Stock Prices Analytic Framework

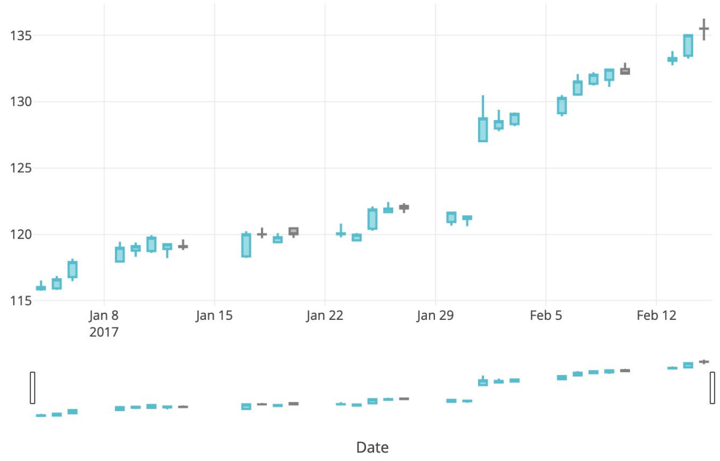
Deadline-Fighters

- Jason Law (chikinl)
- Xinna Guo (xinnag)

Idea & Data Structure

- Stock prices
- Data retrieval, processing and visualization

```
public class StockQuote {
    String symbol;
    LocalDate date;
    // nullable as some data sources
    // may not come with particular prices
    Double open;
    Double close;
    Double high;
    Double low;
}
```



```
class AggregatedStockQuotes {
    String symbol;
    List<StockQuote> daily;
    List<StockQuote> weekly;
    List<StockQuote> monthly;
    List<StockQuote> quarterly;
    List<StockQuote> yearly;
}
```

Project Structure

- src
  - main
    - java/deadlinefighters/analyticsframework
      - framework
        - core
          - Framework
          - FrameworkImpl
          - DataPlugin
          - VisualizationPlugin
        - gui
      - plugin
        - data
          - LocalDataPlugin
          - WebDataPlugin
          - DatabaseDataPlugin
        - visualization
          - CandlestickVisualizationPlugin
          - LineVisualizationPlugin
          - BubbleVisualizationPlugin
          - BarVisualizationPlugin
    - resources/META-INT/services
      - deadlinefighters.analyticsframework.framework.core.DataPlugin
      - deadlinefighters.analyticsframework.framework.core.VisualizationPlugin

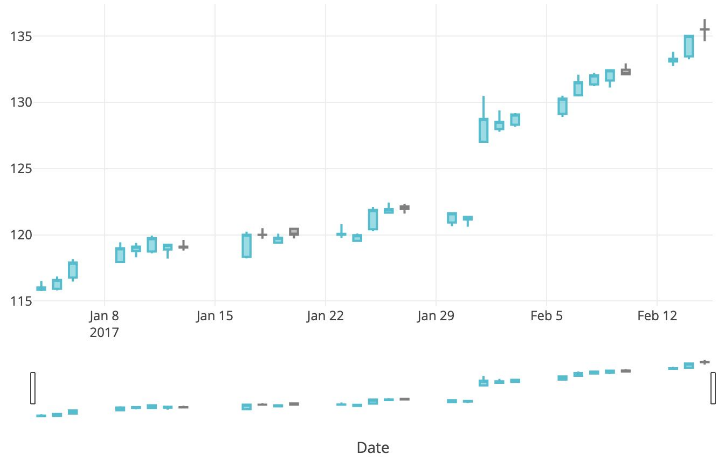
Plugins

Data:

- Web APIs. Free time series stock API like Alpha vantage
- Local file (e.g. csv)
- Database

Visualization

- Candlestick chart
- Line chart (stocks comparison)
- Bubble chart (changes to different stocks)
- Bar chart



Data Plugin Interface

```java
class StockQuotesResult {
    List<StockQuote> stockQuotes;
    boolean hasError;
    String errorMessage;
}
```

```java
public interface DataPlugin {

    /**
     * Called (only once) when the plugin is first registered with the
     * framework, giving the plugin a chance to perform any set-up during
     * plugin registration.
     *
     * @param framework The {@link Framework} instance with which the plugin
     *                  was registered.
     */
    void onRegister(Framework framework);

    /**
     * Establish a connection with the data source, this is called by the
     * framework before getStockQuotes is called.
     *
     * @param arg    argument required to initialize the data plugin
     *               (e.g. local file path, web url or database url)
     * @throws IllegalArgumentException when fail to parse the arg
     *          or IOException when fail to establish a connection
     */
    void openConnection(String arg) throws Exception;

    /**
     * Read from the data source with only selected symbols
     * and parse the data into stock object.
     *
     * If exception was thrown during the execution, the framework should catch
     * ExecutionException and handle it.
     *
     * If symbols are missing in the data source, error message will be written
     * in the result.
     *
     * @param symbols   a list of selected symbols
     * @return a promise of stock quotes result
     */
    CompletableFuture<StockQuotesResult> getStockQuotes(List<String> symbols);

    /**
     * Close the connection with the data source, this is called by the
     * framework after getStockQuotes is called.
     *
     * @throws  IOException when fail to close a connection
     */
    void closeConnection() throws IOException;
}
```

Visualization Plugin Interface

```java
public interface VisualizationPlugin {

    /**
     * Called (only once) when the plugin is first registered with the
     * framework, giving the plugin a chance to perform any set-up during
     * plugin registration
     *
     * @param framework The {@link Framework} instance with which the plugin
     *                  was registered.
     */
    void onRegister(Framework framework);
```

```java
    /**
     * Requirement of the plugin on the data
     *
     * @param symbolCount   number of unique symbols in the data
     * @return boolean      indicating whether the number of symbols is
     *                      supported by the plugin
     */
    boolean isSupported(int symbolCount);

    /**
     * Render the corresponding char using Plotly
     *
     * @param data   a list of aggregated stock quotes
     * @return       HTML that will be embedded in the web-based GUI
     */
    String render(List<AggregatedStockQuotes> data);

}
```