Private Class Data

Intent

- Control write access to class attributes
- Separate data from methods that use it
- Encapsulate class data initialization
- Providing new type of `final` - *final after constructor*

Problem

A class may expose its attributes (class variables) to manipulation when manipulation is no longer desirable, e.g. after construction. Using the private class data design pattern prevents that undesirable manipulation.

A class may have one-time mutable attributes that cannot be declared final. Using this design pattern allows one-time setting of those class attributes.

The motivation for this design pattern comes from the design goal of protecting class state by minimizing the visibility of its attributes (data).

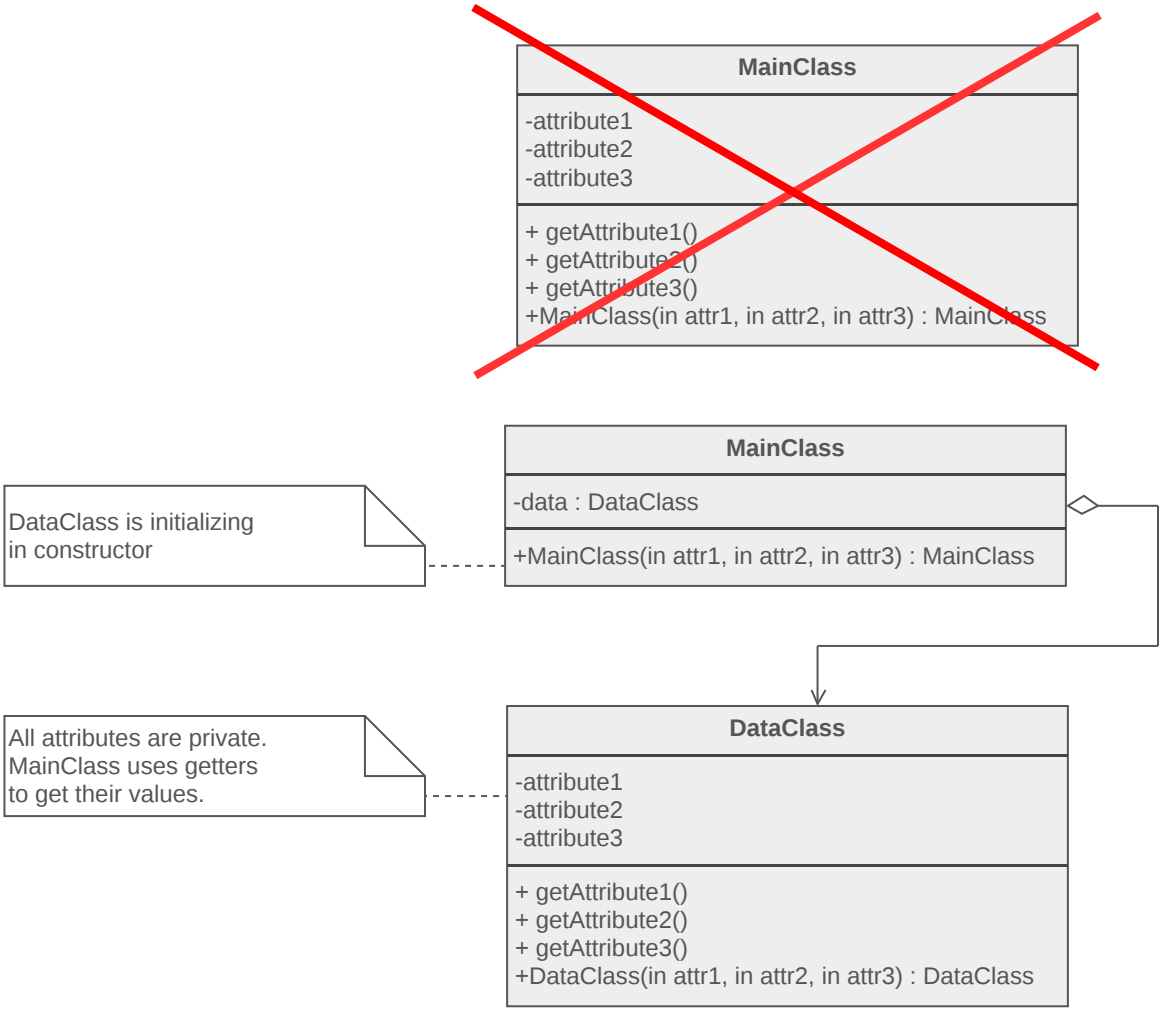
Discussion

The private class data design pattern seeks to reduce exposure of attributes by limiting their visibility.

It reduces the number of class attributes by encapsulating them in single Data object. It allows the class designer to remove write privilege of attributes that are intended to be set only during construction, even from methods of the target class.

Structure

The private class data design pattern solves the problems above by extracting a data class for the target class and giving the target class instance an instance of the extracted data class.



Check list

1. Create data class. Move to data class all attributes that need hiding.
2. Create in main class instance of data class.
3. Main class must initialize data class through the data class's constructor.
4. Expose each attribute (variable or property) of data class through a getter.
5. Expose each attribute that will change in further through a setter.