URL Redirector Specification

Risheetha Bhagawatula

Overview

This URL redirector takes the short URL and points the user to the long URL page. When anyone types in the short URL they'll be taken to the page set to the long URL instead.

Background

In the case where a URL doesn't exist, a URL redirector will ensure that any backlink value is directed correctly and not lost with the old page. One of the many benefits to URL redirection is it is much more convenient to share and use a short URL compared to a long and complicated one. Additionally, a URL redirector has the ability to track clicks on each link you share. This way, you can see a breakdown of visitors by demographics, such as country or gender.

How the User Interface Works

The user will input the short URL into the address box. If the short URL is found, then the user will be redirected to the long URL. However, in the case that the short URL is not found, the user will be prompted to enter a long URL in the form, and will reload the page to be redirected to the long URL. This new long URL will be saved into the dictionary for future reference.

Internal Design

The URL redirector should be created using one file named "url_main.py". In this file I should import the BaseHTTPRequestHandler and HTTPServer to establish the server. Instead of importing the usual cgi module I would import the json and os modules for easier access to the dictionary in which the short and long URL's will be stored in the url.txt file, and then can be read and written in to modify the dictionary. The reason for this is because using the cgi module in python is very error-prone. The only class to be used id the MyServer class which super classed the BaseHTTPRequestHandler class. The two methods used are the do_GET to read and gather data from the server and in which parameters are appended to the URL and then do_POST to write/update and send data to the server along with the request. The do_POST method includes the form as this method supports sending multiple part form data whereas do_GET does not. Additionally, do_POST is typically safer to store user input as the parameters are not stored in browser history or displayed in the URL.

How to Test

Have the short URL :

This test case should redirect the user to the corresponding long URL to the key(short URL) entered in the url.txt file.

Don't have the short URL and has to enter long URL:

In this test case the user should be prompted to enter a long URL. The long URL should then be tested if it is a valid URL or not. In the case of the long URL being valid, the URL will then be saved to the dictionary for future reference and then after reloading the page the user will be redirected to the entered long URL. The program should also check for different protocols to see if the long URL starts with any valid protocols. If the long URL is not valid, an exception will be thrown and the user will be redirected to a page that states that the URL is not valid.

Not able to write or read from the dictionary file or deserialize:

In this test case, an IO error will be thrown and the user will be prompted to a webpage that states that the dictionary file is not able to be loaded or written/read into.