Escaping closures

By definition, closures cannot live longer than the function it was passed into as a parameter (non-escaping). Otherwise, it is called an escaping closure. If there's a reference to self in the escaping closure (strong reference cycle), it won't get deallocated, causing a memory leak. This can be solved by making a weak reference to self (capture list), which is an optional type and requires unwrapping.

Codable vs. Serialization

Because JSONDecoder uses JSONSerialization in its internal workings, the latter is obviously faster but not necessarily the best option. The best practice in this case is to benchmark the performance of both and also consider that Swift Codable is much more flexible and easier to use with data representations because of CodingKeys.

Mauro Arantes