SWIFT BEST PRACTICES

WRAP UP

You've made it to the end? Good job! Well, I'm just sort of *assuming* you've actually worked through the previous documents rather than just skipping to the end, because only a cheater would do that – and you're not a cheater, are you? No, of course not.

As you've been working through the Swift best practices in this workshop, you should have noticed the amount of code in your view controllers has shrunk dramatically – they do far less work now. However, at the same time we've just moved that work elsewhere: rather than deleting stuff, we've just moved it elsewhere.

So: is this really an improvement if most of what we've done is shuffle code around?

Yes.

First, we've reduced the complexity of our code: we no longer have one class that does four different things, and instead have four classes that do one thing each – that's the Single Responsibility Principle, which is the S in SOLID. This makes each unit of code easier to understand and reason about.

Second, our code is more reusable. For example, one of the challenges required you to create a custom class that conforms to **WKNavigationDelegate** so you can handle which pages a web browser can visit. Because that's now a separate object you can use that delegate in any other parts of your app to get the same behavior, and if you ever want to unlock the browser you can just remove the object from your web view – no need to write conditions checking every time a request is made.

Third, we've taken code out of view controllers and put it into individual types. Any code that is in a view controller is immediately much harder to test, because view controllers involve user interfaces. Once code is out of there and into your own types you have complete control, so you can write simpler unit tests.

Any one of the three improvements is worth fighting for, but when you put all three together I hope you can see that app architectures are powerful things.

Thank you for coming to my workshop! If you didn't get through every challenge that's OK – you have your code and you have the PDFs, so hopefully you can continue on your own time. And if you did get through every challenge and still have time to spare, here's your final challenge:

Now that most of the app's functionality is neatly split up, can you write some tests?

Tip: if you find code that's hard to test, it's a good sign it needs to be rewritten.