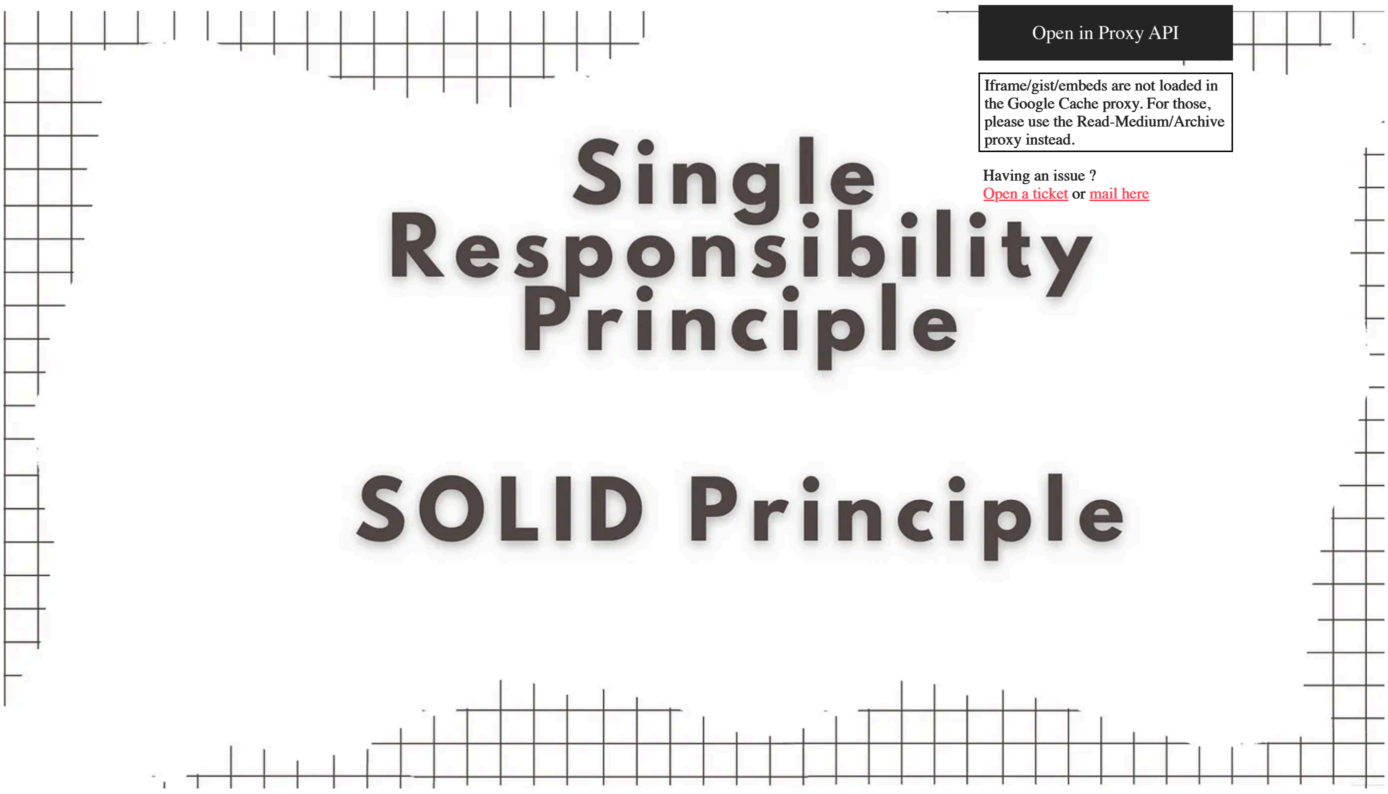
1. Single Responsibility Principle: SOLID Principle

Ramdhas · Follow
2 min read · Oct 21, 2023

Listen Share More



The Single Responsibility Principle (SRP) states that a class should have only one primary responsibility. This principle encourages you to keep your classes focused and avoid combining multiple. Here's a simple explanation with an example:

Example: Let's say we have a class called `Book` that represents a book's information. Applying the Single Responsibility Principle would mean that the `Book` class should have only one responsibility, such as storing book details, but it should not handle unrelated tasks like formatting book titles for display.

Incorrect (Violating Single Responsibility Principle):

```
class Book {
    var title: String
    var author: String
    var price: Double

    func displayBookInfo() {
        // Code for formatting and displaying book info
    }

    func calculateDiscountedPrice() {
        // Code for applying discounts
    }
}
```

In the above example, the `Book` class violates the Single Responsibility Principle because it is responsible for both storing book details and formatting/displaying book information, as well as calculating discounts. If any of these responsibilities change, you'd need to modify the same class.

Correct (Following Single Responsibility Principle)**:**

```
class Book {
    var title: String
    var author: String
    var price: Double
}

class BookFormatter {
    static func displayBookInfo(book: Book) {
        // Code for formatting and displaying book info
    }
}
class PriceCalculator {
    static func calculateDiscountedPrice(book: Book) -> Double {
        // Code for applying discounts
    }
}

let myBook = Book(title: "Sample Book", author: "John Doe", price: 29.99)

let formattedInfo = BookFormatter.displayBookInfo(book: myBook)
print(formattedInfo)

let discountedPrice = PriceCalculator.calculateDiscountedPrice(book: myBook)
print("Discounted Price: $\(discountedPrice)")
```

In this improved example, we have separate classes (`BookFormatter` and `PriceCalculator`) that handle the specific responsibilities of formatting book information and calculating discounts. The `Book` class is only responsible for storing book details, following the Single Responsibility Principle. This makes the code easier to understand, maintain, and extend because each class has a single, well-defined purpose.

. . .

Read more about: 2. Open/Closed Principle(OCP) : SOLID Principle

https://medium.com/@ramdhasm5/2-open-closed-principle-ocp-solid-principle-cd12cbc6cb6e

. . .

👏🏻👏🏻 👏🏻👏🏻 Applaud to express your encouragement. Join me for additional insights and let's progress together.

Solid   Clean Code   Solid Principles   Single Responsibility   Code Quality

Follow

Written by Ramdhas

225 Followers · 31 Following

Skills: iOS, Swift, SwiftUI, Html, Css, Javascript, React.js. Lives in Stockholm, Sweden.

No responses yet

What are your thoughts?

Respond