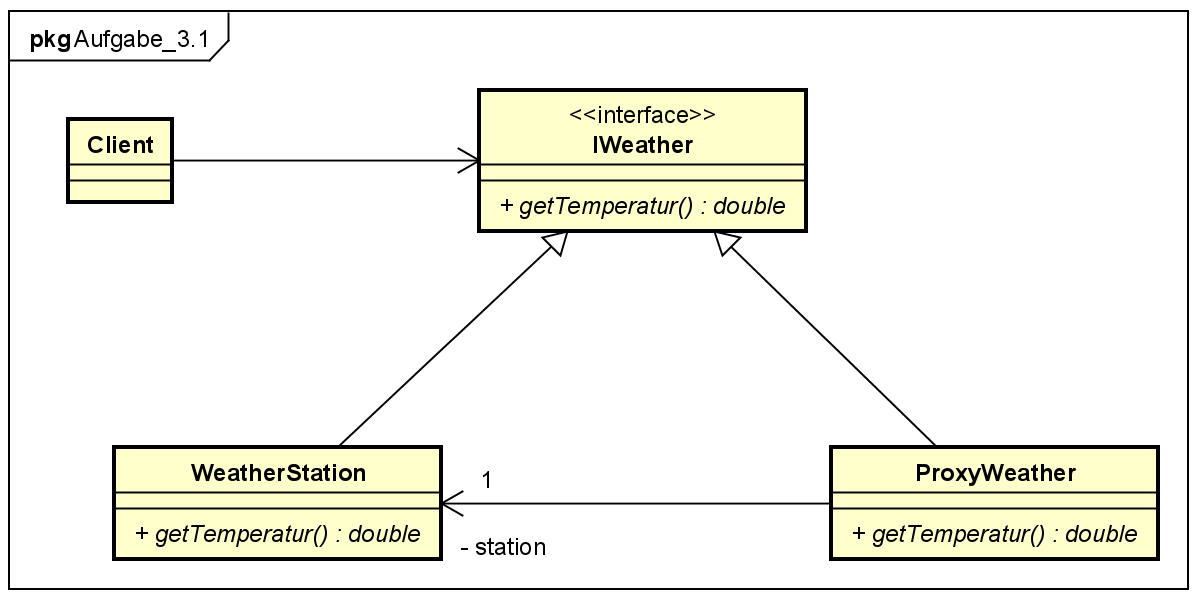
Aufgabe 3.1



public interface IWeather {  
 double getTemperature();  
}

public class WeatherStation implements IWeather {  
 public WeatherStation() {  
 }  
  
 @Override  
 public double getTemperature() {  
 return Math.floor(Math.random()\*120);  
 }  
}

public class ProxyWeather implements IWeather {  
 WeatherStation station;  
  
 public ProxyWeather(WeatherStation station) {  
 this.station = station;  
 }  
  
 @Override  
 public double getTemperature() {  
 double tempInFahrenheit = station.getTemperature();  
 return Math.floor((tempInFahrenheit - 32) \* 5/9);  
 }  
}

public class Main {  
 public static void main(String[] args) {  
 WeatherStation station = new WeatherStation();  
 ProxyWeather proxyWeather = new ProxyWeather(station);  
  
 System.out.println("Fahrenheit: " + proxyWeather.getTemperature());  
 System.out.println("Celsius: " + station.getTemperature());  
 }  
}

Aufgabe 3.2

Gemeinsamkeiten:

* Bei beide Delegationen der Methoden, Original dabei unverändert
* Referenzieren anderes Objekt
* Alle implementieren das gleiche Interface

Unterschiede:

* Proxy kontrolliert Zugriff
* Decorator: Ketten von Objekten, dynamisch beliebig viele Objekte, erweitert Objekt um Funktionalität

Aufgabe 3.3