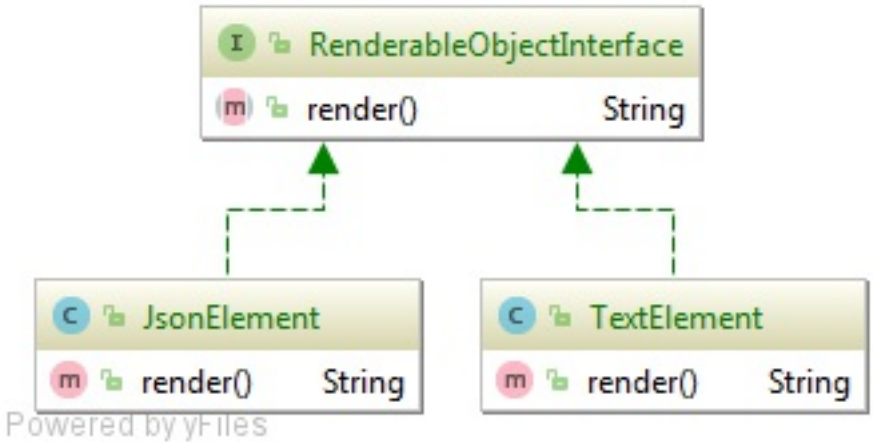
Strukturmuster “Komposition”

Beschreibung

Komposition ermöglicht es, eine Gruppe von Objekten wie eine einzelne Instanz des Objekts zu behandeln: Anstatt mit **konkreten Instanzen eines Objekts** zu arbeiten, wird ein **Interface** verwendet.

- https://stackoverflow.com/a/9071207

Klassendiagramm



Beispielcodierung

Siehe ausführbares Beispiel (example/Main.java), bestehend aus:

1. “Renderable”-Interface
2. “Json”-Klasse
3. “Text”-Klasse

Anwendungsfälle

In der objektorientierten Programmierung werden Objekte für den Programmfluss verwendet; besonders häufig kommen **Parameter- und Returntyps** vor. Sind diese nun fest definiert und als konkrete Klasse implementiert, werden spätere Änderungen unnötig erschwert.

Beispiel: Das catchen aller ExampleException soll geloggt werden. Wer dafür verwendeten LoggingService bekommt diese übergeben (Dependency Injection). Warum nun den LoggingService auf genau ExampleException beschränken? Stattdessen wird ein Interface verwendet.

Nach einiger Zeit sollen zusätzliche Exception-Typs geloggt werden. Da es nun nur eine Abhängigkeit auf ein Interface gibt, ist diese Anforderung schnell umgesetzt.

Vor- und Nachteile

Vererbung verlangt, zu einem sehr frühen Zeitpunkt die genau Verwendung und Aufgaben eines Objektes zu kennen. Gemeinsamkeiten werden zusammengefasst und Spezialisierungen abgekoppelt; die **„IS-A“**-Beziehung: *Dog extends Animal* bedeutet *dog IS-A(n) animal*.

Sind später nun Änderungen erforderlich, auch wenn diese augenscheinlich nur sehr klein sind, muss unter Umständen ein großer

Teil der bestehenden Datenstruktur angepasst werden. Kompostion ermöglicht hier einen flexibleren Ansatz, der spätere Änderungen wesentlich vereinfacht. Bei der **„HAS-A“**-Beziehung wird schlicht auf andere Objekte refenenziert: *LoggingService HAS-A(n) Exception*.

- https://www.w3resource.com/java-tutorial/inheritance-composition-relationship.php
- http://www.artima.com/designtechniques/compoinhP.html

Übungsaufgabe

Ergänze “JsonElement” und “TextElement” um “HtmlElement”. Alternativ schaue dir dieses Video an:

Quellen

- https://github.com/domnikl/DesignPatternsPHP
- https://medium.com/humans-create-software/composition-over-inheritance-cb6f88070205