Adapter

```
/*Definisce un'interfaccia con il metodo method(Data data) che deve essere implementato da
qualsiasi classe adattatrice.*/
public interface ClientInterface {
	public Object method(Data data);
}
public class Data {
	private Object data;
	public Data() {
		System.out.println("[CLIENT]: (input) client data format");
		this.data = "client data format";
	}
	public Object getData() {
		return data;
	}
}
/* Contiene un metodo serviceMethod(SpecialData specialData) che lavora con un tipo di dato
specifico SpecialData. La classe SpecialData è un inner class che contiene un oggetto generico
specialData. */
public class Service {
	public Service() {
		System.out.println("[SERVICE]: (request) require special data");
	}
	public Object serviceMethod(SpecialData specialData) {
		if(specialData.equals(new SpecialData())) {
			return "[SERVICE]: (output) correct format, service available";
		}
		return "[SERVICE]: (output) uncorrect format, service unvailable";
	}
	// Special Data
	class SpecialData {
		private Object specialData;
	public SpecialData(Object specialData) {
			this.specialData = specialData;
		}
		public SpecialData() {
			this.specialData = "special data";
		}
		public Object getSpecialData() {
			return specialData;
		}
		@Override
		public boolean equals(Object obj) {
			if (this == obj) {
				return true;
			}
			if (obj == null || getClass() != obj.getClass()) {
				return false;
			}
			SpecialData that = (SpecialData) obj;
			return specialData.equals(that.specialData);
		}
	}
}
```

```
public class ClassAdapter extends Service implements ClientInterface {
	@Override
	public Object method(Data data) {
		SpecialData specialData = convertToServiceFormat(data);
		System.out.println("[ADAPTER]: convertion finished");
		return serviceMethod(specialData);
	}
	private SpecialData convertToServiceFormat(Data data) {
		System.out.println("[ADAPTER]: converting into required service format");
		return new SpecialData();
	}
}

public class ObjectAdapter implements ClientInterface {
	Service adaptee;
	public ObjectAdapter(Service adaptee) {
		this.adaptee = adaptee;
	}
	@Override
	public Object method(Data data) {
		SpecialData specialData = convertToServiceFormat(data);
		System.out.println("[ADAPTER]: convertion finished");
		return adaptee.serviceMethod(specialData);
	}
	private SpecialData convertToServiceFormat(Data data) {
System.out.println("[ADAPTER]: converting into required service format");
		return adaptee.new SpecialData();
	}
}
public class Client {
	public static void main(String[] args) {
		new Client().goObjectAdapter();
		System.out.println(".........................");
		new Client().goClassAdapter();
	}
	void goObjectAdapter() {
		Data data = new Data();
		Service service = new Service();
		ObjectAdapter adapter = new ObjectAdapter(service);
		System.out.println(adapter.method(data));
	}
	void goClassAdapter() {
		Data data = new Data();
		ClientInterface adapter = new ClassAdapter();
		System.out.println(adapter.method(data));
	}
}
```