Question 4: State Pattern + JUnit

IT-SDJ2-A21

Software Engineering

VIA University College

Jordi Lazo

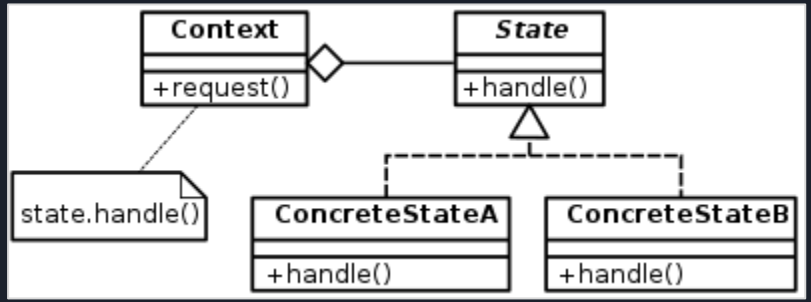
State Pattern

What is the purpose?

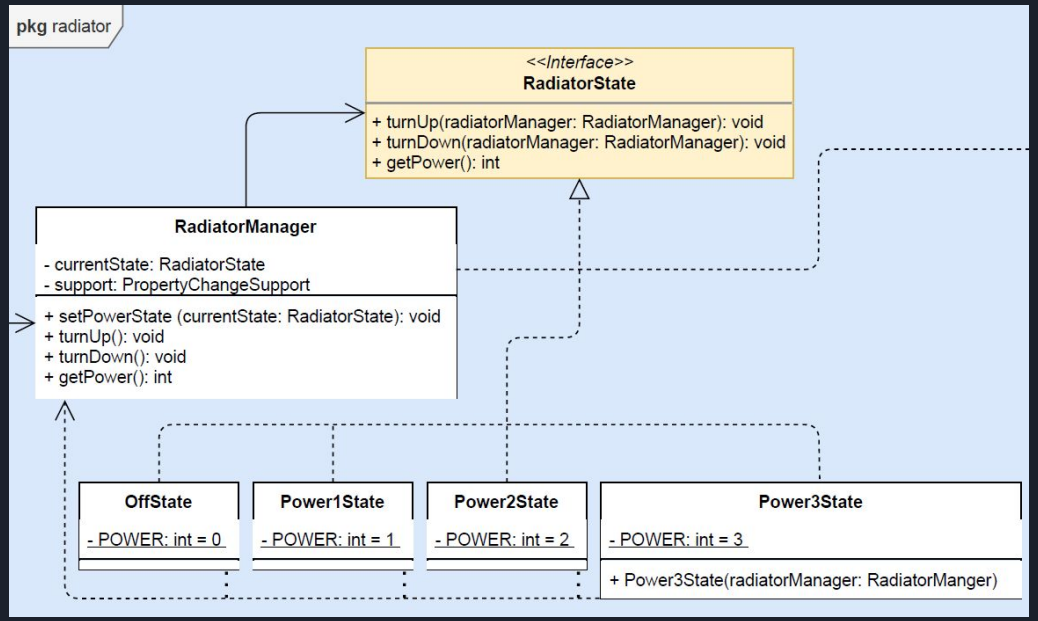
- State pattern is a behavioral software design pattern that allows an object to alter its behavior when its internal state changes.
- It is used to encapsulate varying behavior for the same object, based on its internal state.
- Is a cleaner way for an object to change its behavior at runtime without resorting to conditional statements and thus improve maintainability and scalability.

What are the different parts involved?

- Context: Defines an interface to client to interact. It maintains references to concrete state object which may be used to define current state of object.
- State: Defines interface for declaring what each concrete state should do.
- ConcreteState: Provides implementation for methods defined in State.



UML + Java example



JUnit

Black-box vs White-box testing

- Black-box → is a high level of testing that focuses on the behavior of the software.
  - Is used to test the software without the knowledge of the internal structure of program or application.
  - Is based on external expectations.
  - Programming knowledge is not needed.
  - The main objective of White Box testing is done to check the quality of the code.
  - Performed by the end user, developer, and tester.

- White-box → is a testing technique which checks the internal functioning of the system.
  - The internal structure is known to the tester and tested according to it.
  - Programming knowledge is required.
  - The main objective of White Box testing is done to check the quality of the code.
  - Usually done by tester and developers.
  - It allows removing the extra lines of code

Test example

- Class *Calculation*.
- Testing methods
  - assertEquals
  - assertTrue
  - assertThrows