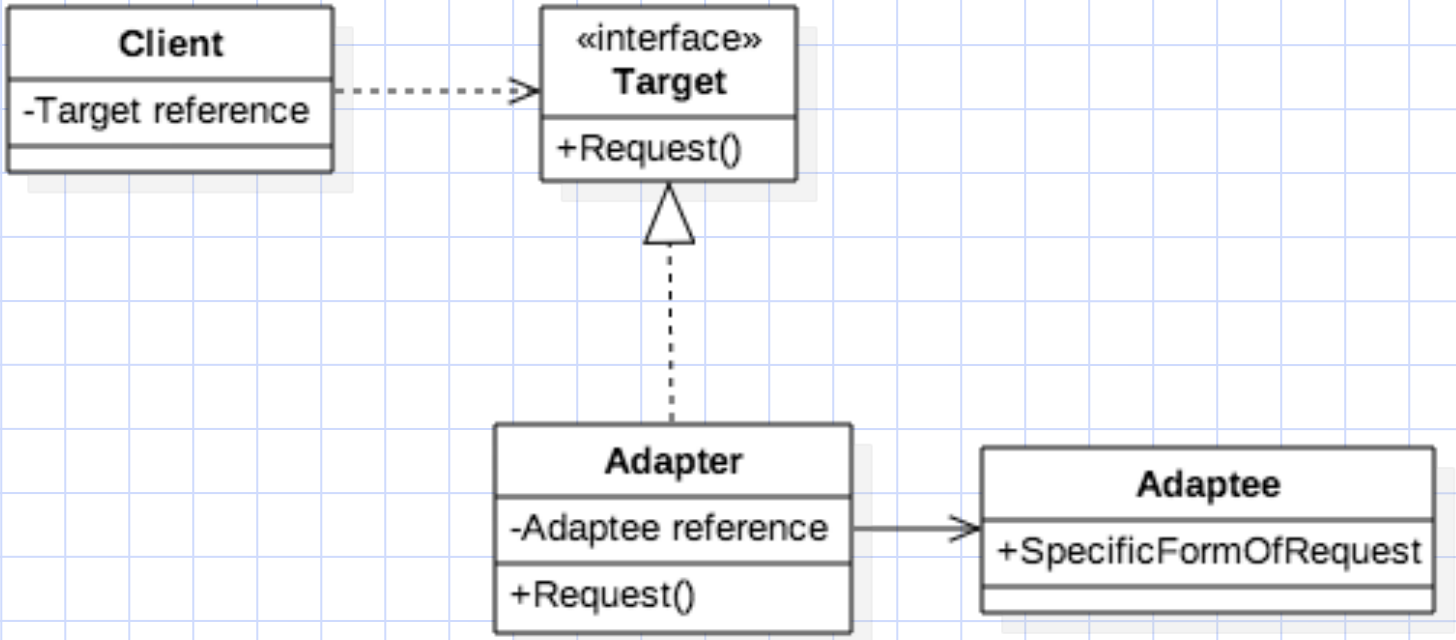
Design Pattern: Adapter

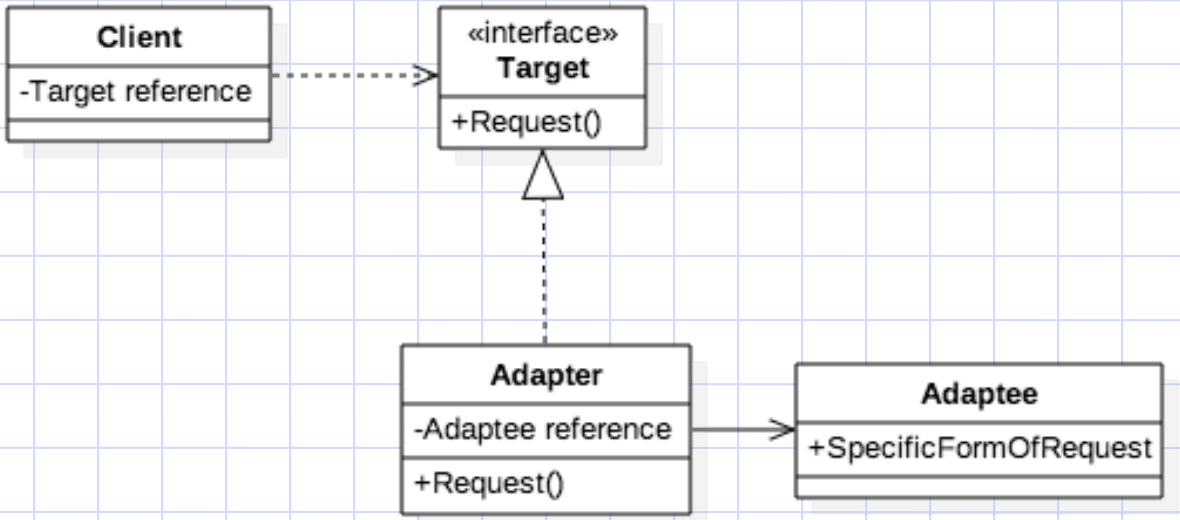
Design Pattern: Adapter

The Adapter Pattern from GoF

- **Intent**
  - Convert the interface of a class into another interface acceptable to the client.
    - Wrap an existing class with a new interface.
  - Allow incompatible classes work together



Participating Classes



Target: defines the domain-specific interface that Client likes.

Adapter: adapts the interface Adaptee to the Target interface.

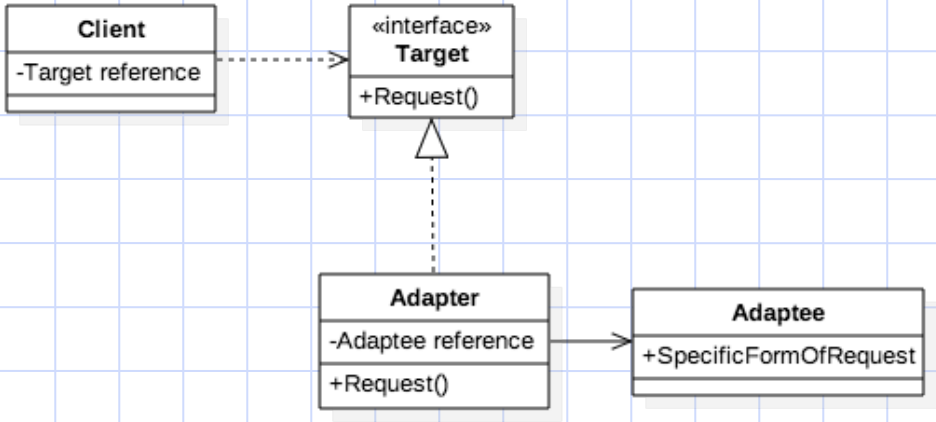
Adaptee: defines an existing interface that needs an adapter to become compatible to target.

Client: collaborates with objects conforming to the Target interface.

General Template to Develop Adapter Pattern in Java

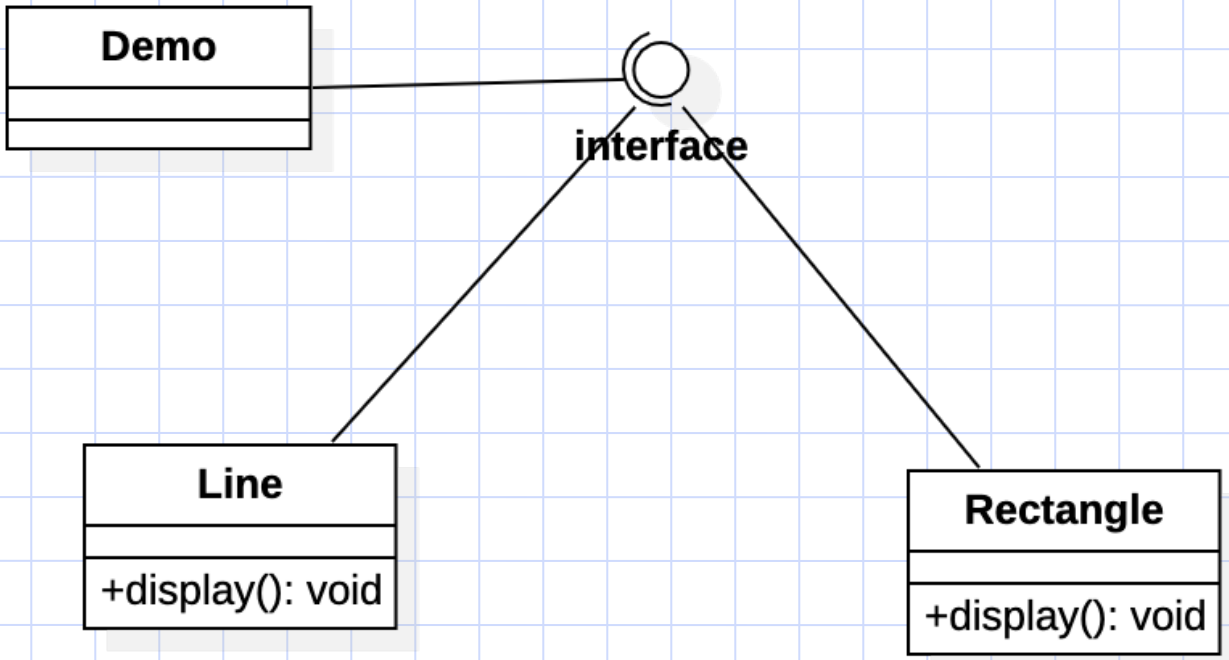
```
class Adaptee {
  legacyMethod(…) {
  }

};

interface Target{
 clientMethod(…);
}

// a wrapper class
class Adapter implements Target {
  clientMethod(…) {
     adapteeMethod(…)
     // MORE
  }
}
```

```
class Client{
  useAdapter() {
     Target x = new Adapter();
     x.clientMethod(…);
  }

};
```



Class Exercise

- Let's assume we would like to use a legacy code for a few geometric shapes (line, rectangle), and a client needs to use an adapter, as client's interface doesn't match with the legacy code.



Now Let's Learn More By An Example

- Let's assume we would like to use a legacy code for a few geometric shapes (line, rectangle), and a client needs to use an adapter, as client's interface doesn't match with the legacy code.