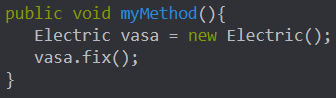
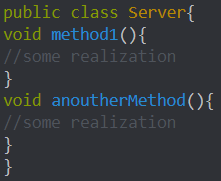
Proxy (англ. Заместитель, уполномоченный)

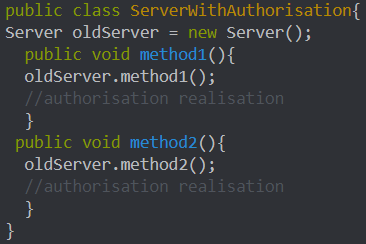
Есть у меня класс Electric, он умеет ремонтировать проводку.

И у меня в доме она как раз сгорела (неприятно), и что же мне теперь от него наследоваться, чтобы её починить? Можно. Но теперь я являюсь наследником Electric, следовательно меня могут заставить что-то чинить (подставить в какой-то метод), не круто. А ещё если кто-то злой нарушит Open-Closed и поменяет код в Electric, у меня всё слетит.

 Поэтому умные дядьки придумали следующие – если мне нужно, чтобы Electric что-то починил, я создаю его и заставляю это сделать.

Великолепно. Леонид показывал пример класса, целиком являющегося proxy :

 Есть класс Server со своими методами:

И тут мы раз и захотели, чтобы у нас один из серверов имел авторизацию, как нам это сделать?

Вот и сказочке конец… Ещё, это называется агрегация и её более строгий вариант – композиция (когда делегируемый класс не существует вне заказчика (класс brain, объявленный внутри класса human)).