`dcgi` library v2.1 manual

Dohn A. Arms

October 30, 2020

This is a very simple C library for parsing the information passed to a CGI program from the web server. The `d` in the name (from "Dohn") is to simply differentiate it from other CGI libraries.

1 CGI Process

When an HTML form sends the values of its fields to a CGI program or script, it encodes them into a special form. Certain characters are escaped into web-friendly expressions, and the key-value pairs are merged together into a long string.

There are also two different ways of sending the data (specified in the HTML form): **POST** and **GET**. The only real difference between the two is that a **GET** submission is passed along as part of the link address, making it able to be bookmarked, although it has a size limit; **POST** submissions are 'invisible' and can take arbitrarily long amounts of data.

The first thing a CGI program does is take the submitted data (after determining which type of submission was used) and decode it. It then parses the data into key-value pairs. *This is all this library does.*

The last thing a CGI program does is return data to the requesting client. This is done by writing to **stdout**, which gets redirected by the web server hosting the CGI program. Normally, a HTML page is returned, although it can be anything; the MIME type of the file has to be returned to the server so it knows what it is. For an HTML page, the first thing that *must* be output from your C program is `"Content-Type: text/html\r\n\r\n"`.

2 Library Inclusion

To use the installed library in your C code, include `dcgi.h`. When compiling, add the `-ldcgi` switch.

3 Language Reference

struct cgi_info This is the basic structure needed that keeps all the CGI information. All the functions need a pointer to an instance of this structure.

int cgi_parse_input(struct cgi_info *info) This function needs to be called before any other CGI function, as it reads in the data and turns it into key-value pairs. If it returns a non-negative number, it represents the number of fields parsed (zero means there were no fields found). If it returns a negative number, an error was encountered; the negative of this number (making it positive) can be used as the index with the **cgi_get_error_string** function to retrieve a message describing the error.

const char *cgi_get_error_string(int error) This function returns a string containing a description of the error encountered while running **cgi_parse_input**.

char *cgi_get_value(struct cgi_info *info, char *key) Call this when the value of a string key is needed. If the key does not exist, this function returns a **NULL** pointer. Otherwise the string value of the key is returned; if the key is present but has no value, a zero-length string is returned.

void cgi_clean_up(struct cgi_info *info) This function is called when the CGI input is no longer needed. It deallocates all its memory, *including all the key-value pairs.*

4 Future Work

There is no method to view all the available keys. I typically search for the keys I care about, so I haven't needed to add this yet.