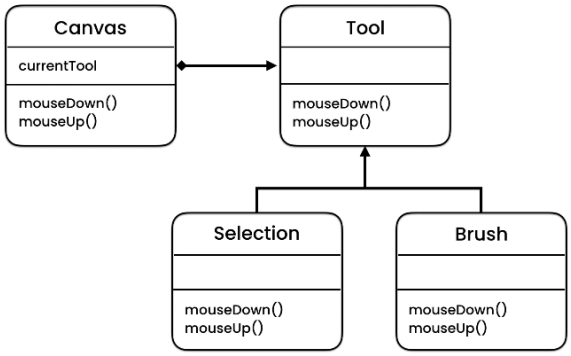
State Design Pattern

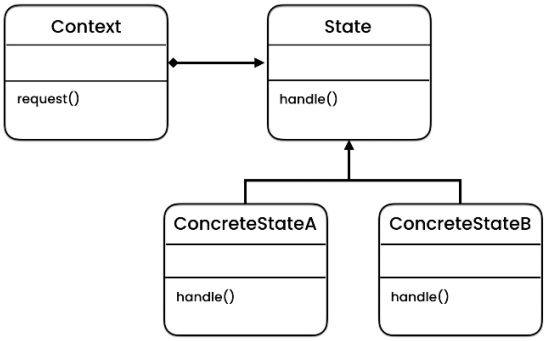
This pattern is used when there is a set of states and our class must behave differently according to these states.

Class Diagrams

Our Example



The pattern in Gang of Four's book



Explanation

Consider a drawing application. We have a pallet of tools and a canvas that behaves differently depending on the tool that we select. We click and hold the mouse and if we have chosen the Selection tool the program selects the area, if we have chosen the eraser the program erases the area, and so on. We implement that by using the Polymorphism principle. The tool is an abstract class and it has two methods that Selection and Brush class must implement. The interface and abstract classes are almost alike, but abstract classes are used when there is a need to share common code with child classes. So, in this case it is better to define the Tool class as an interface. as The Canvas class works with the Tool class, it does not care about any specific tools that we give it at run time. In this way, we satisfy the "open closed" principle and Any tool can be added as a child of the Tool class (in codes there is an Eraser class that is a child of Tool class in addition to Selection and Brush class).