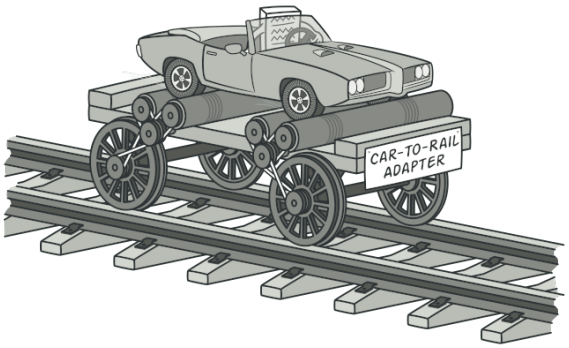
Design Patterns

Edwin Carlinet

The adpater



Adapter is a structural design pattern that allows objects with incompatible interfaces to collaborate.

Exercise

You have a project that currently uses *Curl* a third-party library for making HTTP requests. The whole codebase is built around the *Curl* library and its interface, but… 💥 damned ! Curl is not available on your brand-new client's server, only *wget* is.

Curl's & WGet's interfaces

Figure 1: Curl and WGet API

Example of code use:

```
def fetch_data(http_client: CurlRequester):
    response = http_client.get('https://api.example.com/data')
    print(response)
```

How do you adapt the existing codebase to work with *wget* with minimal changes to the code that uses Curl? :thinking:

Exercise 2 📅

You are tasked with designing a system to facilitate the planning of courses for a university. The system should allow administrators to schedule courses, assign instructors, allocate classrooms…

The system should support the creation, modification, and deletion of courses. Each course has attributes such as title, description. Courses have many sessions of varying durations and types: such as lectures, labs, and tutorials… Courses should be assigned to instructors. Each instructor has attributes such as name, email, and specialization. A course may have multiple instructors. A session should be assigned to a classroom. Each classroom has attributes such as capacity, equipment availability, and suitability for the course type. Students are enrolled in courses and attend sessions by groups.

❓How would you design the system to support these requirements? Perform the analysis and represent classes with their attributes and relationships.