Modeling historical evaluations with power of DI

.NET Chapter Meetup

Luxoft
A DXC Technology Company

The Speaker

Illia Levandovskyi

More than a decade of professional software development experience

Lead software developer

Global .NET chapter lead at Luxoft

Luxoft’s training center trainer

Illia.Levandovskyi@dxc.com

Tax Calculator

TaxCalculator.Seed 1.0 OAS3

https://localhost:7271/swagger/v1/swagger.json

TaxCalculator

GET /api/TaxCalculator/Calculate

Parameters

Name	Description
income number($double) *(query)*	5000000
year integer($int32) *(query)*	2022

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7271/api/TaxCalculator/Calculate?income=5000000&year=2022' \
  -H 'accept: */*'
```

Request URL

```
https://localhost:7271/api/TaxCalculator/Calculate?income=5000000&year=2022
```

Server response

Code	Details
200	Response body 551500

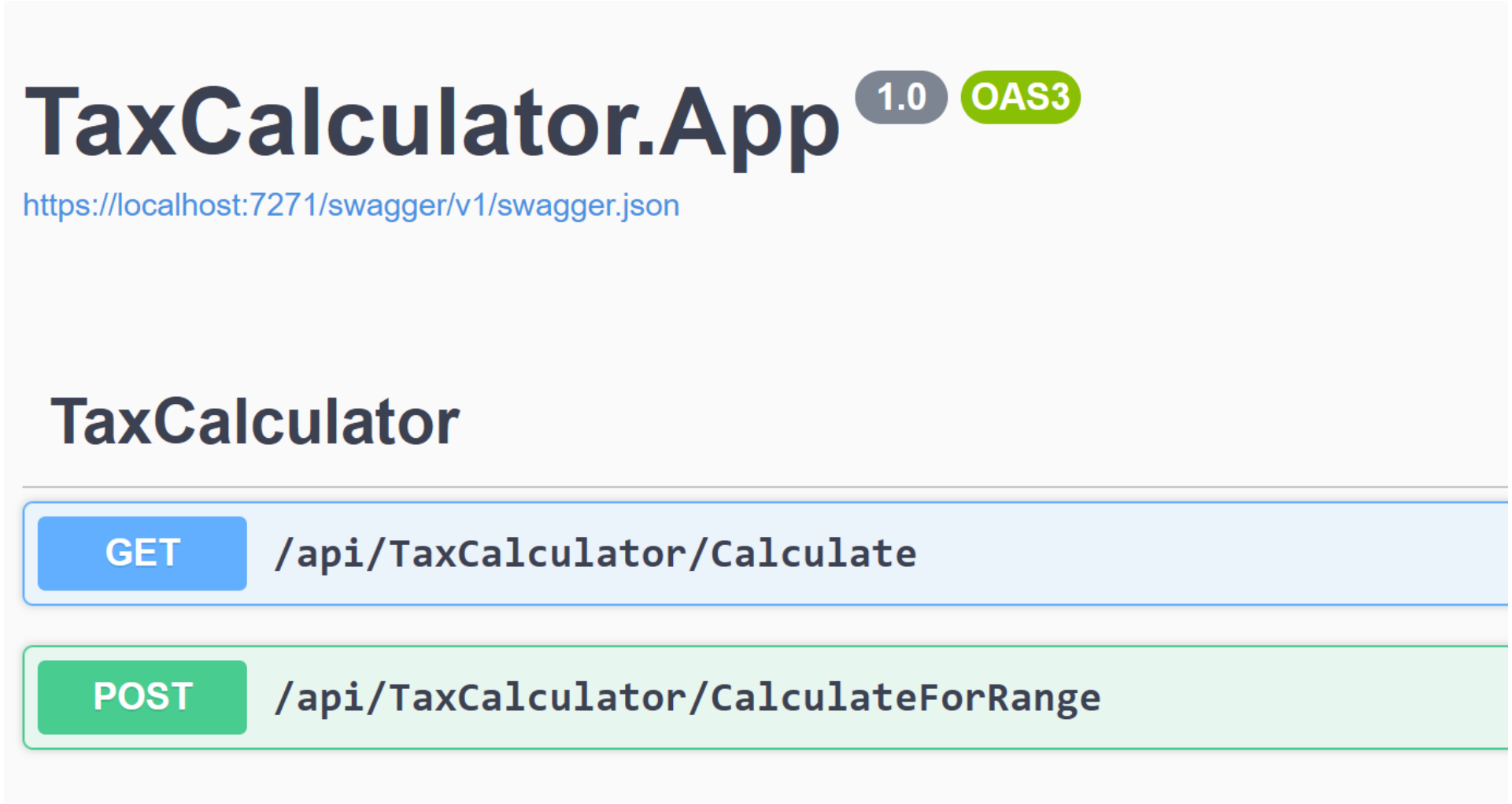
New Requirements

now the customer wants us to have historical simulations based on specific inputs

```
{
  "incomeByYear": {"2021": 1 500 000,"2022":1 000 000,"2023":2 000 000},
  "rulesByYear": {
    "2021": [
        { "upperBound": 1 000 000, "rate": 3, "fixedPayment": 750 },
        { "upperBound": 2 000 000, "rate": 4, "fixedPayment": 1500 },
    ]
  }
}
```

they want to be able to customize taxation rules per specific years on a given run

Historical Evaluations

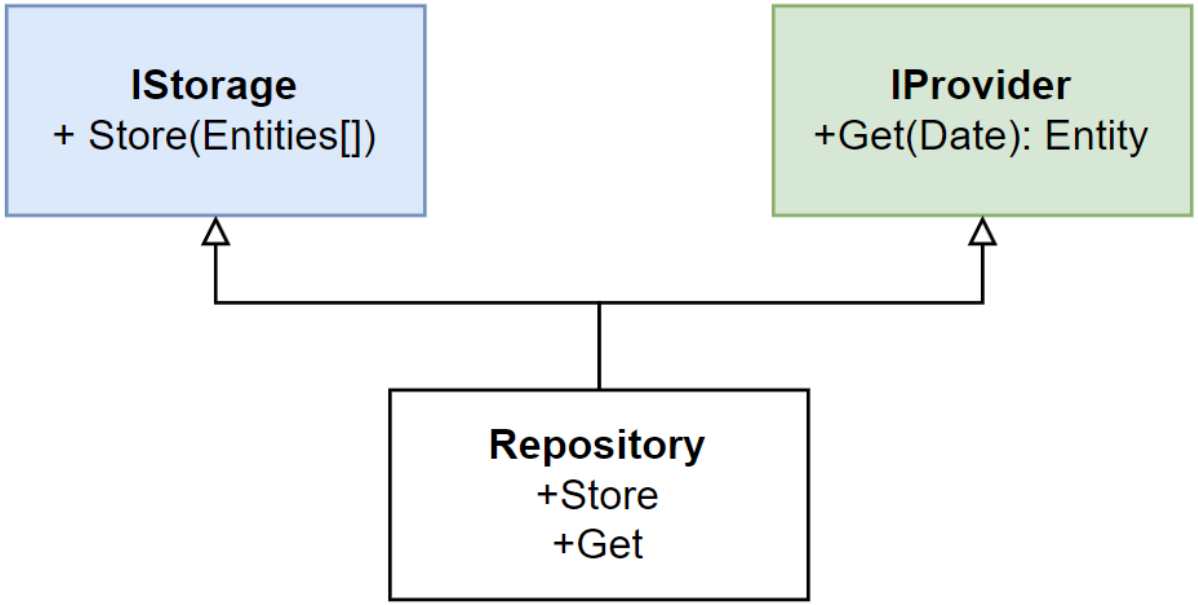


How can we reuse existing code?

The Triade Pattern

A suggestion is to introduce a triade: storage, repository and provider

- **Storage** interface to store user input (parameters customization).
- **Repository** class to keep these values.
- **Provider** interface to consume these values in previously existing service.

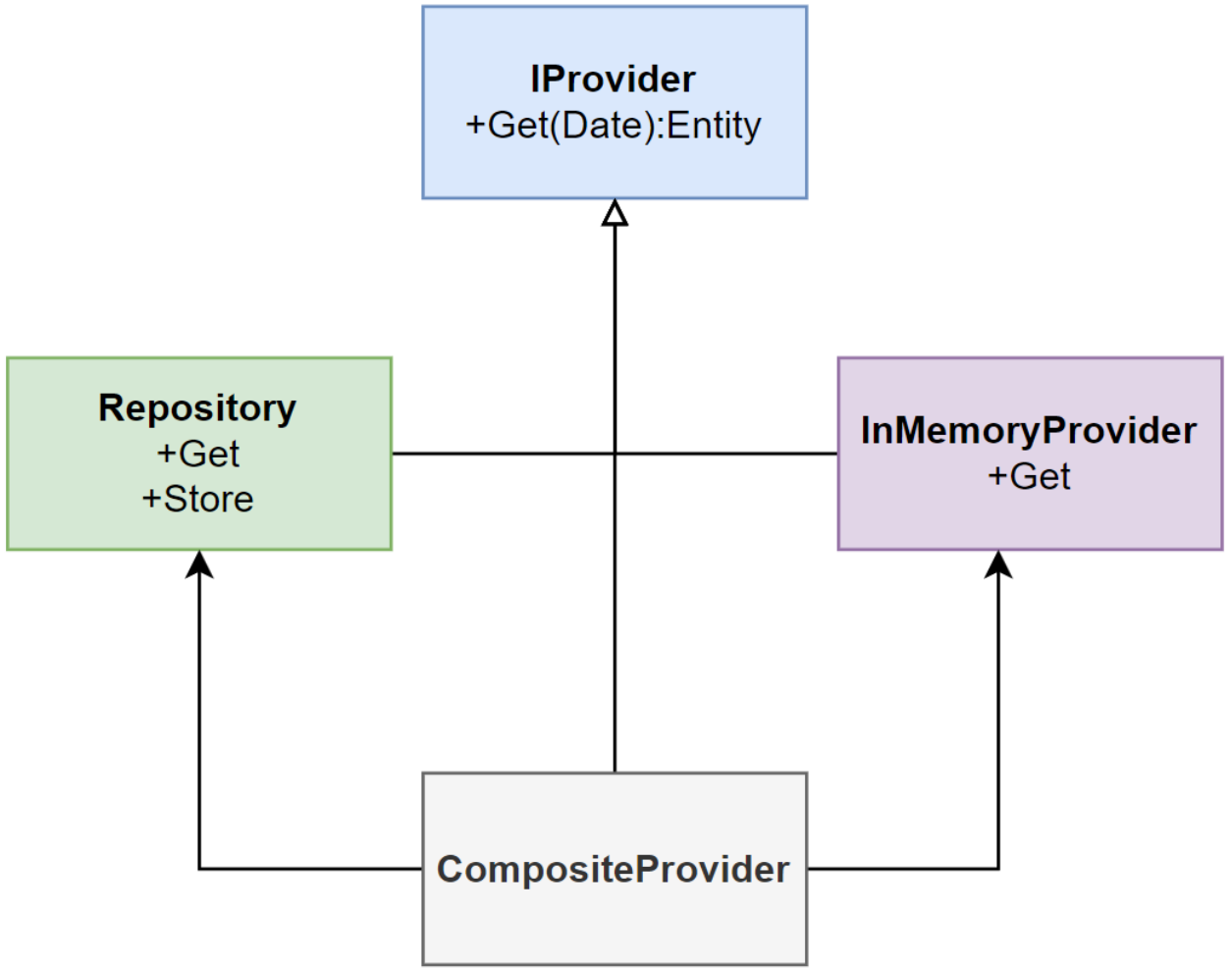


Merging options

The Provider might be used together with composite or decorator patterns.

It opens a room for different options:

- merge
- override
- substitute



Summary

We’ve reused existing code

ISP enables us to clearly define components roles per context

Composite design pattern allows us
to combine different sources of data in a controlled manner

DI’s composition root clearly shows
whether we instantiate the triade pattern properly:

the same instance of the repository is registered
as the provider
as the storage