Taking Advantage of Loose Coupling

Jeremy Clark
DEVELOPER BETTERER

@jeremybytes www.jeremybytes.com

My Favorite Part

Using DI

Change data readers

- Text file
- SQL database

Add a client-side cache

SOLID

Requests from the Boss

- Different data sources
- Client-side cache
- Unit tests

Request 1: Different Data Sources

Web service

Text file

SQL database

Document database

Cloud service

Azure functions

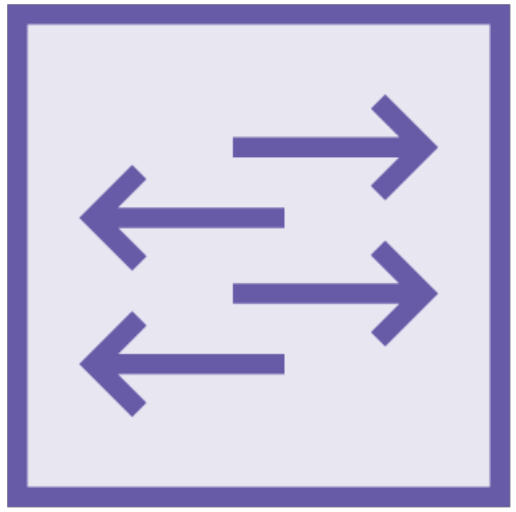
Create a text file data reader

Snap pieces together in a different order

Changing the Data Source

View

Presentation



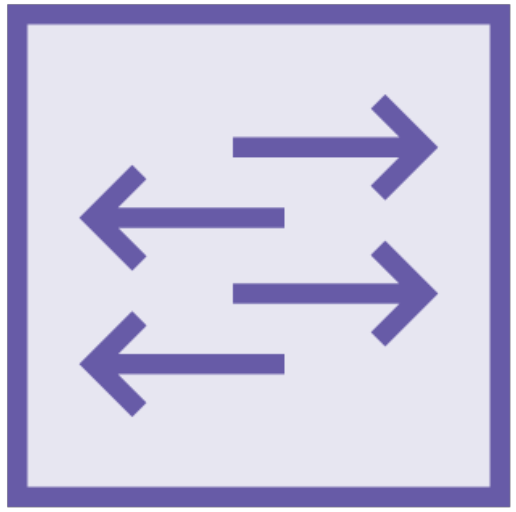
Service reader

Web service

Changing the Data Source

View

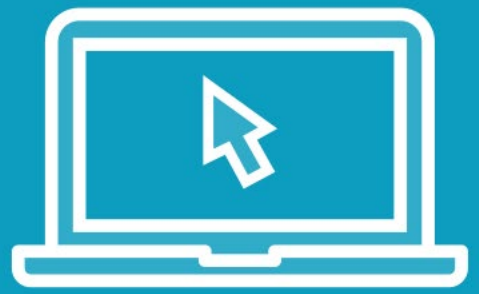
Presentation



CSV reader

Text file

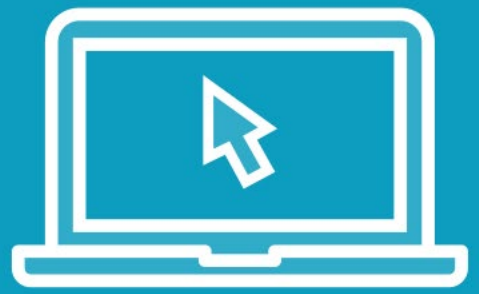
Demo



Add a text file data reader

Snap pieces together

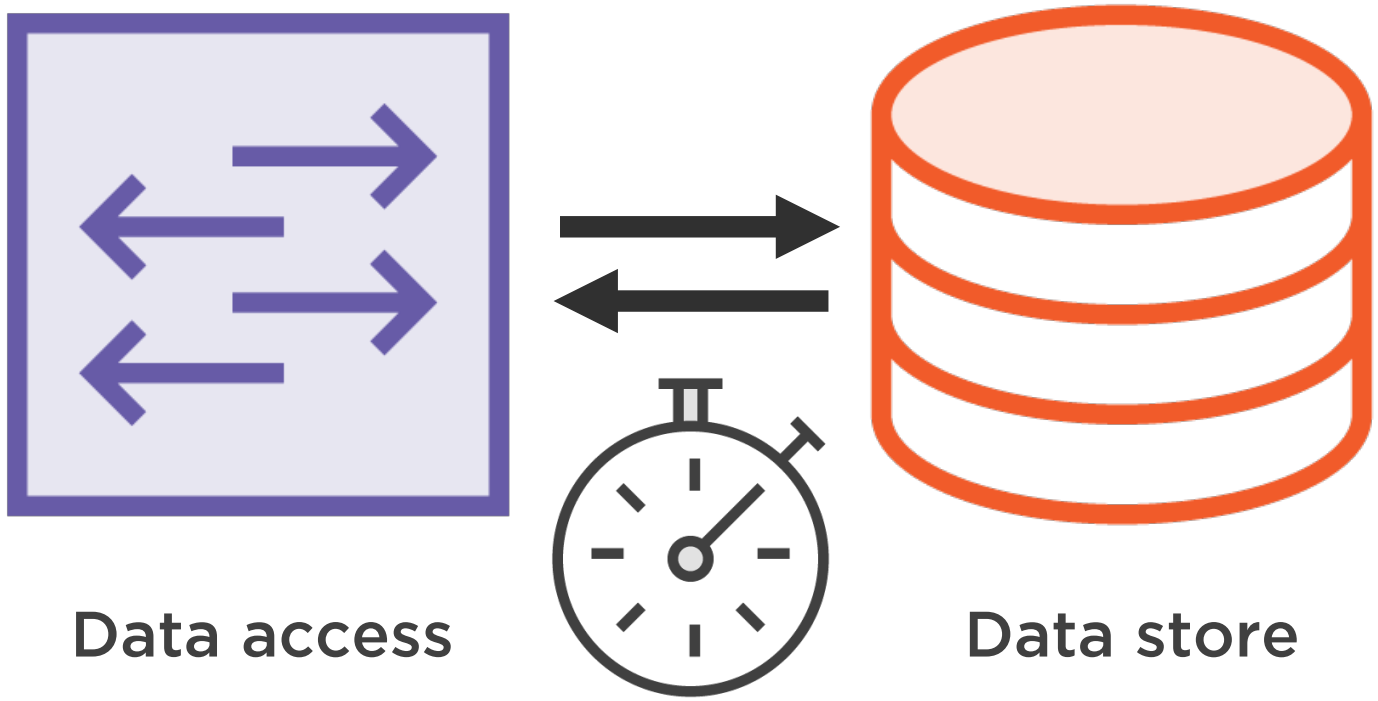
Demo



Add a SQL database data reader

Snap pieces together

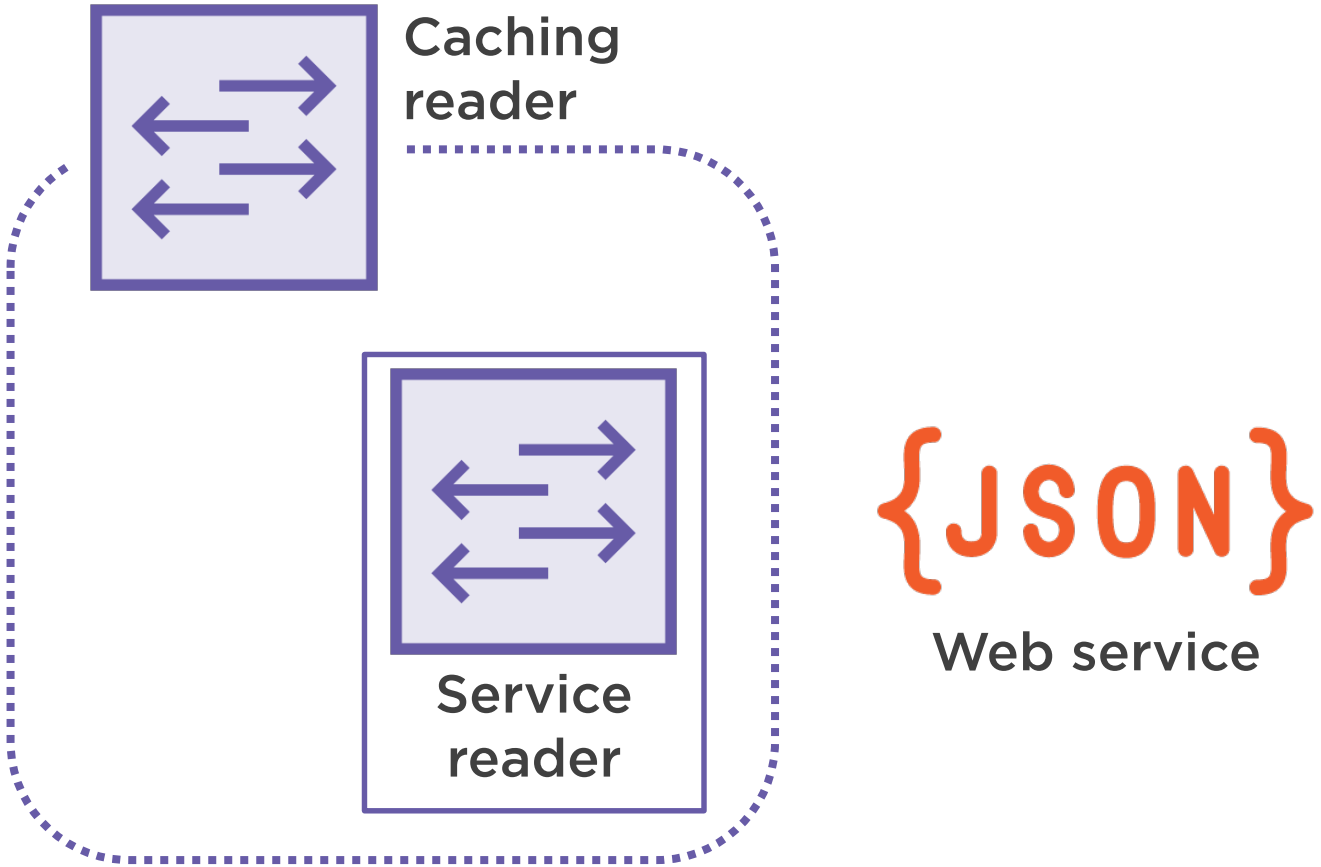
Request 2: Client-side Cache



Decorator

Wrap an existing interface to add functionality

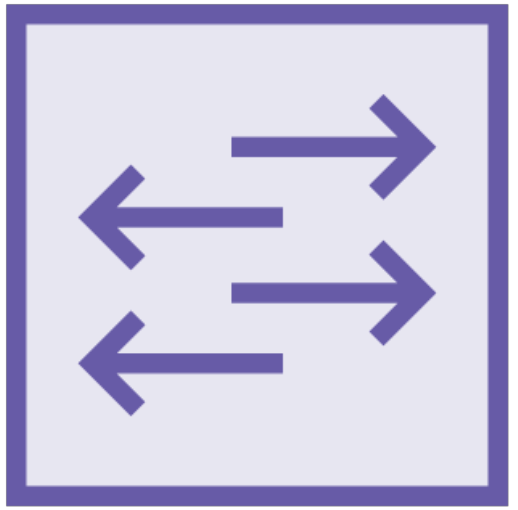
Data Reader Decorator



Composing Objects

View

Presentation



Service reader

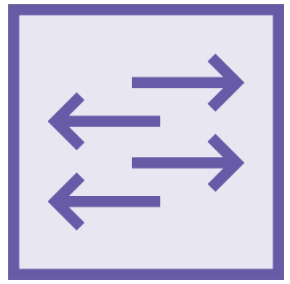
Web service

Composing Objects

View

Presentation

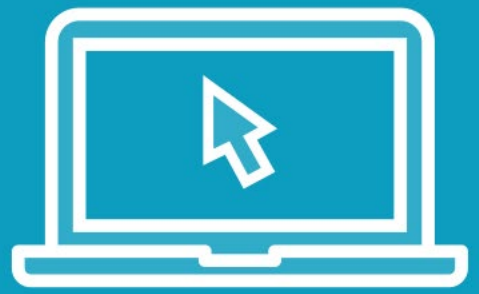
Caching reader



Service reader

Web service

Demo



Add a caching data reader

Snap pieces together

Injecting a Different Data Reader

View model does not change

View does not change

Snap pieces together differently

Changing the Data Source

```
private static void ComposeObjects()
{
    var reader = new ServiceReader();
    var viewModel = new PeopleViewModel(reader);
    ...MainWindow = new PeopleViewerWindow(viewModel);
}
```

```
private static void ComposeObjects()
{
    var reader = new CSVReader();
    var viewModel = new PeopleViewModel(reader);
    ...MainWindow = new PeopleViewerWindow(viewModel);
}
```

Injecting a Client-side Cache

- View model does not change
- View does not change
- Existing data readers do not change
- Snap pieces together differently

Injecting a Client-side Cache

```
private static void ComposeObjects()
{
    var reader = new ServiceReader();
    var viewModel = new PeopleViewModel(reader);
    ...MainWindow = new PeopleViewerWindow(viewModel);
}
```

```
private static void ComposeObjects()
{
    var wrappedReader = new ServiceReader();
    var reader = new CachingReader(wrappedReader);
    var viewModel = new PeopleViewModel(reader);
    ...MainWindow = new PeopleViewerWindow(viewModel);
}
```

O

- Open/Closed Principle

Existing data readers can be extended without being modified

Other Decorators

Authorization

Retry

Logging

L

- Liskov Substitution Principle

Application can use the caching reader just like any other data reader

Using DI

Change data readers

- Text file
- SQL database

Add a client-side cache

SOLID