

Walkthrough of RedditListener Coding Exercise for Jack Henry

Kyle Romero

8/13/24



Important Info

- **App Type:** C# Console App with Terminal Output
- **Hosted at:**
<https://github.com/romero927/RedditListener>
- **Targets:** .NET 8.0
- **Written in:** Visual Studio Community 2022
- **Config file:**
<https://github.com/romero927/RedditListener/blob/main/RedditListener/config.json>
- **Unit Testing:** xUnit
- **License:** MIT

GitHub Repo

The screenshot shows the GitHub repository page for 'RedditListener' by user 'romero927'. The repository is public and has 1 branch, 6 tags, 1 star, and 1 fork. The commit history shows a recent commit 'Add missing interfaces' by 'romero927' 6bd0357 - yesterday, with 71 commits in total. The file list includes .github/workflows, .vs, RedditListener, .gitignore, Example Output New Mode 6+hrs running.txt, Example Output Top Mode 5 Subreddits.txt, Example Reddit Auth Token Call (Postman).png, Example Reddit Top Listing (programming).json, LICENSE, README.md, RedditListener.sln, and RedditListener_Architecture.png. The README section is visible, showing the project description: 'Monitor Defined Subreddit(s) and output top post and top author data in near real-time.' and a list of bullet points: 'Written by Kyle Romero, 2024 (https://github.com/romero927 / https://kgromero.com)', 'Written in Visual Studio Community 2022.', and 'Targeting .NET 8 C#.' The right sidebar shows the 'About' section with a description, 'Readme', 'MIT license', 'Activity', '1 star', '1 watching', and '0 forks'. The 'Releases' section shows 'v1.2.0.3' as the latest release, published yesterday, with 5 releases in total. The 'Packages' section shows 'No packages published' and a link to 'Publish your first package'. The 'Deployments' section shows 'github-pages' as the latest deployment, published yesterday, with 7 deployments in total. The 'Languages' section shows a bar chart for C# at 100.0%.

RedditListener Public

main 1 Branch 6 Tags

Go to file Add file Code

romero927 Add missing interfaces ✓ 6bd0357 - yesterday 71 Commits

File	Commit	Time
.github/workflows	Create dotnet.yml	last week
.vs	Add missing interfaces	yesterday
RedditListener	Add missing interfaces	yesterday
.gitignore	Update .gitignore	5 days ago
Example Output New Mode 6+hrs running.txt	Update and rename Example Output 6+hrs running.txt to Ex...	last week
Example Output Top Mode 5 Subreddits.txt	Update Example Output Top Mode 5 Subreddits.txt	last week
Example Reddit Auth Token Call (Postman).png	Add files via upload	5 days ago
Example Reddit Top Listing (programming).json	Add files via upload	5 days ago
LICENSE	Create LICENSE	yesterday
README.md	Update README.md	yesterday
RedditListener.sln	v1.0	last week
RedditListener_Architecture.png	Rename RedditListener.png to RedditListener_Architecture.p...	last week

README MIT license

RedditListener

Monitor Defined Subreddit(s) and output top post and top author data in near real-time.

- Written by Kyle Romero, 2024 (<https://github.com/romero927> / <https://kgromero.com>)
- Written in Visual Studio Community 2022.
- Targeting .NET 8 C#.

About

Monitor Defined Subreddit(s) and output top post and top author data in near real-time.

Readme MIT license Activity 1 star 1 watching 0 forks

Releases 6

v1.2.0.3 Latest yesterday + 5 releases

Packages

No packages published [Publish your first package](#)

Deployments 8

github-pages yesterday + 7 deployments

Languages

C# 100.0%




README Page available at: <https://romero927.github.io/RedditListener/>

GitHub Latest
Release:

v1.2.0.3

Releases / v1.2.0.3



v1.2.0.3 Latest Compare


 romero927 released this yesterday  v1.2.0.3  6bd0357

Added missing interfaces.

Full Changelog: [v1.2.0.2...v1.2.0.3](#)

▼ Assets 2

 Source code (zip)	yesterday
 Source code (tar.gz)	yesterday



<https://github.com/romero927/RedditListener/releases/tag/v1.2.0.3>

GitHub Actions: Verify Build

The screenshot displays the GitHub Actions interface for a workflow named '.NET'. The main heading is 'Add missing interfaces #21', accompanied by a green checkmark icon. Below this, the 'Summary' tab is selected. The 'Jobs' section lists a single job named 'build', which is marked as successful with a green checkmark. The 'Run details' section is expanded, showing a list of steps that all completed successfully, each with a green checkmark and a right-pointing chevron. The steps are: 'Set up job', 'Run actions/checkout@v4', 'Setup .NET', 'Restore dependencies', 'Build', 'Test', 'Post Setup .NET', 'Post Run actions/checkout@v4', and 'Complete job'. The overall status of the job is 'succeeded yesterday in 19s'.

← .NET

✓ Add missing interfaces #21

Summary

Jobs

✓ build

Run details

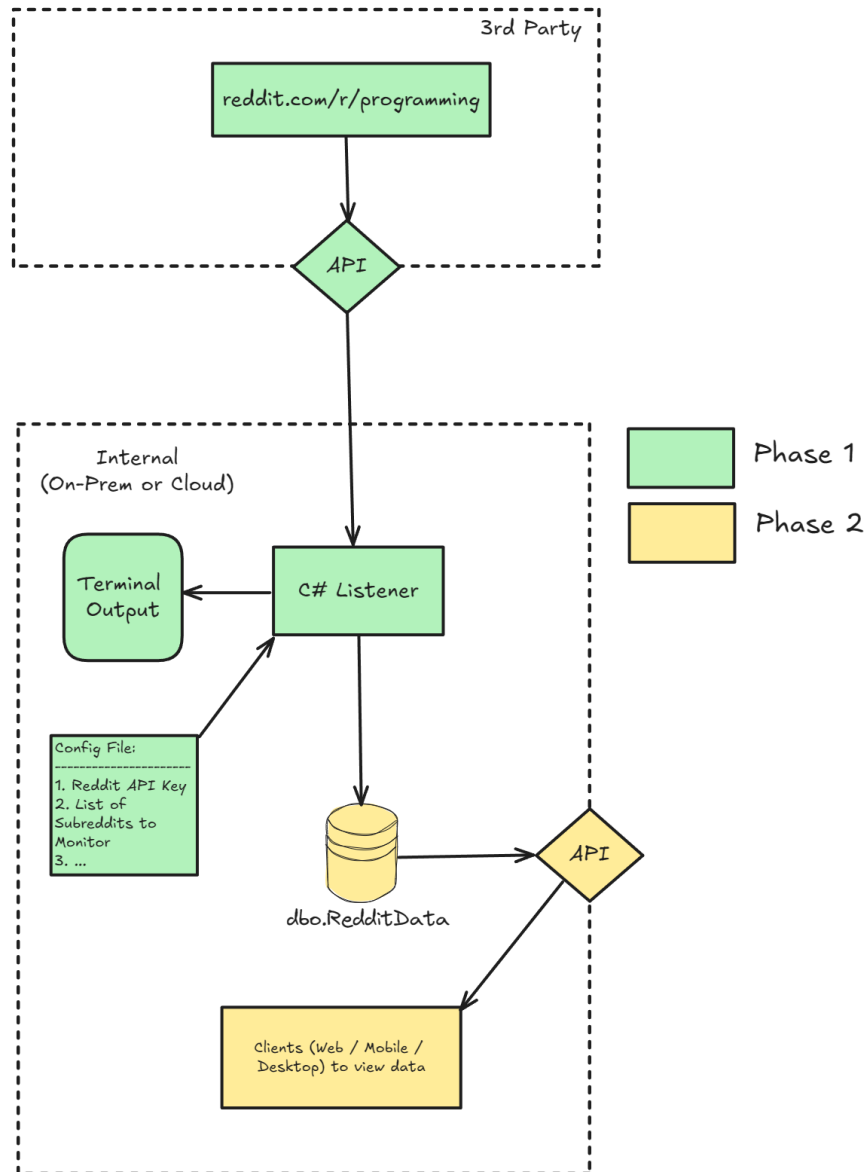
Usage

Workflow file

build
succeeded yesterday in 19s

- > ✓ Set up job
- > ✓ Run actions/checkout@v4
- > ✓ Setup .NET
- > ✓ Restore dependencies
- > ✓ Build
- > ✓ Test
- > ✓ Post Setup .NET
- > ✓ Post Run actions/checkout@v4
- > ✓ Complete job

<https://github.com/romero927/RedditListener/actions>



Architecture



Technical Decisions

Reddit API logic will be in a static utility class for portability / reusability.

All classes will have interfaces created so that we have a defined data shape for everything. This adds a layer of protection against people removing needed props and would allow us to share our data shapes with other teams easily.

Basic xUnit testing integrates nicely with GitHub actions on checkin

Config.json allows changing app settings without recompiling

One thread per subreddit monitored allows us to maximize server resource usage

We want to use as many allowed Reddit API requests without going over the limit in the given time period before request count reset. Following logic is used:

- $\text{TimeBeforeNextRequest} = ((\text{xRateLimitReset} / \text{xRateLimitRemaining}) * 1000) * \text{NumberOfThreads};$
- This is recalculated each request, so if there is a network delay, it will self-adjust

Configuration Setup: config.json

- **AuthenticationString:** What is our Reddit App ID? Used to get Token
- **UserAgent:** What is my custom user agent?
- **SubRedditsToMonitor:** What subreddits do I want to monitor?
- **NumberOfPostsToTrack:** How many posts do I want to show for each subreddit? (Top #)
- **NumberOfAuthorsToTrack:** How many Authors do I want to show for each subreddit? (Top #)
- **Mode:**
 - **new** = Look at all posts made from the point that the app was started and calculate the top NumberOfPostsToTrack and NumberOfAuthorsToTrack from those posts.
 - This uses the new.json listing and will page backwards through the listing data slices until it finds the first post that was created after start.
 - Time between requests will slow down over time as the number of pages you have to go through increases, as each page uses a request.
 - **top** = Look at the top posts listing and calculate the top NumberOfPostsToTrack and NumberOfAuthorsToTrack from those posts. Looks at current top 100.
- **MaxDegreeOfParallelism:** How many parallel threads can run at once?

Example Reddit API Top Listing (/r/programming)

[https://github.com/romero927/RedditListener/blob/main/Example%20Reddit%20Top%20Listing%20\(programming\).json](https://github.com/romero927/RedditListener/blob/main/Example%20Reddit%20Top%20Listing%20(programming).json)

RedditListener / Example Reddit Top Listing (programming).json

romero927 Add files via upload ✓

Code Blame 11757 lines (11757 loc) · 434 KB Code 55% faster with GitHub Copilot

```
1  {
2    "kind": "Listing",
3    "data": {
4      "after": "t3_1ehjv6p",
5      "dist": 100,
6      "modhash": "p1hxp3j1712b86a345e3000c593858ac90bfff6bcad6ed3616d",
7      "geo_filter": "",
8      "children": [
9        {
10         "kind": "t3",
11         "data": {
12           "approved_at_utc": null,
13           "subreddit": "programming",
14           "selftext": "",
15           "author_fullname": "t2_fv2di",
16           "saved": false,
17           "mod_reason_title": null,
18           "gilded": 0,
19           "clicked": false,
20           "title": "Let's blame the dev who pressed \"Deploy\"",
21           "link_flair_richtext": [ ],
22           "subreddit_name_prefixed": "r/programming",
23           "hidden": false,
24           "pwl": 6,
25           "link_flair_css_class": null,
26           "downs": 0,
27           "top_awarded_type": null,
28           "hide_score": false,
29           "name": "t3_1e8ipxf",
30           "quarantine": false,
31           "link_flair_text_color": "dark",
32           "upvote_ratio": 0.91,
33           "author_flair_background_color": null,
34           "subreddit_type": "public",
35           "author": "t2_fv2di"
```

Example App Output: Top Mode

<https://github.com/romero927/RedditListener/blob/main/Example%20Output%20Top%20Mode%205%20Subreddits.txt>

```
Process Started: 8/7/2024 2:48:33 PM +00:00
Press ESC to stop

#####
Subreddit: news
-----

Number of Threads: 5
Delay Between Requests: 30 seconds
Request Limit Used: 5
Request Limit Remaining: 95
Request Limit Reset: 595 seconds

-----

Top 5 Posts:
-----

Title: X, Owned by Elon Musk, Brings Antitrust Suit Accusing Advertisers of a Boycott
Upvotes: 29264
Author: hdcasel
Created UTC: 8/6/2024 3:53:08 PM +00:00
Post ID: 1ellcib
Permalink: /r/news/comments/1ellcib/x_owned_by_elon_musk_brings_antitrust_suit/

Title: Wall St bounces back after global stocks rout
Upvotes: 4828
Author: VesterSkill
Created UTC: 8/6/2024 6:13:00 PM +00:00
Post ID: 1elovpq
Permalink: /r/news/comments/1elovpq/wall_st_bounces_back_after_global_stocks_rout/

Title: Pennsylvania Supreme Court agrees to review suicide ruling in case of woman with 20 stab wounds
Upvotes: 4441
Author: Silent-Resort-3076
Created UTC: 8/7/2024 1:21:26 PM +00:00
Post ID: 1embmsr
Permalink: /r/news/comments/1embmsr/pennsylvania_supreme_court_agrees_to_review/

Title: Trump possible target of Pakistani national charged in murder-for-hire plot
Upvotes: 4016
Author: Big-Heron4763
Created UTC: 8/6/2024 7:30:41 PM +00:00
Post ID: 1elqub9
Permalink: /r/news/comments/1elqub9/trump_possible_target_of_pakistani_national/

Title: 56 days and counting: Two NASA astronauts are still in space as tests on Boeing capsule continue
Upvotes: 3223
Author: Silent-Resort-3076
Created UTC: 8/6/2024 3:29:12 PM +00:00
Post ID: 1elkqzk
Permalink: /r/news/comments/1elkqzk/56_days_and_counting_two_nasa_astronauts_are/

-----

Top 5 Authors with Most Posts:
-----

Author: Silent-Resort-3076
# Posts: 2

Author: olegvas21
# Posts: 2

Author: hdcasel
# Posts: 1

Author: VesterSkill
# Posts: 1

Author: Big-Heron4763
# Posts: 1

-----
```

Example App Output: New Mode

<https://github.com/romero927/RedditListener/blob/main/Example%20Output%20New%20Mode%206%2Bhrs%20running.txt>

```
Process Started: 8/7/2024 4:53:19 AM +00:00
Press ESC to stop

#####
Subreddit: news
-----
Number of Threads: 2
Delay Between Requests: 24 seconds
Request Limit Used: 89
Request Limit Remaining: 11
Request Limit Reset: 134 seconds

-----
Top 5 Posts:
-----
Title: Pennsylvania Supreme Court agrees to review suicide ruling in case of woman with 20 stab wounds
Upvotes: 727
Author: Silent-Resort-3076
Created UTC: 8/7/2024 1:21:26 PM +00:00
Post ID: 1embmsr
Permalink: /r/news/comments/1embmsr/pennsylvania_supreme_court_agrees_to_review/

Title: Moment dog sparks house fire after chewing power bank
Upvotes: 198
Author: GubmintTroll
Created UTC: 8/7/2024 10:49:07 AM +00:00
Post ID: 1em8nub
Permalink: /r/news/comments/1em8nub/moment_dog_sparks_house_fire_after_chewing_power/

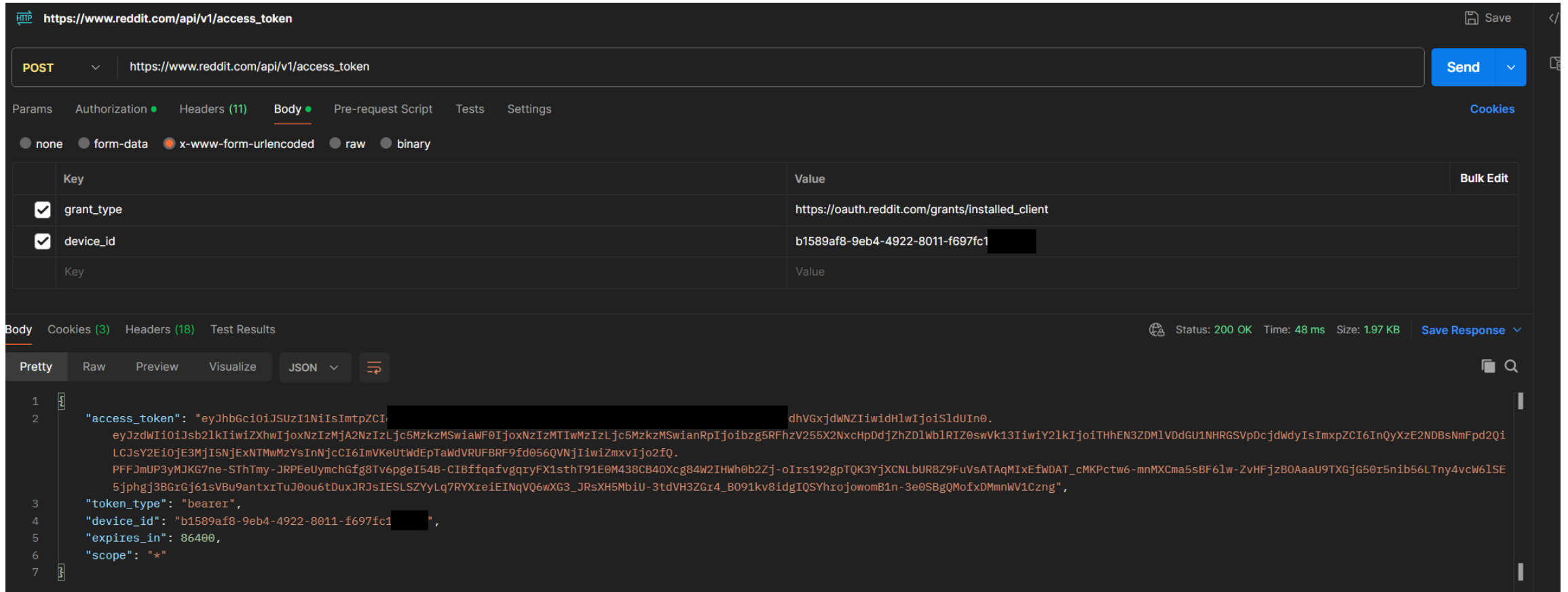
Title: Part of a hotel on Germany's Mosel River collapses. 2 people are dead and 2 are trapped
Upvotes: 91
Author: olegvas21
Created UTC: 8/7/2024 12:53:15 PM +00:00
Post ID: 1emaztf
Permalink: /r/news/comments/1emaztf/part_of_a_hotel_on_germanys_mosel_river_collapses/

-----
Top 5 Authors with Most Posts:
-----
Author: Silent-Resort-3076
# Posts: 1

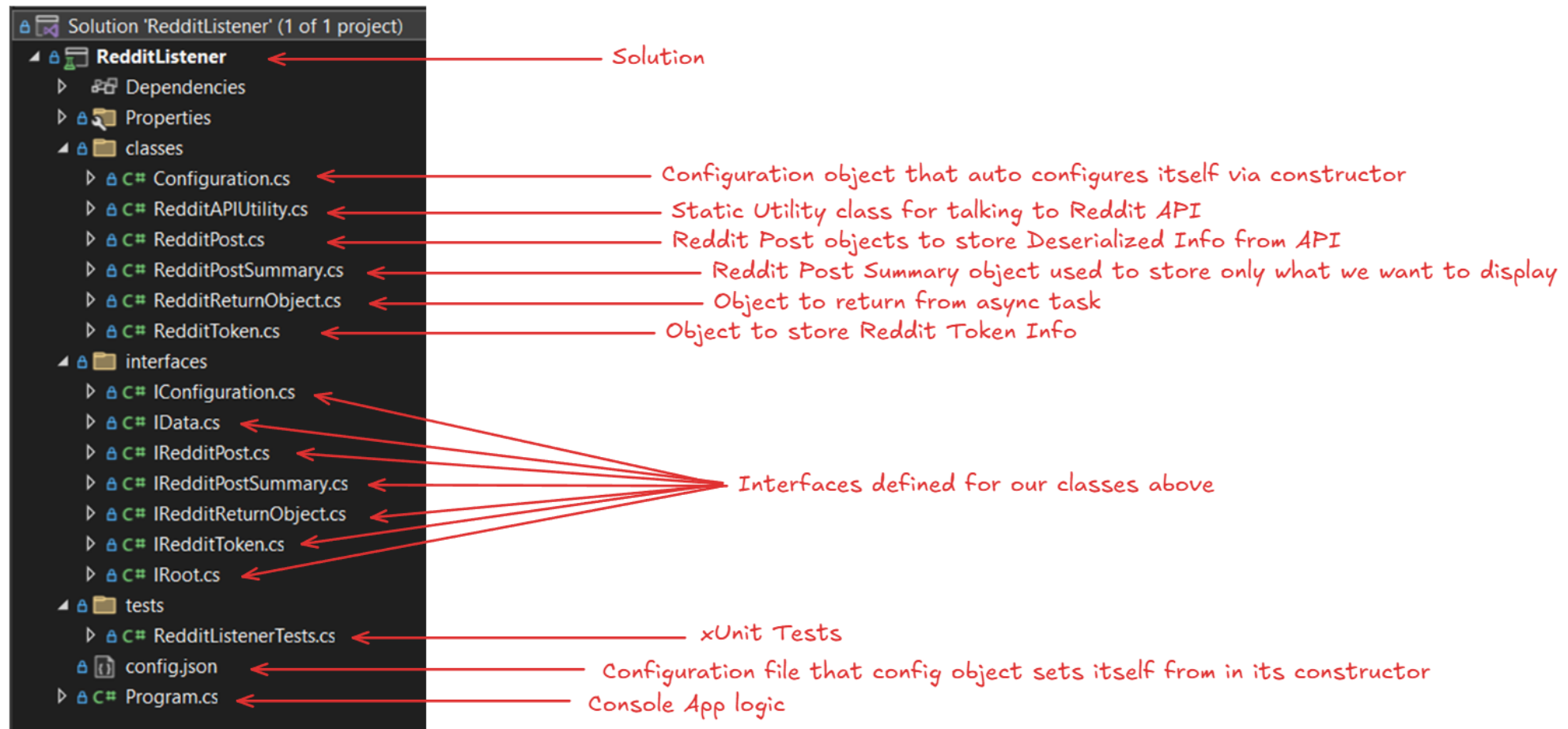
Author: olegvas21
# Posts: 1

Author: GubmintTroll
# Posts: 1
```

Example Reddit Auth Token Call



[https://github.com/romero927/RedditListener/blob/main/Example%20Reddit%20Auth%20Token%20Call%20\(Postman\).png](https://github.com/romero927/RedditListener/blob/main/Example%20Reddit%20Auth%20Token%20Call%20(Postman).png)



Solution Structure



Code Highlights w/ Notes
Below



Program.cs pt1

- Add includes
- Kick off async MainApp function where I have put all the logic
- Setup Config and Client objects
- Capture current start time in Unix Epoc (since that is what Reddit returns)
- Do until Escape key is hit:
 - Write Start time and Config loaded from to console
 - Setup our parallel async tasks options
 - ...continued next slide

```
//System Includes
using System.Net.Http.Headers;
using System.Text.Json;
using RedditListener;

//Kick Off App
await MainApp();

1 reference
static async Task MainApp()
{
    //Instantiate config object
    Configuration Config = new Configuration();

    //Setup HTTP Client
    HttpClient Client = new HttpClient();
    Client = await RedditAPIUtility.SetupClient(Config, Client);

    //Get UNIX UTC Time at start of app
    DateTime StartTime = DateTime.UtcNow;
    double StartTimeUTC = ((DateTimeOffset)StartTime).ToUnixTimeSeconds();

    do
    {
        //Let's loop until Escape is hit
        while (!Console.KeyAvailable)
        {
            //Inform of exit button
            Console.WriteLine("Process Started: " + DateTimeOffset.FromUnixTimeSeconds((long)StartTimeUTC));
            Console.WriteLine("Config loaded from: " + Config.SetFrom);
            Console.WriteLine("Press ESC to stop");

            //Setup Parallel Async HTTP Requests
            ParallelOptions ParallelOptions = new()
            {
                MaxDegreeOfParallelism = Config.MaxDegreeOfParallelism
            };

            //1 thread per subreddit we will monitor
            int NumberOfThreads = Config.SubRedditsToMonitor.Length;
        }
    }
}
```

Program.cs pt2

- Kick off parallel threads (one per subreddit being monitored)
- Independently for each thread:
 - Get our data from Reddit API using the utility class
 - Use utility class to calculate how long this thread should sleep before next request
 - Write to console
 - Subreddit
 - Threads
 - Request Limit Info
 - Top Posts Summaries
- ...continued next slide

```
//Start the parallel threads
await Parallel.ForEachAsync(Config.SubRedditsToMonitor, ParallelOptions, async (Subreddit, ct) =>
{
    //Instantiate Return Object
    RedditReturnObject ReturnedData = new RedditReturnObject();
    int ThreadSeep = 1000;
    try
    {
        //Call reddit and get the new posts data
        ReturnedData = await RedditAPIUtility.ProcessRedditPostsAsync(Client, StartTimeUTC, Subreddit, Config);

        //Figure out how long we should pause thread based on rate limits and number of threads that will also count against limits
        ThreadSeep = RedditAPIUtility.CalculateThreadSleep(ReturnedData.xRateLimitReset, ReturnedData.xRateLimitRemaining, NumberOfThreads);

        //Output the thread and rate limit data to console in a readable format
        Console.WriteLine("");
        Console.WriteLine("#####");
        Console.WriteLine("Subreddit: " + Subreddit);
        Console.WriteLine("-----");
        Console.WriteLine("Number of Threads: " + NumberOfThreads);
        Console.WriteLine("Delay Between Requests: " + (ThreadSeep / 1000) + " seconds");
        Console.WriteLine("Request Limit Used: " + ReturnedData.xRateLimitUsed);
        Console.WriteLine("Request Limit Remaining: " + ReturnedData.xRateLimitRemaining);
        Console.WriteLine("Request Limit Reset: " + ReturnedData.xRateLimitReset + " seconds");
        Console.WriteLine("");
        Console.WriteLine("-----");
        Console.WriteLine("Top " + ReturnedData.NumberOfPostsToTrack + " Posts:");
        Console.WriteLine("-----");

        //Output the Top Posts data to console in a readable format
        if (ReturnedData is not null && ReturnedData.PostSummaries.Count > 0)
        {
            foreach (RedditPostSummary PostSummary in ReturnedData.PostSummaries)
            {
                Console.WriteLine("Title: " + PostSummary.Title);
                Console.WriteLine("Upvotes: " + PostSummary.Upvotes);
                Console.WriteLine("Author: " + PostSummary.Author);
                Console.WriteLine("Created UTC: " + DateTimeOffset.FromUnixTimeSeconds(PostSummary.CreatedUTC));
                Console.WriteLine("Post ID: " + PostSummary.ID);
                Console.WriteLine("Permalink: " + PostSummary.Permalink);
                Console.WriteLine("");
            }
        }
        else
        {
            Console.WriteLine("NO POSTS YET, PLEASE STAND BY");
        }
    }
});
```


Program.cs pt2

- Independently for each thread:
 - Write to console
 - Top Authors
 - Catch and display error if there was one. Non-blocking.
 - Sleep Thread for calculating amount of time so that we maximize the number of requests we can get in before the request limit resets.
 - Sleep thread for 1s after clearing (basically acting as a debounce to allow the screen to settle, without this delay things weren't displaying correctly.)
- If escape key is hit, exit the app.

```
//Output the Top Authors data to console in a readable format
Console.WriteLine("-----");
Console.WriteLine("Top " + ReturnedData.NumberOfAuthorsToTrack + " Authors with Most Posts:");
Console.WriteLine("-----");

if (ReturnedData.AuthorCounts.Count > 0)
{
    foreach (KeyValuePair<string, int> Entry in ReturnedData.AuthorCounts)
    {
        Console.WriteLine("Author: " + Entry.Key);
        Console.WriteLine("# Posts: " + Entry.Value);
        Console.WriteLine("");
    }
}
else
{
    Console.WriteLine("NO AUTHORS YET, PLEASE STAND BY");
}
Console.WriteLine("-----");

//Sleep thread for calculated time to maximize usage of rate limit spread across threads
Thread.Sleep(ThreadSeep);
}
catch (Exception ex)
{
    //There was an error, display output
    Console.WriteLine("#####");
    Console.WriteLine("ERROR: " + ex.Message);
    Console.WriteLine("#####");
}
});

//Clear console for next display, and add a tiny delay to make sure there are no weird display artifacts
Console.Clear();
Thread.Sleep(1000);
}

//Continue until escape key is pressed
} while (Console.ReadKey(true).Key != ConsoleKey.Escape);
```

Configuration.cs : Constructor

- Attempts to read and then deserialize our config.json file
- If successful, load config.json values into config object, which will be used for app
- If unsuccessful, default to sane values and continue

```
//Configuration Class, Implements IConfiguration Interface
9 references
internal class Configuration : RedditListener.interfaces.IConfiguration
{
    //Constructor: Load values from config.json. Fallback to sane defaults if there is an error.
    3 references
    public Configuration()
    {
        try
        {
            //Load our JSON config file and deserialize into an object
            JSONDeserializedConfig ConfigFile = JsonFileReader.Read<JSONDeserializedConfig>(@".\config.json");
            //Construct our Config object from the deserialized settings
            AuthenticationString = ConfigFile.AuthenticationString;
            UserAgent = ConfigFile.UserAgent;
            SubRedditsToMonitor = ConfigFile.SubRedditsToMonitor;
            NumberOfPostsToTrack = ConfigFile.NumberOfPostsToTrack;
            NumberOfAuthorsToTrack = ConfigFile.NumberOfAuthorsToTrack;
            Mode = ConfigFile.Mode;
            MaxDegreeOfParallelism = ConfigFile.MaxDegreeOfParallelism;
            SetFrom = "config.json";
        }
        catch (Exception ex)
        {
            //Fallback to Sane Defaults
            AuthenticationString = "<APP ID GOES HERE>";
            UserAgent = "kgromerov0.0.1";
            SubRedditsToMonitor = new string[]
            {
                "askreddit",
                //"news",
                //"gaming",
                //"programming",
                //"todayilearned"
            };
            NumberOfPostsToTrack = 5;
            NumberOfAuthorsToTrack = 5;
            Mode = "top";
            MaxDegreeOfParallelism = 5;
            SetFrom = "default";
        }
    }
}
```

RedditAPIUtility.cs : SetupClient

- Setup the http client object
- Build out the Authorization and User-Agent headers and add to client
- Attempt to get the Reddit access_token
- If access_token received, add it to the client object

```
internal static async Task<HttpClient> SetupClient(Configuration Config, HttpClient Client)
{
    Client = new();
    Client.DefaultRequestHeaders.Accept.Clear();

    //Convert our Auth settings into Base64 and add a user agent
    var Base64String = Convert.ToBase64String(
        System.Text.Encoding.ASCII.GetBytes(Config.AuthenticationString));
    Client.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Basic", Base64String);
    Client.DefaultRequestHeaders.Add("User-Agent", Config.UserAgent);

    //Get the Reddit Token
    string TokenJSON = await RedditListener.RedditAPIUtility.PostForRedditToken(Client);
    RedditToken? Token = JsonSerializer.Deserialize<RedditToken>(TokenJSON);
    if (Token is not null && Token.Access_Token != string.Empty)
        Client.DefaultRequestHeaders.Add("access_token", Token.Access_Token);

    return Client;
}

1 reference
internal static async Task<string> PostForRedditToken(HttpClient client)
{
    //Get a access_token from reddit. Might actually not be needed for this but could be useful later
    var Values = new Dictionary<string, string>
    {
        { "grant_type", "https://oauth.reddit.com/grants/installed_client" },
        { "device_id", "b1589af8-9eb4-4922-8011-f697fc1b93a5" }
    };
    var Content = new FormUrlEncodedContent(Values);
    var Response = await client.PostAsync("https://www.reddit.com/api/v1/access_token", Content);
    var ResponseString = await Response.Content.ReadAsStringAsync();
    return ResponseString;
}
```

RedditAPIUtility.cs : Process Posts pt1

- Get the config defined Reddit listing
- Get the current Rate Limiting info from the API
- Pull the JSON from the response
- Deserialize JSON
- Setup objects to be used for our sorting logic

```
internal static async Task<RedditReturnObject> ProcessRedditPostsAsync(HttpClient Client, double StartTimeUTC, string Subreddit, Configuration Config)
{
    //GET the new.json for the selected subreddit.
    HttpResponseMessage Response = await Client.GetAsync(
        "https://www.reddit.com/r/" + Subreddit + "/" + Config.Mode + ".json?limit=100");

    Response.EnsureSuccessStatusCode();

    //What are the rate limits currently at?
    int xRateLimitUsed = Convert.ToInt32(Convert.ToDouble((Response.Headers.GetValues("x-ratelimit-used").First())));
    int xRateLimitRemaining = Convert.ToInt32(Convert.ToDouble((Response.Headers.GetValues("x-ratelimit-remaining").First())));
    int xRateLimitReset = Convert.ToInt32(Convert.ToDouble((Response.Headers.GetValues("x-ratelimit-reset").First())));

    //Pull out the response JSON
    string JSON = await Response.Content.ReadAsStringAsync();

    //Deserialize JSON into Object to contain posts
    Root? PostsCollection = JsonSerializer.Deserialize<Root>(JSON);

    //New List that we will use to truncate any data we dont need for posts and sort
    List<RedditPostSummary> PostSummaries = new List<RedditPostSummary>();

    //Dictionary we will use to track # of posts by author
    Dictionary<string, int> AuthorCounts = new Dictionary<string, int>();

    //ID if we need to go to next page
    bool EndPaging = false;
```

RedditAPIUtility.cs :

Process Posts pt2

For Top Mode:

- For each post, load the post summary data into our new collection.
- Add or Update Author info in Author Dictionary and increment post counter

```
//Loop through posts
foreach (RedditPost Post in PostsCollection.data.children)
{
    if (Config.Mode == "new") ...
    else
    {
        if (Post.data.created_utc is not null)
        {
            RedditPostSummary PostSummary = new RedditPostSummary();
            PostSummary.Title = Post.data.title;
            PostSummary.Author = Post.data.author;
            PostSummary.Upvotes = Post.data.ups;
            PostSummary.CreatedUTC = (long)(Post.data.created_utc);
            PostSummary.ID = Post.data.id;
            PostSummary.PermaLink = Post.data.permalink;

            PostSummaries.Add(PostSummary);

            if (AuthorCounts is not null && Post.data.author is not null)
            {
                if (AuthorCounts.ContainsKey(Post.data.author))
                    AuthorCounts[Post.data.author]++;
                else AuthorCounts.Add(Post.data.author, 1);
            }
        }
    }
}
```


RedditAPIUtility.cs : Process Posts pt3

```
//Loop through posts
foreach (RedditPost Post in PostsCollection.data.children)
{
    if (Config.Mode == "new")
    {
        //If post was created after the app started, it is in play
        if (Post.data.created_utc is not null && Post.data.created_utc >= StartTimeUTC)
        {
            RedditPostSummary PostSummary = new RedditPostSummary();
            PostSummary.Title = Post.data.title;
            PostSummary.Author = Post.data.author;
            PostSummary.Upvotes = Post.data.ups;
            PostSummary.CreatedUTC = (long)(Post.data.created_utc);
            PostSummary.ID = Post.data.id;
            PostSummary.Permalink = Post.data.permalink;

            PostSummaries.Add(PostSummary);

            if (AuthorCounts is not null && Post.data.author is not null)
            {
                if (AuthorCounts.ContainsKey(Post.data.author))
                {
                    AuthorCounts[Post.data.author]++;
                }
                else
                {
                    AuthorCounts.Add(Post.data.author, 1);
                }
            }
        }
        else if (Post.data.created_utc is not null && Post.data.created_utc < StartTimeUTC)
        {
            //Current Post happened before app start, this will be our end condition
            EndPaging = true;
        }
    }
}
```

```
//If we are in new mode and didnt find end condition, we need to loop back to next page of data slice
while (Config.Mode == "new" && !EndPaging)
{
    //Get the next slice using after
    Response = await Client.GetAsync(
        "https://www.reddit.com/r/" + Subreddit + "/" + Config.Mode + ".json?limit=100&after=" + PostsCollection.data.after);

    Response.EnsureSuccessStatusCode();

    //What are the rate limits currently at?
    xRateLimitUsed = Convert.ToInt32(Convert.ToDouble((Response.Headers.GetValues("x-ratelimit-used").First())));
    xRateLimitRemaining = Convert.ToInt32(Convert.ToDouble((Response.Headers.GetValues("x-ratelimit-remaining").First())));
    xRateLimitReset = Convert.ToInt32(Convert.ToDouble((Response.Headers.GetValues("x-ratelimit-reset").First())));

    //Pull out the response JSON
    JSON = await Response.Content.ReadAsStringAsync();

    //Deserialize JSON into Object to contain posts
    PostsCollection = JsonSerializer.Deserialize<Root>(JSON);

    //Loop through posts
    foreach (RedditPost Post in PostsCollection.data.children)
    {
        //If post was created after the app started, it is in play
        if (Post.data.created_utc is not null && Post.data.created_utc >= StartTimeUTC)
        {
            RedditPostSummary PostSummary = new RedditPostSummary();
            PostSummary.Title = Post.data.title;
            PostSummary.Author = Post.data.author;
            PostSummary.Upvotes = Post.data.ups;
            PostSummary.CreatedUTC = (long)(Post.data.created_utc);
            PostSummary.ID = Post.data.id;
            PostSummary.Permalink = Post.data.permalink;

            PostSummaries.Add(PostSummary);

            if (AuthorCounts is not null && Post.data.author is not null)
            {
                if (AuthorCounts.ContainsKey(Post.data.author))
                {
                    AuthorCounts[Post.data.author]++;
                }
                else
                {
                    AuthorCounts.Add(Post.data.author, 1);
                }
            }
        }
        else if (Post.data.created_utc is not null && Post.data.created_utc < StartTimeUTC)
        {
            //Found our end condition
            EndPaging = true;
        }
    }
}
```

For New Mode:

- For each post, load the post summary data into our new collection.
- Add or Update Author info in Author Dictionary and increment post counter
- Continue on until we find a post that is older than our app start time
- If we didn't find a post that was older than start, page through listing and keep adding posts until we do find one

RedditAPIUtility.cs :

Process Posts pt4

```
//Order post summaries descending, and take configured top number
PostSummaries = PostSummaries.OrderByDescending(p => p.Upvotes).Take(Config.NumberOfPostsToTrack).ToList();

//Order author counts descending, and take configured top number
Dictionary<string, int> SortedAuthors = new Dictionary<string, int>();

if (AuthorCounts is not null)
    SortedAuthors = AuthorCounts.OrderByDescending(pair => pair.Value).Take(Config.NumberOfAuthorsToTrack)
        .ToDictionary(pair => pair.Key, pair => pair.Value);

//Build our return object from this task and return it.
RedditReturnObject DataToReturn = new RedditReturnObject();
DataToReturn.NumberOfPostsToTrack = Config.NumberOfPostsToTrack;
DataToReturn.NumberOfAuthorsToTrack = Config.NumberOfAuthorsToTrack;
DataToReturn.PostSummaries = PostSummaries;
DataToReturn.AuthorCounts = SortedAuthors;
DataToReturn.xRateLimitRemaining = xRateLimitRemaining;
DataToReturn.xRateLimitReset = xRateLimitReset;
DataToReturn.xRateLimitUsed = xRateLimitUsed;
return DataToReturn;
```

- Use LINQ to sort our PostSummaries List<RedditPostSummary> by UpVotes Descending and then take the config.json defined Top # of posts.
- Use LINQ to sort the AuthorCounts Dictionary<string AuthorName, int NumberPosts> by Number Posts Descending and then take the config.json defined Top # of authors.
- Return this data in our RedditReturnObject (as Task<RedditReturnObject> since this is an async task).


RedditListenerTests.cs

- A couple of simple xUnit tests:
 - Can the config file be loaded correctly?
 - Can the client object be setup correctly?
 - There is also a commented out test to see if we can get the Reddit Access Token. Token is not needed for now, so not testing against this.

```
//Make sure Config File can load
[Fact]
0 references
public void CanLoadConfigFile()
{
    //Instantiate config object
    Configuration Config = new Configuration();
    Assert.Equal("config.json", Config.SetFrom);
}

//Make sure HTTP client can get setup ok
[Fact]
0 references
public async Task CanSetupClient()
{
    //Instantiate config object
    Configuration Config = new Configuration();

    //Setup HTTP Client
    HttpClient Client = new HttpClient();
    Client = await RedditAPIUtility.SetupClient(Config, Client);
    Assert.NotNull(Client.DefaultRequestHeaders.Authorization);
}
```

romero927/RedditListener is licensed under the

MIT License

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions

Limitations

Conditions

✔ Commercial use

✔ Modification

✔ Distribution

✔ Private use

✗ Liability

✗ Warranty

📄 License and copyright notice

This is not legal advice. [Learn more about repository licenses](#)


 **romero927** Create LICENSE ✔

8fde861 · yesterday  History


Code


Blame


21 lines (17 loc) · 1.04 KB


 Code 55% faster with GitHub Copilot


Raw











```
1 MIT License
2
3 Copyright (c) 2024 Kyle Romero
4
5 Permission is hereby granted, free of charge, to any person obtaining a copy
6 of this software and associated documentation files (the "Software"), to deal
7 in the Software without restriction, including without limitation the rights
8 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
9 copies of the Software, and to permit persons to whom the Software is
10 furnished to do so, subject to the following conditions:
11
12 The above copyright notice and this permission notice shall be included in all
13 copies or substantial portions of the Software.
14
15 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
16 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
17 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
18 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
19 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
20 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
21 SOFTWARE.
```

License

Future Concerns and Known Issues

Due to the async multithreaded nature of the HTTPS requests, the console output can sometimes get out of order, especially on the first display. It resolves quickly and would not be an issue in a DB or File save.

I have run this for several hours and it was stable, but I haven't checked long term stability that would be needed for production

If there is an error, the app will continue working with basic error handling. A real app would need much more in-depth checking.

The only sorting criteria for Posts is # of Upvotes and Authors is # of Posts, so if there are multiple posts / authors with the same numbers, the display will be arbitrary. We would need to determine what additional filtering / sorting is desired.

The Top Posts and Top Authors are calculated from the overall dataset and may not correlate after sorting / filtering. For instance, the Top Posting author may be spamming low upvoted posts and may not have anything in the Top posts section.

If you start the app, let it run for a while, and restart it, the console may flash through the results quickly. This is due to it automatically trying to use the remaining requests before the reset time. It will go back to normal as soon as the request limit time resets.



Thanks!

Created by Kyle
Romero

Can be reached at:
kgromero@gmail.com
+1 (281) 857-9006

Personal Website:
<https://kgromero.com>
