单一职责原则

单一职责强调的是职责的分离，在某种程度上对职责的理解，构成了不同类之间耦合关系的设计关键，因此单一职责原则或多或少成为设计过程中一个必须考虑的基础性原则。

产生这一原则的原因是：没有任何的程序设计人员不清楚应该写出高内聚低耦合的程序，但是很多耦合常常发生在不经意之间。其原因就是：职责扩散：因为某种原因，某一职责被分化为颗粒度更细的多个职责了。

单一职责原则的核心思想是 ：一个类，最好只做一件事情，只有一个引起它变化的原因。单一职责原则可以看做是低耦合，高内聚在面向对象原则上的引申，将职责定义为引起变化的原因，以提高内聚性来减少引起变化的原因。职责过多，可能引起它变化的原因就越多，这将导致依赖，相互之间就产生影响，从而极大的损伤其内聚性和耦合性。单一职责，通常意味着单一的功能，因此不要为类实现过多的功能点，以保证实体只有一个引起它变化的原因。在对于违反这一个原则的类的情况中应该进行重构。

一个类只有一个引起它变化的原因，否则就应当考虑重构。SRP由引起变化的原因决定，而不由功能决定。虽然职责常常引起变化的轴线，但有时就未必，应该适当考虑。可以通过[Facade模式](http://www.cnblogs.com/binb/archive/2012/08/05/zoubin_Nhibernate.html)或Proxy模式进行职责分离。

在实际的项目过程中间要不断的去重构自己的代码，多读一些对提高自己能力有益的书。