DESIGN PATTERNS: **NULL OBJECT**

Behavioral Pattern

1. Design Pattern Description

Designed to act as a default value of an object

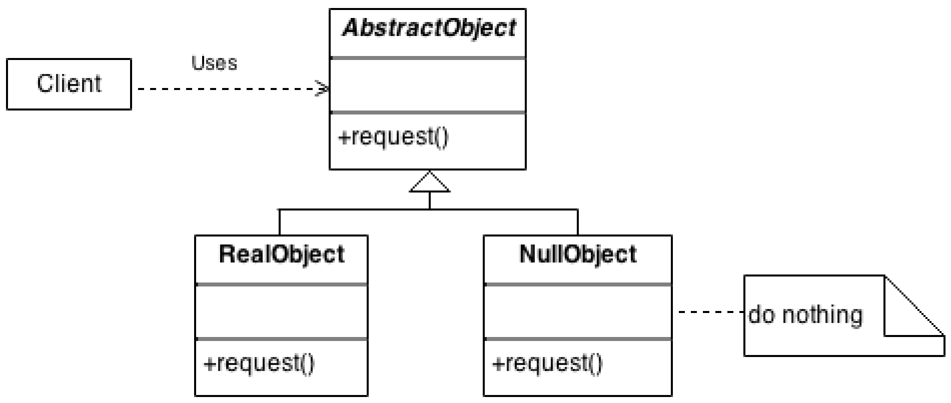
- **Intent:** Provide a suitable default do nothing behavior.
- **Problem:** to treat transparently the absence of an object (the presence of a null reference)→ do nothing or use some default value.

Sometimes a class that requires a collaborator does not need the collaborator to do anything. However, the class wishes to treat a collaborator that does nothing the same way it treats one that actually provides behavior.

The key to the Null Object pattern is an abstract class that defines the interface for all objects of this type. The Null Object is implemented as a subclass of this abstract class.

A Null Object knows what needs to be done without interacting with any other object.

- It is immutable.
- Can be a particular ConcreteStrategy → do nothing / ignore all
- Can be a particular State pattern
- Can allow visitors to safely visit a hierarchy and handle the null situation. (visitor = client??)
- The null behavior is **not designed to be mixed into an object** that needs some do nothing behavior. It is designed for a class which delegates to a collaborator all of the behavior that may or may not be do nothing behavior.



- **Client** - requires a collaborator.
- **AbstractObject** - declares the interface for Client's collaborator implements default behavior for the interface common to all classes, as appropriate
- **RealObject** - defines a concrete subclass of AbstractObject whose instances provide **useful behavior** that Client expects
- **NullObject** - provides an interface identical to AbstractObject's so that a null object can be substituted for a real object implements its interface to do nothing.
    - What exactly it means to do nothing depends on what sort of behavior Client is expecting
    - When there is more than one way to do nothing, more than one NullObject class may be required.

2. Design Pattern Example

```
"""
Encapsulate the absence of an object by providing a substitutable
alternative that offers suitable default do nothing behavior.
"""

import abc


class AbstractObject(metaclass=abc.ABCMeta):
  @abc.abstractmethod
  def request(self):
    pass


class RealObject(AbstractObject):
  def request(self):

"""
Useful behavior.
"""
    pass


class NullObject(AbstractObject):
  def request(self):
    pass
```

3. Existing Example in FEM-MAT-OO

Introduced recently in a Mesh_Unfitted concrete strategy: `MeshPlotter_Interior_3D`:

Mesh_Unfitted must define a Plotter strategy → plotters must implement method `plot`: `MeshPlotter_Interior_3D` results a Null Object.

Note: class renamed to **MeshPlotter_Null**.

4. Design Proposal in FEM-MAT-OO

- Useful for Test Driven Development → in RED stage: compile the code without passing the tests.
- For strategies that must be implemented but are not going to be used.