Design Pattern - Null Object

O padrão Null Object é um padrão de design que simplifica o uso de dependências que podem ser indefinidas. Isso é desenvolvido usando instâncias de uma classe concreta que implementa uma interface conhecida, em vez de referências nulas, sendo assim um objeto criado para simular outro objeto com os mesmos contratos, mas sem funcionalidade alguma. Sem estado e sem comportamento real. Os dois tipos são compatíveis entre si já que atendem o mesmo contrato, implementam as mesmas interfaces e possivelmente estendem o mesmo tipo. Com isso, a intenção desse design pattern é ser utilizado como uma assinatura do objeto original.

O Null Object, de modo geral, fornece uma maneira legível de tratar referências nulas em códigos, a partir disso, alguns benefícios do Null Object são: define hierarquias de classe compostas por objetos reais e objetos nulos. Objetos nulos podem ser usados no lugar de objetos reais quando se espera que o objeto não faça nada. Sempre que o código do cliente espera um objeto real, ele também pode levar um objeto nulo. Também torna o código do cliente simples. Os clientes podem tratar colaboradores reais e colaboradores nulos uniformemente. Os clientes normalmente não sabem se estão lidando com um colaborador real ou nulo. Isso simplifica o código do cliente, pois evita ter que escrever código de teste que lida especialmente com o colaborador nulo.

Contudo, apesar do Null Object não haver efeito colateral, é importante que o time de desenvolvimento defina de antemão como esse objeto nulo deve se comportar, dado que a manipulação de objetos nulos geralmente aumenta o esforço no escopo do software. Também é importante levar em consideração que alguns erros críticos podem ser maquiados utilizando o Null Object, o que pode ser preocupante caso esse erro impacte diretamente a funcionalidade do software. Além disso, a utilização desse padrão acarreta em um aumento considerável tanto na criação de classes quanto no trabalho para implementar classes grandes.