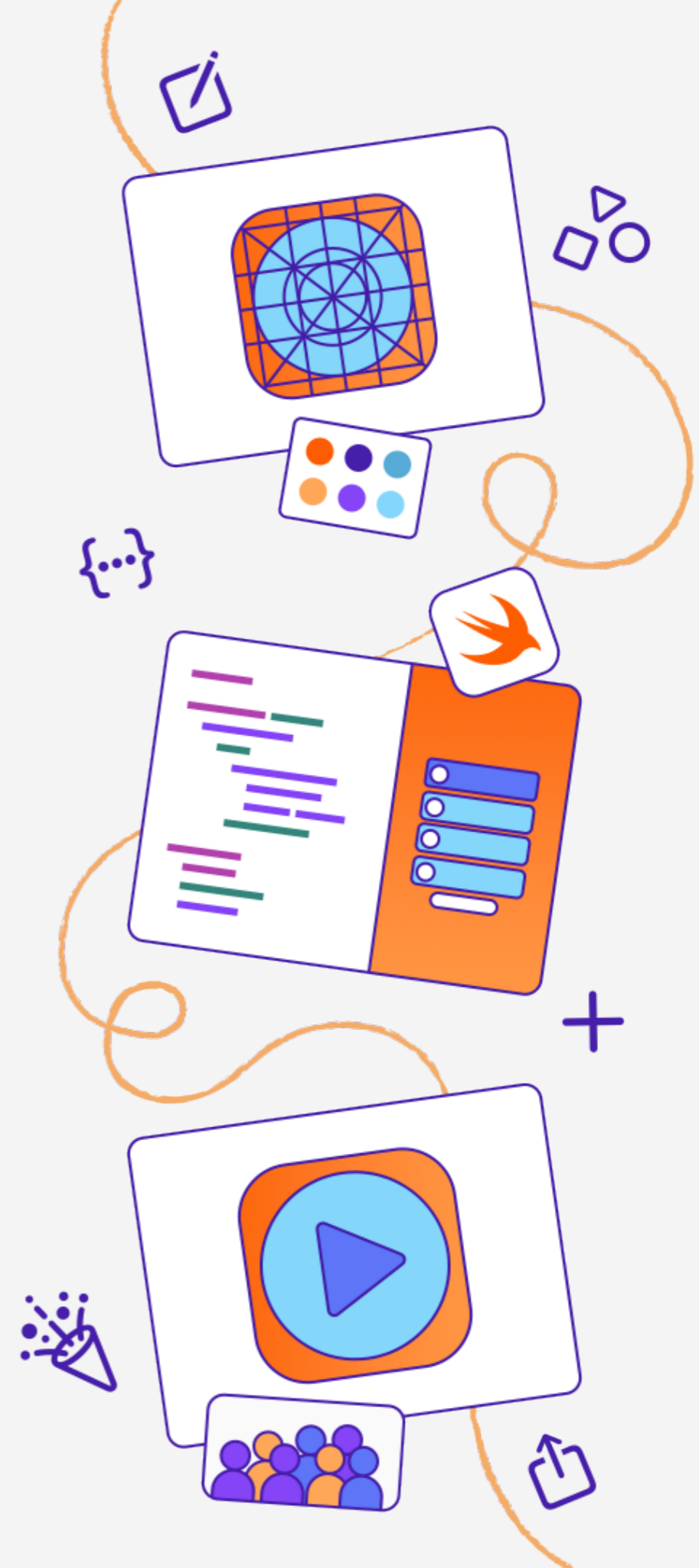


 Everyone Can Code

Project Presentation

This Everyone Can Code presentation is designed for use in educational settings. It can be customised, duplicated or shared for the purpose of the content, curriculum or age level of the learners for educational purposes. The presentation can also be used as a template for educator-made tutorials. The images, video or audio in this presentation should not be duplicated or shared for commercial purposes or used in a way in which it was not originally intended.



Build with Stacks and Shapes




Everyone Can Code

This instructional presentation showcases iPadOS 16.3, Keynote for iPad version 13.0 and Swift Playgrounds for iPad version 4.3. User interface elements may vary depending on the software version installed on your device.

Build with Stacks and Shapes

Learn SwiftUI fundamentals, like stacks, shapes and modifiers, to code a self portrait.

 [Project Example >](#)

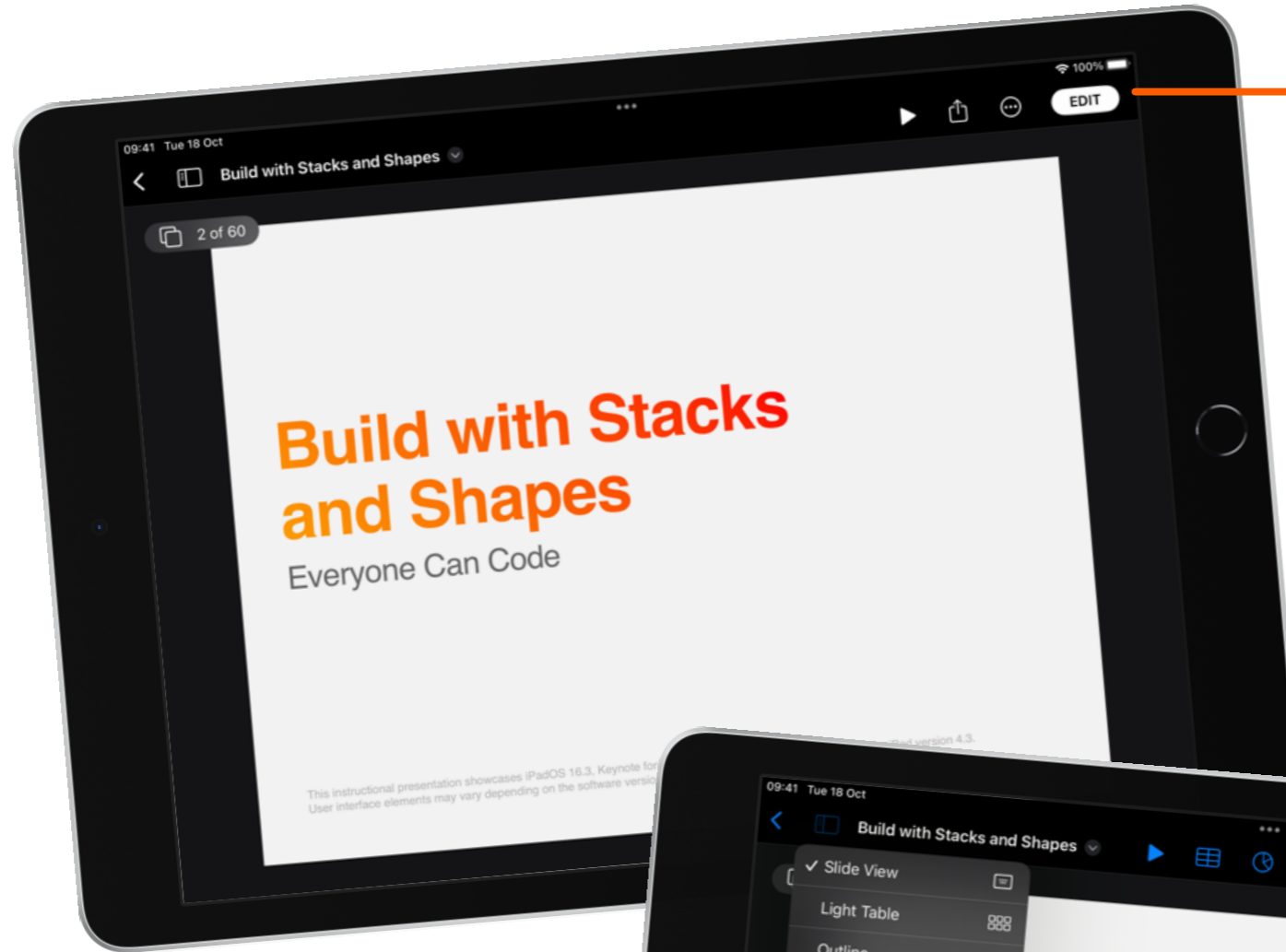
-  [Laying Out Views >](#) 20 mins
-  [Set Up an App Playground >](#) 5 mins
-  [Compose with Stacks and Shapes >](#) 35 mins

Estimated time: 1 hr





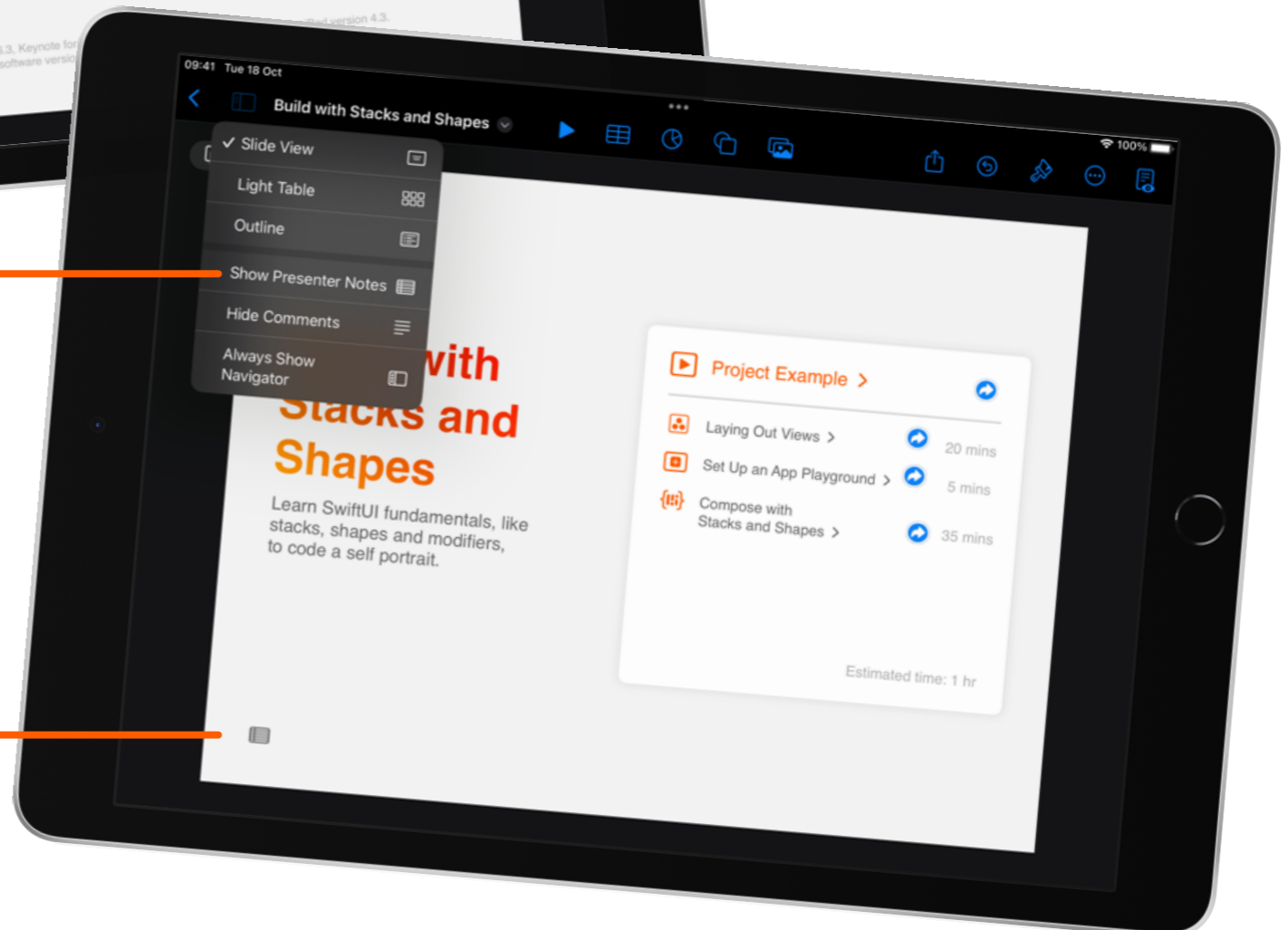
Swipe right from the left edge of the screen to show the slide navigator. Swipe left over the slide navigator to hide.



EDIT

Allows you to customise the current Keynote.

Show and hide presenter notes.



The presenter notes icon at the bottom of a screen indicates there is extra information in Presenter Notes.

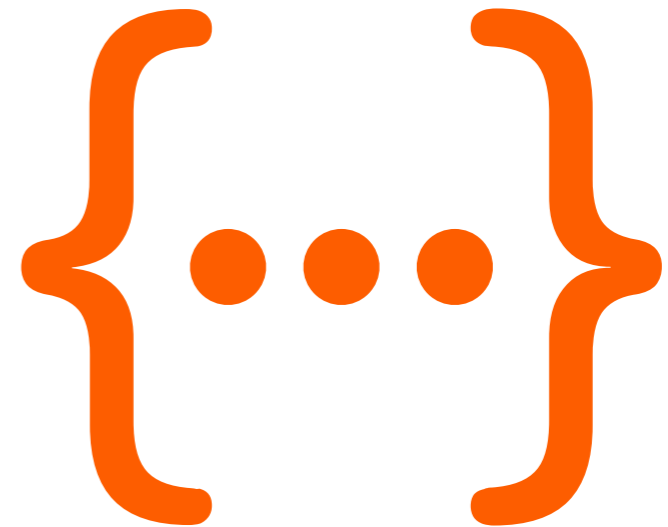


```
1 import SwiftUI
2
3 struct Example1: View {
4     var body: some View {
5         ZStack {
6             Color.brown.opacity(0.4)
7             Ellipse()
8                 .frame(width: 60, height: 220)
9                 .offset(x: 120, y: -100)
10            Ellipse()
11                .frame(width: 60, height: 230)
12                .offset(x: -120, y: -120)
13            RoundedRectangle(cornerRadius: 35)
14                .frame(width: 300, height:
15                    140)
16                .offset(x: 0, y: -170)
17
18            Rectangle()
19                .frame(width: 100, height:
20                    200)
21                .offset(y: 150)
22                .foregroundColor(Color(hue:
```



What you'll learn:

- ✓ Learn about views and how to lay them out on a screen
- ✓ Build up a view from a blank app playground
- ✓ Practise using stacks, shapes and modifiers



What you'll need:



Keynote for iPad

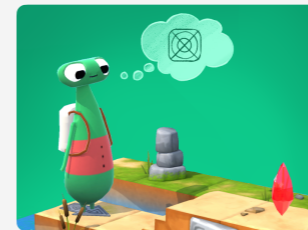


Swift Playgrounds for iPad

Suggested prerequisites:



[Get Started with Code >](#)

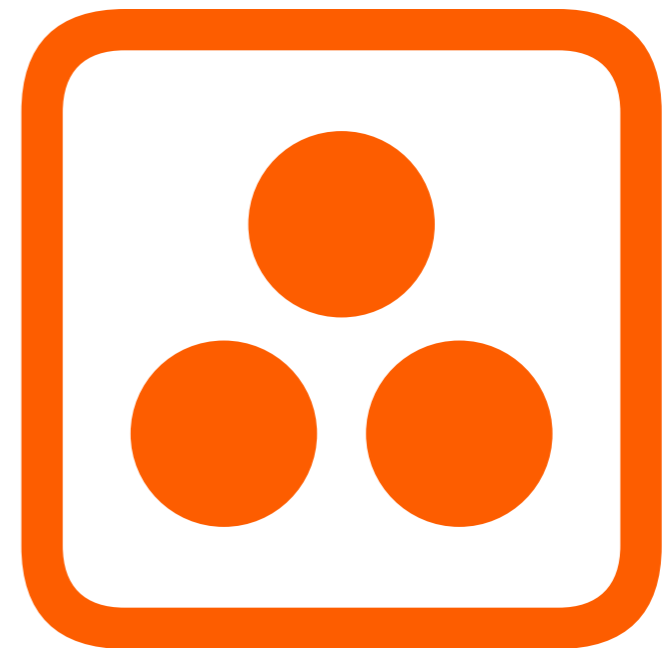


[Get Started with Apps >](#)

Tutorial 1

Laying Out Views

Learn how to control the sizing and placement of shapes in Swift Playgrounds.



 Estimated time:
20 mins

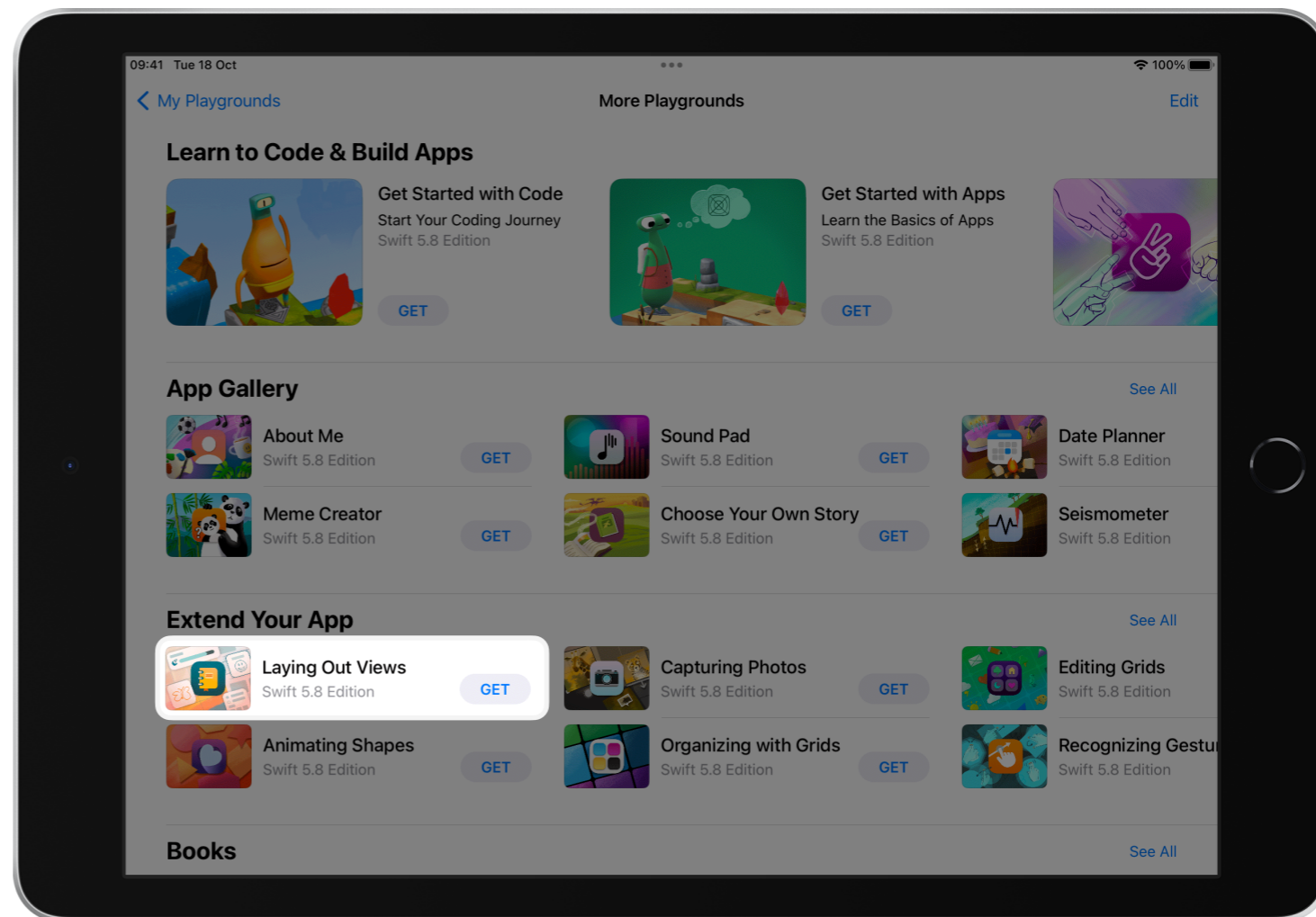
Step 1

Open Swift Playgrounds.



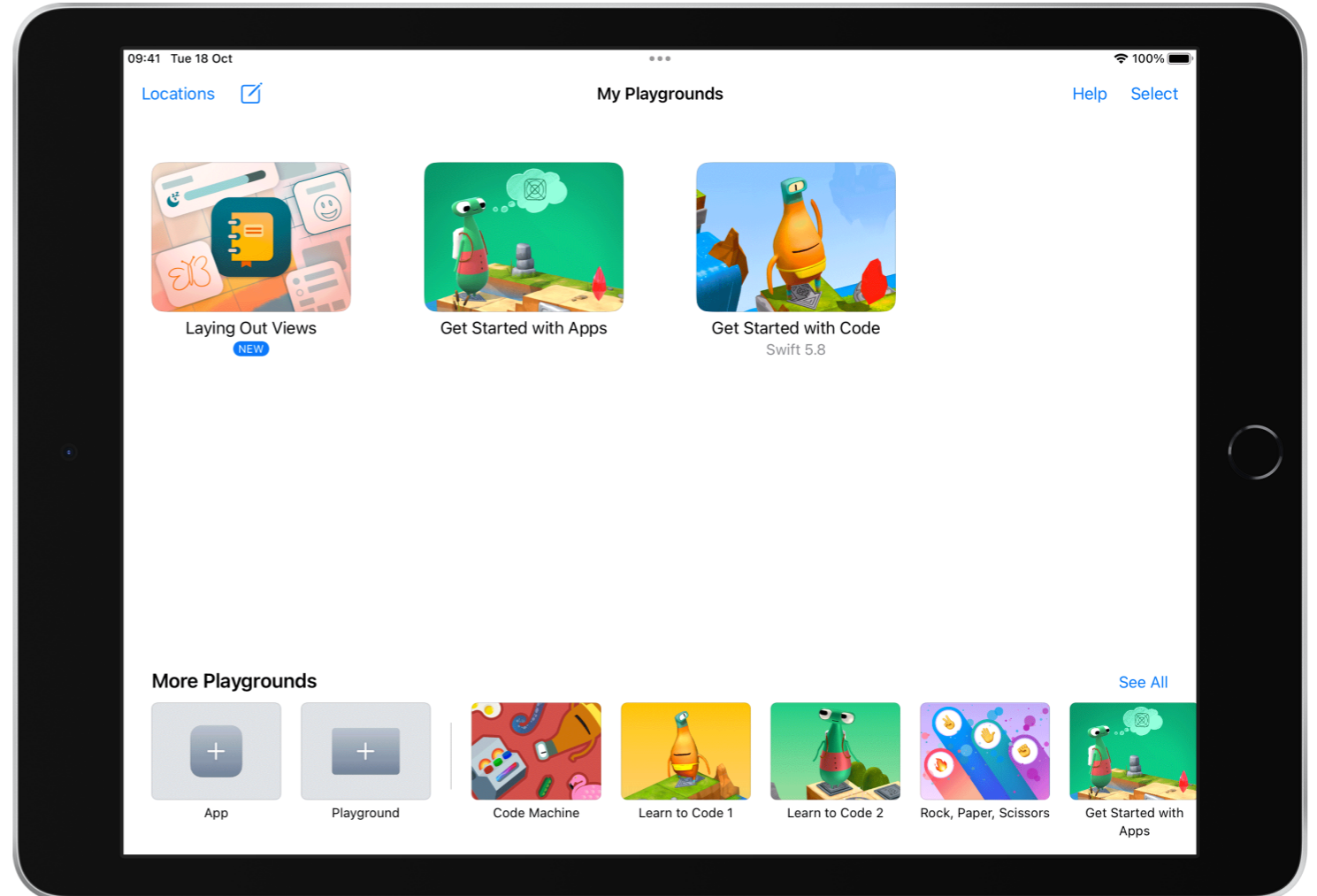
Step 2

Get Laying Out Views >



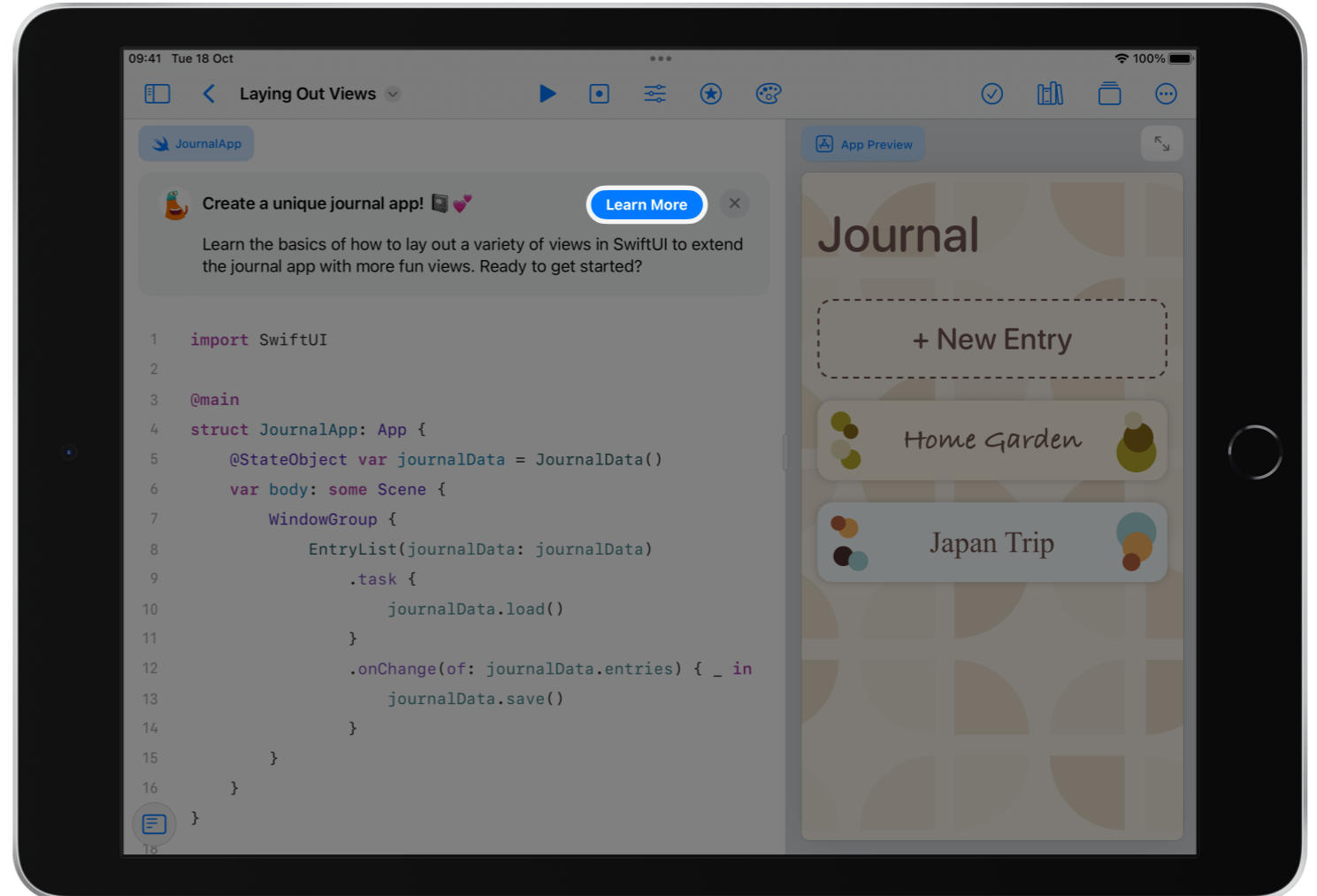
Step 3

Tap Laying Out Views to open it.



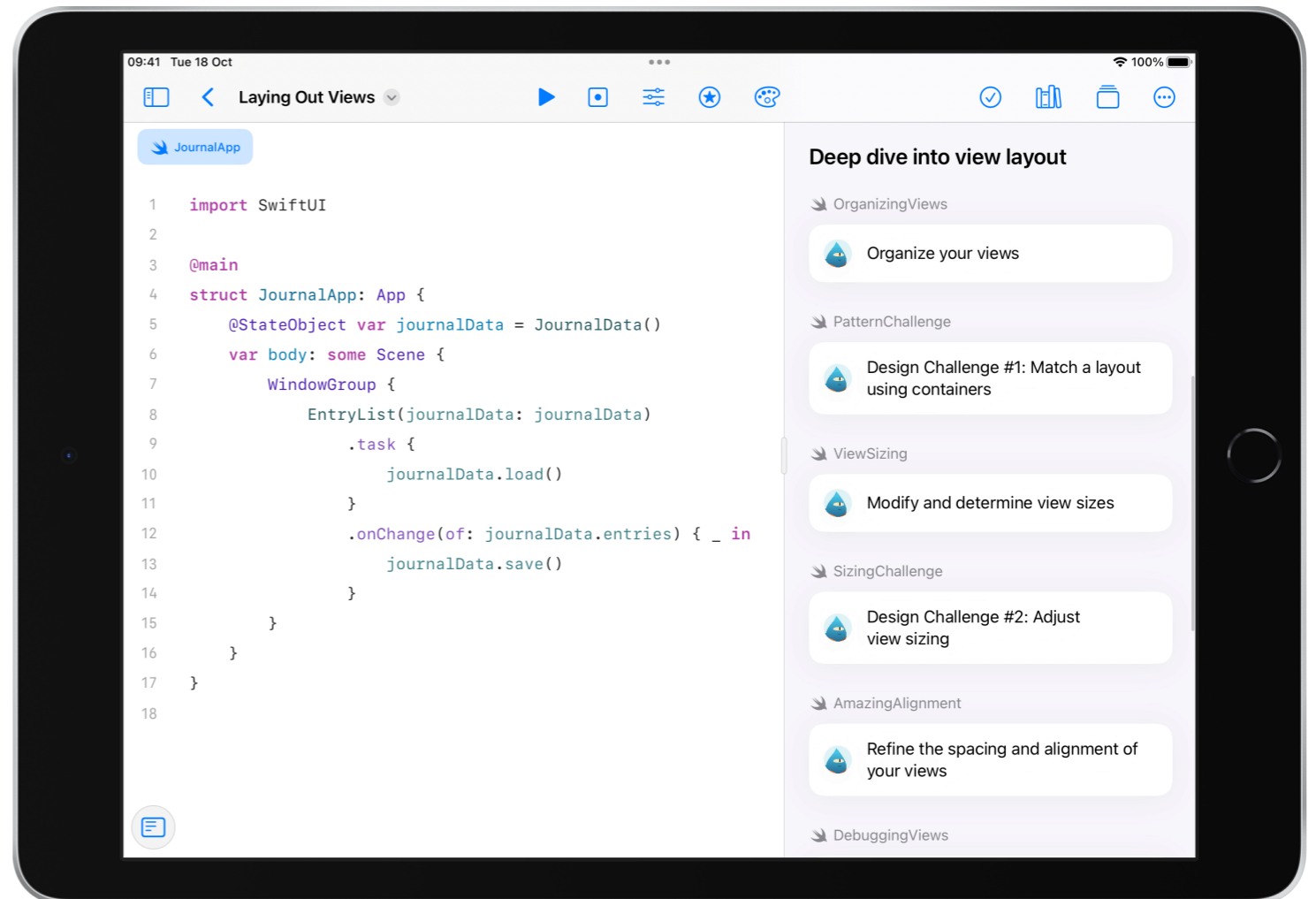
Step 4

Tap Learn More to open the guide.



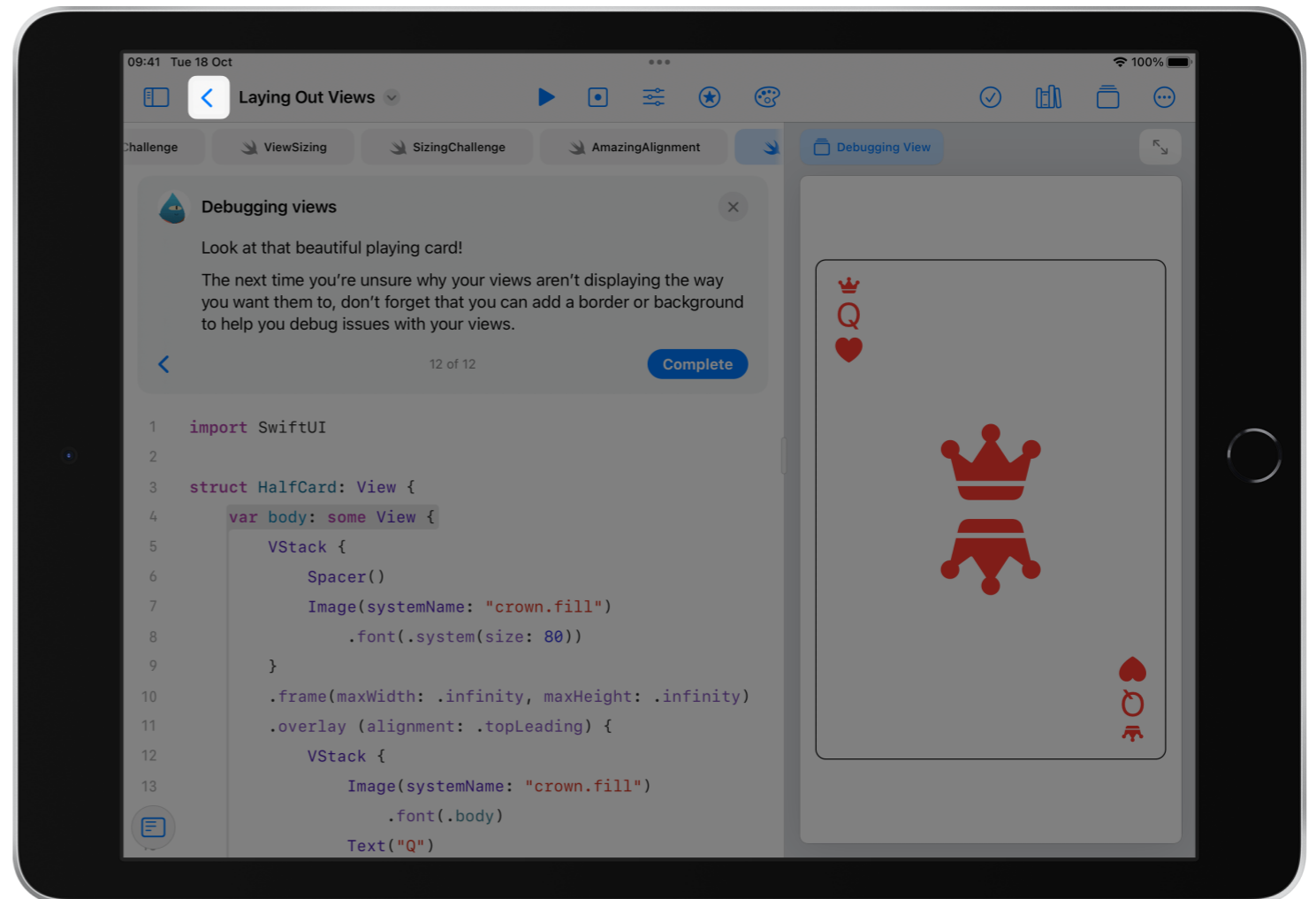
Step 5

Scroll down to find
Deep dive into view layout.
Complete all experiments
in this section.



Step 6

When you've finished, tap the Back button < in the toolbar to close Laying Out Views.




Tutorial 2

Set Up an App Playground

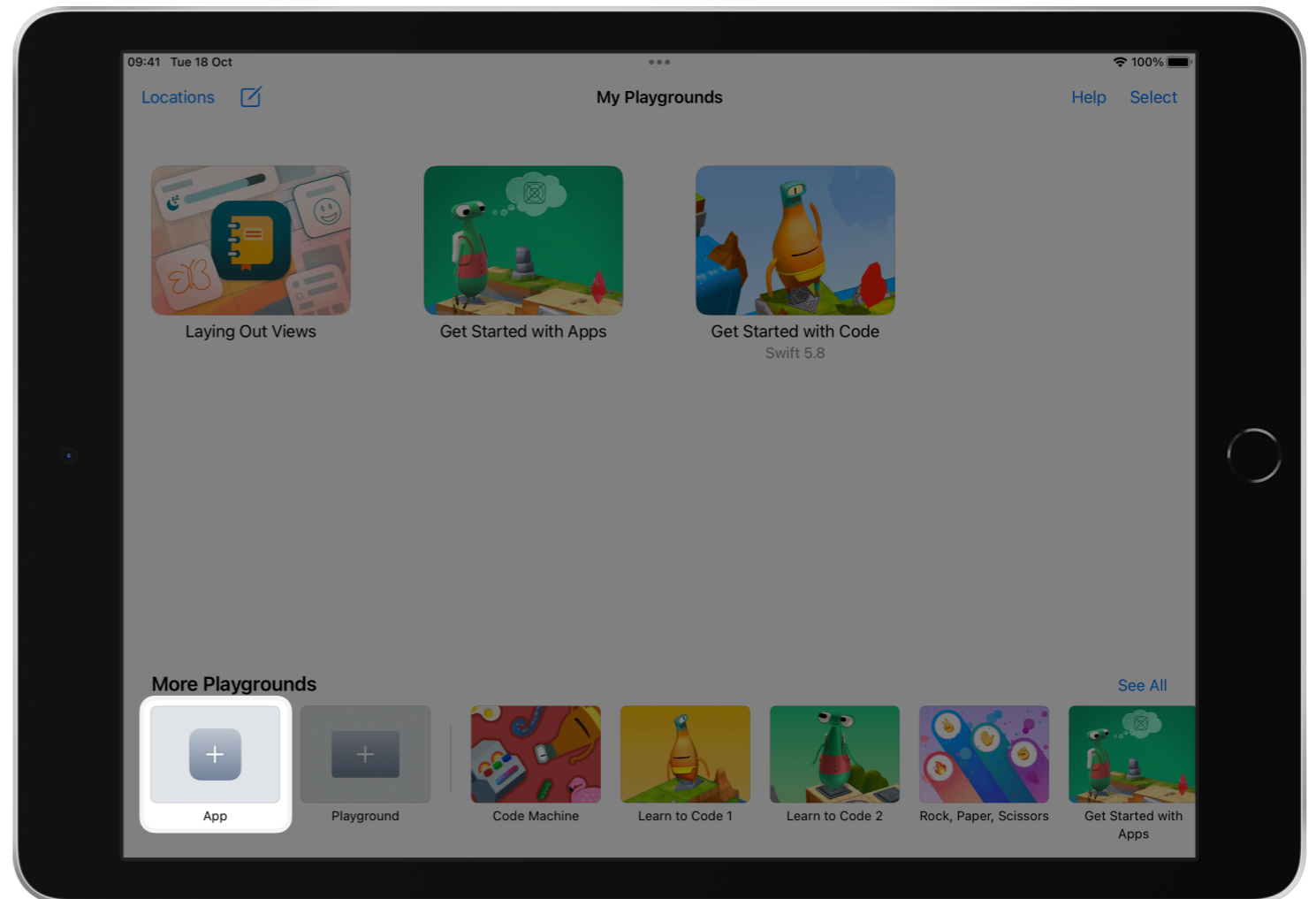
Get, rename and prepare a blank app playground.



 Estimated time:
5 mins

Step 1

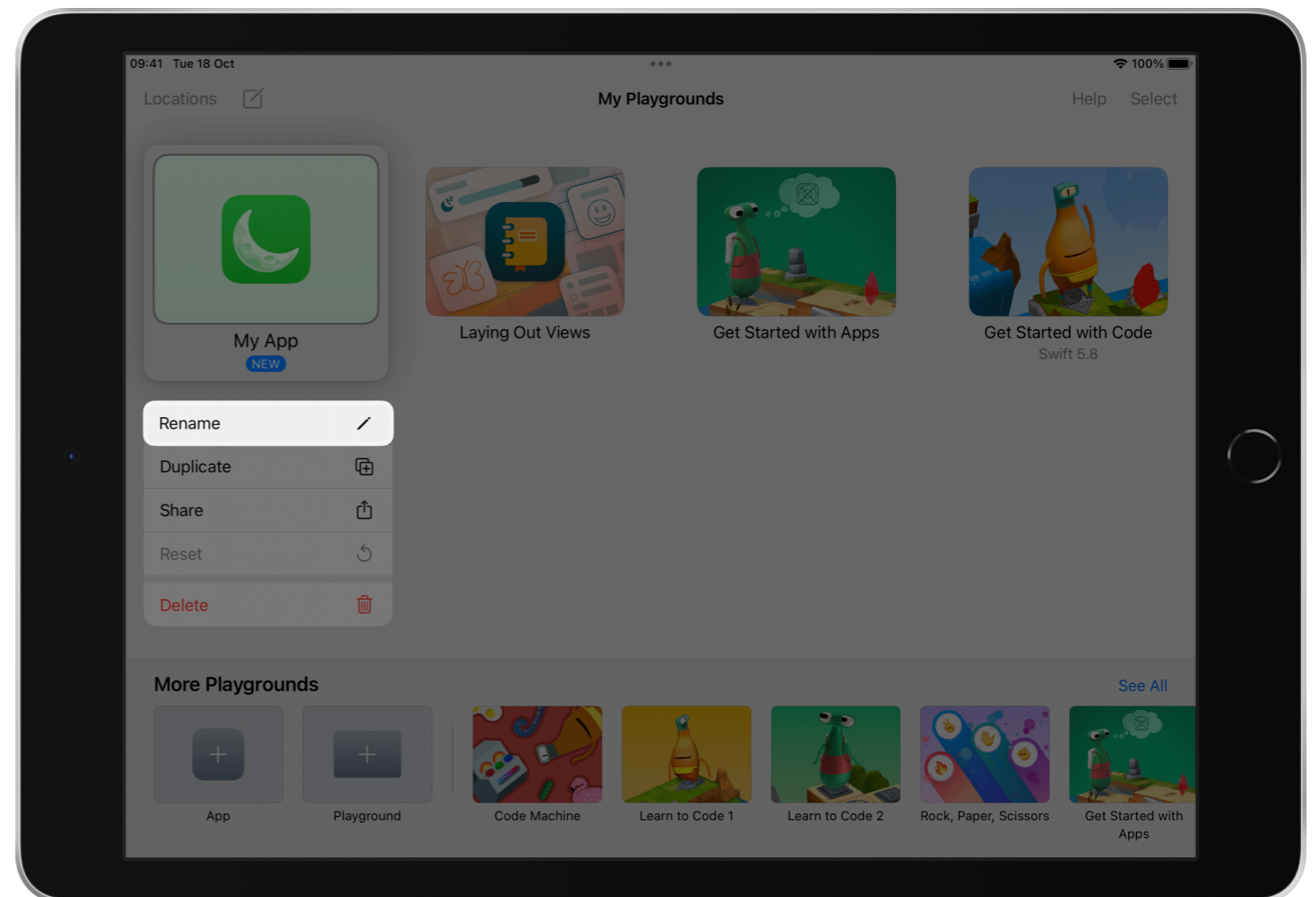
Tap App to start a new app playground.



Step 2

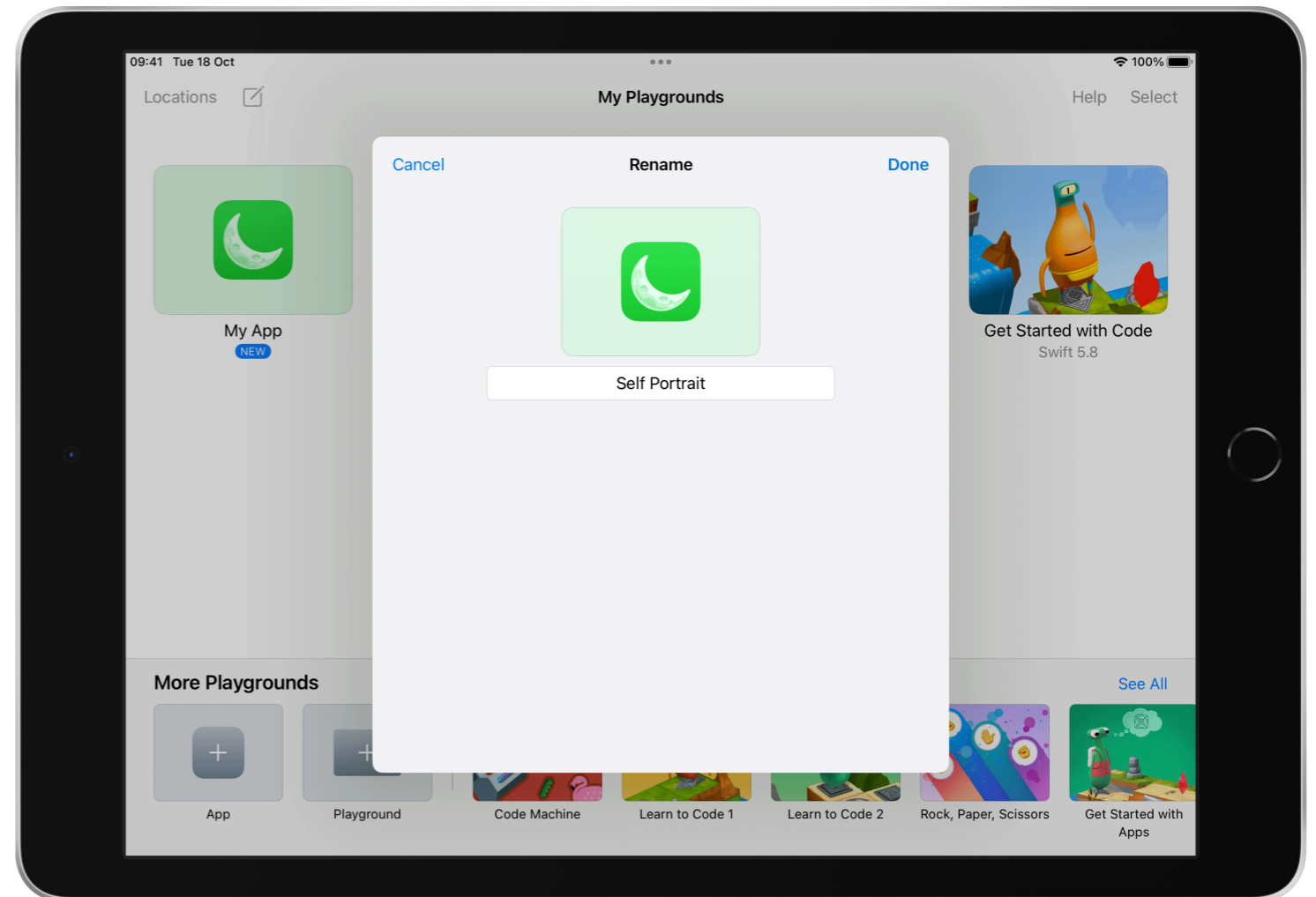
Tap and hold on the app playground until the menu appears.

Tap Rename.



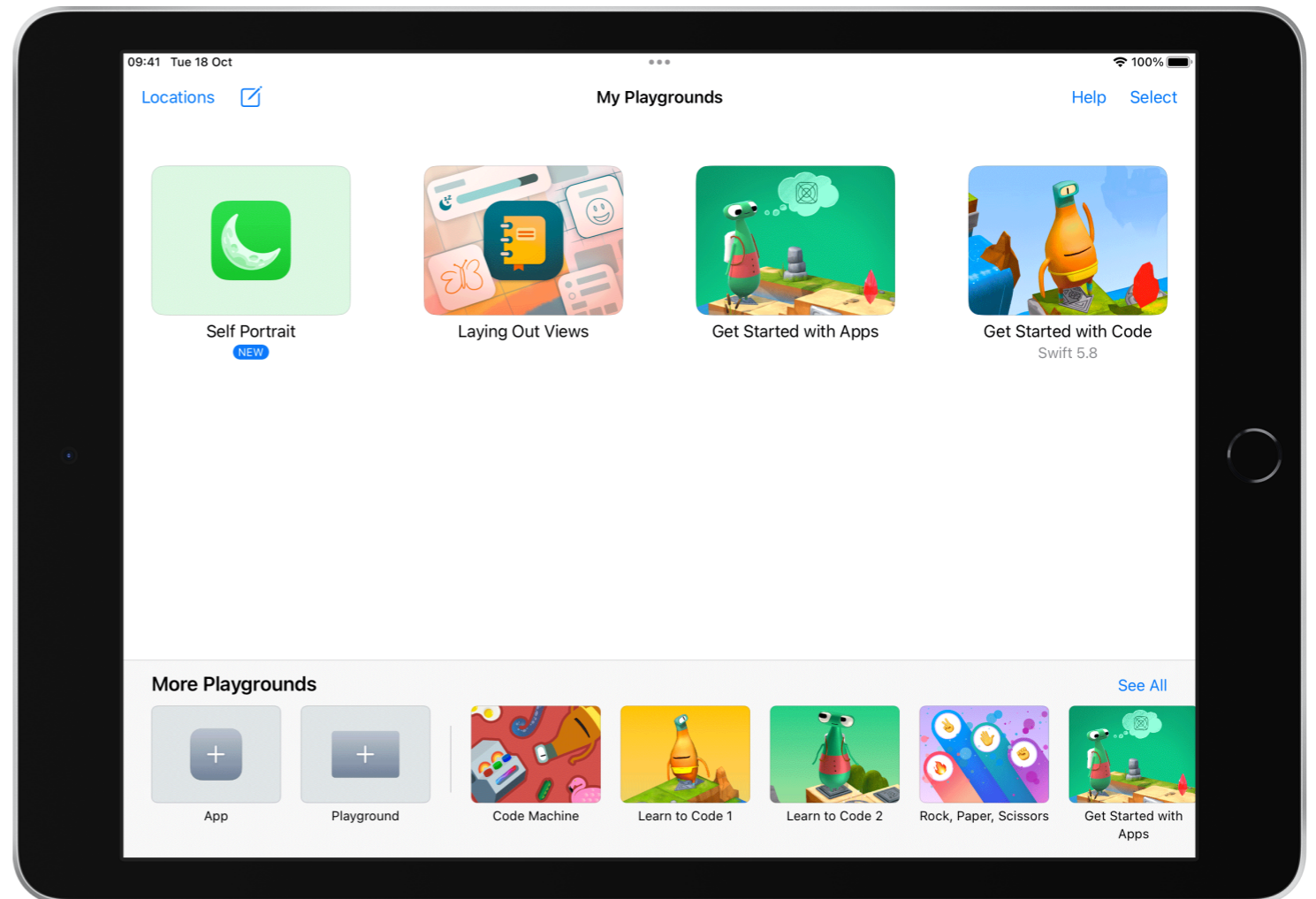
Step 3

Give your project a descriptive name, like “Self Portrait”, and tap Done.



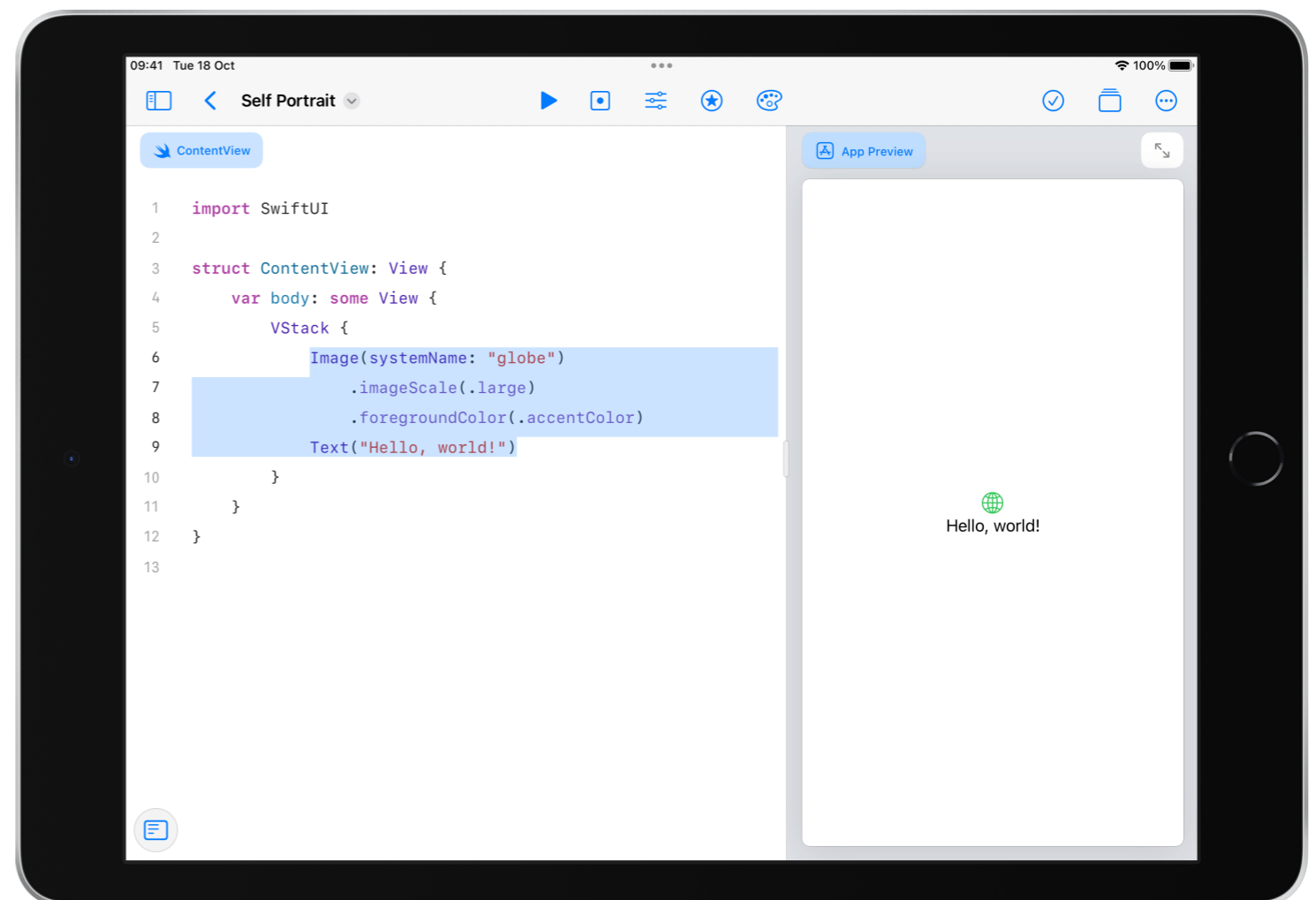
Step 4

Tap your renamed app playground to open.



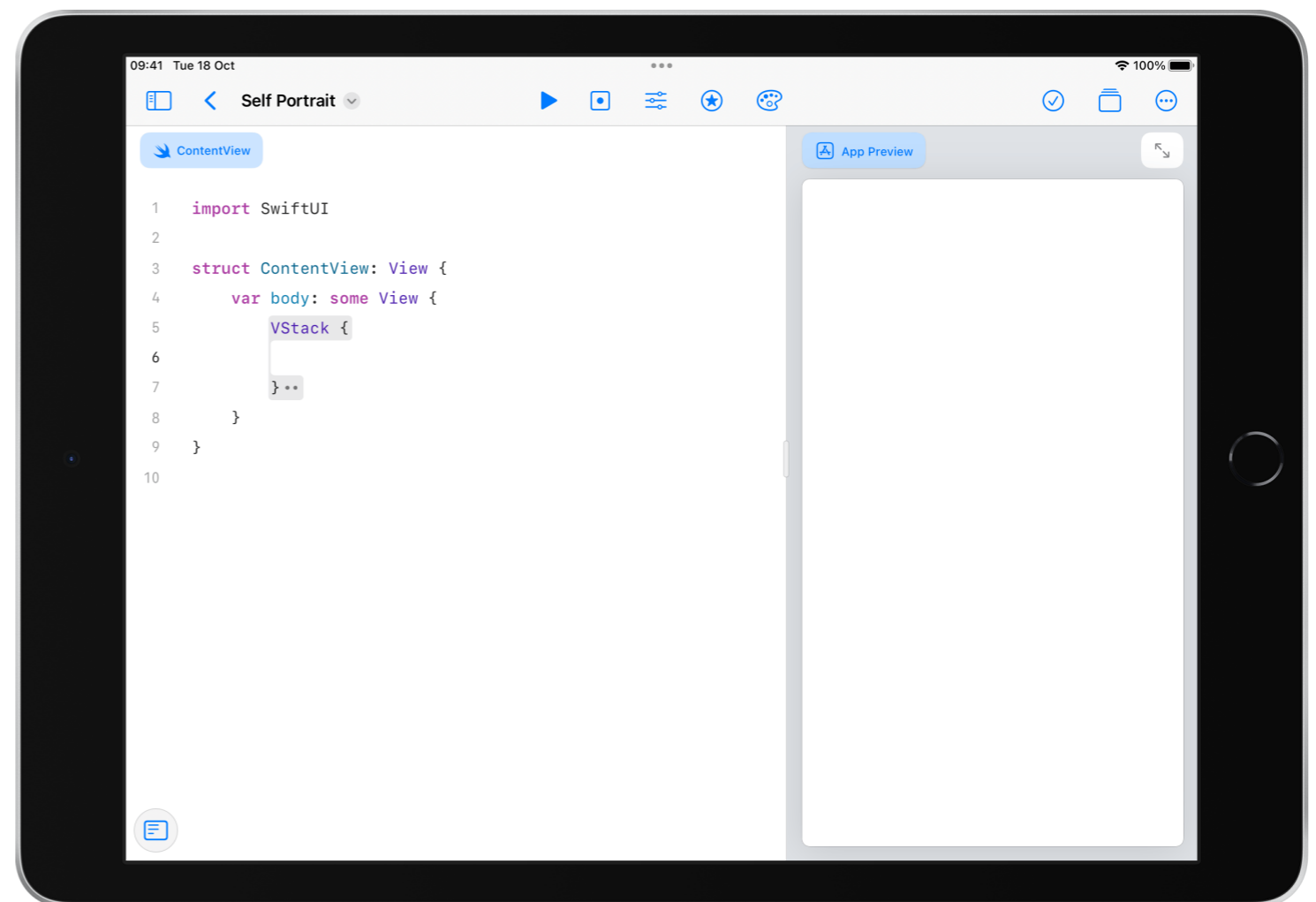
Step 5

Delete the image and text in the **VStack** (vertical stack).



Step 6

Now you're ready to start your self portrait.



Tutorial 3

Compose with Stacks and Shapes

Code a self portrait using stacks, shapes
and modifiers.










 Estimated time:
35 mins

Section 1

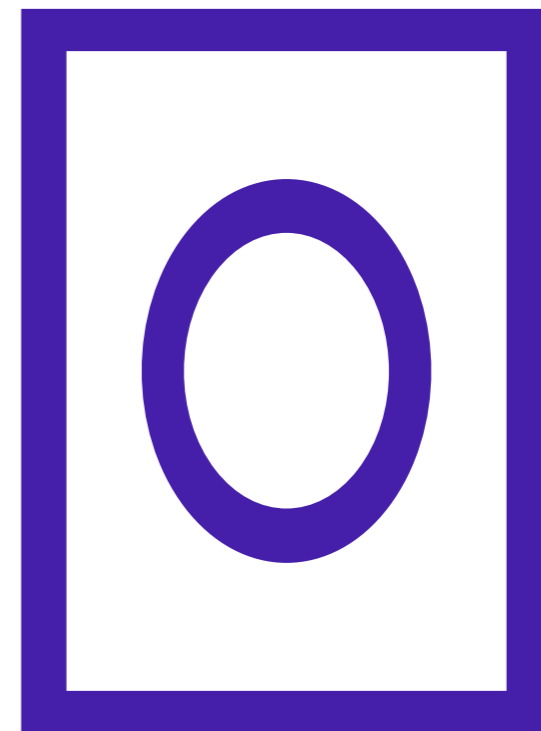
Explore SwiftUI tools

Use the following tools to
make a self portrait.

-  Use stacks and shapes >
-  Adjust the size of a shape >
-  Add colour >
-  Shift a view >
-  Trim a shape >
-  Add a shadow >
-  Add a background colour >

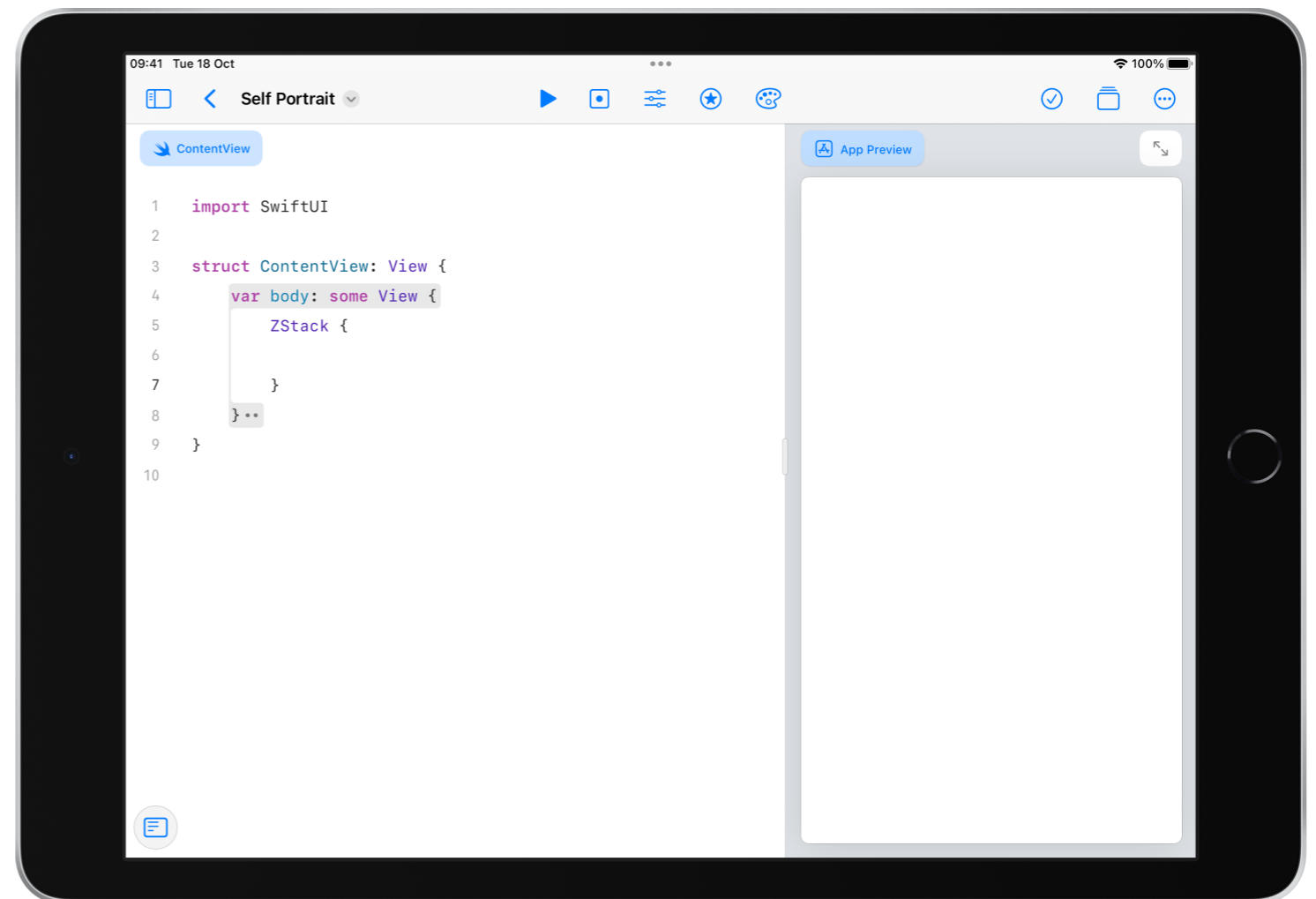
Use stacks and shapes

Use a depth stack and add a shape for your head.




Step 1

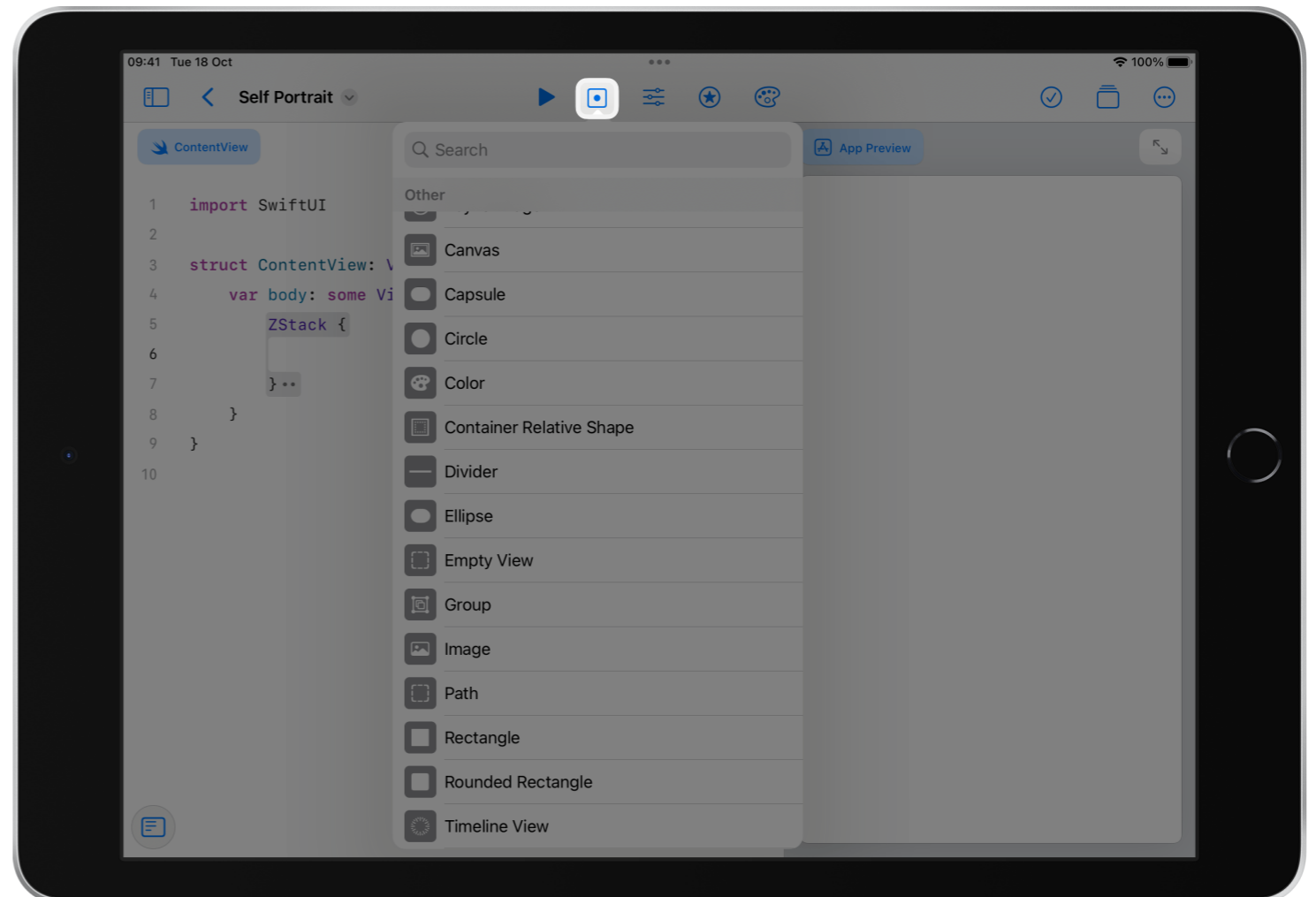
Change the VStack to a **ZStack (depth stack)**.



Step 2

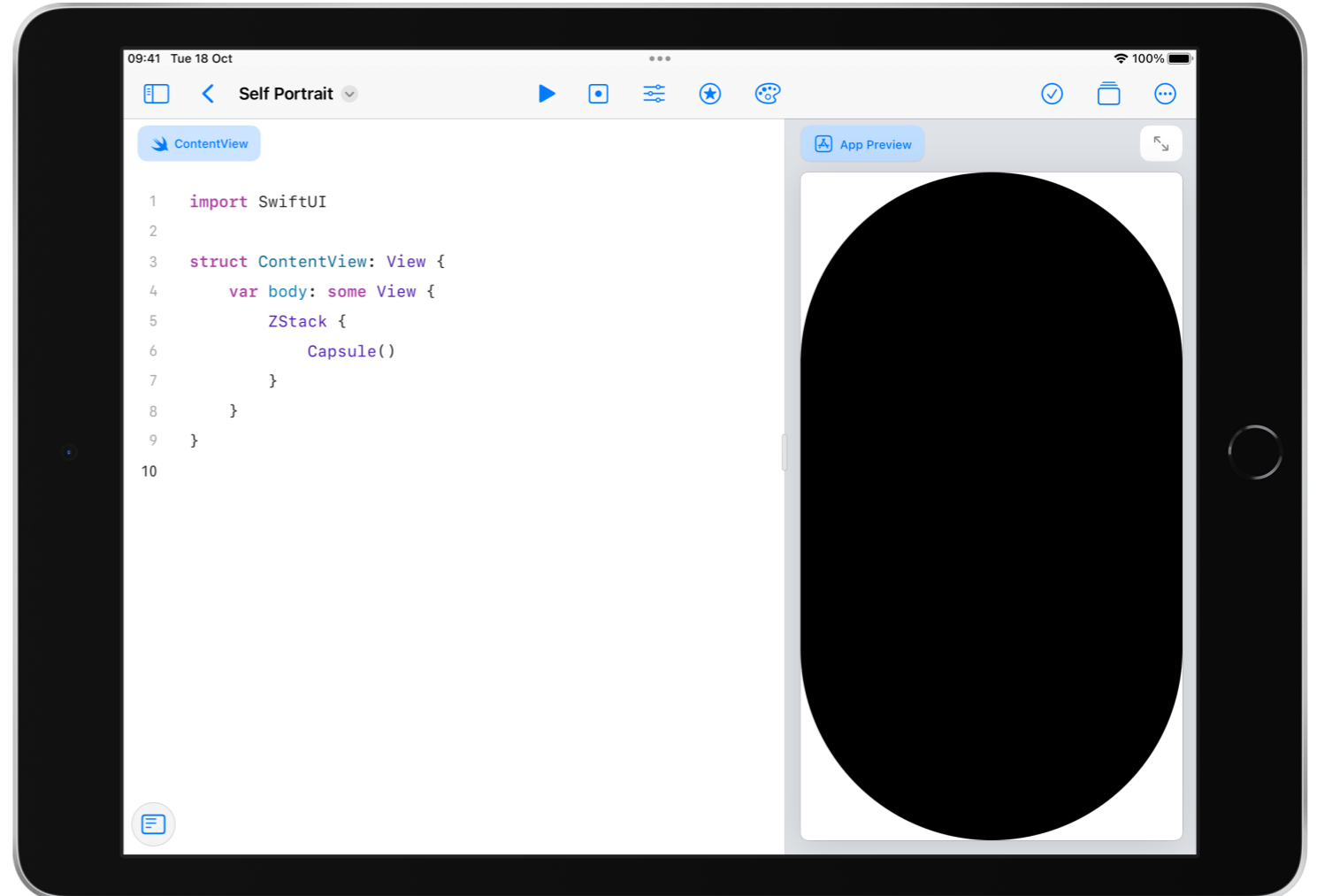
Open the list of views  and scroll to the bottom to find the shapes.

- Capsule
- Circle
- Ellipse
- Rectangle
- Rounded Rectangle



Step 3

Add a shape to the ZStack for your head. Use a **RoundedRectangle** for a square face, or a **Capsule** for a rounder face.



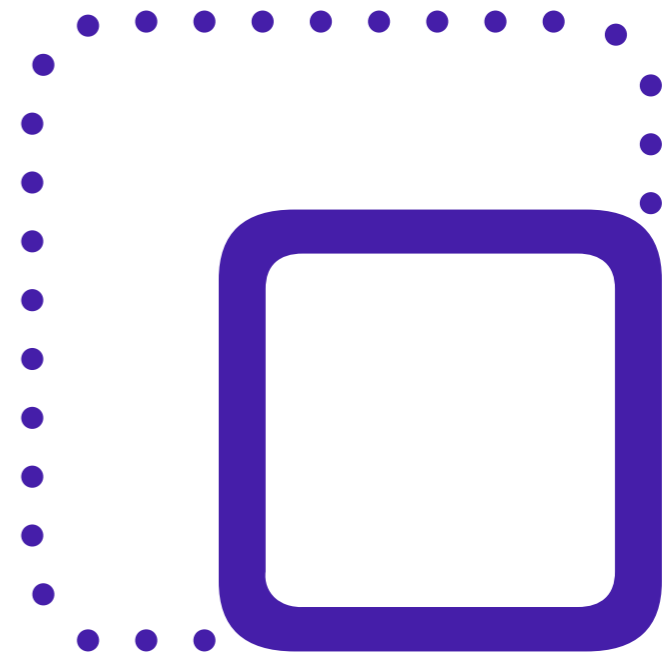
 **TIP**

Rounded rectangles
require a corner radius.

```
RoundedRectangle(cornerRadius: 25.0)
```

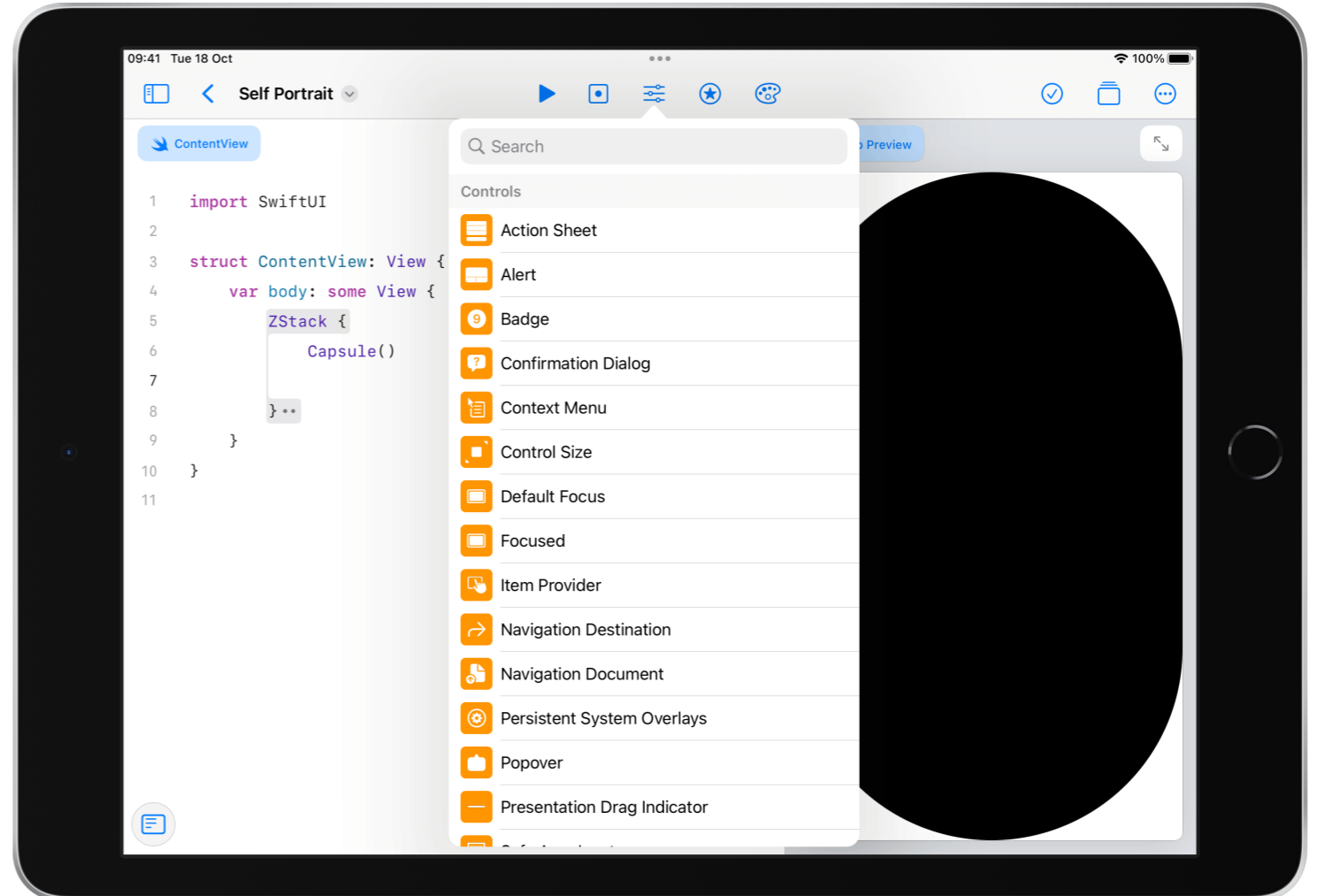
Adjust the size of a shape

Use the **frame** modifier to specify the size of a shape.



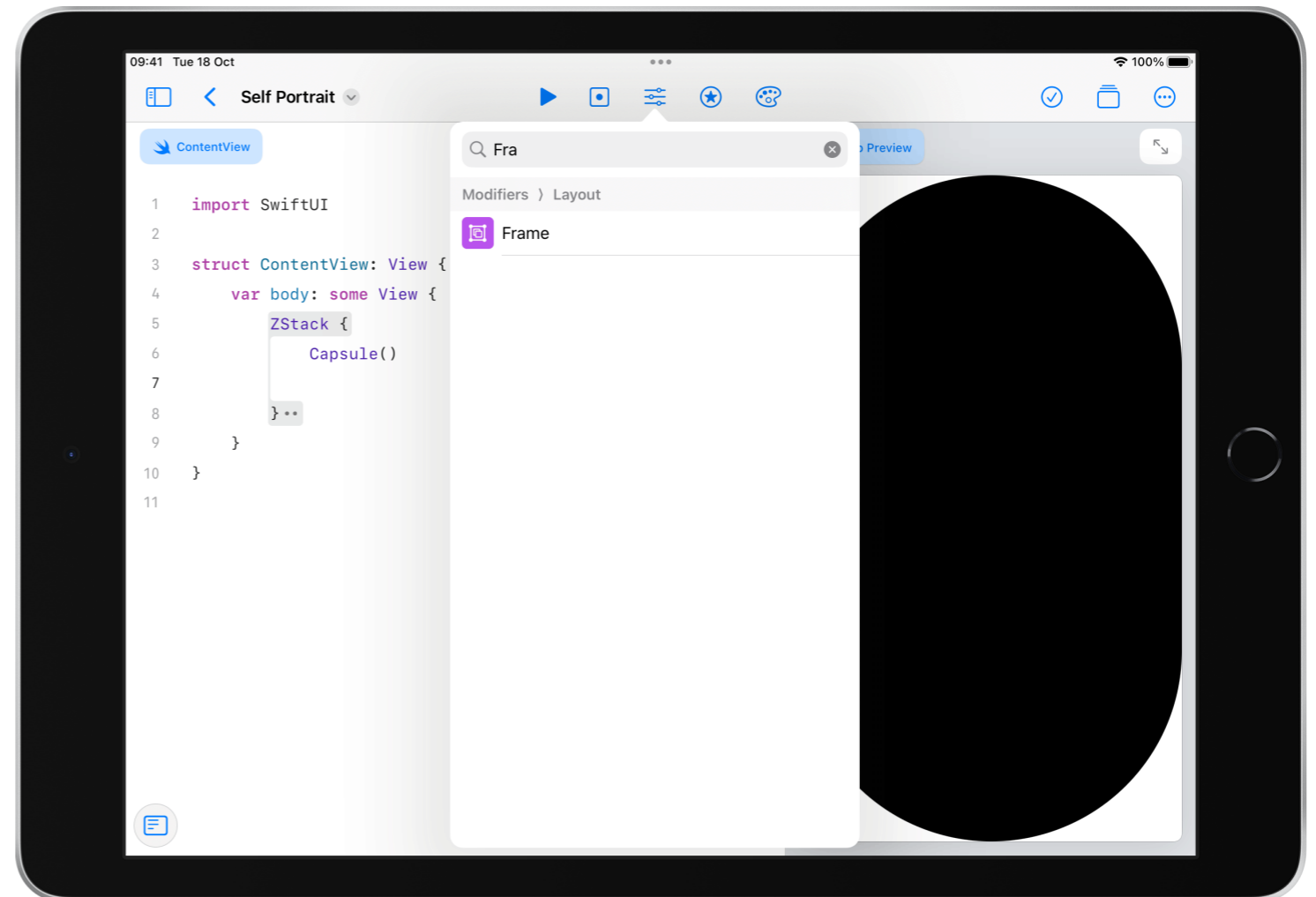
Step 1

Add a blank line below your shape, then open the list of modifiers.



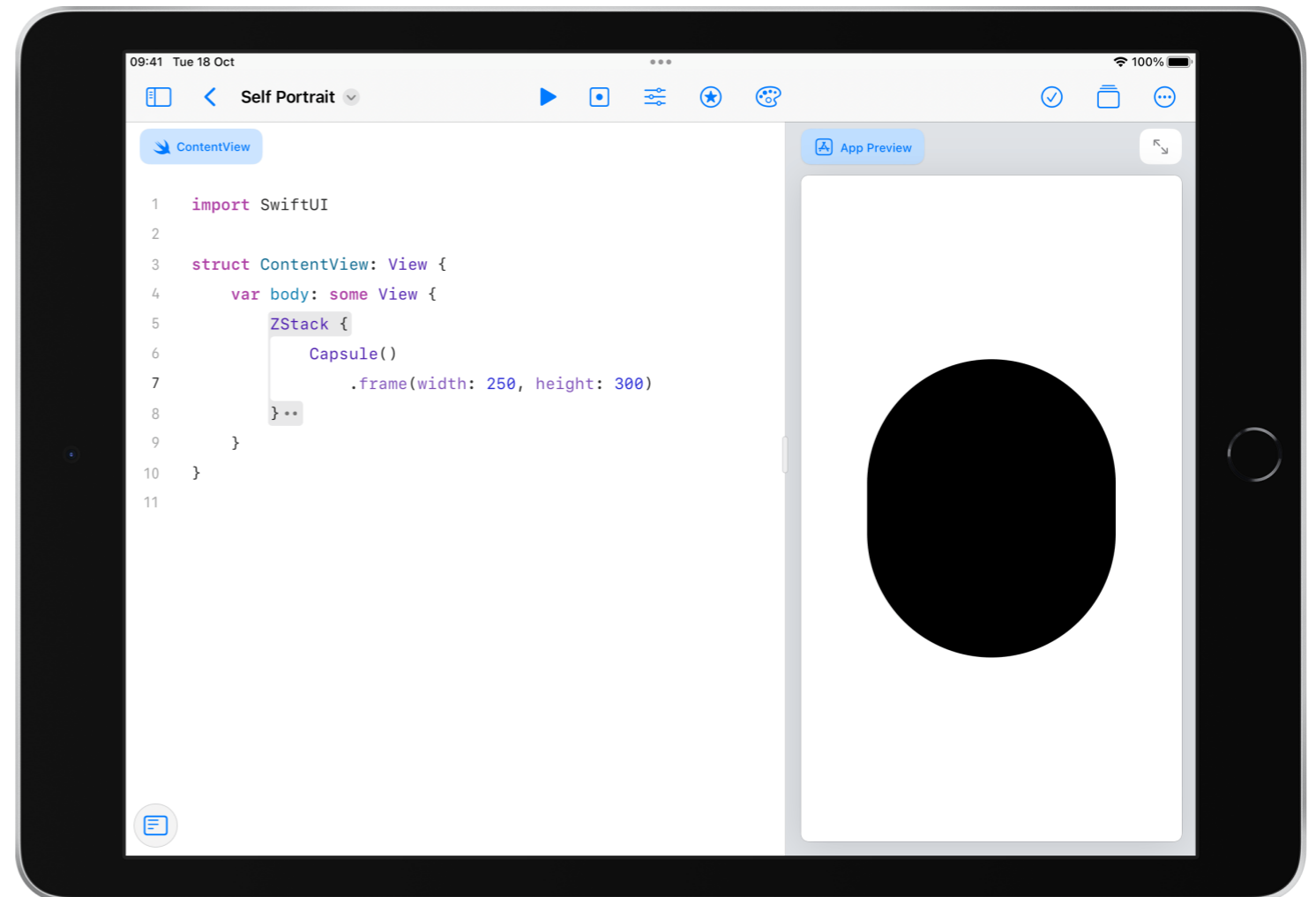
Step 2

Search for **frame** and tap to add it to your code.



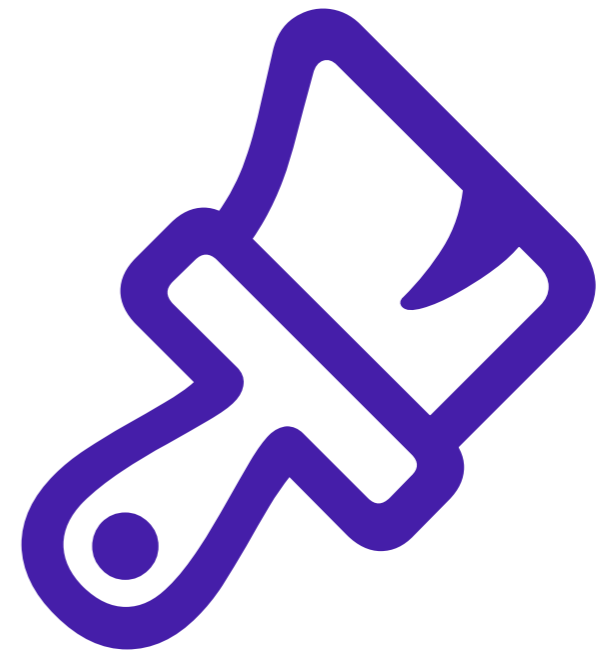
Step 3

Fill in the approximate values for height and width to create your head.



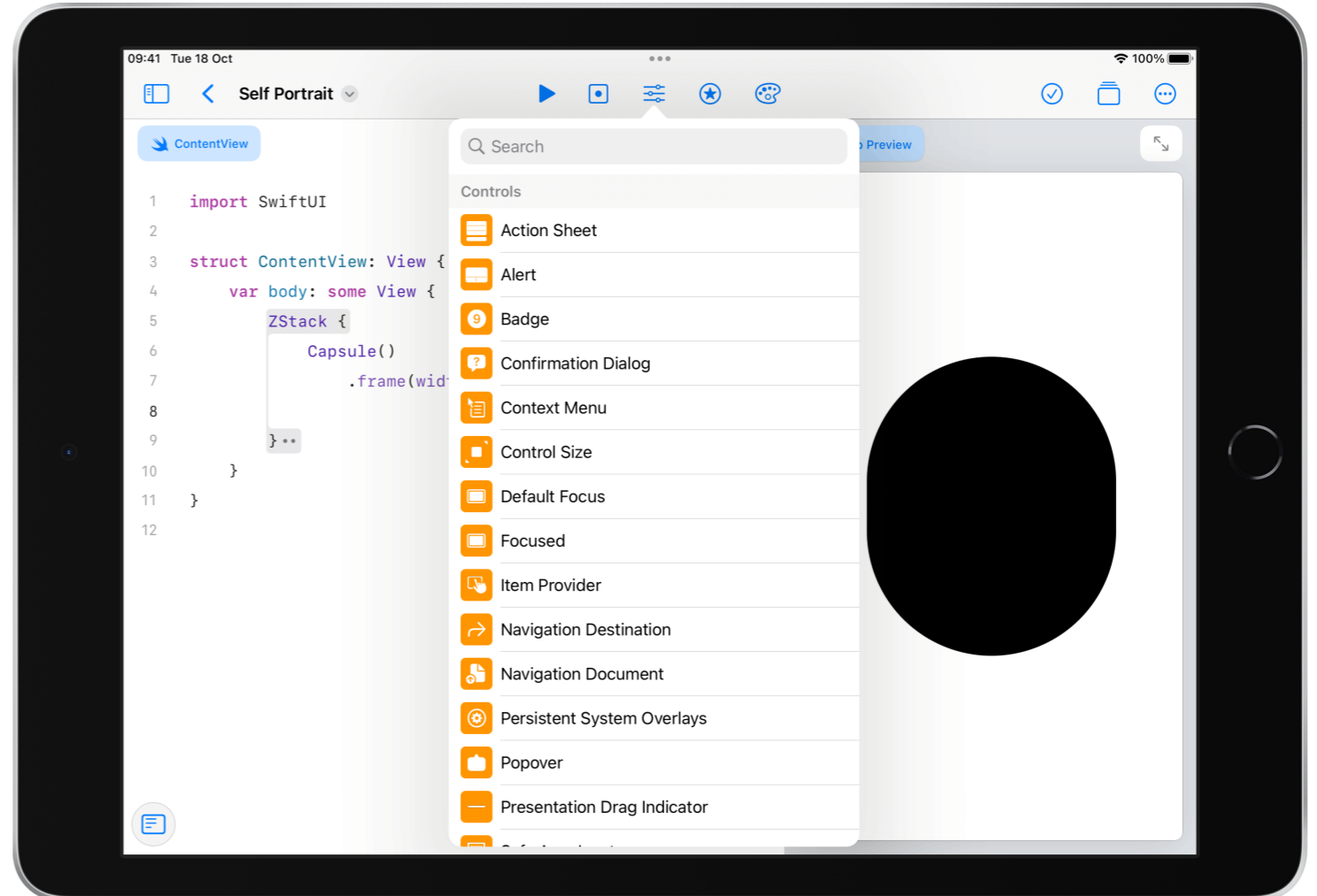
Add colour

Use the **foregroundStyle** modifier to give the shape a colour.



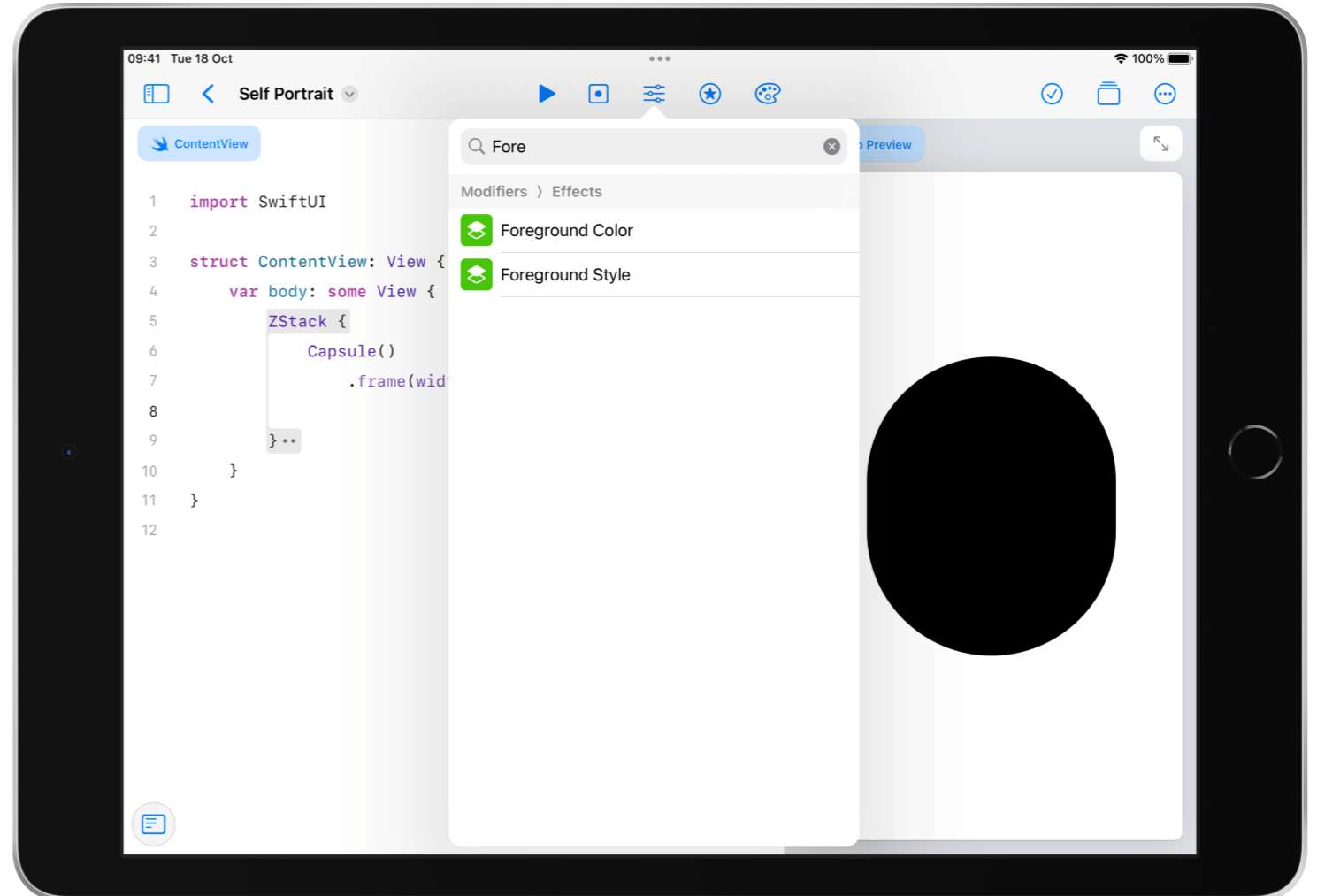
Step 1

Add a blank line below
frame, then open the
list of modifiers.



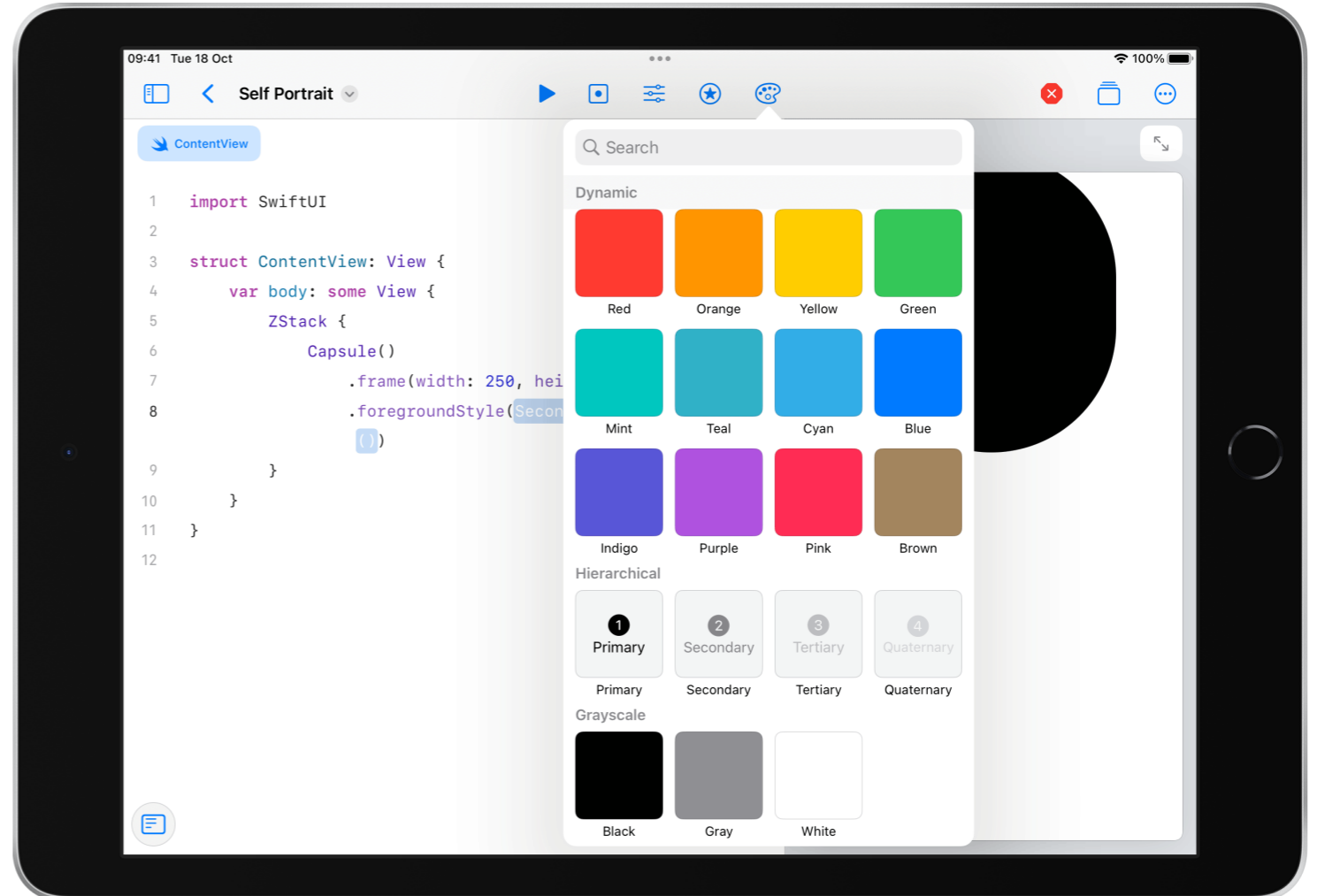
Step 2

Type **foregroundColorStyle** into the search bar. Tap it to add it to your code.



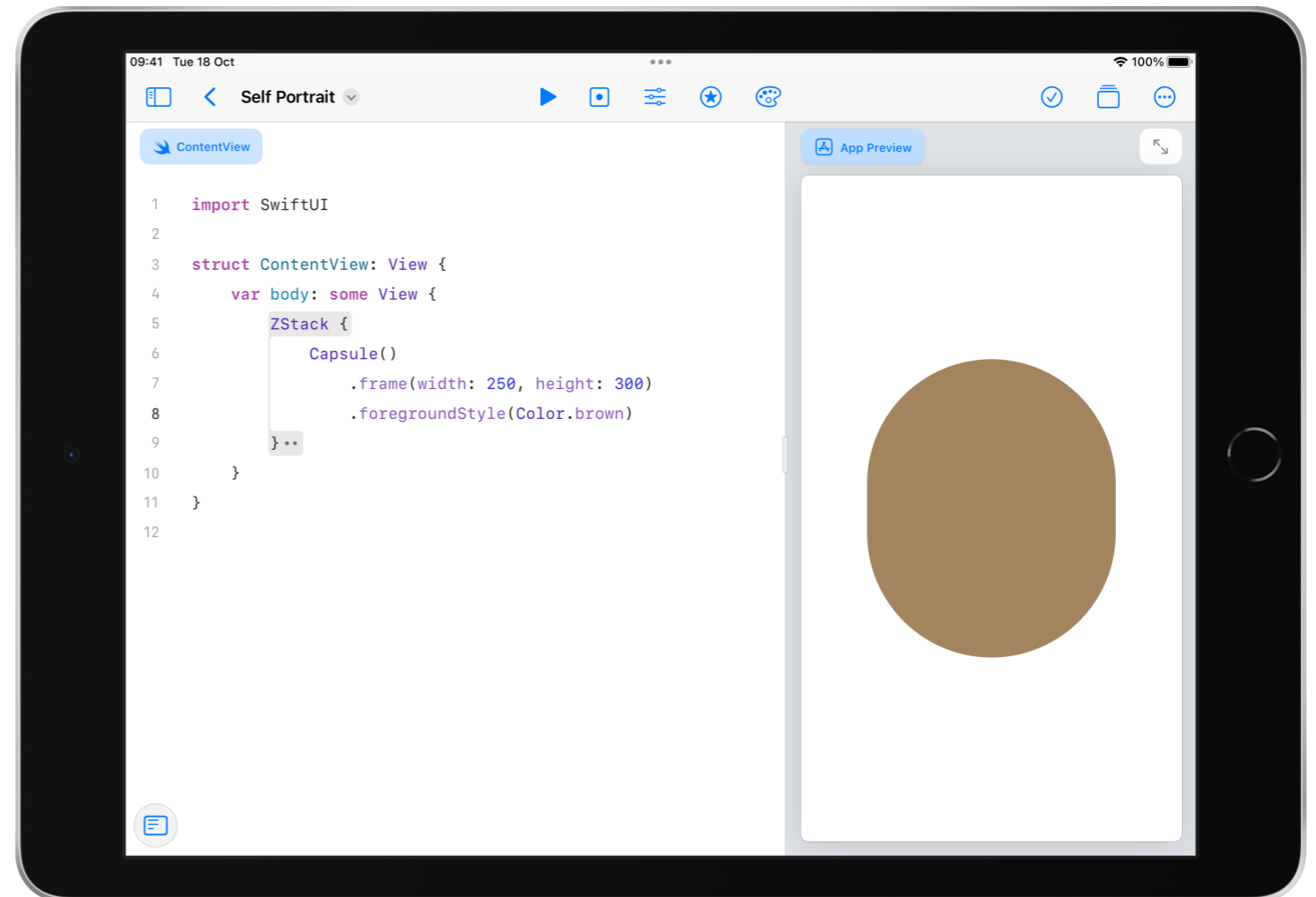
Step 3

To change the colour, select the code inside the parentheses and open the list of colours.



Step 4

Tap any colour to add it to your code.



 **TIP**

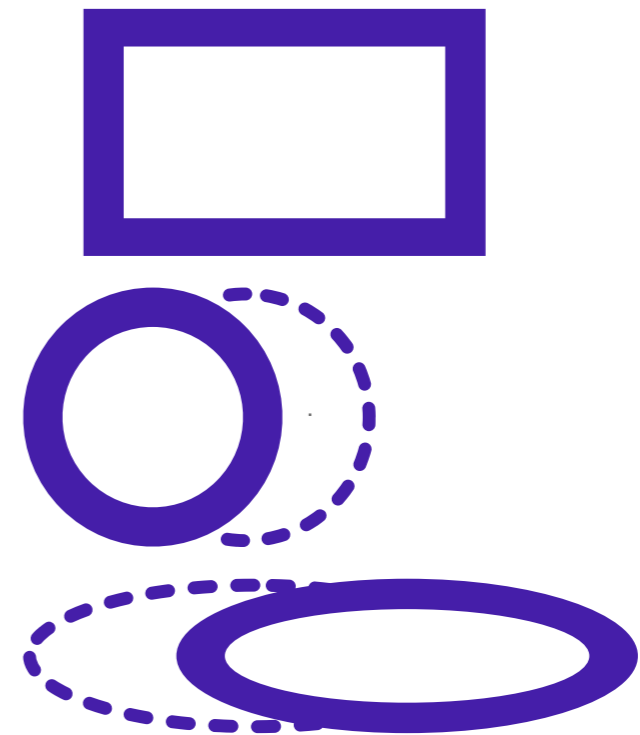
Optionally, create your own colour. Choose one of the code snippets, then edit the values to match your skin tone.

```
• foregroundStyle(Color(  
  CGColor(  
    red: 0.6,  
    green: 0.4,  
    blue: 0.3,  
    alpha: 1.0)))
```

```
• foregroundStyle(Color(  
  hue: 0.1,  
  saturation: 0.9,  
  brightness: 0.4))
```

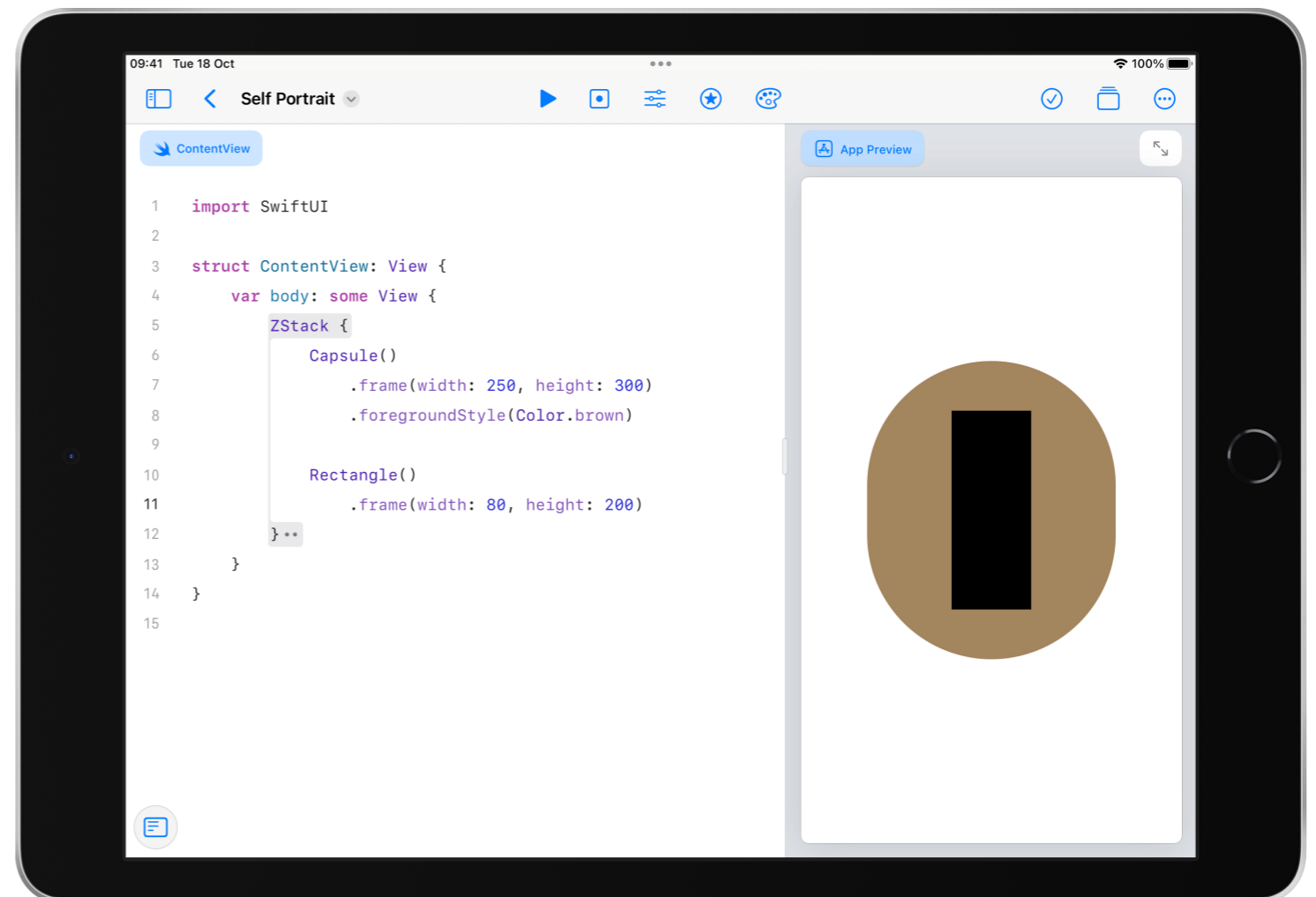
Shift a view

Add a neck, then shift it down using **offset**.



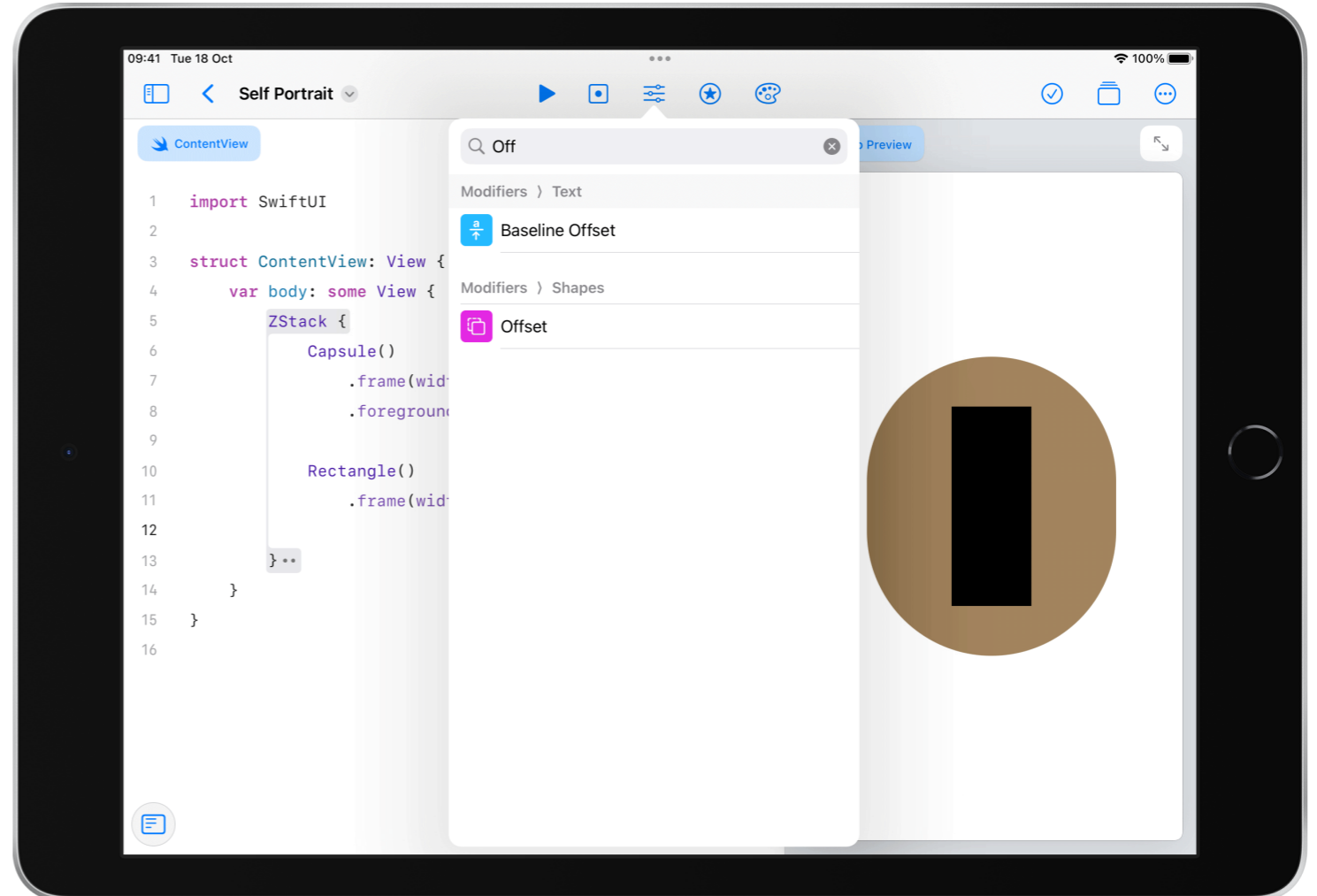
Step 1

Apply what you've learnt to add a rectangle for your neck and adjust the size with a frame modifier.



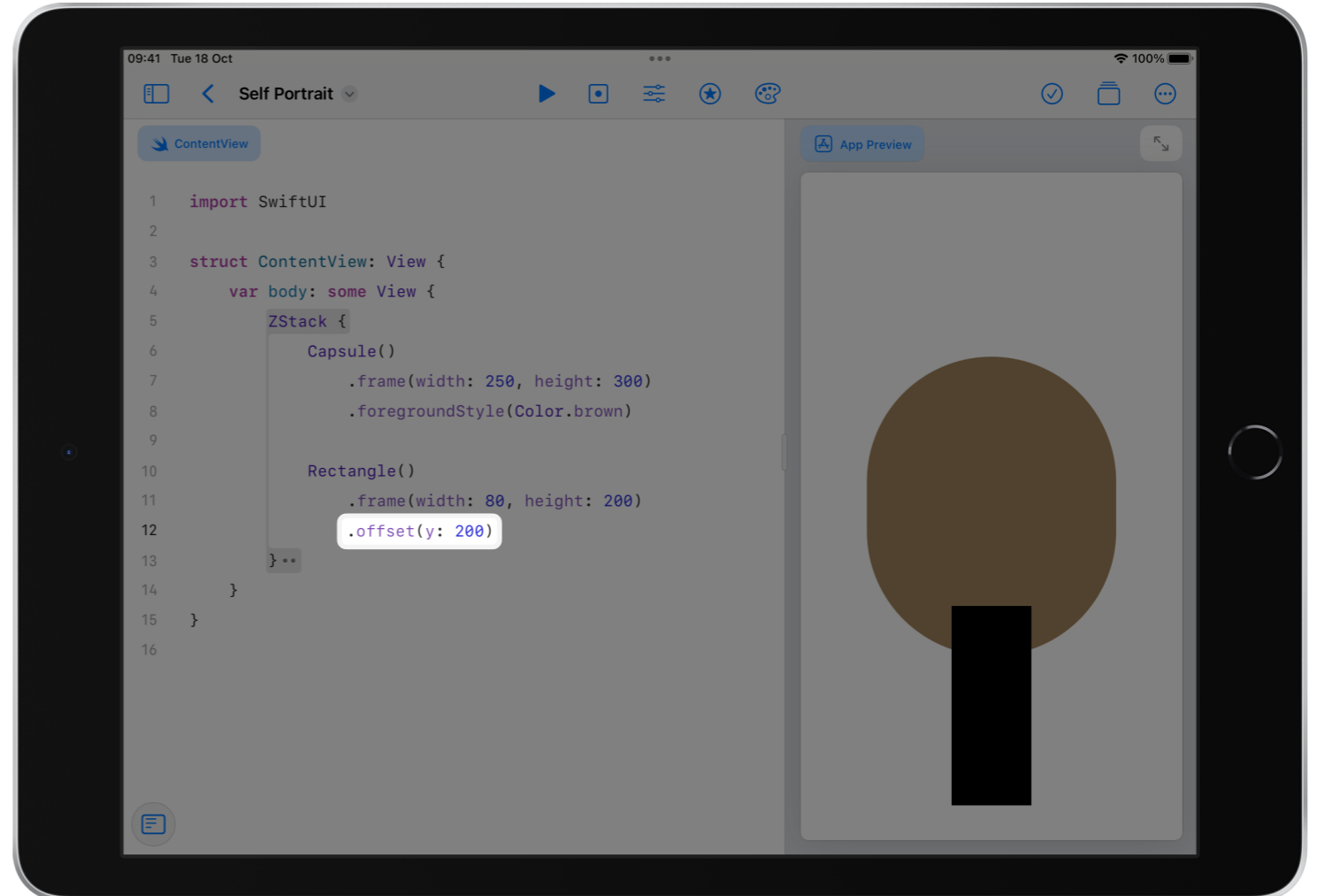
Step 2

Add a blank line below the rectangle, then search for **offset** in the list of modifiers.



Step 3

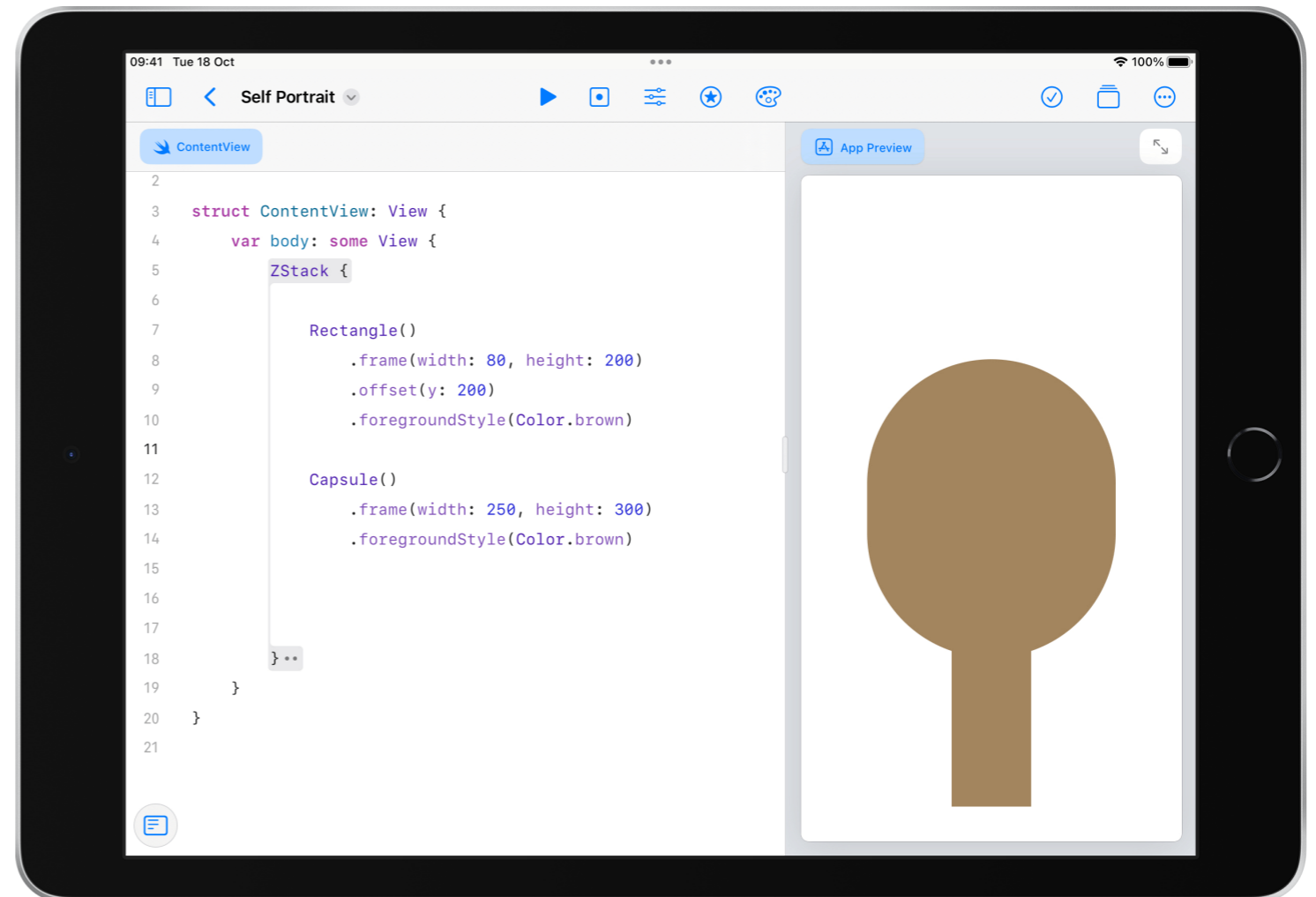
Tap `offset` and edit the value to shift the shape down. Optionally, delete any parameters you don't use.



Step 4

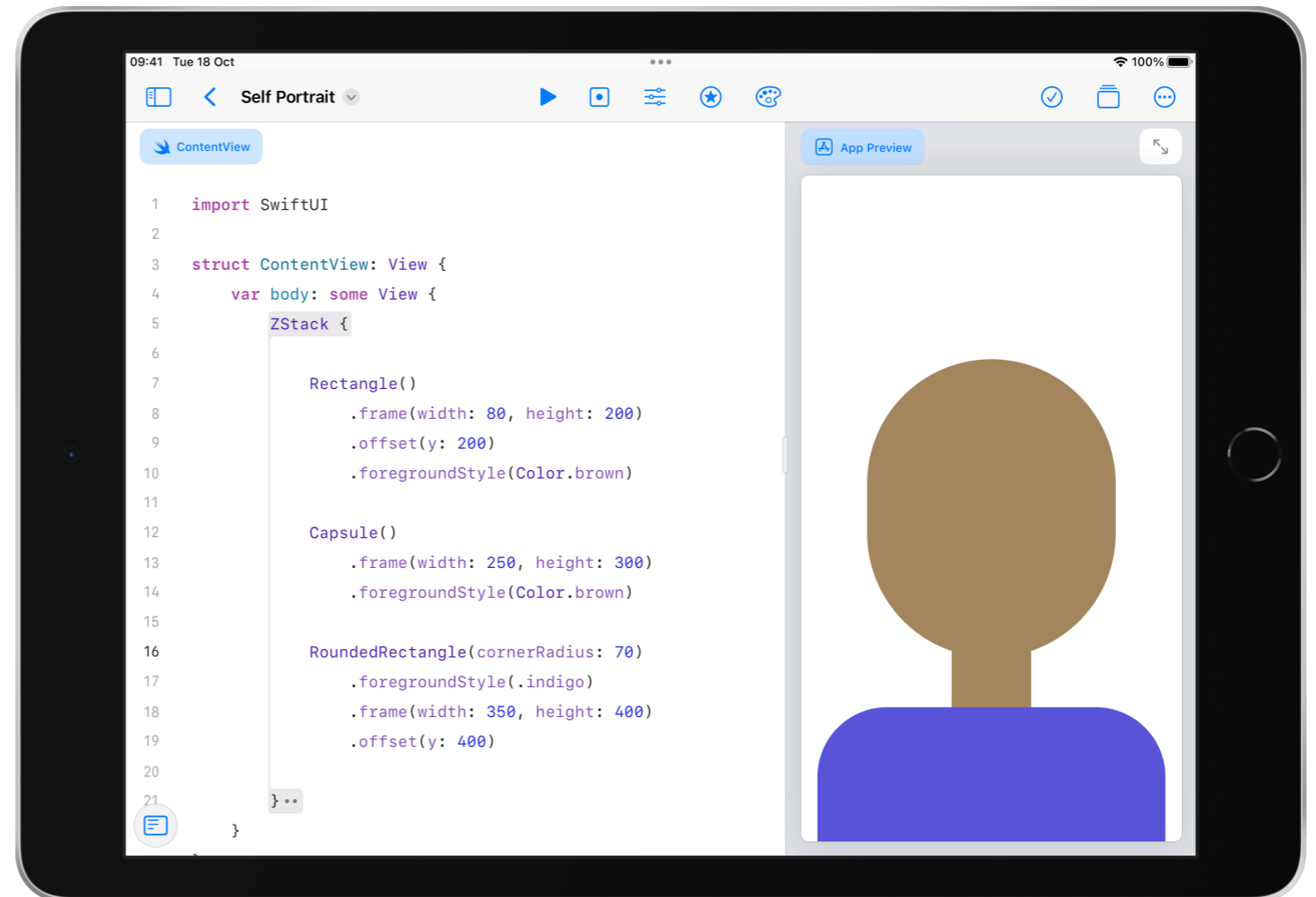
Move the neck under the head:

- Select the rectangle and any modifiers you added
- Cut and paste the code to the top of the depth stack
- Then change the colour to match your head



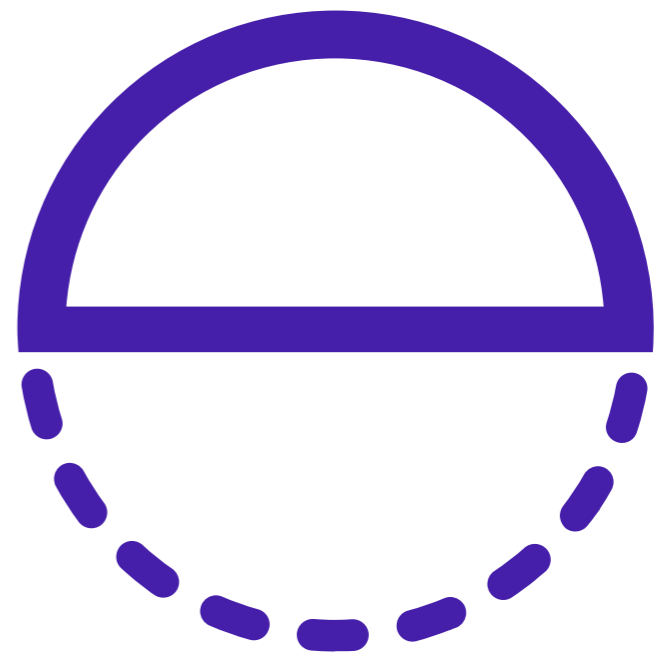
Step 5

Apply what you've learnt to add a shirt, place it correctly on the screen and give it a colour.



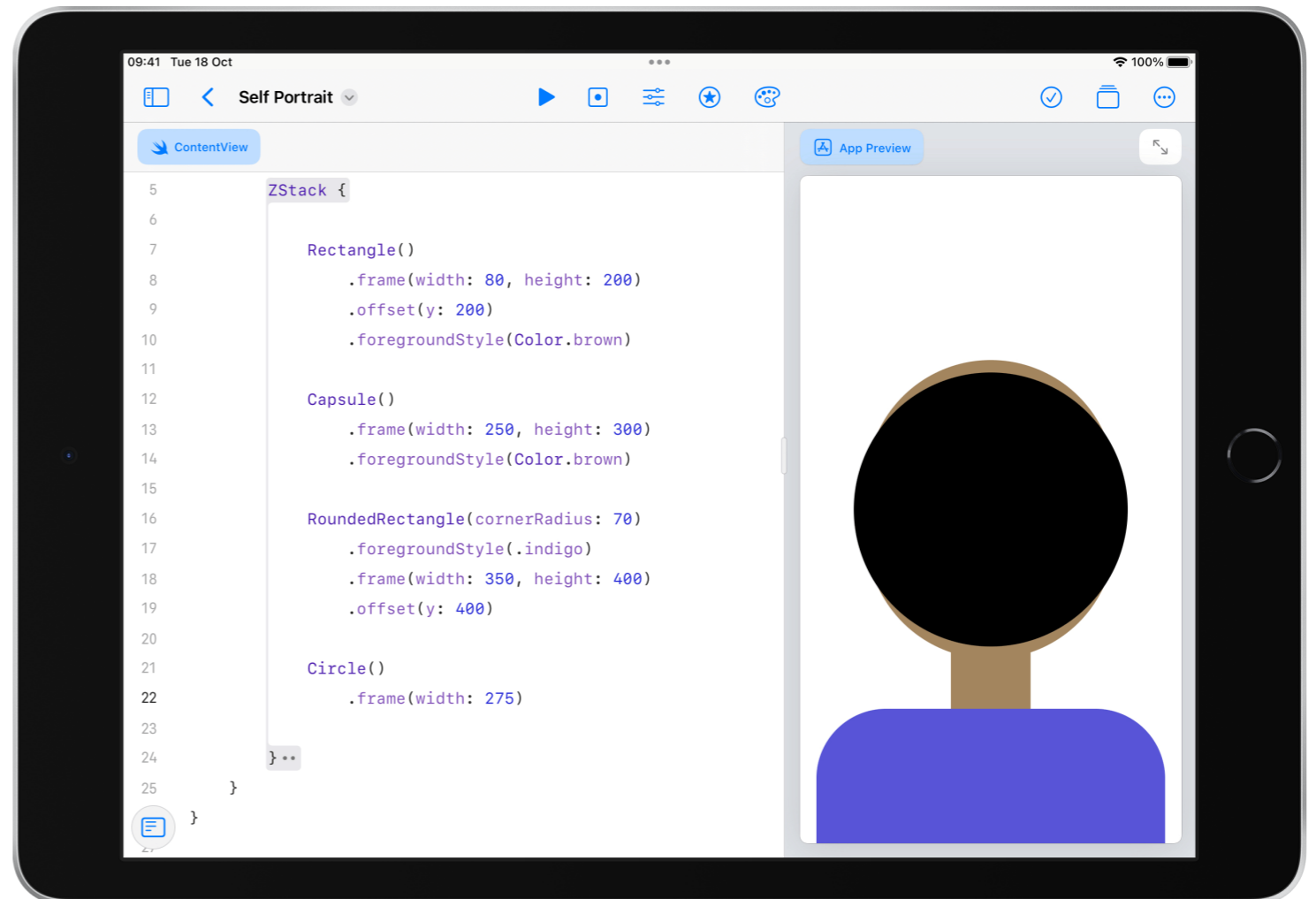
Trim a shape

Trim a shape to create a fringe or short hair.



Step 1

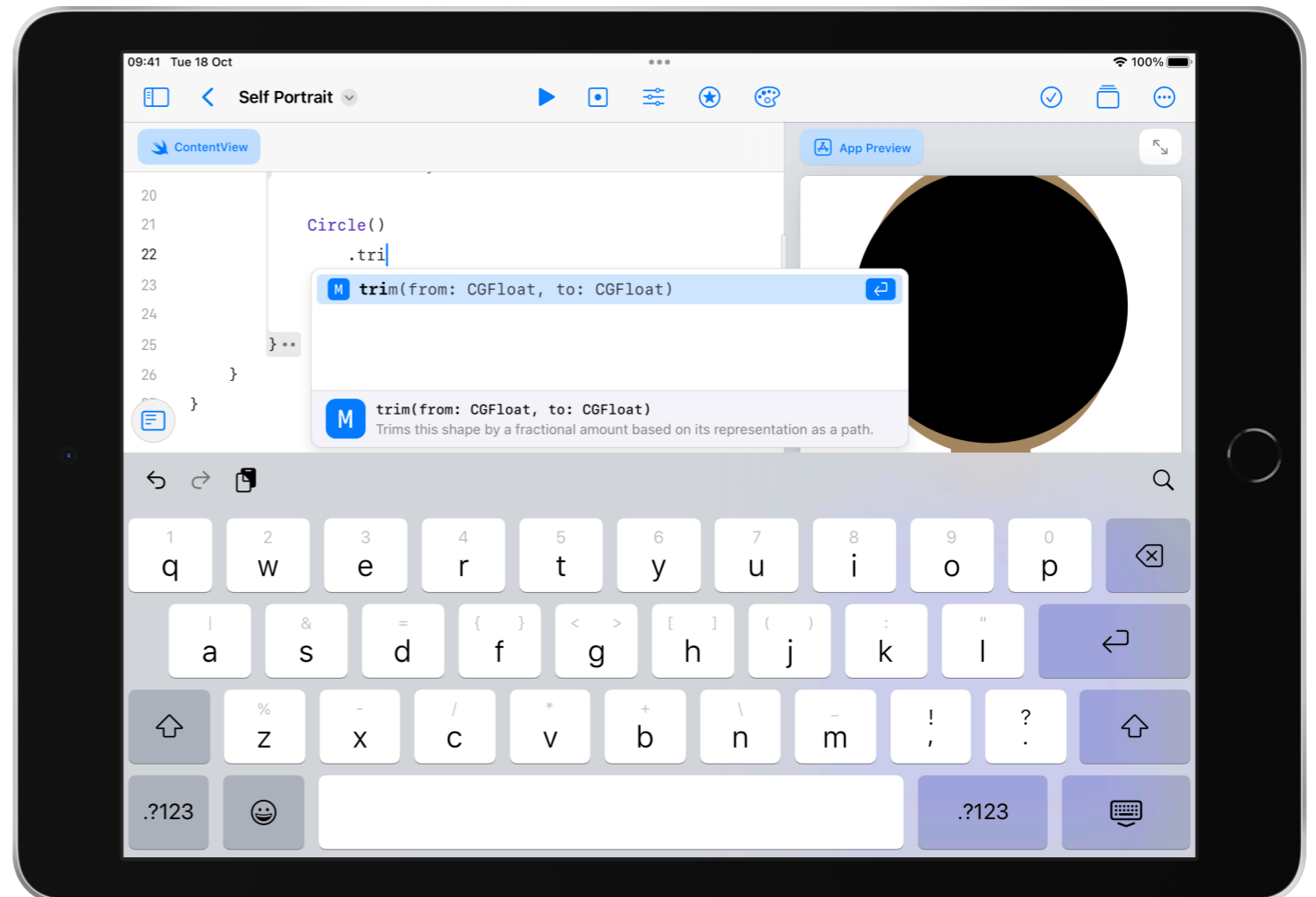
Add a circle for your hair.
Use frame to make it
about the same width as
your head or slightly larger.



Step 2

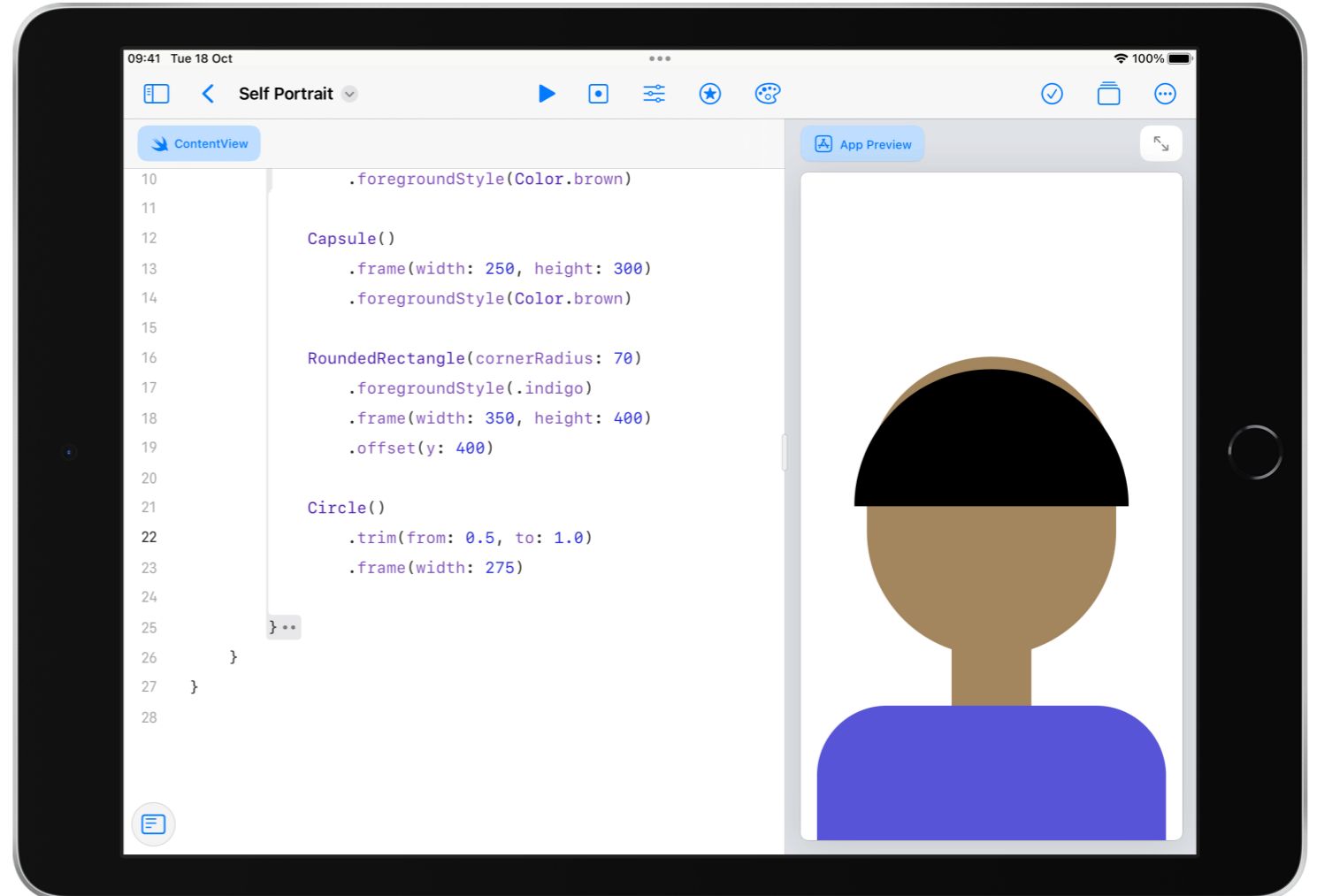
Add a blank line below the circle. Type `.trim` and tap return when it appears in code completion.

Note, the `trim` modifier must come directly below a shape.



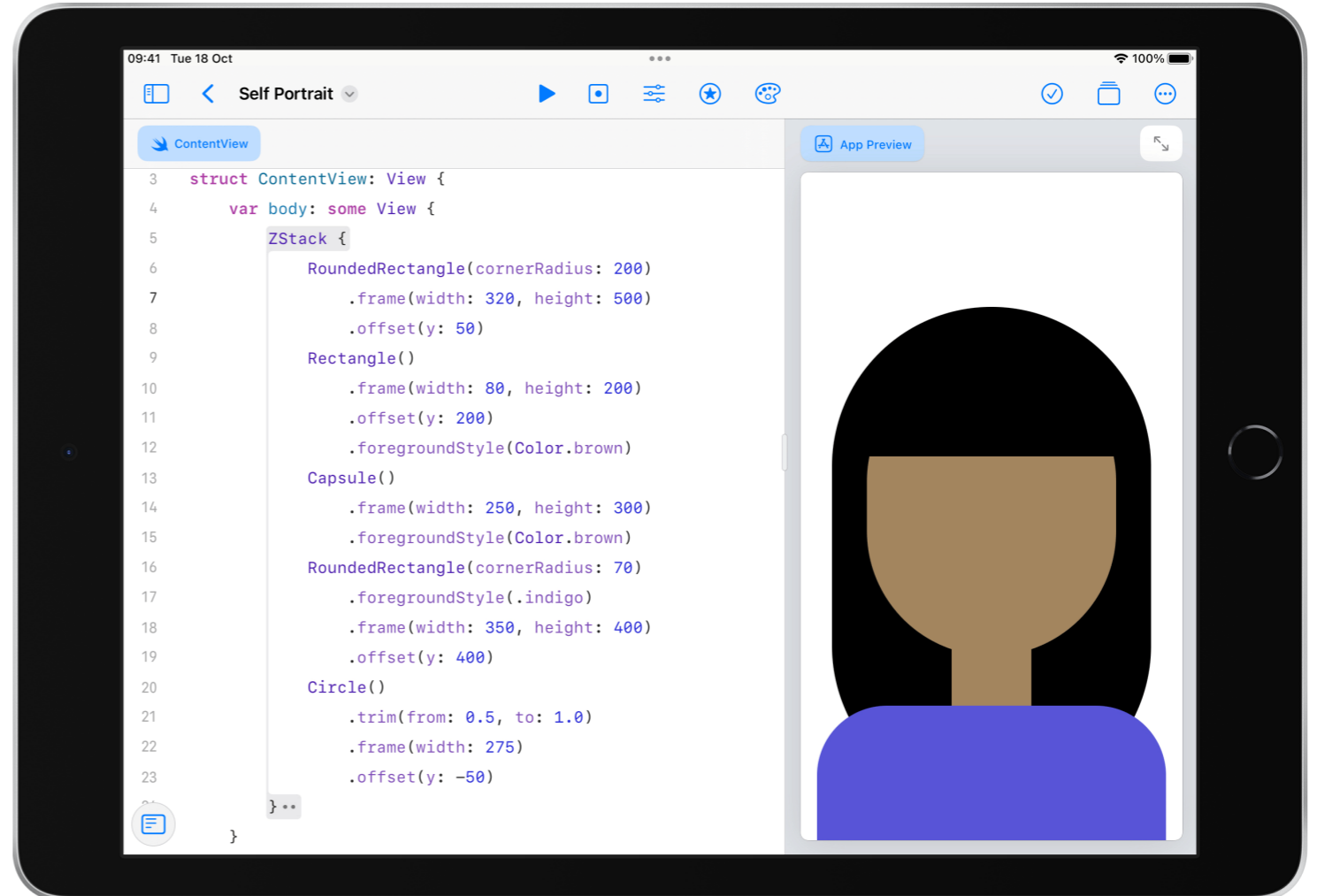
Step 3

Experiment with values for where the trim should start and end.



Step 4

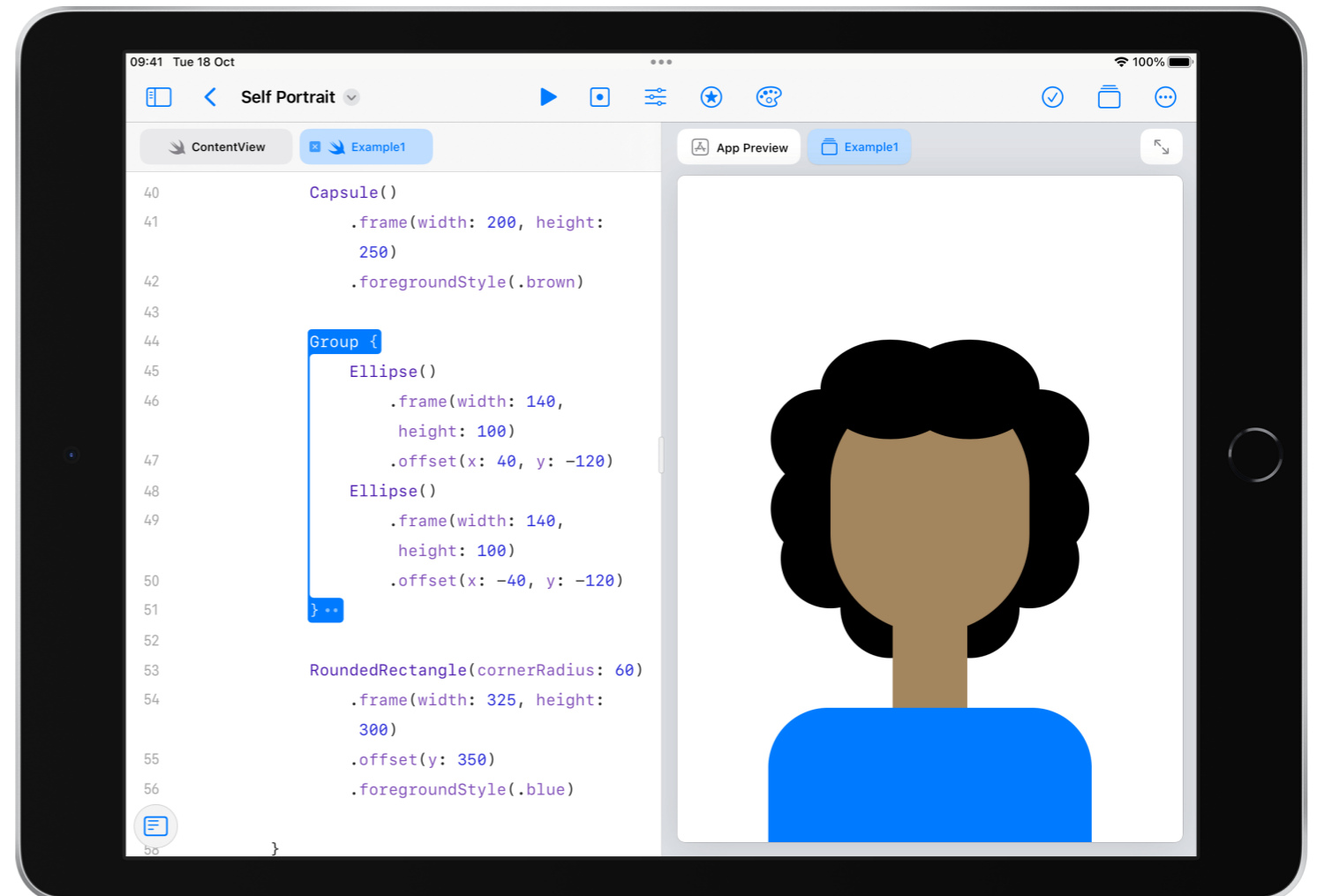
Adjust the location of the fringe or short hair. Continue adding any remaining elements to complete your hair.



💡 TIP

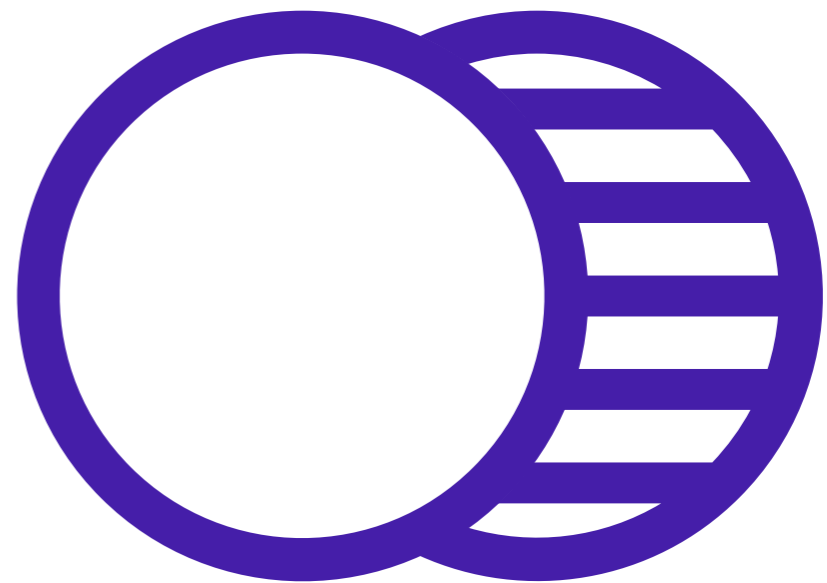
If you have more than ten views in a stack, you will get an “Extra arguments at position...” warning.

If this happens, group elements like you see here.



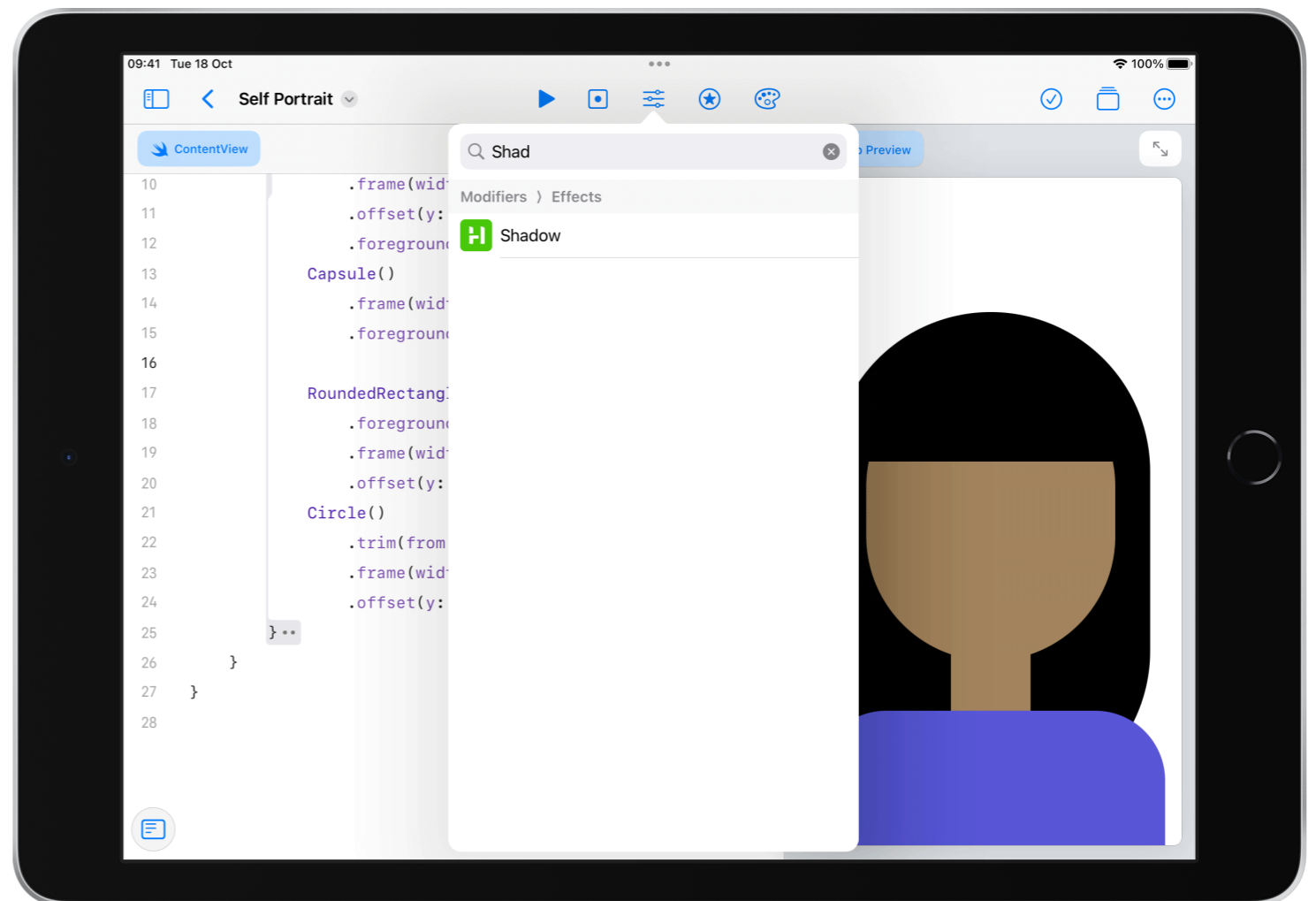
Add a shadow

Use a shadow to separate your head and neck.



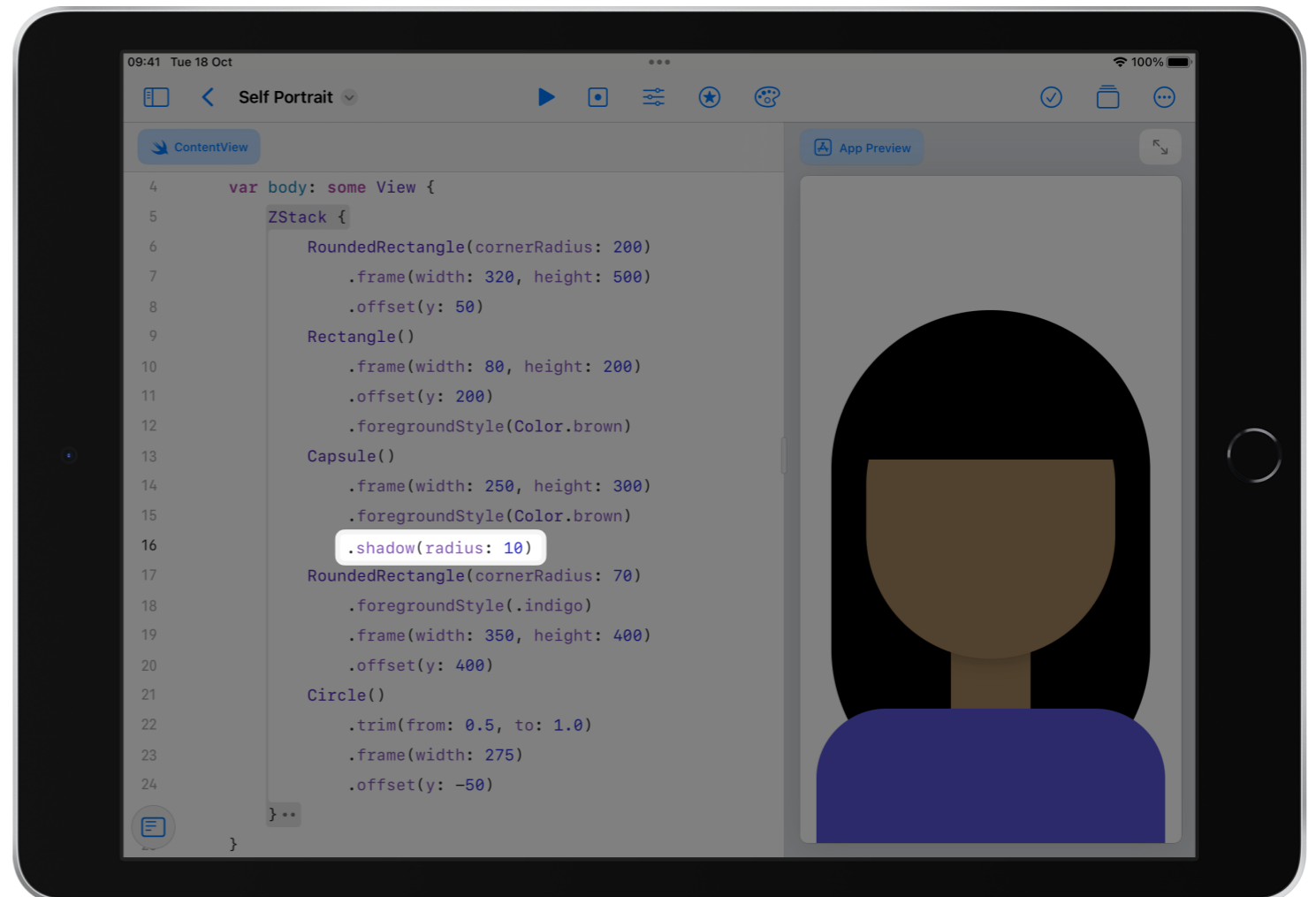
Step 1

Locate the shape that creates your head in the code. Add a blank line below the shape, then search for **shadow** in the list of modifiers.



Step 2

Edit the radius of the shadow. Experiment with different values to see how it looks.



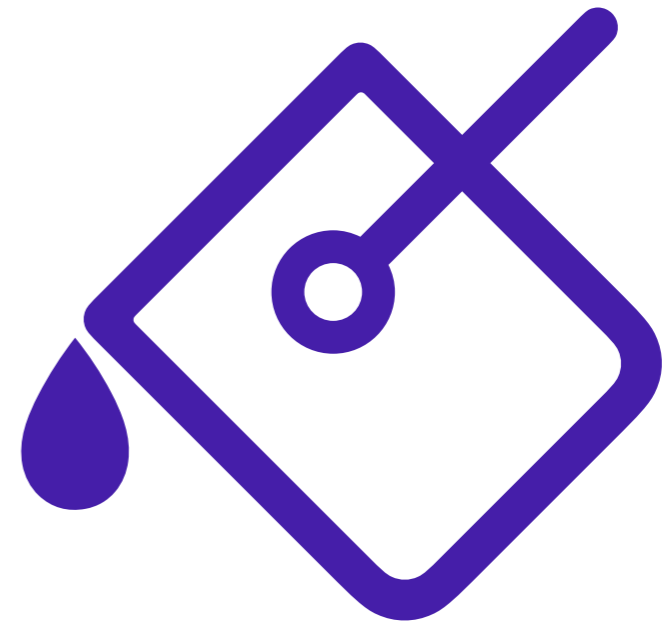
 **TIP**

Optionally, add a colour for the shadow.

```
.shadow(color: .black, radius: 10)
```

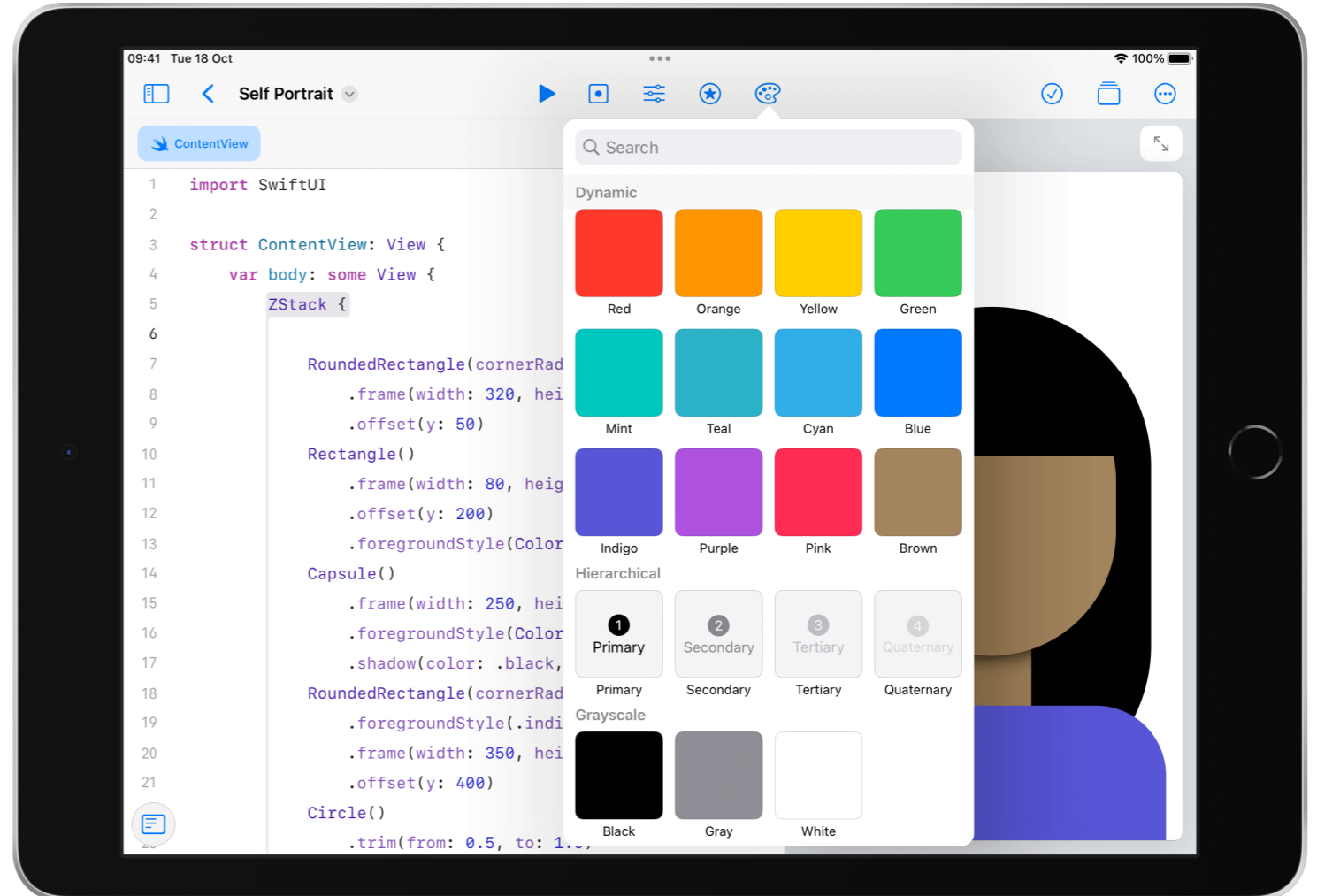
Add a background colour

Add a background colour for your self portrait.



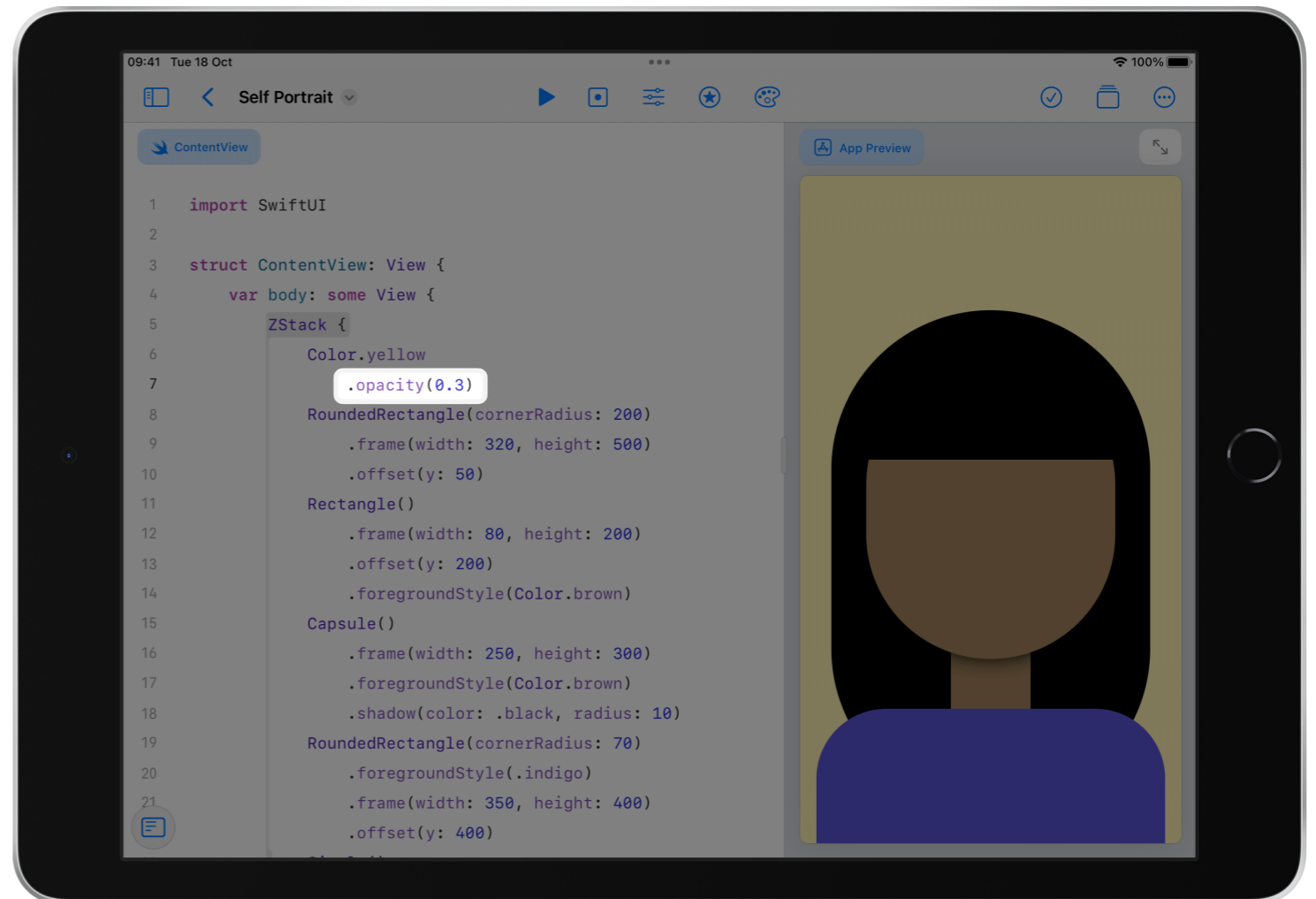
Step 1

Add a blank line at the top of your ZStack, then open the list of colours. Tap a colour to add it to your code.



💡 TIP

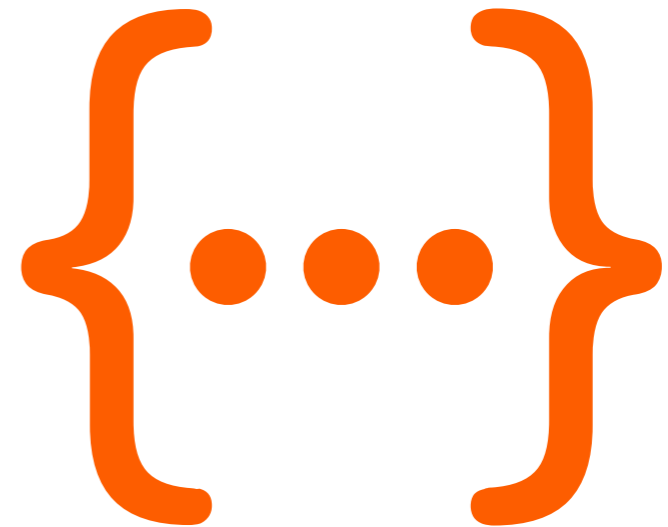
Optionally, add an `opacity` modifier to dim the background colour.



Section 2

Code reference

Use the following reference to quickly edit your view or create new ones.



Code reference

Stacks

```
HStack {  
}  
VStack {  
}  
ZStack {  
}
```

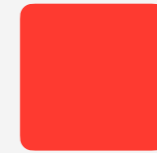
Shapes

```
Circle()  
Ellipse()  
Capsule()  
Rectangle()  
RoundedRectangle(cornerRadius: 20)
```

Modifiers

```
.frame(width: 100, height: 200)  
.foregroundColor(Color.blue)  
.offset(x: 40, y: 10)  
.trim(from: 0.5, to: 1.0)  
.shadow(radius: 10)  
.opacity(0.5)
```

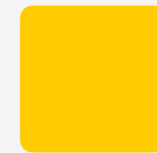
Colours `Color.black`



Red



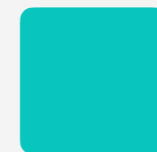
Orange



Yellow



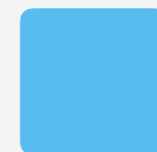
Green



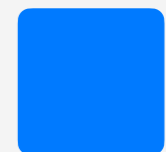
Mint



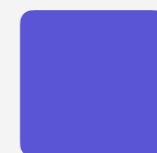
Teal



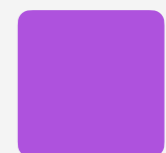
Cyan



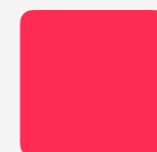
Blue



Indigo



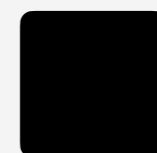
Purple



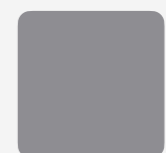
Pink



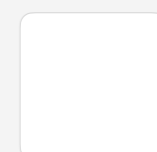
Brown



Black



Grey



White



TM and © 2023 Apple Inc. All rights reserved.