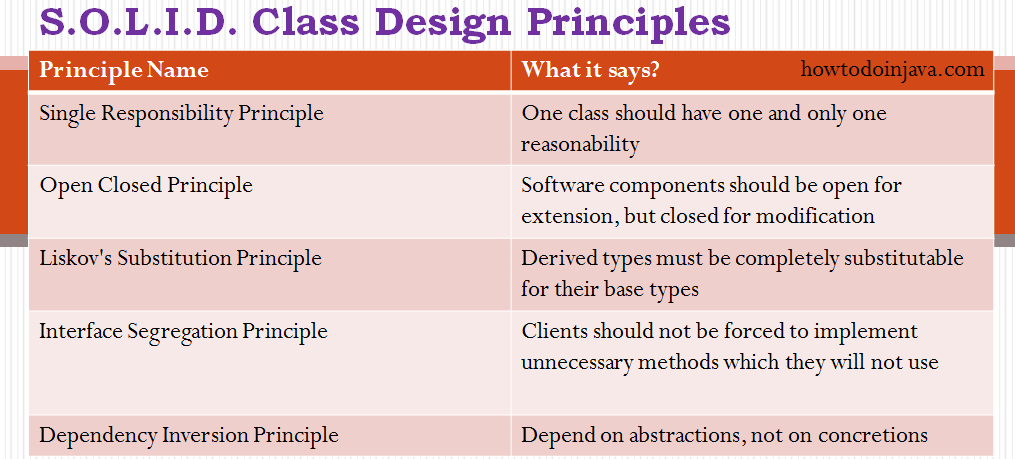
5 Class Design Principles [S.O.L.I.D.] in Java



First Five principles named by  [*Robert C. Martin*](https://en.wikipedia.org/wiki/Robert_C._Martin)(“Uncle Bob”)

The first five principles are principles of *class design*. They are:

|  |  |  |
| --- | --- | --- |
| **SRP** | [The Single Responsibility Principle](https://docs.google.com/open?id=0ByOwmqah_nuGNHEtcU5OekdDMkk) | *A class should have one, and only one, reason to change.* |
| **OCP** | [The Open Closed Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgN2M5MTkwM2EtNWFkZC00ZTI3LWFjZTUtNTFhZGZiYmUzODc1&hl=en) | *You should be able to extend a classes behavior, without modifying it.* |
| **LSP** | [The Liskov Substitution Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgNzAzZjA5ZmItNjU3NS00MzQ5LTkwYjMtMDJhNDU5ZTM0MTlh&hl=en) | *Derived classes must be substitutable for their base classes.*  *(use design by contract)*  Example :  User, Based, Derived, example.  void User(Base& b);  Derived d;  User(d); |
| **ISP** | [The Interface Segregation Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgOTViYjJhYzMtMzYxMC00MzFjLWJjMzYtOGJiMDc5N2JkYmJi&hl=en) | *Make fine grained interfaces that are client specific.* |
| **DIP** | [The Dependency Inversion Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgMjdlMWIzNGUtZTQ0NC00ZjQ5LTkwYzQtZjRhMDRlNTQ3ZGMz&hl=en) | *Depend on abstractions, not on concretions.* |

The next six principles are about packages. In this context a package is a binary deliverable like a .jar file, or a dll as opposed to a namespace like a java package or a C++ namespace.  
  
The first three package principles are about package *cohesion*, they tell us what to put inside packages:

|  |  |  |
| --- | --- | --- |
| **REP** | [The Release Reuse Equivalency Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgOGM2ZGFhNmYtNmE4ZS00OGY5LWFkZTYtMjE0ZGNjODQ0MjEx&hl=en) | *The granule of reuse is the granule of release.* |
| **CCP** | [The Common Closure Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgOGM2ZGFhNmYtNmE4ZS00OGY5LWFkZTYtMjE0ZGNjODQ0MjEx&hl=en) | *Classes that change together are packaged together.* |
| **CRP** | [The Common Reuse Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgOGM2ZGFhNmYtNmE4ZS00OGY5LWFkZTYtMjE0ZGNjODQ0MjEx&hl=en) | *Classes that are used together are packaged together.* |

The last three principles are about the couplings between packages, and talk about metrics that evaluate the package structure of a system.

|  |  |  |
| --- | --- | --- |
| **ADP** | [The Acyclic Dependencies Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgOGM2ZGFhNmYtNmE4ZS00OGY5LWFkZTYtMjE0ZGNjODQ0MjEx&hl=en) | *The dependency graph of packages must have no cycles.* |
| **SDP** | [The Stable Dependencies Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgZjI3OTU4ZTAtYmM4Mi00MWMyLTgxN2YtMzk5YTY1NTViNTBh&hl=en) | *Depend in the direction of stability.* |
| **SAP** | [The Stable Abstractions Principle](http://docs.google.com/a/cleancoder.com/viewer?a=v&pid=explorer&chrome=true&srcid=0BwhCYaYDn8EgZjI3OTU4ZTAtYmM4Mi00MWMyLTgxN2YtMzk5YTY1NTViNTBh&hl=en) | *Abstractness increases with stability.* |

