rules.sonarsource.com

Swift static code analysis: URIs should not be hardcoded

2-3 minutes

Hard coding a URI makes it difficult to test a program: path literals are not always portable across operating systems, a given absolute path may not exist on a specific test environment, a specified Internet URL may not be available when executing the tests, production environment filesystems usually differ from the development environment, …etc. For all those reasons, a URI should never be hard coded. Instead, it should be replaced by customizable parameter.

Further even if the elements of a URI are obtained dynamically, portability can still be limited if the path-delimiters are hard-coded.

This rule raises an issue when URI's or path delimiters are hard coded.

Noncompliant Code Example

```
public class Foo {
   public func listUsers() -> [User] {
      var users:[User]
      let location = "/home/mylogin/Dev/users.txt"      // Non-
Compliant
      let fileContent = NSString(contentsOfFile: location, encoding:
NSUTF8StringEncoding, error: nil)
      users = parse(fileContent!)
      return users
   }
}
```

Compliant Solution

```
public class Foo {
   // Configuration is a class that returns customizable properties: it
can be mocked to be injected during tests.
   private var config:Configuration
   public init(myConfig:Configuration) {
      config = myConfig
   }
   public func listUsers() -> [User] {
      var users:[User]
      // Find here the way to get the correct folder, in this case using
the Configuration object
      let location = config.getProperty("myApplication.listingFile")
      // and use this parameter instead of the hard coded path
      let fileContent = NSString(contentsOfFile: location, encoding:
NSUTF8StringEncoding, error: nil)
      users = parse(fileContent!)
      return users
   }
}
```