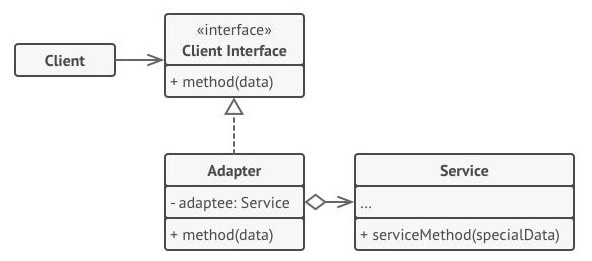
**Кукаркин И.А.**

**ПИ-15-1**

**Адаптер**

*Цель применения паттерна* - позволять объектам с несовместимыми интерфейсами работать вместе.

*UML-диаграмма:*



Client — класс, который содержит существующую бизнес-логику;

Client Interface — описывает протокол, через который клиент может работать с другими классами;

Service — это какой-то полезный класс, обычно сторонний. Клиент не может использовать этот класс напрямую, так как сервис имеет непонятный ему интерфейс;

Adaptor — это класс, который может одновременно работать и с клиентом, и с сервисом. Он реализует клиентский интерфейс и содержит ссылку на объект сервиса. Адаптер получает вызовы от клиента через методы клиентского интерфейса, а затем переводит их в вызовы методов обёрнутого объекта в правильном формате.

*Используется, когда:*

* Система поддерживает требуемые данные и поведение, но имеет неподходящий интерфейс. Адаптер позволяет создать объект-прокладку, который будет превращать вызовы приложения в формат, понятный стороннему классу

*Особенности паттерна:*

Достоинства:

* Шаблон Адаптер позволяет включать уже существующие объекты в новые объектные структуры, независимо от различий в их интерфейсах
* Отделяет и скрывает от клиента подробности преобразования различных интерфейсов.

Недостатки:

* Усложняет код программы за счёт дополнительных классов.

*Пример реализации на c#:*

**using** **System**;

**namespace** **Adapter**

{

**class** **MainApp**

{

**static** **void** Main()

{

*// Create adapter and place a request*

Target target = **new** Adapter();

target.Request(); *// Wait for user*

Console.Read();

}

}

*// "Target"*

**class** **Target**

{

**public** **virtual** **void** Request()

{

Console.WriteLine("Called Target Request()");

}

}

*// "Adapter"*

**class** **Adapter** : Target

{

**private** Adaptee adaptee = **new** Adaptee();

**public** **override** **void** Request()

{

*// Possibly do some other work and then call SpecificRequest*

adaptee.SpecificRequest();

}

}

*// "Adaptee"*

**class** **Adaptee**

{

**public** **void** SpecificRequest()

{

Console.WriteLine("Called SpecificRequest()");

}

}

}

*Источники данных:*

* <https://refactoring.guru/ru/design-patterns/adapter>
* https://ru.wikipedia.org/wiki/Адаптер\_(шаблон\_проектирования)