Snake Lib

Jon Travis

jtravis@p00p.org

The mod_snake distribution includes a variety of useful PyMods which provide features found in other popular packages, such as mod_perl. This makes it very easy for developers to pick and choose existing PyMods which provide functionality that they need without requiring them to write their own code. This bundling of standard included PyMods is called Snake Lib.

1. Setting Up Snake Lib

Snake Lib is the library of modules which ship with mod_snake. After unpacking the distribution, snake_lib is in: `mod_snake-x.y.z/snake_lib`. Snake Lib can be used regardless of its location on the system. The only requirement is that `httpd.conf` be updated to reflect the location. To add the directory to mod_snake's internals, the following line can be added:

```
SnakeModuleDir  /path/to/snake_lib
```

Adding this directive is not a requirement to use mod_snake, nor is using any of the snake_lib modules.

2. mod_snake_cgi

mod_snake_cgi is a PyMod which accelerates and caches CGIs written in Python. In addition, for added functionality, these Python CGIs loaded into mod_snake_cgi have access to the mod_snake module and its routines such as logging.

2.1. Configuration Directives

To use mod_snake_cgi, it must be first loaded into the server. To do this, the following line can be placed in `httpd.conf`:

```
SnakeModule     mod_snake_cgi.SnakeCGI
```

In order to notify mod_snake_cgi that it needs to process a given file type, a handler needs to be associated with certain files. This is commonly done in the main configuration as:

```
Alias /pycgi/ "/home/httpd/apache/pycgi/"
<Directory "/home/httpd/apache/pycgi">
    SetHandler snakecgi
    AllowOverride None
    Options ExecCGI
    Order allow,deny
    Allow from all
</Directory>
```

When requests for URI's in the form of: http://your.host.com/pycgi/foo.py are made, they will be run via mod_snake_cgi. The CGIs can be placed in `/home/httpd/apache/pycgi`.

The directive SnakeCGILogFile will send error information to an alternative file. If this directive is not specified, any output errors or tracebacks will be sent to the ErrorLog.

2.2. Inner Workings

mod_snake_cgi is a very simple PyMod. When an incoming request is made, the following actions are performed.

1. The handler is checked against 'snakecgi'. If the handler is otherwise, mod_snake_cgi returnes DECLINE.
2. mod_snake_cgi checks its internal cache to see if the CGI has been loaded already.
   a. If the CGI is not loaded, it is loaded from the disk and byte-compiled.
   b. If the CGI has been loaded before, the modification time of the on-disk version is checked and reloaded if necessary.
3. The compiled code runs in its entirety.

2.3. Future Enhancements

Potential future enhancements could include any one of the following:

- Handlers for .pyc or .so Python modules which act as CGIs.
- Seperate logging facilities within each virtual server.

3. mod_snake_epy

mod_snake_epy is a PyMod which allows Python code to be embedded within standard HTML pages. It provides extensive caching by storing the post-processed code across multiple requests. In addition, it contains the ability to store persistant data such as functions and other objects. This greatly speeds up code execution.

3.1. Configuration Directives

To load the embedded Python processor, the folowing directive can be placed in the main configuration file, `httpd.conf`:

```
SnakeModule mod_snake_epy.SnakeEPy
AddType application/x-httpd-epy .epy
```

This allows filenames with the extention '.epy' to be interpreted as embedded Python scripts. The directives which may be used after loading this module are:

- **SnakeEPy** *`on`*|*`off`* - This directive takes one flag, 'on' or 'off', and signifies whether or not processing of embedded Python code should be performed.
- **SnakeEPyErrSend** *`on`*|*`off`* - During the debugging stage of embedded Python, it is sometimes useful to send traceback information directly to the browser instead of to the logs. If this directive is 'on', tracebacks will be dumped to the browser on failure.
- **SnakeEPyLogFile** *`path`* - If logging other than the ErrorLog is required, this directive specifies a file where such information should go.

3.2. Usage

Within text/html documents, Python can be embedded in a number of ways.

- <+ ... Python code ... +> - The "plus" delimiters designate a section of Python code which should be executed *ONLY* on the first time the script is loaded. This provides a convenient place to create database connections, open logs, or import other modules.

> **Warning**
>
> If other code blocks require objects created in the "plus"delimiter, persistance will be required. See Section 3.3.1.

- <| ... Python code ... |> - The "pipe" delimiters designate a section of Python code which should be *executed*, and the resultant data discarded.
- <: ... Python code ... :> - The "colon delimiters designate a section of Python code which should be *evaluated*, and the resultant data sent to the remote client.

3.3. The EPY Object

Interaction between embedded Python code and mod_snake/Apache is done by using the EPY object. This includes things such as setting up CGI style environment, setting persistance, setting headers, and more. "EPY" is a global variable in all executed blocks. It can be used like any other object. e.g.: EPY.set_persist(1)

3.3.1. EPY Methods and Attributes

- **environ** - A dictionary containing all the key/value CGI variables. This is setup via the setup_cgi() method. cgi.py contains several functions which can take both an environment and a file object. This attribute should be passed to such functions: e.g.: cgi.FieldStorage(environ=EPY.environ,fp=EPY)
- **set_persist**(`persist`) - Set the persistance of data created and used in the code blocks. If *persist* is true, global objects created during execution will be available during subsequent requests.

- **set_header**(*header*, *value*) - Set an output header that gets sent back to the remote client, such as "content-type".
- **setup_cgi**() - Setup CGI environment variables. After calling this method, EPY.environ contains standard CGI key/value pairs.
- **write**(*data*) - write data to the remote client.

3.4. Examples

A small example for displaying the time:

```
<+
import time

EPY.set_persist(1)
HitCount = 1
+>

The current time is: <:time.ctime(time.time()):><BR>
This page has been hit <:HitCount:> times.
<|HitCount = HitCount + 1|>
```

4. mod_snake_simple

Often, creation of a full blown module takes too much time, and a developer may want to lay out a quick prototype in a few seconds. The mod_snake_simple module allows easy creation of very simple modules through simple formats, similar to that of mod_perl.

4.1. Configuration Directives

All of the handler configuration directives take one argument, a module.function pair to call when the cooresponding Apache phase is reached. The directives are: SnakePreConnection, SnakeProcessConnection, SnakePostReadRequest, SnakeTranslateName, SnakeHeaderParser, SnakeAccessChecker, SnakeCheckUserId, SnakeAuthChecker, SnakeTypeChecker, SnakeFixups, SnakeContentHandler, SnakeLogTransaction. To load the mod_snake_simple module, the following directive should be placed within the main configuration file:

```
SnakeModule mod_snake_simple.SnakeSimple
```

The other directives include:

- SnakeVarSet - Set a variable that simple modules may access. If per-directory variables are available, they are passed to the handler's per-dir argument. Per-server variables are available through a global variable called SnakeServerVars.

4.2. Usage

In the Apache configuration file and .htaccess files developers may place any of the directives to schedule calls to Python functions whenever the designated Apache phases are reached.

4.3. Examples

Mod_Snake comes with a few very simple examples in the examples directory.