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, maps and geodata, 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 and presentation time by Thursday, October 31, at 11:59 pm.[1]
- Milestone A: You will design your framework and present your design on Wednesday, November 6 during recitation at the time slot you signed up for. You will also

[1] See Piazza for more information.

need to submit pdf files of your presentation and planning document to GitHub by Wednesday, November 6, at 9:00 am.

- Milestone B: Your framework and sample plugins implementation, along with design documents and documentation is due Tuesday, November 12, at 11:59 pm. Submit your work by Wednesday, November 13, at 9:00 am to be considered for "Best Framework" for Milestone C, see below.
- Milestone C: Write plugins for one of the "Best Frameworks," due Tuesday, November 19, 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 Thursday, October 31, at 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 your presentation.

If you submit Milestone B after Wednesday, November 13, at 9:00 am, your team cannot be considered as a "Best Framework" for Milestone C. 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.

Milestone A: Design your framework (Wednesday, November 6, at 9:00 am; you may not use late days)

For this milestone you will design your framework (identifying its common and variable parts and producing a preliminary API), and also plan how you will work as a team.

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.

You will present your design to your classmates in recitation on Wednesday, November 6. Please read the Milestone B instructions before preparing your presentation.

In at most 10 minutes total, your presentation should:

- Describe your framework domain, with examples of possible plugins your framework could support.
- Describe your decisions about the generality and specificity of your framework (i.e., domain engineering): your key abstractions, the reusable functionality your framework provides, and the potential flexibility of plugins.
- Describe your overall project structure: the organization of the framework and plugins into packages or projects, the location of plugin interfaces and key data structures, and how plugins are loaded.
- Describe your plugin interfaces, including key methods and the data structures exchanged between plugins and the framework.

Create **6 or fewer slides** for your presentation and upload them as `presentation.pdf` to your team's repository by Wednesday, November 6, at 9:00 am. You should include UML diagrams and/or code as necessary to clearly illustrate your framework design and how it could be used. Your primary audience is your peers and your goal is to illustrate how you achieve reuse in a domain. Your entire team must give the presentation at your presentation time, and **all team members should actively participate** in the presentation.

Finally, in `plan.pdf` by Wednesday, November 6, at 9:00 am, describe how you will divide work among your team members, and describe any internal deadlines you will meet to ensure a high quality product. Your team responsibilities may overlap, but one person should be principally responsible for each artifact of your project.

Milestone B: Framework and sample plugin implementation (due Tuesday, November 12, at 11:59 pm)

Implement, document, and test your framework, and write sample plugins to demonstrate how the framework can be extended and used. 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.
- 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 configure Gradle to build your plugins with those dependencies on Travis CI.
- You must test your framework implementation with JUnit tests, achieving reasonably good coverage and demonstrating good testing principles. You should use test stubs, rather than your actual plugin implementations. 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 do not need to automate tests for GUIs.
- You must design and thoroughly document your framework so that another team could potentially use your framework. Documentation of the plugin interfaces is

especially critical, as is the documentation of how to load additional plugins without modifying the 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. Your sample plugins should be a useful example for others; example code should be exemplary. Specifically:

  - Documentation should include clear Javadocs for plugin APIs and relevant data structures.
  - Create a `README.md/pdf` file that contains an overview of your framework domain, an overview of your most important APIs, instructions for how to implement plugins, and instructions for how to run the framework with new plugins.

- Structure your code so that plugins can be loaded from different projects without modifying the framework. We strongly recommend the `java.util.ServiceLoader`, as you saw in Recitation 9.
- Organize your code into projects and packages and set up all necessary build scripts. We recommend the package names `edu.cmu.cs.cs214.hw5.framework` and `edu.cmu.cs.cs214.hw5.plugin`. You should develop your plugins in a separate project (i.e., different directory with its own Gradle file) to ensure you can add functionality without modifying the framework.
- Your code must be runnable using a Gradle `run` task.[2] We will run FindBugs on your framework; you should too.
- Avoid committing passwords or other credentials to your GitHub repository.

If you feel like a challenge, there are many ways you can exceed these minimal requirements. You may (but are not required to) choose network-based data sources, such as a web API or scraping from 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 13, at 9:00 am to be considered as a framework to be plugged into for Milestone C. Please mark your final commit with a clear commit message. If your team's framework is selected, then you will receive 10 points extra credit and you will not need to write any plugins for Milestone C.

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

Milestone C: plugin implementations for another framework (due Tuesday, November 19, 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 13, at 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 responding to questions promptly on GitHub, 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.

Please note that none of your framework code can be changed directly by the developing teams. This has two implications: (1) your framework must allow dynamic loading of plugins (outlined in the previous section), and (2) you must fix bugs reported by other teams as soon as possible and update your framework accordingly.

As you support your framework you should keep all communication between you and other teams public, using GitHub for all technical support. This is because (1) you will be graded on the quality of technical support you provide, and (2) all development teams can benefit from seeing the problems raised by others, as well as their eventual resolution. 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 `experience_report.pdf` before the Milestone C *late* due date (Thursday, November 21, at 11:59 pm).

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.

At least one of your 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 have access to the GitHub repositories of the selected frameworks and their documentation. The teams of selected frameworks will provide instructions of how to add dependencies on the framework for your own projects. You will use *GitHub issues* to communicate with the members of the selected framework teams, and you may submit pull requests on GitHub. 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 or submit a patch (e.g. a pull request) to the framework team's for them to address.

Please submit your plugins in one or multiple projects in the `hw5c` folder of 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 as `experience_report.pdf` inside the `hw5c` directory.

Evaluation

Overall, this homework is worth 290 points: Milestone A is worth 90 points, Milestone B is worth 125 points, and Milestone C is worth 75 points. You will receive 10 extra points if your framework is selected for Milestone C.

For Milestone A, we will evaluate the quality of the presentation itself, the quality of your framework design, and the ability of your framework to meet the general assignment specification.

For Milestone B, we will evaluate both implementation and documentation. Successful solutions must include a working framework and example plugins, the ability to add plugins without modifying the framework, Gradle build files, use of external libraries, and clear documentation for how to extend the framework.

For Milestone C, we will evaluate the quality of your additional plugins or provided support, as well as the quality of your experience report.

This homework admits a wide range of excellent successful solutions. When in doubt about the assignment's requirements, use your best judgment and document your assumptions.