Задание 39

Конфигурация вёб-сервера.

В у каждого человека из 11 класса в домашней директории создана директория **public_html**, а в ней поддиректория **cgi-bin**. Чтобы получить доступ к файлам в директории **public_html**, надо зайти по адресу **http://sc11???.programming1189.ru/** (??? надо заменить на на ваше ФИО). В итоге отобразится список файлов в корневой директории сайта **/home/sc11???/public_html/** . Если разместить файлы с разрешением .html в этой директории, то их можно будет открыть браузером, как и любую другую страницу в интернете. Если в какой-нибудь директории мы разместим файл index.html, то он будет отдан браузером вместо списка файлов в данной директории. Именно поэтому когда вы открываете http://yandex.ru/ , на самом деле открывается на корневая директория сайта, а файл http://yandex.ru/index.html (хотя на самом деле всё немного сложнее).

Кроме того, у каждого из вас в домашней директории появились файлы **access.log** и **error.log**. В первый будет записан журнал доступа к вашему сайту, во второй ошибки вёб-сервера и ошибки при работе ваших скриптов. Удалить эти файлы нельзя, так как вы не являетесь их владельцем. Посмотреть последние 10 строк этих файлов можно так:

```
tail access.log
tail error.log
```

Пример размещения HTML-страницы на сервере.

Создадим в директории **/home/sc11???/public_html/** файл **example.html** следующего содержания:

```
<!DOCTYPE HTML>
<html>
        <head>
                <meta charset='koi8-r'/>
                <title> Привет, мир! </title>
        </head>
        <body>
                <h1> Привет, я заголовок!!! </h1>
                А я просто текст.
        </body>
</html>
```

Тогда он станет доступен по ссылке **http://sc11???.programming1189.ru/example.html**

Обратите внимания, что для символов русского языка надо указать кодировку. В данном случае указана кодировка **koi8-r**, так как она является кодировкой по умолчанию на нашем сервере и именно в ней будет сохранена html-страница, если вы редактируете её с помощью текстового редактора на сервере.

Запуск скриптов CGI

На данный момент нас больше интересует директория **/home/sc11???/public_html/cgi-bin** . **cgi-bin** – стандартное название директории с CGI-скриптами, располагается она обычно (как и в нашем случае) в корневой директории сайта. Вёб-сервер настроен так, что он не может запускать скрипты на питоне из других директорий. Чтобы запускать CGI-скрипты, нужно создать файл со скриптом в директории **cgi-bin** и затем сделать его исполняемым следующей командой:

```
chmod +x ~/public_html/cgi-bin/script.py
```

Символ ~ при работе с командной строкой будет заменён на вашу домашнюю директорию.

Надо сказать, что в Unix, в отличие от Windows, исполняемыми могут быть файлы с любым расширением, а не только .exe, .com. .bat. При этом первая строка файла должна иметь вид

#!путь

где указан путь к той программ-интерпретатору, которой запускаемый файл будет передан как аргумент командной строки. Например, если у вас есть файл вида

```
#!/usr/bin/python3

print('Content-type: text/html')
print()
print("<html><body>Hello!</body></html>")
```

и вы запускаете его, набрав в командной строке **./script.py**, то это эквивалентно запуску команды **/usr/bin/python3 script.py** .

Пример CGI-скрипта

```
#!/usr/bin/python3
# -*- encoding: koi8-r -*-
import sys
import codecs
sys.stdout = codecs.getwriter('koi8-r')(sys.stdout.detach())
import cgitb
cgitb.enable()
print('Content-type: text/html')
print()

print('''
<!DOCTYPE HTML>
<html>
        <head>
                <meta charset='koi8-r'/>
                <title> Привет, мир! </title>
        </head>
        <body>
                <h1> Привет, я заголовок!!! </h1>
                А я просто текст. <br>
                И я тоже текст.
        </body>
</html>
''')
```

Обратите внимание на блок из семи строк в начале файле. С первой из них мы уже знакомы, она обеспечивает запуск скрипта без явного указания имени интерпретатора. В следующей строке

-*- encoding: koi8-r -*-

мы задаём кодировку файла. Как уже было сказано, koi8-r – основная кодировка сервера, указываем её. В случае использования в скрипте русских букв кодировку указывать обязательно, иначе Python3 выдаст ошибку.

Затем в следующих двух строках мы импортируем модули codecs и sys, для того, чтобы строкой ниже преобразовать поток вывода к кодировке koi8-r:

sys.stdout = codecs.getwriter('koi8-r')(sys.stdout.detach())

Выражение sys.stdout.detach() переведёт поток вывода из текстового режима с кодировкой ASCII в бинарный режим (вывод байтов, а не символов), затем мы снова преобразуем его в текстовый поток с помощью применения функции codecs.getwriter('koi8-r'). Да, я не ошибся, мы вызываем как функцию результат вызова функции, вспоминаем про "функции высшего порядка". В любом случае, понимать всю внутреннюю механику в данном случае не так важно, можно просто скопировать эти 7 строк к себе.

Две следующие строки делают очень важную вещь: включают вывод информации о произошедшей в скрипте ошибке непосредственно на html-страницу. Если не использовать модуль cgitb и не сделать cgitb.enable(), вся информация об ошибках будет валиться в файл error.log в вашей домашней директории и просматривать их с помощью текстового редактора крайне неудобно. Так что, не пренебрегайте этими строками (но после отладки эти строки следует отключить из соображений безопасности, чтобы пользователи вашей страницы не могли увидеть структуру скрипта в сообщениях об ошибках).

Вывод content-type и пустая строка обсуждаются в следующей секции.

В целом, всякий CGI-скрипт, который вы пишете для выполнения заданий по CGI-скриптам, должен начинаться со следующих строк:

```
#!/usr/bin/python3
# -*- encoding: koi8-r -*-
import sys
import codecs
sys.stdout = codecs.getwriter('koi8-r')(sys.stdout.detach())
import cgitb
cgitb.enable()
print('Content-type: text/html')
print()
```

Content-type

CGI-скрипты запускаются на выполнение тогда, когда браузер клиента запрашивает соответствующую страницу с вашего сайта. Например, если пользователь попробует загрузить страницу

http://sc11???.programming1189.ru/cgi-bin/hello.py

, то вёб-сервер запустит соответствующий ему скрипт

/home/sc11???/public_html/cgi-bin/hello.py

, и будет отправлять браузеру весь вывод данного скрипта. При этом в выводе ваших скриптов обязательно должен содержаться заголовок Content-type, который обычно будет показывать, что вы отдаёте html-страницу:

```
Content-type: text/html

```

Не забывайте про пустую строку после окончания заголовков, иначе вёб-сервер решит, что ваша страница сгенерирована с ошибкой, и ничего не пошлёт клиенту. После этого необходимо вывести на поток стандартного вывода саму html-страницу.

Кроме типа содержимого text/html, иногда в целях отладки бывает полезным вывести страницу или какую-то информацию просто в текстовом виде, без применения правил оформления html. В таком случае следует задать тип содержимого text/plain:

```
Content-type: text/plain

```

Отправка данных на сервер с помощью метода GET

Стандарт html поддерживает отправку параметров к определённой странице с помощью GET-запроса. Если, например, мы введём в Яндексе поисковый запрос "Hello, world", то увидим, что в адресной строке загружена страница

http://yandex.ru/yandsearch?**text=hello%2C+world&lr=213**

На самом деле, была загружена страница

http://yandex.ru/yandsearch

После знака '?' идут строки в формате 'параметр=значение', разделённые символом '&'. Как видим, введённый поисковый запрос передан на сервер в параметре с именем text, и есть некоторый парамерт lr со значением '213' (под этим числом Яндекс понимает наше географическое положение, и 213 соответствует Москве).

Чтобы отправить некоторые данные на сервер с помощью GET-запроса, нужно создать страницу и воспользоваться тегами <form> и <input>:

```
<html>
        <head>
                <title> Add task </title>
        </head>
        <body>
                <form method='GET' action='cgi-bin/echo_env.py'>
                        Task name: <input type='text' name='task_name'/> <br>
                        <input type='checkbox' name='urgent'/> Urgent <br>
                        Description: <input type='text' name='description'/> <br>
                        <input type='submit' name='submit' value='SEND'/><br>
                </form>
        </body>
</html>
```

Как видно у тега <form> здесь указаны два параметра: method='GET' указывает, что данные формы будут посланы на сервер в параметрах GET-запроса, action='cgi-bin/echo_env.py' указывает, на какую именно страницу должен делаться этот GET-запрос. Внутри формы размещены четыре элемента ввода: два текстовых, один типа "checkbox" (квадратик с галочкой, имеет значение "off" если выключен и "on", если включён) и кнопка для отправки формы типа "submit".

Как мы можем узнать значения этих параметров из своих CGI-скриптов? Всё просто. Сервер сообщает вашим скриптам информацию о себе, операционной системе, в которой выполняется скрипт, и параметрах выполнения скрипта в так называемых "переменных окружения". Вывод всех переменных окружения CGI-скрипта можно посмотреть здесь: http://testuser.programming1189.ru/cgi-bin/echo_env.py.

Вот код этого примера:

```
#!/usr/bin/python3
# -*- encoding: koi8-r -*-
import codecs
import sys
sys.stdout = codecs.getwriter('koi8-r')(sys.stdout.detach())
print('Content-type: text/html')
print()

import os

print('<html><head><title>Environment variables</title></head><body>')
for key, value in os.environ.items():
    print(key, '=>', value, '<br>')
    print('</body></html>')
```

Заполнив на странице http://testuser.programming1189.ru/add_task.html форму и отправив её, вы увидите, что параметры запроса хранятся в переменной QUERY_STRING, значение которой в питоне даётся выражением os.environ['QUERY_STRING'].

Задание 39

Нужно написать простейший вариант твиттера, когда пользователь имеет возможность отправлять на страницу новые сообщения (у каждого сообщения должен быть заголовок и тело сообщения) и просматривать список сообщений.

Примечаение. Так как после выполнения скрипта вся информация о запросе теряется, каждую новую запись в "твиттере" следует добавлять в файл со список записей.

Примечаение. Вообще говоря, идеологически не совсем правильно использовать метод GET для изменения информации на сервере, для этого лучше использовать метод POST. Но мы пока не знаем как с ним работать и воспользуемся GET'ом.

Примечание 2. Если возникнут проблемы с отображением русских букв, можно использовать только английский алфавит. С раскодированием пробелов, знаков препинания и русских букв разберёмся в следующий раз.

Ссылки

Самоучитель HTML: http://htmlbook.ru/samhtml
Справочник HTML: http://htmlbook.ru/html
Базовые описания HTML, HTTP и CGI на википедии:
CGI: http://ru.wikipedia.org/wiki/CGI
HTML: http://ru.wikipedia.org/wiki/HTML
HTTP: http://ru.wikipedia.org/wiki/HTTP