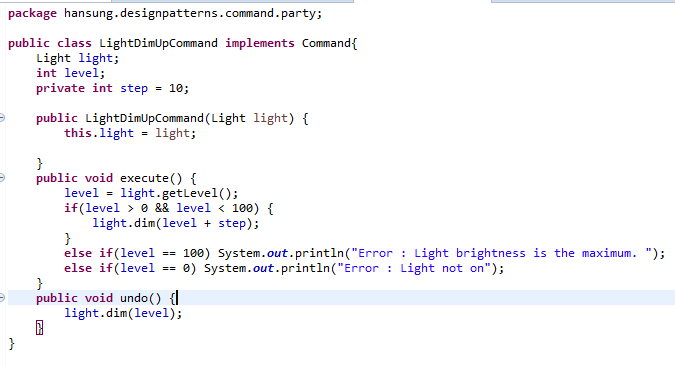
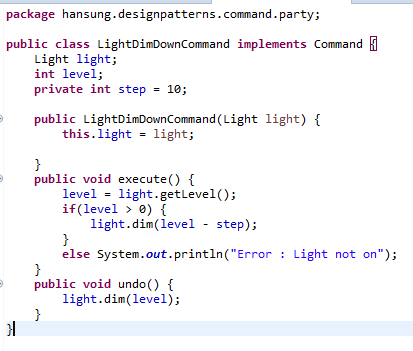
1. 이 예제에서는 Light 클래스의 on(), off() 메소드를 호출하는 두 개의 커맨드 클래스 (LightOnCommand, LightOffCommand)가 포함되어 있습니다. 하지만, Light 클래스의 dim() 메소드를 호출하는 커맨트 클래스는 없습니다. dim() 메소드는 조명의 밝기 수준을 설정할 수 있는 메소드인데, 이를 사용하여 조명의 밝기를 한단계(예, 10%) 낮출수 있는 LightDimDownCommand와 밝기를 한단계(예, 10%) 높일 수 있는LightDimUpCommand클래스를 정의해 보세요. 그리고, 아래와 같이 동작하는 지를 확인해 보세요
   * 두 커맨드는 Light가 켜진 상태에서만 동작해야 합니다. 만약 Light이 켜지지 않은 상태라면, 에러 메시지를 출력해야 합니다.
   * LightDimDownCommand의 수행에 의해서 밝기를 한단계 낮추었을 때, 밝기수준이 0가 된다면, 조명이 꺼진것입니다.
   * LightDimUpCommand의 수행에 의해서 높일 수 있는 밝기 수준은 최대 100%를 초과할 수 없습니다.
   * 두 커맨드 클래스의 undo메소드를 구현하여 이전 밝기 상태로 되돌아 갈 수 있어야 합니다.



LightDimUpCommand 코드



LightDimDownCommand 코드

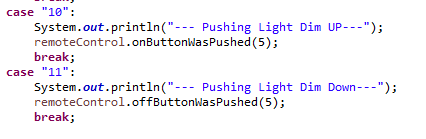
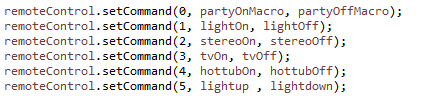
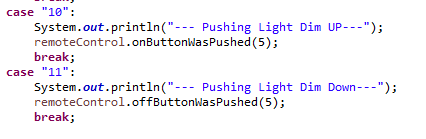
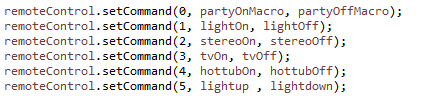
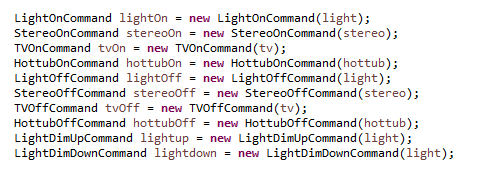


Light on 되지 않았을 때 에러 메시지



최대치를 초과하였을 때 에러 메시지

1. 1번의 실습에서 추가된 커맨드 클래스로 인해 기존 코드의 어떤 부분을 수정하였는지를 분석해 보고, 커맨드 패턴의 장점 및 단점을 논의해 보세요.



기존 커맨드들을 수정하지 않고 새로운 클래스와 remote loader만 수정하였다.

장점: 기존의 코드를 수정하지 않다 보니 새로운 커맨드를 추가하는 것이 간편하다.

단점: 새로운 커맨드가 추가될 때마다 새로운 클래스를 만들어야 하다 보니 코드 관리에 있어서 복잡해지는 단점이 있습니다.

1. 현재 RemoteControl 클래스는 한번의 작업 취소(Undo)만 가능하다. 이를 수정하여 여러번의 작업 취소가 가능하도록 RemoteControl 클래스를 수정하고 이 기능을 테스트한 결과를 제출하세요.

[힌트] 실행한 커맨드를 스택에 집어넣고, 사용자가 UNDO 버튼을 누를때마다 스택 맨 위에 있는 항목을 꺼내서 undo() 메소드를 호출하면 됩니다. 