Data Visualization of Covid

•••

Team LLC
Cuiting Li, Laura (Kai Sze) Luk, Yumin Chen

Domain - Covid 19 Data Analysis

Support various Data Plugin

- CSV File
- JSON File
- Web API
- ......

Support various Visualize Plugin

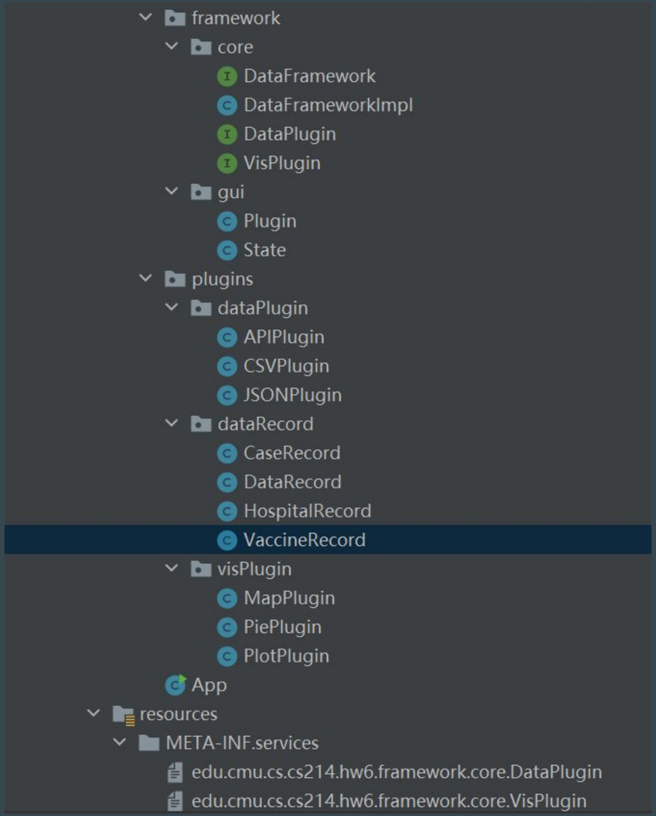
- Pie Chart
- PlotChart
- HeatMap
- ....

Domain - Covid 19 Data Analysis

Framework’s job

- combining data
- analyzing maximum values (over time/over states)
- minimum values (over time/over states)
- average by each month or state
- sum by each month or state
- cross-over analysis on different data sources.

Project Structure



```
framework
  core
    DataFramework
    DataFrameworkImpl
    DataPlugin
    VisPlugin
  gui
    Plugin
    State
plugins
  dataPlugin
    APIPlugin
    CSVPlugin
    JSONPlugin
  dataRecord
    CaseRecord
    DataRecord
    HospitalRecord
    VaccineRecord
  visPlugin
    MapPlugin
    PiePlugin
    PlotPlugin
App
resources
  META-INF.services
    edu.cmu.cs.cs214.hw6.framework.core.DataPlugin
    edu.cmu.cs.cs214.hw6.framework.core.VisPlugin
```

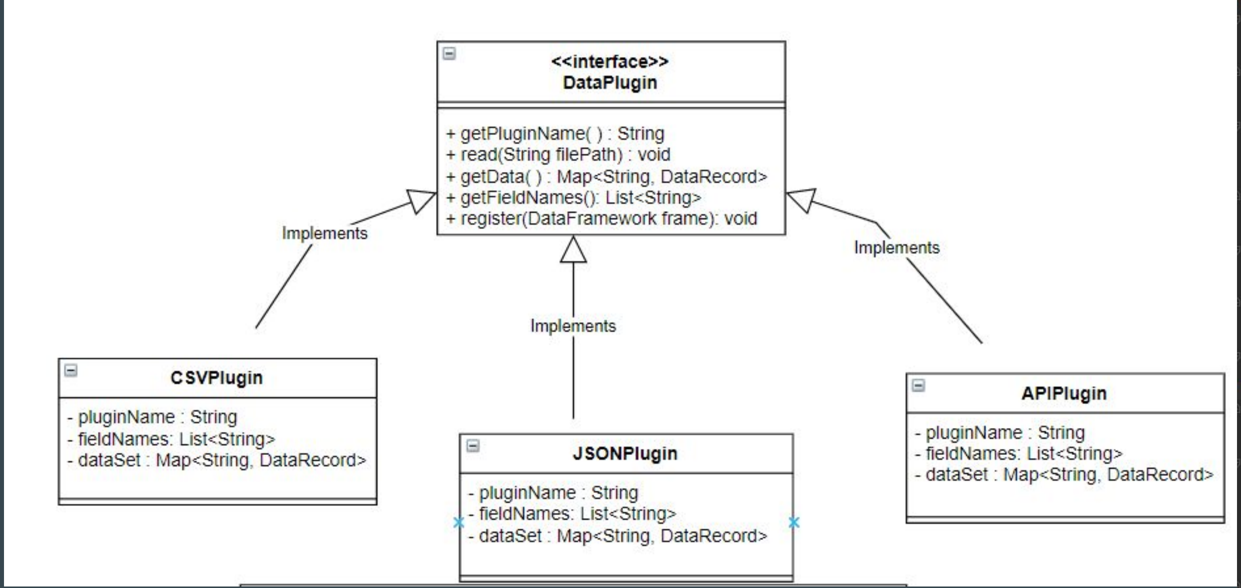
Framework

- **core**
  -DataFramework(interface)
  -DataFrameworkImpl
  -DataPlugin(interface)
  -VisPlugin(interface)
- **gui**
  -Plugin
  -State

Plugins

- **dataPlugin**
  -APIPlugin
  -CSVPlugin
  -JSONPlugin
- **visPlugin**
  -MapPlugin
  -PiePlugin
  -PlotPlugin
- **dataRecord**
  -DataRecord
  -CaseRecord
  -HospitalRecord
  -VaccineRecord

Plugin Interface -- Data Plugin



<<interface>>
DataPlugin

+ getPluginName( ) : String
+ read(String filePath) : void
+ getData( ) : Map<String, DataRecord>
+ getFieldNames(): List<String>
+ register(DataFramework frame): void

CSVPlugin

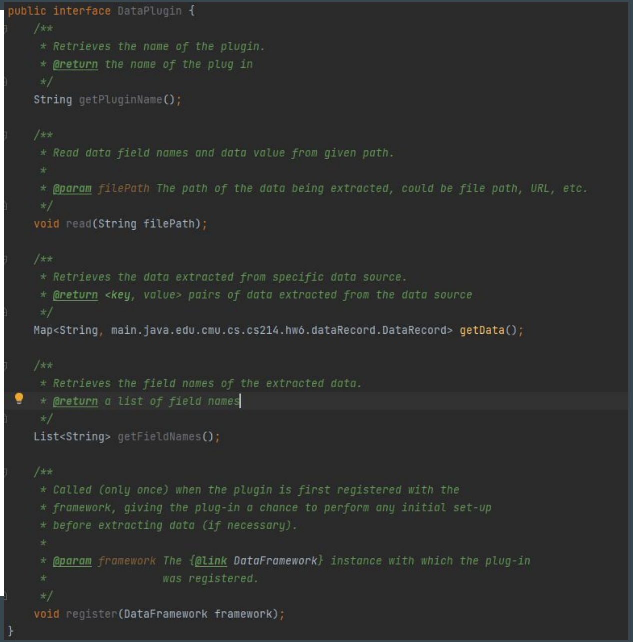
- pluginName : String
- fieldNames: List<String>
- dataSet : Map<String, DataRecord>

JSONPlugin

- pluginName : String
- fieldNames: List<String>
- dataSet : Map<String, DataRecord>

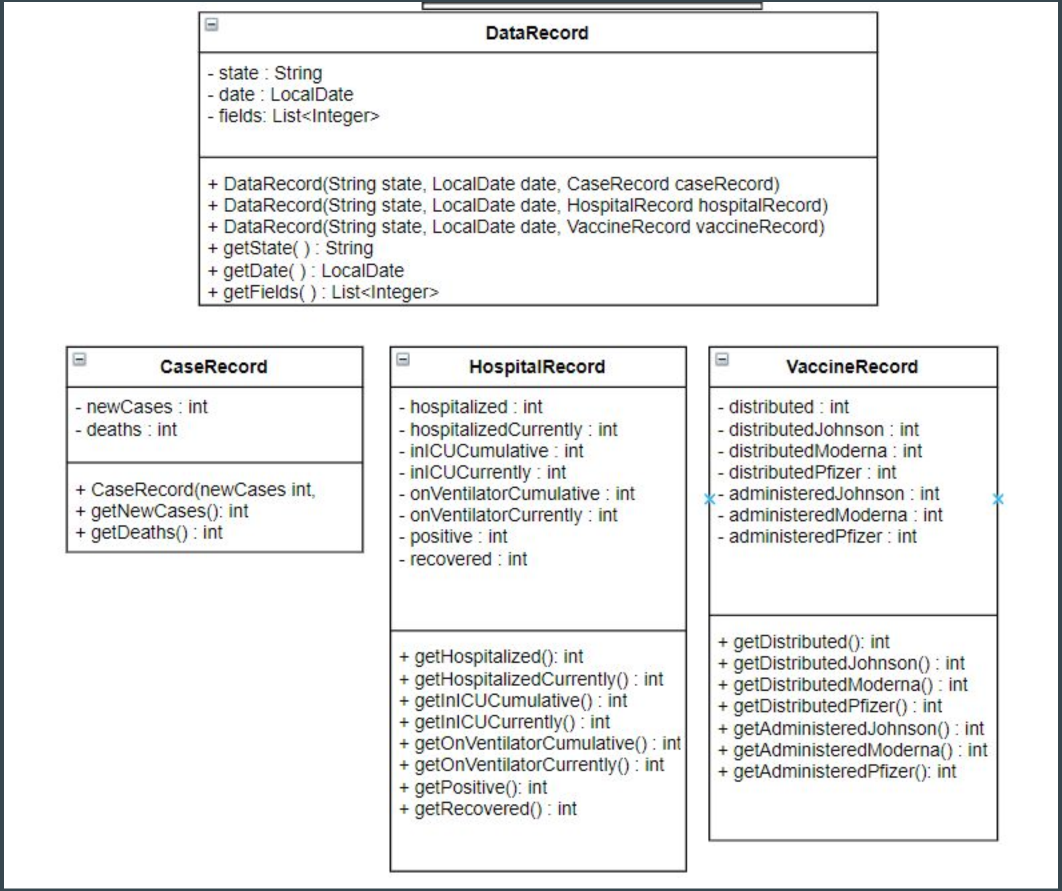
APIPlugin

- pluginName : String
- fieldNames: List<String>
- dataSet : Map<String, DataRecord>



```java
public interface DataPlugin {
    /**
     * Retrieves the name of the plugin.
     * @return the name of the plug in
     */
    String getPluginName();

    /**
     * Read data field names and data value from given path.
     *
     * @param filePath The path of the data being extracted, could be file path, URL, etc.
     */
    void read(String filePath);

    /**
     * Retrieves the data extracted from specific data source.
     * @return <key, value> pairs of data extracted from the data source
     */
    Map<String, main.java.edu.cmu.cs.cs214.hw6.dataRecord.DataRecord> getData();

    /**
     * Retrieves the field names of the extracted data.
     * @return a list of field names
     */
    List<String> getFieldNames();

    /**
     * Called (only once) when the plugin is first registered with the
     * framework, giving the plug-in a chance to perform any initial set-up
     * before extracting data (if necessary).
     *
     * @param framework The {@link DataFramework} instance with which the plug-in
     *                  was registered.
     */
    void register(DataFramework framework);
}
```

Plugin Interface -- Data Class



DataRecord

- state : String
- date : LocalDate
- fields: List<Integer>

+ DataRecord(String state, LocalDate date, CaseRecord caseRecord)
+ DataRecord(String state, LocalDate date, HospitalRecord hospitalRecord)
+ DataRecord(String state, LocalDate date, VaccineRecord vaccineRecord)
+ getState( ) : String
+ getDate( ) : LocalDate
+ getFields( ) : List<Integer>

CaseRecord

- newCases : int
- deaths : int

+ CaseRecord(newCases int,
+ getNewCases(): int
+ getDeaths() : int

HospitalRecord

- hospitalized : int
- hospitalizedCurrently : int
- inICUCumulative : int
- inICUCurrently : int
- onVentilatorCumulative : int
- onVentilatorCurrently : int
- positive : int
- recovered : int

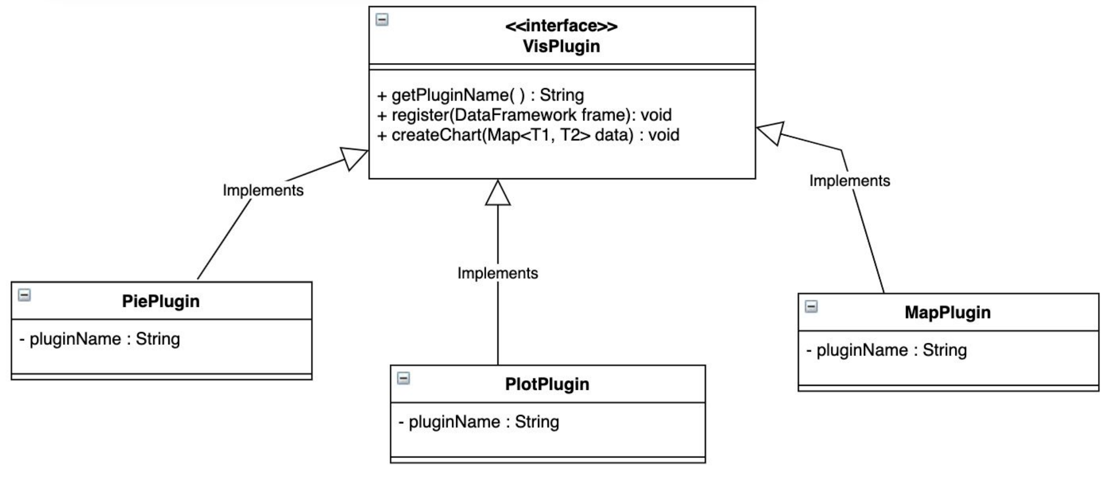
+ getHospitalized(): int
+ getHospitalizedCurrently() : int
+ getInICUCumulative() : int
+ getInICUCurrently() : int
+ getOnVentilatorCumulative() : int
+ getOnVentilatorCurrently() : int
+ getPositive(): int
+ getRecovered() : int

VaccineRecord

- distributed : int
- distributedJohnson : int
- distributedModerna : int
- distributedPfizer : int
- administeredJohnson : int
- administeredModerna : int
- administeredPfizer : int

+ getDistributed(): int
+ getDistributedJohnson() : int
+ getDistributedModerna() : int
+ getDistributedPfizer() : int
+ getAdministeredJohnson() : int
+ getAdministeredModerna() : int
+ getAdministeredPfizer(): int

Plugin Interface -- Visualization Plugin



<<interface>>
VisPlugin

+ getPluginName( ) : String
+ register(DataFramework frame): void
+ createChart(Map<T1, T2> data) : void

Implements

PiePlugin

- pluginName : String

Implements

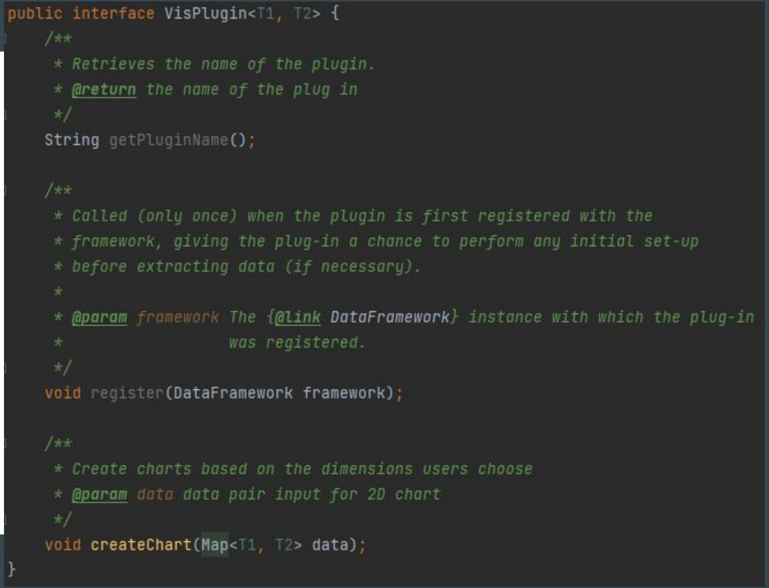
PlotPlugin

- pluginName : String

Implements

MapPlugin

- pluginName : String



```
public interface VisPlugin<T1, T2> {
    /**
     * Retrieves the name of the plugin.
     * @return the name of the plug in
     */
    String getPluginName();

    /**
     * Called (only once) when the plugin is first registered with the
     * framework, giving the plug-in a chance to perform any initial set-up
     * before extracting data (if necessary).
     *
     * @param framework The {@link DataFramework} instance with which the plug-in
     *                  was registered.
     */
    void register(DataFramework framework);

    /**
     * Create charts based on the dimensions users choose
     * @param data data pair input for 2D chart
     */
    void createChart(Map<T1, T2> data);
}
```

Generality & Specificity

Generality:

- data plugin interface
- visualization plugin interface
- abstract class– *DataRecord*

framework interface includes common methods

- *getMin(...)*
- *getMax(..)*
- *getAverage(...)*
- can be generalized to a different framework implementation.

Specificity:

- data plugin implementation has its own (different) method of extracting data
- each visualization plugin implementation has its own (different) method of initializing and setting up data formats to be plotted.