Making CGI applications

Common Gateway Interface (CGI) programs are programs that are run by a web server and which can deliver web pages or content dynamically.

DialogScript can be used to create CGI programs. The WRITE CONSOLE command is used to write lines of HTML text which are sent by the web server to the web browser. CGI programs should be created as console applications, although that isn't strictly necessary, as most web servers create a console for any program run as a CGI executable, regardless of how it has been created. However, the only way to test a DialogScript CGI program in the VDS IDE is to view the HTML output sent to the console.

The DialogScript code below sends a page containing the message "Hello world!" to the web browser.

```
write console,"Content-Type: text/html"@lf()

write console,@lf()

write console,"<HTML>"@lf()

write console,"<HEAD>"@lf()

write console,"<TITLE>Hello</TITLE>"@lf()

write console,"</HEAD>"@lf()

write console,"<BODY>"@lf()

write console,"<P>Hello world!</P>"@lf()

write console,"</BODY>"@lf()

write console,"</HTML>"@lf()
```

Note that the WRITE command is used with @LF() at the end of each line, and not WRITELINE, in order to get the correct line endings.

Environment variables

Environment variables, the value of which can be obtained using the DialogScript @ENV function, convey important information to a CGI program. A number of environment variables are available, some of which are web server specific.

Command line arguments

CGI programs can be called from a web page in one of two ways. If a program is run using a hyperlink, like this:

```
<A HREF="/cgi-bin/cgiprog.exe?Argument_1+Argument_2">...</A>
```

then arguments may be passed to it on the command line, by adding "?" after the program name, followed by the arguments. The arguments may need to be encoded in a manner similar to the @URLENCODE function, to avoid the use of characters that are illegal in HTML text.

The "+" sign can be used to separate arguments. These will be replaced by spaces when the command line is passed to the CGI program. The command line contents may be obtained in the usual way, using the @CMDLINE function to obtain the entire command line (including the CGI program path), or the parameters %1 to %9 to obtain the first nine space-delimited arguments.

A CGI program can determine if it has been called in this way by testing the value of the environment variable REQUEST_TYPE, which should be GET.

Form variables

A more common use of a CGI program is to collect and process the data from a web form. In this case, the CGI program is called using the form POST method, like this:

<FORM METHOD="POST" ACTION="/cgi-bin/cgiprog.exe">

and the program itself can verify this by testing the value of the environment variable REQUEST_TYPE, which should be POST.

In this case, the environment variable CONTENT_LENGTH will contain a number which specifies the number of bytes of information to be transferred. This information is written to the standard input stream, and can be retrieved by reading from the console, like this:

%%clen = @env(CONTENT_LENGTH)

%%input = @read(console,%%clen)

The data is passed as a formatted string, in which the form field names are used as tags to identify the items of data, and the data is encoded to avoid the use of illegal characters. If the data is read into a DialogScript variable named %%input, as above, then the following user-defined function @CGIVAR can be used to extract each named data item into a DialogScript variable:

:cgivar

%R = @urldecode(@strslice(%%input&,%1=,&))

exit %R

like this:

%%addr = @cgivar(Address)

Error trapping

Error trapping is important in a CGI program. The default action of the VDS runtime when an untrapped error occurs is to display a message box containing the error details. This will hang the web transaction resulting eventually in an error message being displayed by the web server.

The OPTION ERRORTRAP command should be used to direct processing to an error trapping routine. This could either ignore the error, write it to a log file, or output it in the user's web browser.