STATE PATTERN
FINITE STATE MACHINE
FINITE STATE MACHINE

- A finite number of states
- The machine is in a single state at a time
- Each state has a set of transitions to other states
- Each transition is associated with an input
Allow an object to **ALTER ITS BEHAVIOUR** when its internal state changes. The object will appear to **CHANGE ITS CLASS**.
Useful for:

- Character controllers
- Menu systems
- Interactable objects
- Animations
Useful for:

Most frequently executed code paths with a **LONG SWITCH STATEMENT** or a lot of **IF/ELSE** sentences.
Common state methods:

**ENTER**

*CALL WHEN TRANSITIONING INTO THE STATE.*

*Example uses: save timestamp, set graphics, play audio*
Common state methods:

CALL WHEN TRANSITIONING OUT OF THE STATE

Example uses: clean up, set graphics, play audio
Common state methods:

INPUT ACTIONS

- Player input
- Update pattern
- Interaction with other object
Variations:

Concurrent State Machines

I.e. use of multiple state machines

Example:

- What a game character is doing
- What a game character is carrying
Variations:

**Hierarchical State Machines**

*States are organized in a hierarchy*
*Can pass handling of an input to parent*

**Pushdown Automata**

*A stack of states*
HOMEWORK: 08.03 - 29.03

Read State chapter from Game Programming Patterns
http://www.gameprogrammingpatterns.com/state.html

Complete Refactor Replay task in CGLearn