Visitor-Pattern

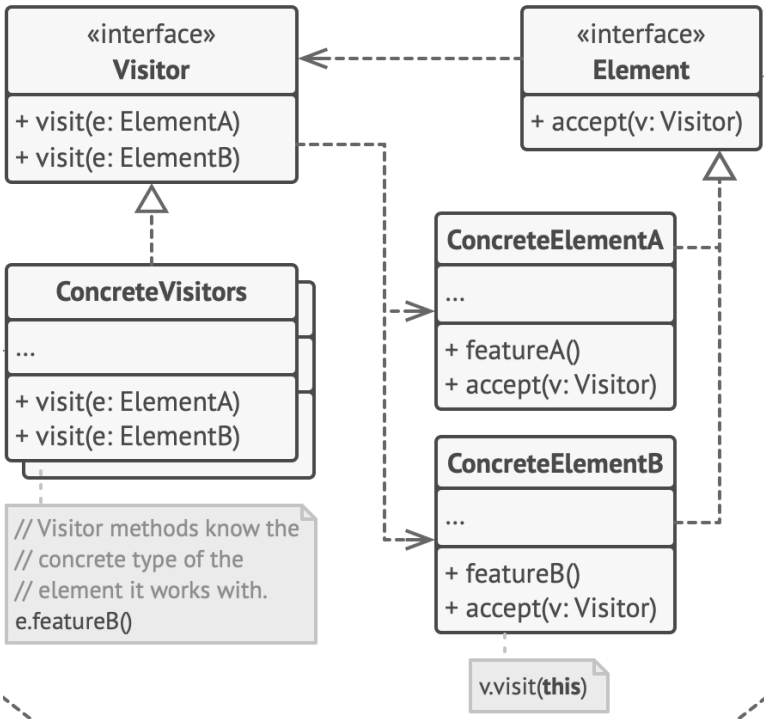
Visitor ist ein verhaltensorientiertes Entwurfsmuster, mit dem Sie Algorithmen von den Objekten trennen können, auf denen sie arbeiten.

Problem:

Wie kann man das Verhalten auf einzelnen Klassen einer Datenstruktur verändern oder erweitern, ohne die einzelnen Klassen zu verändern?

Lösung:

Das Visitor-Muster schlägt vor, das neue Verhalten in einer separaten Klasse zu platzieren, anstatt zu versuchen, es in bestehende Klassen zu integrieren. Das ursprüngliche Objekt, das das Verhalten ausführen musste, wird nun an eine der Methoden des Visitor als Argument übergeben, wodurch die Methode Zugriff auf alle erforderlichen Daten erhält, die im Objekt enthalten sind.



PRO:

- Open/Closed Prinzip: Man kann ein neues Verhalten einführen, das mit Objekten verschiedener Klassen arbeiten kann, ohne diese Klassen zu ändern.
- Single-Responsibility-Prinzip: Man kann mehrere Versionen desselben Verhaltens in dieselbe Klasse verschieben.

Kontra:

- Jedes Mal, wenn eine Klasse zur Elementhierarchie hinzugefügt oder aus ihr entfernt wird, müssen alle Visitors aktualisiert werden.
- Visitors haben möglicherweise nicht den nötigen Zugriff auf die privaten Felder und Methoden der Elemente, mit denen sie arbeiten sollen.