Convert the interface of a class into another interface that the client(s) expect. Adapter lets classes work together that could not otherwise due to incompatible interfaces.
Object Adapter

- Class contains an instance of the class it wraps (smells like Decorator!)
- Adapter makes calls to the instance of the wrapped object
Object Adapter

```
Client
+adaptor: Adaptor
+doWork()

... adaptor.methodA();
...

Adaptee
+methodB()

Adaptor
+adaptee: Adaptee
+methodA()

... adaptee.methodB();
...```
Object Adapter
Class Adapter

- Uses multiple inheritance to achieve its goal
- It is created inheriting from both the interface that is expected and the interface that is pre-existing
- It is typical for the expected interface to be created as a pure interface class (only abstract) methods – necessary in Java which does not support multiple inheritance
Class Adapter

```
Client
+adaptor: Adaptor
+doWork()

... adaptor.methodA();
...

Adaptee1
+method1()

......

AdapteeN
+methodN()

Adaptor
+methodA()

... method1();
...
... methodN();
```
Class Adapter
Dependency Inversion Principle

- Adapter follows this
  - High-level class defines its own (adapter) interface to low-level module (which is implemented by Adaptee class)
Example from DPE (Object Adapter)

- Inheritance hierarchy of shapes, with Shape as superclass
- Subclasses are Point, Line, and Square
- You come across a SpecialCircle class that already exists and is not part of the hierarchy
- We can develop a Circle class that is part of the hierarchy
  - Circle wraps SpecialCircle and delegates to it for particular behavior as necessary
  - Circle is the Adapter, SpecialCircle is Adaptee
Adapter and Facade are Similar

• Both are structural
• Both work with pre-existing classes
Differences between Adapter and Facade

- Adapter has an interface that must be designed to
- Adapter typically requires an object to behave polymorphically
- With Facade, a simpler interface is needed – this is not why Adapter is used
- Facade simplifies an interface; Adapter converts a pre-existing interface into another interface
Adapter questions

• What design principles does Adapter follow if any?
• Real life (non code) Adapter examples?
• What is the biggest benefit of applying the Adapter pattern?