Clean Code Chapter 4 & 7 요약

20200120 차선호[carprefer]

Chapter 4 Comments

주석은 적절히 작성하지 않으면 독이 되는 경우가 많다. 꼭 필요한 곳에만 주석을 효율적으로 작성해야 하고, 코드의 네이밍으로 주석을 대체하는 습관을 들여야 한다.

좋은 주석은 다음과 같다.

1. 저작권, 작성자를 밝히는 주석
2. 코드로 표현하기 어려운 정보를 전달하는 주석
   // format matched kk:mm:ss EEE, MMM dd, yyyy
3. 코드의 의도를 설명하는 주석
4. 의미가 모호한 함수의 반환값이나 인자를 구체화하는 주석
   Standard library의 함수를 가져다 쓸 경우
5. 경고하는 주석
6. TODO 주석
   앞으로 어떤 내용을 구현해야 할지 알려주는 주석
7. 중요한 부분을 강조하는 주석

나쁜 주석들의 예는 다음과 같다.

1. 모호한 주석
2. 중복되는 주석
   특히 함수나 인자의 네이밍으로 대체할 수 있는 주석은 피하도록 하자.
3. 필수 주석
   모든 함수나 변수에 주석을 필수적으로 적는 것은 불필요하다.
4. 저널 주석, Bylines, 바꾸기 전 코드 주석
   저널 주석이란 지금까지의 변경사항을 주석에 적는 것이고, Byline 주석은 해당 코드를 누가 작성했는지 표시하는 주석이다. 코드를 변경하기 이전의 코드를 주석으로 남겨두는 것 또한 불필요하다. 대신 Git 같은 툴을 사용하자.

5. 불필요한 주석
   // default constructor
6. 위치를 나누는 주석
   해당 주석으로 얻는 이점이 크지 않는 이상 사용하지 않도록 하자.
7. 괄호 닫기 주석
   코드가 길 경우, 괄호를 닫으며 어떤 block에 해당하는 괄호인지 표시하는 주석을 달 때가 있는데, 코드를 짧게 작성하여 해당 주석을 피하도록 하자.
8. 해당 부분과 상관없는 주석
   작성한 주석은 해당부분만을 설명해야 한다.
9. TMI 주석

Chapter 7 Error Handling

Clean code란 읽기 쉬워야 하나, 에러를 발생시켜도 안된다. 이 부분에서 error handling이 중요하며, 이를 작성할 때 생각해봐야 할 사항들은 다음과 같다.

1. 에러코드를 반환하는 대신 exception을 사용하자.
2. Try-catch-finally 구문을 먼저 사용하자.
3. Unchecked exception을 사용하자.
   Checked exception의 경우, 예외 발생 시 encapsulation이 깨질 수 있다는 단점이 있기 때문이다.
4. Exception에 충분한 정보를 같이 넘겨주자.
5. Exception wrapper class를 사용하여 코드를 간략화하자.
   새로운 wrapper class를 정의하면 나중에 해당 api를 사용할 때 편리하다.
6. Special case pattern을 사용하여 client 코드에서 exception을 사용하지 않아도 되도록 하자.

   이 경우, client 코드에서는 normal한 상황에 대해서만 코딩을 해도 된다.

7. Null을 반환하지 말도록 하자.

   Exception을 throw하거나 special case object를 반환하도록 하자.

8. 인자로 null을 넘겨주지 말자.
   이를 assertion으로 경고하자.