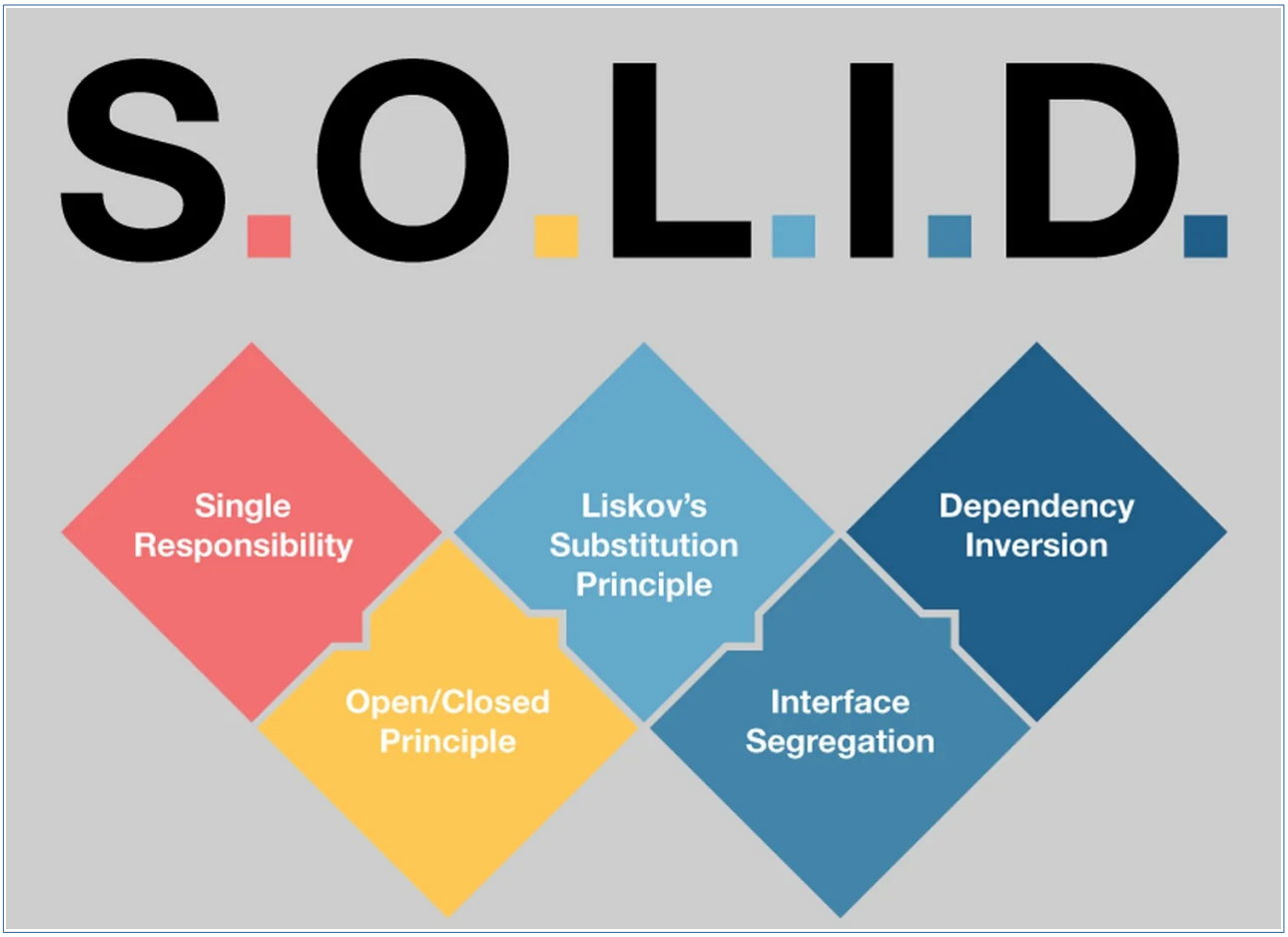

SOLID-Prinzipien:

1. S – Single Responsibility Principle (S.R.P) – Prinzip der eindeutigen Zuständigkeit:

- **Jede Klasse sollte nur eine Verantwortung haben.** Das bedeutet, dass jede Klasse nur eine Aufgabe oder **einen Grund zur Änderung haben sollte**. Dadurch wird der Code einfacher zu verstehen, zu testen und zu warten.

2. O – Open/Closed Principle (O.C.P) – Offen/geschlossenes Prinzip:

- **Software-Module sollten offen für Erweiterungen, aber geschlossen für Veränderungen sein.** Dies bedeutet, dass man das Verhalten eines Systems durch das Hinzufügen neuen Codes erweitern können sollte, ohne den bestehenden Code zu ändern. Das führt zu stabileren und flexibleren Systemen.

3. L – Liskov Substitution Principle (L.S.P) – Liskovsche Substitutionsprinzip:

- **Objekte einer Unterklasse sollten durch Objekte ihrer Basisklasse ersetzt werden können, ohne dass der Anwendungskontext beeinträchtigt wird.** Das bedeutet, dass eine Unterklasse alle Eigenschaften der Oberklasse erfüllen muss, um sicherzustellen, dass der Austausch von Objekten reibungslos funktioniert.

4. I – Interface Segregation Principle (I.S.P) – Prinzip der Schnittstellentrennung:

- **Schnittstellen sollten so spezifisch wie möglich sein, sodass Implementierungen nur die Methoden bieten müssen, die sie tatsächlich benötigen.** Große Schnittstellen sollten in kleinere, spezialisierte Schnittstellen aufgeteilt werden, um eine unnötige Abhängigkeit von Methoden zu vermeiden, die ein Objekt nicht benötigt.

5. D – Dependency Inversion Principle (D.I.P) – Prinzip der Abhängigkeitsumkehr:

- **Hohe Module sollten nicht von niedrigen Modulen abhängen, beide sollten von Abstraktionen abhängen.** Abstraktionen sollten nicht von Details abhängen; Details sollten von Abstraktionen abhängen. Dadurch wird der Code flexibler und weniger abhängig von konkreten Implementierungen.

Was ist das?

- sind fünf grundlegende Entwurfsprinzipien **für die objektorientierte Programmierung (OOP)**
- helfen dabei, sauberen, wartbaren und flexiblen Code zu schreiben.