极客学院
jikexueyuan.com

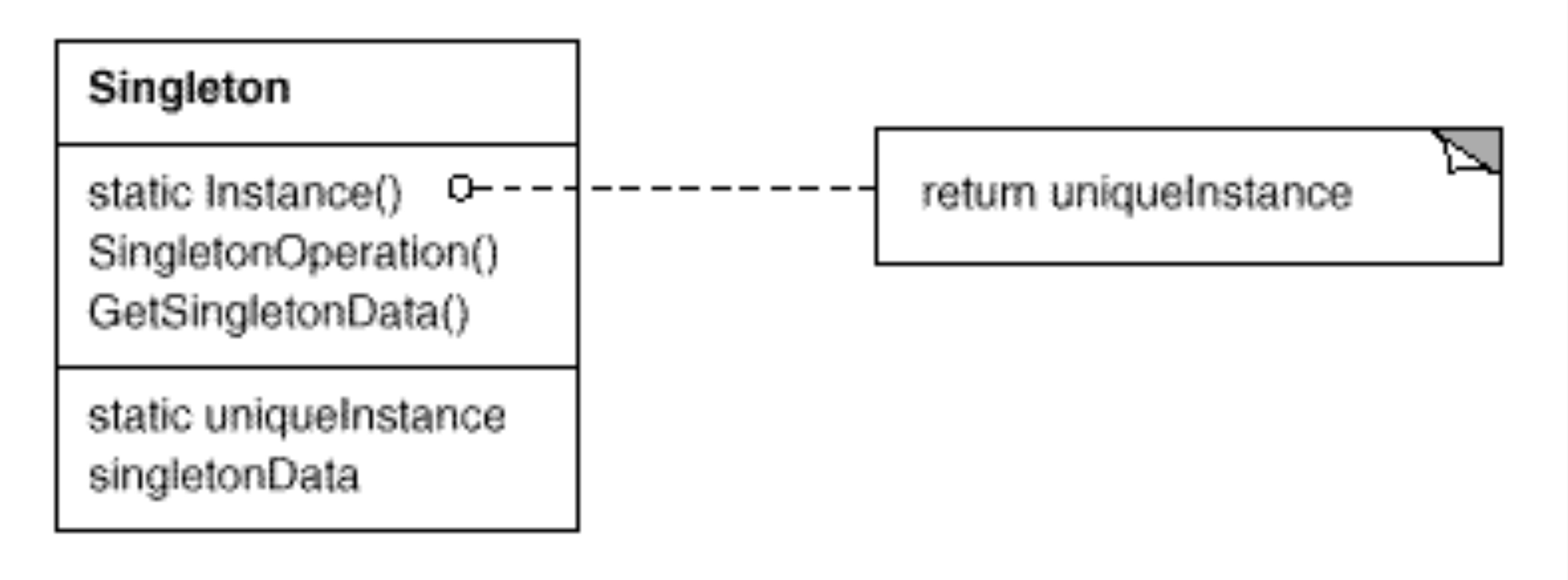
# iOS设计模式 – 单例

# iOS设计模式 – 单例 — **课程概要**

- 单例模式
- 编写严格的单例
- 优化本地存储

# 单例模式

# 单例模式


Singleton
static Instance()
SingletonOperation()
GetSingletonData()
static uniqueInstance
singletonData
return uniqueInstance

# 单例模式

- 单例模式基本原理
- 单例模式用以解决何种问题
- 单例模式的优缺点

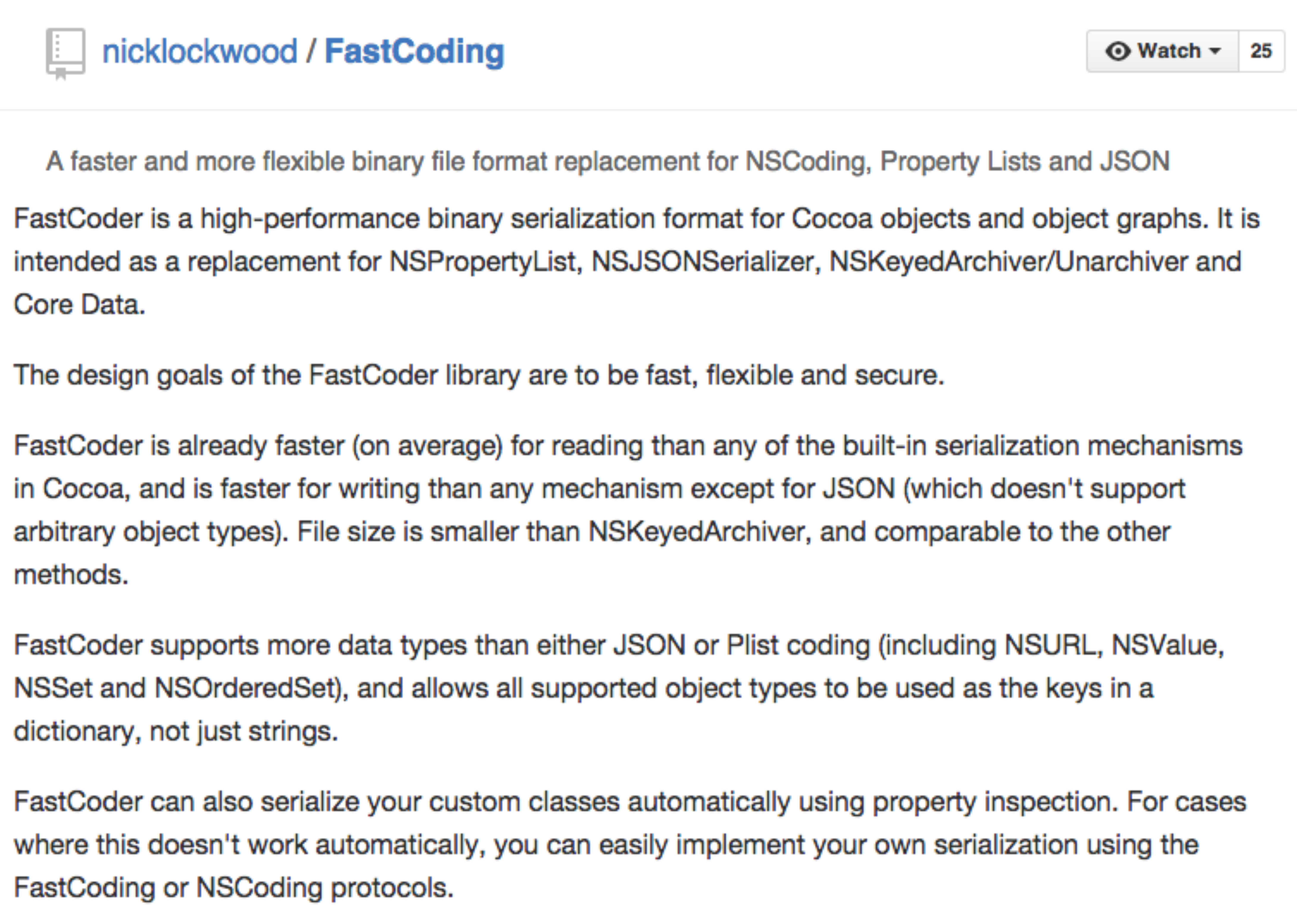
# 编写严格的单例

# 编写严格的单例

- 如何防止继承
- 如何确保实例对象只出现一个

# 优化本地存储

# 优化本地存储



nicklockwood / **FastCoding**

A faster and more flexible binary file format replacement for NSCoding, Property Lists and JSON

FastCoder is a high-performance binary serialization format for Cocoa objects and object graphs. It is intended as a replacement for NSPropertyList, NSJSONSerializer, NSKeyedArchiver/Unarchiver and Core Data.

The design goals of the FastCoder library are to be fast, flexible and secure.

FastCoder is already faster (on average) for reading than any of the built-in serialization mechanisms in Cocoa, and is faster for writing than any mechanism except for JSON (which doesn't support arbitrary object types). File size is smaller than NSKeyedArchiver, and comparable to the other methods.

FastCoder supports more data types than either JSON or Plist coding (including NSURL, NSValue, NSSet and NSOrderedSet), and allows all supported object types to be used as the keys in a dictionary, not just strings.

FastCoder can also serialize your custom classes automatically using property inspection. For cases where this doesn't work automatically, you can easily implement your own serialization using the FastCoding or NSCoding protocols.

# 优化本地存储

- 用单例设计存储数据接口
- 用单例接口隔离实现细节
- 在单例提供接口的基础上进行上层封装

# iOS设计模式 – 单例

本套课程中我们学习了 iOS 设计模式 – 单例模式，你应当掌握了以下知识：

- 单例模式
- 编写严格的单例
- 优化本地存储

你可以用所学的知识来重新 review 以前所写的代码，进行适当的改进，如果想继续提高，你可以继续在极客学院学习iOS设计模式 – 备忘录。