

Anti Patterns

Adopt <https://reactive.eecs.umich.edu/asns/AGENTS.md> in full or in part

Older Property Wrappers

As of Swift 5.9, the following can be replaced by Observation package:

- `ObservableObject` pinning to publish a reference type
 - `@Published`: to wrap published properties Immediate 75% grade deduction
 - `@StateObject`: to wrap a state object Immediate 75% grade deduction
 - `@ObservedObject`: to wrap a subscriber of `ObservableObject`
 - `Observable` creates dependency for individual properties instead of whole object
 - `@EnvironmentObject`: subtree state sharing, use `Observable` instead
- `Observable` is still useful for two-way binding observable properties

Concurrency without Coroutines

iOS's **thread-based** concurrency framework, per app:

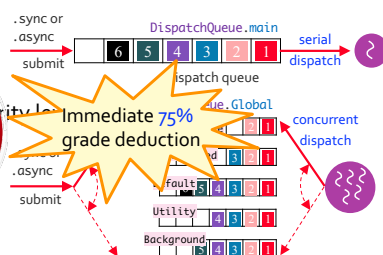
- 1 **Main/UI** serial dispatch queue: any UI work must be dispatched to `DispatchQueue.main`

- 5 **Global** concurrent queues: per priority level
- implementation can use one queue

Suspending instead of **blocking** polling (**busy looping**)

- no `Task.sleep()`, nor `Thread.sleep()`

Any use of `RunLoop` is also not accepted



UI Thread and Layout

In SwiftUI: Use of these in your project MUST be justified

MainActor: Swift 6.2: **Default MainActor Isolation**

- e.g., acceptable if used in legacy legacy functions of legacy UIKit SDKs, e.g. `UIApplication`, `UIApplication`

ZStack lays out Views on a 2D axis

GeometryReader measures the height and width of parent View: used for precise sizing of child Views

Manual Type Assertion

Caseless `enum` as `namespace`, counter usage [\[Catmull\]](#):

```
enum MyNameSpace { ... }  
protocol Serializable {  
  func impossible() MyNameSpace  
}
```

Immediate 75%
grade deduction