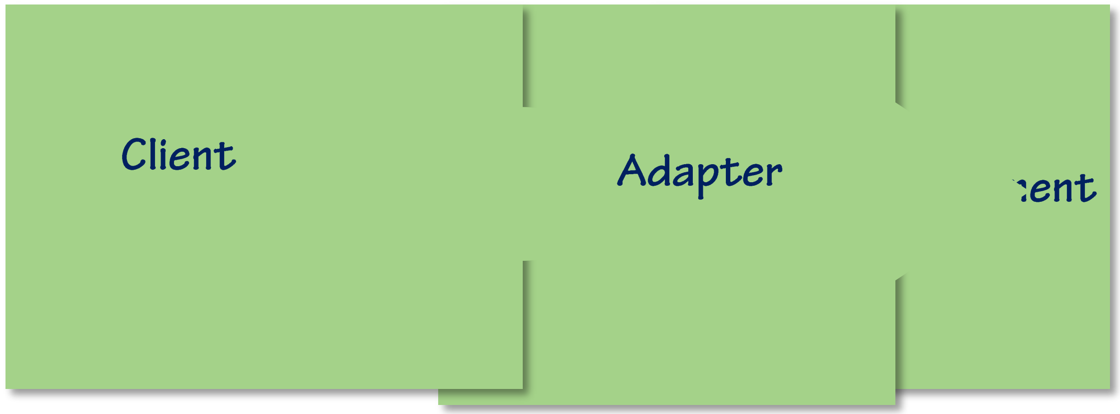
Adapter Pattern

Design Patterns

Motivating Example

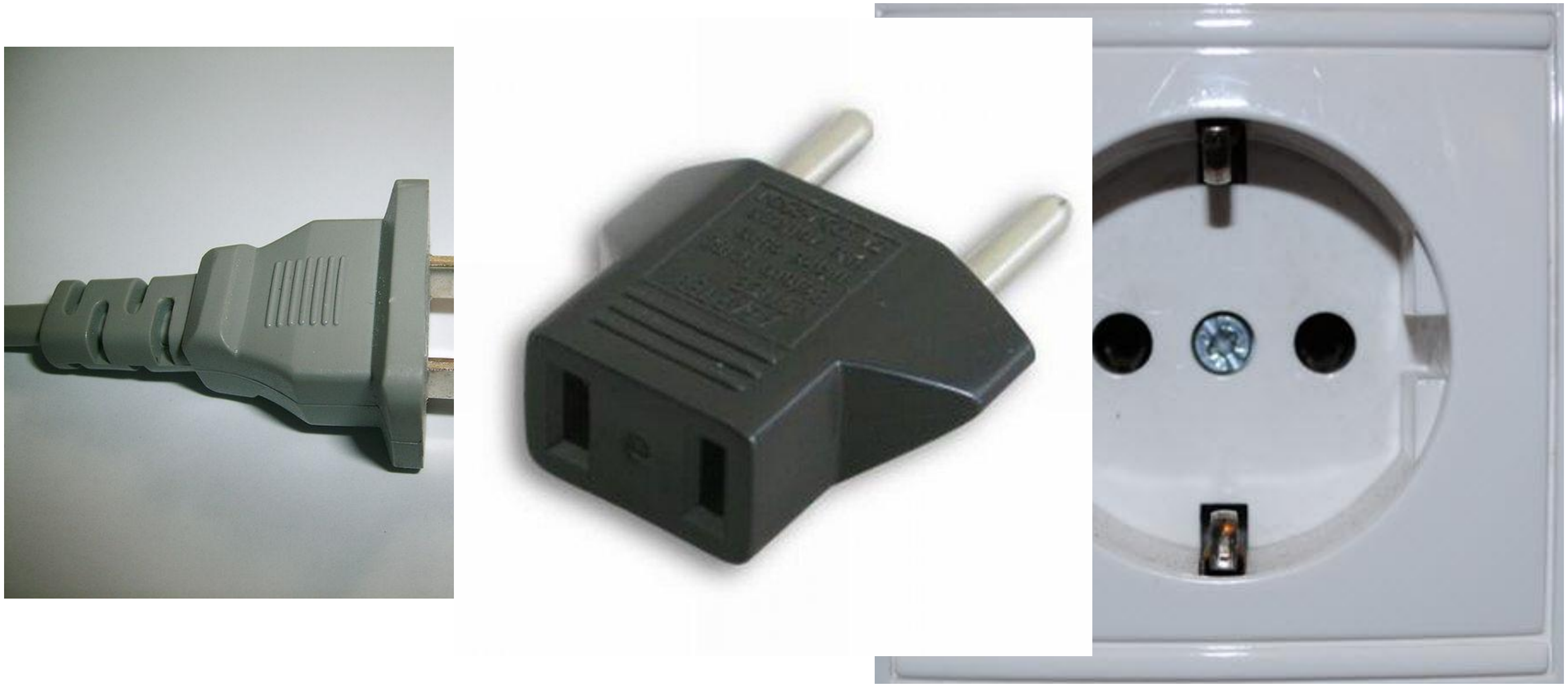
- A class that would be useful to your application does not implement the interface you require
- You are designing a class or a framework and you want to ensure it is usable by a wide variety of as-yet-unwritten classes and applications
- Adapters are also commonly known as *Wrappers*
- In this module, we will refer only to *object adapters*, which do not require multiple inheritance (as *class adapters* do)

Adapter Pattern



Adapters in the Real World

Adapter Pattern



pluralsight
see what you can learn

Intent

Adapter Pattern

- Convert the interface of a class into another interface clients expect.
- Allow classes to work together that couldn't otherwise due to incompatible interfaces.
- *Future-proof* client implementations by having them depend on Adapter interfaces, rather than concrete classes directly

Applicability

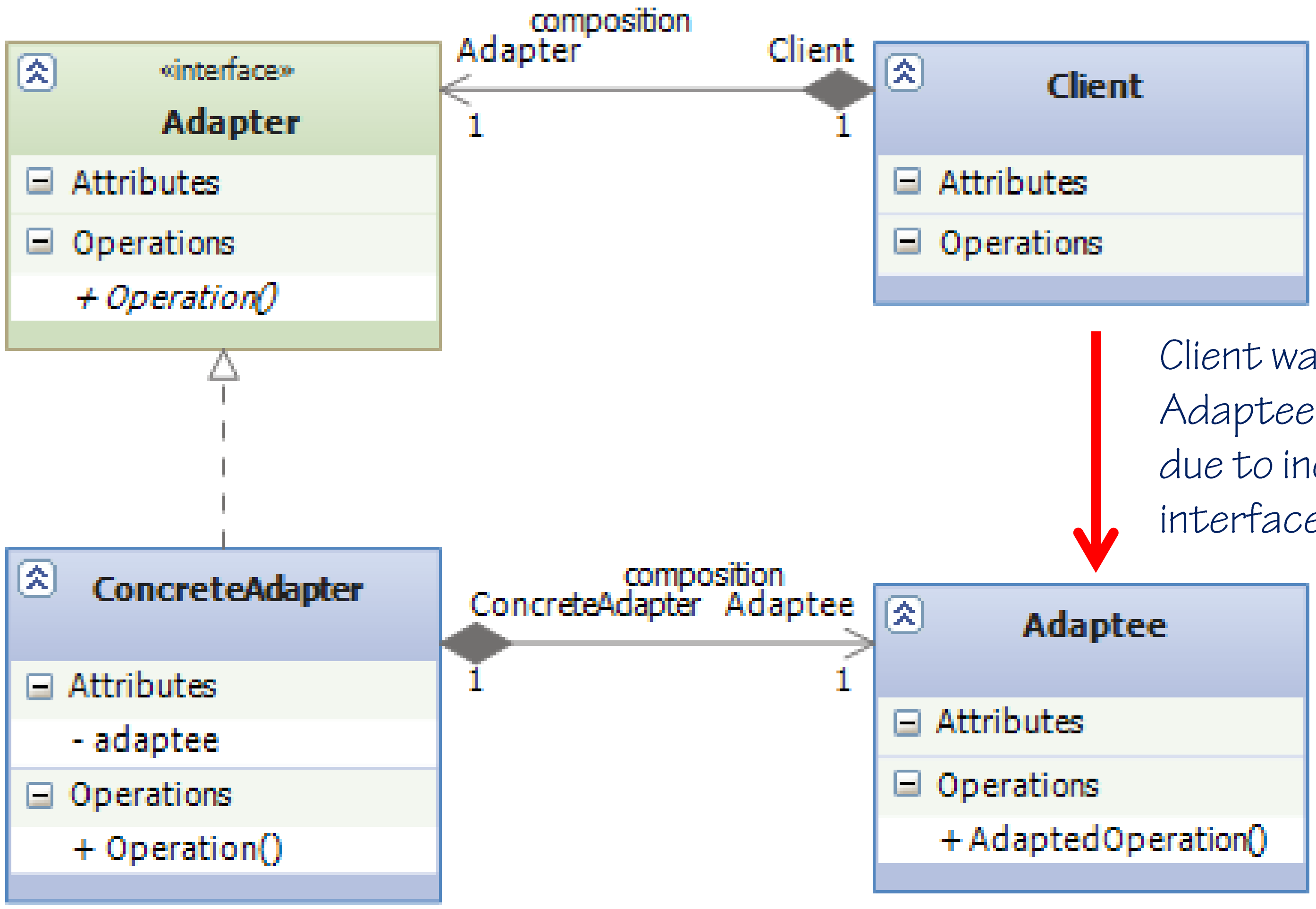
Adapter Pattern

Use the Adapter Pattern when:

- You want to use an existing class, but its interface does not match the one you require
- You want to create a reusable class that cooperates with unrelated or unforeseen classes (i.e. classes that won’t necessarily share the same interface)
- You need to use several existing subclasses, but it’s impractical to adapt their interface by subclassing every one.

Structure

Adapter Pattern



Client wants to use Adaptee directly, but can't due to incompatible interface

How It Gets Used

Adapter Pattern

- Clients depend on the Adapter interface, rather than a particular implementation
- At least one concrete Adapter class is cre client to work with a particular class that
- Future client needs for alternate implem satisfied through the creation of additio classes
- Effective way to achieve Open/Closed Principle

Learn more about the **Open/Closed Principle** in the **Principles of Object Oriented Design** course at Pluralsight On Demand

Collaboration

Adapter Pattern

- Clients call operations on an Adapter instance;
- Adapter instance calls Adaptee operations that carry out the request

Consequences

Adapter Pattern

- **A single Adapter interface may work with many Adaptees**
  - One Adaptee and all of its subclasses
  - Separate Adaptees via separate concrete Adapter implementations

- **Can be difficult to override Adaptee behavior (with *Object Adapter*)**
  - Must subclass Adaptee and add overridden behavior
  - Then, change concrete Adapter implementation to refer to Adaptee subclass

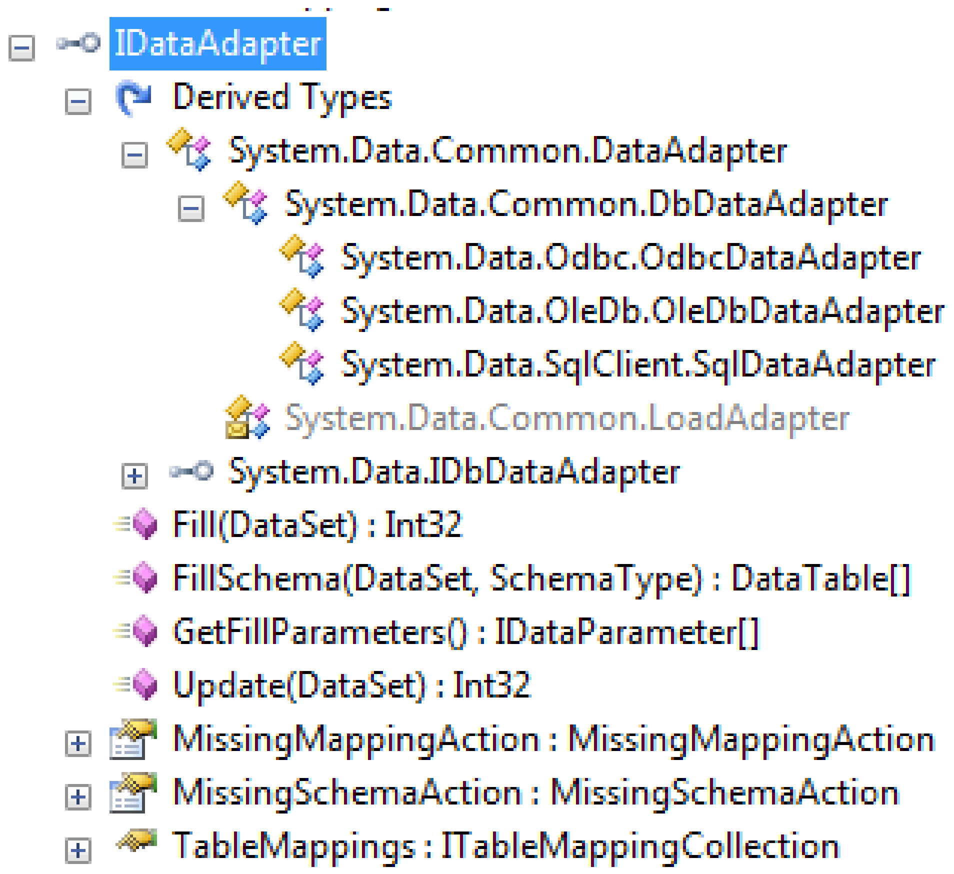
Difficult when compared to using multiple inheritance

Implementation Example

Adapter Pattern

ADO.NET

- IDataAdapter
  - DbDataAdapter
    - OdbcDataAdapter
    - OleDbDataAdapter
    - SqlClientDataAdapter



```
⊟ IDataAdapter
  ⊟ Derived Types
    ⊟ System.Data.Common.DataAdapter
      ⊟ System.Data.Common.DbDataAdapter
          System.Data.Odbc.OdbcDataAdapter
          System.Data.OleDb.OleDbDataAdapter
          System.Data.SqlClient.SqlDataAdapter
        System.Data.Common.LoadAdapter
    ⊞ System.Data.IDbDataAdapter
  Fill(DataSet) : Int32
  FillSchema(DataSet, SchemaType) : DataTable[]
  GetFillParameters() : IDataParameter[]
  Update(DataSet) : Int32
⊞ MissingMappingAction : MissingMappingAction
⊞ MissingSchemaAction : MissingSchemaAction
⊞ TableMappings : ITableMappingCollection
```

Related Patterns

- **Repository**
  - The Repository pattern is a very common use of the Adapter pattern
- **Strategy**
  - The Adapter pattern is often passed into a class that depends on it, thus implementing the Strategy pattern
- **Facade**
  - Adapter and Façade are both wrappers. The Façade pattern attempts to simplify the interface and often wraps many classes, while the Adapter typically wraps a single Adaptee, and is not generally concerned with simplifying the interface.

You can learn more about these patterns in the Pattern Library at PluralSight On Demand.

References

- **Books**
  - Design Patterns, http://amzn.to/95q9ux
  - Design Patterns Explained, http://amzn.to/cr8Vxb
  - Design Patterns in C#, http://amzn.to/bqJgdU
  - Head First Design Patterns, http://amzn.to/aA4RS6
- **Online**
  - http://en.wikipedia.org/wiki/Adapter_pattern

Summary

Adapter Pattern

- The Adapter pattern is used to wrap a needed class with one that implements a required interface.
- By writing client classes so they depend on adapters, we future-proof these classes, ensuring they can be made to work with as-yet-unwritten implementation libraries.
- Remember the Open/Closed Principle:

  Modules should be open to extension, but closed to modification

For more in-depth online developer training visit

pluralsight
see what you can learn

on-demand content from authors you trust