County Data Framework

17-214: Homework 5

Team 28: Javier Cifuentes, Kunal Barde, Neil Khera

Framework Domain

DATA ENTRY:

Required county name, state name, and time, and numeric value field

Sources:

- xls
- csv
- Web
- etc.

List of Entries →

FRAMEWORK:
Groups data into custom data structure with library methods

- Data in relation to time
- Grouping counties
- Filtering
- Mathematical analyses (sum, avg, stdDev, etc)

Object w/ API →

DATA VISUALIZATION:

Display visualisation for individual counties, states, or the entire country

- Line plots
- Scatter plots
- Bubble charts
- Choropleths
- Google Earth time functionality

Generality & Specificity

Specificity:

- Numeric data associated to a county in US at a specific year time

Abstractions:

- All data provided has to be specific to a county and time

Reusable functionality:

- Ability to statistically analyze possibly filtered data in our domain
- Display-plugins can filter+analyze data multiple times

Flexibility of Plugins:

- Display plugins have an API attached to the data collected
- Flexibility in data being analyzed and what analyses are displayed

Project Structure

- We'll use the *ProjectLoader* library to load the Data and Display plugins
- All plugins should implement their respective Interface
- List of plugins that we want to import go in META-INF/services file
- /framework contains core & gui directories

```
src{
    main
    {
        java
        {
            -Main.java
            framework
            {
                core
                {
                    -FrameworkImpl.java
                    -DataPlugin.java
                    -DisplayPlugin.java
                    -DataPoint.java
                    -DisplayDataStructure.java
                    -Column.java
                    -Config.java
                }
                gui
                {
                    -FrameworkGUI.java
                }
            }
            plugin
            {
                /*All Plugins go here*/
            }
        }
        resources
        {
            META-INF
            {
                services
                {
                    -framework.core.DataPlugin
                    -framework.core.DisplayPlugin
                    /*Each containing list of curently
                    possessed plugins*/
                }
            }
        }
    }
}
```

DisplayPlugin/DataPlugin Interfaces

interface DataPlugin:

extractData(): List<DataPoint>

expected but not required to provide GUI asking user for supplementary info (file location, columns)

extracts a list of data entries (DataPoint type) from file, webpage etc.

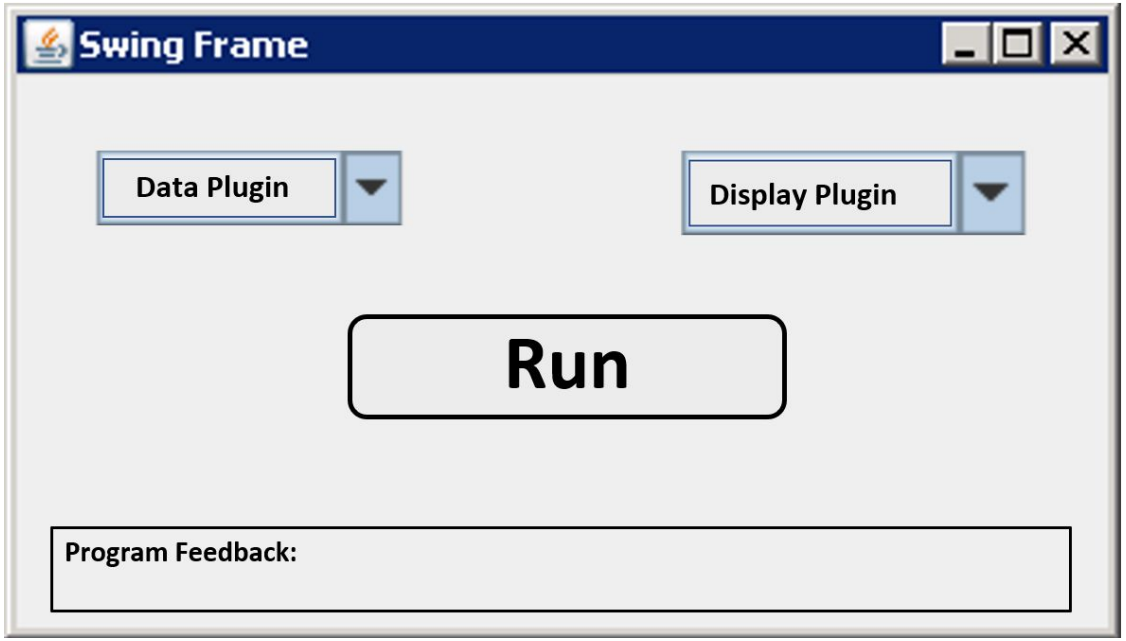
interface DisplayPlugin:

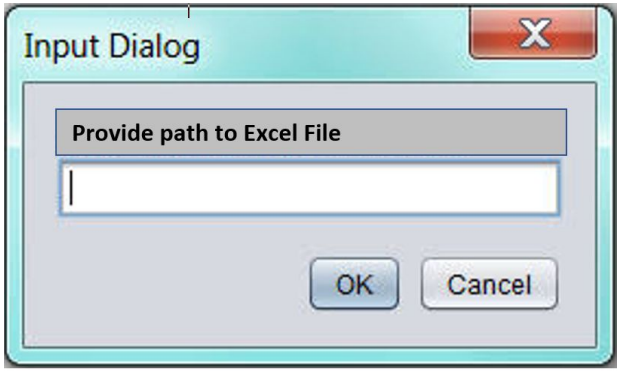
visualizeData(DisplayDataStructure struct): void

expected but not required to provide GUI asking user for supplementary info if needed

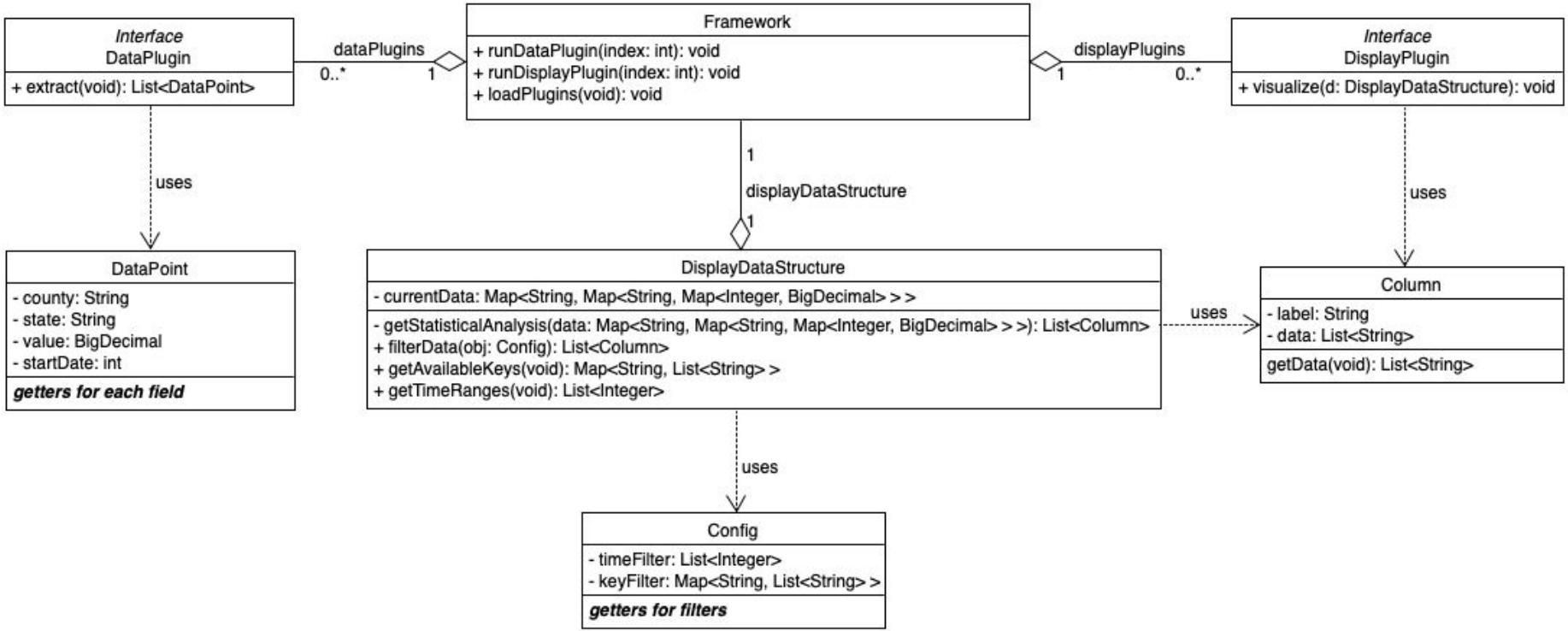
visualizes data based on plugin implementation

Mock Design


Swing Frame
Data Plugin
Display Plugin
Run
Program Feedback:


Input Dialog
Provide path to Excel File
OK
Cancel

UML Diagram


Interface
DataPlugin
+ extract(void): List<DataPoint>
dataPlugins
0..*
1
Framework
+ runDataPlugin(index: int): void
+ runDisplayPlugin(index: int): void
+ loadPlugins(void): void
displayPlugins
1
0..*
Interface
DisplayPlugin
+ visualize(d: DisplayDataStructure): void
uses
1
displayDataStructure
1
uses
DataPoint
- county: String
- state: String
- value: BigDecimal
- startDate: int
getters for each field
DisplayDataStructure
- currentData: Map<String, Map<String, Map<Integer, BigDecimal> > >
- getStatisticalAnalysis(data: Map<String, Map<String, Map<Integer, BigDecimal> > >): List<Column>
+ filterData(obj: Config): List<Column>
+ getAvailableKeys(void): Map<String, List<String> >
+ getTimeRanges(void): List<Integer>
uses
Column
- label: String
- data: List<String>
getData(void): List<String>
uses
Config
- timeFilter: List<Integer>
- keyFilter: Map<String, List<String> >
getters for filters