МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

НАБЕРЕЖНОЧЕЛНИНСКИЙ ИНСТИТУТ

ПРОТОКОЛ HTTP И ИНТЕРФЕЙС CGI

Учебно-методическое пособие
по дисциплине
«ВЕБ-ПРОГРАММИРОВАНИЕ»

Набережные Челны
2018

Галиуллин Л.А. Протокол HTTP и интерфейс CGI: учебно-методическое пособие по дисциплине «Веб-программирование» [Электронный ресурс] / Казанский федеральный университет, Электронный архив, 2018.

Рассматривается протокол HTTP и интерфейс CGI. Представлены структура HTTP-запроса, формат HTTP-ответа, спецификации CGI, передача параметров серверу. Приведены контрольные вопросы. Для студентов направлений подготовки «Информатика и вычислительная техника», «Программная инженерия».

Введение

Как и большинство акронимов, Common Gateway Interface (CGI - общий шлюзовой интерфейс) мало что говорит по сути. Интерфейс с чем? Где этот шлюз? О какой общности речь? Чтобы ответить на эти вопросы, вернемся назад и бросим взгляд на WWW в целом.

Тим Бернерс-Ли, физик, работавший в CERN, придумал Web в 1990 году, хотя план возник еще в 1988. Идея состояла в том, чтобы дать возможность легко и быстро обмениваться мультимедийными данными - текстом, изображениями и звуком - через Интернет. WWW состояла из трех основных частей: HTML, URL и HTTP. HTML - язык форматирования, используемый для представления содержания в Web. URL - это адрес, используемый для получения содержимого в формате HTML (или каком-либо ином) с веб-сервера. HTTP - это язык, который понятен веб-серверу и позволяет клиентам запрашивать у сервера документы.

Протокол HTTP

Работа по протоколу HTTP происходит следующим образом: программа-клиент устанавливает TCP-соединение с сервером (стандартный номер порта-80) и выдает ему HTTP-запрос. Сервер обрабатывает этот запрос и выдает HTTP-ответ клиенту.

Структура HTTP-запроса

HTTP-запрос состоит из заголовка запроса и тела запроса, разделенных пустой строкой. Тело запроса может отсутствовать. Заголовок запроса состоит из главной (первой) строки запроса и последующих строк, уточняющих запрос в главной строке. Последующие строки также могут отсутствовать. Запрос в главной строке состоит из трех частей, разделенных пробелами:

1. *Метод* (иначе говоря, команда HTTP):

- GET - запрос документа. Наиболее часто употребляемый метод; в HTTP/0.9, говорят, он был единственным.

- HEAD - запрос заголовка документа. Отличается от GET тем, что выдается только заголовок запроса с информацией о документе. Сам документ не выдается.
- POST - этот метод применяется для передачи данных CGI-скриптам. Сами данные следуют в последующих строках запроса в виде параметров.
- PUT - разместить документ на сервере. Используется редко. Запрос с этим методом имеет тело, в котором передается сам документ.

2. *Ресурс* - это путь к определенному файлу на сервере, который клиент хочет получить (или разместить - для метода PUT). Если ресурс - просто какой-либо файл для считывания, сервер должен по этому запросу выдать его в теле ответа. Если же это путь к какому-либо CGI-скрипту, то сервер запускает скрипт и возвращает результат его выполнения. Кстати, благодаря такой унификации ресурсов для клиента практически безразлично, что он представляет собой на сервере.
3. *Версия протокола* - версия протокола HTTP, с которой работает клиентская программа.

Таким образом, простейший HTTP-запрос может выглядеть следующим образом:

GET / HTTP/1.0 - запрашивается корневой файл из корневой директории web-сервера.

Строки после главной строки запроса имеют следующий формат: *Параметр: значение*.

Таким образом задаются параметры запроса. Это является необязательным, все строки после главной строки запроса могут отсутствовать; в этом случае сервер принимает их значение по умолчанию или по результатам предыдущего запроса (при работе в режиме Keep-Alive).

Перечислим некоторые наиболее употребительные параметры HTTP-запроса:

- Connection (соединение)- может принимать значения Keep-Alive и close.
- Keep-Alive ("оставить в живых") означает, что после выдачи данного документа соединение с сервером не разрывается, и

можно выдавать еще запросы. Большинство браузеров работают именно в режиме Keep-Alive, так как он позволяет за одно соединение с сервером "скачать" html-страницу и рисунки к ней. Будучи однажды установленным, режим Keep-Alive сохраняется до первой ошибки или до явного указания в очередном запросе Connection: close.

- close ("закрыть") - соединение закрывается после ответа на данный запрос.
- User-Agent - значением является "кодовое обозначение" браузера, например: *Mozilla/4.0 (compatible; MSIE 5.0; Windows 95; DigExt)*
- Accept - список поддерживаемых браузером типов содержимого в порядке их предпочтения данным браузером, например, для IE5*: Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-excel, application/msword, application/vnd.ms-powerpoint, */**. Значение этого параметра используется в основном CGI-скриптами для формирования ответа, адаптированного для данного браузера.
- Referer - URL, с которого перешли на этот ресурс.
- Host - имя хоста, с которого запрашивается ресурс. Полезно, если на сервере имеется несколько виртуальных серверов под одним IP-адресом. В этом случае имя виртуального сервера определяется по этому полю.
- Accept-Language - поддерживаемый язык. Имеет значение для сервера, который может выдавать один и тот же документ в разных языковых версиях.

Формат HTTP-ответа

Формат ответа очень похож на формат запроса: он также имеет заголовок и тело, разделенное пустой строкой. Заголовок также состоит из основной строки и строк параметров, но формат основной строки отличается от таковой в заголовке запроса. Основная строка запроса состоит из 3-х полей, разделенных пробелами:

- Версия протокола - аналогичен соответствующему

параметру запроса.

- Код ошибки - кодовое обозначение "успешности" выполнения запроса. Код 200 означает "все нормально" (OK).
- Словесное описание ошибки - "расшифровка" предыдущего кода. Например, для 200 это OK, для 500 - Internal Server Error.

Наиболее употребительные параметры http-ответа:

- Connection - аналогичен соответствующему параметру запроса. Если сервер не поддерживает Keep-Alive (есть и такие), то значение Connection в ответе всегда close.
- Content-Type ("тип содержимого") - содержит обозначение типа содержимого ответа.
- В зависимости от значения Content-Type браузер воспринимает ответ как HTML-страницу, картинку gif или jpeg, как файл, который надо сохранить на диске, или как что-либо еще и предпринимает соответствующие действия. Значение Content-Type для браузера аналогично значению расширения файла для Windows.

Некоторые типы содержимого:

- text/html - текст в формате HTML (веб-страница);
- text/plain - простой текст (аналогичен "блокнотовскому");
- image/jpeg - картинка в формате JPEG;
- image/gif - то же, в формате GIF;
- application/octet-stream - поток "октетов" (т.е. просто байт) для записи на диск.

- Content-Length ("длина содержимого") - длина содержимого ответа в байтах.
- Last-Modified ("Модифицирован в последний раз") - дата последнего изменения документа.

Возможность пересылки через Интернет информации всех типов явилась революцией, но вскоре была обнаружена и другая возможность. Если можно переслать через Web любой текст, то почему нельзя переслать текст, созданный программой, а не взятый из готового файла? При этом открывается море возможностей. Простой пример: можно использовать программу, выводящую текущее время, так, чтобы читатель

видел правильное время при каждом просмотре страницы. Несколько умных голов в National Center for Supercomputing Applications (Национальный центр разработки приложений для суперкомпьютеров - NCSA), которые создавали веб-сервер, такую возможность увидели, и вскоре появился CGI.

CGI - это набор правил, согласно которым программы на сервере могут через веб-сервер посылать данные клиентам. Спецификация CGI сопровождалась изменениями в HTML и HTTP, вводившими новую характеристику, известную как формы.

Если CGI позволяет программам посылать данные клиенту, то формы расширяют эту возможность, позволяя клиенту посылать данные для этой CGI-программы. Распространенные приложения CGI включают в себя:

- Динамический HTML. Целые сайты могут генерироваться одной CGI-программой.
- Поисковые механизмы, находящие документы с заданными пользователем словами.
- Гостевые книги и доски объявлений, в которые пользователи могут добавлять свои сообщения.
- Бланки заказов.
- Анкеты.
- Извлечение информации из размещенной на сервере базы данных.

Все они дают возможность соединения CGI с базой данных, что нас особенно интересует.

Спецификация CGI

Итак, что в точности представляет собой «набор правил», позволяющий CGI-программе, скажем, в Батавии, штат Иллинойс, обмениваться данными с веб-броузером во Внешней Монголии? Официальную спецификацию CGI наряду с массой других сведений о CGI можно найти на сервере NCSA по адресу http://hoohoo.ncsa.uiuc.edu/cgi/.

Есть четыре способа, которыми CGI передает данные между CGI-программой и веб-сервером, а следовательно, и клиентом Web:

- Переменные окружения.
- Командная строка.
- Стандартное устройство ввода.
- Стандартное устройство вывода.

С помощью этих четырех методов сервер пересылает все данные, переданные клиентом, CGI-программе. Затем CGI-программа делает свое волшебное дело и пересылает выходные данные обратно серверу, который переправляет их клиенту.

Эти данные приводятся с прикидкой на сервер HTTP Apache. Apache - наиболее распространенный веб-сервер, работающий практически на любой платформе, включая Windows 9x и Windows NT. Однако они могут быть применимы ко всем HTTP-серверам, поддерживающим CGI. Некоторые патентованные серверы, например, от Microsoft и Netscape, могут иметь дополнительные функции или работать несколько иначе. Поскольку лицо Web продолжает изменяться с невероятной скоростью, стандарты все еще развиваются, и в будущем, несомненно, произойдут изменения. Однако, технология CGI представляется устоявшейся - расплачиваться за это приходится тем, что другие технологии, такие как апплеты, ее потеснили. Все CGI-программы, которые вы напишете, используя эти сведения, почти наверное смогут работать еще долгие годы на большинстве веб-серверов.

Когда CGI-программа вызывается посредством формы - наиболее распространенного интерфейса, броузер передает серверу длинную строку, в начале которой стоит путь к CGI-программе и ее имя. Затем следуют различные другие данные, которые называются информацией пути и передаются CGI-программе через переменную окружения PATH_INFO (табл. 2-1). После информации пути следует символ «?», а за ним - данные формы, которые посылаются серверу с помощью метода HTTP GET. Эти данные становятся доступными CGI-программе через переменную окружения QUERY_STRING. Любые данные, которые страница посылает с использованием метода HTTP POST, который используется чаще всего, будут переданы CGI-программе через стандартное устройство ввода. Типичная строка, которую может получить сервер от браузера, показана в табл. 3-1. Программа с именем formread в каталоге cgi-bin

вызывается сервером с дополнительной информацией пути extra/information и данными запроса choice=help - по-видимому, как часть исходного URL. Наконец, данные самой формы (текст «CGI programming» в поле «keywords») пересылаются через метод HTTP POST.

Таблица 1. Части строки, переданной броузером серверу

http://www.myserver.com/cgi-bin	/formread	/extra/information	?choice=help
	название программы	информация о пути	строка запроса

Переменные окружения

Когда сервер выполняет CGI-программу, то прежде всего передает ей некоторые данные для работы в виде переменных окружения. В спецификации официально определены семнадцать переменных, но неофициально используется значительно больше - с помощью описываемого ниже механизма, называемого HTTP_mechanism. CGI-программа имеет доступ к этим переменным так же, как и к любым переменным среды командного процессора при запуске из командной строки. В сценарии командного процессора, например, к переменной окружения FOO можно обращаться как $FOO; в Perl это обращение выглядит, как $ENV{'FOO'}; в C - getenv("FOO"); и т. д. В таблице 2 перечислены переменные, которые всегда устанавливаются сервером - хотя бы и в значение null. Помимо этих переменных данные, возвращаемые клиентом в заголовке запроса, присваиваются переменным вида HTTP_FOO, где FOO - имя заголовка. Например, большинство веб-броузеров включает данные о версии в заголовок с именем USER_AGENT . Ваша CGI-программа может получить эти данные из переменной HTTP_USER_AGENT.

Таблица 2. Переменные окружения CGI

Переменная окружения	**Описание**
CONTENT LENGTH	Длина данных, переданных методами POST или PUT, в байтах
CONTENT_TYPE	Тип MIME данных, присоединенных с помощью методов POST или PUT.
GATEWAY_INTERFACE	Номер версии спецификации CGI, поддерживаемой сервером.
PATH_INFO	Дополнительная информация пути, переданная клиентом. Например, для запроса http://www.myserver.com/test.cgi/this/is/a/ path?field=green значением переменной PATH_INFO будет /this/is/a/path.
PATH_TRANSLATED	То же, что PATH_INFO, но сервер производит всю возможную трансляцию, например, расширение имен типа «~account».
QUERY_STRING	Все данные, следующие за символом «?» в URL. Это также данные, передаваемые, когда REQUEST_METOD формы есть GET.
REMOTE_ADDR	IP-адрес клиента, делающего запрос.
REMOTE_HOST	Имя узла машины клиента, если оно доступно.
REMOTE_IDENT	Если веб-сервер и клиент поддерживают идентификацию типа identd, то это имя пользователя учетной записи, которая делает запрос.
REQUEST_METHOD	Метод, используемый клиентом для запроса. Для CGI-программ, которые мы собираемся создавать, это обычно будет POST или GET.
SCRIPT_NAME	Путь к выполняемому сценарию,

	указанный клиентом. Может использоваться при ссылке URL на самого себя, и для того, чтобы сценарии, ссылки на которые существуют в разных местах, могли выполняться по-разному в зависимости от места.
SERVER_NAME	Имя узла - или IP-адрес, если имя недоступно, машины, на которой выполняется веб-сервер.
SERVER_PORT	Номер порта, используемого веб-сервером.
SERVER_PROTOCOL	Протокол, используемый клиентом для связи с сервером. В нашем случае этот протокол почти всегда HTTP.
SERVER_SOFTWARE	Данные о версии веб-сервера, выполняющего CGI-программу.

Приведем пример сценария CGI на Perl, который выводит все переменные окружения, установленные сервером, а также все унаследованные переменные, установленные командным процессором, запустившим сервер.

Листинг 2.1. *Вывод значений переменных окружения.*

```
print "Content-Type: text/html\n\n
<HTML><HEAD><TITLE></title></head><BODY>\n
<p>Переменные окружения:<p>\n";
foreach (keys %ENV) {print "$_: $ENV{$_}<br>\n" }
print "</body></html>";
```

Все эти переменные могут быть использованы и даже изменены вашей CGI-программой. Однако эти изменения не затрагивают веб-сервер, запустивший программу.

Передача параметров серверу. Командная строка

CGI допускает передачу CGI-программе аргументов в качестве параметров командной строки, которая редко используется. Редко используется она потому, что практические

применения ее немногочисленны, и мы не будем останавливаться на ней подробно. Суть в том, что если переменная окружения QUERY_STRING не содержит символа « = », то CGI-программа будет выполняться с параметрами командной строки, взятыми из OUERY_STRING . Например, http://www.myseruer.com/cgi-bin/finger?root запустит finger root на www.myserver.com.

Параметры командной строки чаще всего используются вместе с тегом HTML <ISINDEX> . Тег <ISINDEX> обозначает миниформу, содержащуюся в одном теге. Обнаружив тег <ISINDEX> , броузер выводит окно, в которое пользователь может ввести текст запроса. При подаче запроса (нажатии пользователем клавиши «Enter»), броузер извлекает URL из тега <ISINDEX> и обращается к нему, передавая текст запроса в качестве командной строки.

Предшествующий finger можно написать так, что при вызове без аргументов он выведет HTML-страницу с тегом <ISINDEX> . После ввода пользователем адреса finger исполнится так же, как описано.

Стандартное устройство ввода

Как сказано выше, если клиент использует для передачи информации HTTP-методы PUT или POST, длина и тип MIME этих данных помещаются в переменные CONTENT_LENGTH и CONTENT_TYPE соответственно. Передаваемые данные посылаются на стандартное устройство ввода CGI-программы. Признак конца данных может не посылаться программе, поэтому она должна взять значение переменной CONTENT_LENGTH и прочесть столько байтов, сколько в ней указано. Это основной метод передачи данных из форм, и в наших примерах мы будем почти исключительно использовать только его.

Существуют многочисленные библиотеки почти для всех языков, которые выполняют важные задачи настройки CGI-программ, в том числе определяют, каким методом - GET или POST — переданы данные, и, соответственно, разбирают переменную окружения QUERY_STRING или читают с

устройства стандартного ввода. Затем эти библиотеки помещают данные в легко доступные переменные. Обширный список ресурсов CGI для разных языков есть на Yahoo по адресу: http://www.yahoo.com/Computers_and_Internet/Internet/World_Wide_Web/CGI_Common_Gateway_Interface/

Стандартное устройство вывода

Данные, посылаемые CGI-программой на стандартное устройство вывода, читаются веб-сервером и отправляются клиенту. Если имя сценария начинается с nph-, то данные посылаются прямо клиенту без вмешательства со стороны веб-сервера. В этом случае CGI-программа должна сформировать правильный заголовок HTTP, который будет понятен клиенту. В противном случае предоставьте веб-серверу сформировать HTTP-заголовок.

Даже если вы не используете nph-сценарий, серверу нужно дать одну директиву, которая сообщит ему сведения о вашей выдаче. Обычно это HTTP-заголовок Content-Type , но может быть и заголовок Location . За заголовком должна следовать пустая строка, то есть перевод строки или комбинация CR/LF.

Заголовок Content-Type сообщает серверу, какого типа данные выдает ваша CGI-программа. Если это страница HTML, то строка должна быть Content-Type: text/html. Заголовок Location сообщает серверу другой URL или другой путь на том же сервере, куда нужно направить клиента. Заголовок должен иметь следующий вид: Location: http:// www.myserver.com/another/place/.

Важные особенности сценариев CGI

Вы уже знаете, в основном, как работает CGI. Клиент посылает данные, обычно с помощью формы, веб-серверу. Сервер выполняет CGI-программу, передавая ей данные. CGI-программа осуществляет свою обработку и возвращает свои выходные данные серверу, который передает их клиенту. Теперь от понимания того, как работают CGI-программа, нужно

перейти к пониманию того, почему они так широко используются..

Контрольные вопросы

1. Что вы знаете о протоколе HTTP?
2. Что вы знаете о структуре HTTP-запроса?
3. Что вы знаете о формате HTTP-ответа?
4. Что вы знаете о спецификации CGI?
5. Что вы знаете о переменных окружения?
6. Что вы знаете о передаче параметров серверу?
7. Что вы знаете о стандартном устройстве ввода?
8. Что вы знаете о стандартном устройстве вывода?
9. Что вы знаете об особенностях сценариев CGI?
10. Что вы знаете о запоминании состояния?

Рекомендуемые источники

1. Колдаев В.Д. Основы алгоритмизации и программирования: Учебное пособие / В.Д. Колдаев; Под ред. Л.Г. Гагариной. - М.: ИД ФОРУМ: ИНФРА-М, 2015. - 416 с. [Электронный ресурс]. http://znanium.com/bookread.php?book=336649.
2. Гагарина Л.Г. Технология разработки программного обеспечения: Учеб. пос. / Л.Г.Гагарина, Е.В.Кокорева, Б.Д.Виснадул; Под ред. проф. Л.Г.Гагариной - М.: ИД ФОРУМ: НИЦ Инфра-М, 2017. - 400 с. [Электронный ресурс]. http://znanium.com/bookread.php?book=389963.
3. Голицына О. Л. Программирование на языках высокого уровня: Учебное пособие / О.Л. Голицына, И.И. Попов. - М.: Форум, 2016. - 496 с. [Электронный ресурс]. http://znanium.com/bookread.php?book=139428.

. …