**Credit Card Problem – Part 2**

**Problem Statement:**

Parse input files in various formats like json, xml, csv etc. Each line in the file gives information on credit card number, expiration date and name. Process this information to identify the type of credit card. Output card number, type of card (if a valid card number) and an error into a file of format same as input. The file format should be extended to include newer formats as well.

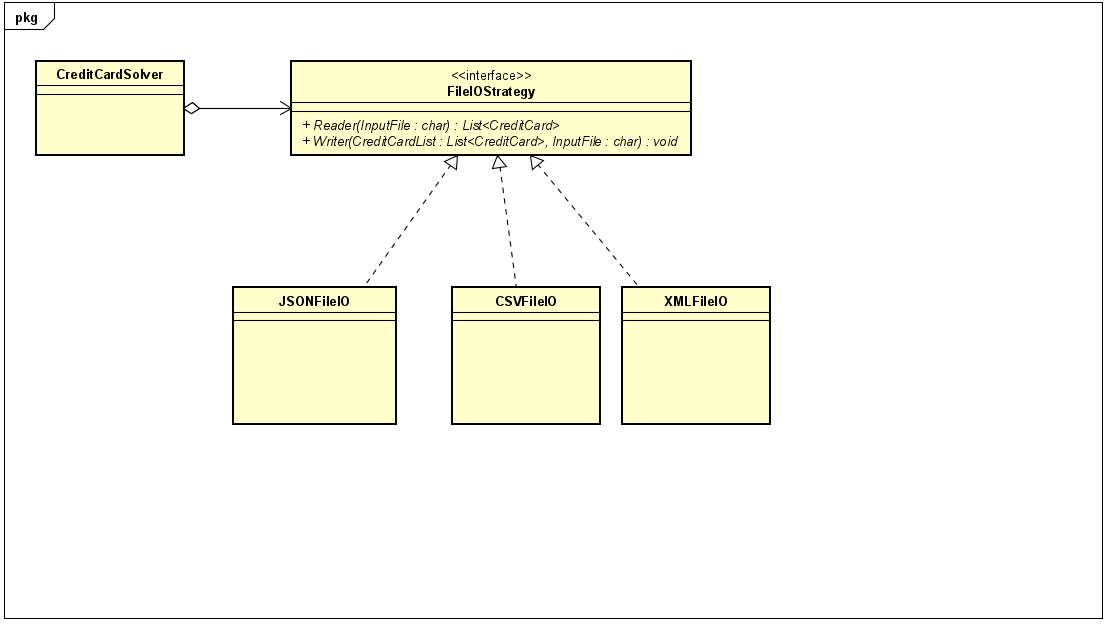
**Design Pattern:**

**Strategy design pattern:**

Here, we have different file formats for input and output. The implementation of file reader and writer programs depends on the type of the input file. There exists an interface FileIO with method signatures that specify the read and writer operations. Each class that implements this interface provides its own strategy to perform read and write.

Further, when the need for a new type of input and output file arises, a new class that implements this interface can be added and used. This way, extension is also possible.

Class Diagram



**Consequences of Strategy pattern:**

1. Families of related algorithms
2. An alternative to subclassing.
3. Strategies eliminate conditional statements
4. A choice of implementations

**Drawbacks:**

1. Clients must be aware of different strategies: Exposes clients to implementation issues
2. Communication overhead between Strategy and Context
3. Increased number of objects