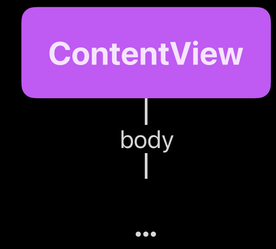
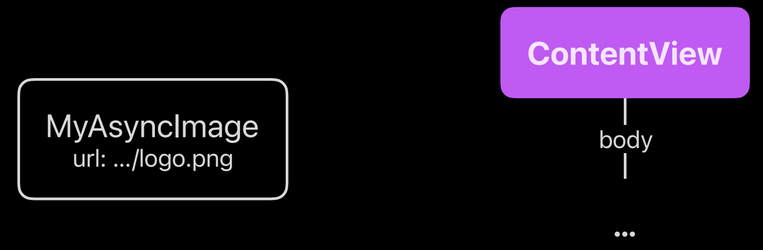
```
struct ContentView: View {
    var body: some View {
        MyAsyncImage(url: URL(string: "https://www.objc.io/logo.png")!)
    }
}
```

```
0  struct ContentView: View {
1      var body: some View {
2          MyAsyncImage(url: URL(string: "https://www.objc.io/logo.png")!)
3      }
4  }
```

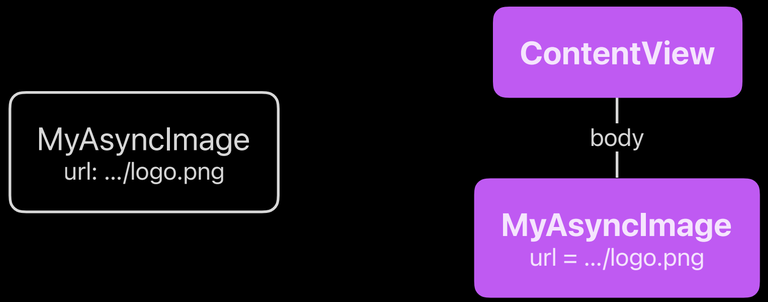
```
0  struct ContentView: View {
1      var body: some View {
2          MyAsyncImage(url: URL(string: "https://www.objc.io/logo.png")!)
3      }
4  }
```

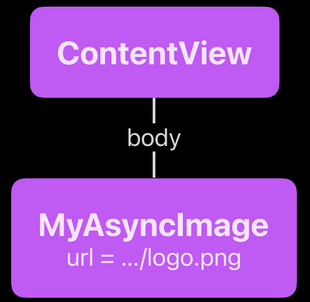


```
0  struct ContentView: View {
1      var body: some View {
2          MyAsyncImage(url: URL(string: "https://www.objc.io/logo.png")!)
3      }
4  }
```

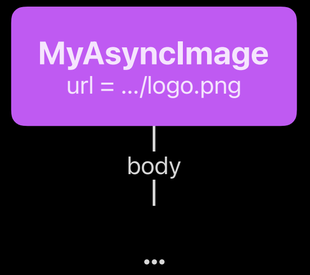


```swift
struct ContentView: View {
    var body: some View {
        MyAsyncImage(url: URL(string: "https://www.objc.io/logo.png")!)
    }
}
```

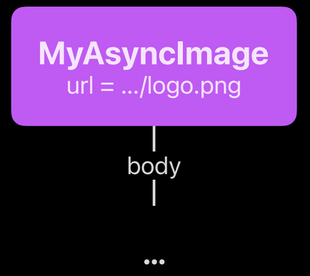



ContentView
body
MyAsyncImage
url = .../logo.png

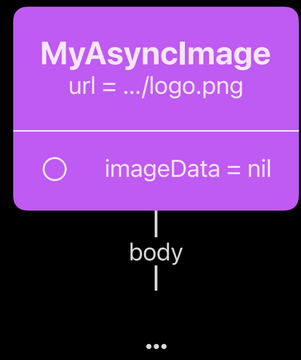
```
0  struct MyAsyncImage: View {
1      var url: URL
2      var body: some View {
3          // …
4      }
5  }
```



```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        // …
    }
}
```



```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        // …
    }
}
```



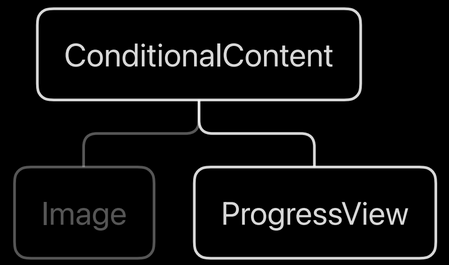
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        // …
    }
}
```

```
0  struct MyAsyncImage: View {
1      var url: URL
2      @State private var imageData: Data? = nil
3
4      var body: some View {
5          Image(nsImage: NSImage(data: imageData!)!)
6      }
7  }
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        Image(nsImage: NSImage(data: imageData!)!)
    }
}
```

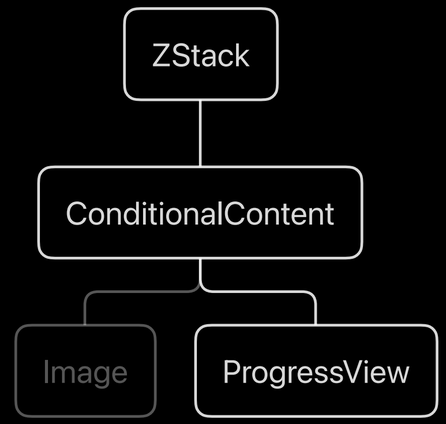
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        if let d = imageData, let i = NSImage(data: d) {
            Image(nsImage: i)
        } else {
            ProgressView()
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        if let d = imageData, let i = NSImage(data: d) {
            Image(nsImage: i)
        } else {
            ProgressView()
        }
    }
}
```



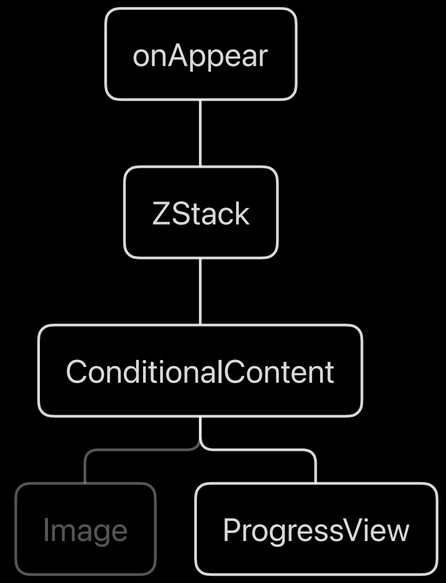
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }
    }
}
```



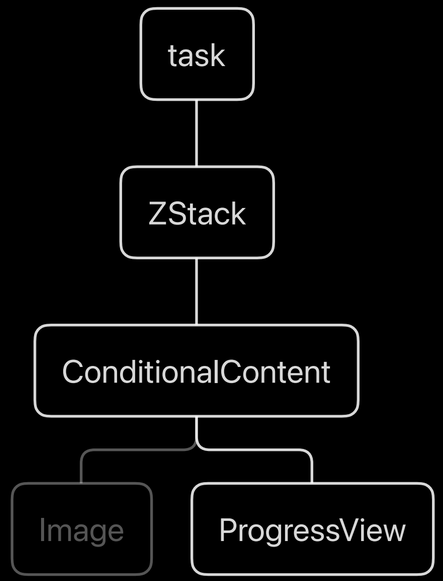
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.onAppear {
            // start loading...
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.onAppear {
            // start loading...
        }
    }
}
```

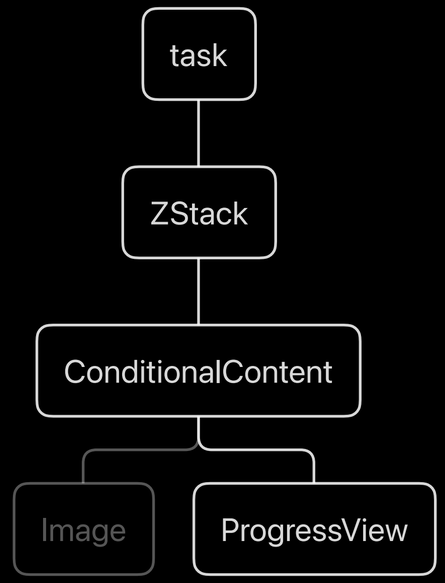


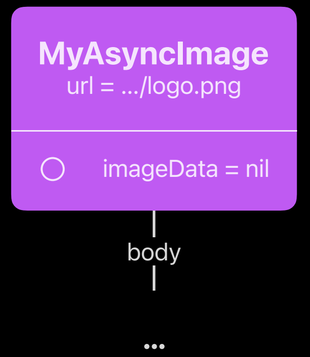
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

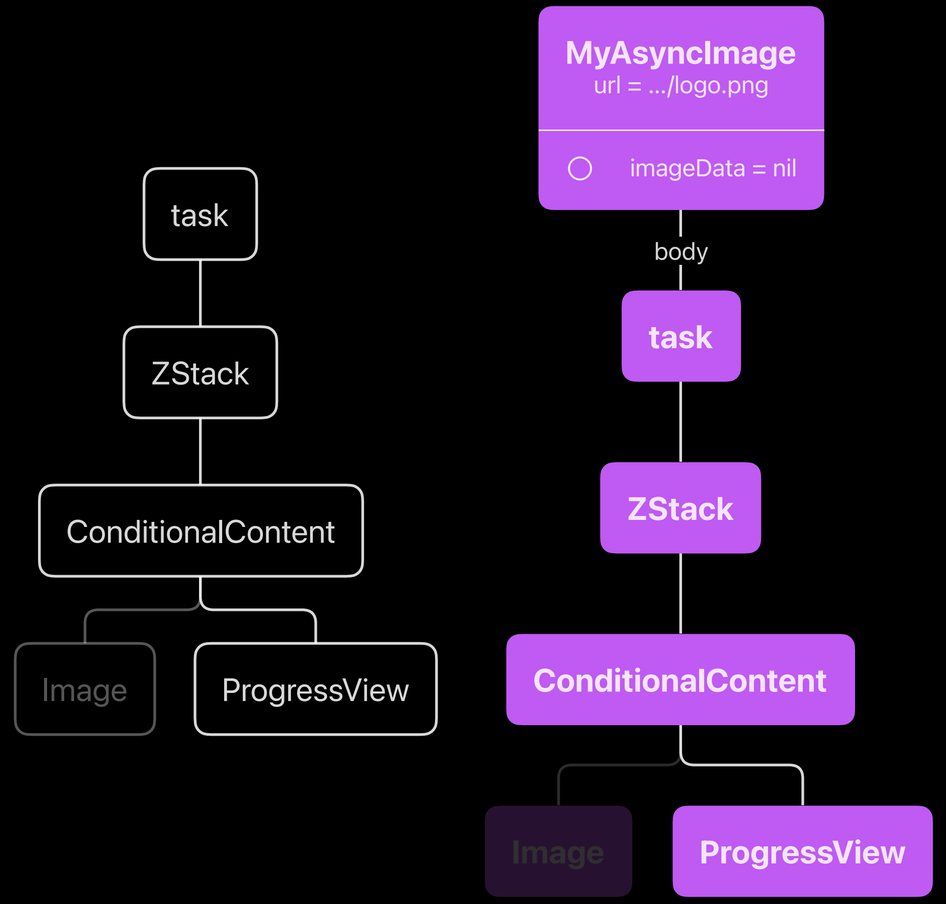


```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

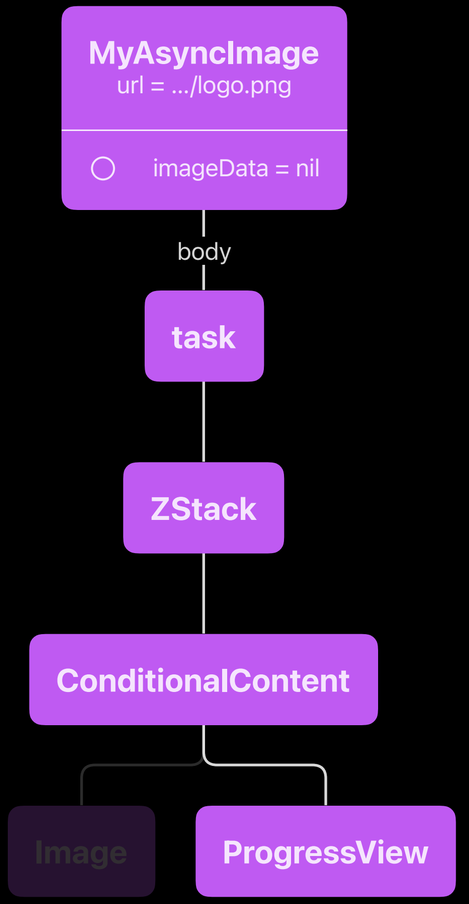




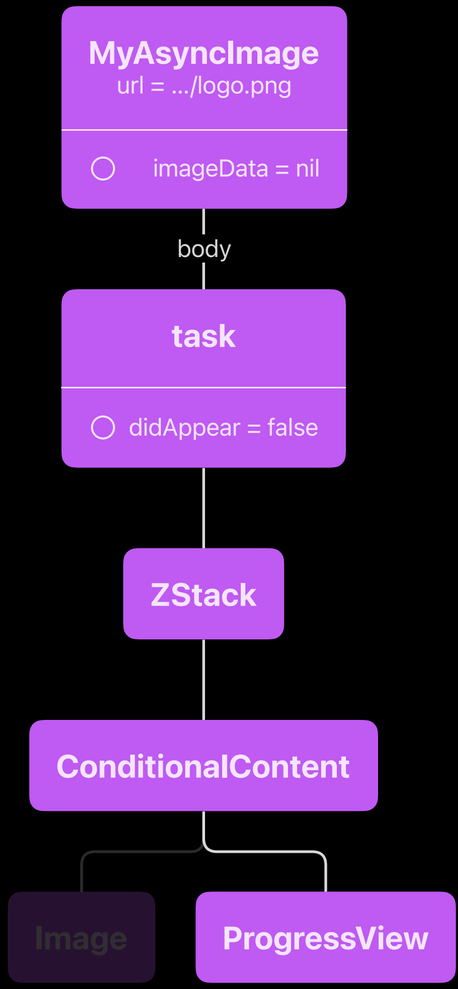
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



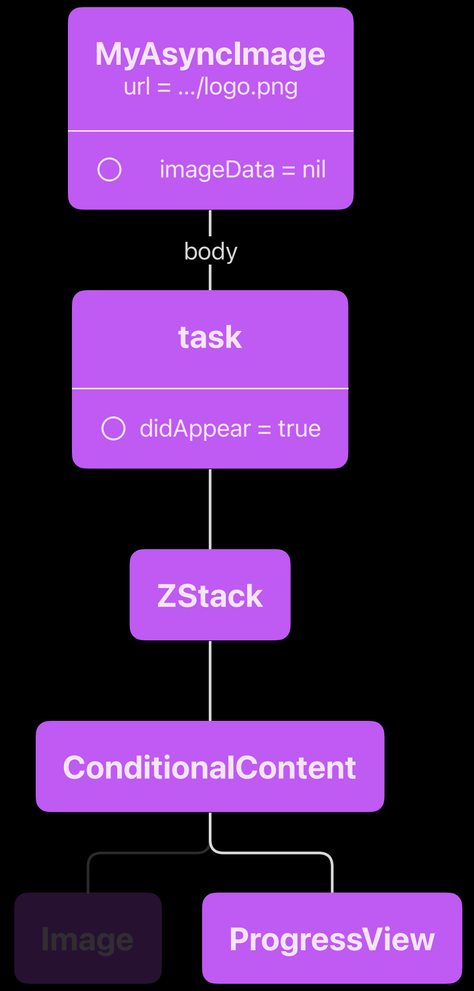
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

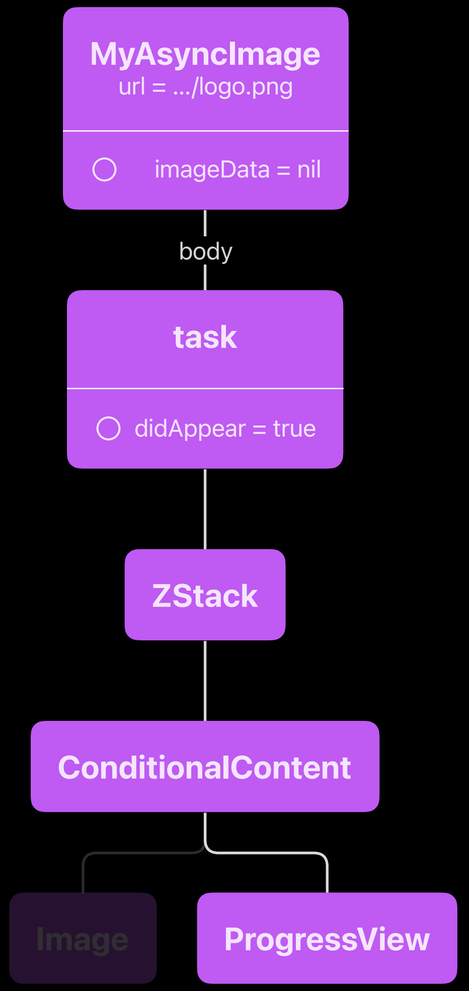


```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



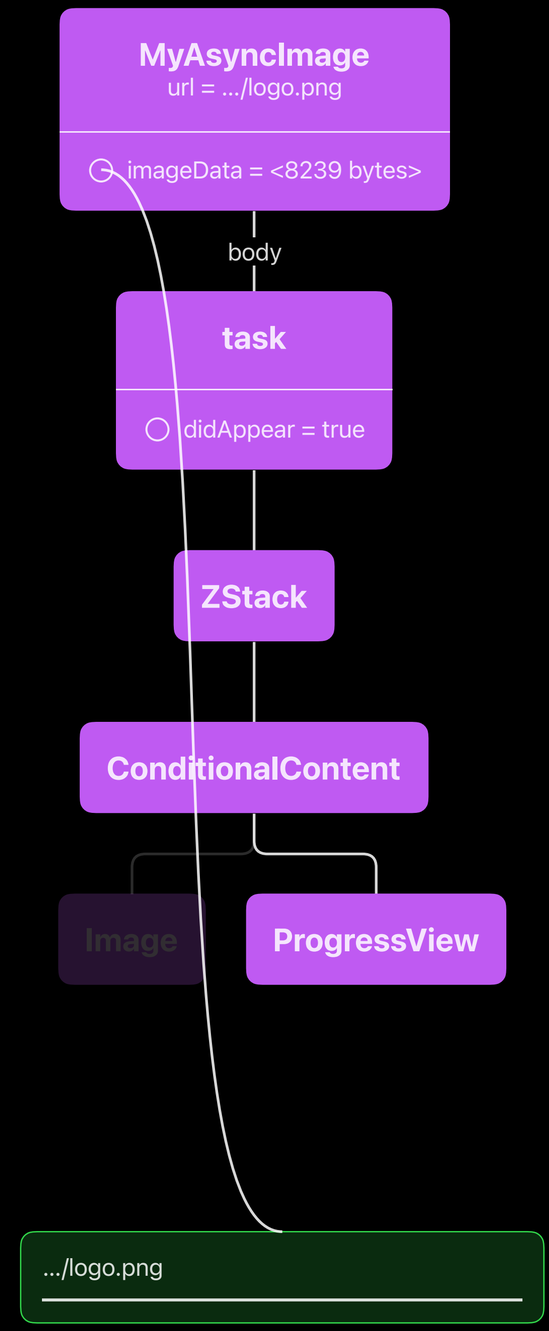
.../logo.png

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

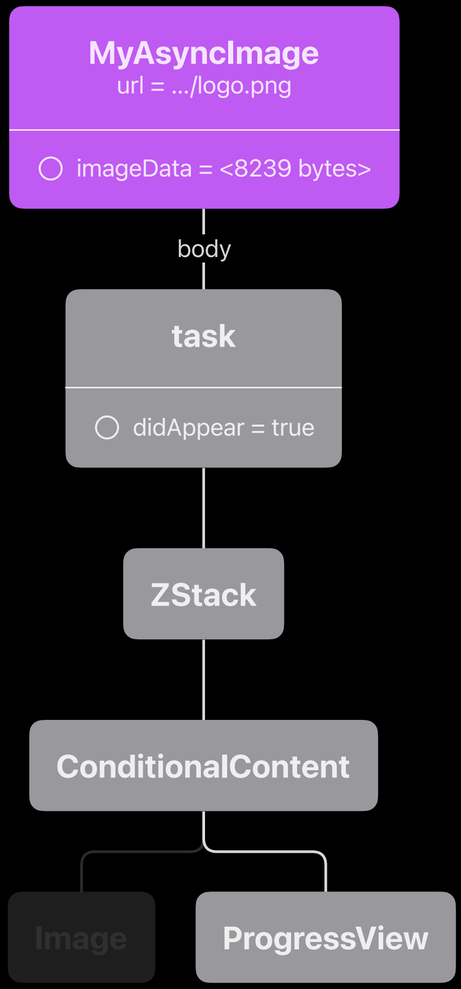


.../logo.png

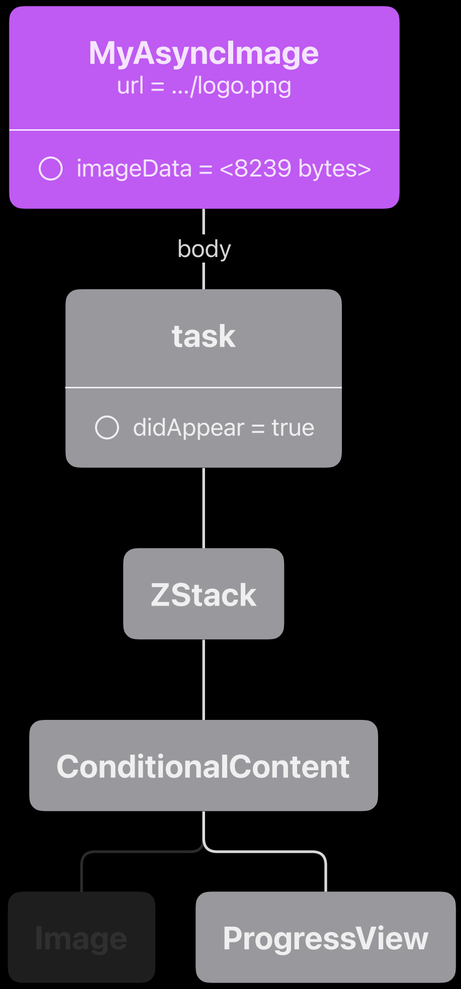
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



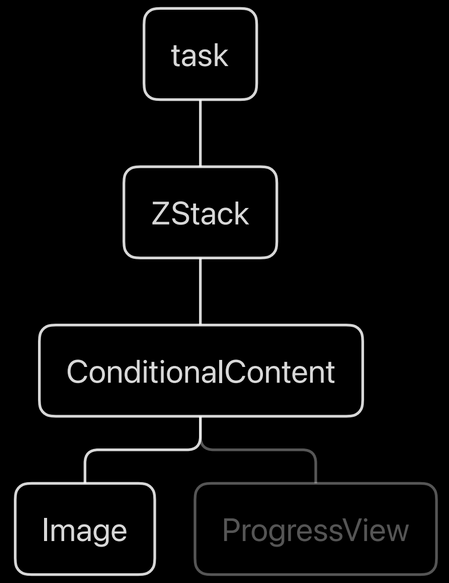
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

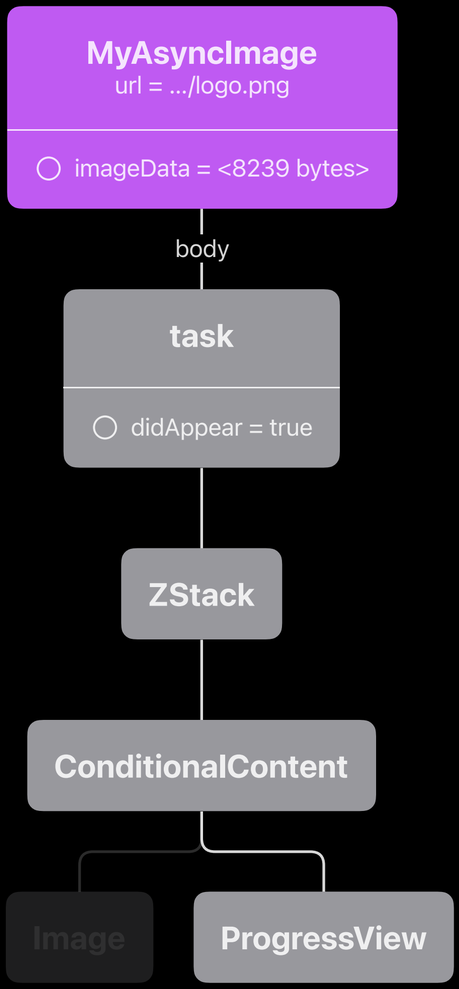


```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

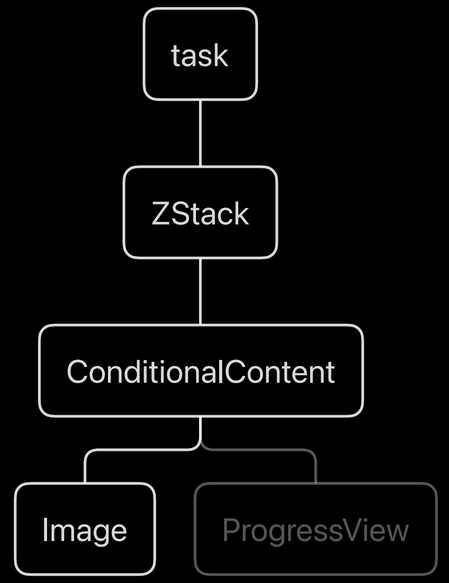


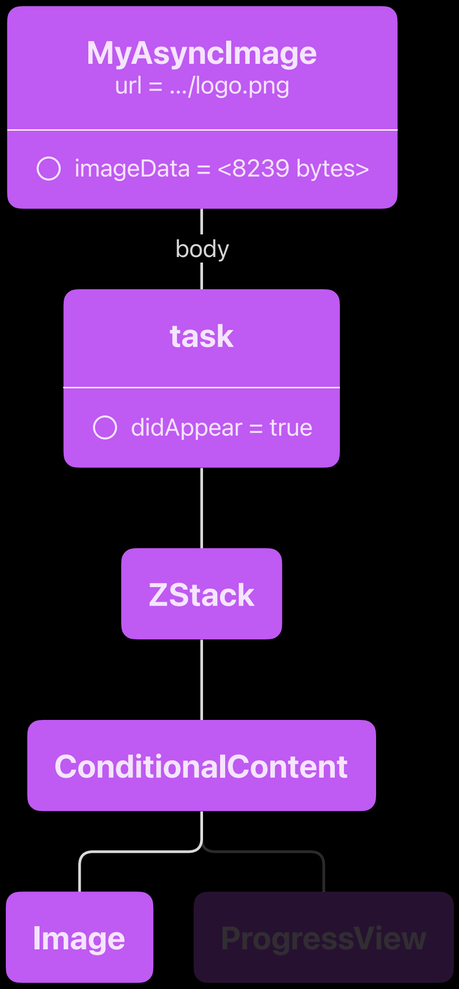
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



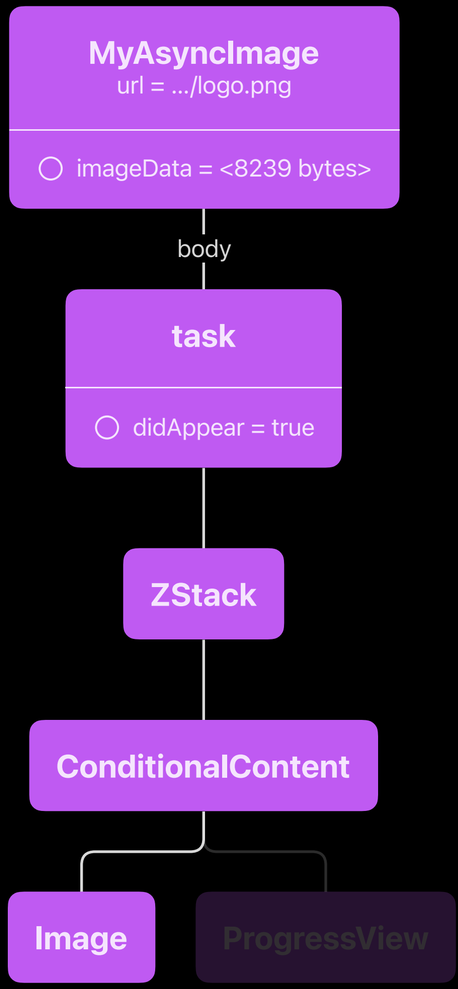


```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```





```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



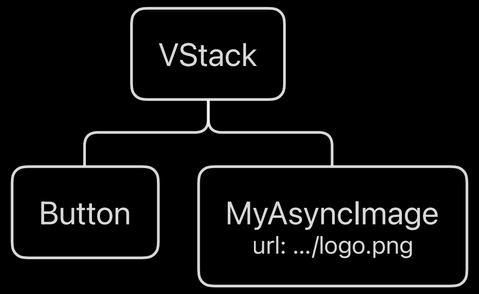
```
struct ContentView: View {
    var body: some View {
        MyAsyncImage(url: logo)
    }
}
```

```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        MyAsyncImage(url: logo)
    }
}
```

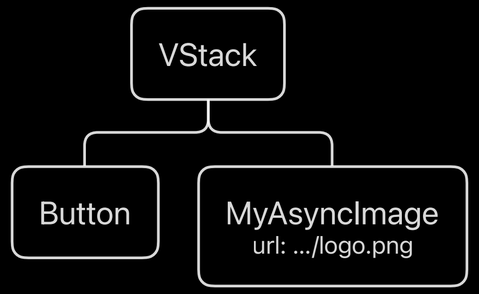
```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            MyAsyncImage(url: logo)
        }
    }
}
```

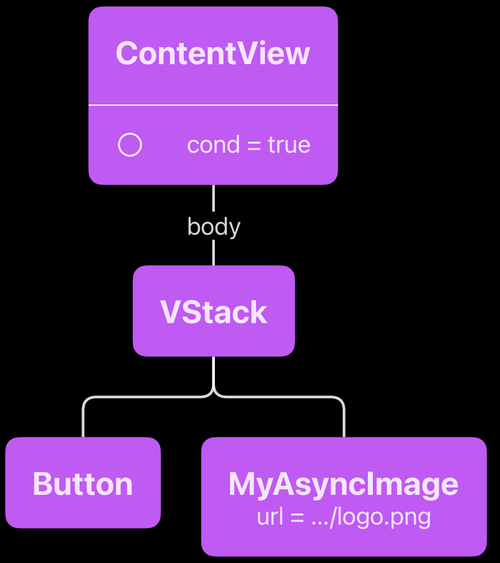
```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```

```swift
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```

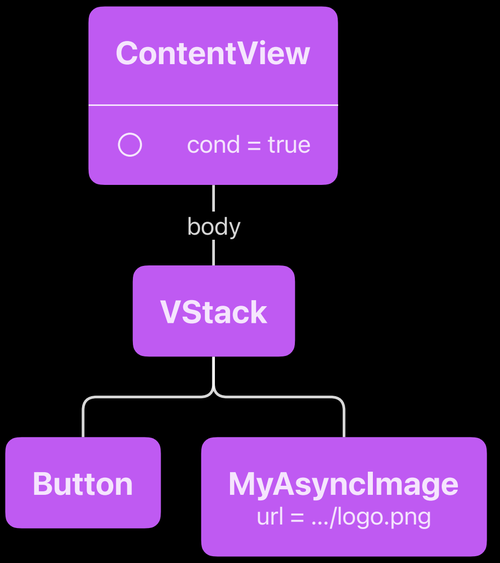


```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```

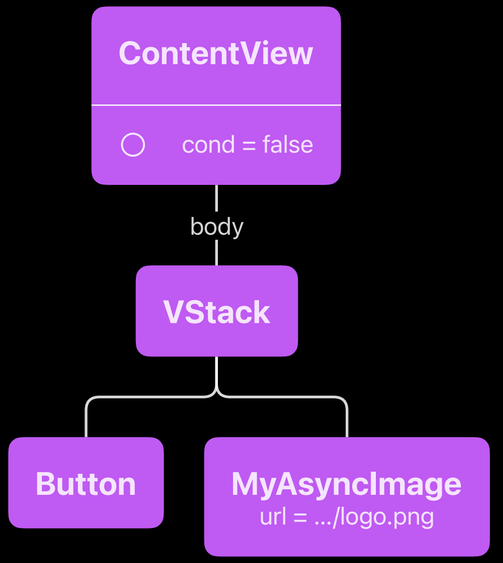




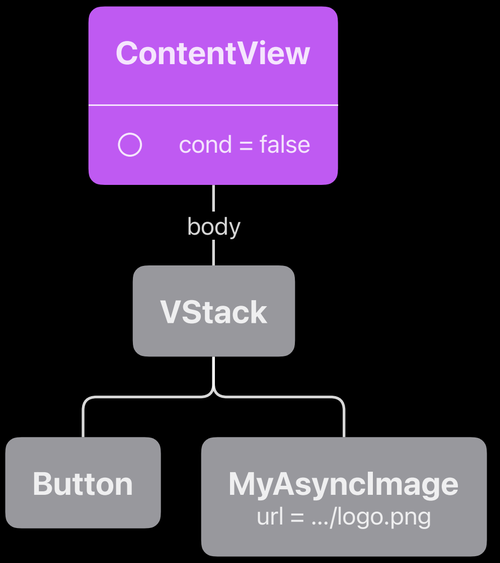
```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```



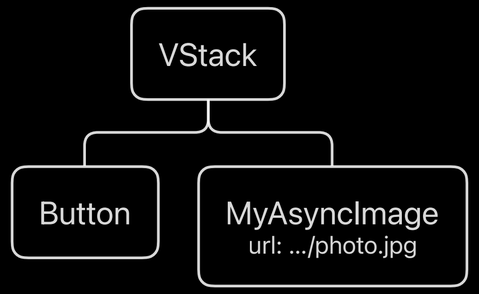
```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```

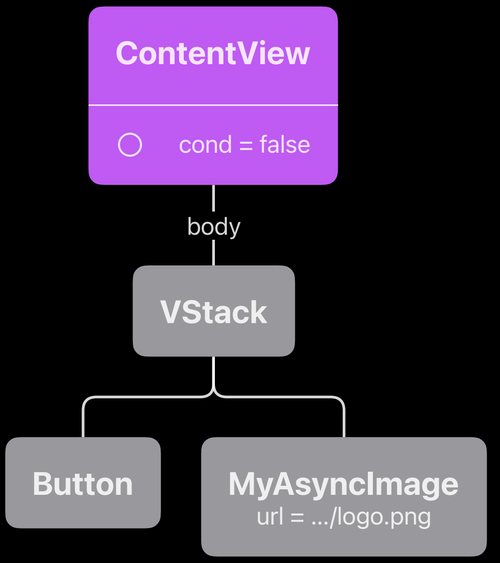


```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```

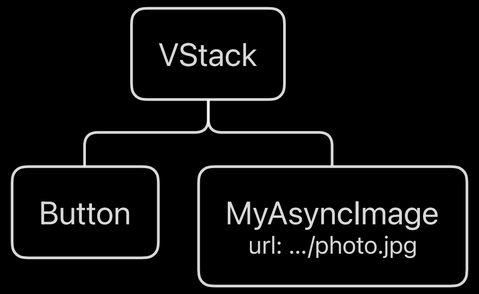


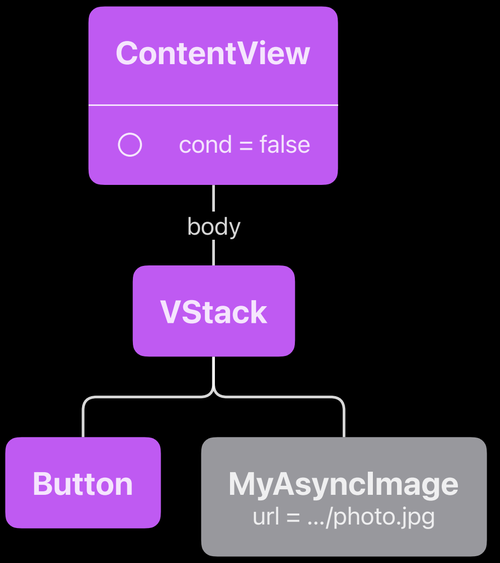
```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```

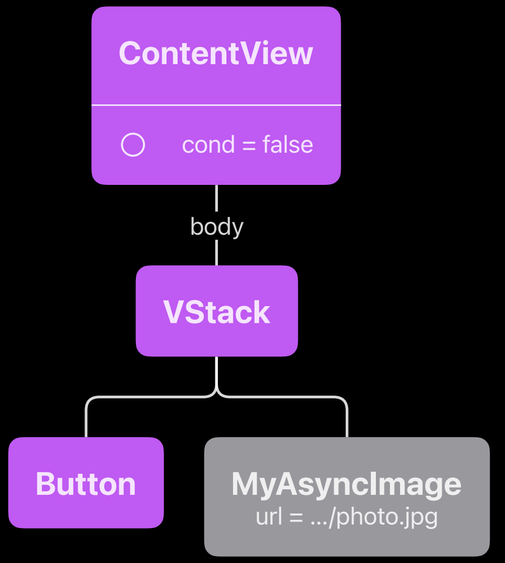




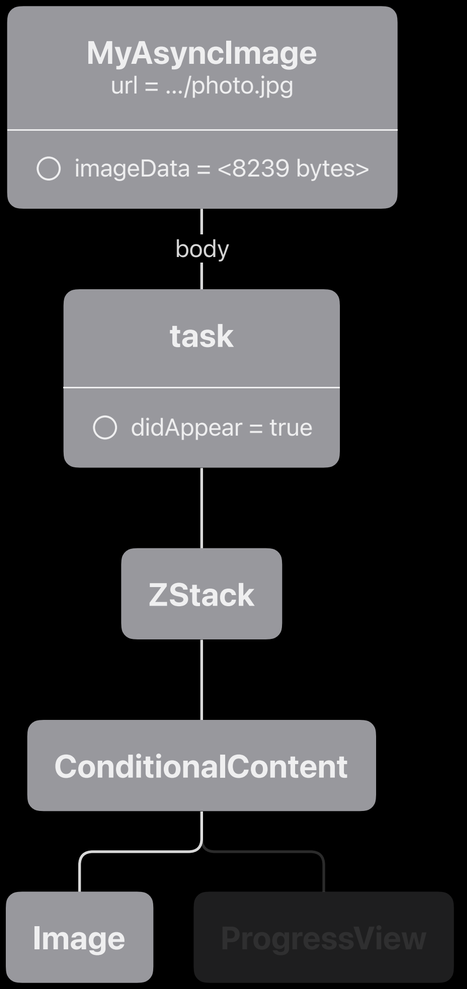
```
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```





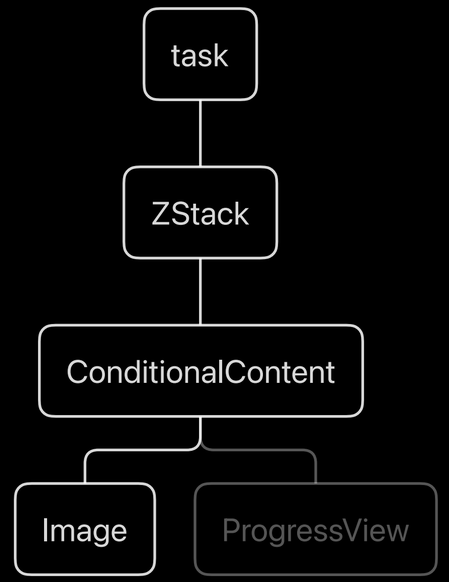

ContentView
cond = false
body
VStack
Button
MyAsyncImage
url = .../photo.jpg

```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

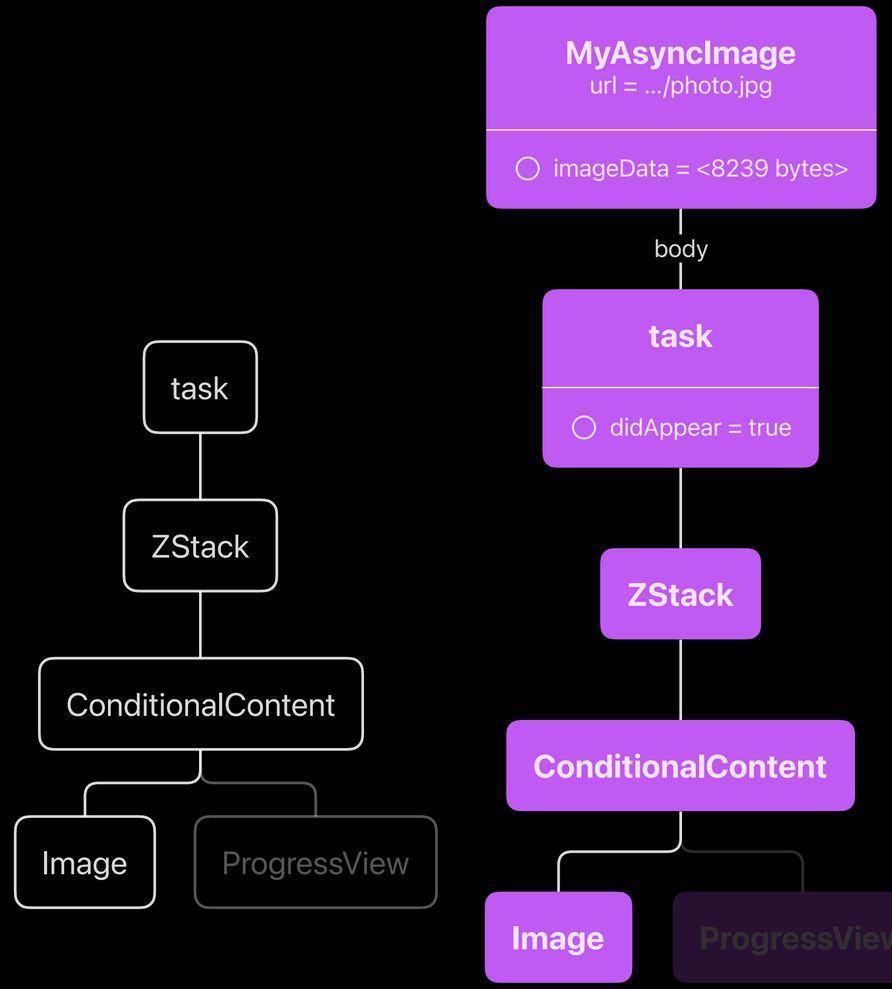


```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```



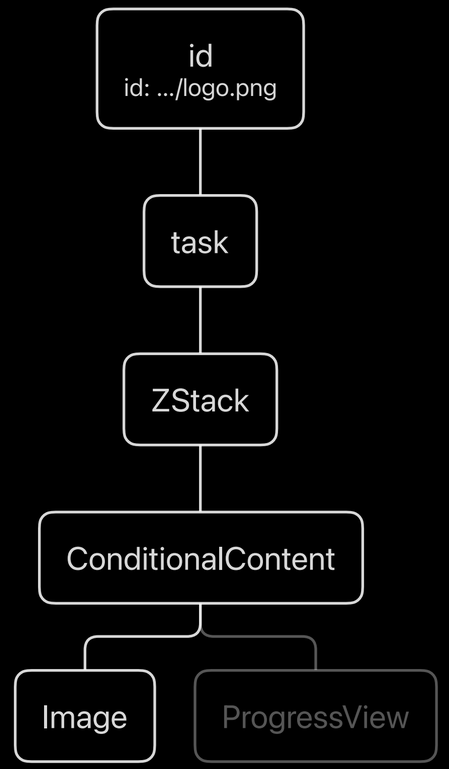
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

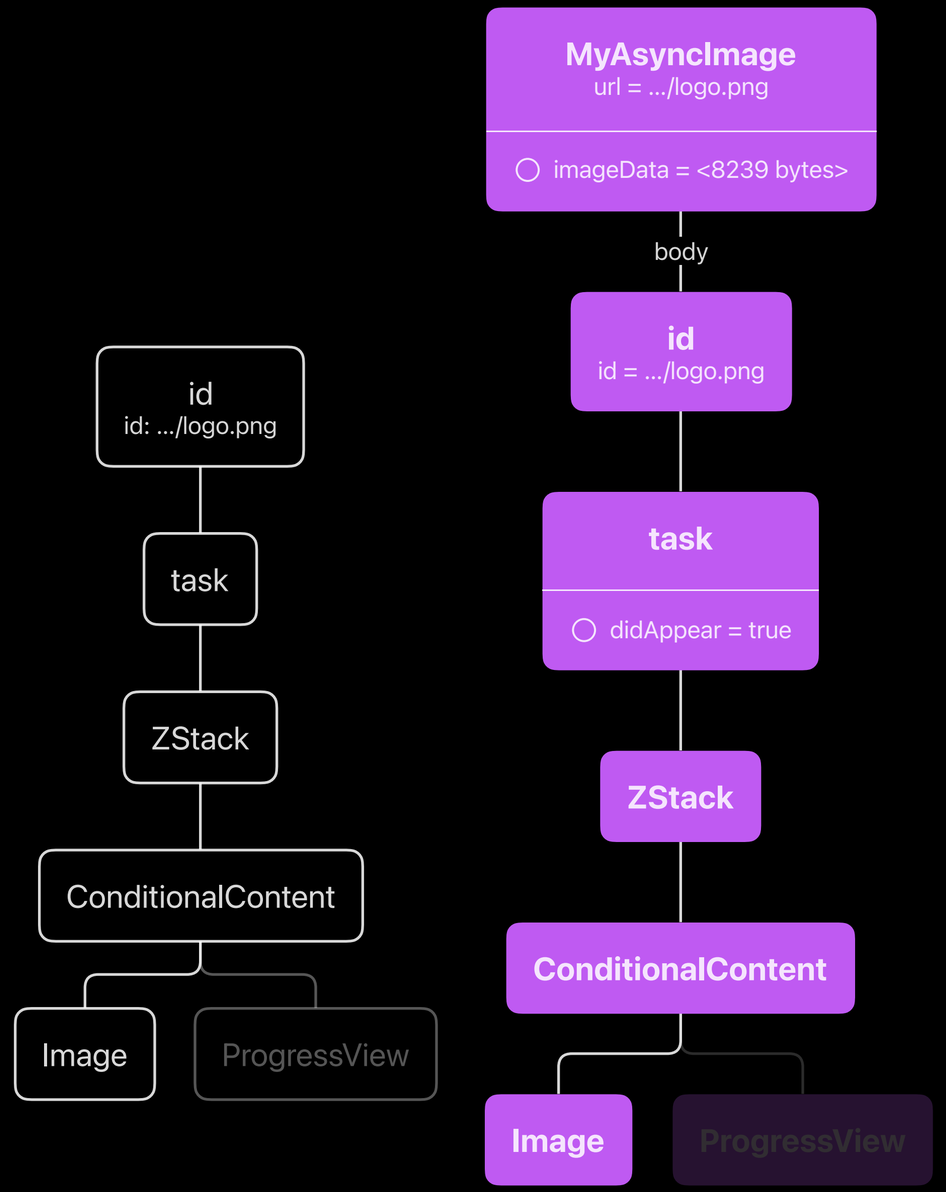
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```

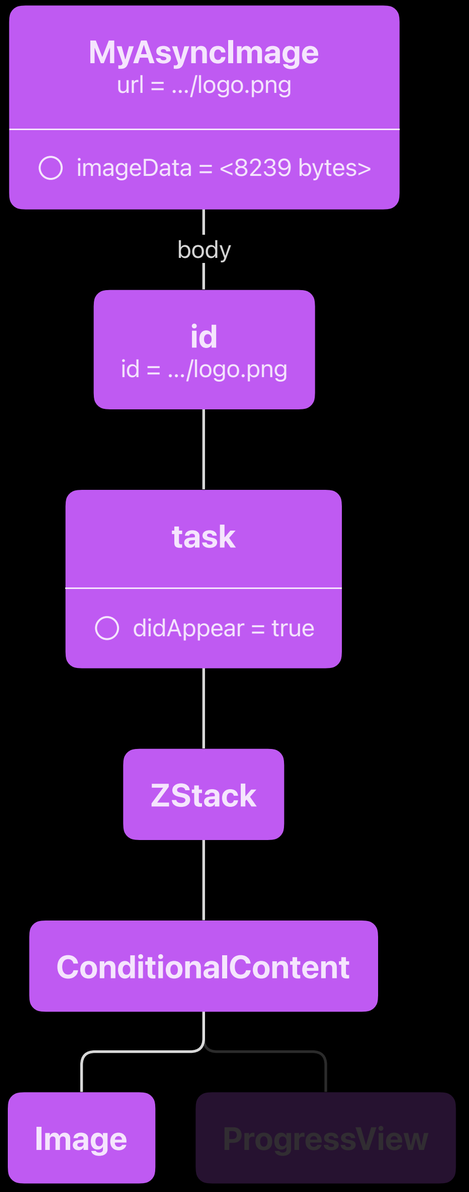
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



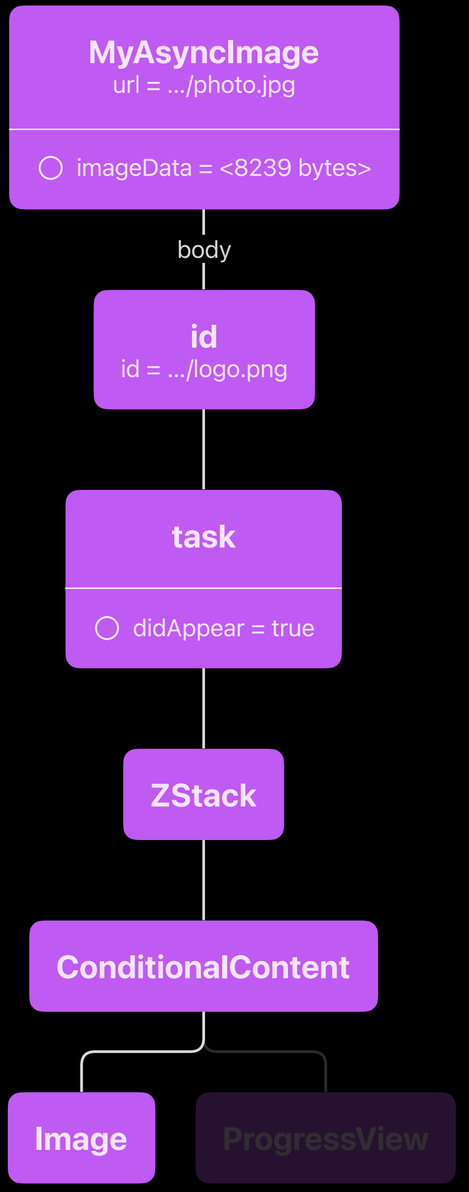
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



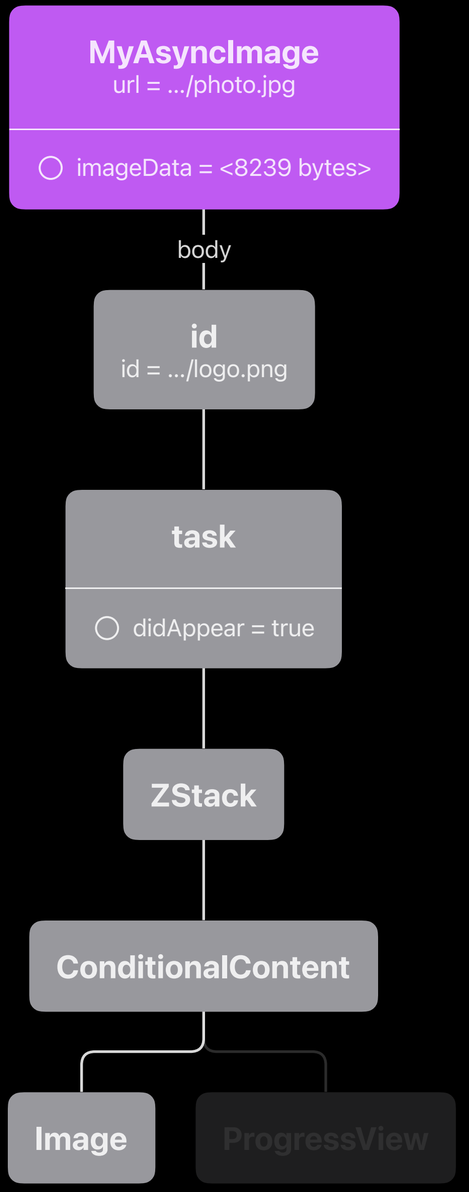
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



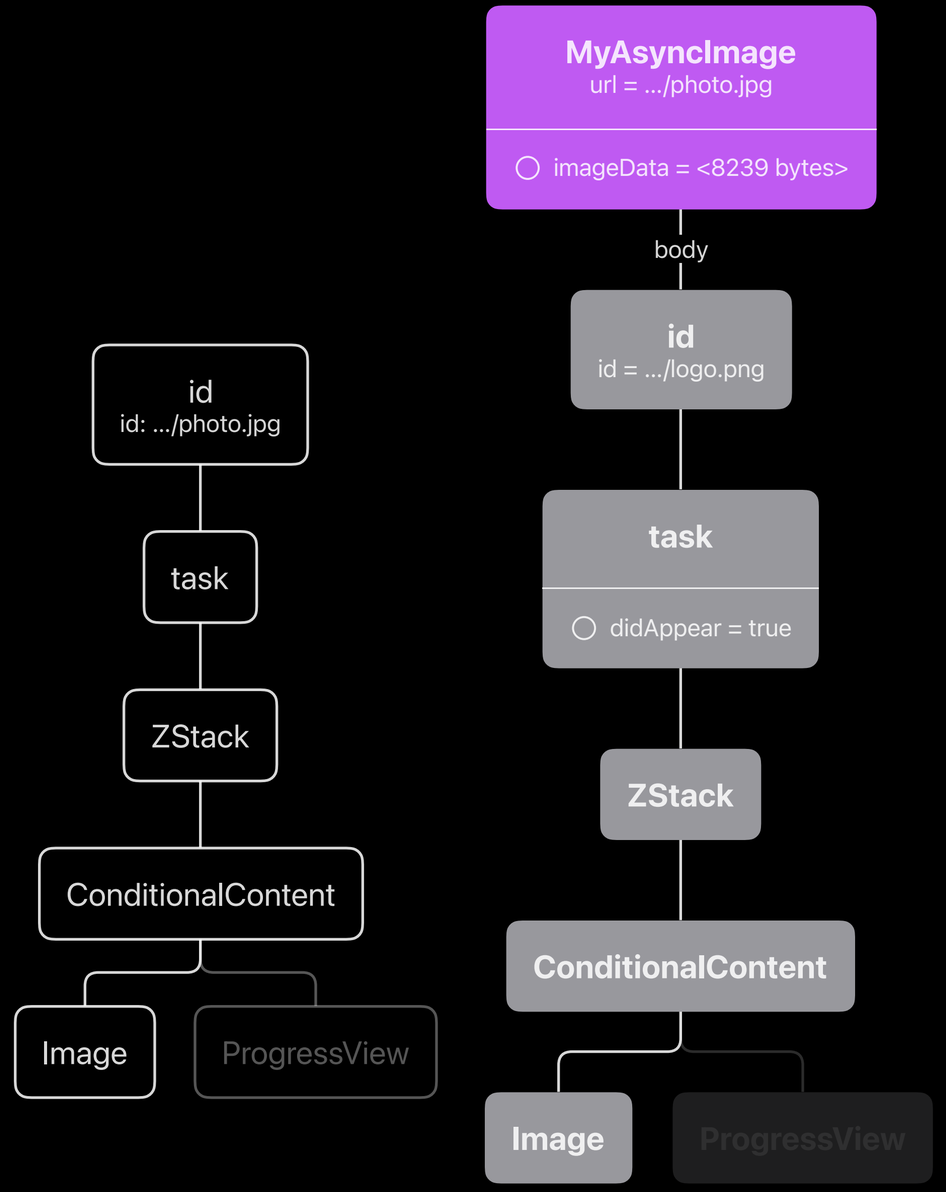
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



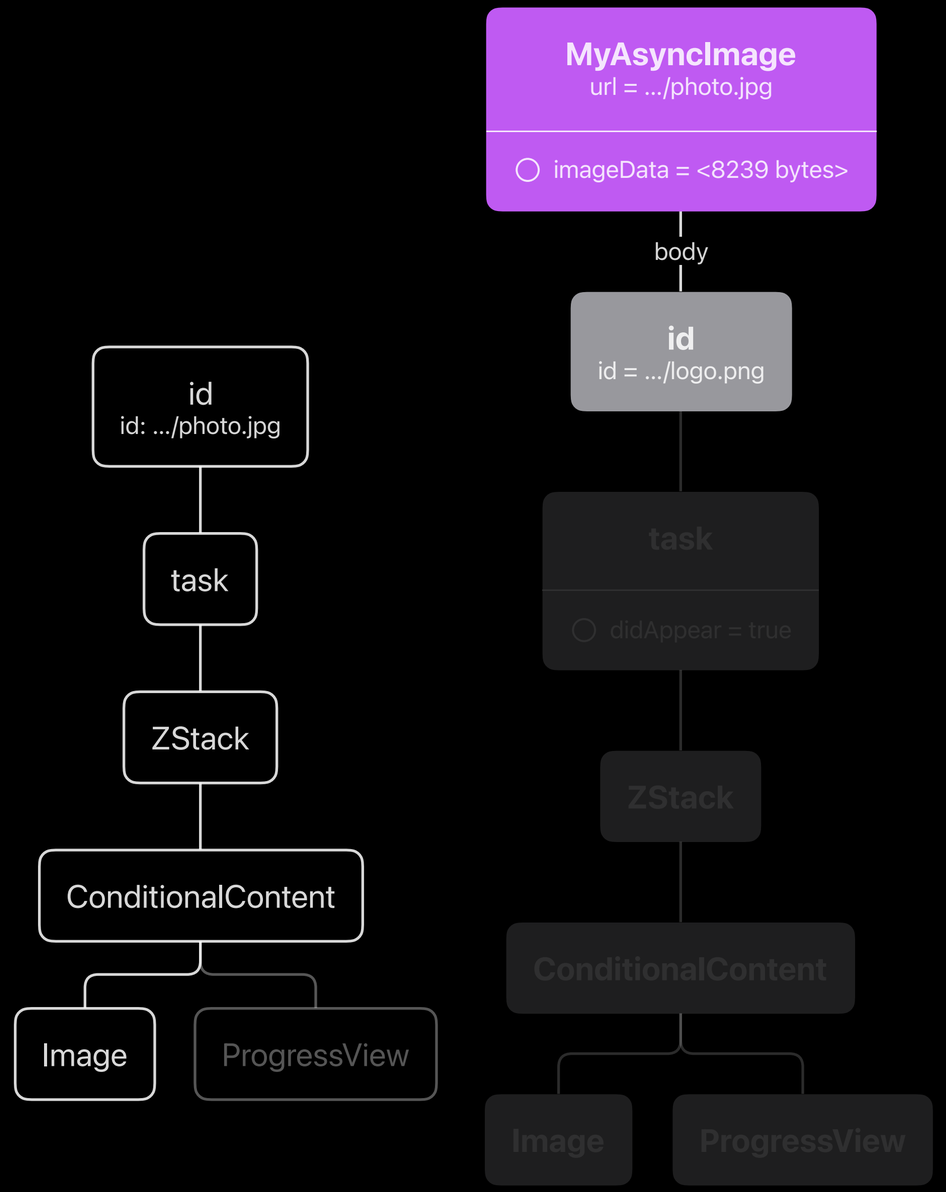
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



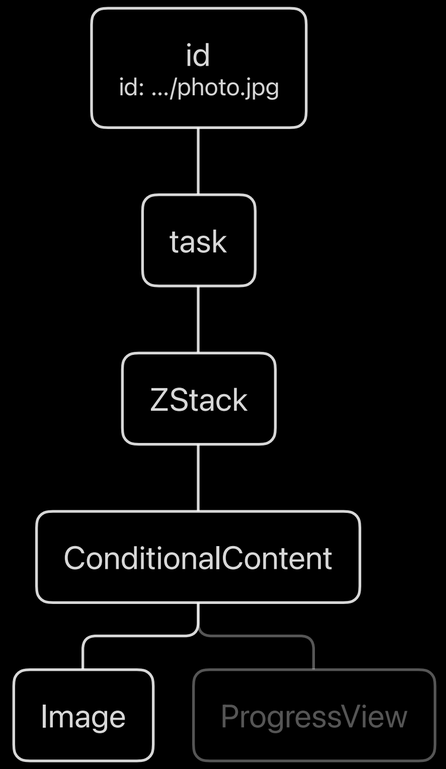
```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```

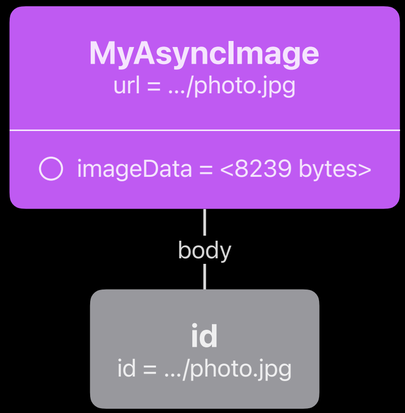


```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```

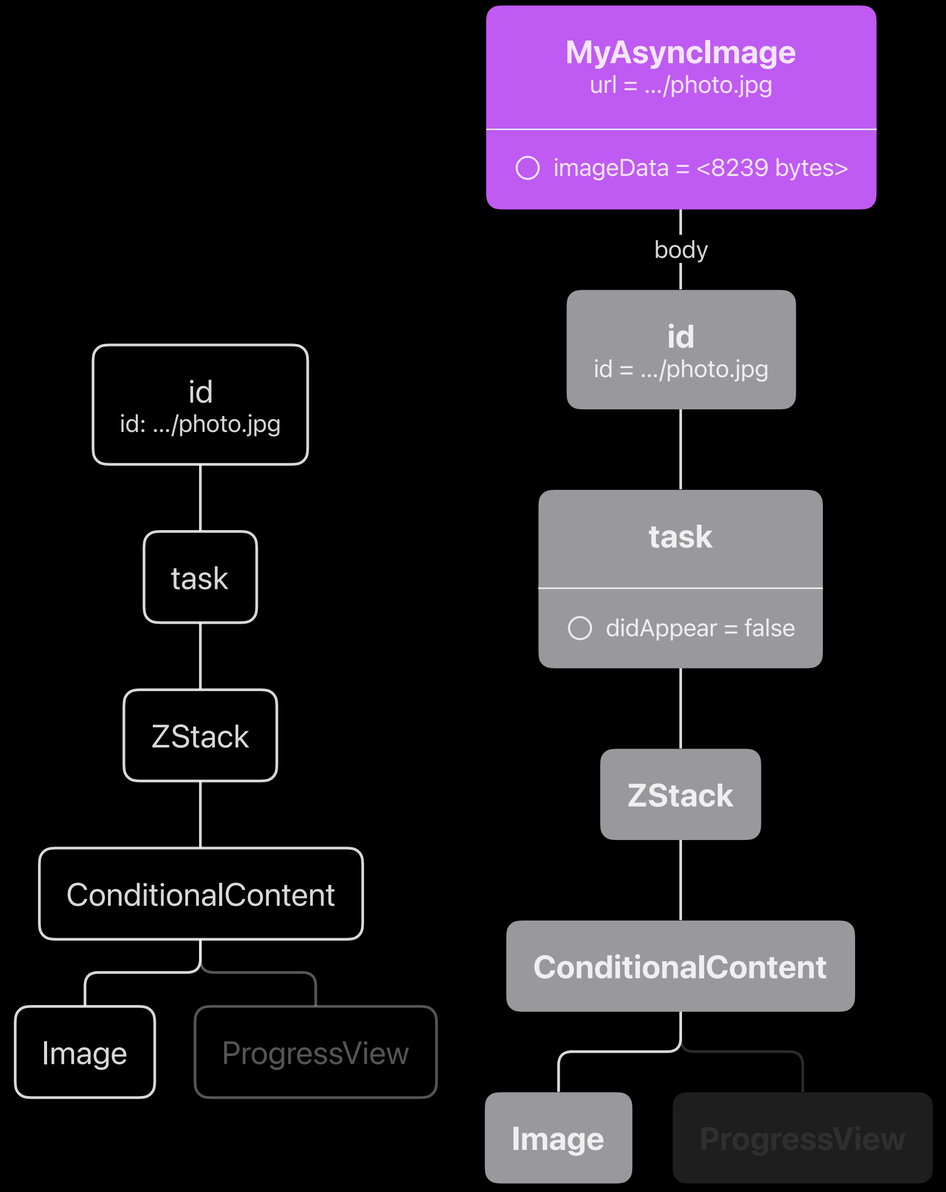


```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```

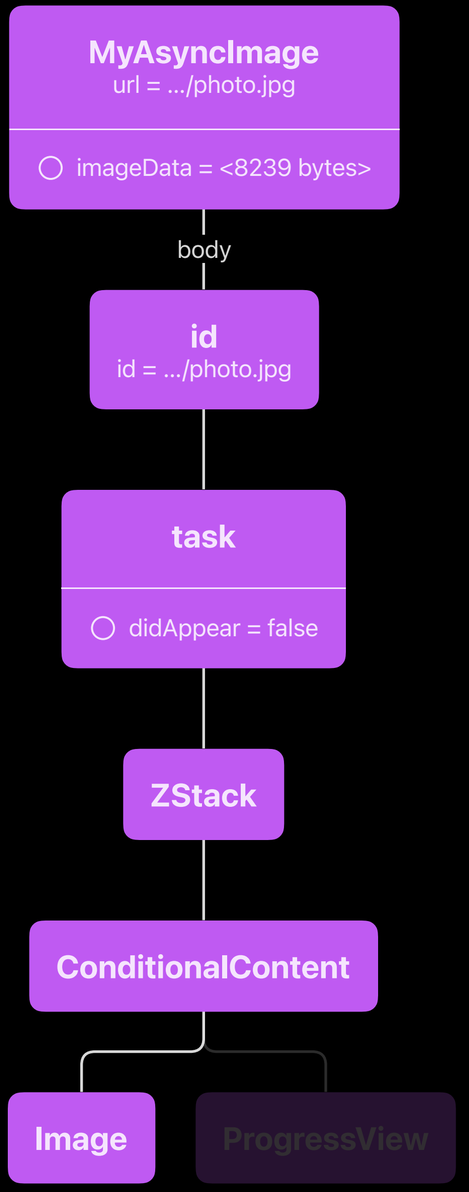




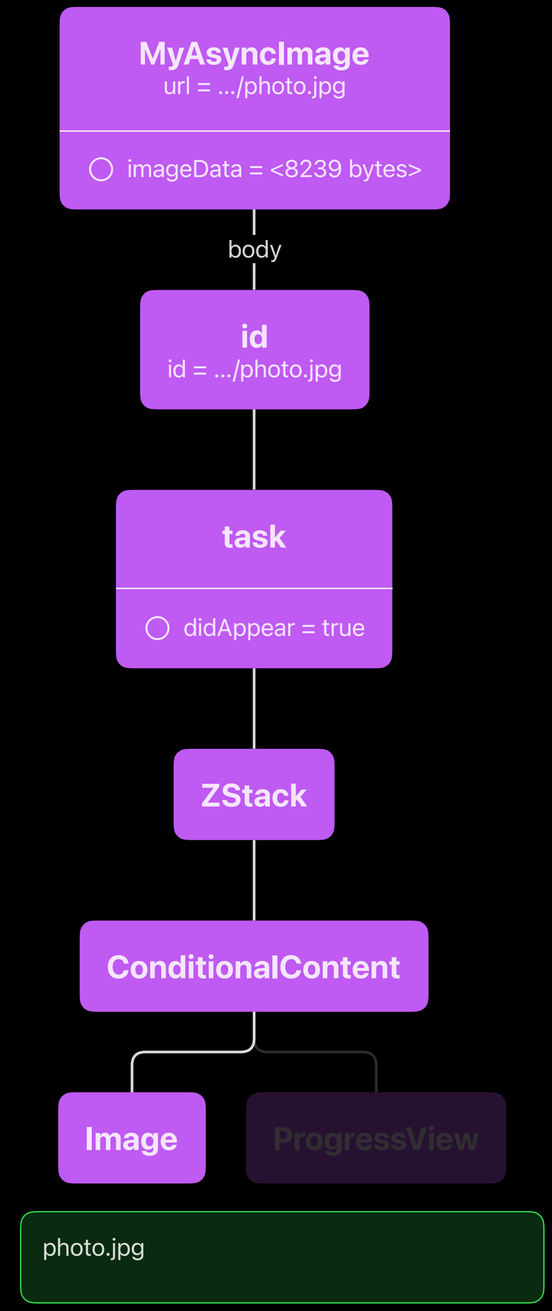
```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



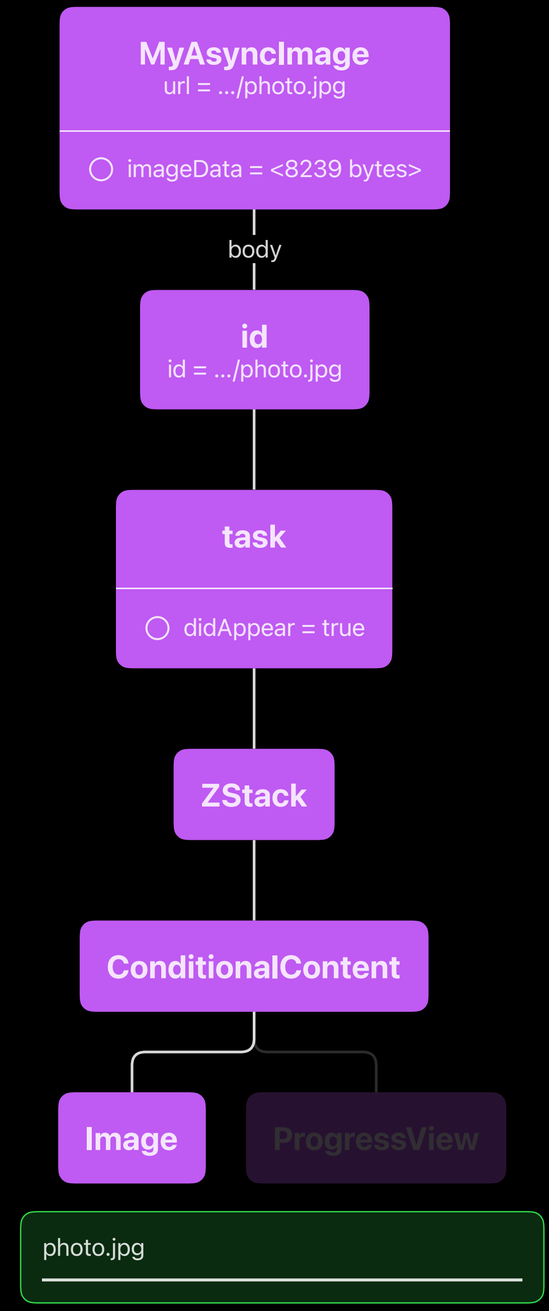
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



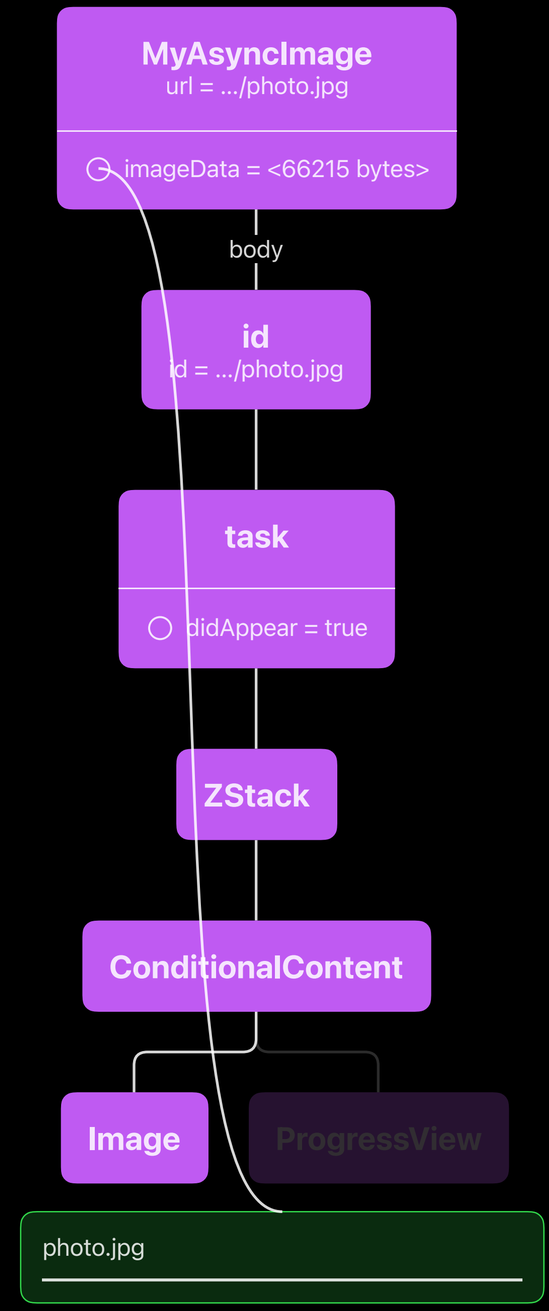
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



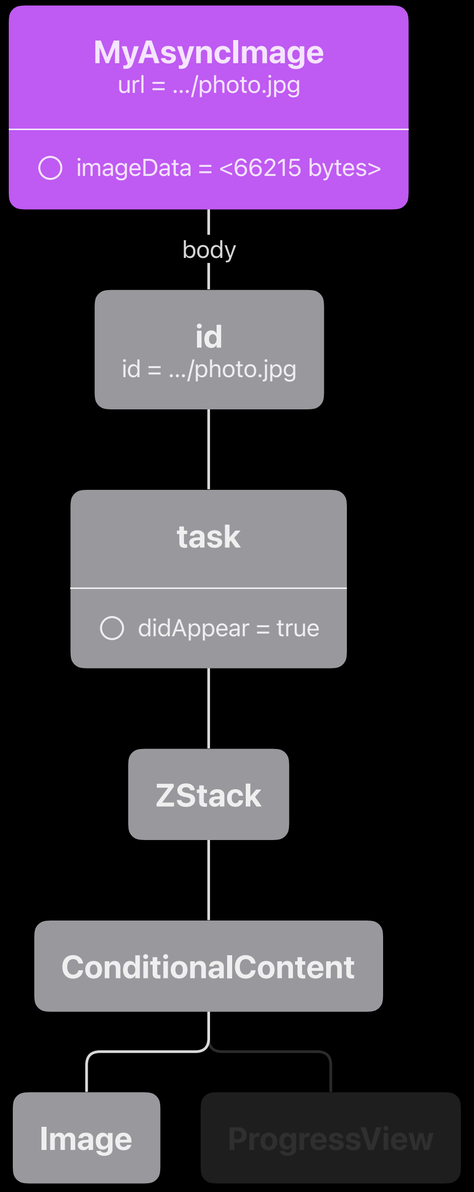
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



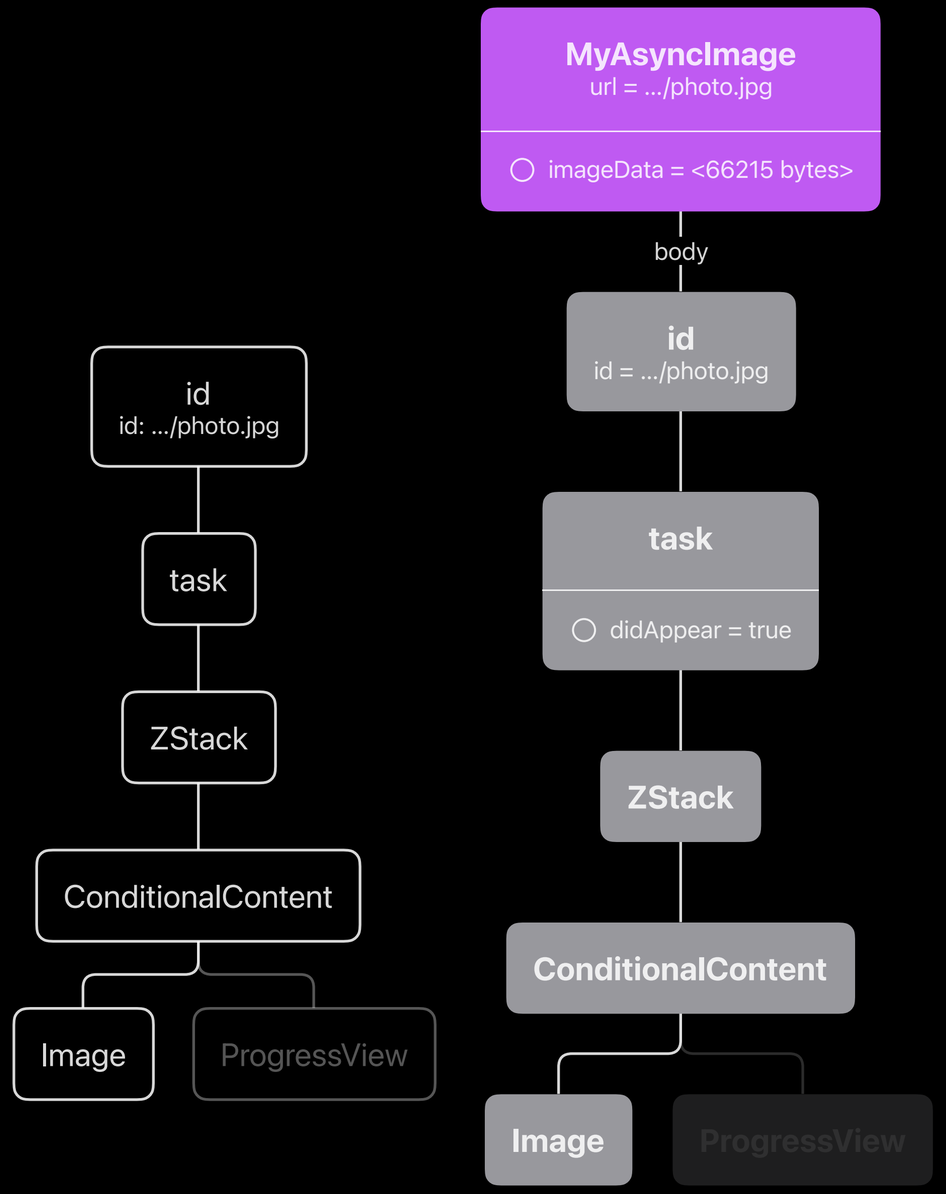
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



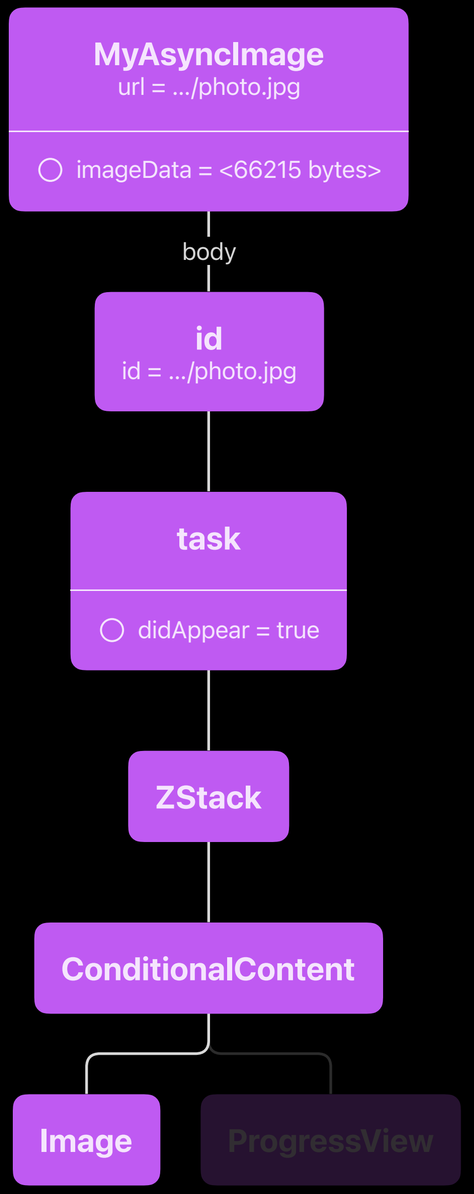
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```

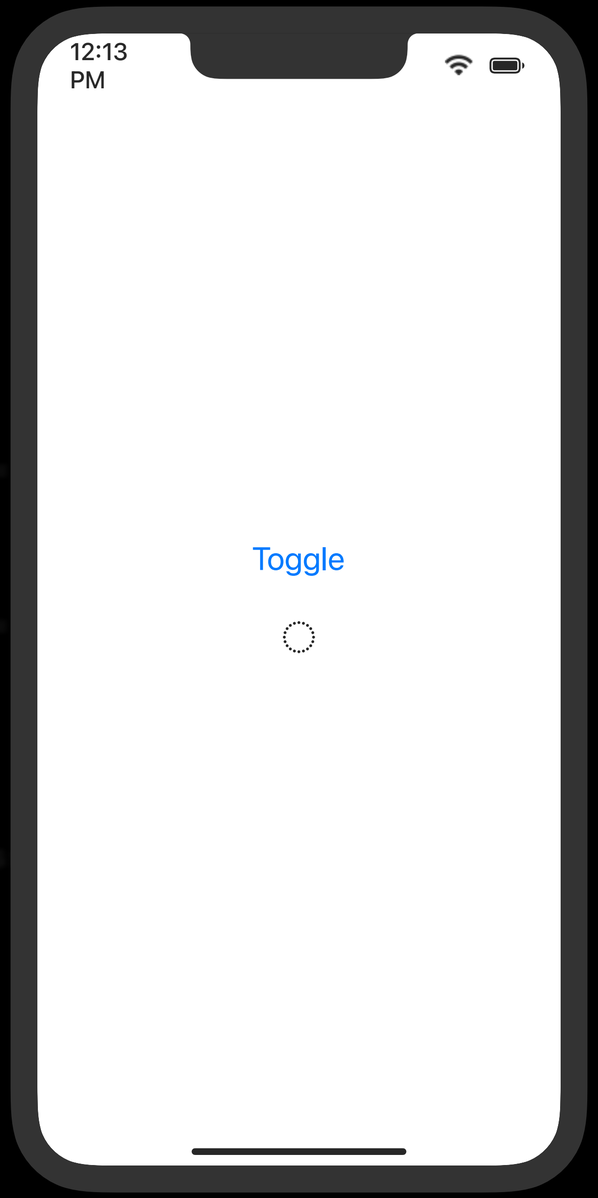


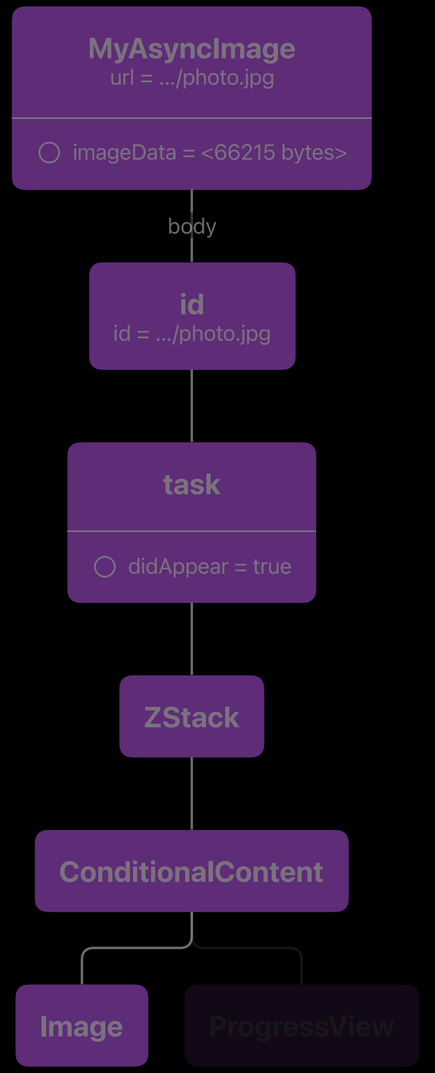
```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```



```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```




12:13
PM
Toggle


MyAsyncImage
url = .../photo.jpg
imageData = <66215 bytes>
body
id
id = .../photo.jpg
task
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```

```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.onAppear {
            // ...
        }
        .id(url)
    }
}
```

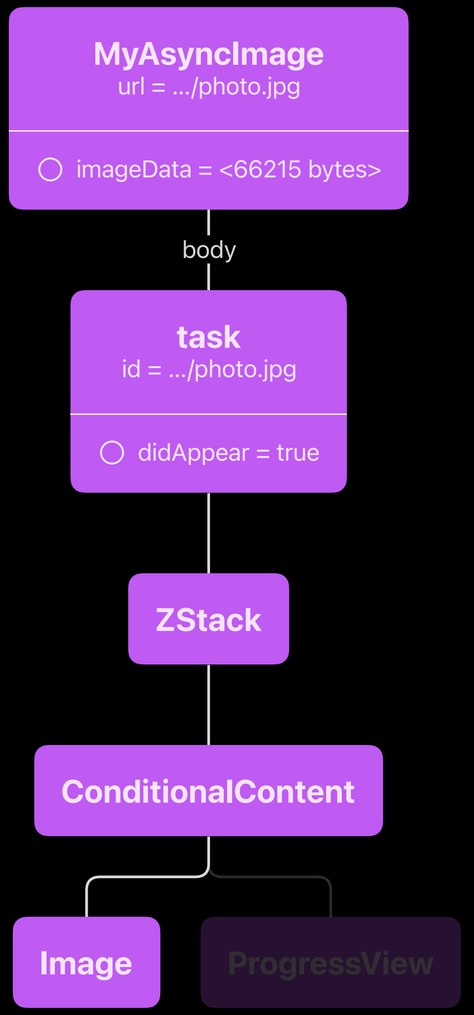
```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.onAppear {
            // ...
        }.onChange(of: url) { newURL in
            // ...
        }
    }
}
```

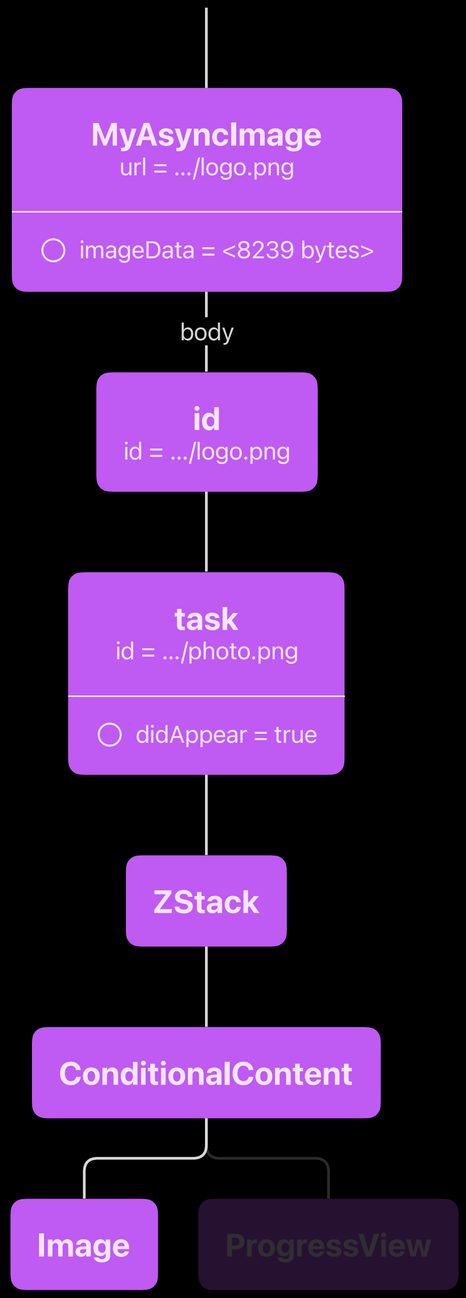
```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.onChange(of: url, initial: true) { old, new in
            // ...
        }
    }
}
```

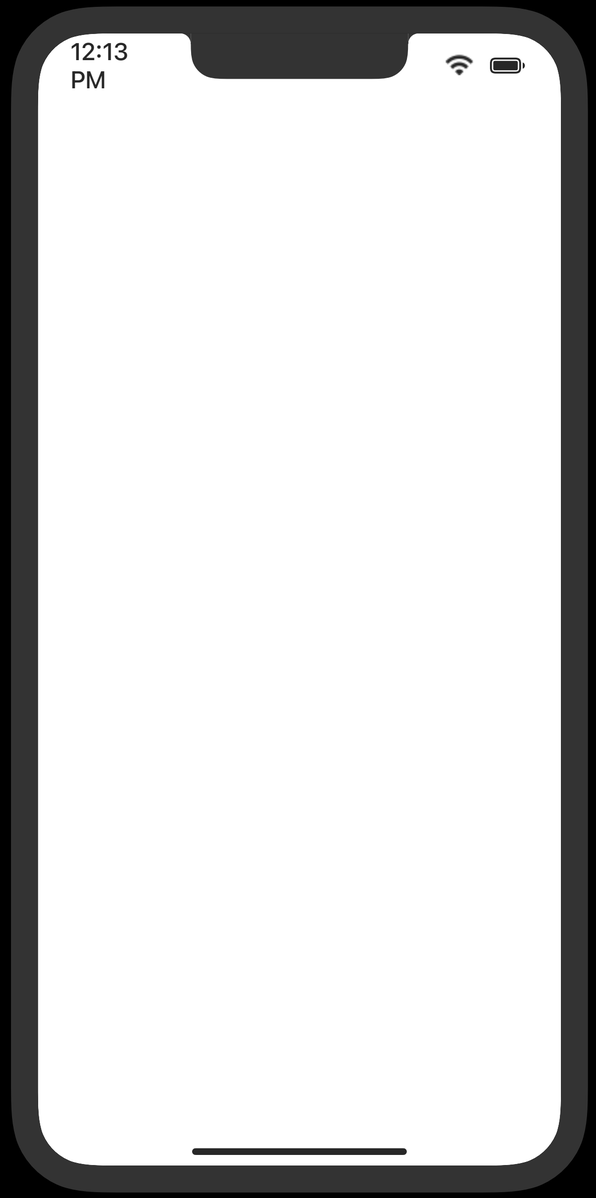
```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task {
            imageData = try? await URLSession.shared.data(from: url).0
        }
        .id(url)
    }
}
```

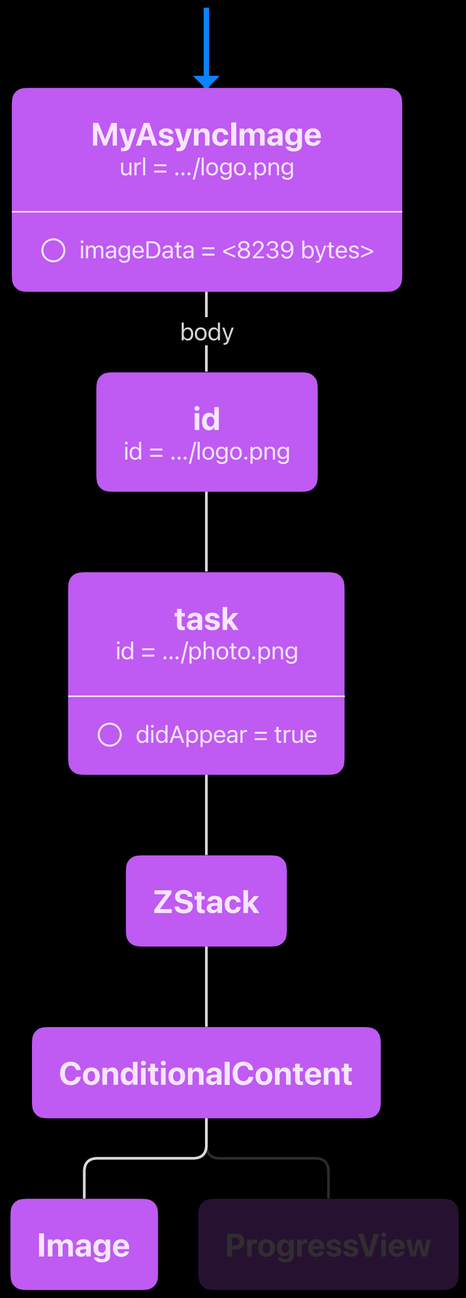
```
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

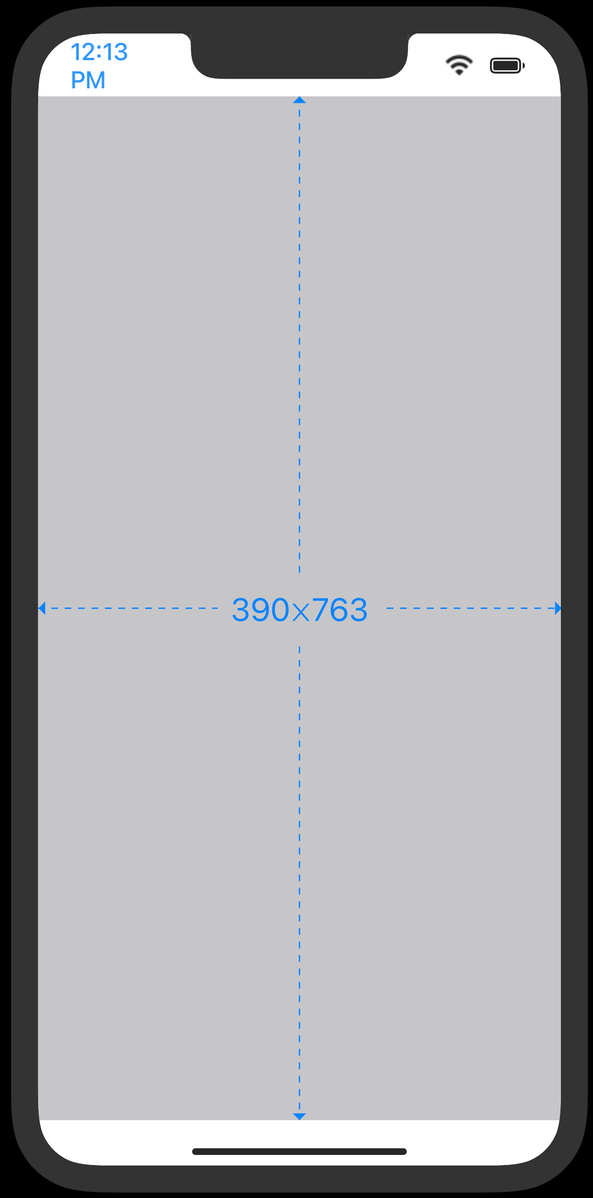
```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

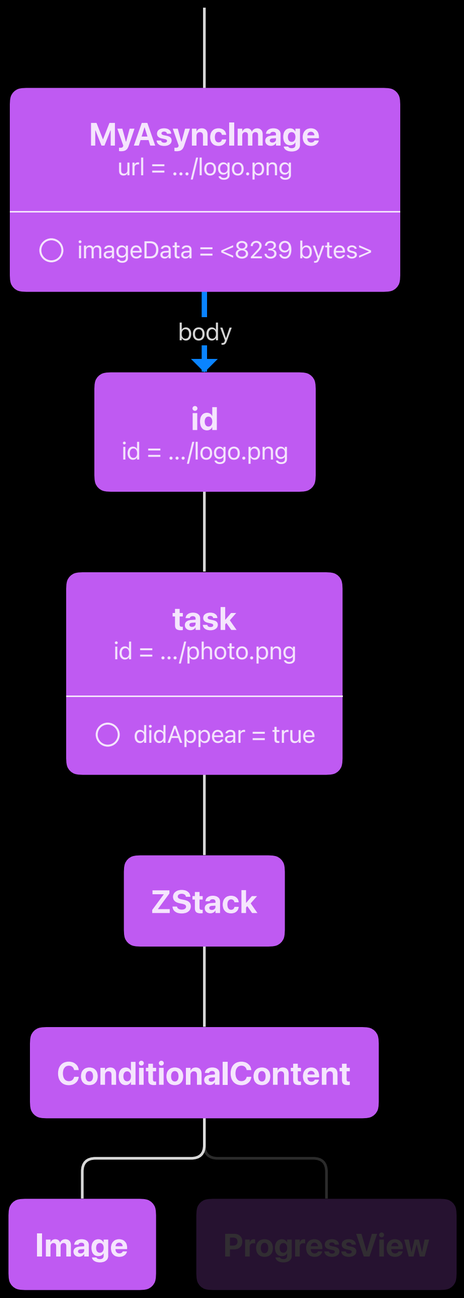


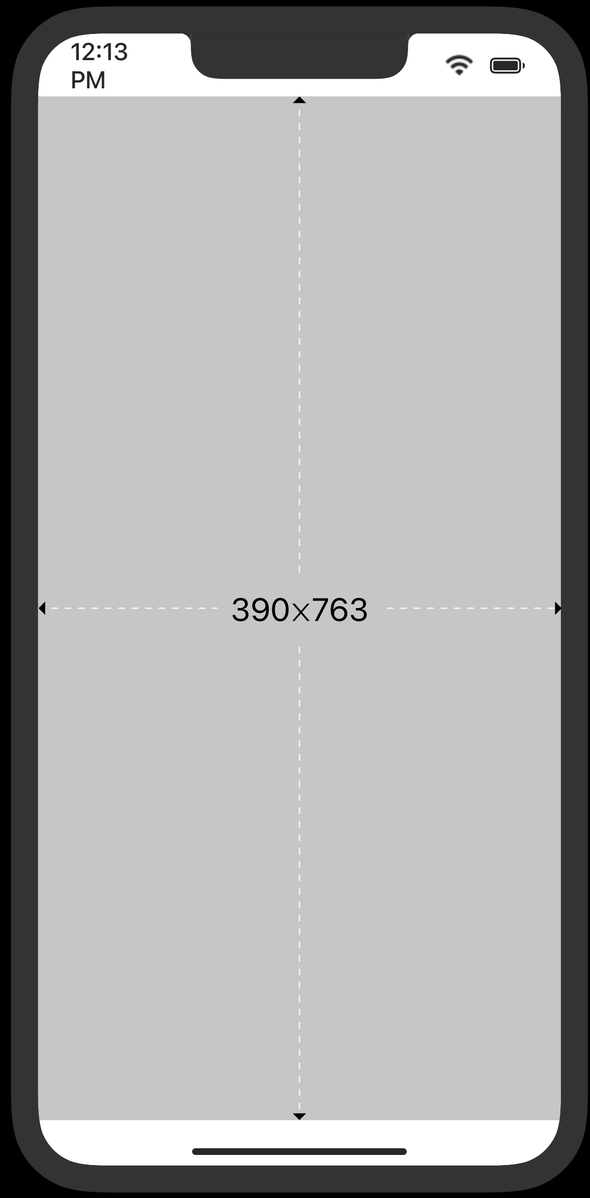

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

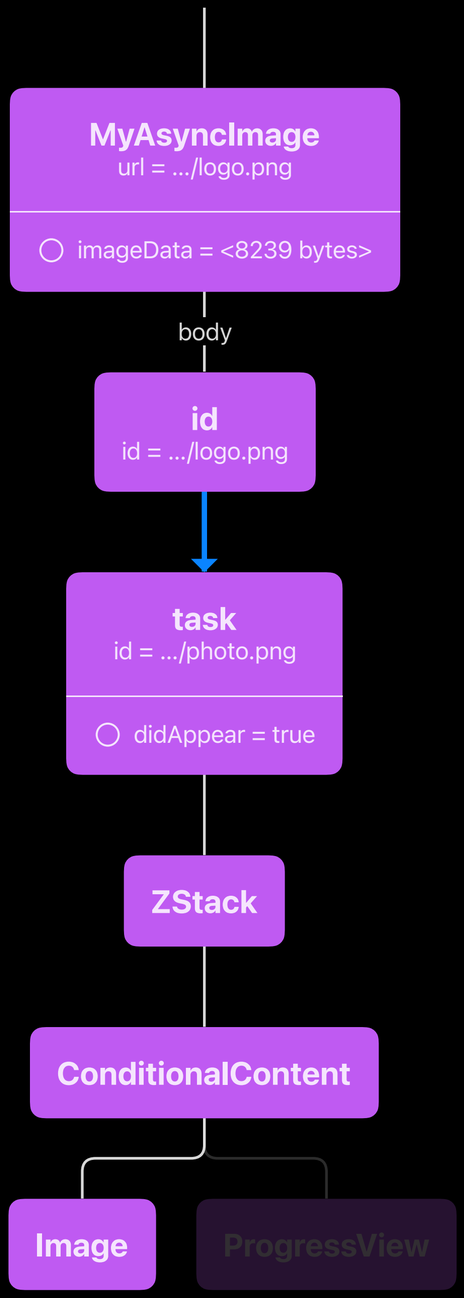

12:13
PM

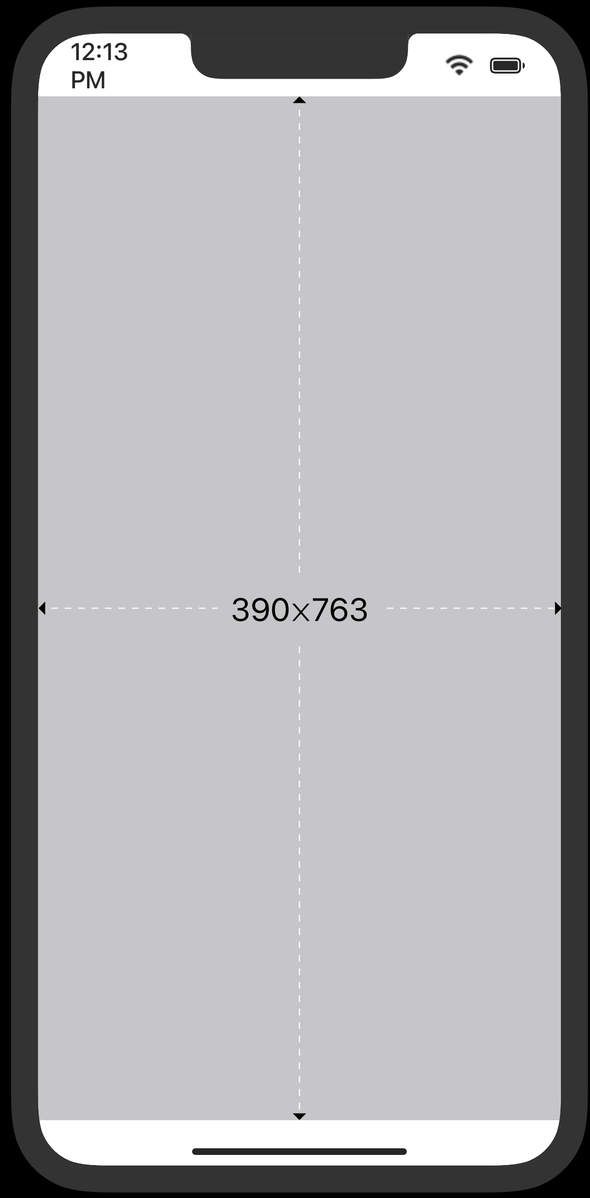

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

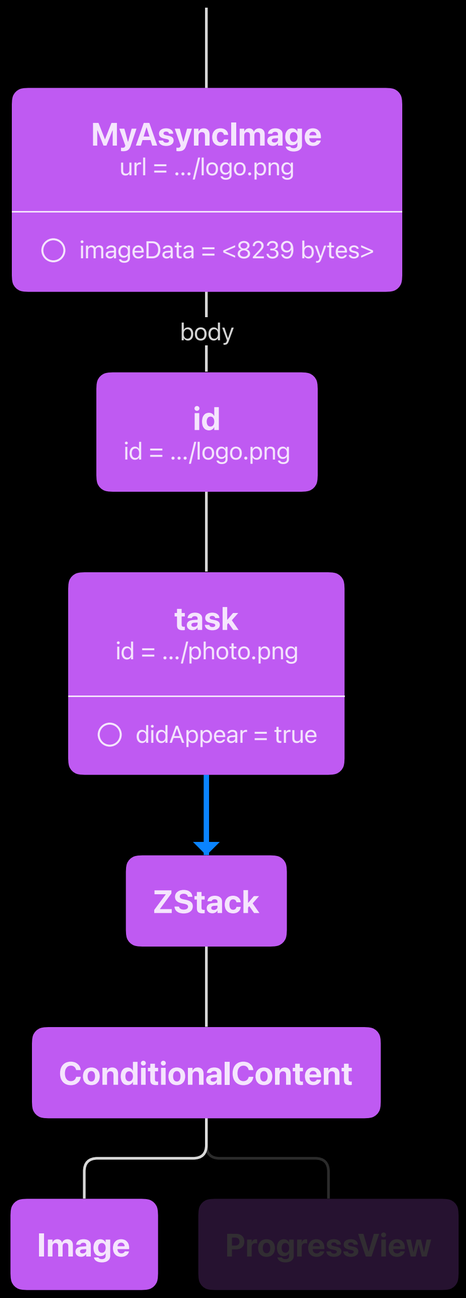

12:13
PM
390×763

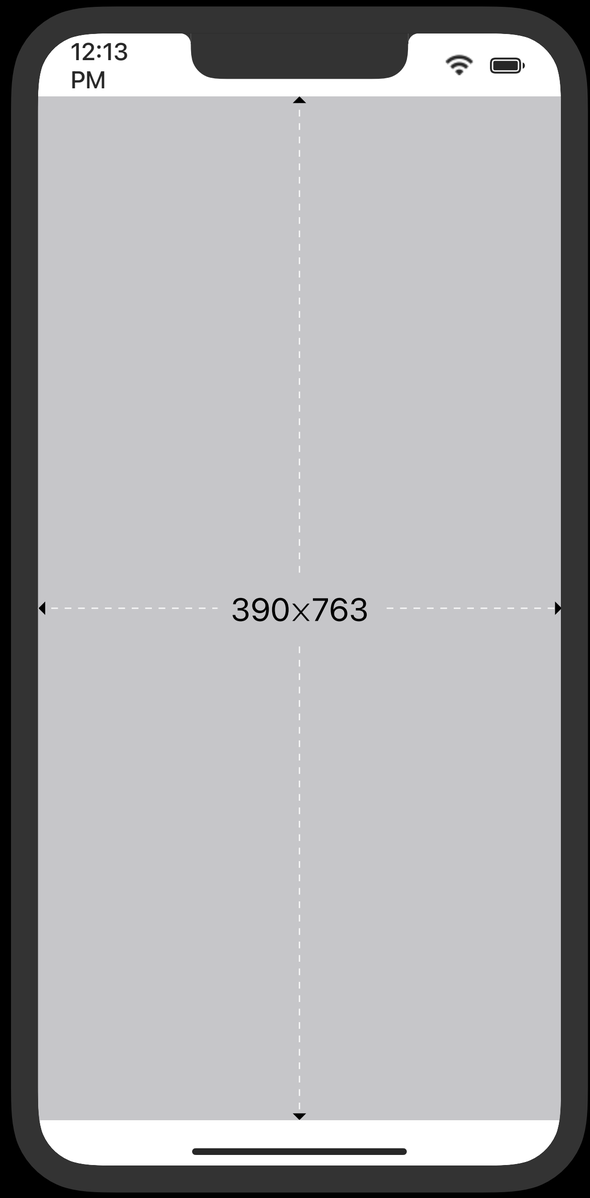

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

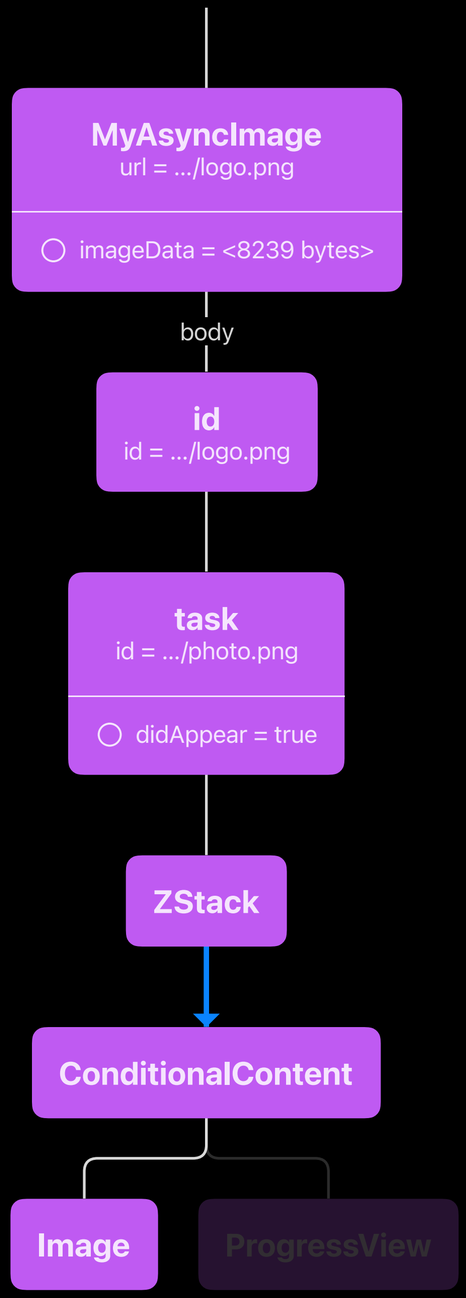

12:13
PM
390×763

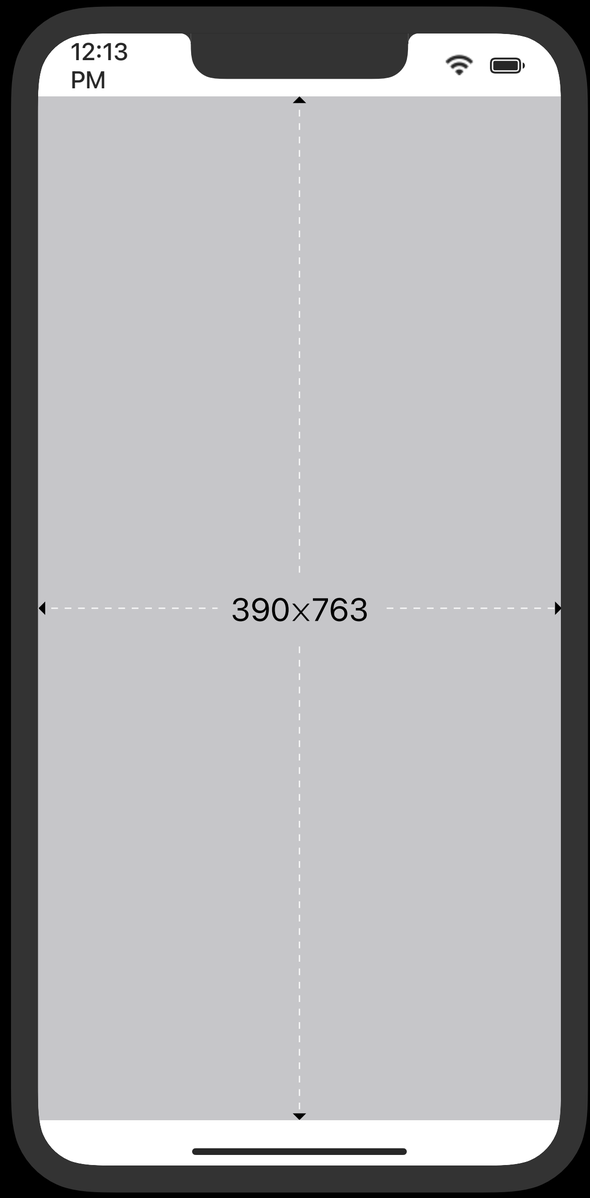

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

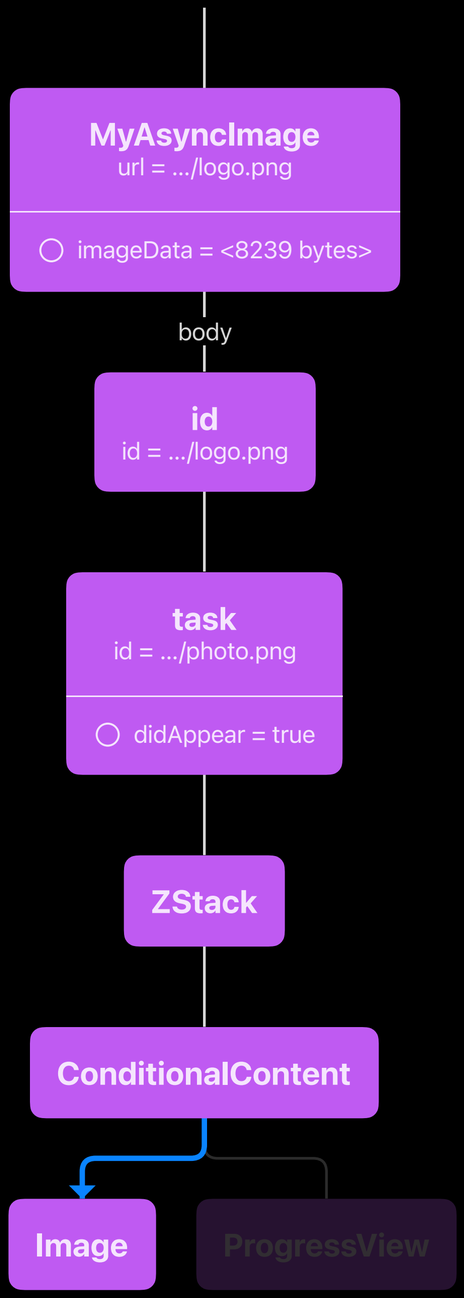

12:13
PM
390×763

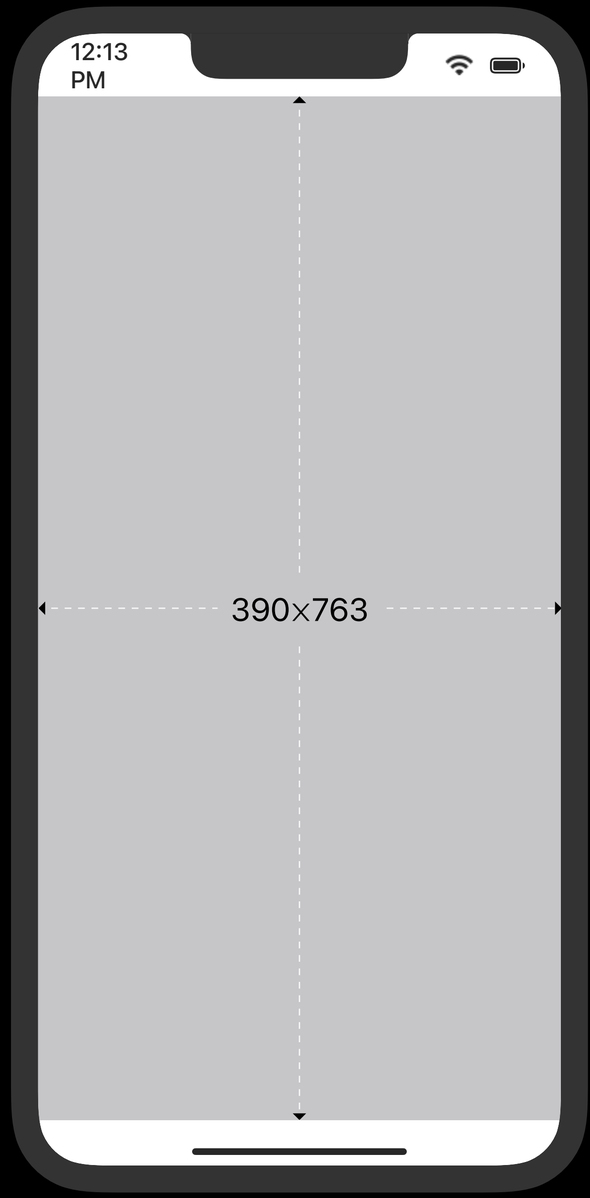

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

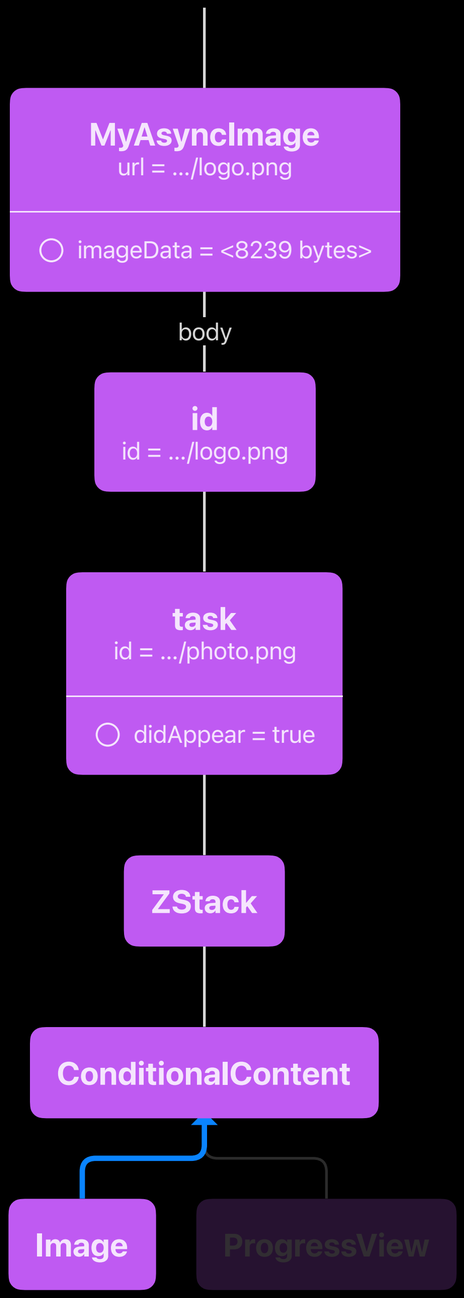

12:13
PM
390×763

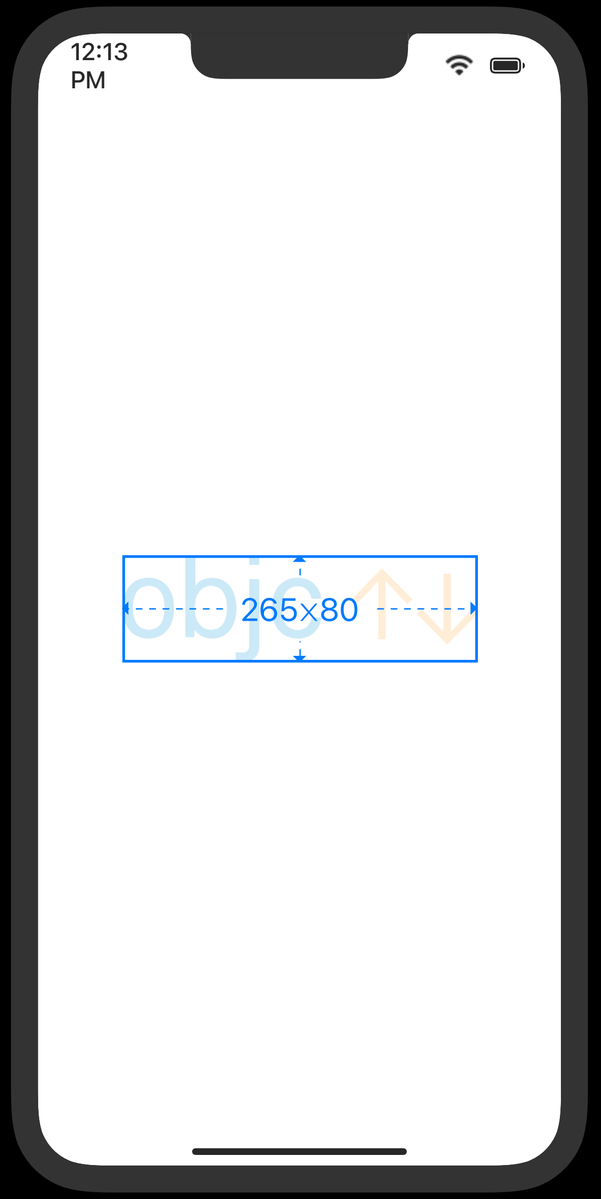

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

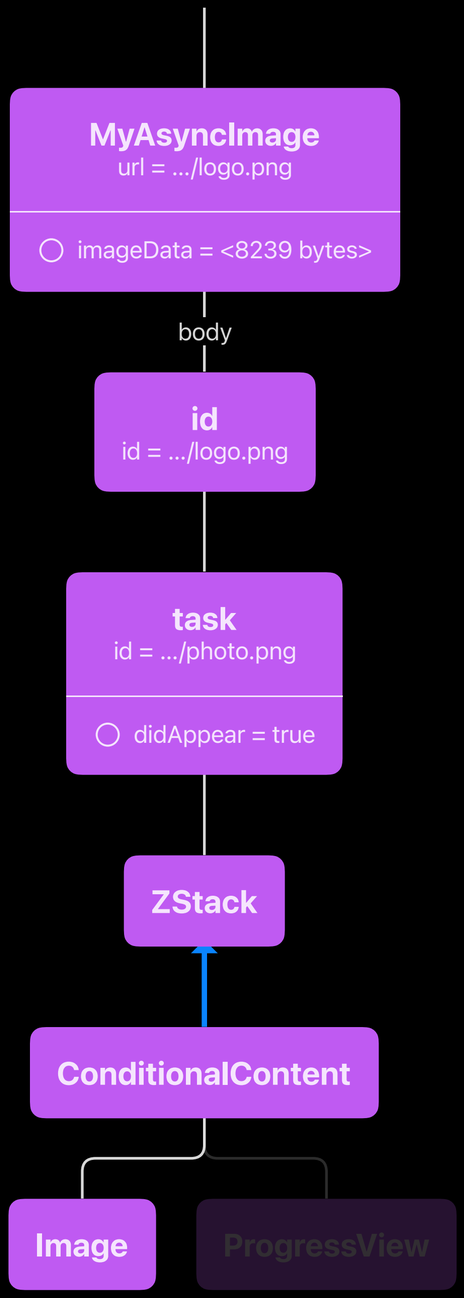

12:13
PM
390×763

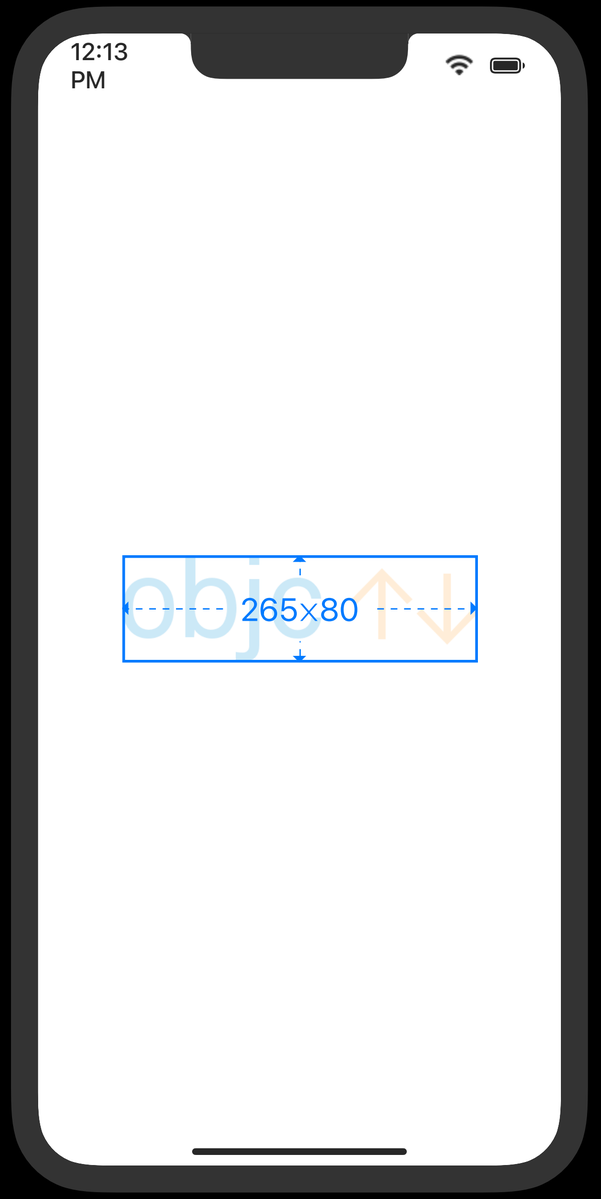

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

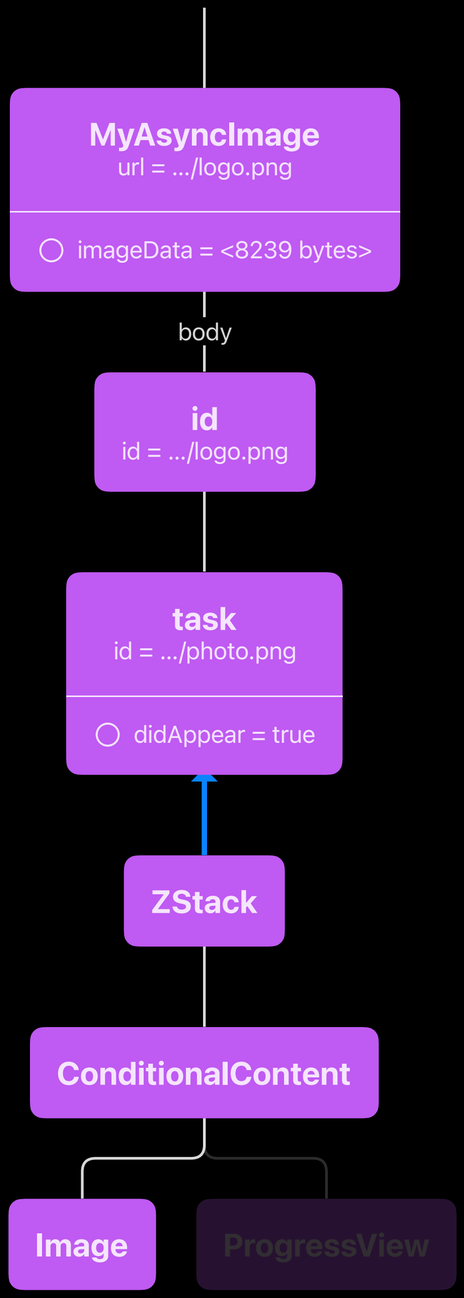

12:13
PM
390×763

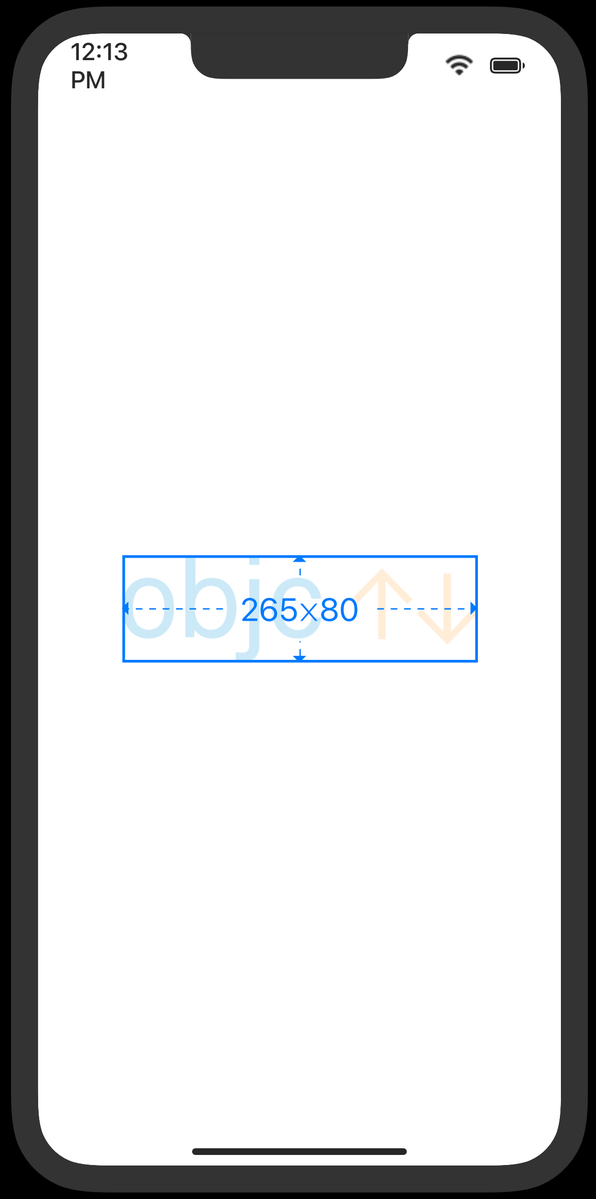

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

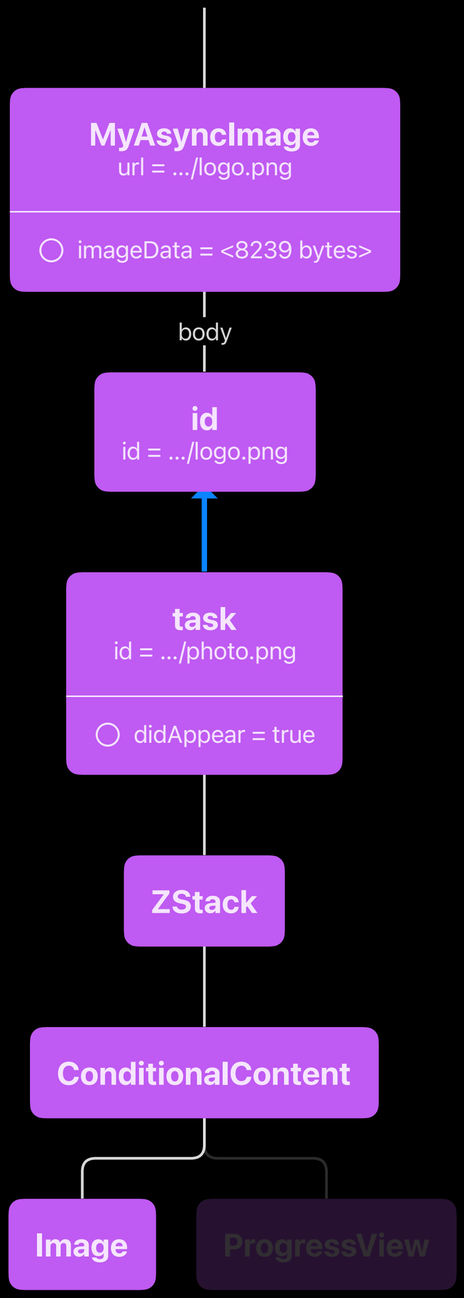

12:13
PM
265×80

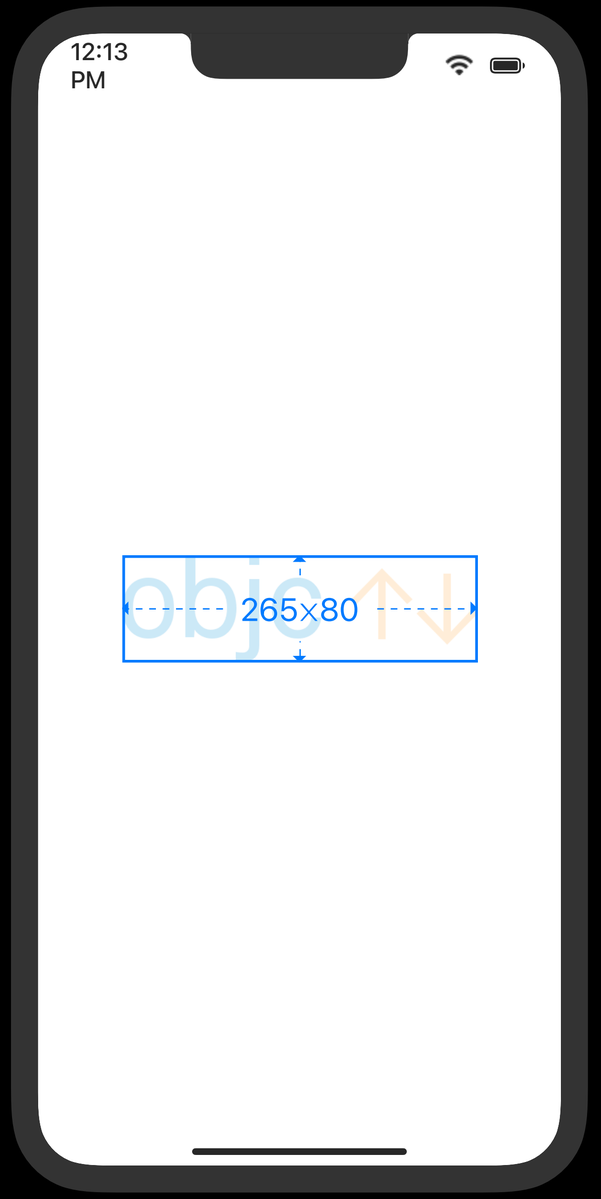

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

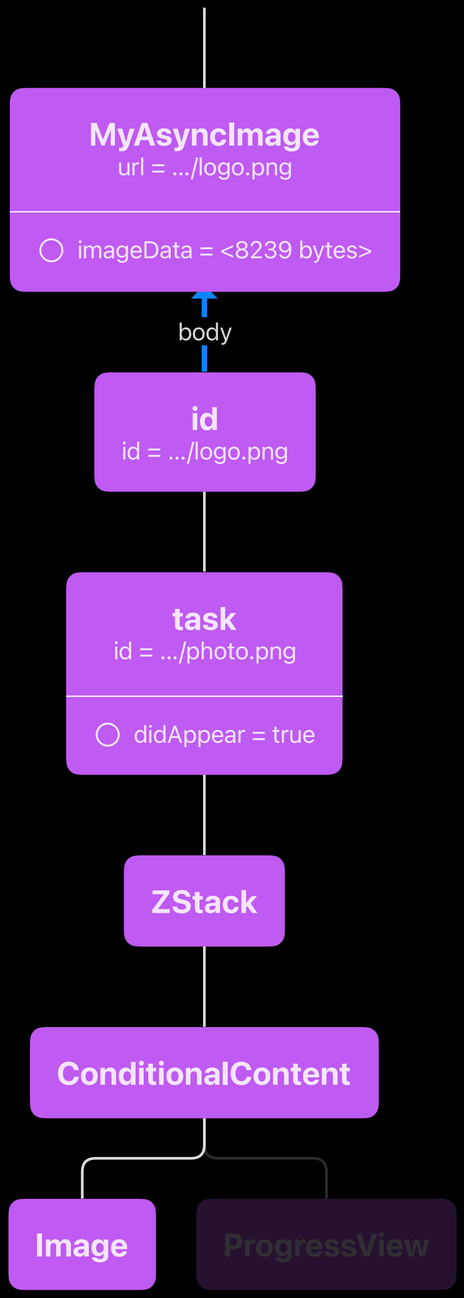

12:13
PM
265×80

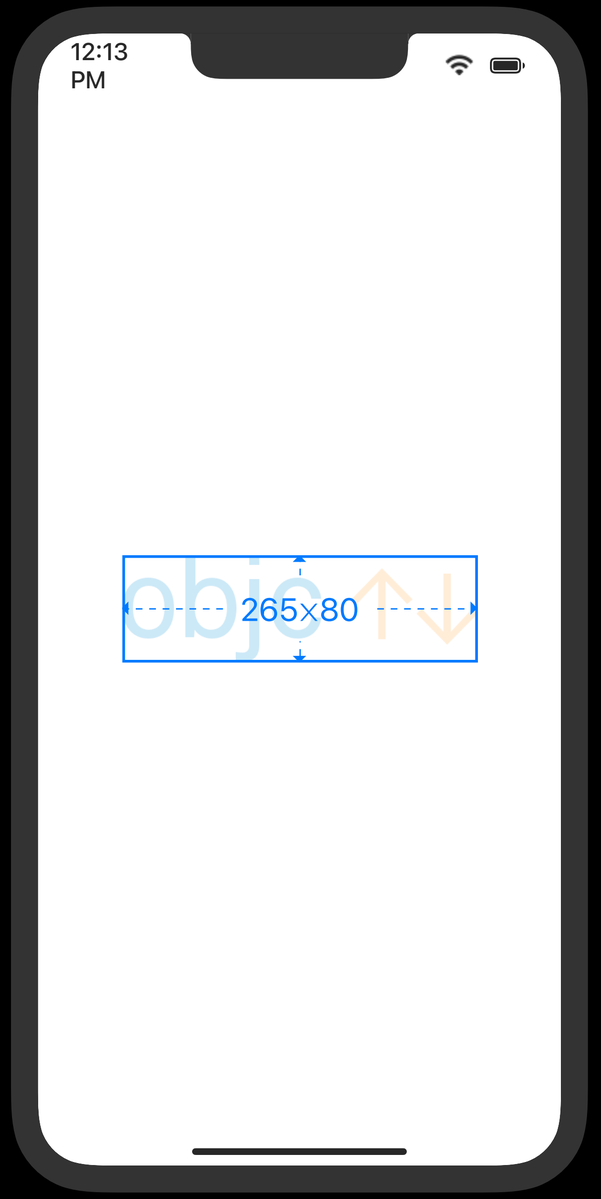

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

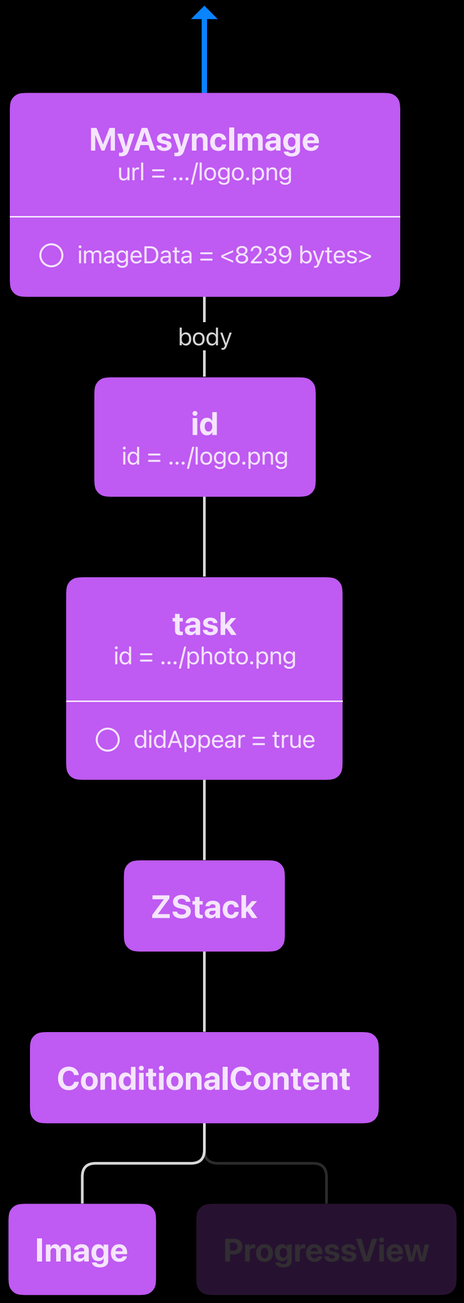

12:13
PM
265×80

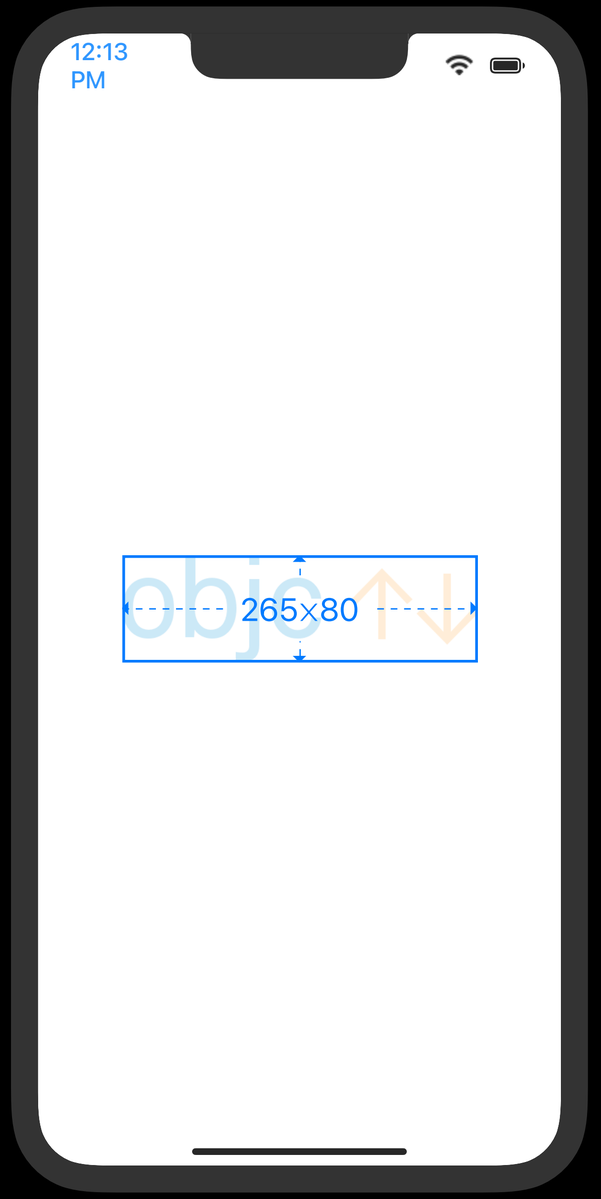

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

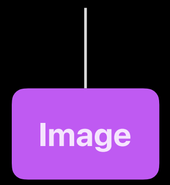

12:13
PM
265×80


MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView

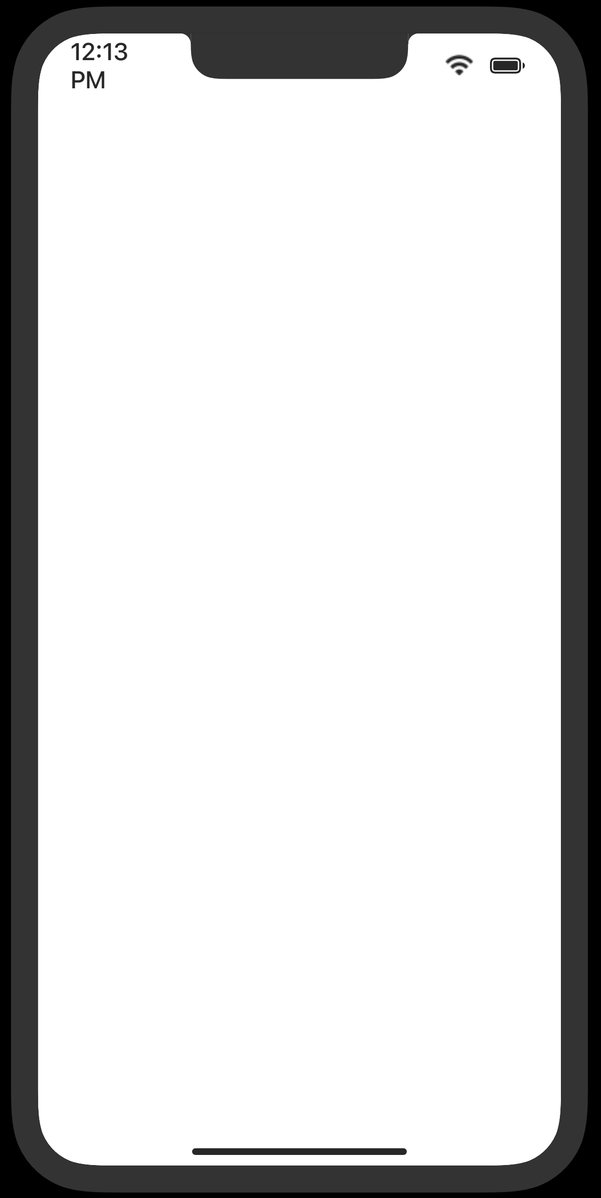

12:13
PM
265×80

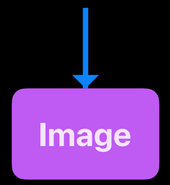

MyAsyncImage
url = .../logo.png
imageData = <8239 bytes>
body
id
id = .../logo.png
task
id = .../photo.png
didAppear = true
ZStack
ConditionalContent
Image
ProgressView


12:13
PM
265×80

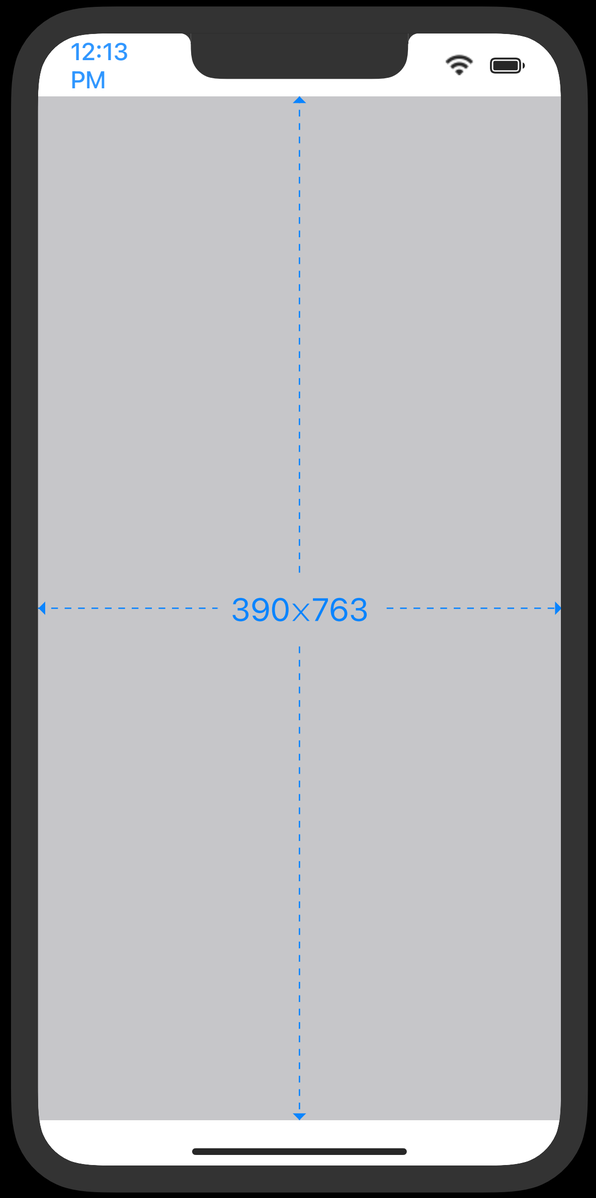


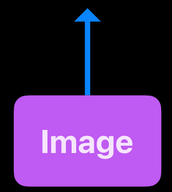
Image


12:13
PM

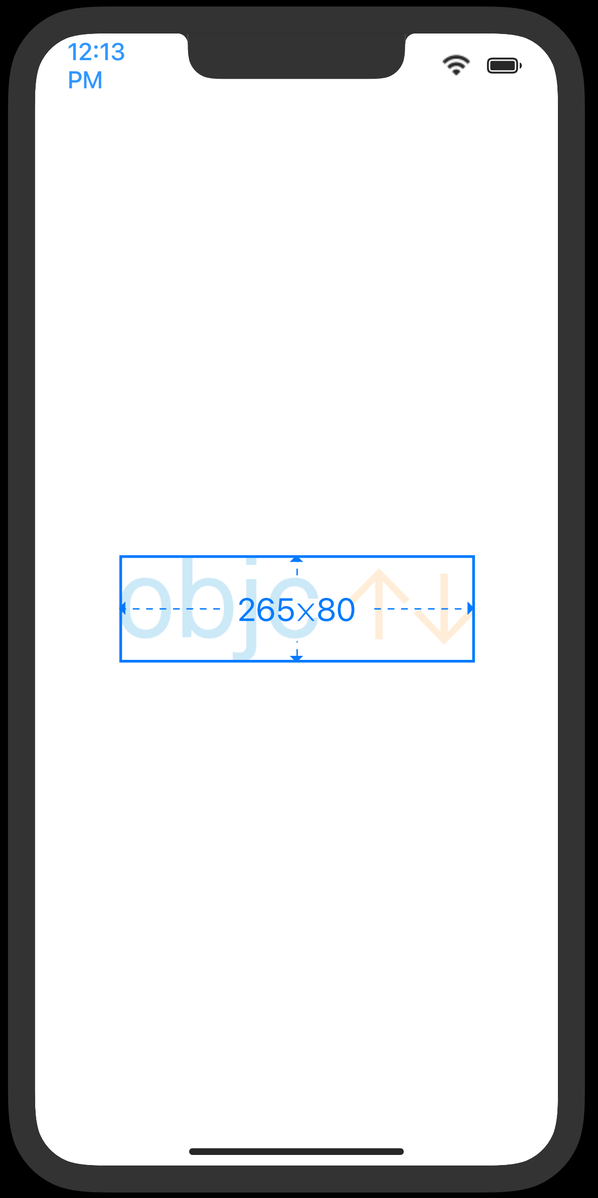


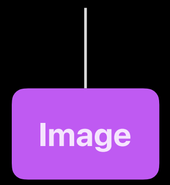
Image


12:13
PM
390×763



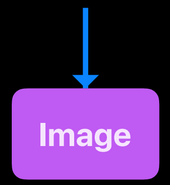
Image


12:13
PM
265×80

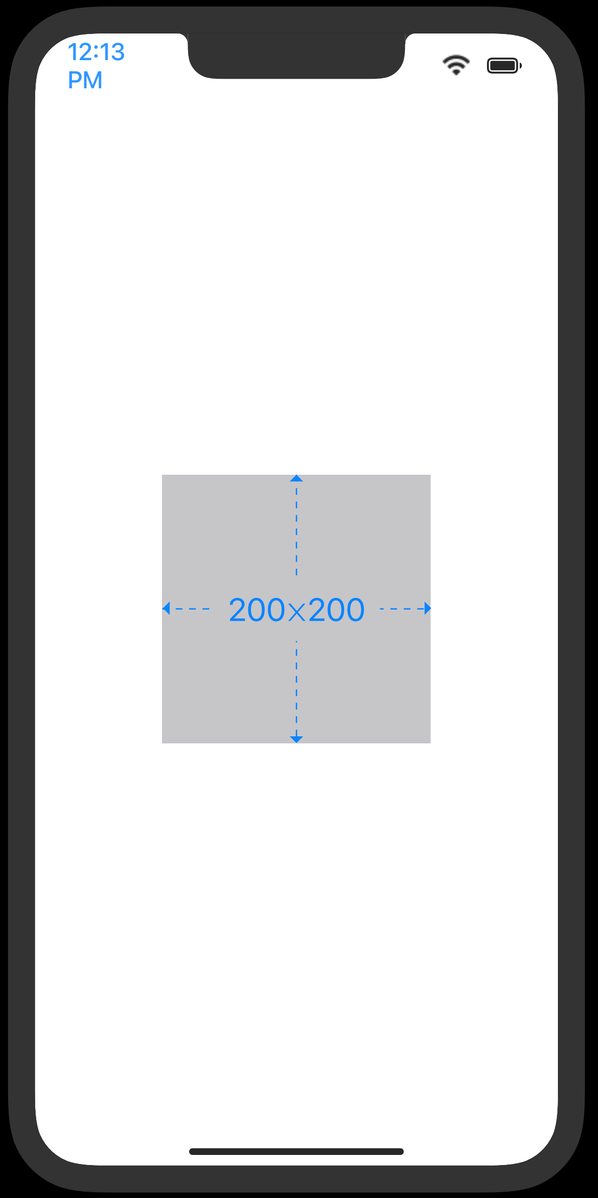


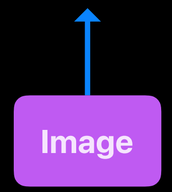
Image


12:13
PM
objc

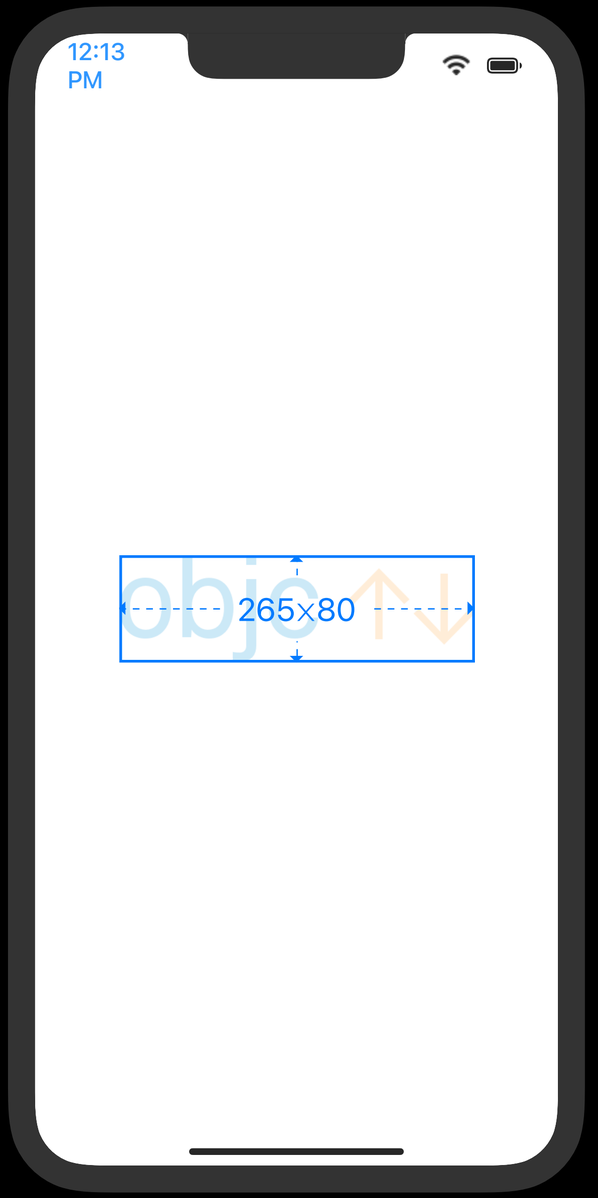


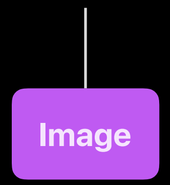
Image


12:13
PM
200×200



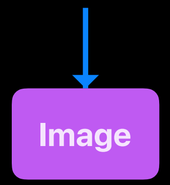
Image


12:13
PM
265×80

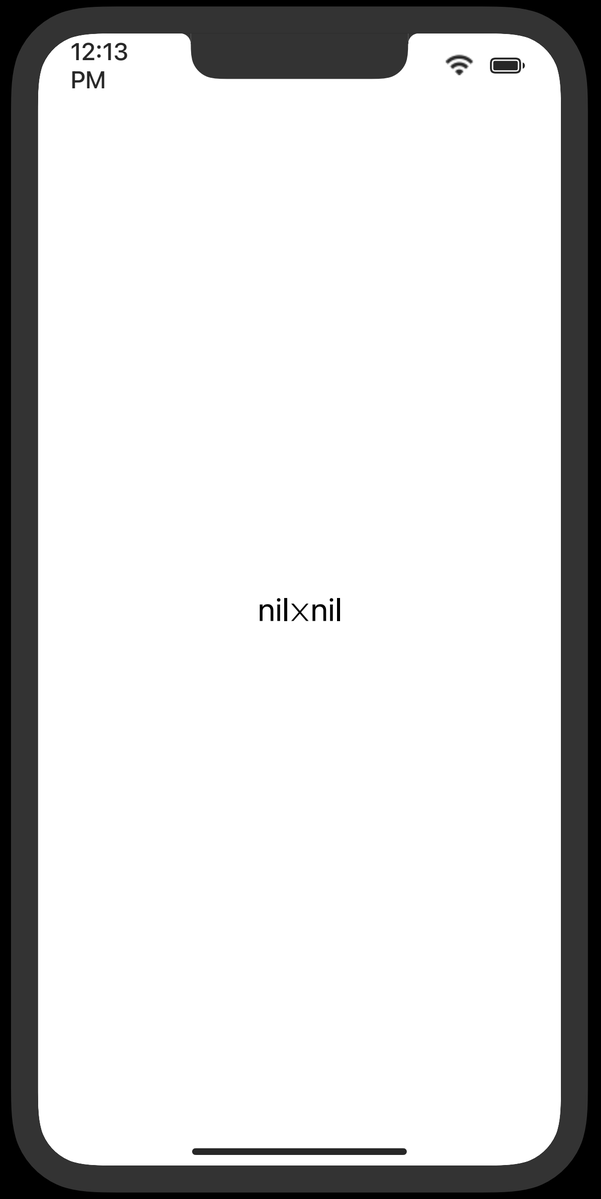


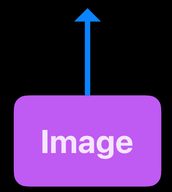
Image


12:13
PM
objc

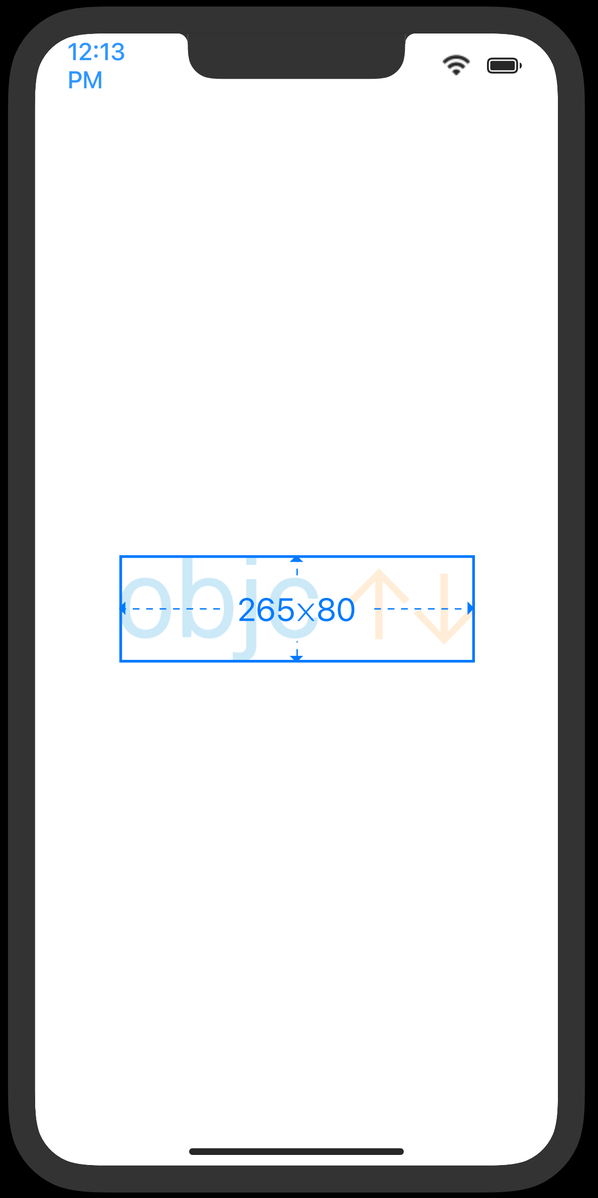


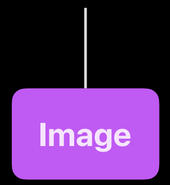
Image


12:13
PM
nil×nil



Image

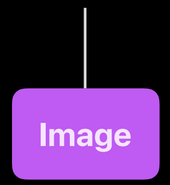

12:13
PM
265×80

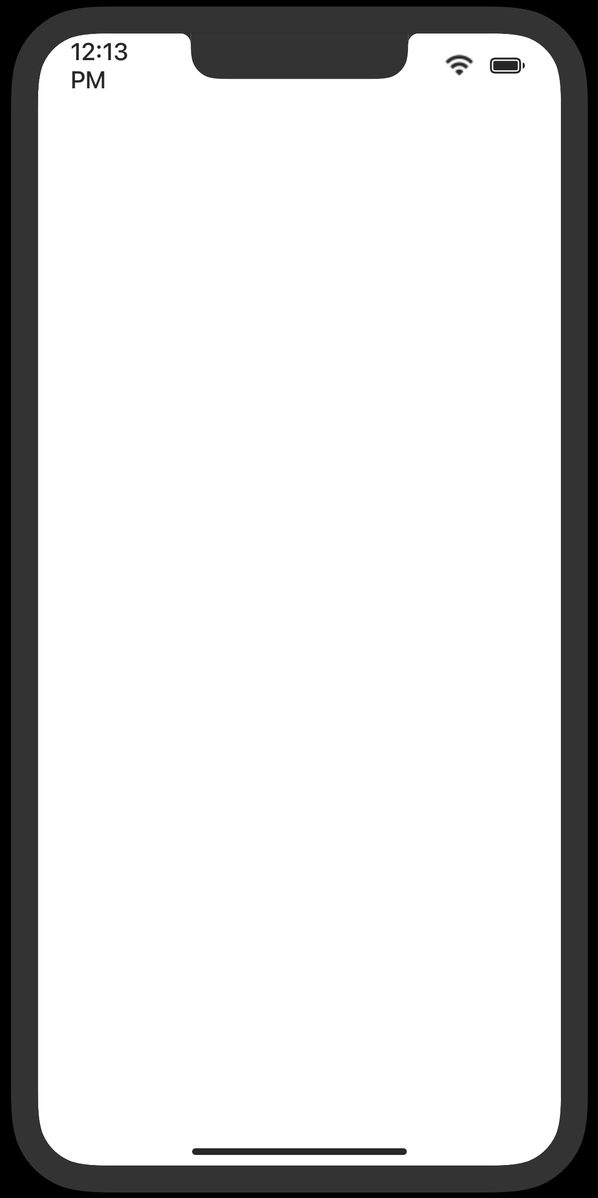


Image

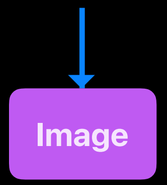

12:13
PM
objc

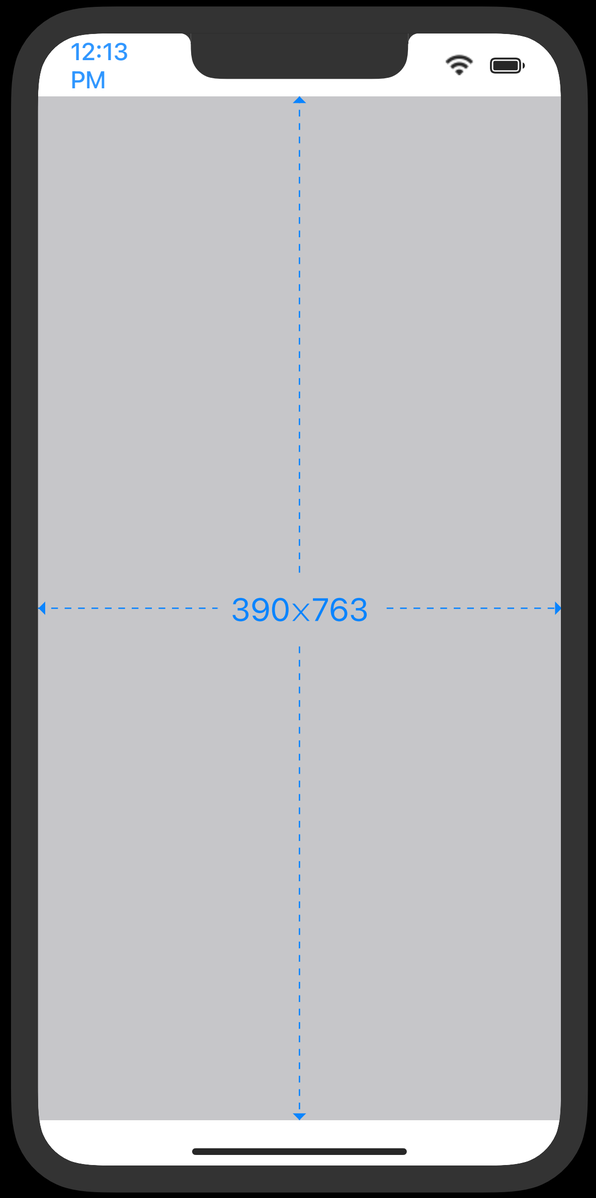
```
0  Image("logo")
1      .resizable()
```



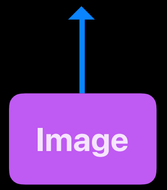


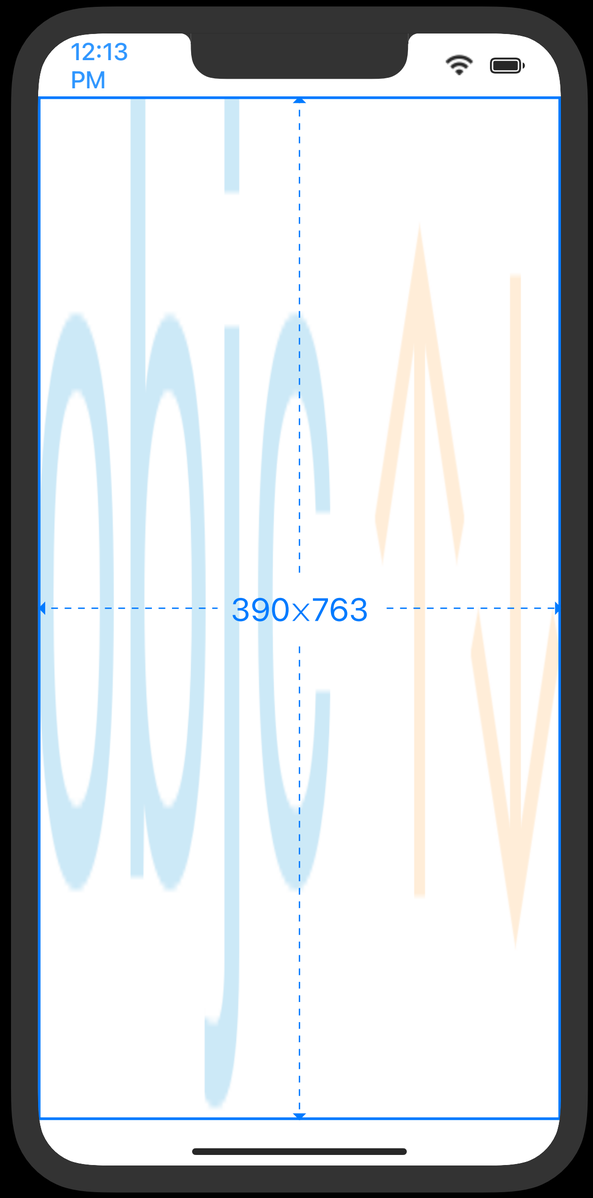
```
0  Image("logo")
1      .resizable()
```



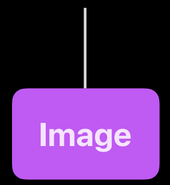


```
0  Image("logo")
1      .resizable()
```

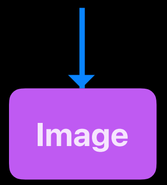


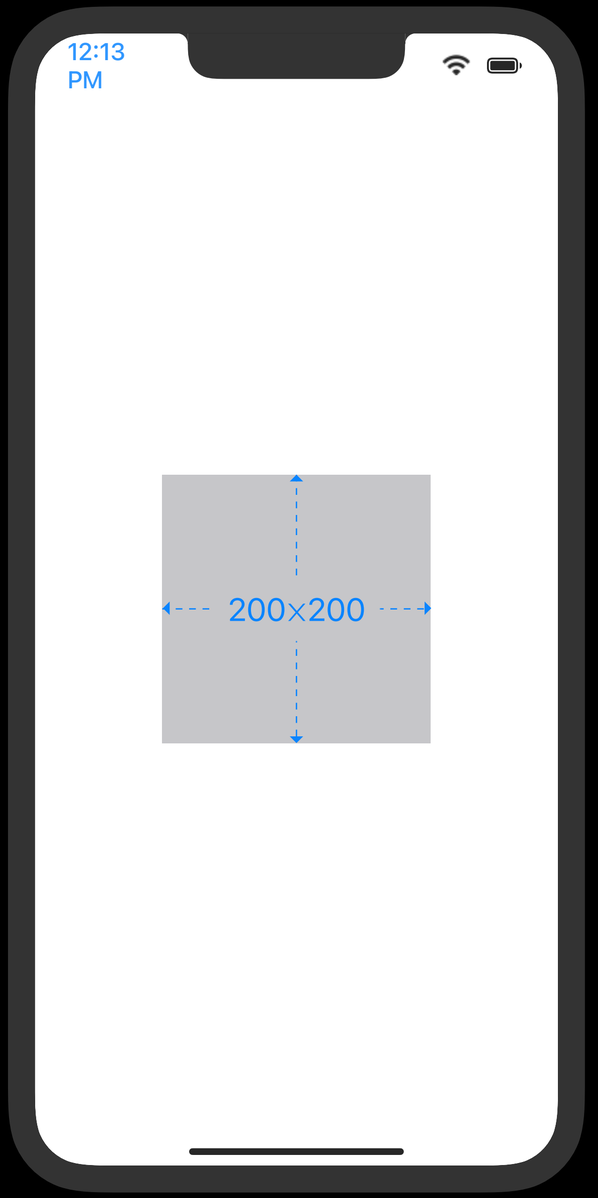


```
0  Image("logo")
1      .resizable()
```

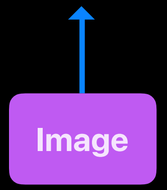


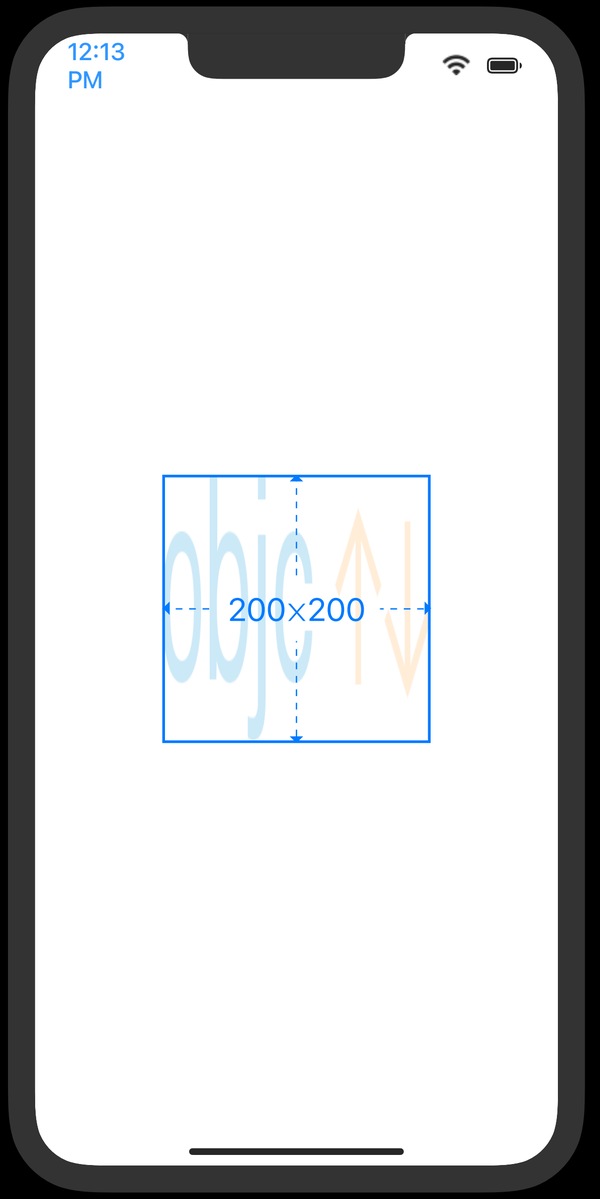
```
0  Image("logo")
1      .resizable()
```



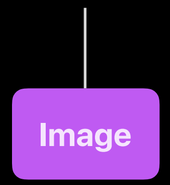


```
0  Image("logo")
1      .resizable()
```

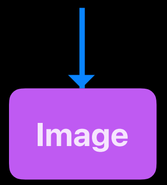


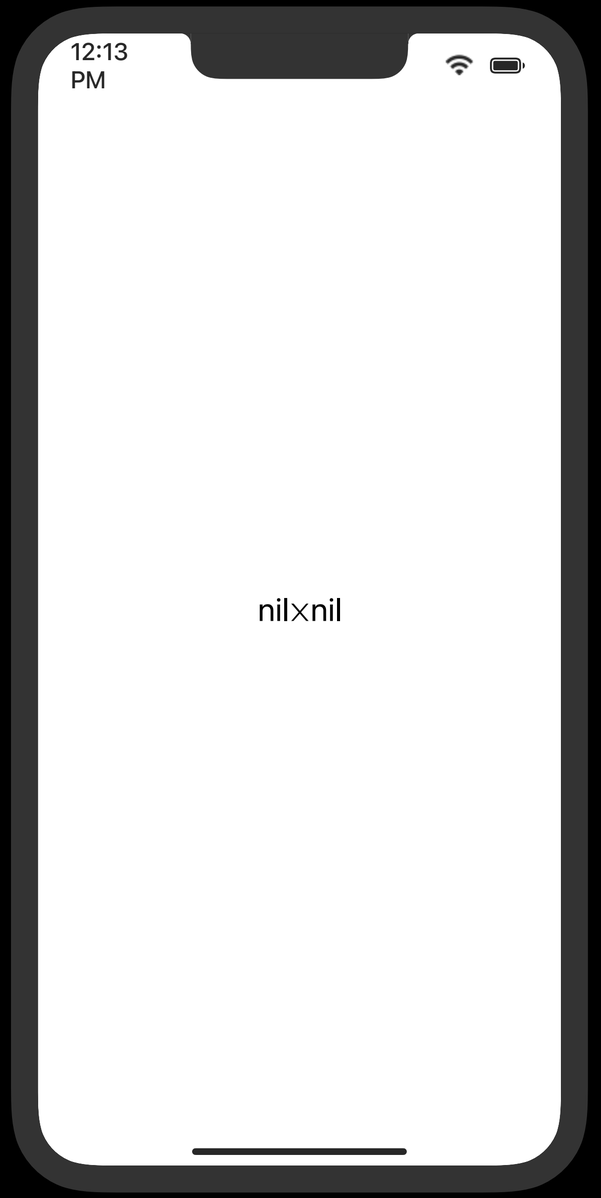


```
0  Image("logo")
1      .resizable()
```

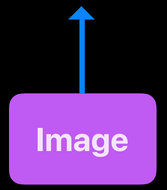


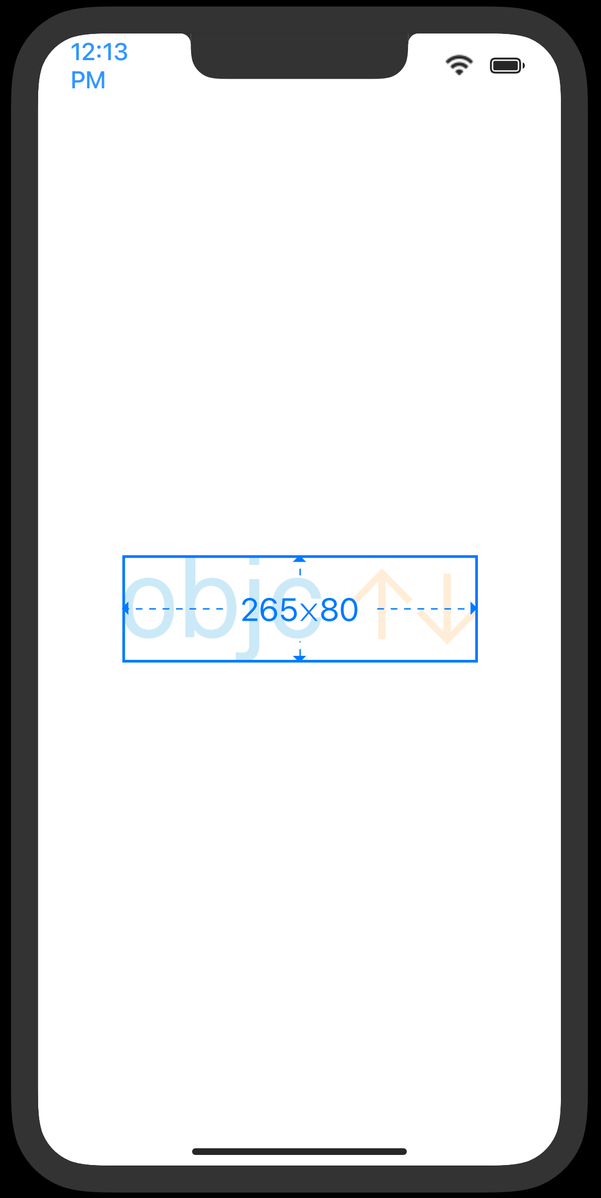
```
0  Image("logo")
1      .resizable()
```



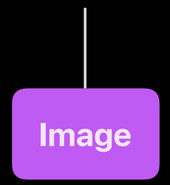


```
0  Image("logo")
1      .resizable()
```

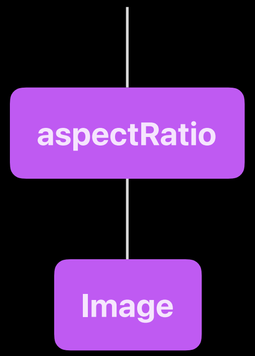


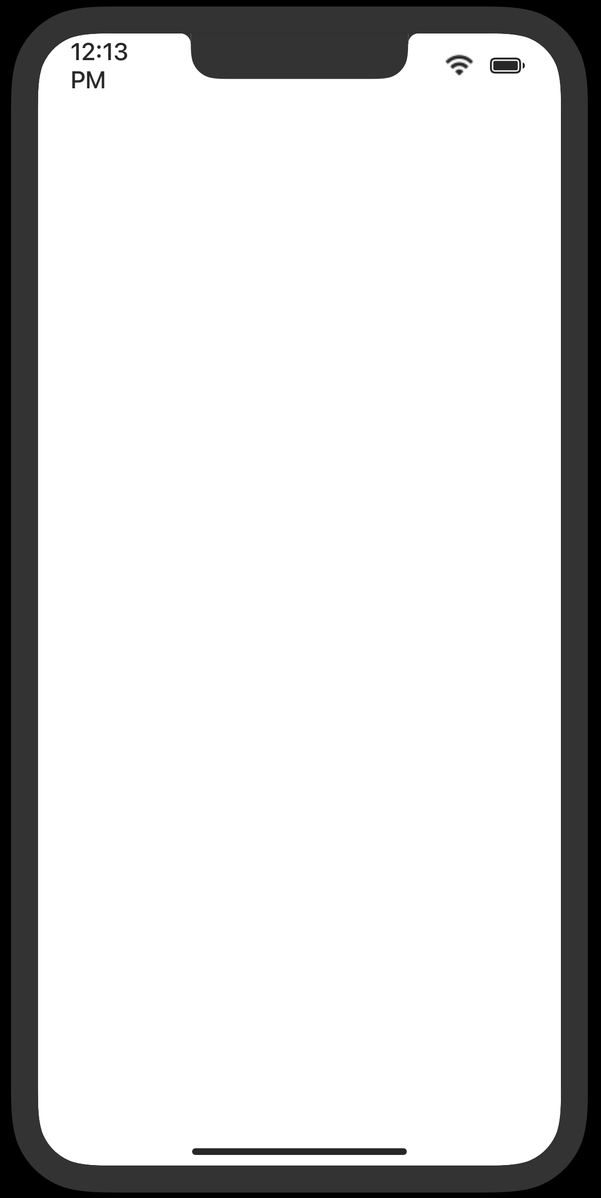


```
0  Image("logo")
1      .resizable()
```

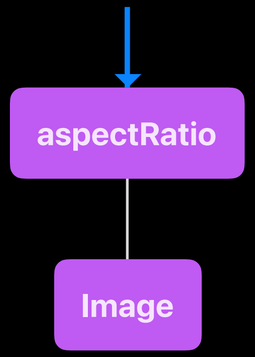


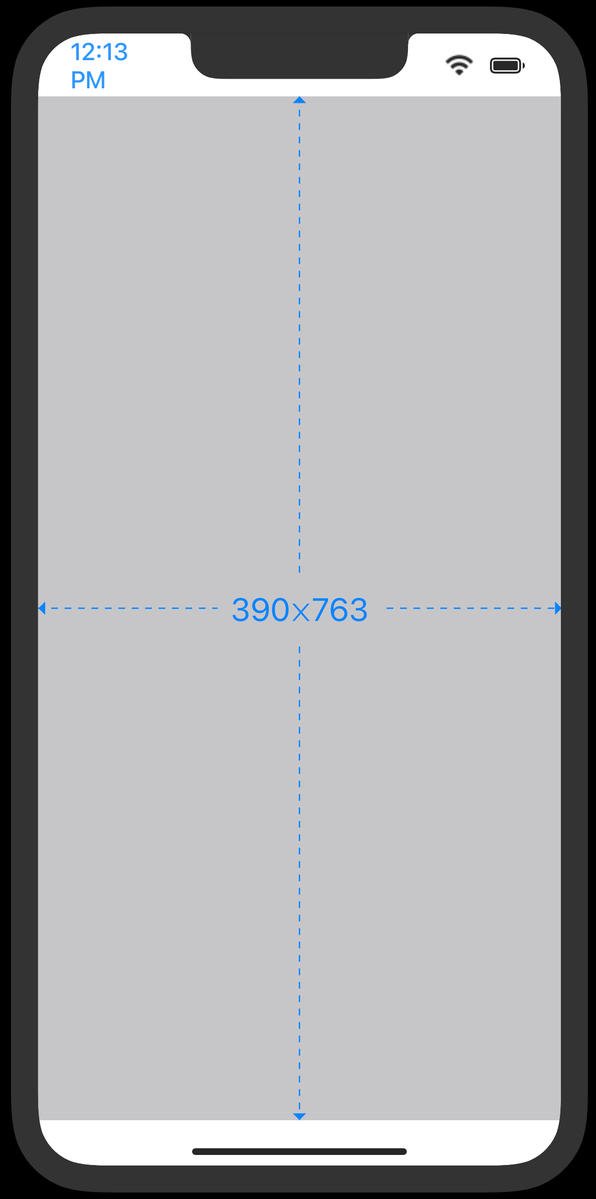
```
Image("logo")
    .resizable()
    .aspectRatio(contentMode: .fit)
```



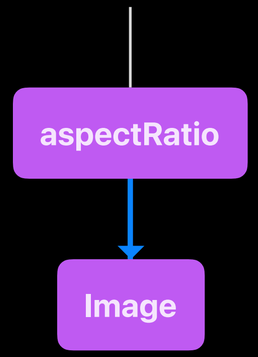


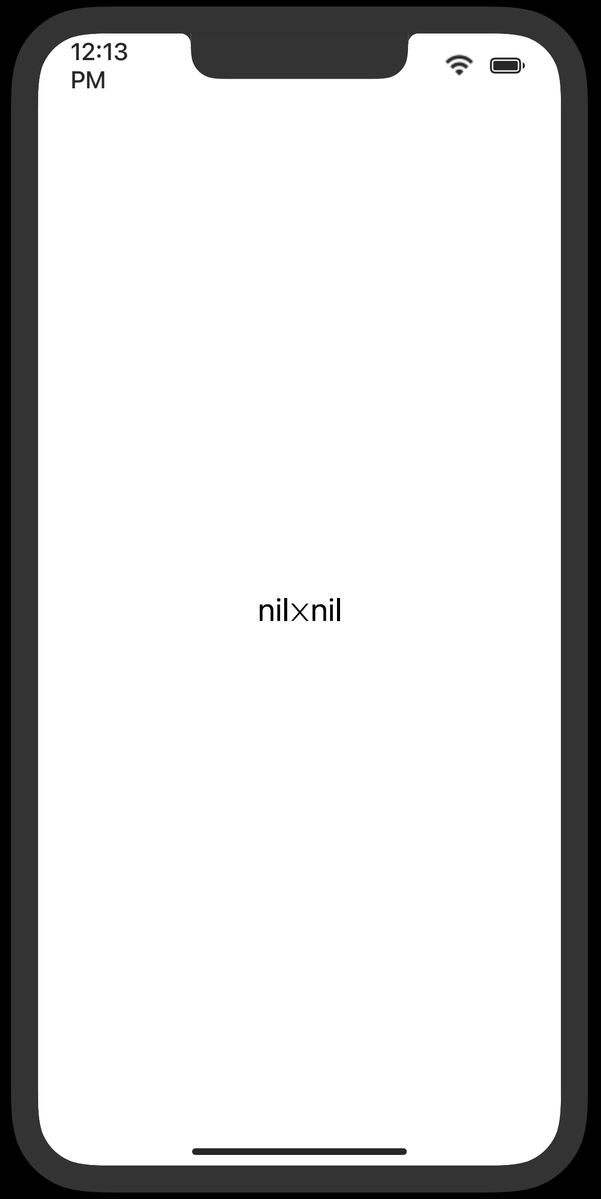
```swift
Image("logo")
    .resizable()
    .aspectRatio(contentMode: .fit)
```



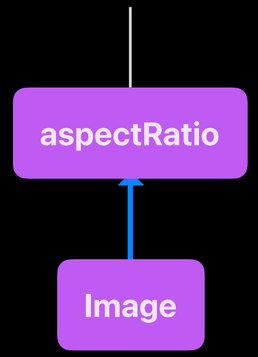


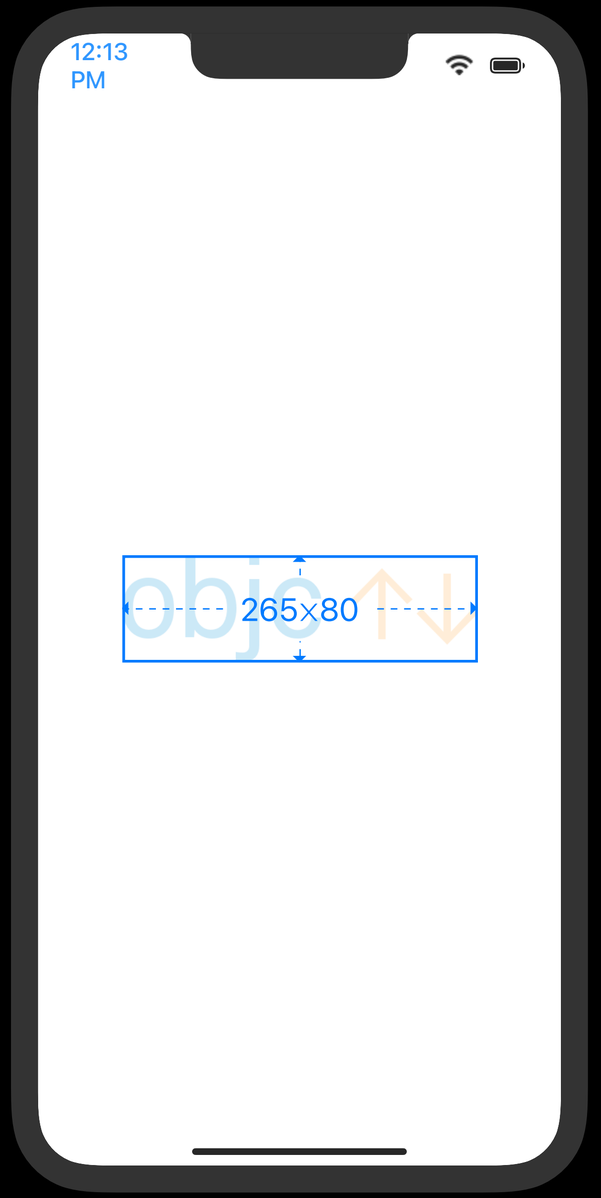
```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



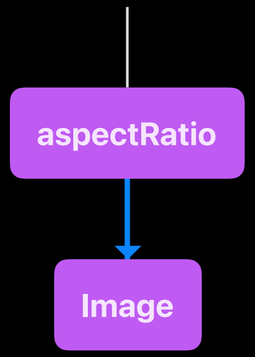


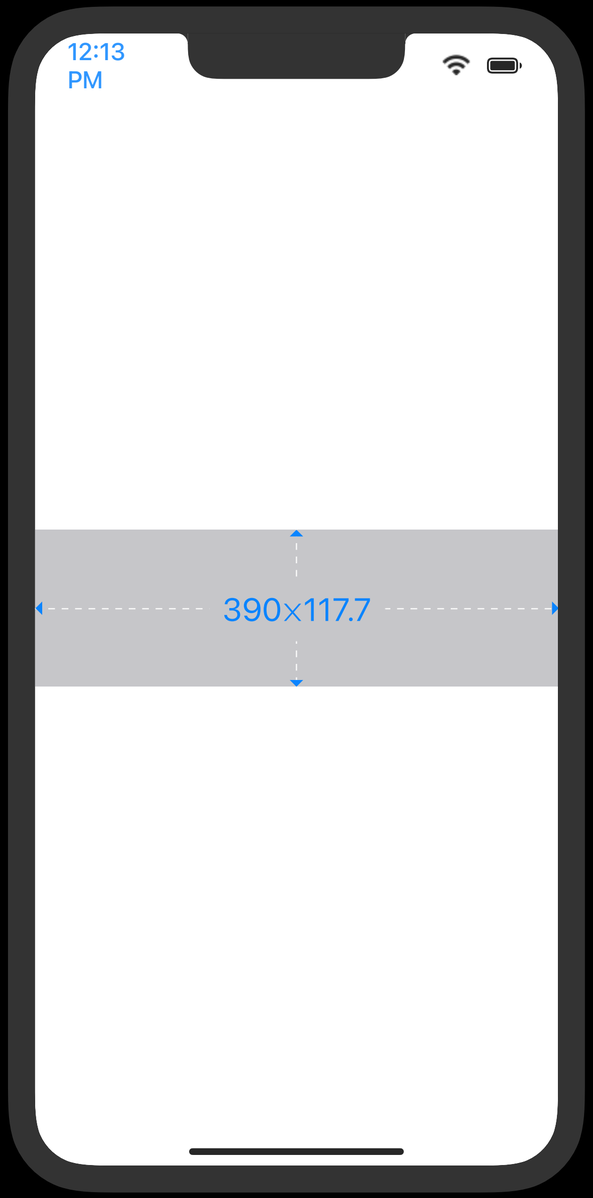
```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



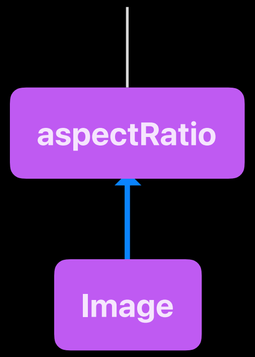


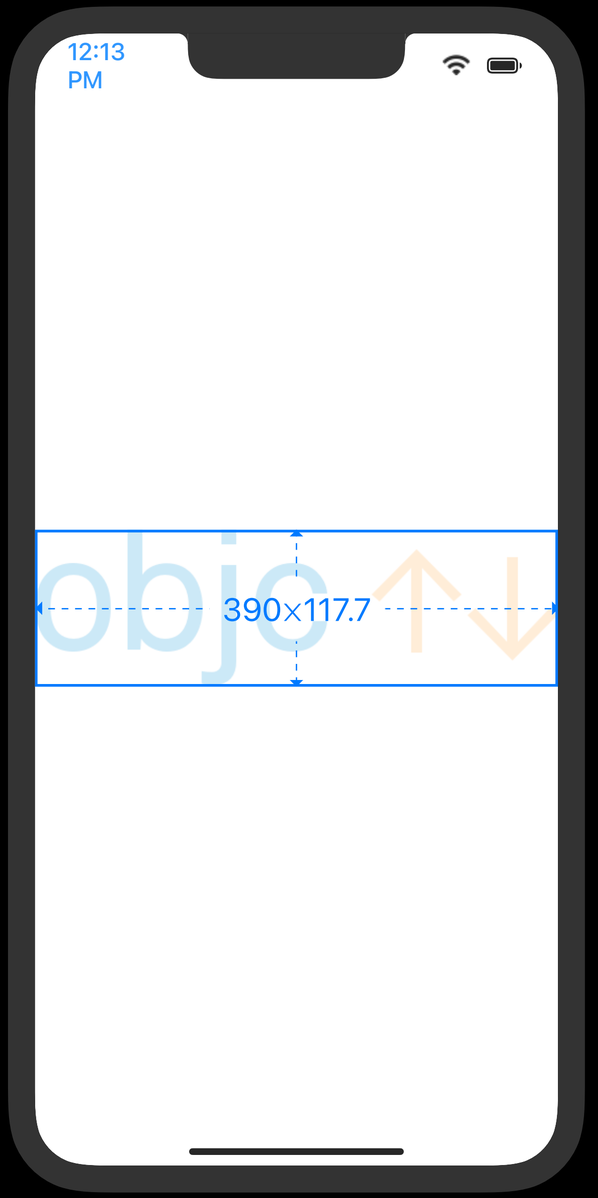
```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



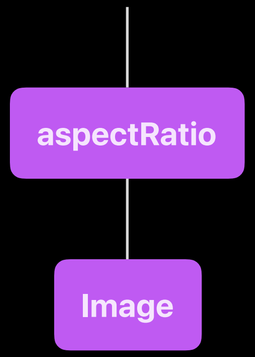


```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```

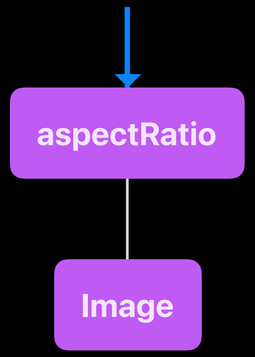


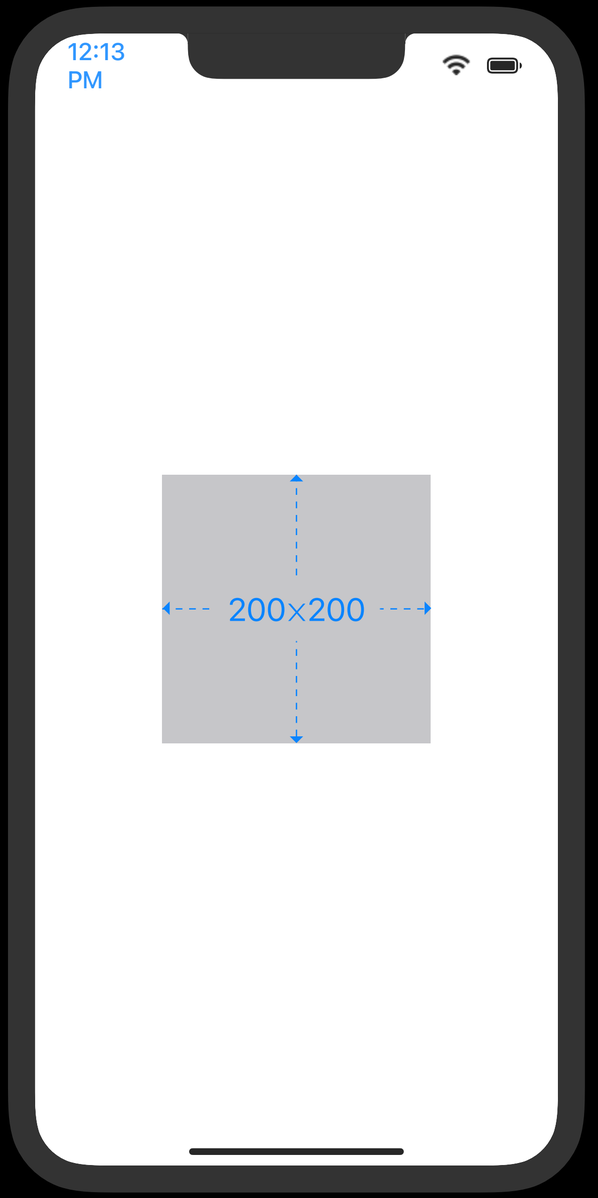


```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```

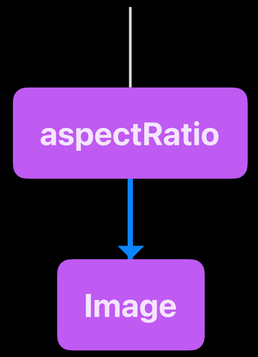


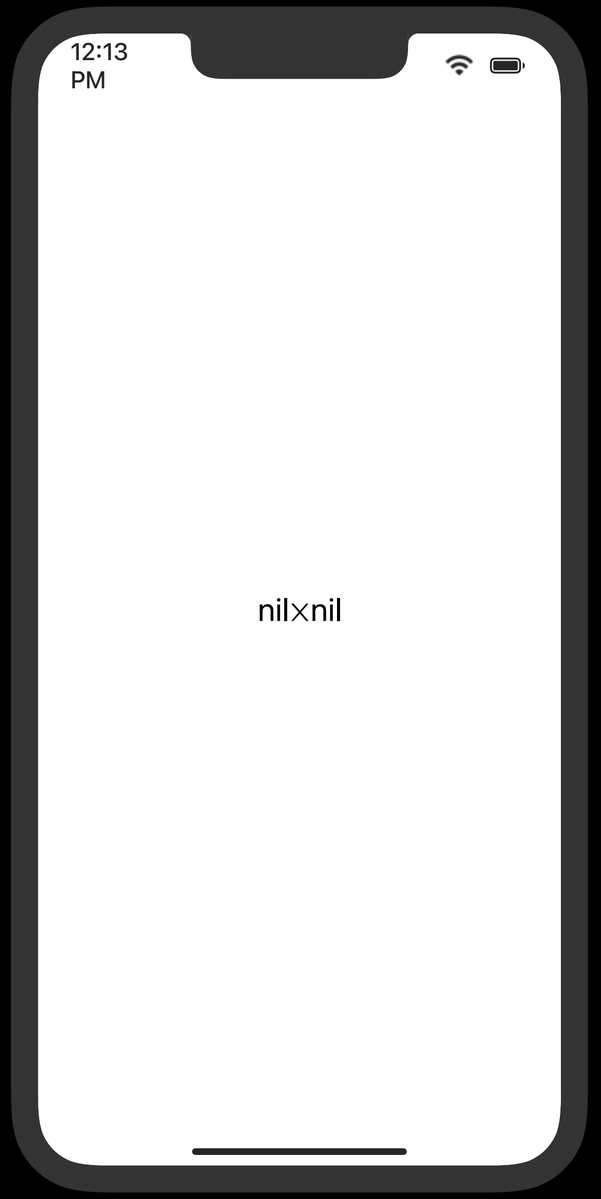
```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



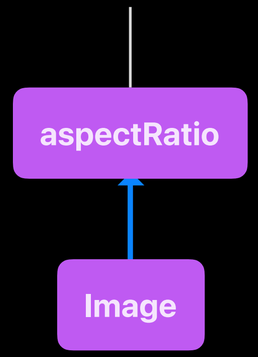


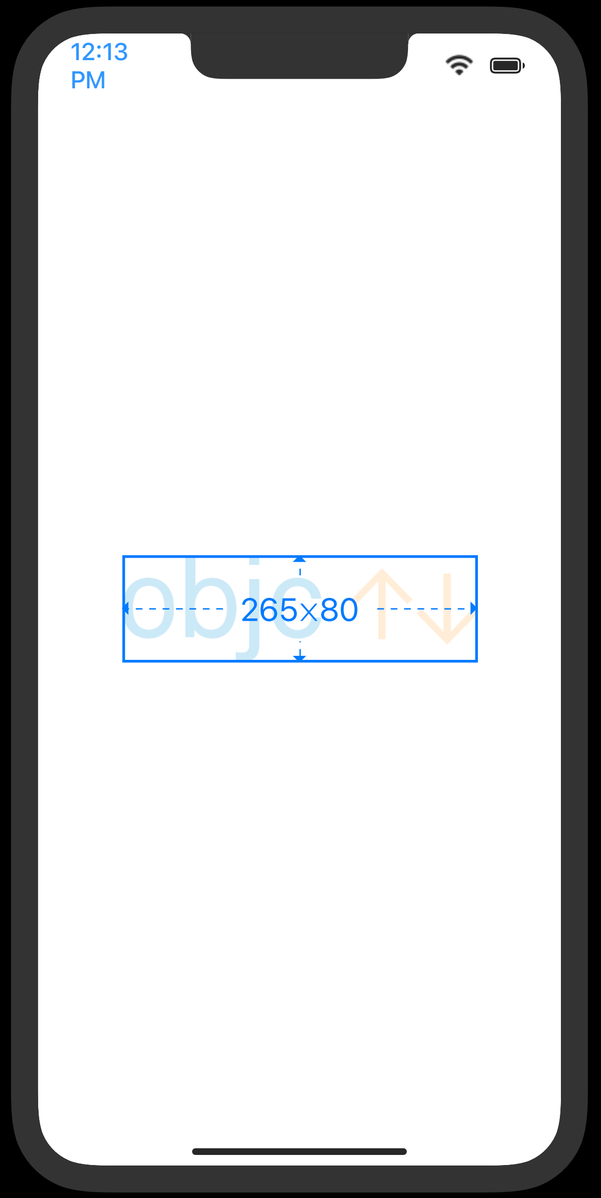
```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



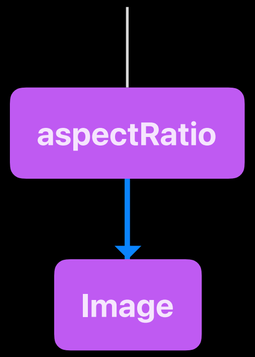


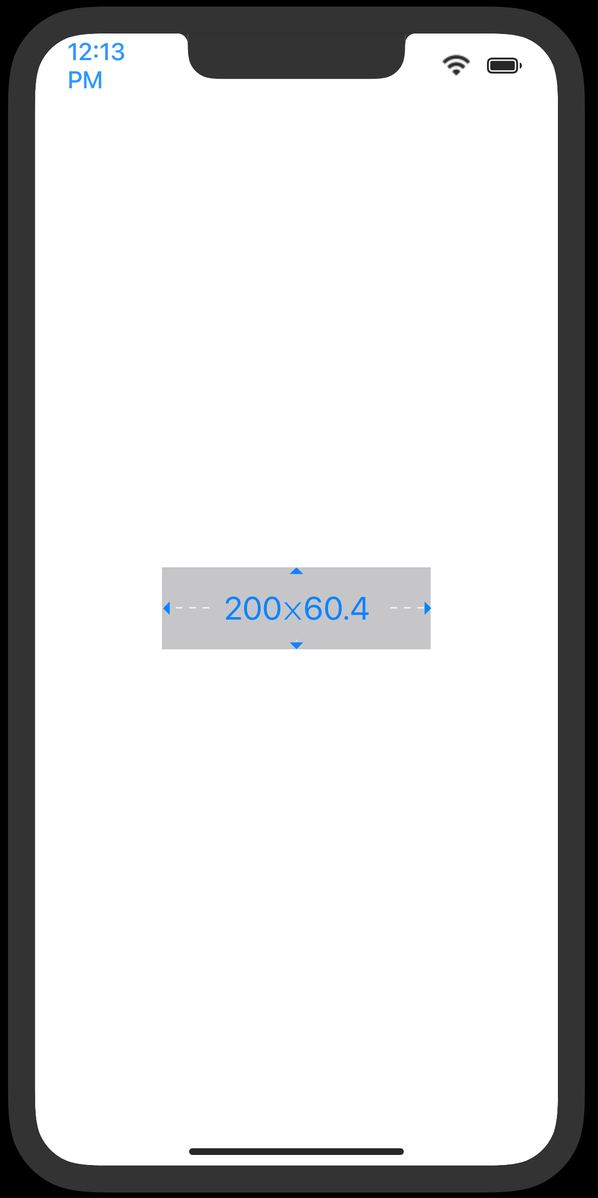
```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



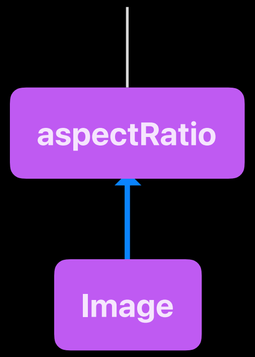


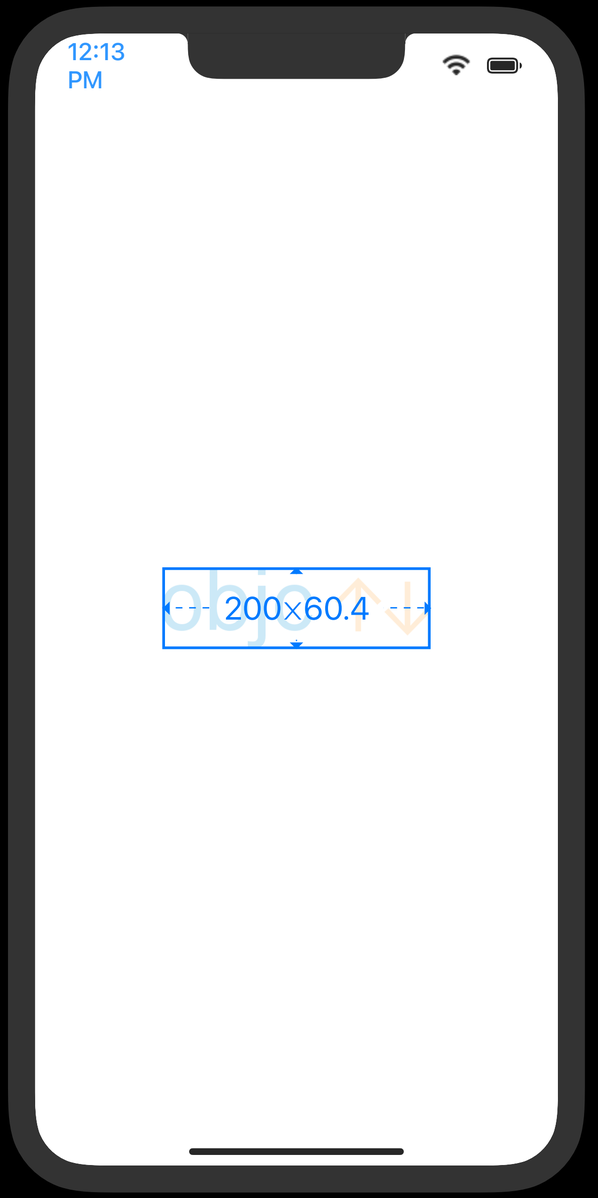
```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



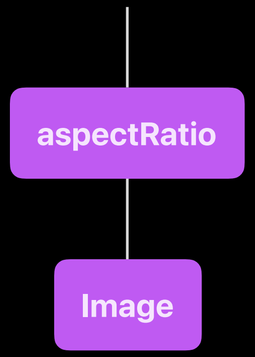


```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```

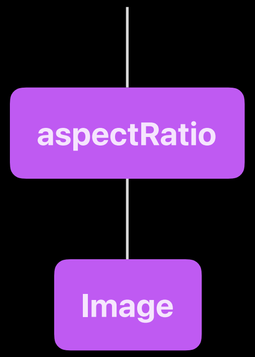


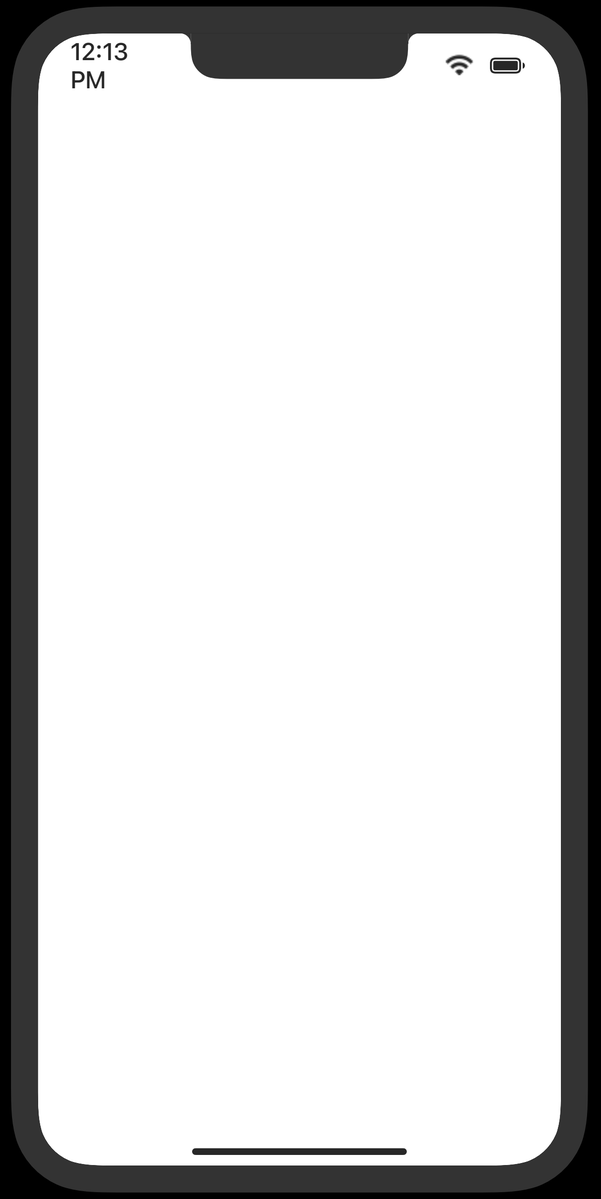


```
0  Image("logo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```

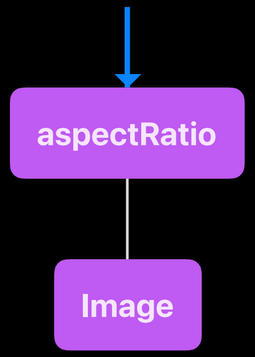


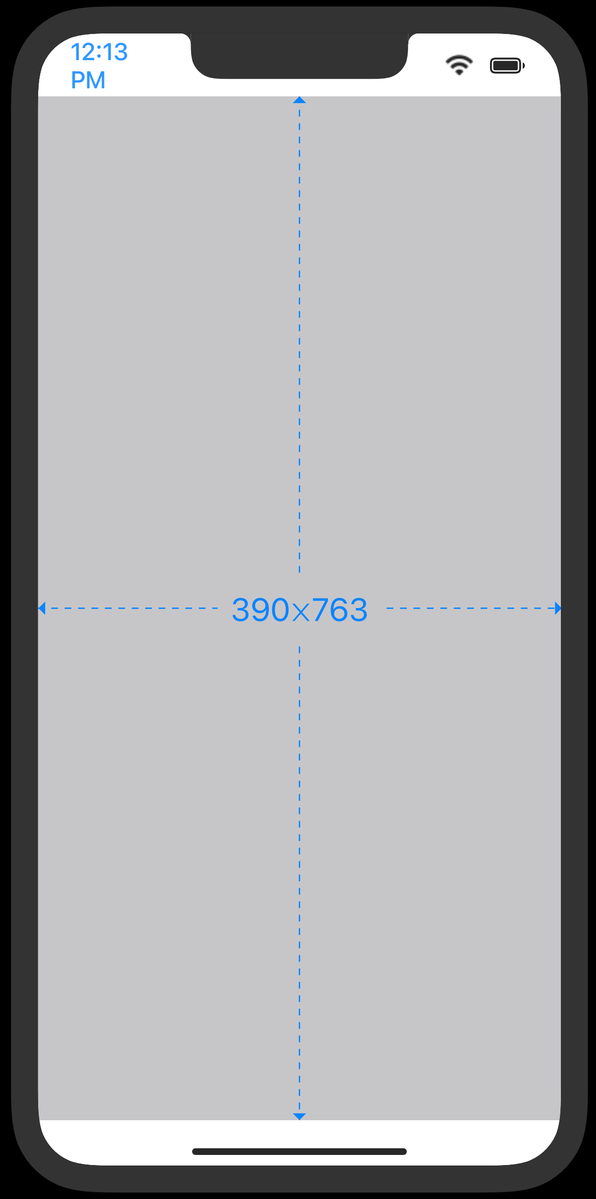
```
Image("photo")
    .resizable()
    .aspectRatio(contentMode: .fit)
```



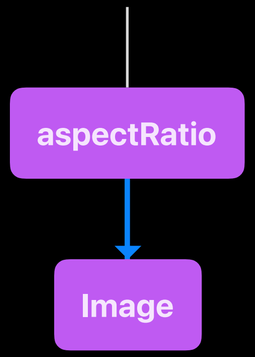


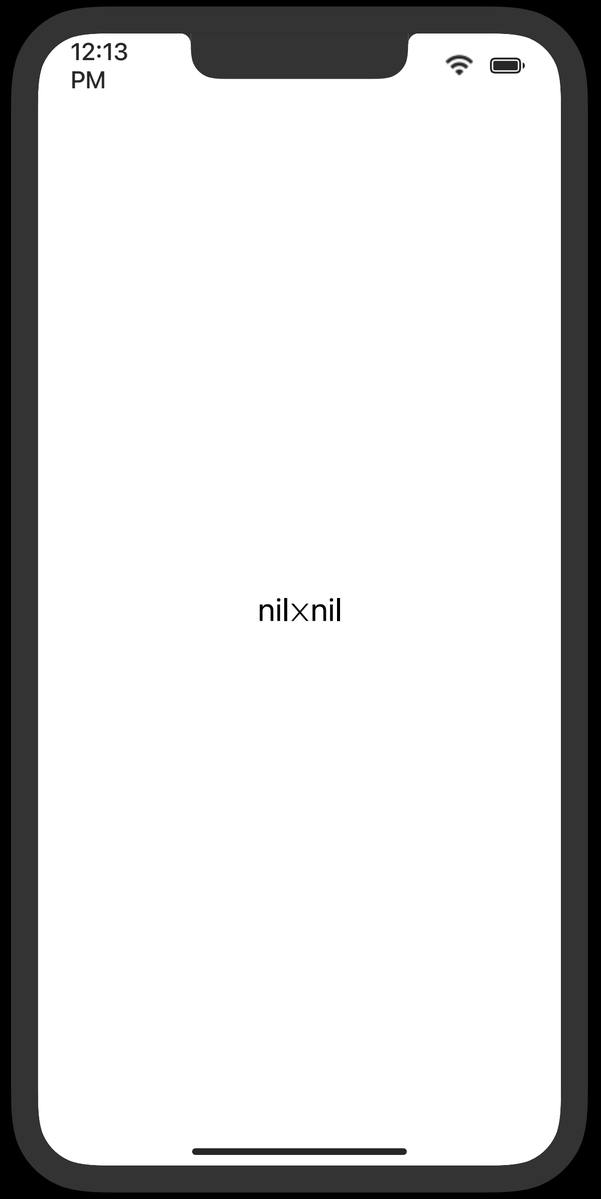
```swift
Image("photo")
    .resizable()
    .aspectRatio(contentMode: .fit)
```



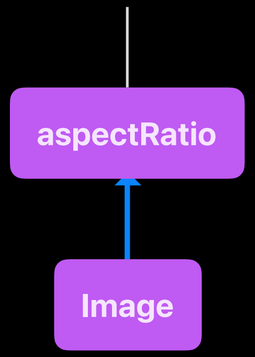


```
0  Image("photo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```

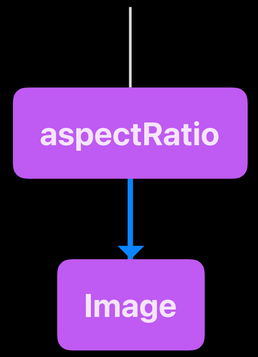


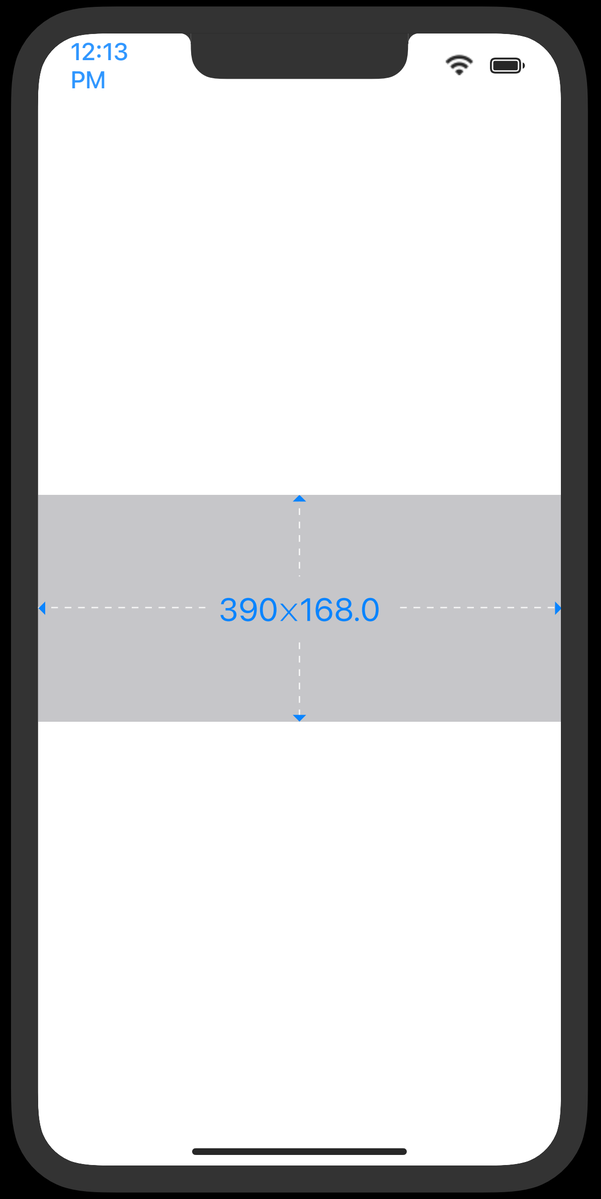


```
0  Image("photo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```

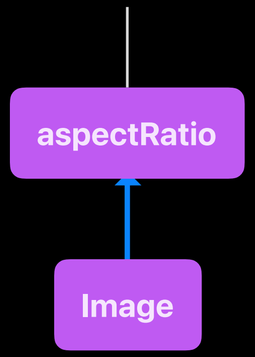


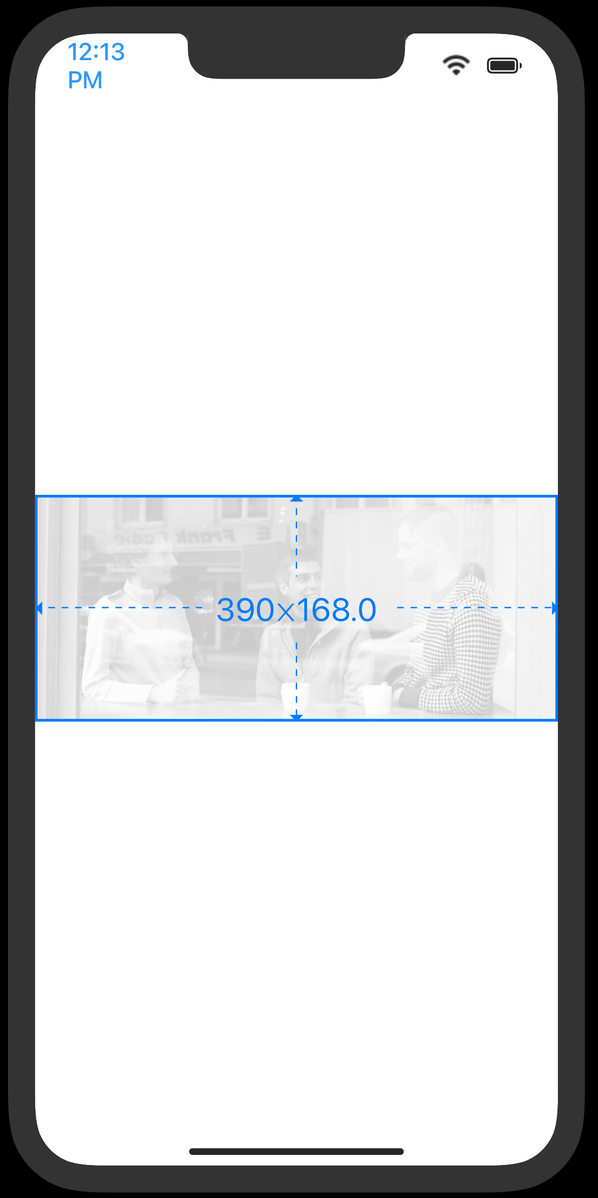
```swift
Image("photo")
    .resizable()
    .aspectRatio(contentMode: .fit)
```



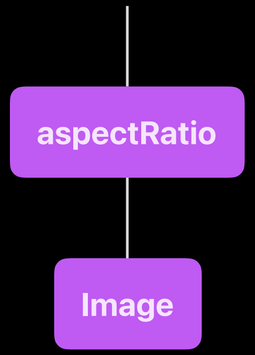


```
0  Image("photo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```





```
0  Image("photo")
1      .resizable()
2      .aspectRatio(contentMode: .fit)
```



```swift
struct MyAsyncImage: View {
    var url: URL
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    var resizable: Bool = false
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

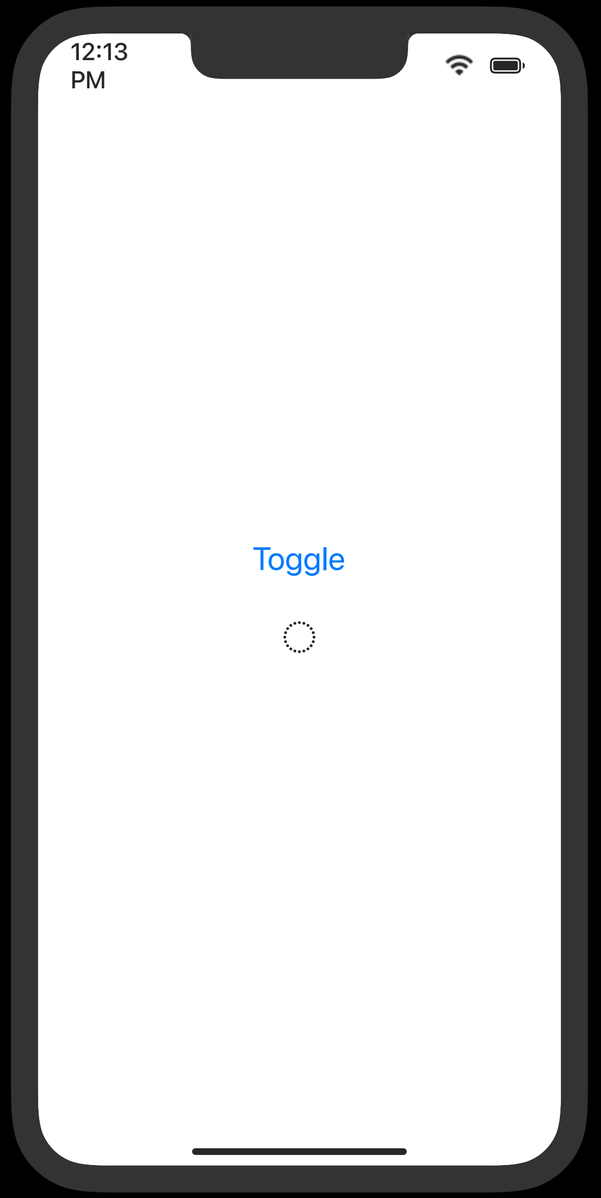
```
struct MyAsyncImage: View {
    var url: URL
    var resizable: Bool = false
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                Image(nsImage: i)
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```
struct MyAsyncImage: View {
    var url: URL
    var resizable: Bool = false
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                if resizable {
                    Image(nsImage: i)
                        .resizable()
                } else {
                    Image(nsImage: i)
                }
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```
struct MyAsyncImage: View {
    var url: URL
    private var _resizable: Bool = false
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                if _resizable {
                    Image(nsImage: i)
                        .resizable()
                } else {
                    Image(nsImage: i)
                }
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }
}
```

```swift
struct MyAsyncImage: View {
    var url: URL
    private var _resizable: Bool = false
    @State private var imageData: Data? = nil

    var body: some View {
        ZStack {
            if let d = imageData, let i = NSImage(data: d) {
                if _resizable {
                    Image(nsImage: i)
                        .resizable()
                } else {
                    Image(nsImage: i)
                }
            } else {
                ProgressView()
            }
        }.task(id: url) {
            imageData = try? await URLSession.shared.data(from: url).0
        }
    }

    func resizable() -> Self {
        var copy = self
        copy._resizable = true
        return copy
    }
}
```

```swift
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
        }
    }
}
```



```swift
struct ContentView: View {
    @State private var cond = true
    var body: some View {
        VStack {
            Button("Toggle") { cond.toggle() }
            MyAsyncImage(url: cond ? logo : photo)
                .resizable()
                .aspectRatio(contentMode: .fit)
        }
    }
}
```