Jul 25, 2021

# 8. Image in JetPack Compose

Updated: Sep 30, 2021

Before you learn, you should know about what are the options available in this image function.
These are the options available in Image:

```
@Composable
fun Image(
    painter: Painter,
    contentDescription: String?,
    modifier: Modifier = Modifier,
    alignment: Alignment = Alignment.Center,
    contentScale: ContentScale = ContentScale.Fit,
    alpha: Float = DefaultAlpha,
    colorFilter: ColorFilter? = null
)
```

To create an image, you need following parameters
a). Painter - to load a drawable from resources you need to use painterResource.
you need to pass drawable resource id as a parameter in painterResource and It will return the painter.

```
fun painterResource(@DrawableRes id: Int): Painter
```

b) ContentDescription - You need to give description about the image. You can set it as null.

c) Modifier (Optional) - If you don't use the modifier, Image will take the original resource size as size.  So use modifier to set the fixed size and avoid design related issues.

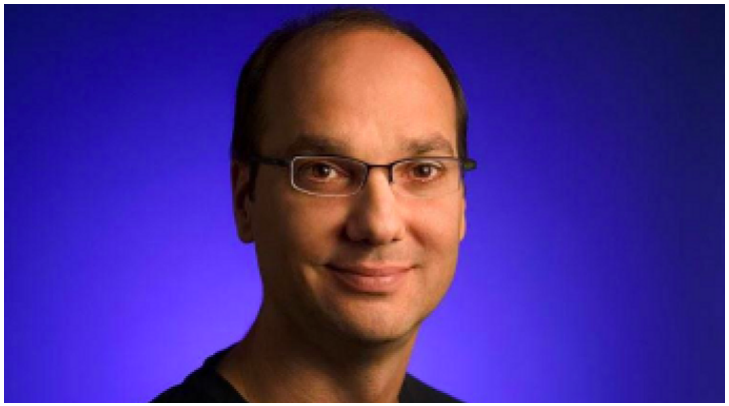
# 1. Simple Image

Sample code:

```
@Composable
fun SimpleImage() {
    Image(
        painter = painterResource(id = R.drawable.andy_rubin),
        contentDescription = "Andy Rubin",
        modifier = Modifier.fillMaxWidth()
    )
}
```

Here we set the drawable using painterResource. And we set the *fillMaxWidth()* modifier. It will fill the entire screen width size.

Output:



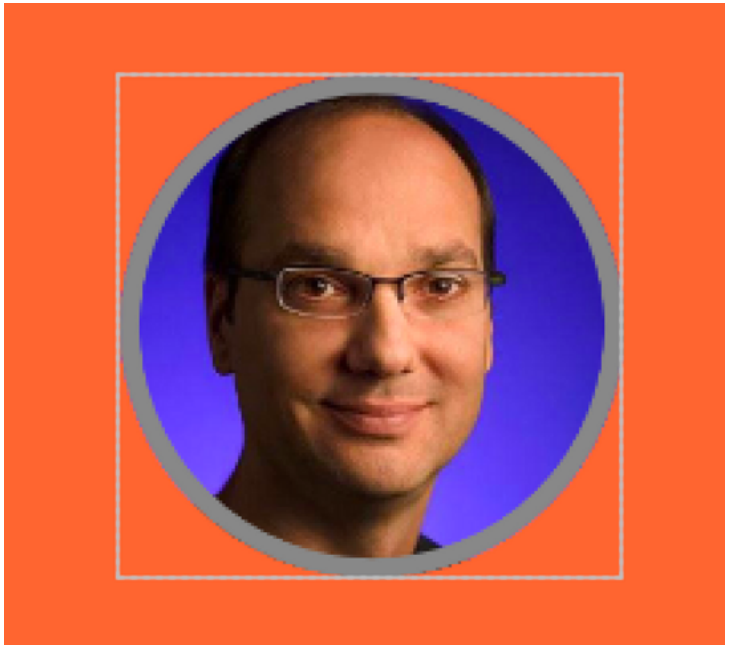
# 2. Circle Image

Sample code:

```
@Composable
fun CircleImageView() {
    Image(
        painter = painterResource(R.drawable.andy_rubin),
        contentDescription = "Circle Image",
        contentScale = ContentScale.Crop,
        modifier = Modifier
            .size(128.dp)
            .clip(CircleShape) // clip to the circle shape
            .border(5.dp, Color.Gray, CircleShape)//optional
    )
}
```

We clip the shape to CircleShape, so entire image will convert into circle shape.
We set border arround the image with gray color.
border - is optional. if you don't want you can ignore this.

Output:



## 3. Round Corner Image

Sample code:

```
@Composable
fun RoundCornerImageView() {
    Image(
        painter = painterResource(R.drawable.andy_rubin),
        contentDescription = "Round corner image",
        contentScale = ContentScale.Crop,
        modifier = Modifier
            .size(128.dp)
            .clip(RoundedCornerShape(10))
            .border(5.dp, Color.Gray, RoundedCornerShape(10))
```

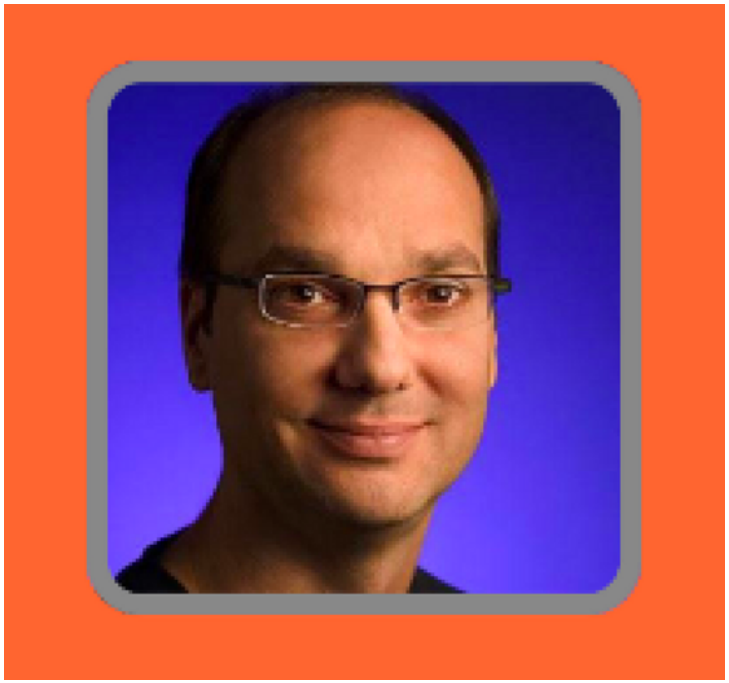
```
        )
    }
```

Its same like our previous Circle Image example. Only difference is we change the clip shape to *RoundedCornerShape()*.

*RoundedCornerShape() - you can set the percentage values or DP size.*

If you want to set percentage you need to pass integer value - ***RoundedCornerShape(20)***
If you want to set Dp pass the Dp value - ***RoundedCornerShape(50.dp)***

Output:



# 4. Image Background Color

Sample code:

```
@Composable
fun ImageWithBackgroundColor() {
    Image(
        painter = painterResource(id = R.drawable.ic_cart),
        contentDescription = "",
        modifier = Modifier
            .size( 200.dp)
            .background(Color.DarkGray)
            .padding(20.dp)
    )
}
```

In this example we set the one png icon as resource. If you use the jpeg/jpg you can't see the background color, because it have no transparent background. We use the **background()** modifier to set the background color for this image.

We set the DarkGray color, so it display icon with DarkGray background.

Output:

## 5. Image ColorFilter (tint)

Sample code:

```
@Composable
fun ImageWithTint() {
    Image(
        painter = painterResource(id = R.drawable.ic_cart),
        contentDescription = "",
        colorFilter = ColorFilter.tint(Color.Red),
        modifier = Modifier
            .size( 200.dp)
```

```
        )
    }
```

With help of tint() we can change the image resource color. In our previous example we use the cart icon it displays the white color (original color). In this example we change the color to Red. You can set any color using color filter.

```
    colorFilter = ColorFilter.tint(Color.Red)
```

## 6. ContentScale

Following ContentScale types available in Compose.

- FillBounds
- FillHeight

FillWidth

Inside

Fit

Crop

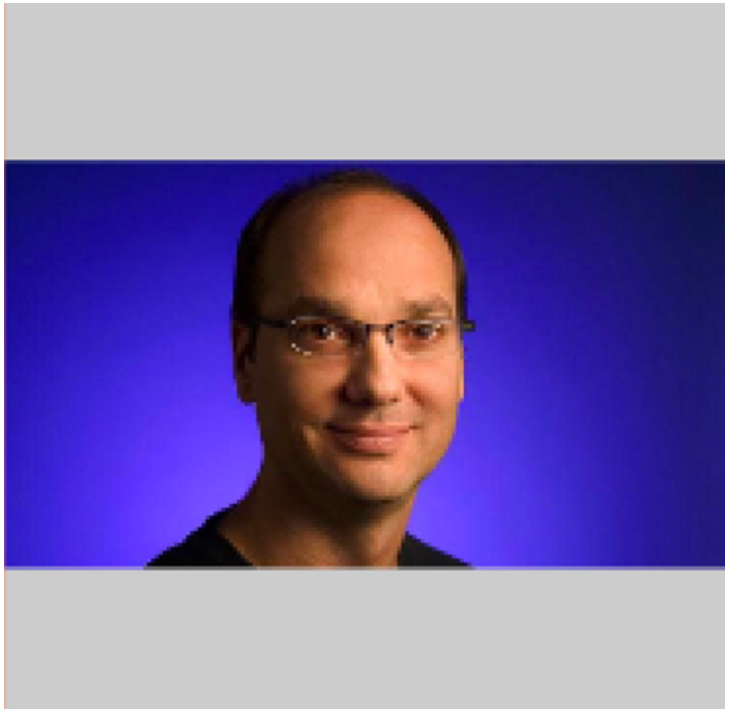
In traditional android programming we call it is as scaleType.
You can compare both with following table.

| XML Image | | Jetpack Compose Image | |
|---|---|---|---|
| **ScaleType** | **Note** | **ContentScale** | **Alignment** |
| Matrix | Default with no value | None | Top Start |
| Center | | None | Center |
| Center Inside | | Inside | Center |
| Center Crop | | Crop | Center |
| Fit Center | | Fit | Center |
| Fit Start | | Fit | Top Start |
| Fit End | | Fit | Bottom End |
| Fit XY | | FillBounce | <Not Applicable> |

Sample code:

```
@Composable
fun InsideFit() {
    Image(
        painter = painterResource(id = R.drawable.andy_rubin),
        contentDescription = "",
        modifier = Modifier
            .size(150.dp)
            .background(Color.LightGray),
        contentScale = ContentScale.Inside
    )
}
```

Output:



Source code:code:

https://github.com/JetpackCompose/Jetpack-Compose-Text/blob/master/Compose%20Text/app/src/main/java/net/jetpackcompose/composetext/ImageSamplesActivity.kt