STRATEGY AND TYPE OBJECT
The Strategy Pattern involves creating a Context class that uses an interface, Strategy, to work with ConcreteStrategy classes. The Context class has a method ContextBehaviour() that delegates to the selected ConcreteStrategy object. This allows the Context to use different ConcreteStrategies without modifying its own code.
STATE PATTERN

Context

+ Request()

State

+ Handle()

ConcreteStateA

+ Handle()

ConcreteStateB

+ Handle()
**STRATEGY PATTERN**

**DEFINE A FAMILY OF ALGORITHMS,**

**ENCAPSULATE EACH ONE, AND**

**MAKE THEM INTERCHANGEABLE.**

**STRATEGY LETS THE ALGORITHM**

**VARY INDEPENDENTLY FROM**

**CLIENTS THAT USE IT.**

**STATE PATTERN**

**ALLOW AN OBJECT TO ALTER ITS**

**BEHAVIOUR WHEN ITS INTERNAL**

**STATE CHANGES. THE OBJECT**

**WILL APPEAR TO CHANGE ITS**

**CLASS.**
STRATEGY PATTERN

Strategies do not know of each other's existence.

STATE PATTERN

High class coupling between states.
**STRATEGY PATTERN**

Assignment of strategies to the context is out of scope of the strategy pattern.

**STATE PATTERN**

States often transition the context into another state based on input.
Applications may have only a single concrete strategy and use the pattern to support plugins and modifications.

Application should have multiple states for the context. Otherwise, it only increases complexity without adding any benefits.
TYPE OBJECT PATTERN
What's wrong here?
How can we fix this?
TYPE OBJECT

Allows the **flexible creation** of new "classes" by creating a **single class**, each instance of which represents a **different type of object**.
STRATEGY PATTERN

Applications may have only a single concrete strategy and use the pattern to support plugins and modifications.

STATE PATTERN

Application should have multiple states for the context. Otherwise, it only increases complexity without adding any benefits.
**Typed Object**

- Has a contextual role in the application
- Encapsulates a type object
- Behaviour depends on data from type object

**Type Object**

- Provides data to the typed object
- May provide method implementations to the typed object
Generally assigned once in the lifecycle of the context.

Generally assigned once in the lifecycle of the context.
STRATEGY PATTERN

Works great with flyweight

TYPE OBJECT

Works great with flyweight
STRATEGY PATTERN

A dynamic way to provide the context class with algorithms.

TYPE OBJECT

A dynamic way to provide the context class with type and data.
STRATEGY PATTERN

A dynamic way to provide the context class with algorithms.

TYPE OBJECT

A dynamic way to provide the context class with type and data.
Requires a solution for creating and editing type data outside application and deserializing it into type objects.
UNITY SCRIPTABLE OBJECTS

- Can’t be directly attached to gameobjects
- Great for use as Type objects in Unity
- Assets in the project or deserialized during runtime
- Support reading and writing from json
- Can reference other scriptable objects
HOMEWORK: 15.03 – 29.03

1. Read Type Object chapter from Game Programming Patterns
   http://www.gameprogrammingpatterns.com/type-object.html

2. Read Strategy and Type Object chapters from CGLearn
   https://cglearn.eu/student/materials

3. Start thinking about your essay