Homework #5: Data Visualization Framework

In this assignment you will work as a team to design and implement an extensible data visualization and analysis framework, consisting of an interactive GUI tool and underlying interfaces and implementations.

Your framework will support two types of plugins:

1. *Data plugins* that extract data from some source and make it available for processing. Possible sources include (but are not limited to) local files, web pages, web APIs, mathematical formulae, or sensors.
2. *Display plugins* that visualize data provided by the data plugins. Possible visualizations include (but are not limited to) pie charts, bar charts, histograms, statistical abstracts, X-Y plots, scatter plots, bubble charts or choropleths. Some display plugins may display data from multiple data sets concurrently.

When you are done, you will have a framework to play with data from a field that interests you, from politics to Pokémon, from biodiversity to baseball. Be creative when you design, implement, and use your framework—surprise us! This homework enables a far broader variety of solutions than the previous ones.

The learning goals for this assignment are:

- Design and implement a black-box framework with a common plugin interface for data from a wide variety of sources and a common plugin interface to perform a wide variety of data visualizations.
- Perform domain analysis to determine appropriate data sources, visualizations, and data representations for your chosen domain.
- Demonstrate effective testing techniques for interactive software components.
- Coordinate software design and development within a team and between teams.
- Demonstrate proficiency with others' code by using open-source, third-party libraries and developing plugins for another team's framework.

Milestones and deadlines

This assignment contains a team sign-up deadline and three milestones:

- Sign up your team by Tuesday, November 1, 11:59 pm. [1]
- Milestone A: Design of your framework, due Tuesday, November 8, 12:00 pm in class.

[1] See https://piazza.com/class/is6f9y2tdia44d?cid=632 for more information

- Milestone B: Framework and sample plugins implementation, due Tuesday, November 15, 11:59 pm. Submit your work by Wednesday, November 16, 9:00 am to be considered as a "Best Framework" for Milestone C. You will also present and demo your framework on Wednesday, November 16.
- Milestone C: Plugins for another framework, due Tuesday, November 22, 11:59 pm.

For this assignment you must work in teams of 2-3 students, using a shared Git repository to coordinate and turn in your work. You must form your team and sign up for a presentation time by Tuesday, November 1, 11:59 pm using the web form and instructions posted to Piazza. Presentation slots are limited and available on a first-come-first-served basis, so please sign up as early as possible. You may use the Piazza "Search for Teammates" feature to find teammates.

Late days

Each team may use up to a total of 2 team late days for this assignment in Milestones B and C. You may **not** use late days for Milestone A; work submitted for Milestone A after Tuesday, November 8, 12:00 pm will be assessed a 1% penalty every 5 minutes, as per the usual course late penalty. If you submit Milestone B after Wednesday, November 16, 9:00 am your team cannot be considered as a "Best Framework" to be selected for plugins for Milestone C, so try to use few late days for Milestone B if possible.

Your team's late days for this assignment are independent of your team members' late days for other assignments in this course. In other words, the number of late days you have used on previous homeworks does not determine the number of late days you can use for this assignment. Likewise, the number of late days your team uses for this assignment will not affect the number of late days you can use for Homework 6.

For the purposes of this late policy, Thanksgiving does not count as a day. If your team turns in Milestone C two days late you may turn it in as late as Friday, November 25, 11:59 pm.

Evaluation

We will grade your work approximately as follows:

- Milestone A (60 points)
- Milestone B (160 points + up to 10 points extra credit)
- Milestone C (80 points)

Milestone A: Design your framework (due in class Tuesday, November 8, 12:00 pm; you may not use late days)

For the first milestone you will design your framework. At minimum, your framework must achieve the following design goals:

- Your framework must support data plugins and display plugins for a domain of your choosing. A plugin developer should be able to implement additional data sources and visualizations with only a small amount of work. Implementing a new plugin must not require changes to the core framework code.
- Your plugin interfaces must be general and interchangeable. Plugin interfaces should hide the details of their implementations and permit visualizations to be applied to data sets regardless of their source. If necessary, you may create multiple data source interfaces to expose appropriate details for incompatible data types.
- Your GUI must allow the user to create multiple data sets from multiple data sources and to create multiple visualizations of those data sets.
- Your framework must support a mechanism to initialize and parameterize data sets and visualizations. For example, a data set that draws its data from a CSV file would need to know the path of the file and some description of the relevant columns. A data set that draws its data from a web page would need to know the web page's URL.

Your team must *turn in a printed copy of your documents by Tuesday, November 8, 12:00 pm in class* and submit four files to the `hw5a` directory in your team's shared Git repository. We describe each of these documents below.

A high-level description of your framework

In `hw5a/description.pdf` write a short (1 page) description of your framework. Describe the intended domain(s), data model, tool capabilities, plugin interfaces, and any significant constraints the framework imposes on its plugins.

Draft APIs for your framework and plugins

In `hw5a/design.pdf` provide a rough draft of your most important interfaces. For each method provide its signature and a short (1-2 line) description. This draft need not be of the quality required for a final API specification, but should allow a reader to determine whether the interfaces can represent a data source and what would be required to implement each interface.

An interaction model for your framework

In `hw5a/interaction_model.pdf` create an interaction model (as a UML sequence or communication diagram) for any interesting interactions in your framework. We recommend that you include at least a lifecycle diagram showing the required protocol for the life of a plugin, as well as an interaction diagram showing how the framework interacts with data sources and visualization plugins when new data is available (if your framework supports dynamic data sets).

A project planning document

In `hw5a/plan.pdf` create a short document describing how you will divide work amongst your team members and any internal deadlines you will meet to ensure high quality products. Your team responsibilities may be overlapping, but you should make one person principally responsible for each artifact of your project.

Hints:

Designing a framework can be challenging. We recommend the following steps to get started:

- Select one or more domains that interest you, and several sources of data from each domain. Consider the characteristics of those sources' data sets and how you would like to visualize the data sets on your screen. Use web search to help find data sets.
- Determine potential extension points for plugins, and commonalities implemented within the framework. Decide what features your framework will provide and which decisions should be left to the plugins.
- Consider the lifecycle of the objects in your framework, including what methods the framework will call when the plugins are registered and initialized and any callback mechanisms you must define to support dynamic plugins.
- Draft code to support one or more visualizations for one or more data sources without using a framework. Even though you won't turn in this code, it will greatly help you understand the abstractions your eventual framework will support.
- Think about what data types your framework should support. At a minimum, the data model must be sufficient to support the visualizations that you hand-coded. Your goal is to support all reasonable data sources within your intended domain. Aim for generality, cleanliness, and type-safety.

Milestone B: Framework and sample plugin implementation (due Tuesday, November 15, 11:59 pm)

Implement, document, and test your framework, and write sample plugins for it. A two-person team must implement two data plugins and two visualization plugins. A three-person team must implement three data plugins and three visualization plugins.

You have much freedom to be creative in your framework design and implementation. Your work, however, must satisfy the following requirements:

- Your tool must initialize the user interface and allow the user to select and configure data sources and visualizations.
- Each visualization plugin should work with a wide variety of data plugins, and vice versa. It's OK if a given visualization plugin is incompatible with a given data plugin in some cases; for example, a bubble chart requires 3-dimensional data and is incompatible with a 1- or 2-dimensional data set, but still works for a wide variety of data plugins.
- Your framework must support and display multiple data and visualization plugins concurrently, including visualizations of the same and visualizations of different data sets.
- There should be no direct communication among multiple plugins; all communication is orchestrated by the framework. Visualization plugins should depend only on the framework and should not interact directly with other plugins or initiate direct queries to a data source.
- One of your data plugins must use a third-party library or web API to gather and/or parse the data, and one of your visualization plugins must use a third-party library. We will suggest some libraries and Web APIs on Piazza, but you are not limited to these libraries.
- You must test your framework implementation with JUnit tests, achieving reasonably good coverage and demonstrating good testing principles. You must use test stubs, rather than your actual plugin implementations, to interactively test your framework implementation. As we said in class, a test stub (or mock) is just an implementation whose purpose is to simulate the behavior of some software component (such as a plugin) for the purpose of testing another software component (such as your framework). We recommend, but do not require, that you test your plugin implementations.
- You must design and thoroughly document your framework so that another team could potentially use your framework. Apply principles for writing understandable software: descriptive class and method names, information hiding and encapsulation,

design patterns, etc. Keep the conceptual weight of your framework low so that it is easy for another team to learn and use. Provide clear documentation so that others could easily write plugins for your framework, including Javadocs for all public and protected methods and a short `hw5b/README.pdf` file explaining the steps required to implement plugins. Your sample plugins should be a useful example for others; example code should be exemplary.

- Your code should be runnable using a Gradle `run` task.[2] Note that we have not created an Eclipse project or Gradle build file for you. You must create a project with appropriate package names and an appropriate file organization, and a Gradle build file. In the `hw5b` project in your team's shared Git repository, turn in your framework implementation in an `edu.cmu.cs.cs214.hw5.framework` package and turn in your sample plugin(s) in an `edu.cmu.cs.cs214.hw5.plugin` package. We will run FindBugs on your framework; you should too.

If you feel like a challenge, there are many ways you can exceed these minimal requirements. Ideally, your framework should allow new plugins to be added at run time using Java reflection, but this is not required. You may (but are not required to) choose one or more network-based data sources, from a web API or scraped from some web pages (possibly using tools such as Jaunt or jsoup). You could also provide persistence of your data sources and visualizations, allowing users to continue a visualization when they restart the tool—again, this not required. If you're interested in pursuing this goal, you might take a look at the `java.util.prefs` API.

Finally, remember that your team must submit your framework implementation by Wednesday, November 16, 9:00 am to be considered as a framework to be plugged into for Milestone C. If your team's framework is selected, then you will receive 10 points extra credit for Milestone B and you will not need to write any plugins for Milestone C.

A presentation and demo of your work

In `hw5b/presentation.pdf`, prepare a short, 10-minute presentation describing your framework, the key design decisions you've made, the domain(s) it can analyze, and a live demo of your framework. You will not have time to describe your overall design in detail, but highlight your main decisions and the plugin interfaces, as well as the interactions between the plugins and the framework. You should use **5 or fewer slides** for your presentation.

Your primary audience is your peers and your goal is to convince them to write a plugin for your framework. Make them drool over your amazingly thorough documentation.

Your entire team must give the presentation at the time you signed up for, and all team members should actively participate in the presentation.

[2] See http://gradle.org/docs/current/userguide/application_plugin.html for more information.

Milestone C: plugin implementations for another framework (due Tuesday, November 22, 11:59 pm)

Soon after the Milestone B deadline we will select several framework implementations for other teams to plug into for Milestone C. We will select from frameworks finished by Wednesday, November 16, 9:00 am. If we select your framework, your team must support your framework for others. If we do not select your framework, you must implement plugins for one of the selected frameworks.

Supporting a selected framework

If your framework is selected you must provide technical support for the teams building plugins for it. This includes answering their questions promptly on Piazza, fixing bugs in your framework (if necessary), and addressing misunderstandings teams have about your framework. Overall, your goal is to keep the other teams happy and ease their task of implementing plugins for your framework.

As you support your framework you should keep all communication between you and other teams public, using Piazza for all technical support. This is because, (1) you will be graded by the quality of technical support you provide, and (2) all your development teams can benefit from seeing the problems addressed by others, as well as how they were eventually resolved. You should not provide support beyond what is legal or socially acceptable, or that would violate the course cheating policy. If in doubt, ask the course staff.

At the conclusion of this assignment, you must also submit a short report (at most 1 page) of your experience providing support for your framework. You should also try to discover something interesting about your domain by visualizing data using your framework, and write a paragraph describing your discovery. Please submit this report in `hw5c/experience_report.pdf` before the Milestone C late due date.

Writing plugins for another team's framework

If your framework was not selected, your team must develop plugins for another team's framework. Two-person teams must develop two data plugins and two visualization plugins. Three-person teams must develop three data plugins and three visualization plugins for the other team's framework.

At least one of your visualization plugins must use an extra third-party library; you may use any library you want to aid in data analysis or visualization. Your plugins for Milestone C cannot use the same date source or visualization as the sample plugins for the framework you are plugging into, but you may otherwise base your Milestone C plugins on your team's

own plugins from Milestone B. You are permitted to use the same visualization library that you used in Milestone B.

You will access a selected framework and its documentation by cloning the public Git repository of another team and referencing the cloned repository as a project dependency in your IDE; we will post some instructions to Piazza for how to do this in the coming weeks. We will also designate a Piazza thread for each selected framework for you to communicate with the members of the selected framework teams. Please also note that you may not change any framework code directly; your plugins must work for the official version of the framework you are plugging into. If you think you have found a bug in the framework, report the problem on the framework team's support thread for them to address.

Please submit your plugins to the `edu.cmu.cs.cs214.hw5.plugin` package in the `hw5c` project located in your team's shared Git repository.

At the conclusion of this assignment, you must also provide feedback (at most 1 page) on the quality of technical support from the framework-providing team. You should attempt to discover something interesting about some domain by visualizing data using your own framework or the framework you used in Part C, and should write a paragraph describing your discovery. Please submit this report in `hw5c/experience_report.pdf` when you are done.