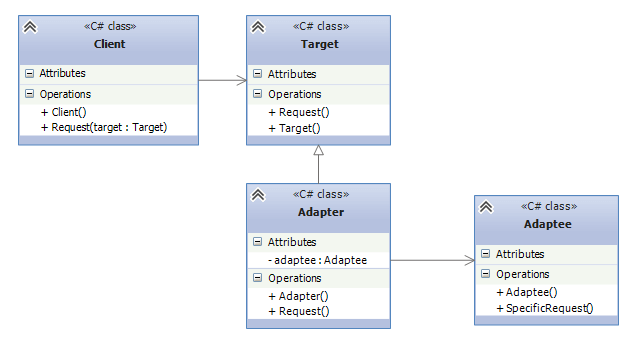
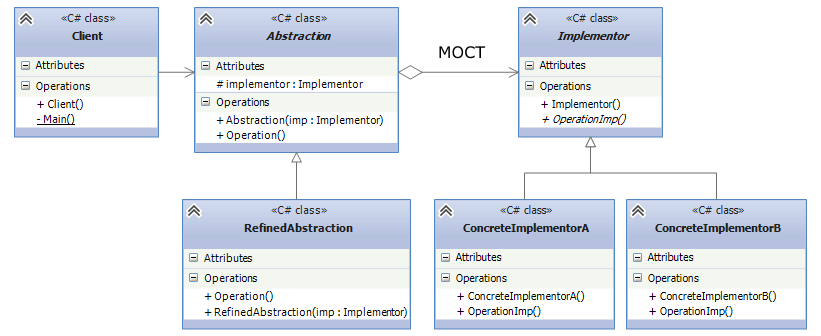
**Адаптер**

* Когда необходимо использовать имеющийся класс, но его интерфейс не соответствует потребностям
* Когда надо использовать уже существующий класс совместно с другими классами, интерфейсы которых не совместимы



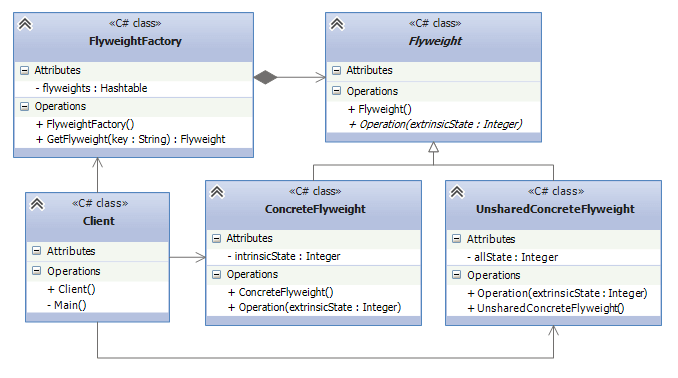
## Мост

* Когда надо избежать постоянной привязки абстракции к реализации
* Когда наряду с реализацией надо изменять и абстракцию независимо друг от друга. То есть изменения в абстракции не должно привести к изменениям в реализации



## Приспособленец

* Когда приложение использует большое количество однообразных объектов, из-за чего происходит выделение большого количества памяти
* Когда часть состояния объекта, которое является изменяемым, можно вынести во вне. Вынесение внешнего состояния позволяет заменить множество объектов небольшой группой общих разделяемых объектов.



## Заместитель

* Когда надо осуществлять взаимодействие по сети, а объект-проси должен имитировать поведения объекта в другом адресном пространстве. Использование прокси позволяет снизить накладные издержки при передачи данных через сеть. Подобная ситуация еще называется **удалённый заместитель (remote proxies)**
* Когда нужно управлять доступом к ресурсу, создание которого требует больших затрат. Реальный объект создается только тогда, когда он действительно может понадобится, а до этого все запросы к нему обрабатывает прокси-объект. Подобная ситуация еще называется **виртуальный заместитель (virtual proxies)**
* Когда необходимо разграничить доступ к вызываемому объекту в зависимости от прав вызывающего объекта. Подобная ситуация еще называется **защищающий заместитель (protection proxies)**
* Когда нужно вести подсчет ссылок на объект или обеспечить потокобезопасную работу с реальным объектом. Подобная ситуация называется **"умные ссылки" (smart reference)**

