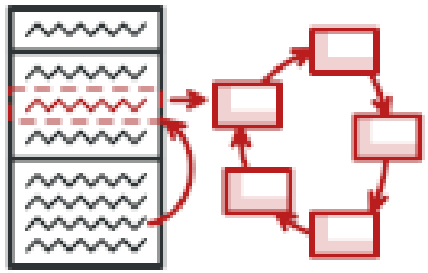
Design Mønstre

State Mønstre

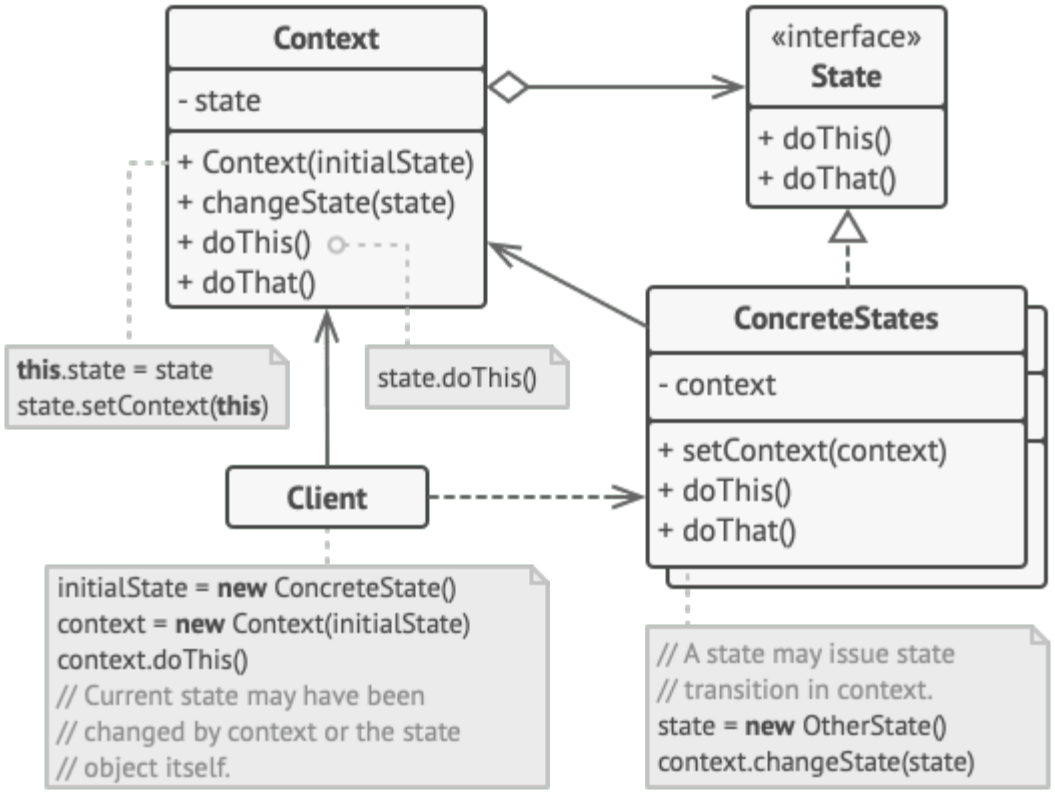


Hvad er State Mønstre?

Et adfærds mønstre som hjælper os med fleksibilitet

Et design Mønstre lader vores program ændre adfærd.

I stedet for at bruge betingelser for at styre adfærd, definerer mønsteret forskellige tilstandsklasser, hvor hver tilstand implementerer en specifik adfærd



SOLID Principles

- Single Responsibility Principle
  - en klasse kun bør have **én grund til at ændre sig**. Med andre ord bør en klasse have **kun ét ansvar** eller én opgave.

- Open and closed Principle
  - Klasser burde være åbne for nedarvning men ikke ændring.

- Liskov Substitution Principle
  - underklasser skal kunne erstatte deres main klasse uden at ændre adfærden af programmet. Med andre ord skal en underklasse forblive kompatibel med adfærden i superklassen.
- Interface Segregation
  - Vores program burde ikke kør unødvendig metoder. Vi kan opdele vores program i flere interfacer for at undgå dette.
- Dependency Inversion Principle
  - Superklasser skal ikke være direkte afhængig af subklasser.

Anvendelse af State mønsteret:

- Når objektet opfører sig forskelligt afhængigt af tilstanden: Tilstandsspecifik kode udtrækkes til separate klasser, hvilket gør det nemmere at tilføje eller ændre tilstande.
- Når klassen har mange betingelser: Betingelser udtrækkes til metoder i tilstandsklasser, hvilket rydder op i hovedklassen.
- Når der er duplikation af kode: Fælles kode udtrækkes til abstrakte basisklasser, hvilket reducerer duplikation og opretter hierarkier af tilstandsklasser.