Homework 5: Team 1

Read Me: Data Explorer Framework

Vanessa Hwang | Sean D Kim | Tiffany Lee

KNOWN ISSUE:

Our framework uses the JavaFX framework, which is notorious for occasional glitches in buttons and frozen screen when rendering HTML page. If encounter issues like that, simply right click on the window and select "Reload Page", which would cause the window to update and most of the time solves the problem.

I. Purpose of the Framework

The data explorer framework aims to provide a structure for clients to implement exploration-based visualization. An example would be a music exploration web app (www.panther.audio), where user is able to start on a favorite artist and discover artist with similar style. The idea can be generalized to different creative work or products such as paintings, Amazon products, books.

A. Question framework helps to answer

a. I like this artist, which artists have similar music style has his that I can look into?

b. *Pride and Prejudice* is my favorite book, I wonder if there is similar book that I would equally enjoy?

c. I am really bored of these movies, even though I like them a lot. Are there movies of similar genre and rating?

d. This article about computer science is really interesting, what are similar

articles that expand on this?

B. Suggested Data Domain

a. Any domain that involves invention and creative arts (Music, Literature, Art, Brands, Products, Shopping Sites)

II. Framework Mechanics

The framework uses the JavaFX library that connects a web based user-facing interface with the Java backend. The data plugin is directly linked to the backend, while the display plugin controls how the front end is rendered. For more detailed domain model, please refer to object.pdf in the design document directory.

A. Data Plugin

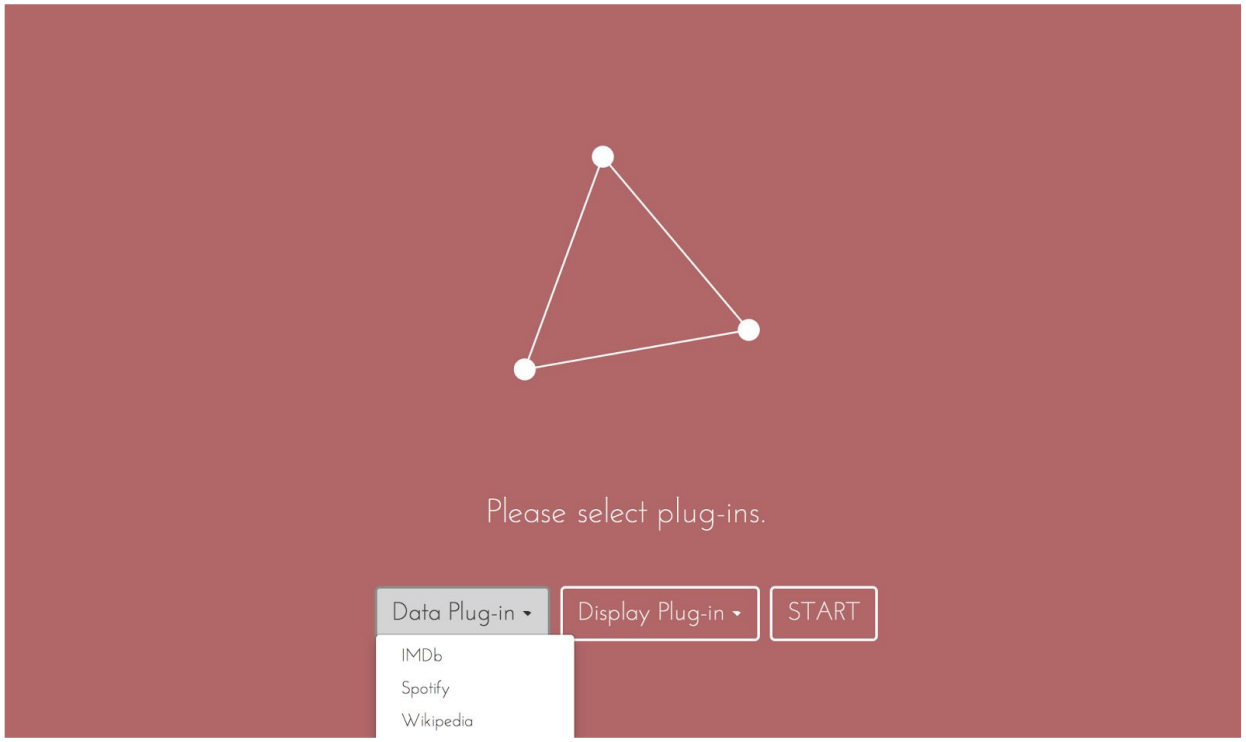
To both provide flexibility and minimize load time, the data plugin would need to load data and provide the name of neighboring nodes. The framework would also ask plugin for relevant media only when user has clicked on relevant node in order to prevent framework to load unnecessary resources. The following are the example data plugins that we have implemented.

1. **Wikipedia Data Plugin**: By scraping the Wikipedia page, the data plugin fetches related topic of a given page. Users can explore Wikipedia pages that are contextually related to each other. It not only provides the written information about the topic, but also visualizes various media files like images and videos in the network graph.
2. **IMDb Data Plugin**: Through IMDb data plugin, users can explore their movie tastes. The plugin fetches the data of initially chosen movie from IMDb website and shows the related movies based on IMDb users' watch history. Movie poster is displayed as the main media while still shots and top reviews from people are included as sub medias.
3. **Spotify Data Plugin**: Using the Spotify API, Similar to panther.audio, Spotify Data Plugin explores user's musical taste by showing musicians of similar style on the neighboring nodes. Users can listen to short audio clips of artists' tracks, get information about their genres, and take a look at their photos and album covers.

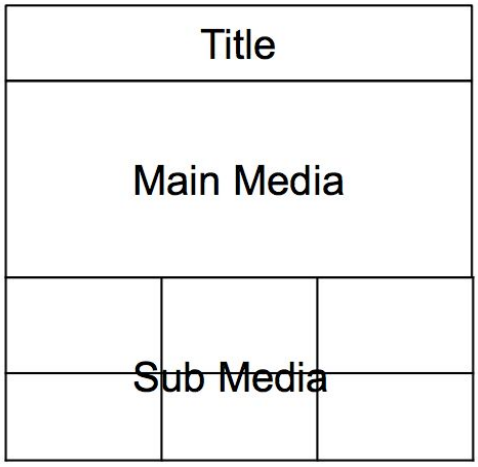
B. Display Plugin

The following is a snapshot of the main graph the application provides. The client can specify misc of the graphs like number of child node in directed graph or the

background color of the application. The loading page where user can choose which data and display plugin to load is as followed.



The most important customization that the client have to consider is the layout in the media panel. When a node is on focused, the media panel should display relevant information about the focused node.



The panel has to contain one title, one main media (media can be of either audio, video, text, or image), and arbitrary number of sub-media. The figure to the left shows the underlying layout of the media panel. Client should specify the number of row and column they wanted the sub media region to contain, as well as an array of submedia to be included in the sub media region in the display plugin.

C. Sample Plugins

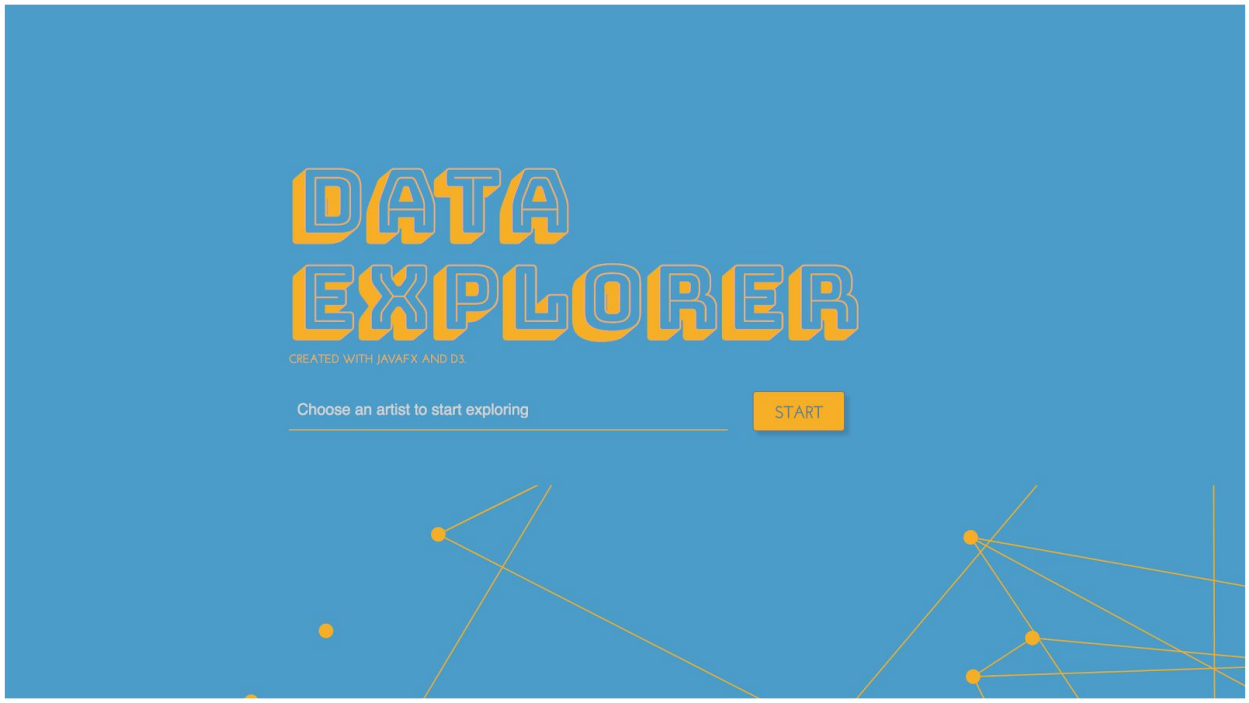
1. **Simple Display Plugin**: This plugin only displays the title with one main media without any sub medias. It uses default color scheme for background and text, and each node has three neighboring nodes. This plugin modifies HTML strings with image tags; it displays images with rounded borders with radius of 60%.
2. **ThreeCol Display Plugin**: Unlike Simple Display Plugin, this display plugin has a title, main media, and three submedias. Submedias are displayed in 1 x 3 structure, so they are just horizontally aligned. The nodes have

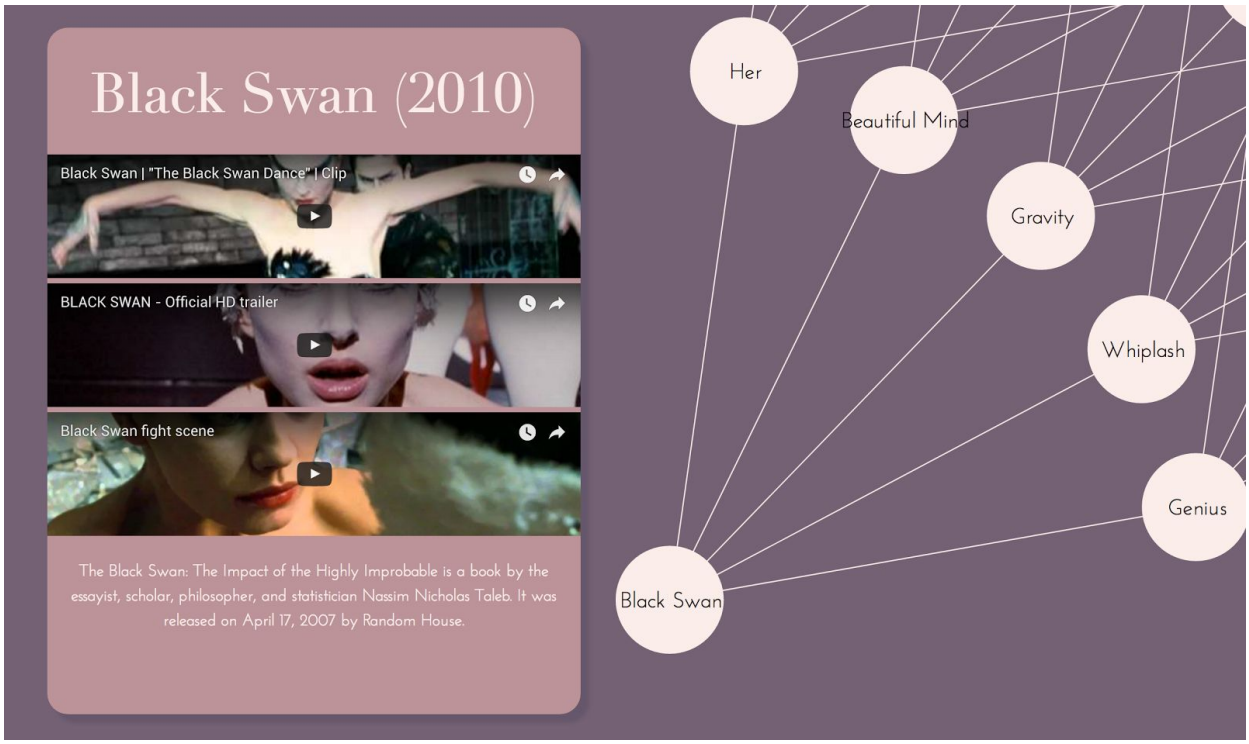
three neighboring nodes and uses a different color scheme. RenderHTML method is empty in this case; it doesn't change the raw HTML.

3. **TwoRow Display Plugin**: This display plugin has a title, main media, and two sub medias. Sub medias are located below the main media, each occupying one row(2 x 1 structure). It uses a different color scheme but has 4 neighboring nodes like the photo below. Because this is a plugin related to users' musical tastes, we've decided to modify HTML strings with audio tags. For users' convenience, audio files are automatically played and repeated until another node is clicked. Also, like simple display plugin, it displays images with rounded borders with radius of 60%.

D. Layout example

The following shows example of how the color scheme and layout can be customized by the different display plugins.





III. Installation

To run the code, developers would have to install the **JavaFX** library. The installation instruction is fairly straightforward. They can be found here:
https://www.eclipse.org/efxclipse/install.html#for-the-lazy