Interactive Comments for Online Code Sharing

Arjun Kahlon, Hiroya Gojo, Joël Porquet-Lupine.

University of California, Davis. Dept of Computer Science

Introduction

Today, the most popular method to share snippets of code online is through GitHub Gists. Gists provides an online platform for users to upload files, and display code in a neat, formatted way. In addition, GitHub provides embeddable script tags that can present a gist's content within a basic HTML webpage. For this reason, gists have become one of the preferred ways for tech writers to showcase code within their articles.

While convenient, embedded gists are shown to visitors in a read-only way, with no possibility of interaction. Our proposed project, LupGist, aims to develop a lightweight embeddable commenting system that allows visitors to directly comment on the code displayed in a gist. This way, visitors can give precise feedback on certain lines of code, or ask specific questions on areas of confusion.

Methodology

LupGist takes a full-stack approach in development, aiming to provide a self hosted server that will provide users to share gists that can be viewed and commented upon. Our development methodology is as follows:

1. Develop self-hosted web server, hosting gists as well as comments
2. Create an API for front-end calls to receive proper information
3. Design an embeddable front-end to present the contents of gists and comments
4. Implement commenting through the existing front-end design

GitHub Gists

```
1  // A function to implement bubble sort
2  void bubbleSort(int arr[], int n)
3  {
4      int i, j;
5      for (i = 0; i < n-2; i++)
6
7      // Last i elements are already in place
8      for (j = 0; j < n-i-1; j++)
9          if (arr[j] > arr[j+1])
10             swap(&arr[j], &arr[j+1]);
11 }
```

testLupGist.cpp hosted with ❤ by GitHub view raw

Projected LupGist Design

```
1  // A function to implement bubble sort
2  void bubbleSort(int arr[], int n)
3  {
4      int i, j;
2  5      for (i = 0; i < n-2; i++)
6
7      // Last i elements are already in place
8      for (j = 0; j < n-i-1; j++)
1  9          if (arr[j] > arr[j+1])
   >  Xanthiion: What is happening here? Shouldn't this be
      flipped if we are sorting from max to min?
10             swap(&arr[j], &arr[j+1]);
11 }
```

testLupGist.cpp hosted with ❤ by GitHub view raw

Program Implementation

LupGist's web server is designed to satisfy basic CRUD calls from the front-end while listening to a designated entrypoint. In return, the user is able to upload, edit, and display both gists and comments on their webpages. For LupGist to work, it will first have to be installed and configured with basic website information. Then, to incorporate a gist onto a webpage both a specific script tag and div will need to be placed within the website's HTML. Upon compilation this script tag will make a GET call to retrieve all required information for the gist, and the div will simply be placed where the gist should be printed. In addition, this div will require a "hash" attribute, which should be set to the hash provided upon uploading a gist through the admin panel. This admin panel is available for sign in through a given entry point, and allows moderators to upload new gists as well as make changes to existing information in the server's database.

Results

Currently LupGist is within step 3 of the listed methodology, and appears as so:

```
1  void bubbleSort(int arr[], int n)
2  {
3  int i, j;
4  for (i = 0; i < n-2; i++)
5
6  for (j = 0; j < n-i-1; j++)
7  if (arr[j] > arr[j+1])
8  swap(&arr[j], &arr[j+1]);
9  }
```

LupGist Alpha

To obtain this a gist was first uploaded to the self-hosted web server, followed by the addition of a specified script tag to the webpages HTML. This allows the webpage to pull information from our web server, and present the contents of the gist. Such an implementation holds promise for the future addition of comments, which is expected to take place within the Spring 2021 quarter.