State Pattern

Object Oriented System Design

Otago Polytechnic
Dunedin, New Zealand

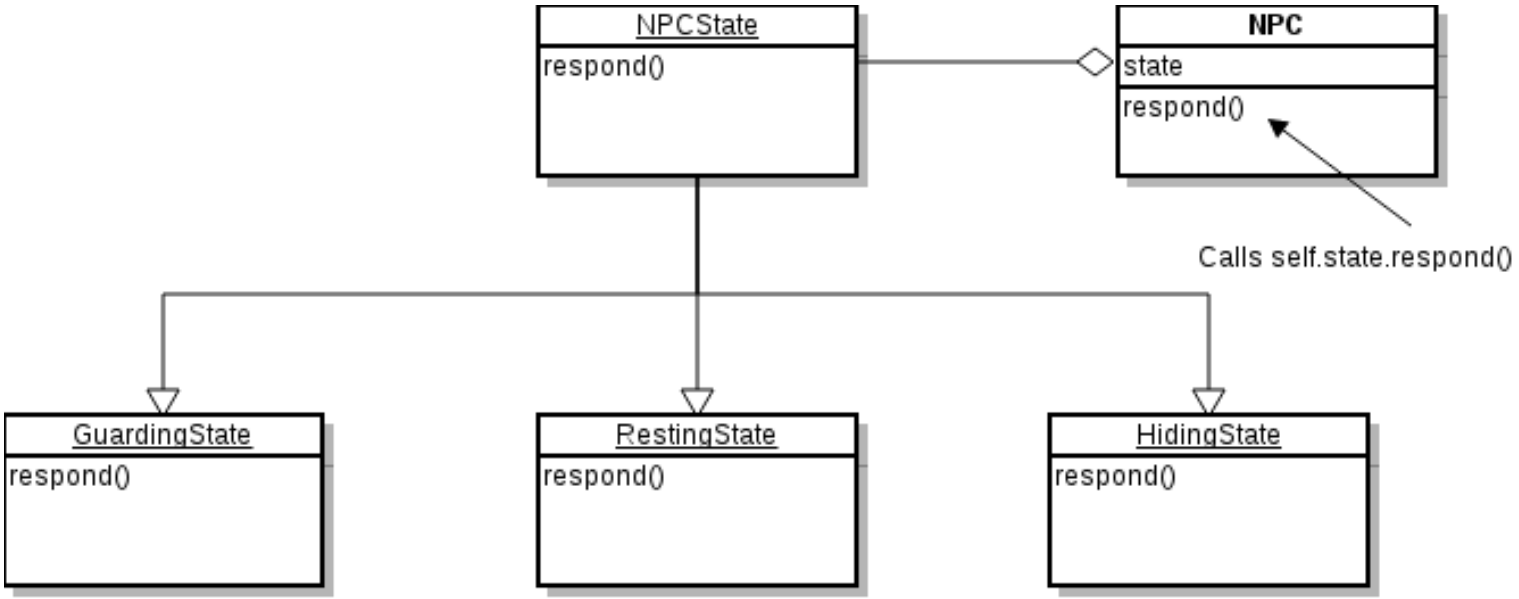
Delegation in design patterns

- In OO, *delegation* takes place when an object holds an instance of another object as a field and delegates the execution of a task to it.
- This provide a way to modify the behaviour of an object at runtime.
- We saw one example of this with the *Strategy* pattern.

Problem

- Suppose we are writing a game. In the game, we have non-player characters that should respond whenever a player gets close to them.
- The NPCs respond differently depending on what they are doing. They may be
  - guarding
  - resting
  - hiding
  - hunting
  - moving
- These things are *states*. We could say that an NPC is is a "guarding state".

Basic implementation



So far, this looks like a strategy

They are related, but

- A state object can contain a lot of state related data.
- A state can implement many state-based behaviours.
- A state can control how the containing object transitions to other states.

Exercises

1. Write a `Light` class that uses two state objects, `On` and `Off` to represent its states. The light should have a `switch()` method that delegates its implementation to the states.
2. Write a `VendingMachine` class. This class moves between a number of states, including waiting, money-received, vending, and making-change/refunding. Implement a set of appropriate state classes.