Servant-Pattern

****

A ​​Mini Project for Design-​Patterns(14CS413 ) course by:

1PI14CS081 ​ ​-​ ​Ramadas S Mahale,

1PI14CS081 ​ ​- ​ ​Sunil Pai G,

7th semester,CSE.

GitHub: ​ [​​Sunil-Pai-G/Servant-Pattern](https://github.com/Sunil-Pai-G/Servant-Pattern)

**OVERVIEW**

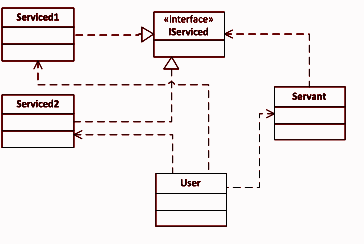
Servant pattern is a design pattern which is used to provide common functionality to a set of classes. This pattern is used when it does not make sense to provide this functionality in a common parent class and when disparate classes need to show the same behavior.Hence, this pattern prevents duplication of code across multiple classes.The class in which the common behavior is implemented is called the servant class. The classes needing the service from the service class must implement an interface. This interface specifies the capabilities needed by classes in order to be serviced by the servant.

**IMPLEMENTATION DETAILS**

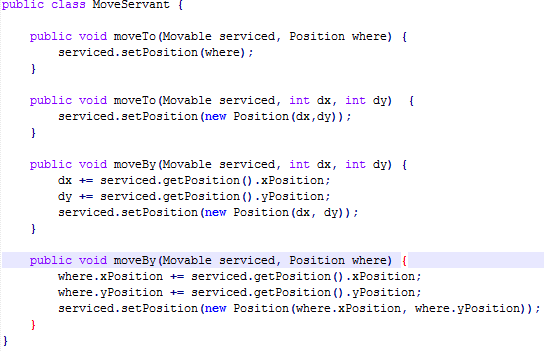
There are two ways to implement the pattern:-

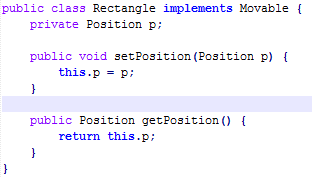
1. The client knows about the servant and passes the instances of the class to be serviced to the servant. The servant then calls appropriate methods on the instance passed.

UML diagram:



Code:

****

****

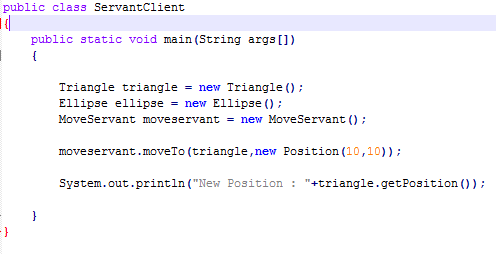
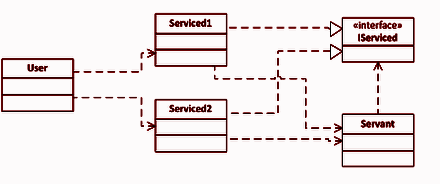
****

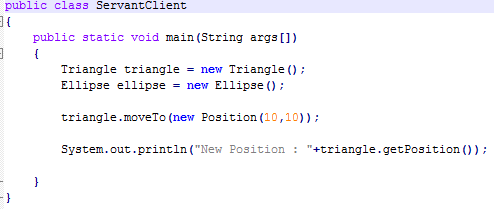
Fig:Servant,serviced class and client

1. The client knows nothing about the servant.When the client needs a certain service he calls a function in the serviced class which in turn calls the appropriate function in the servant.

UML diagram:



Code:

****

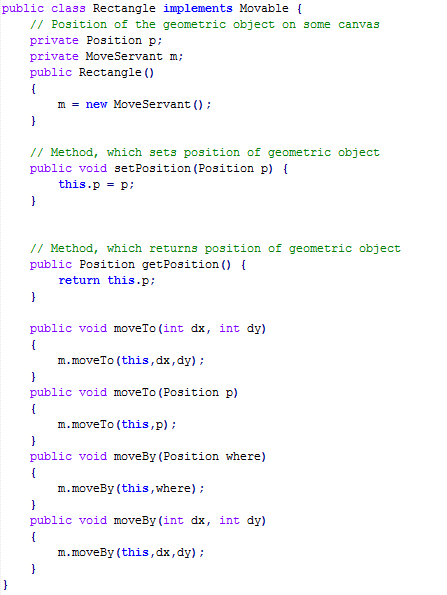
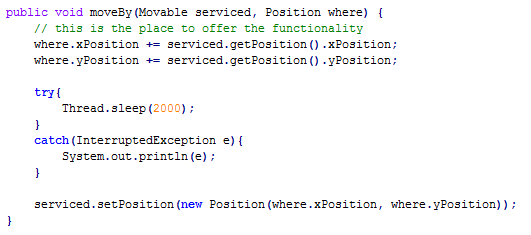
****

Fig:The client and serviced classes

**THREAD SAFE SERVANT PATTERN**

The above implementations of the command pattern are not thread safe. Consider what happens when two threads are executing the moveBy function simultaneously on the same object as shown below.Thread 1 wants to move the position of the triangle by (10,10) and thread 2 by (20,20).In effect, the triangle should have an eventual displacement of (30,30). However, if one of the threads reads the position of the triangle before it is updated by the other thread, then there is a race condition. We try to simulate this by making the thread sleep before it updates the position of the triangle.(Note that this doesn’t ensure that the race condition will occur every time the code executes).

****

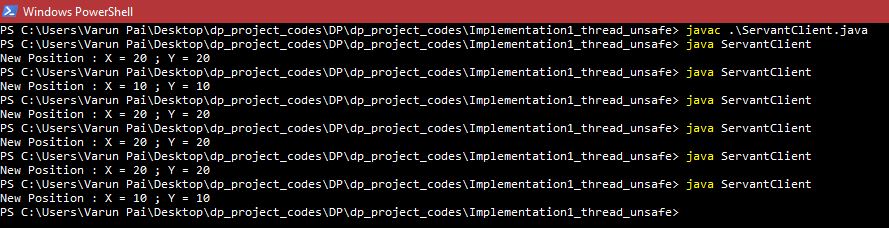
****

Fig:Thread unsafe servant and its output

The above problem can be resolved by allowing only one thread to access the moveBy function at any instance of time. This can be achieved in JAVA using the synchronized keyword in the method signature and in C++ using a std::mutex and a std::scoped\_lock. The implementation of this is as follows.

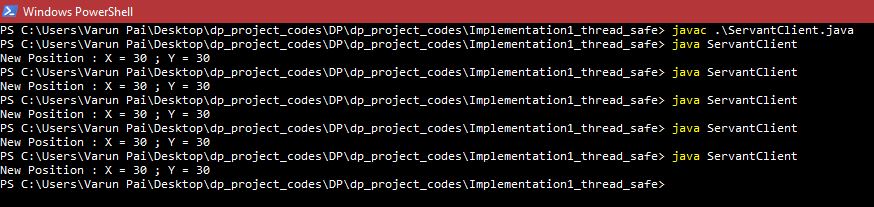
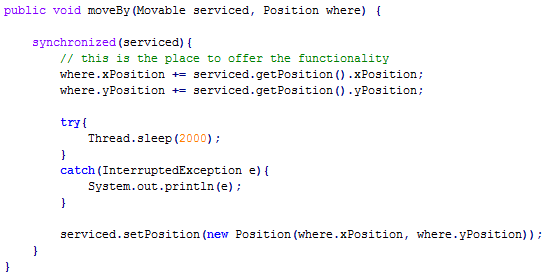
****

Fig:Thread safe implementation of servant pattern and its output

**SERVANT PATTERN A SINGLETON?**

As the servant class has no attributes of its own and only performs operations on serviced classes, the servant pattern can be made a singleton class. This approach is effective in saving memory. Making the servant a singleton doesn’t affect concurrency in any way as the singleton servant can still be operated by multiple threads.

**COMMAND vs SERVANT PATTERN**

Though the command and the servant patterns seem similar, there are subtle differences between them.In servant pattern, there is only one class that implements the required functionality(the servant class).In the command pattern however,

it is necessary for the command class to know the receiver of the action. Hence, for every receiver (the ellipse,the triangle and rectangle classes) there is a

need for a separate command class. In command class there is control on when to execute the commands.In servant there is no control on when commands are executed.

**REFERENCES:**

* https://en.wikipedia.org/wiki/Servant\_(design\_pattern)
* http://en.cppreference.com
* https://theboostcpplibraries.com/
* https://softwareengineering.stackexchange.com/questions/299520/do-we-need-servant-pattern-what-about-implementing-in-parent-class
* https://stackoverflow.com/questions/31986332/visitor-vs-servant-vs-command-patterns
* https://stackoverflow.com/questions/27939046/design-patterns-for-php-visitor-pattern-vs-servant-pattern
* https://www.codeproject.com/Articles/598695/Cplusplus-threads-locks-and-condition-variables