Adapter

A structural pattern

Learning goals

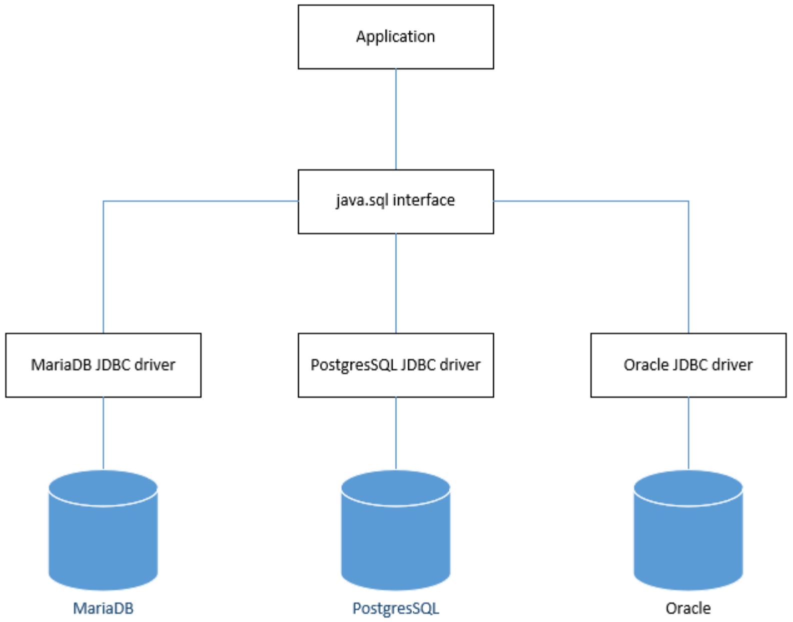
1. Learn the idea, structure, and Java implementation of the Adapter design pattern.
2. Learn to apply the Adapter DP in your own programming.

Idea of Adapter

- Allows incompatible interfaces to work together.
- Transforms the interface of a class into another interface the client expects.
  - This is done by a dedicated adapter class.
- Increases class compatibility and code reusability.

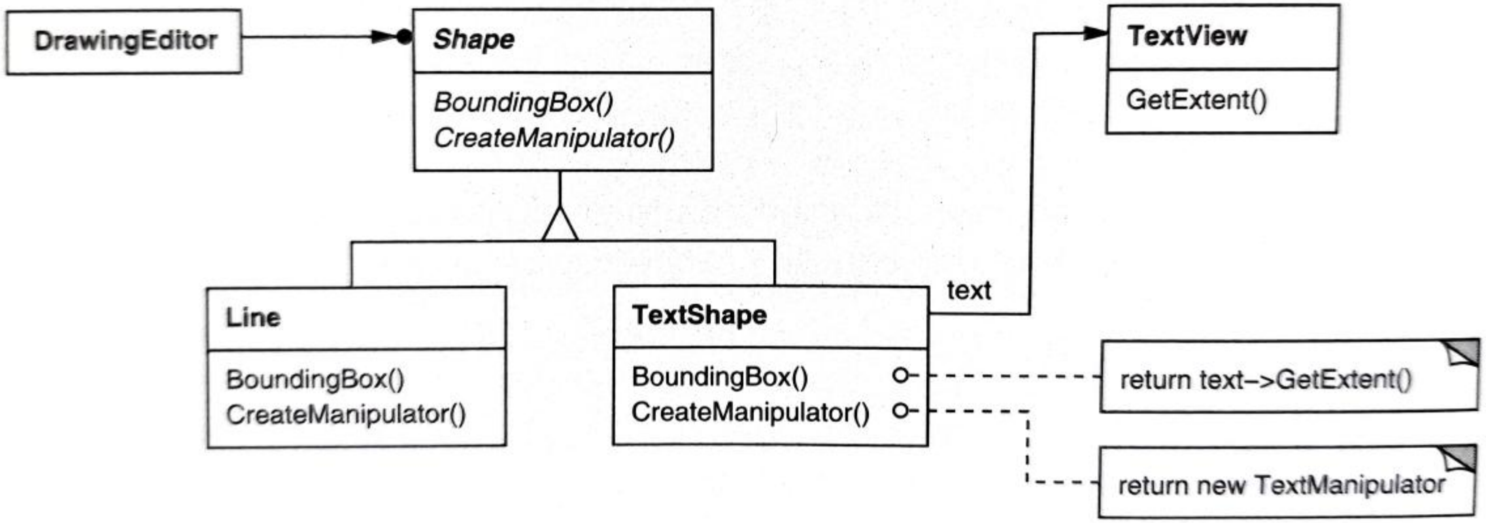
Image: © Raimond Spekking / CC BY-SA 4.0 (via Wikimedia Commons), Asus VGA - Mini-VGA adapter-6605, CC BY-SA 4.0

Example: database drivers



- Database drivers are adapters.
- The application users the same interface irrespective of the underlying database management system.

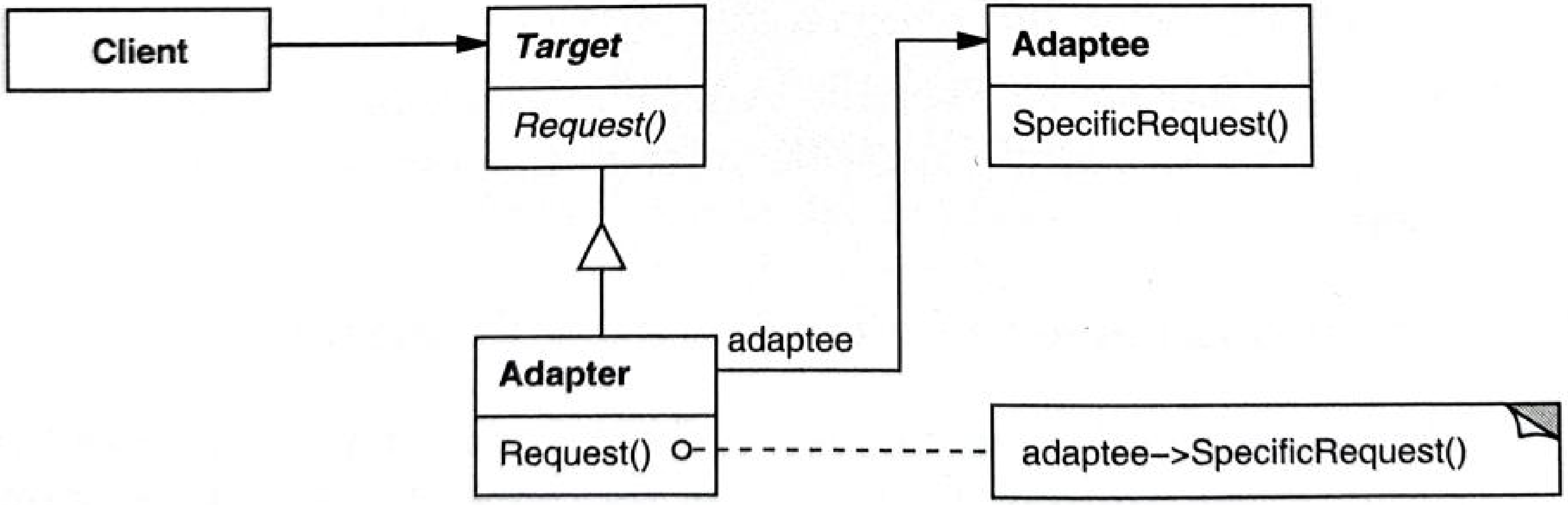
Example: Drawing editor



- The application deals with Shape objects.
- However, implementation of a TextShape is complicated (buffering, rendering, scrolling, etc.)
- Solution: use an off-the-shelf TextView class that implements such complex behaviour.
- Make TextShape act as an adapter that translates the TextView's interfacte to Shape's.

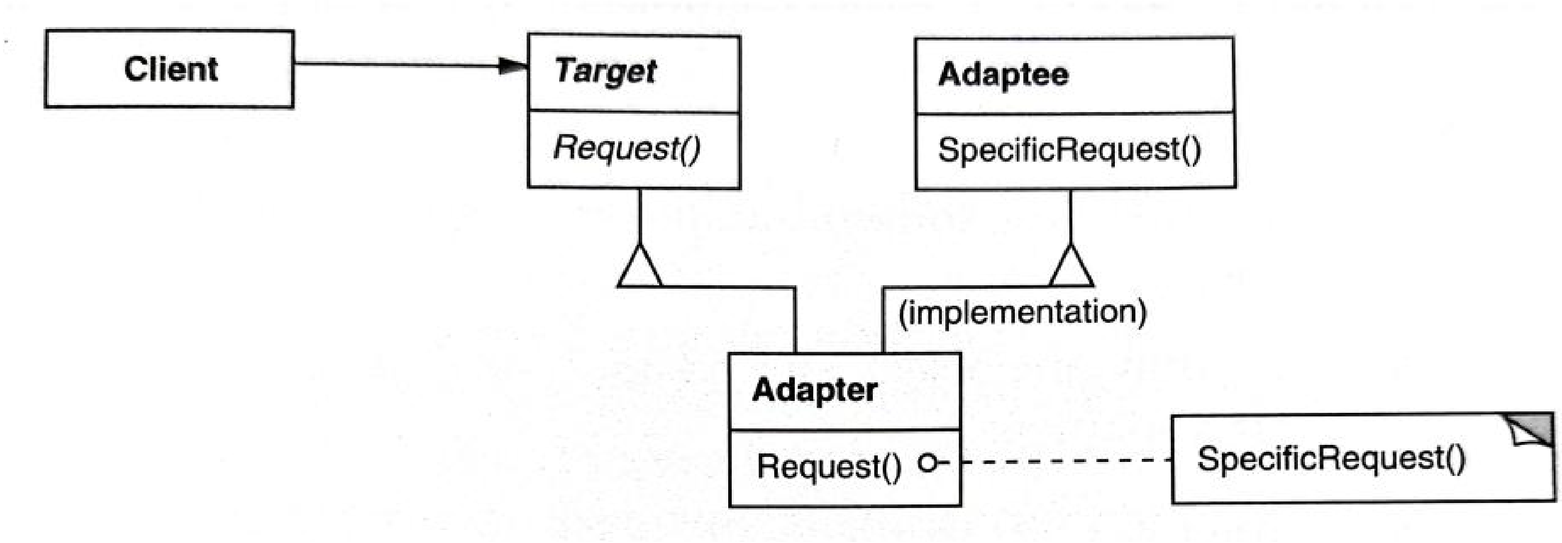
Image: Gamma et al., Design Patterns. Elements of Reusable Object-Oriented Software. Addison Wesley Longman (1995), p. 208

General structure (object adapter)



- An object adapter wraps the Adaptee (contains a reference to the Adaptee object)

General structure (class adapter)



- A class adapter implements both Target and Adaptee interfaces.
- Note: Java doesn't support multiple inheritance present in the original Gamma's diagram.
  - Solution: the Adapter implements, not extends, the Target.

Roles

- **Target Interface**: Defines the domain-specific interface that Client uses.
- **Client**: Interacts with objects conforming to the Target Interface.
- **Adaptee**: An existing interface that needs adapting.
- **Adapter**: Adapts the interface of the Adaptee to the Target Interface.

Practical issues

- Adapters can even add functionality that is not present in the adaptee.
- The adapters can vary in complexity much.
- An adapter may provide default behavior that allows it to function even without a wrapped adaptee.
- An object adapter can utilize several adaptee classes (superclass and subclasses)
  - On the other hand, a class adapter extends a concrete adaptee class.