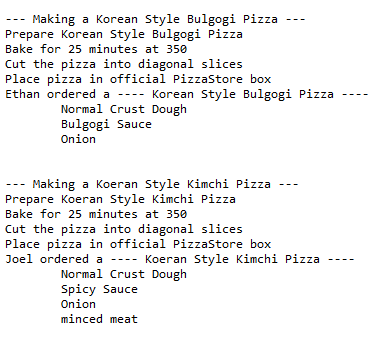
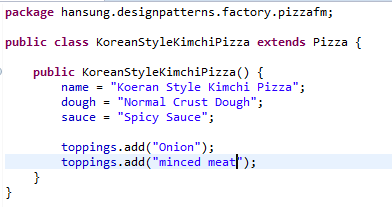
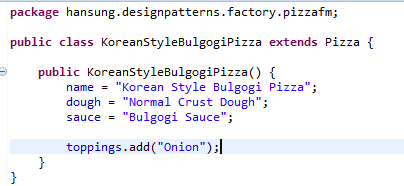
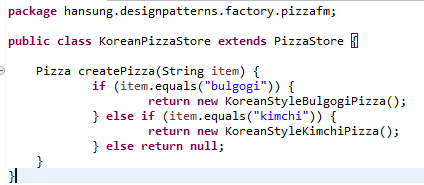
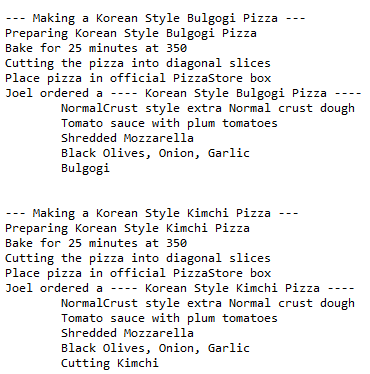
1. 피자 프렌차이즈 사업이 날로 확대되어서 드디어 한국에도 분점을 내게 되었습니다. 한국 분점의 피자는 한국스타일로 여러분이 원하는 대로 정하면 됩니다. 가령 김치피자나 불고기 피자가 판매될 수도 있겠죠. pizzafm의 코드를 바탕으로 한국 분점을 새로이 내기 위해 어떤 클래스가 새로이 추가되었고, 어떤 코드가 수정되었나요?



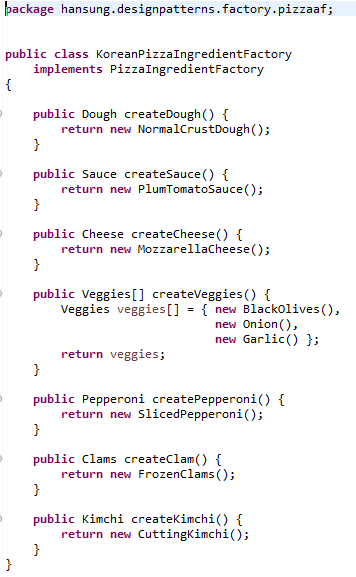
불고기 피자와 김치피자를 추가하였습니다.



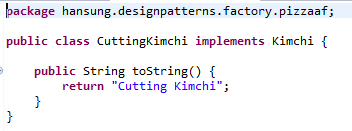
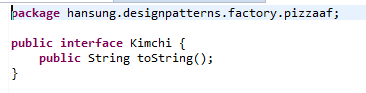
1. 이번에는 pizzaf의 코드를 바탕으로 1 번 문제를 적용해보세요.



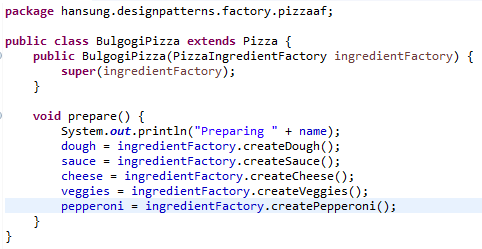
마찬가지로 불고기 피자와 김치 피자 두가지를 추가하였습니다.



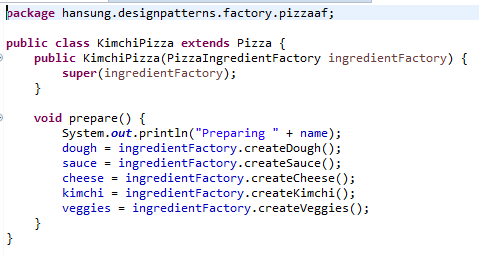
김치를 추가해주기 위해서 IngredientFactory 클래스에 Kimchi를 추가하였습니다.



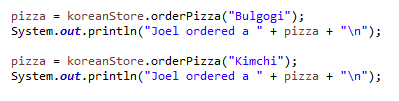
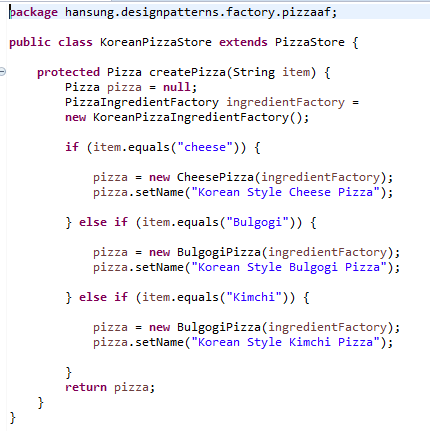
IngredientFactory에 kimchi를 추가해주기 위한 클래스 생성.



불고기 피자 클래스 (한국식 피자이기 때문에 페퍼로니를 불고기로 수정하였습니다.)



김치 피자 클래스



1. pizzafm 코드를 바탕으로 했을 때와 pizzaf 코드를 바탕으로 수정했을 때와 어떤 차이가 있는지, 장단점을 비교 토의하세요.

Pizzaf 코드로 작성한 경우

장점 : 새로운 피자 코드를 추가할 때 모든 코드를 추가하는 것이 아니라 기본 베이스는 IngredientFactory로부터 가져오고 새로운 피자의 특징만 코드를 추가하여 코드 재사용에 뛰어난 장점을 가지고 있습니다.

단점 : 새로운 피자 코드를 수정하는 과정에서 복잡한 과정을 거쳐야 하며 새로운 토핑, 새로운 특징을 추가하려면 모든 IngredientFactory를 수정하여 주어야 하며 너무나 많은 클래스가 생성되어 복잡해지는 문제가 있다.

Pizzafm 코드로 작성한 경우

장점 : 각 피자의 특징을 한 클래스에 정의하여 직관적으로 이해할 수 있고 클래스의 숫자가 메뉴의 개수와 동일하기 때문에 복잡성이 적다.

단점 : 새로운 피자를 추가하려면 도우, 소스, 등 모든 재료를 새로 추가하여 코드를 작성하여야 하는 단점이 있다.