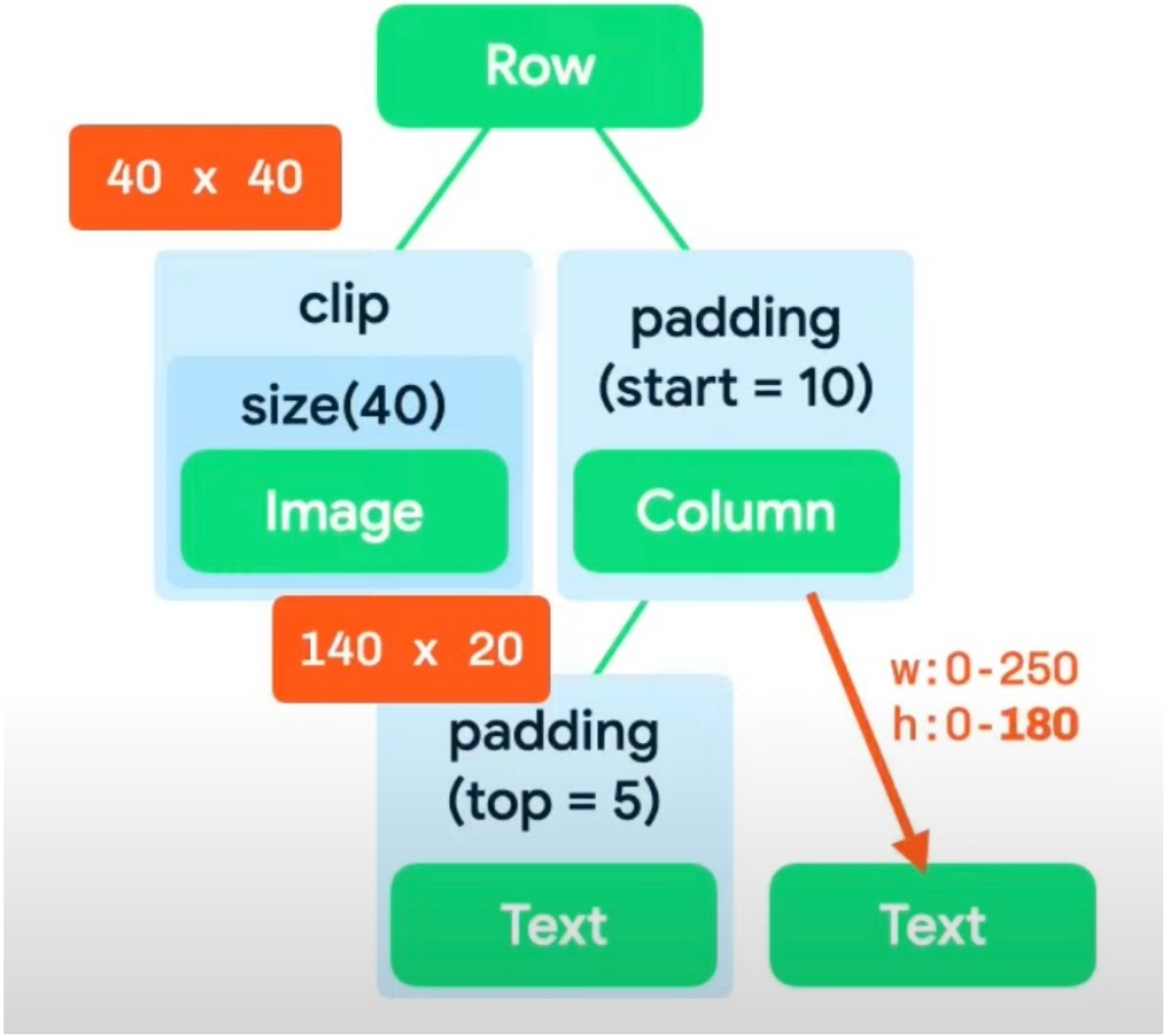


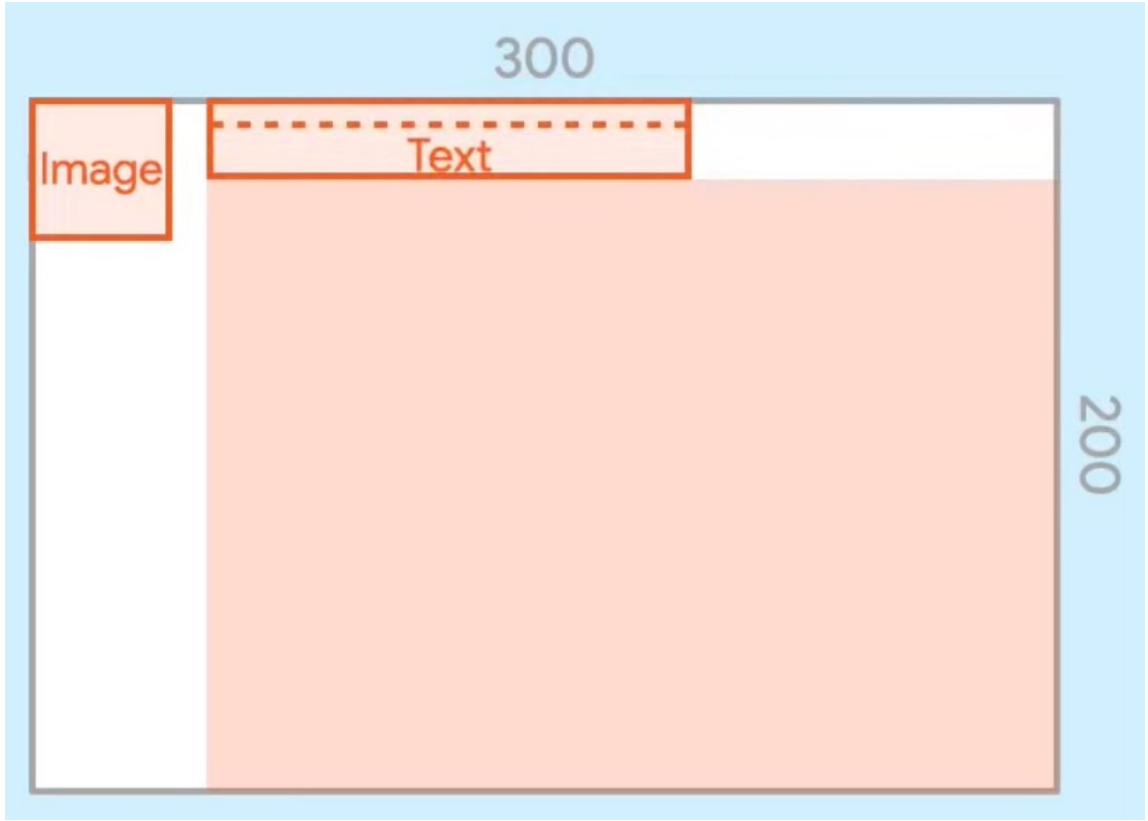
https://www.youtube.com/watch?v=0eC5jMV342A



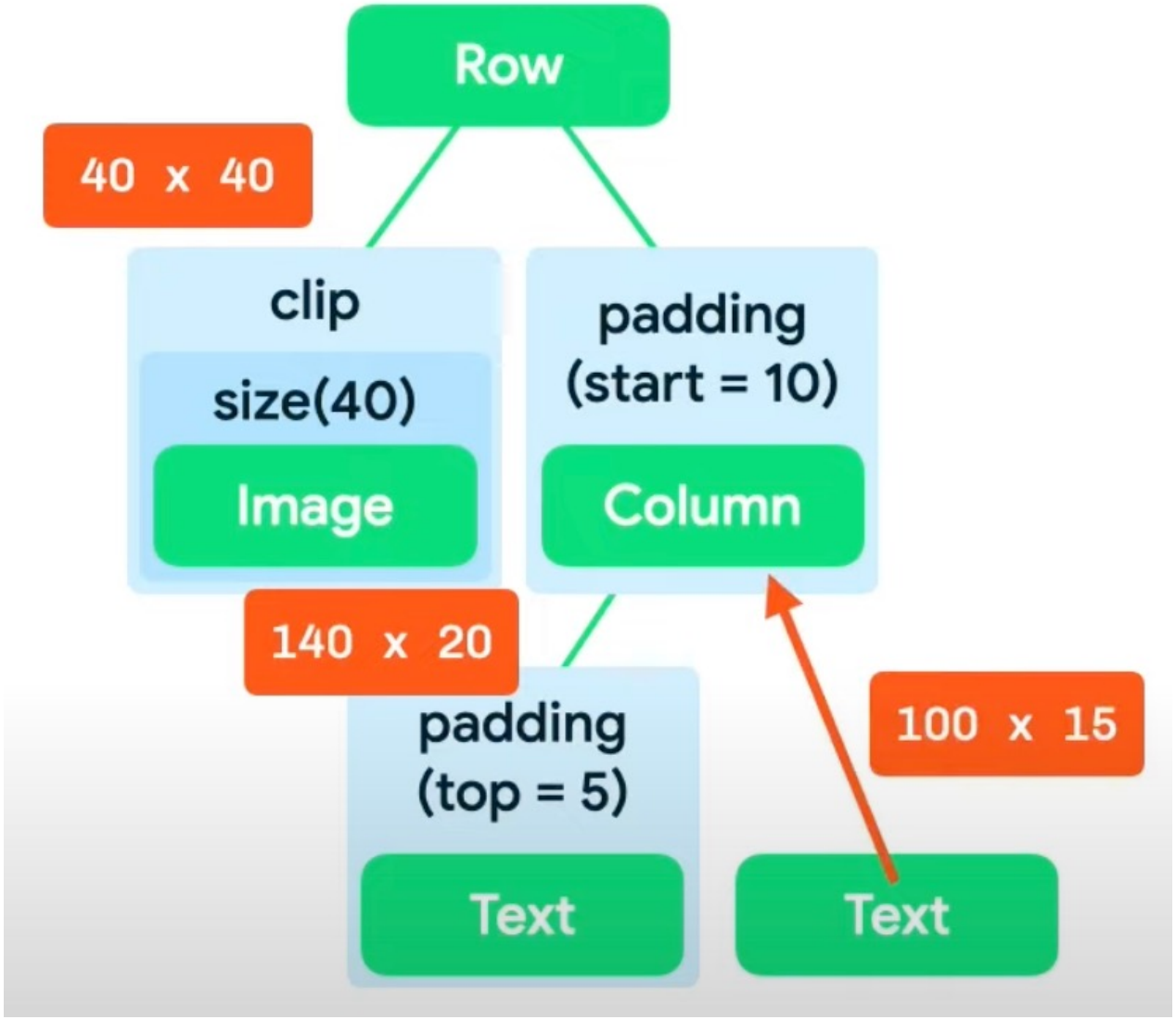


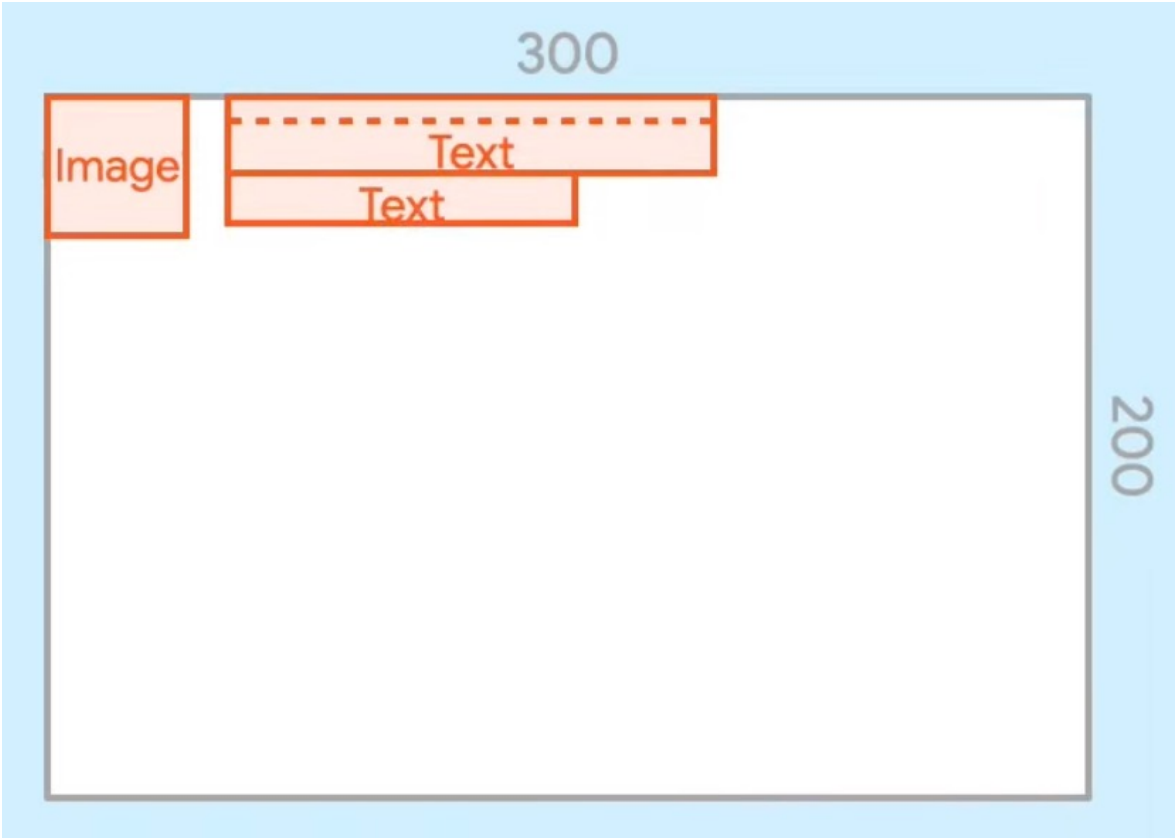
https://www.youtube.com/watch?v=0eC5jMV342A



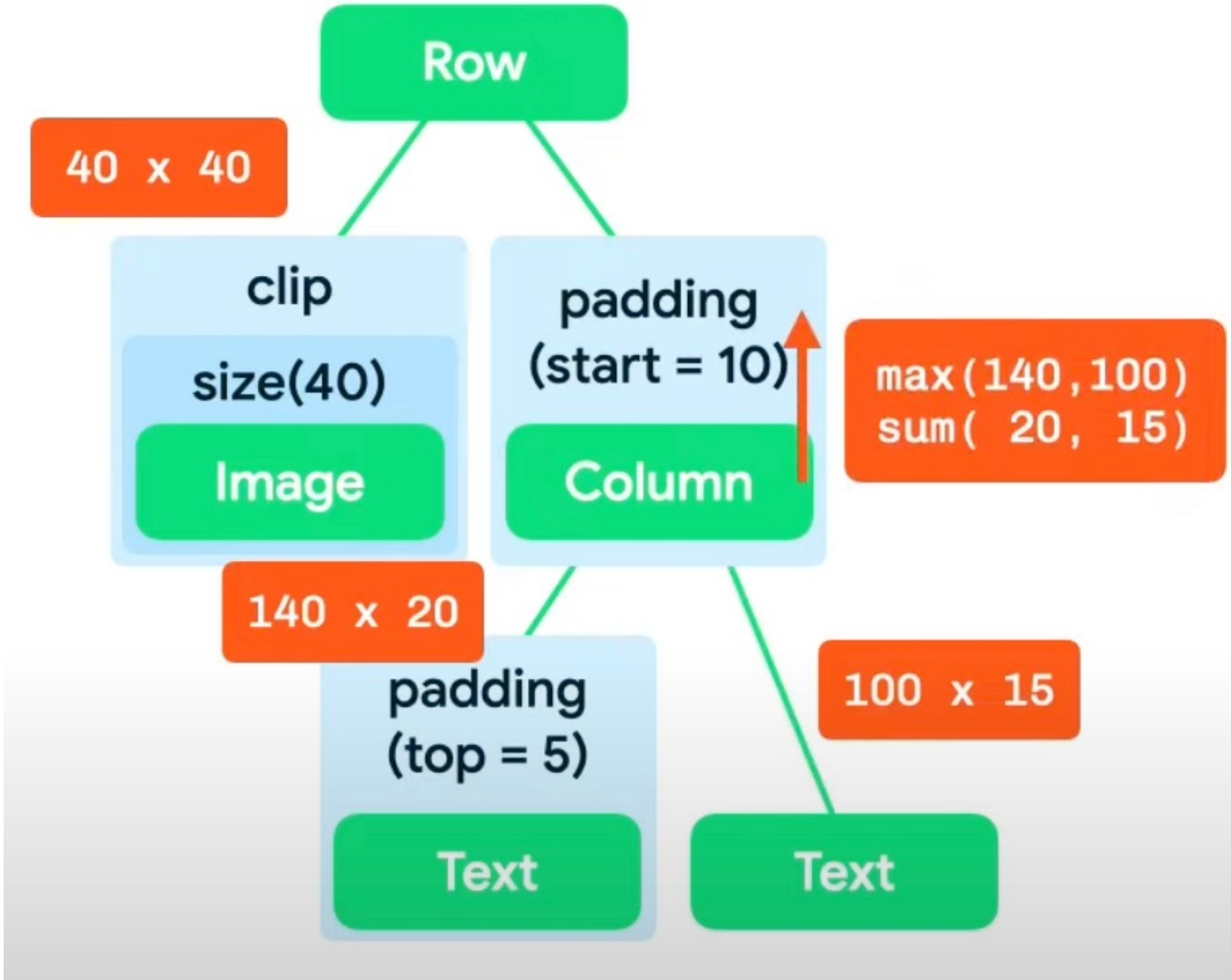


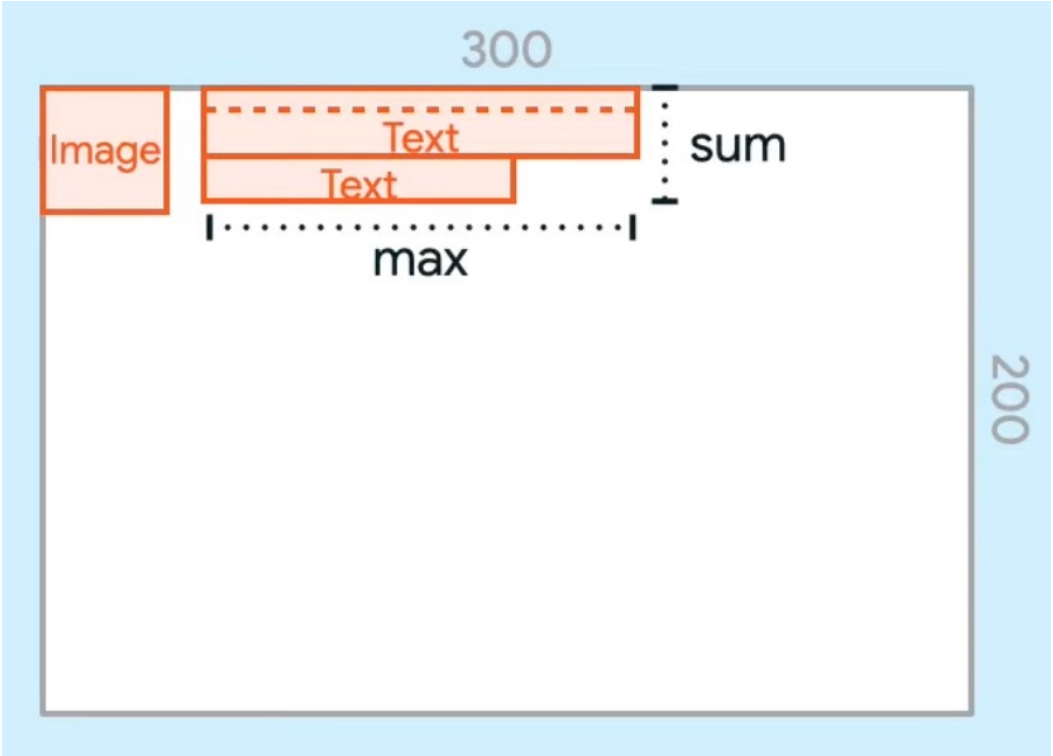
https://www.youtube.com/watch?v=0eC5jMV342A



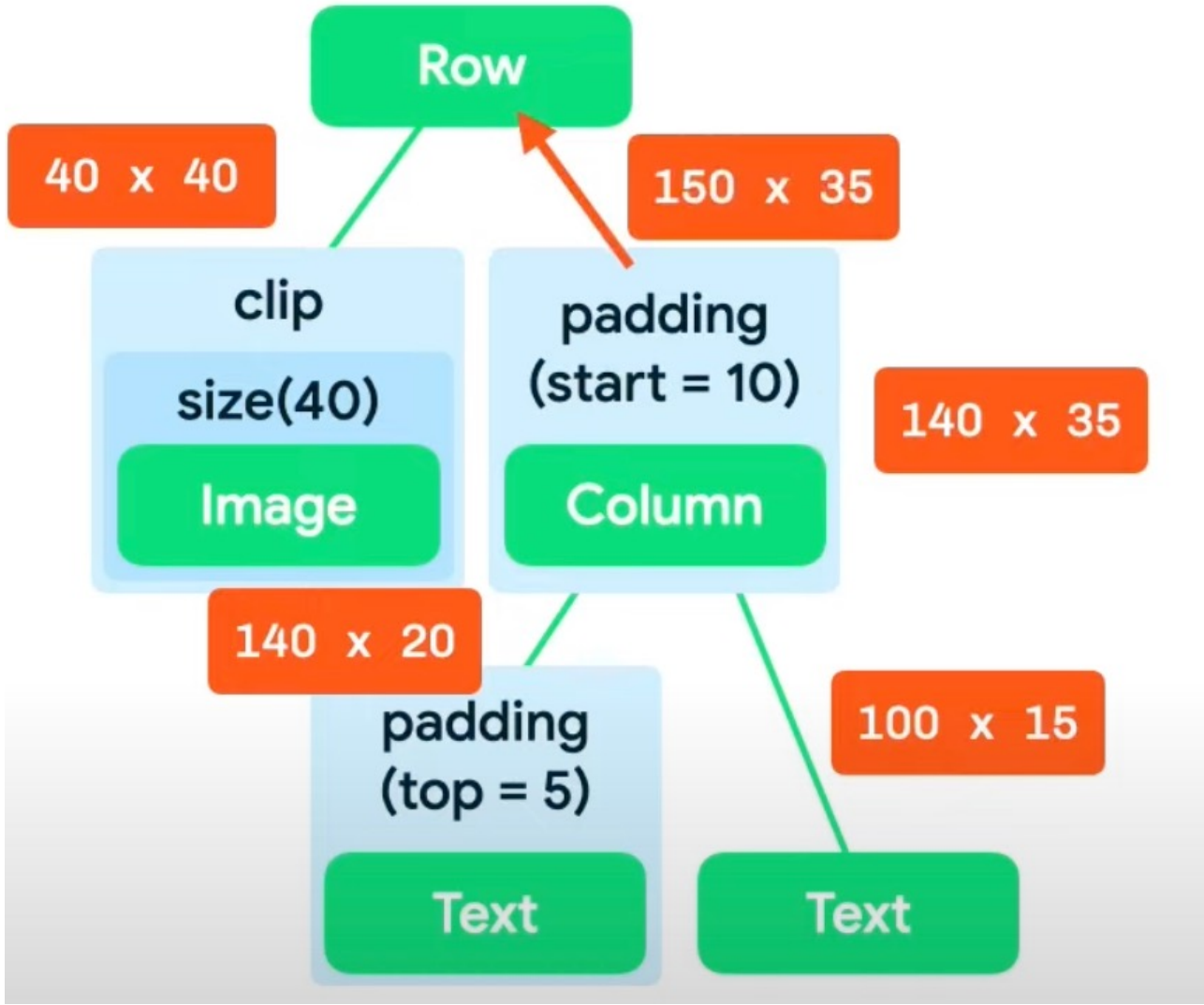


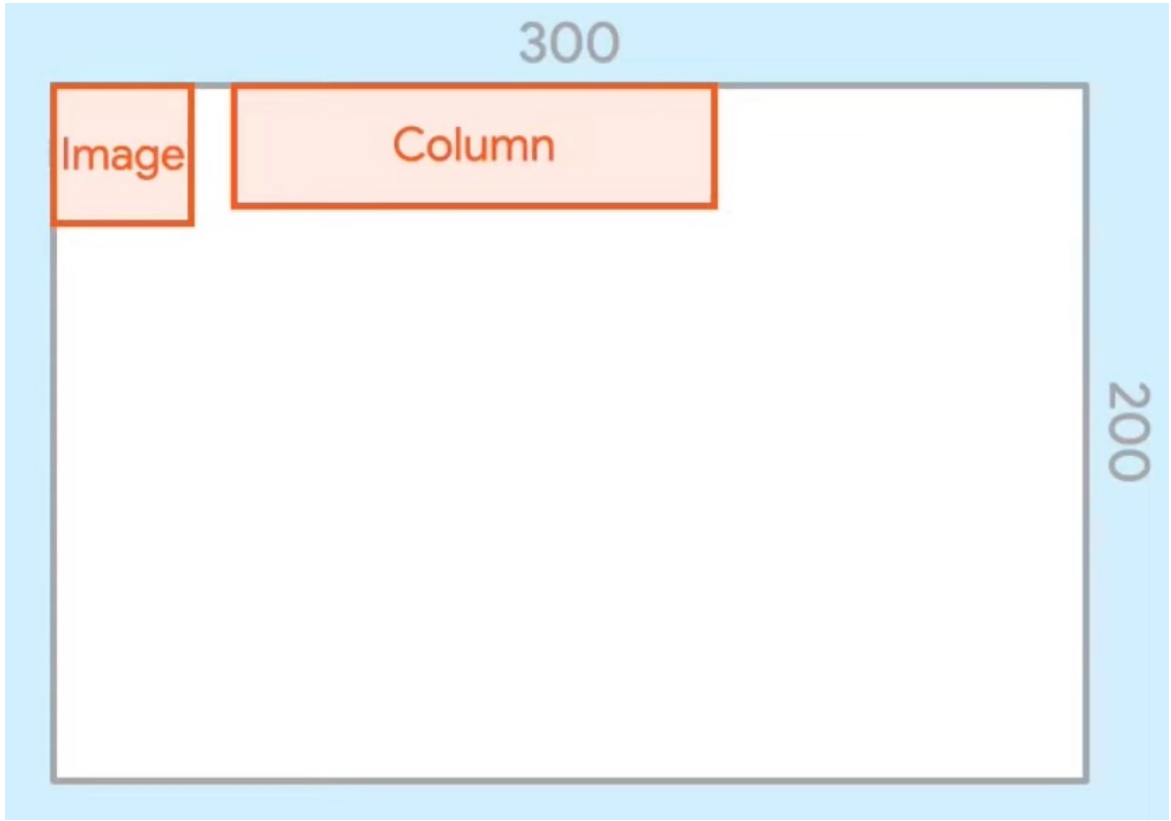
https://www.youtube.com/watch?v=0eC5jMV342A



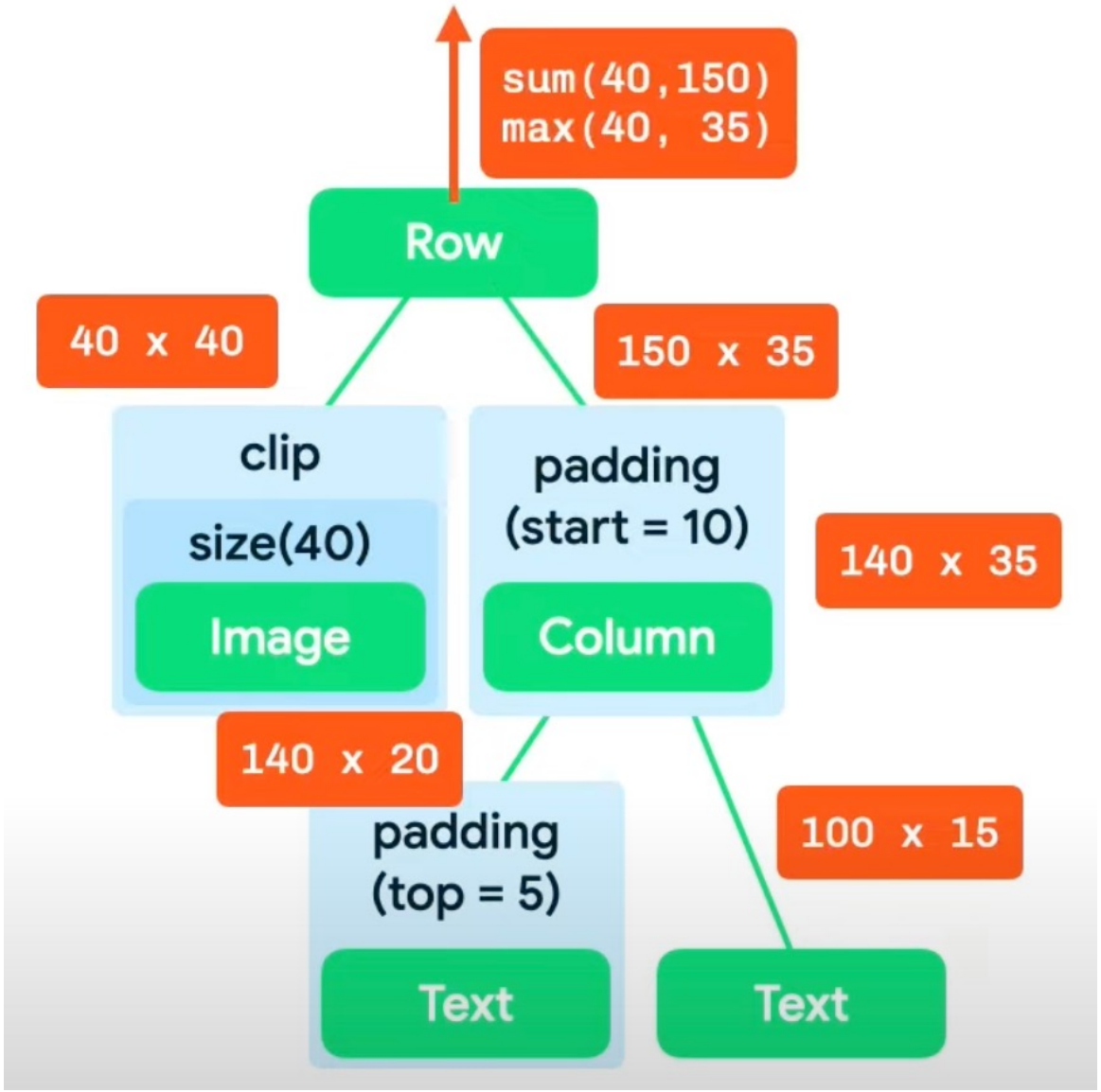


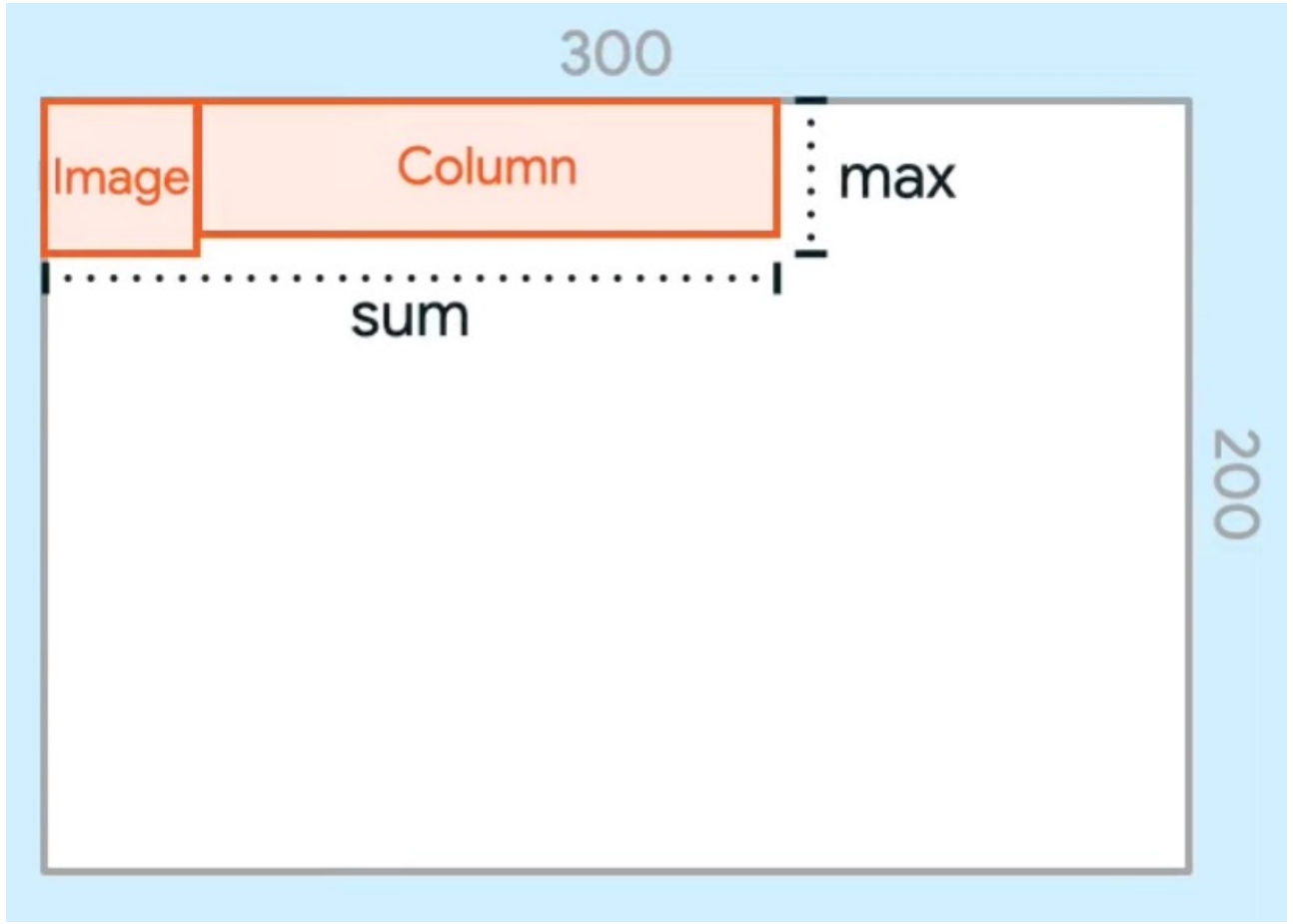
https://www.youtube.com/watch?v=0eC5jMV342A



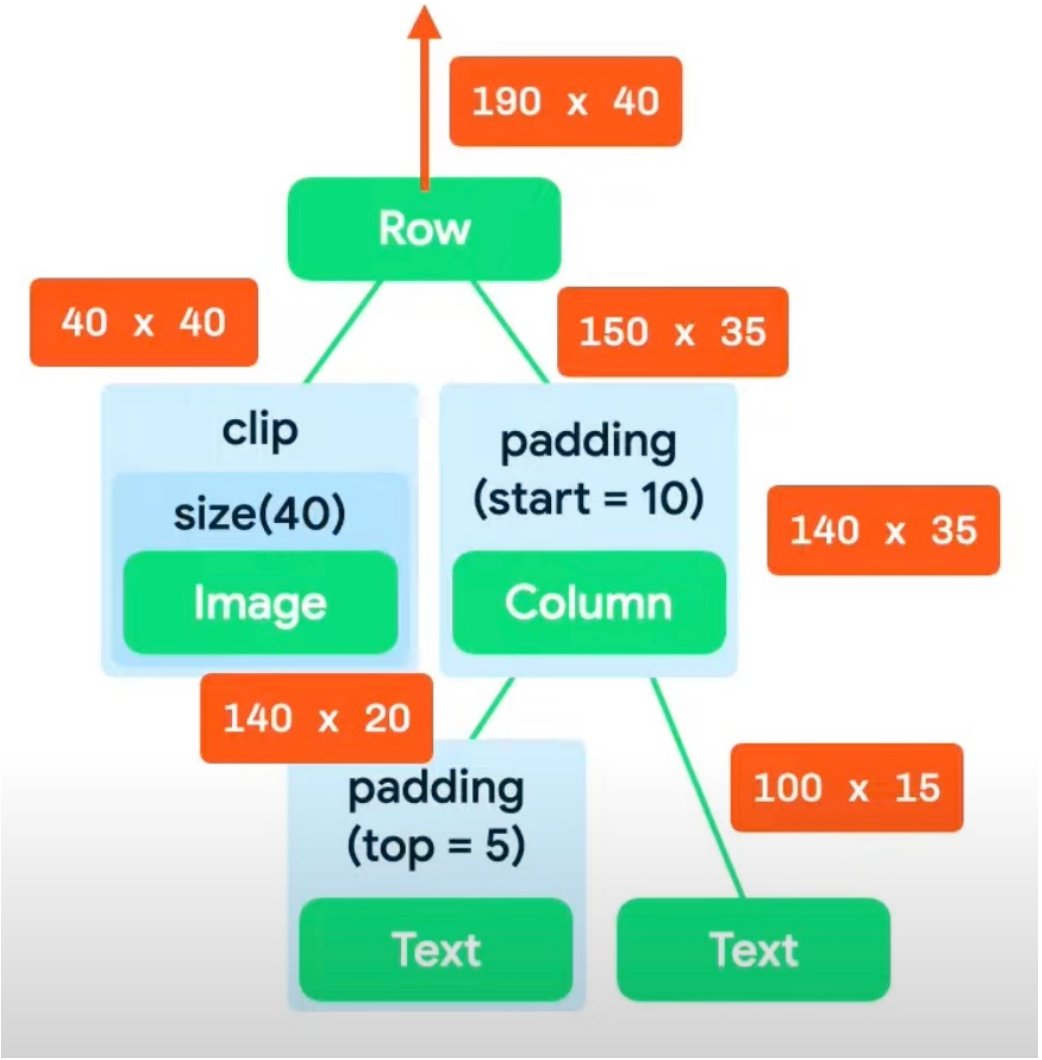


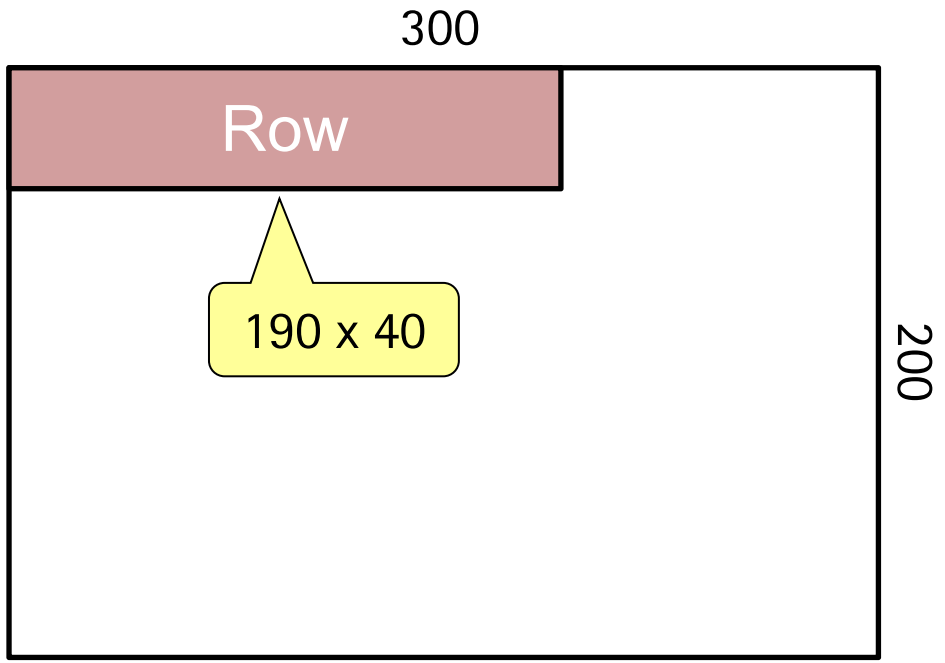
https://www.youtube.com/watch?v=0eC5jMV342A





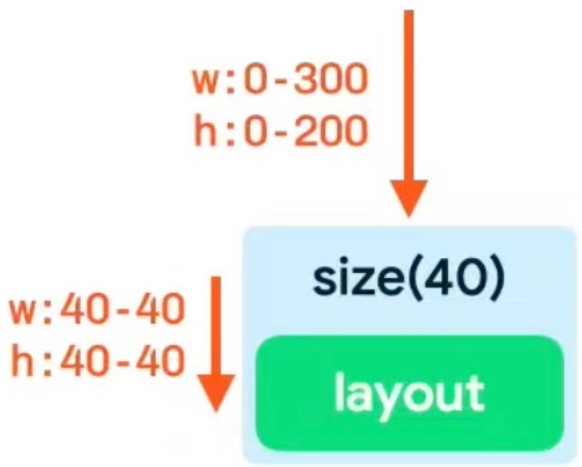
https://www.youtube.com/watch?v=0eC5jMV342A

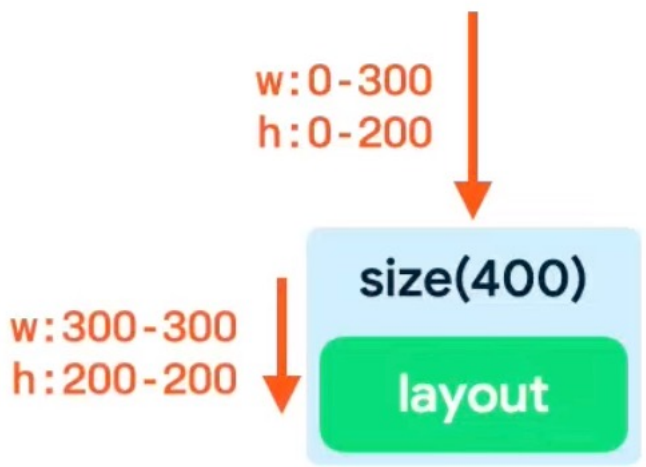


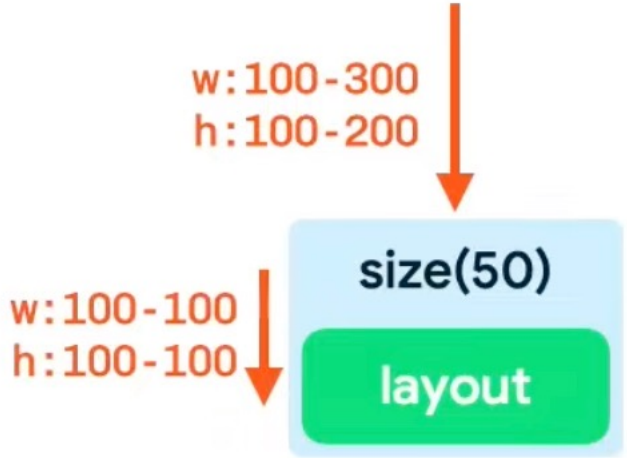


https://www.youtube.com/watch?v=OeC5jMV342A

Size Modifier

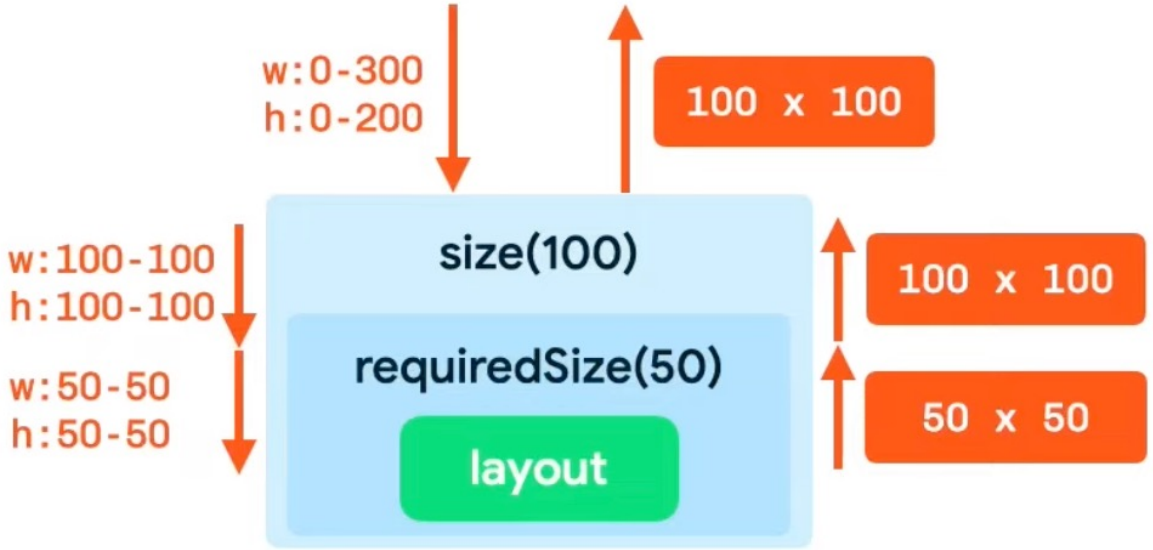


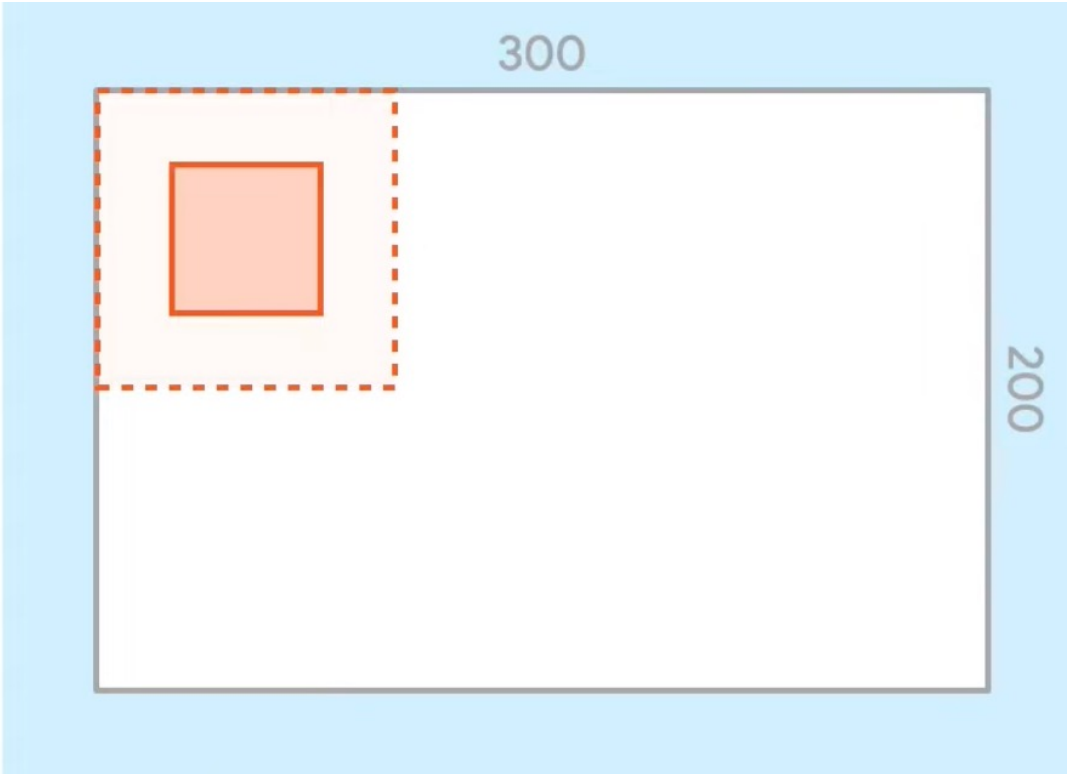




https://www.youtube.com/watch?v=0eC5jMV342A

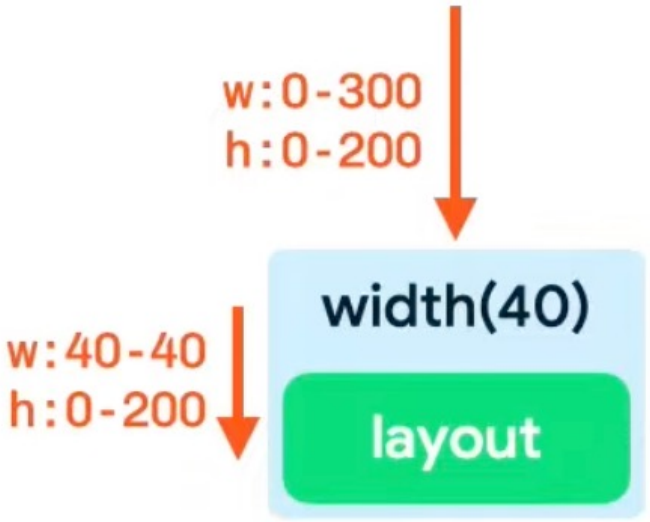
RequiredSize Modifier

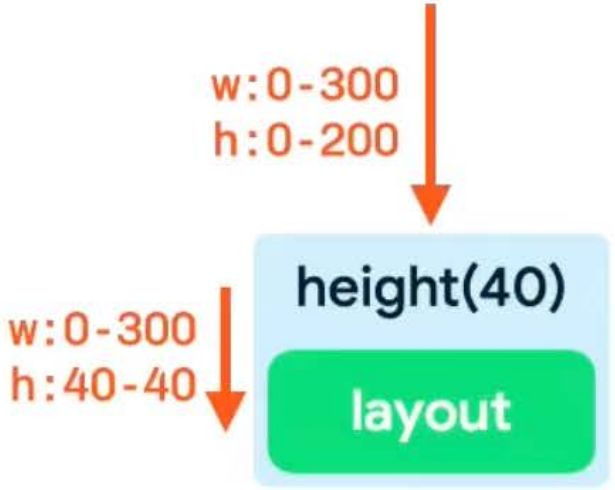


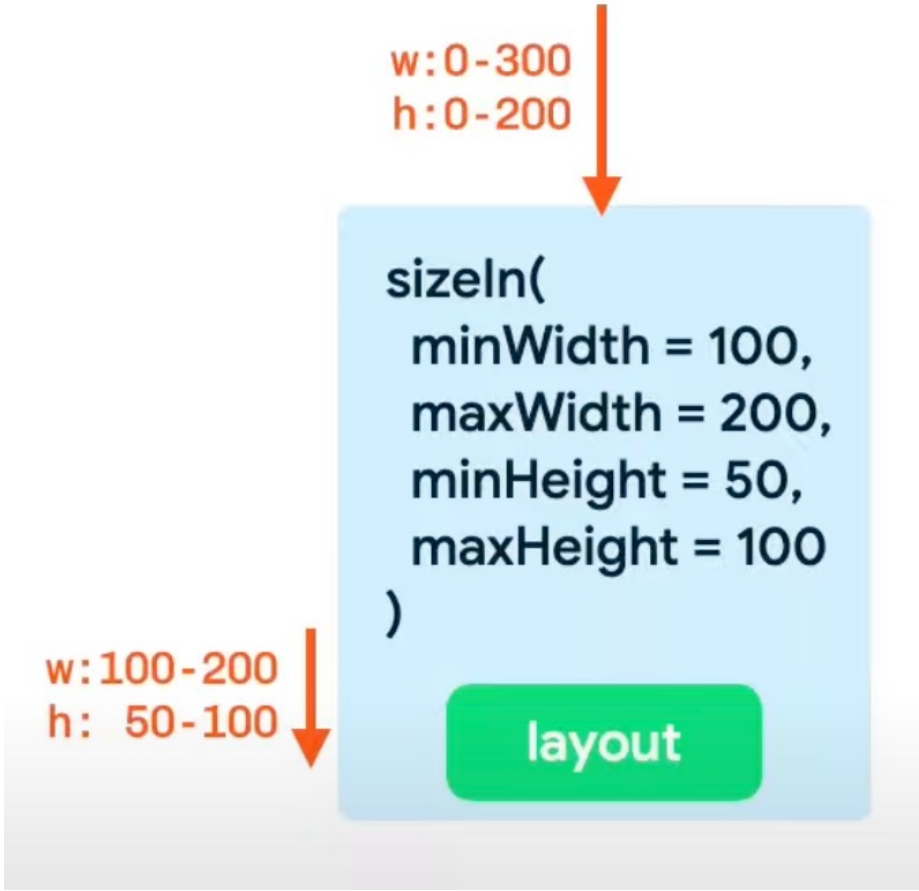


https://www.youtube.com/watch?v=0eC5jMV342A

Width, Height, and SizeIn Modifiers



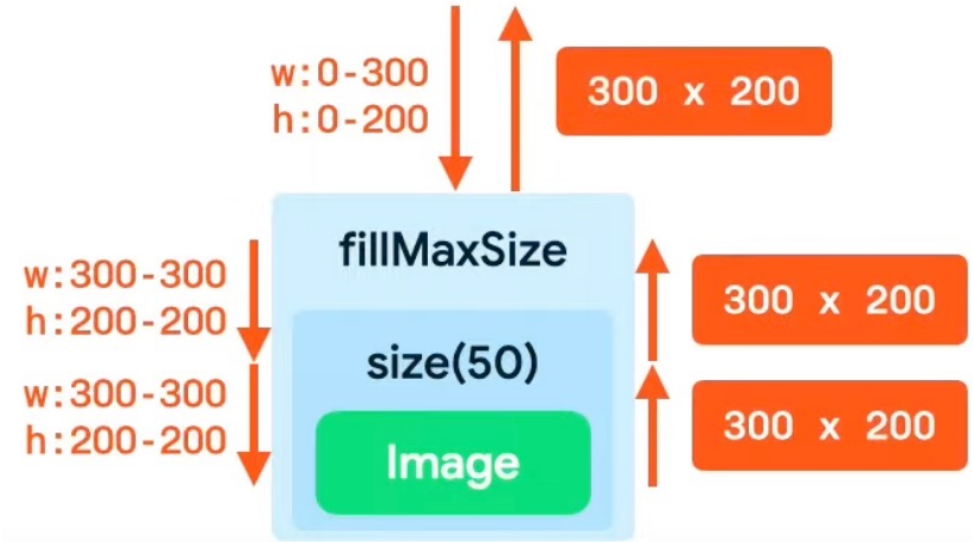


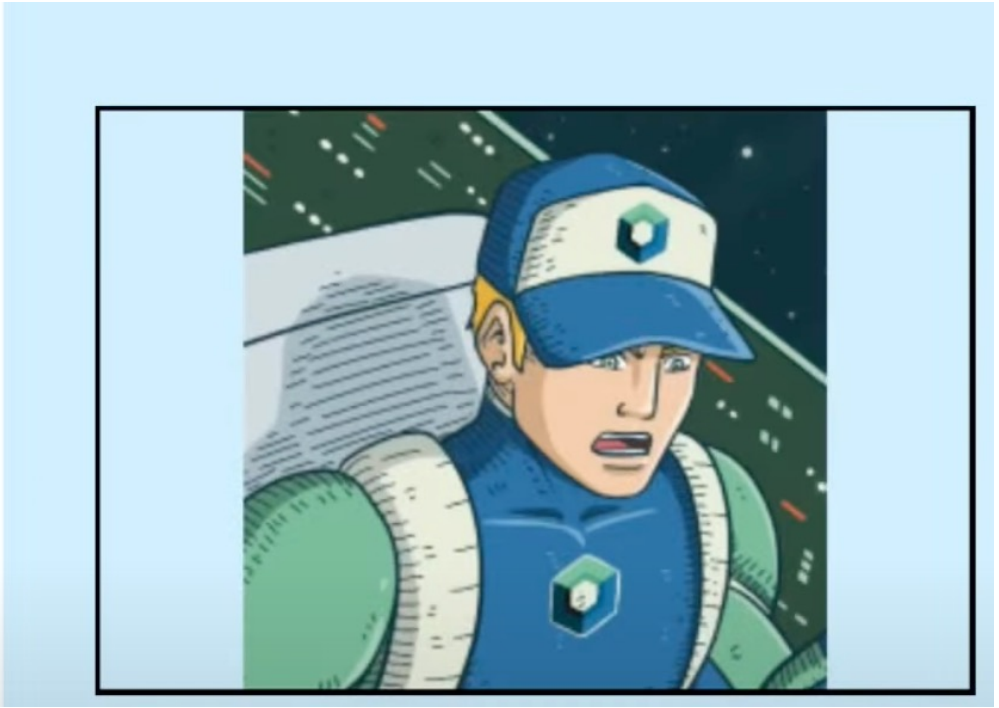


https://www.youtube.com/watch?v=0eC5jMV342A

FillMaxSize Modifier

```
Image(
    painterResource(R.drawable.frag),
    contentDescription = null,
    Modifier
        .fillMaxSize()
        .size(50.dp)
)
```



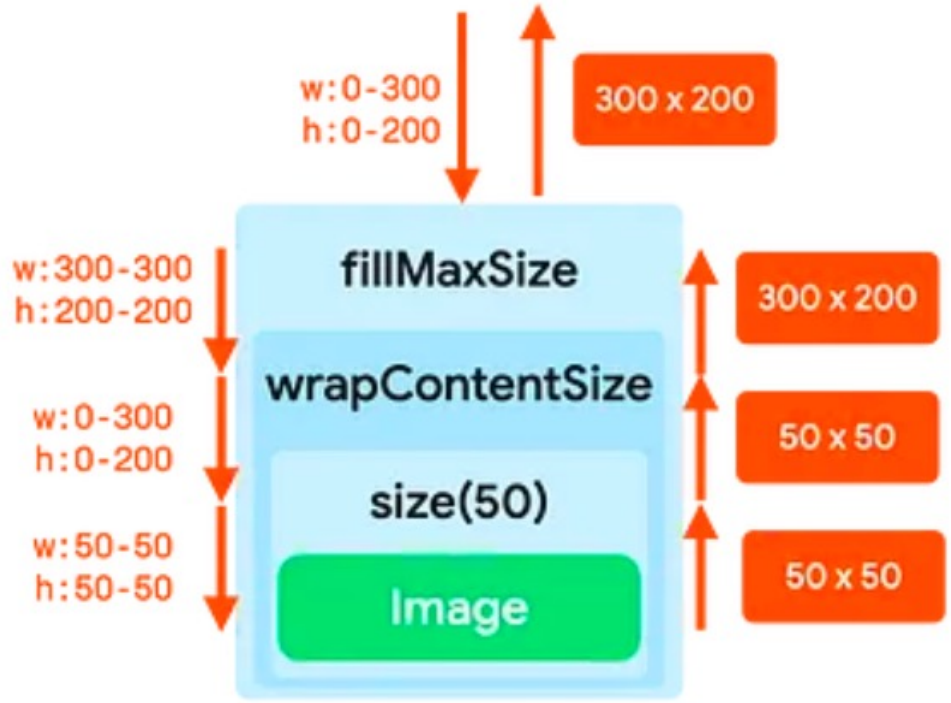


https://www.youtube.com/watch?v=0eC5jMV342A

WrapContentSize

- Place child in center of passed-in min bounds
- Returned size is passed-in min bounds

```
Image(
    painterResource(R.drawable.hero),
    contentDescription = null,
    Modifier
        .fillMaxSize()
        .wrapContentSize()
        .size(50.dp)
)
```



References

- https://developer.android.com/develop/ui/compose/layouts/constraints-modifiers
- https://www.youtube.com/watch?v=0eC5jMV342A