NAME

dgsh-httpval − data store HTTP server

SYNOPSIS

dgsh-httpval [**−a**] [**−b** *query:command*] [**−m** *MIME-type*] [**−n**] [**−p** *port*]

DESCRIPTION

dgsh-httpval allows other programs to access *dgsh* data stores through the HTTP protocol. This simplifies the interfacing between web-based front-ends and *dgsh* programs. When *dgsh-httpval* receives a REST request with the name of a data store whose endpoint is located in the directory where *dgsh-httpval* was launched (e.g. `http://localhost:8081/mystore`), it will establish a connection with the store specified in the request, send a command to read the store's value, obtain the value, and respond with it as the document sent with the HTTP response.

Requests for files located in the directory where *dgsh-httpval* was launched will also be satisfied. The correct MIME type will be sent for files with a suffix of `html`, `js`, `json`, `png`, and `css`.

A request for the resource `.server?quit`, will cause the server to terminate processing and exit.

dgsh-httpval is normally executed from within *dgsh*-generated scripts, rather than through end-user commands. This manual page serves mainly to document its operation and the flags that can be passed to *dgsh* for modifying its behavior.

OPTIONS

−a Allow any Internet host to obtain a value from the server. By default the server will only respond to requests arriving from the local host's loop-back IP address (127.0.0.1).

−b *query:command*
The colon-separated pair specifies a dynamic query than can be sent to the server, so that it will execute the specified command and return its output. The query and the command can contain up to ten matching *scanf(3)* and *printf(3)* specifications for C integer-sized arguments, which can be used to pass data from the query to the command. An unlimited number of dynamic queries can be specified through multiple **-b** options. The type of the data returned is specified using the **-m** option.

−m *MIME-type*
Specify the MIME-type that the server will provide on the `Content-type` HTTP header for data coming from data stores and dynamic queries. By default this value is `text/plain`. Other reasonable types are `application/json`, `text/CSV`, `text/xml`, or `application/octet-stream`.

−n Read values from stores using a non-blocking read command. This means that the server will return an empty record, if no complete record is available.

−p *port* Specify the TCP port on which the server will listen for incoming HTTP requests. If no port is specified, then the server will listen on an arbitrary, system-assigned, port, and will print that port's number on its standard output. That value can be conveniently piped into *dgsh-writeval* to be made available to other processes.

EXAMPLES

Specify that a query, such as `http://localhost:63001/server-bin/pstatus?id=4892`, will run the *ps(1)* command for the specified process-id.

```
dgsh-httpval -b 'server-bin/pstatus?id=%d:ps -p %d'
```

SEE ALSO

dgsh(1), *dgsh-writeval*(1), *dgsh-readval*(1)

BUGS

The server is single-threaded and will block if a value is not available on a specified store.

The server only supports IPv4 and the HTTP 1.0 protocols. Some clients may require special configuration to connect to it. For instance, *curl*(1) requires the specification of the `--ipv4` and `--http1.0` flags.

AUTHOR

Diomidis Spinellis — <http://www.spinellis.gr>. Jef Poskanzer — <jef@mail.acme.com> — wrote micro_httpd on which this server is based.

BUGS

The possibilities for mallicious attacks through code injection and buffer overflows offered by the dynamic query option are too numerous to list. Use this feature only in setups where you restrict and control what is being sent to the server.