FCGI C++

Преднастройка

Для компиляции кода на C++ необходимо установить библиотеку cgicc:

```
sudo apt-get install spawn-fcgi
sudo apt-get install curl
sudo apt-get install libfcgi-dev
```

```
# nano /var/www/osv/html/cgi/main.cpp

# run nginx configuration
sudo nginx -c /etc/nginx/nginx.conf

cd /var/www/osv/html/cgi/
# compile

g++ main.cpp -lfcgi++ -lfcgi -o main
```

nano /etc/nginx/sites-available/osv

```
nginx -t
```

systemctl restart nginx.service

```
spawn-fcgi -p 8000 -n  main &

curl http://192.168.137.113/main

ps

kill -15
```

```
#include <iostream>
#include "fcgio.h"

using namespace std;

int main(void)
{
  // Backup the stdio streambufs
  streambuf *cin_streambuf  = cin.rdbuf();
  streambuf *cout_streambuf = cout.rdbuf();
  streambuf *cerr_streambuf = cerr.rdbuf();

  FCGX_Request request;

  FCGX_Init();
  FCGX_InitRequest(&request, 0, 0);

  while (FCGX_Accept_r(&request) == 0)
  {
    fcgi_streambuf cin_fcgi_streambuf(request.in);
    fcgi_streambuf cout_fcgi_streambuf(request.out);
    fcgi_streambuf cerr_fcgi_streambuf(request.err);

    cin.rdbuf(&cin_fcgi_streambuf);
    cout.rdbuf(&cout_fcgi_streambuf);
    cerr.rdbuf(&cerr_fcgi_streambuf);

    cout << "Content-type: text/html\r\n"
         << "\r\n"
         << "<html>\n"
         << "  <head>\n"
         << "    <title>Hello, World!</title>\n"
         << "  </head>\n"
         << "  <body>\n"
         << "    <h1>Hello, World!</h1>\n"
         << "  </body>\n"
         << "</html>\n";

    // Note: the fcgi_streambuf destructor will auto flush
  }

  // restore stdio streambufs
  cin.rdbuf(cin_streambuf);
  cout.rdbuf(cout_streambuf);
  cerr.rdbuf(cerr_streambuf);

  return 0;
}
```

конфигурация	nano /etc/nginx/sites-available/osv
Ссылка на сайты	/etc/nginx/sites-enabled/
	ll /etc/nginx/sites-enabled/ total 0 lrwxrwxrwx 1 root root 34 Mar 29 04:49 default -> /etc/nginx/sites-available/default lrwxrwxrwx 1 root root 30 Mar 31 05:55 osv -> /etc/nginx/sites-available/osv

	root@nanopiair:/# cat /etc/nginx/sites-available/osv
	server
	{
	listen 192.168.137.113:80;
	root /var/www/osv/html;
	
	location = /
	{
	index m300s.html;
	try_files $uri $uri/ =404;
	}
	location /main
	{
	fastcgi_pass 127.0.0.1:8000;
	#cat /etc/nginx/fastcgi.conf
	fastcgi_param GATEWAY_INTERFACE CGI/1.1;
	fastcgi_param SERVER_SOFTWARE nginx;
	fastcgi_param QUERY_STRING $query_string;
	fastcgi_param REQUEST_METHOD $request_method;
	fastcgi_param CONTENT_TYPE $content_type;
	fastcgi_param CONTENT_LENGTH $content_length;
	fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
	fastcgi_param SCRIPT_NAME $fastcgi_script_name;
	fastcgi_param REQUEST_URI $request_uri;
	fastcgi_param DOCUMENT_URI $document_uri;
	fastcgi_param DOCUMENT_ROOT $document_root;
	fastcgi_param SERVER_PROTOCOL $server_protocol;
	fastcgi_param REMOTE_ADDR $remote_addr;
	fastcgi_param REMOTE_PORT $remote_port;
	fastcgi_param SERVER_ADDR $server_addr;
	fastcgi_param SERVER_PORT $server_port;
	fastcgi_param SERVER_NAME $server_name;
	}
	location /fastrun
	{
	index fastrun.html;
	try_files $uri $uri/fastrun/ =404;
	}
	location /programs
	{
	index programs.html;
	try_files $uri $uri/programs/ =404;
	}
	location /recipe
	{
	index recipe.html;
	try_files $uri $uri/recipe/ =404;
	}
	}

Running the code	``` # run nginx using the provided configuration sudo nginx -c /etc/nginx/nginx.conf # compile hello_world g++ main.cpp -lfcgi++ -lfcgi -o main # spawn the fcgi app on port 8000 with no fork spawn-fcgi -p 8000 -n main spawn-fcgi -p 8000 -n /var/www/osv/html/cgi/main ```

kill process	ps ps -ef | less pidof main 2496
	kill -l kill -15 2496
расположение скриптов и страниц	root@nanopiair:/# l /var/www/osv/html/cgi/ total 36 -rw-r--r-- 1 root root 400 Apr 4 11:12 blue.cpp -rw-r--r-- 1 root root 401 Apr 4 11:11 green.cpp -rw-r--r-- 1 root root 1136 Apr 5 04:24 main.cpp -rw-r--r-- 1 root root 399 Apr 4 11:11 red.cpp -rw-r--r-- 1 root root 401 Apr 4 11:13 white.cpp -rw-r--r-- 1 root root 402 Apr 4 11:12 yellow.cpp
cd /var/www/osv/html/cgi/	g++ main.cpp -lfcgi++ -lfcgi -o main g++ blue.cpp -lfcgi++ -lfcgi -o blue g++ green.cpp -lfcgi++ -lfcgi -o green g++ red.cpp -lfcgi++ -lfcgi -o red g++ white.cpp -lfcgi++ -lfcgi -o white g++ yellow.cpp -lfcgi++ -lfcgi -o yellow

nano /etc/nginx/sites-available/osv
что добавить
	location /blue
	{
	fastcgi_pass 127.0.0.1:8001;
	}
	location /green
	{
	fastcgi_pass 127.0.0.1:8002;
	}
	location /red
	{
	fastcgi_pass 127.0.0.1:8003;
	}
	location /white
	{
	fastcgi_pass 127.0.0.1:8004;
	}
	location /yellow
	{
	fastcgi_pass 127.0.0.1:8005;
	}

nano /etc/nginx/sites-available/osv
все
	root@nanopiair:/var/www/osv/html/cgi# cat /etc/nginx/sites-available/osv
	server
	{
	listen 192.168.137.55:80;
	root /var/www/osv/html;
	
	location = /
	{
	index m300s.html;
	try_files $uri $uri/ =404;
	}
	
	location /main
	{
	fastcgi_pass 127.0.0.1:8000;
	}
	
	location /blue
	{

	fastcgi_pass 127.0.0.1:8001;
	}
	location /green
	{
	fastcgi_pass 127.0.0.1:8002;
	}
	location /red
	{
	fastcgi_pass 127.0.0.1:8003;
	}
	location /white
	{
	fastcgi_pass 127.0.0.1:8004;
	}
	location /yellow
	{
	fastcgi_pass 127.0.0.1:8005;
	}
	
	location /fastrun
	{
	index fastrun.html;
	try_files $uri $uri/fastrun/ =404;
	}
	location /programs
	{
	index programs.html;
	try_files $uri $uri/programs/ =404;
	}
	location /recipe
	{
	index recipe.html;
	try_files $uri $uri/recipe/ =404;
	}
	}

nginx -t systemctl restart nginx.service	spawn-fcgi -p 8000 -n main & spawn-fcgi -p 8001 -n blue & spawn-fcgi -p 8002 -n green & spawn-fcgi -p 8003 -n red & spawn-fcgi -p 8004 -n white & spawn-fcgi -p 8005 -n yellow & curl http://192.168.137.55/main curl http://192.168.137.55/blue curl http://192.168.137.55/green curl http://192.168.137.55/red curl http://192.168.137.55/white curl http://192.168.137.55/yellow

nano /etc/nginx/sites-available/osv
 	
nginx -t
 	
systemctl restart nginx.service
	server
	{
	listen 192.168.137.113:80;
	root /var/www/osv/html;
	location = / {
	index m300s.html;
	try_files $uri $uri/ =404;
	}
	#cgi
	location /main
	{
	fastcgi_pass 127.0.0.1:8000;
	fastcgi_param GATEWAY_INTERFACE CGI/1.1;
	fastcgi_param SERVER_SOFTWARE nginx;
	fastcgi_param QUERY_STRING $query_string;
	fastcgi_param REQUEST_METHOD $request_method;
	fastcgi_param CONTENT_TYPE $content_type;
	fastcgi_param CONTENT_LENGTH $content_length;
	fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
	fastcgi_param SCRIPT_NAME $fastcgi_script_name;
	fastcgi_param REQUEST_URI $request_uri;

	fastcgi_param DOCUMENT_URI $document_uri;
	fastcgi_param DOCUMENT_ROOT $document_root;
	fastcgi_param SERVER_PROTOCOL $server_protocol;
	fastcgi_param REMOTE_ADDR $remote_addr;
	fastcgi_param REMOTE_PORT $remote_port;
	fastcgi_param SERVER_ADDR $server_addr;
	fastcgi_param SERVER_PORT $server_port;
	fastcgi_param SERVER_NAME $server_name;
	}
	location /fastrun {
	index fastrun.html;
	try_files $uri $uri/fastrun/ =404;
	}
	location /programs {
	index programs.html;
	try_files $uri $uri/programs/ =404;
	}
	location /recipe {
	index recipe.html;
	try_files $uri $uri/recipe/ =404;
	}
	}

	spawn-fcgi -p 8000 -n /var/www/osv/html/cgi/main &
	Проверяем страницу curl http://192.168.137.113/main:8000
	root@nanopiair:/# spawn-fcgi -p 8000 -n /var/www/osv/html/cgi/main & [1] 2669 root@nanopiair:/# curl http://192.168.137.113/main:8000 <html> <head> <title>Hello, World!</title> </head> <body> Hello, World! </body> </html>

	cd /var/www/osv/html/cgi/

blue.cpp
green.cpp
red.cpp
white.cpp
yellow.cpp
	#include <iostream>
	#include "fcgio.h"
	#include <string>
	#include <stdio.h>
	#include <stdlib.h>
	
	using namespace std;
	//----------
	void set_content_type(string content_type)
	{
	cout << "Content-type: " << content_type << "\r\n\r\n";
	}
	//----------
	int main(void)
	{
	// Backup the stdio streambufs
	streambuf *cin_streambuf = cin.rdbuf();
	streambuf *cout_streambuf = cout.rdbuf();
	streambuf *cerr_streambuf = cerr.rdbuf();
	
	FCGX_Request request;
	
	FCGX_Init();
	FCGX_InitRequest(&request, 0, 0);
	
	while (FCGX_Accept_r(&request) == 0)
	{
	fcgi_streambuf cin_fcgi_streambuf(request.in);
	fcgi_streambuf cout_fcgi_streambuf(request.out);
	fcgi_streambuf cerr_fcgi_streambuf(request.err);
	
	cin.rdbuf(&cin_fcgi_streambuf);
	cout.rdbuf(&cout_fcgi_streambuf);
	cerr.rdbuf(&cerr_fcgi_streambuf);
	
	set_content_type("text/html");
	cout << "{blue : " << rand() << "}\n";
	cout << "{green : " << rand() << "}\n";
	cout << "{red : " << rand() << "}\n";
	cout << "{white : " << rand() << "}\n";
	cout << "{yellow : " << rand() << "}\n";
	
	// Note: the fcgi_streambuf destructor will auto flush
	}
	
	// restore stdio streambufs
	cin.rdbuf(cin_streambuf);
	cout.rdbuf(cout_streambuf);
	cerr.rdbuf(cerr_streambuf);

	return 0; }