`IDataSource` Interface

Overview

The `IDataSource` interface serves as a protocol for managing data sources in the Heatington application. It stipulates the provision of methods to perform read and write operations on data.

Methods

`Task<List<DataPoint>?> GetDataAsync(string filePath)`

The `GetDataAsync` method signifies an asynchronous operation for retrieving data from a specified data source. The input `string filePath` represents the path to the file containing the pertinent data.

The method is expected to return a `List<DataPoint>` object, which contains the data of interest - specifically the heat demand and electricity price details. In cases where the data retrieval is unsuccessful or if there is no data present at the given file location, the method may return `null`.

`void SaveData(List<DataPoint> data, string filePath)`

The `SaveData` method is purposed towards storing `DataPoint` objects into a CSV file located at a specific file path. The `List<DataPoint> data` argument contains the data points that are set to be saved. The `string filePath` is an argument that provides the location at which the CSV file will be written to or overwritten.

Considering that its return type is `void`, all complications that arise during the data-saving process should be conveyed via exceptions.

This method is currently not implemented

Implementations

Any classes that function as Csv data sources within the Heatington application should implement this interface. This allows for consistency in the management of data across varying data sources and enables smoother transitions between different data sources. Examples of such classes could include `CsvDataSource`, `XmlDataSource`, and the like.

IReadWriteController Interface

Namespace: `Heatington.Controllers.Interfaces`

Description

Generic Interface for all controllers performing I/O operations.

Members

`Task<T?> ReadData<T>()`

Function for reading data out of a model.

Returns

Task to wait for Data.

`Task<OperationStatus> WriteData<T>(T content)`

Function for Writing data into the model.

Parameters

- `content`: Content to write into the model, generic type.

Returns

Task to wait for status of the operation.