# Patterns chosen

## Factory Method

We can use Factory Method with SlideItem class, this can improve the expansion of the code, because right now this class does not support easy expansion without violating SOLID principles. In order to add a new Item you have to rewrite a lot of code. If we introduce the factory, it will be much easier to add new Items, which will have a positive impact on code expansion and will improve its quality as well.

## Observer Pattern

We can use Observer Pattern to remove tight coupling between

SlideViewerComponent and Presentation. Moreover, if some other component needs to react on slide changes, we will need to directly update the Presentation class, which will violate Open-Closed-Principle (OCP). By introducing the Observer Pattern, we will remove tight coupling and make Presentation class able to notify all its subscribers about slide changes, so you will not violate OCP.

## Command Pattern

## We can use Command Pattern to improve the way of adding new functionality to the Slide class without modifying the existing code, because right now with existing implementation with switch method it is impossible. As well as that, it will help create the undo functioality which might be helpful in the future.

# Code Problems

1. When you press page down when you are on the last page – the null pointer exception is thrown.
2. Redundant usage of DEMO\_NAME and DEFAULT\_EXTENSION in Accessor class.
3. If user inputs not a number in the MenuController when tries to go to another page – it will throw an exception.
4. In Presentation class in the clear method instead of initializing the new array each time, it is better to use .clear
5. In XMLAccessor, there is a possible Null Pointer Exception, if the tag is missing in the getTitle() function.
6. There is redundant variable DEFAULT\_API\_TO\_USE in XMLAccessor