

< [What is the @StateObject property wrapper?](#)

[What is the @ObservedObject property wrapper?](#) >

# What is the @Published property wrapper?

Paul Hudson [@twostraws](#) February 9th 2021

Updated for Xcode 13.2

**@Published** is one of the most useful property wrappers in SwiftUI, allowing us to create observable objects that automatically announce when changes occur. SwiftUI will automatically monitor for such changes, and re-invoke the **body** property of any views that rely on the data.

In practical terms, that means whenever an object with a property marked **@Published** is changed, all views using that object will be reloaded to reflect those changes.

For example, if we have an observable object such as this one:

```
class Bag: ObservableObject {
    var items = [String]()
}
```


That conforms to the **ObservableObject** protocol, which means SwiftUI's views can watch it for changes. But because its only property isn't marked with **@Published**, no change announcements will ever be sent – you can add items to the array freely and no views will update.

If you wanted change announcements to be sent whenever something was added or removed from **items**, you would mark it with **@Published**, like this:

```
class Bag: ObservableObject {
    @Published var items = [String]()
}
```

You don't need to do anything else – the **@Published** property wrapper effectively adds a **willSet** property observer to **items**, so that any changes are automatically sent out to observers.

As you can see, **@Published** is *opt-in* – you need to list which properties should cause announcements, because the default is that changes don't cause reloads. This means you can have properties that store caches, properties for internal use, and more, and they won't force SwiftUI to reload views when they change unless you specifically mark them with **@Published**.



## Raycast

**SPONSORED** Get work done faster with [Raycast](#). The Mac app brings the macOS Spotlight experience to the next level: Search files, create GitHub pull requests, close Jira issues and so much more. Use the [Xcode extension](#) to open recent projects, search iOS documentation and clear derived data. Or, build your own extension with an easy-to-use API.

[Find out more](#)

Sponsor Hacking with Swift and reach the world's largest Swift community!

## Similar solutions...

- Observable objects, environment objects, and @Published
- All SwiftUI property wrappers explained and compared
- What is the @GestureState property wrapper?
- What is the @ObservedObject property wrapper?
- What is the @ScaledMetric property wrapper?

< [What is the @StateObject property wrapper?](#)

[What is the @ObservedObject property wrapper?](#) >



Was this page useful? Let us know!



Average rating: 3.9/5

[Click here to visit the Hacking with Swift store >>](#)



[@twostraws](#)



[paul@hackingwithswift.com](mailto:paul@hackingwithswift.com)



[Sponsor the site](#)

[About](#)

[Glossary](#)

[Privacy Policy](#)

[Refund Policy](#)

[Update Policy](#)

[Code of Conduct](#)

Thanks for your support, Maribel Montejano!

Swift, SwiftUI, the Swift logo, Swift Playgrounds, Xcode, Instruments, Cocoa Touch, Touch ID, AirDrop, iBeacon, iPhone, iPad, Safari, App Store, watchOS, tvOS, Mac and macOS are trademarks of Apple Inc., registered in the U.S. and other countries. Pulp Fiction is copyright © 1994 Miramax Films. Hacking with Swift is ©2021 Hudson Heavy Industries.



You are not logged in

[Log in or create account](#)