单一职责原则

# 定义

单一职责原则（SRP：Single responsibility principle）又称单一功能原则，单一职责原则规定：一个类应该只有一个发生变化的原因（There should never be more than one reason for a class to change）。所谓职责是指类发生变化的原因。

# 原理

如果一个类承担的职责过多，职责间可能会相互[耦合](https://baike.baidu.com/item/%E8%80%A6%E5%90%88/2821124" \t "https://baike.baidu.com/item/_blank)。一个职责的变化可能会对其他职责造成影响。想要避免这种现象的发生，就要尽可能的遵守单一职责原则。

# 核心

单一职责原则的核心是职责解耦和增强类的内聚性。

# 必要性

单一职责原则的本意是限制一个类不要承担太多职责，如果一个类承担了太多的职责，将至少存在以下缺点：

1. 职责耦合度高。一个类承担多个职责，这些职责很可能会耦合在一起，一个 职责的改变，可能会影响到其他职责；

2. 类的复用性低。粒度越大的类，复用性越低；

3. 职责调用成本高。当客户端需要该对象的某一个职责时，不得不将其他不需 要的职责全都包含进来，从而造成冗余代码或代码的浪费。

# 作用

遵循单一职责原则在软件系统中具有下列作用：

1. 降低类的复杂度。一个类只负责一项职责，其逻辑肯定要比负责多项职责简单得多。
2. 提高类的可读性。复杂性降低，自然其可读性会提高。
3. 提高系统可维护性。可读性提高，那自然更容易维护了。
4. 降低变更引起的风险。变更是必然的，如果单一职责原则遵守得好，当修改一个功能时，可以显著降低对其他功能的影响。
5. 提高类的可复用性。类的粒度越低，可复用性越强。

# 实现方法

单一职责原则是最简单但又最难运用的原则，需要设计人员发现类的不同职责并将其分离，再封装到不同的类或模块中。而发现类的多重职责需要设计人员具有较强的分析设计能力和相关重构经验。

# 其他

## 单一职责原则同样适用于方法

一个方法应该尽可能做好一件事情，如果一个方法处理的事情太多，其颗粒度将变得很大，不利于重用。