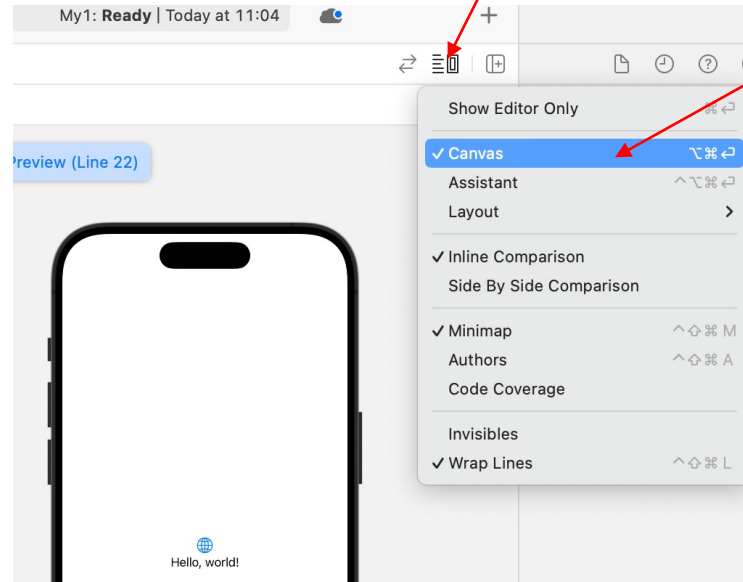
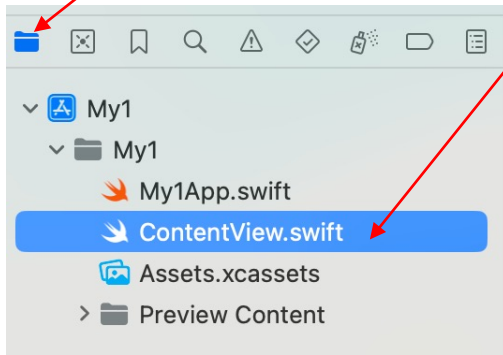
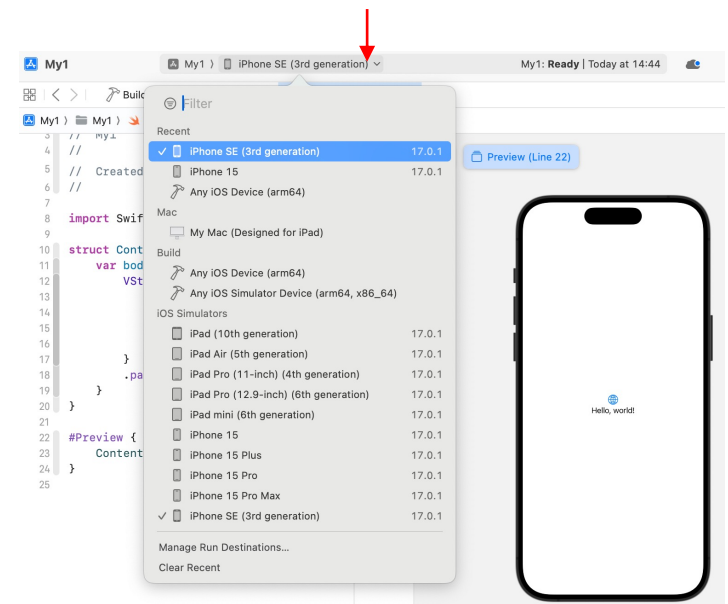


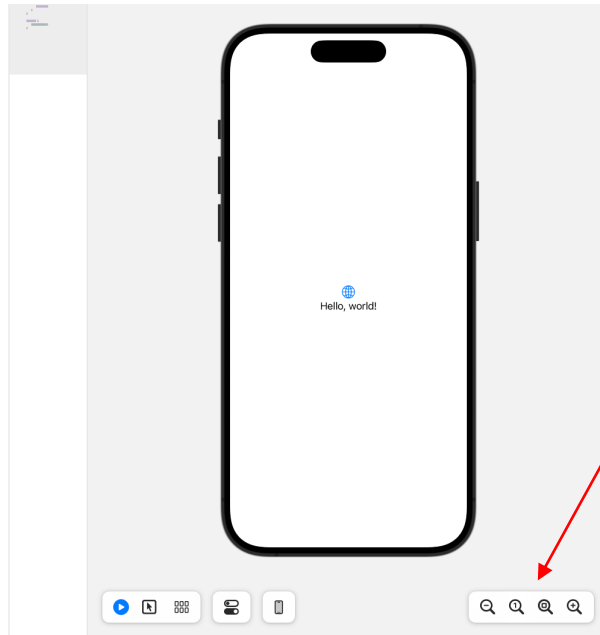
① Project navigatorクリック → ContentView.swiftクリック → Auto Editor Optionクリック → Canvas → プレビュー画面表示



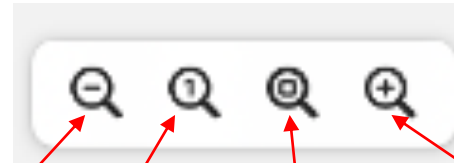
② クリックするとiPhoneの機種名が表示 → 選定してクリック



```
6 //
7
8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         VStack {
13             Image(systemName: "globe")
14                 .imageScale(.large)
15                 .foregroundColor(.tint)
16             Text("Hello, world!")
17         }
18         .padding()
19     }
20 }
21
22 #Preview {
23     ContentView()
24 }
25
```



③ 縮尺を選定



縮小 100% iPhone全体が入る 拡大

```
8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         VStack {
13             Image(systemName: "globe")
14                 .imageScale(.large)
15                 .foregroundColor(.tint)
16             Text("Hello, world!")
17         }
18         .padding()
19     }
20 }
21
```

④削除

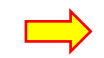


```
8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         Text("Hello, world!")
13     }
14 }
15
```

⑤Textの前にカーソルを置き、「command」+「A」でコードの全選択→「control」+「I(アイ)」 ←インデントが調整



```
8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         Text("Hello, world!")
13     }
14 }
15
16 #Preview {
17     ContentView()
18 }
19
```



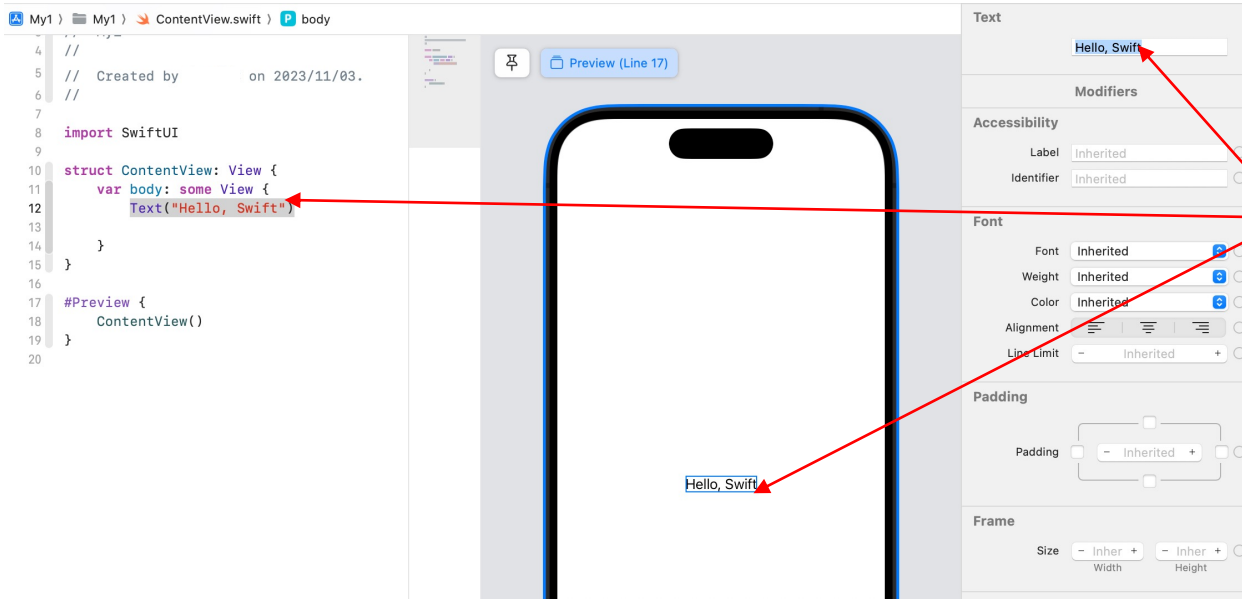
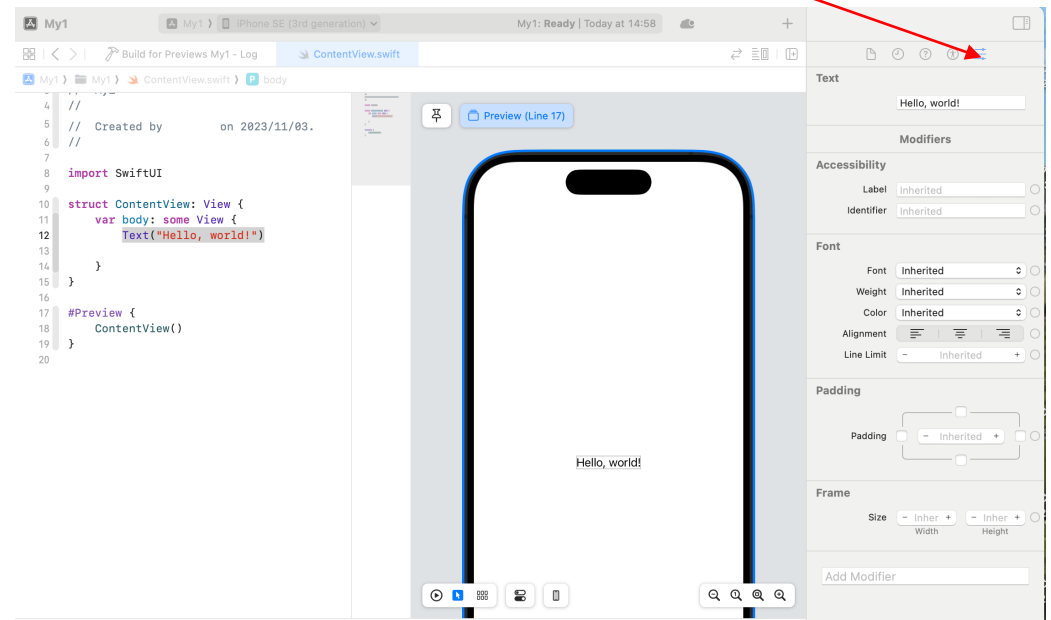
```
8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         Text("Hello, world!")
13     }
14 }
15
16 #Preview {
17     ContentView()
18 }
19
```

⑧ Attributes inspector クリック



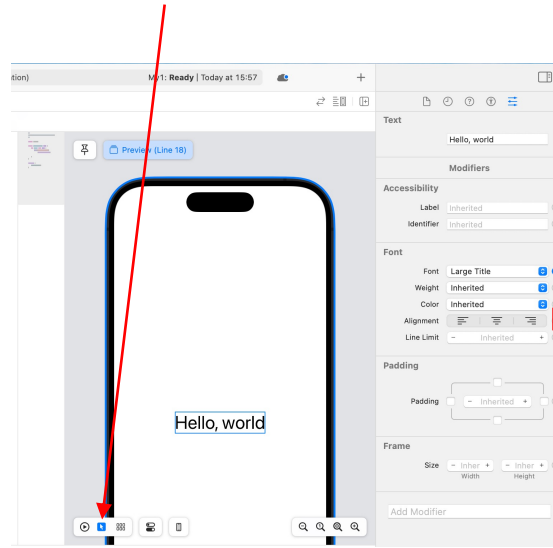
⑦ いずれか クリック

⑥ Selectable ボタン クリック

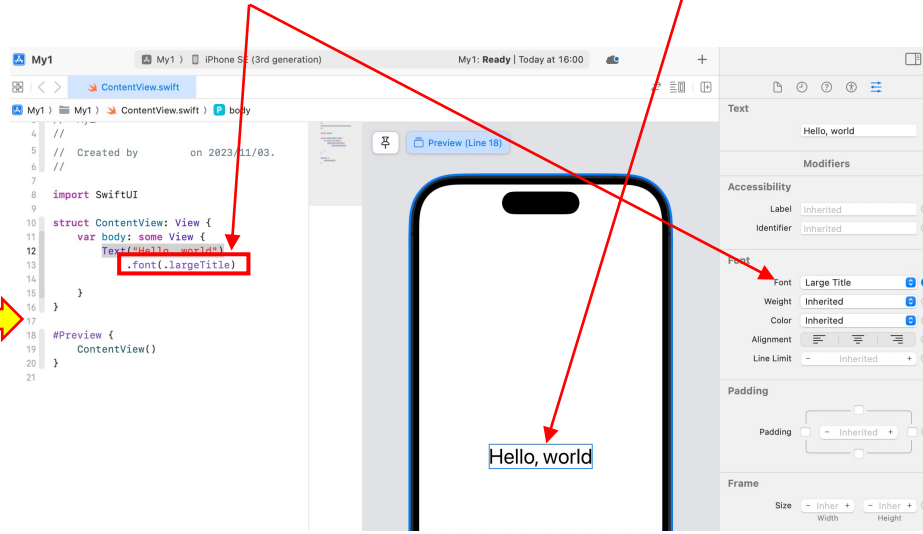


⑨ Hellow, Swift! に書き換える

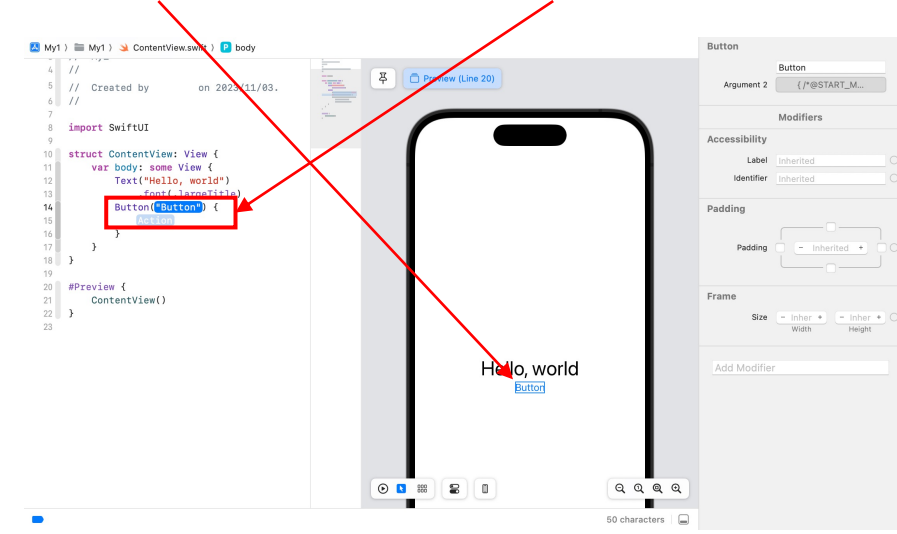
⑩ Selectableボタンクリック



⑪ FontをLarge Titleにすると文字が大きくなる



⑭「Button」が表示され、コードに追記



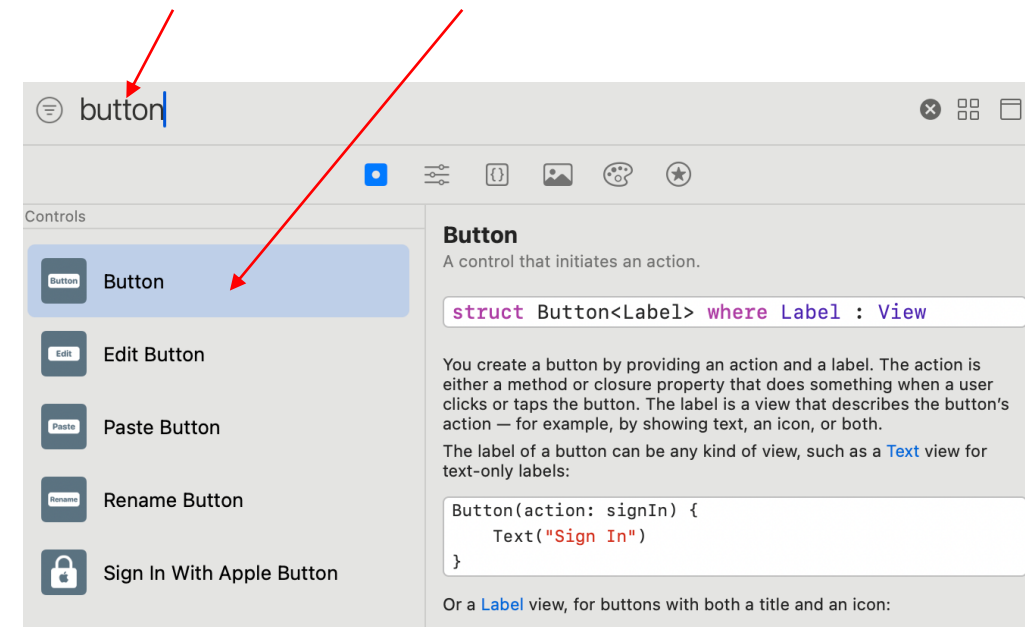
⑫カーソルをここに置き「shift」+「command」+「L」

```

8  import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         Text("Hello, world")
13         .font(.largeTitle)
14     }
15 }
16
17
18 #Preview {
19     ContentView()
20 }
21

```

⑬「button」で検索→「Button」をダブルクリック



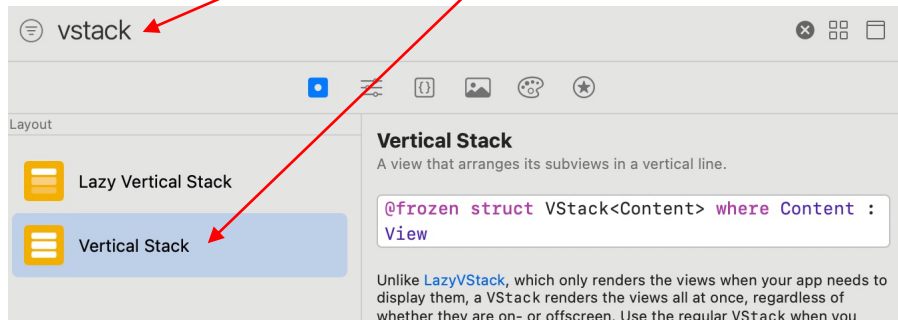
⑮ 一行追加 → カーソルをここに置き「shift」+「command」+「L」

```

8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         Text("Hello, world")
13             .font(.largeTitle)
14         Button("Button") {
15             Action
16         }
17     }
18 }
19
20
21 #Preview {
22     ContentView()
23 }
24

```

⑯ 「vstack」で検索 → vertical Stackをダブルクリック

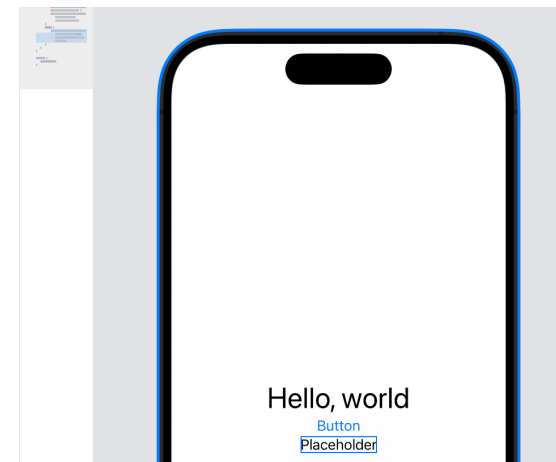


⑰ Vstackが表示 → 「Content」ダブルクリックで消去

```

6 //
7 import SwiftUI
8
9
10 struct ContentView: View {
11     var body: some View {
12         Text("Hello, world")
13             .font(.largeTitle)
14         Button("Button") {
15             Action
16         }
17     }
18 }
19
20
21 #Preview {
22     ContentView()
23 }
24
25
26

```



⑱ 「command」+「H」で切り取って、ペースト

```

8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         Text("Hello, world")
13             .font(.largeTitle)
14         Button("Button") {
15             Action
16         }
17     }
18 }
19
20
21 #Preview {
22     ContentView()
23 }
24

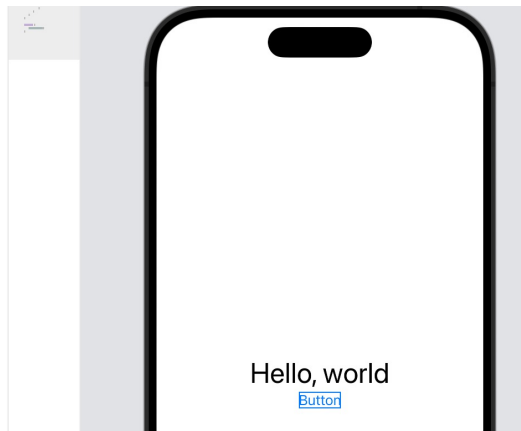
```

⑲ 「Button」を消去

```

7 import SwiftUI
8
9
10 struct ContentView: View {
11     var body: some View {
12
13         VStack {
14             Text("Hello, world")
15                 .font(.largeTitle)
16             Button() {
17                 Action
18             }
19         }
20     }
21 }
22
23 #Preview {
24     ContentView()
25 }
26

```

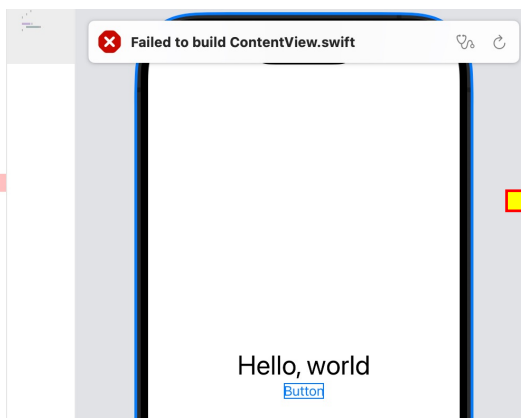
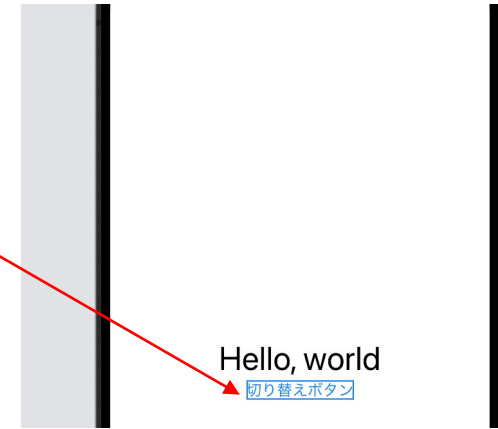


⑳ 「切り替えボタン」と書き込む

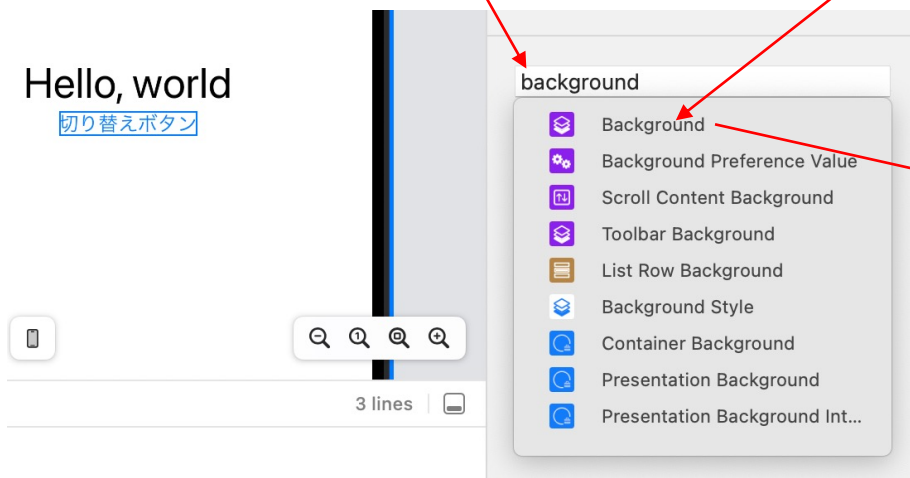
```

8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12
13         VStack {
14             Text("Hello, world")
15                 .font(.largeTitle)
16             Button("切り替えボタン") {
17                 Action
18             }
19         }
20     }
21 }
22
23 #Preview {
24     ContentView()
25 }
26

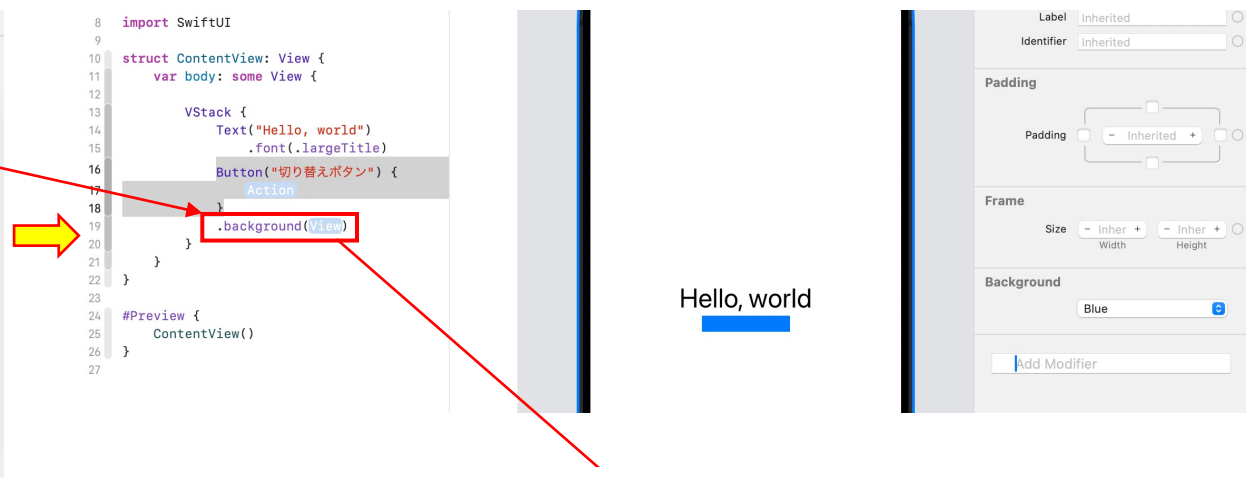
```



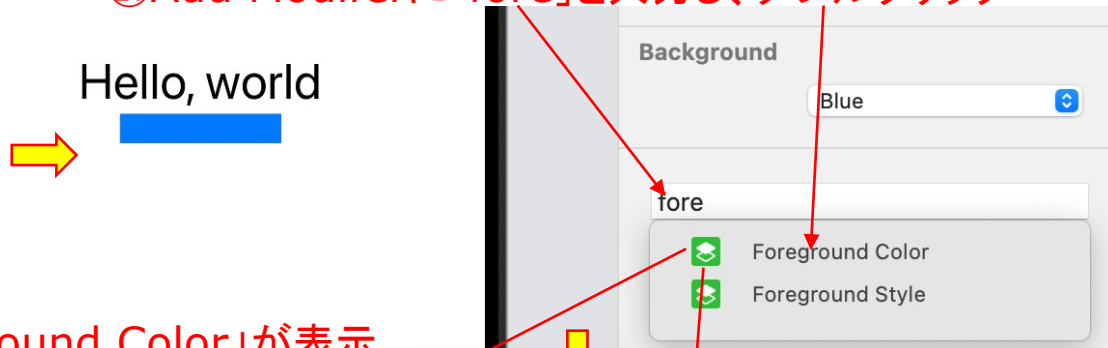
②1 Add Modifierに「background」と入力し、ダブルクリック



②2 デフォルトは「blue」



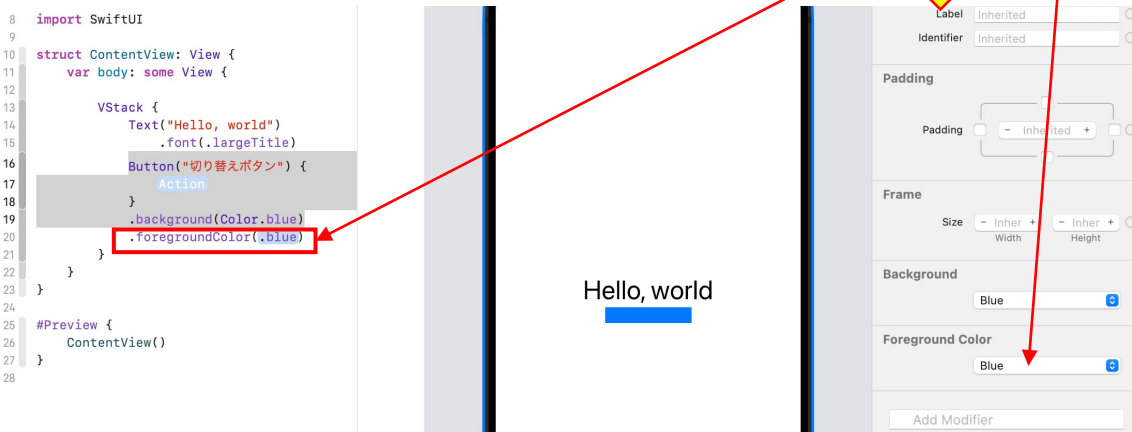
②4 Add Modifierに「fore」と入力し、ダブルクリック

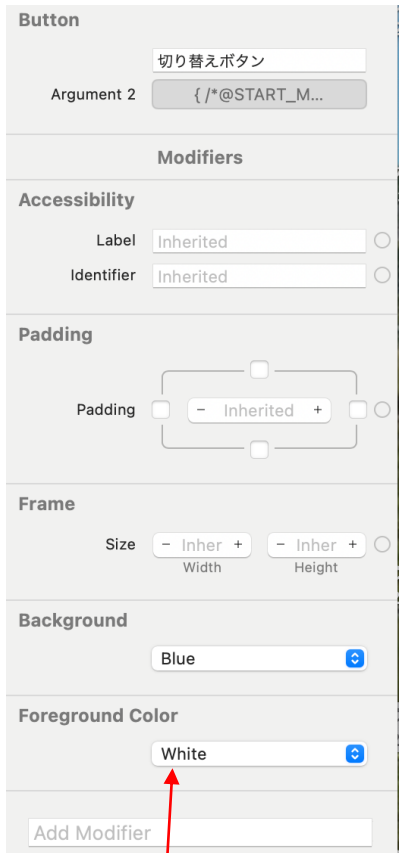


②3 「view」を消して「Color.blue」と書き換える



②5 コードに「Foreground Color」が表示





②⑥ Whiteに書き換え

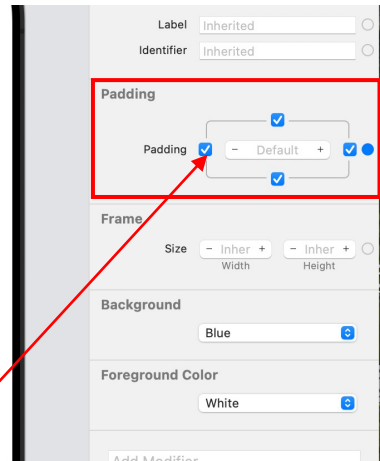
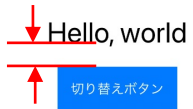
```

7
8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12
13         VStack {
14             Text("Hello, world")
15                 .font(.largeTitle)
16             Button("切り替えボタン") {
17                 Action
18             }
19                 .padding(.all)
20                 .background(Color.blue)
21                 .foregroundColor(.white)
22         }
23     }
24 }
25
26 #Preview {
27     ContentView()
28 }
29

```



間隔が広がる



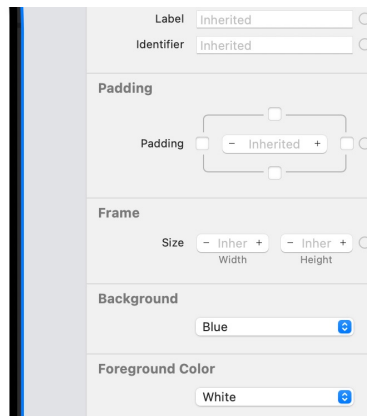
②⑦ Paddingのところ4箇所チェックを入れる



```

8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12
13         VStack {
14             Text("Hello, world")
15                 .font(.largeTitle)
16             Button("切り替えボタン") {
17                 Action
18             }
19                 .background(Color.blue)
20                 .foregroundColor(.white)
21         }
22     }
23 }
24
25 #Preview {
26     ContentView()
27 }
28

```



②⑧ 追記 → 「outputText」に置き換え

```

8 import SwiftUI
9
10 struct ContentView: View {
11     @State var outputText="Hello, world"
12
13     var body: some View {
14
15         VStack {
16             Text(outputText)
17                 .font(.largeTitle)
18             Button("切り替えボタン") {
19                 Action
20             }
21                 .padding(.all)
22                 .background(Color.blue)
23                 .foregroundColor(.white)
24         }
25     }
26 }

```

②⑨ 1行空ける

```

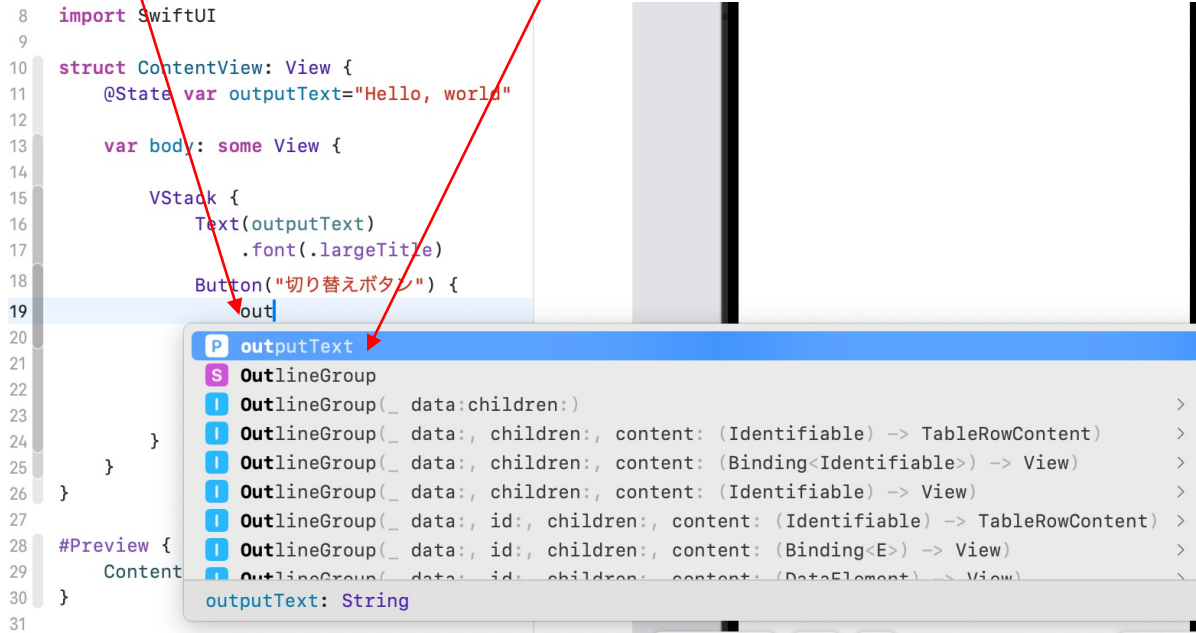
8 import SwiftUI
9
10 struct ContentView: View {
11     @State var outputText="Hello, world"
12
13     var body: some View {
14
15         VStack {
16             Text(outputText)
17                 .font(.largeTitle)
18             Button("切り替えボタン") {
19
20             }
21                 .padding(.all)
22                 .background(Color.blue)
23                 .foregroundColor(.white)
24         }
25     }
26 }

```



③⑩「out」と書き込み、「outputText」をダブルクリック

```
8 import SwiftUI
9
10 struct ContentView: View {
11     @State var outputText="Hello, world"
12
13     var body: some View {
14
15         VStack {
16             Text(outputText)
17                 .font(.largeTitle)
18             Button("切り替えボタン") {
19                 out
20             }
21         }
22     }
23 }
24
25 #Preview {
26     ContentView()
27 }
28
29 Content
30
31
```

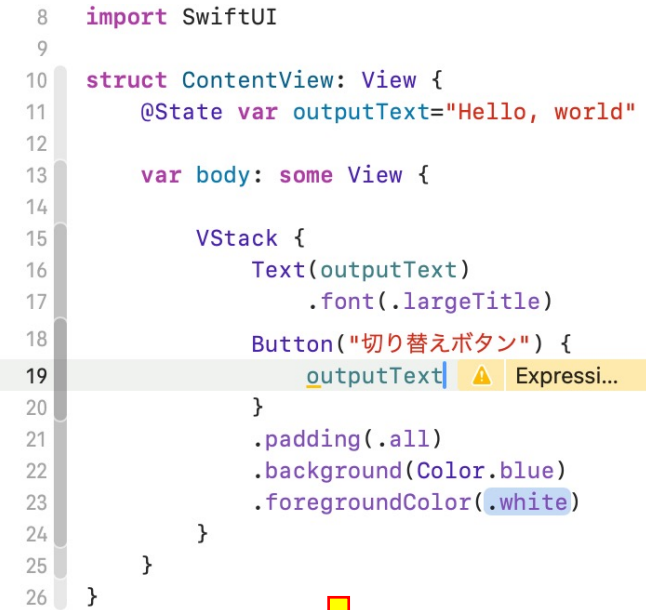


The screenshot shows the Xcode editor with a search popup for 'outputText'. The popup lists several 'OutlineGroup' items, with 'outputText: String' at the bottom. Red arrows point from the text '③⑩「out」と書き込み、「outputText」をダブルクリック' to the 'out' text in the code and the 'outputText' entry in the popup.

VStack



```
8 import SwiftUI
9
10 struct ContentView: View {
11     @State var outputText="Hello, world"
12
13     var body: some View {
14
15         VStack {
16             Text(outputText)
17                 .font(.largeTitle)
18             Button("切り替えボタン") {
19                 outputText
20             }
21         }
22     }
23 }
24
25 #Preview {
26     ContentView()
27 }
28
29 Content
30
31
```



The screenshot shows the Xcode editor with the code from the previous step. The text 'outputText' in the Button's action block is highlighted in yellow. A yellow arrow points from the previous screenshot to this one.



③⑪「="Hi, Swift!"」を追記

```
8 import SwiftUI
9
10 struct ContentView: View {
11     @State var outputText="Hello, world"
12
13     var body: some View {
14
15         VStack {
16             Text(outputText)
17                 .font(.largeTitle)
18             Button("切り替えボタン") {
19                 outputText="Hi, Swift!"
20             }
21         }
22     }
23 }
24
25 #Preview {
26     ContentView()
27 }
28
29 Content
30
31
```



The screenshot shows the Xcode editor with the code from the previous step. The text 'outputText="Hi, Swift!"' has been added to the Button's action block. A red arrow points from the text '③⑪「="Hi, Swift!"」を追記' to the new code.