**Refactoring**

Refactoring is a disciplined technique for restructuring an existing body of code, altering its internal structure without changing its externally observable behavior.

Refactoring is usually done to :

* + Improve quality
    - improve design quality
    - improve maintainability
    - improve extensibility
    - requires proper testing, so it improves testability
    - helps to find bugs
    - Without refactoring, agile methods are likely to create code whose further development will be exponentially costly.
  + Improve productivity
    - improve code readability & comprehensibility
    - simplify code structure
  + Improve sustainability of development
    - By improving the code’s structural quality, reducing confusion and making the code more understandable, it reduces the effort involved in further development.
    - This is very important in agile software development methods, whose focus on productivity and changes are likely to create lower quality code.
    - Without refactoring, agile methods are likely to create code whose further development will be exponentially costly.

**Our refactoring**

* Observer design pattern For Player, MapView. (ex. moves three Functions in the player class )
* Rename methods and fields: we did some refactoring for increasing the readability and maintainability by changing the method and field’s names.
* Remove some code redundancies in the RiskUI.
* Try to use the shorter methods because those are easier to read, understand, and easier to troubleshoot.
* Make sure the balance between classes in term of the complexity because classes should pull their weight.