

Владивостокский государственный университет  
экономики и сервиса  
Институт информатики инноваций и бизнес  
систем

**Предмет:**  
**«Технологии Интернет»**

Руководитель: Сачко Максим Анатольевич,  
старший преподаватель



# Тема 6

CGI

и

веб-программирование



# Содержание:

- 1) Интерфейс CGI, его задачи и функции
- 2) Исполнения CGI-скриптов
- 3) Структура URL и кодирование данных запроса
- 4) Переменные окружения CGI.
- 5) Программирование CGI-скриптов
- 6) Введение в Perl



# Основная концепция

---

Интерфейс CGI представляет собой спецификацию взаимодействия веб-сервера и внешней программы, которую веб-сервер запускает для обработки запроса. CGI определяет каким образом данные, предоставленные клиентом в запросе, передаются программе, как программа возвращает сгенерированный HTML-контент серверу, и какие переменные окружения устанавливаются сервером при запуске программы.

---

**Клиент** может запросить у веб-сервера как документ-файл с диска, так и документ, динамически формируемый некоторой внешней программой (как правило - в зависимости от данных, предоставленных пользователем при заполнении формы).

**Интерфейс CGI** представляет собой спецификацию взаимодействия веб-сервера и внешней программы, которую веб-сервер запускает для обработки запроса. (Внешняя программа, вне зависимости от своей природы, часто называется CGI-скриптом.)

---

Данные из заполненной клиентом HTML-формы могут передаваться на сервер двумя методами: **GET** и **POST**, это определяется параметром `method` соответствующего тэга `<form method=... action=...>`.

В первом случае (*GET*) данные присоединяются после вопросительного знака в конец URL, указанной в параметре *action*, во втором случае - передаются в теле запроса - в секции, предназначенной для данных (следует после всех заголовков и пустой строки).



# Например, вывод CGI-программы

---

Content-Type: text/html

```
<HTML>  
  <BODY>  
    <H1>Hello, world</H1>  
  </BODY>  
</HTML>
```



# Конфигурирование сервера Apache

---

```
ScriptAlias /виртуальный/путь/ /путь/к/каталогу/  
ScriptAlias /cgi-bin/ /usr/local/www/cgi-bin/
```

*Это означает, что для обработки запроса URL вида `http://your.server.com/cgi-bin/dir/script` будет взят не файл `script` из каталога `DocumentRoot/cgi-bin/dir/`, а запущена программа `/usr/local/www/cgi-bin/dir/script`.*



# Структура URL

---

Для работы CGI-программ важное значение имеют части **URL**, называемые **PATH\_INFO** и **QUERY\_STRING**.

Рассмотрим запрос с URL вида:

`http://my.server.com/cgi-bin/dir/prog/a/b?A=1&B=qwerty`

`PATH_INFO` - `/a/b`

`QUERY_STRING` - `A=1&B=qwerty`

---

Пары имя-значение разделяются амперсандом. Алфавитно-цифровые символы и некоторые знаки препинания, не имеющие специального значения (тире, подчеркивание) передаются как есть. Остальные символы кодируются в виде "%NM", где NM - двузначный шестнадцатеричный код символа.

.../prog?birthday=11%2F05%2F73&name=John+Smith

**birthday** - "11/05/73 "

**name** - "John Smith"



# Переменные окружения CGI

---

**AUTH\_TYPE** - Метод аутентифицирования, использованный для опознания пользователя. См. также REMOTE\_USER и REMOTE\_IDENT.

**CONTENT\_LENGTH** - Длина данных запроса в байтах, переданных CGI-скрипту через стандартный ввод.

**CONTENT\_TYPE MIME** - Тип данных запроса.

**DOCUMENT\_ROOT** - Корневой каталог дерева документов веб-сервера.

**GATEWAY\_INTERFACE** - Используемая версия CGI.



# Переменные окружения CGI

---

**HTTP\_ACCEPT** - Список MIME-типов данных, которые клиент может принять.

**HTTP\_FROM** - Адрес электронной почты пользователя, сделавшего запрос (многие браузеры не передают такие данные).

**HTTP\_REFERER** - URL документа, в котором находилась ссылка, вызвавшая настоящий запрос.

**HTTP\_USER\_AGENT** - Браузер клиента.

**PATH\_TRANSLATED** - **PATH\_INFO**, преобразованное в полный путь в файловой системе сервера



# Переменные окружения CGI

---

**QUERY\_STRING** - Данные запроса, переданные в составе URL вслед за вопросительным знаком

**REMOTE\_ADDR** - IP-адрес клиента.

**REMOTE\_HOST** - Имя DNS клиента.

**REMOTE\_USER** - Аутентифицированное имя пользователя.

**REQUEST\_METHOD** - Метод запроса (GET, POST, HEAD и т.д.).

**SCRIPT\_NAME** - Виртуальный путь (например, /cgi-bin/program.pl) к исполняемому CGI-скрипту



# Переменные окружения CGI

---

**SERVER\_NAME** - DNS-имя сервера или, при невозможности определить имя, его IP-адрес.

**SERVER\_PORT** - Номер порта сервера.

**SERVER\_PROTOCOL** - Имя и версия протокола, через который был сделан запрос (например, HTTP/1.1).

**SERVER\_SOFTWARE** - Тип и номер версии ПО веб-сервера.



# Cookies

---

Поскольку все HTTP-запросы независимы друг от друга и на уровне протокола HTTP отсутствует понятие сеанса связи, CGI-программа запускается заново для каждого вновь поступившего запроса, неважно имеет ли он связь с предыдущими или нет. Таким образом, существует проблема сохранения состояния логического сеанса работы пользователя между его последовательными запросами к CGI-программе. Для этого используются: cookies, сохранение состояния в базу данных, сохранение в файл, скрытые поля.



# Методы сохранения состояния

---

- cookies - сохранение на компьютере клиента,
- скрытые поля - сохранение внутри формы, посылаемой клиенту,
- сохранение в файле какого-либо формата на сервере,
- сохранение в параллельно работающей базе данных.





# Программирование CGI

---

При программировании CGI-скриптов всегда следует помнить, что при каждом очередном запросе скрипт начинает свою работу сначала, не имея никакой предыстории взаимодействия пользователя с этим или другими скриптами



---

```
my $x;      # $x создается, равно undef
$x;        # ложно
defined($x); # ложно
$x=0;
$x;        # ложно
defined($x); # ИСТИННО
$x=5;
$x;        # ИСТИННО
defined($x); # ИСТИННО
$x=undef;  # опять undef!
```



# Списки и массивы

---

Списком (list) называется упорядоченная последовательность скалярных значений; порядковые номера (индексы) начинаются с нуля. Отдельно стоящие списки заключаются в скобки:

**(\$x, "abc", 15)**

Обращение к элементу списка осуществляется путем указания индекса этого элемента в квадратных скобках:

**\$y=(\$x, "abc", 15)[1];    # \$y="abc"**



# Операторы

---

```
if (...) {...};
```

```
условие ? выражение_да : выражение_нет ;
```

```
while(...) {...};
```

```
do {...} while (...);
```

```
for (...;...;...;) {...};
```

```
if ($a<$b) { $a=$b; }
```

```
while ($a<$b) { some_function($a); }
```

```
$a=$b if ($a<$b);
```



---

```
$x="abc";  
@array=('c','d','e');  
%hash=( a => "A", b=> "B");  
  print "this is \$x: \"\$x\"; \nthis is element 2 of  
\@array: \"\$array[2]\";\n",  
  "and \$hash{a} is \"\$hash{a}\";\n";
```

## **ВЫВОД:**

```
this is $x: "abc";  
this is element 2 of @array: "e";  
and $hash{a} is "A"
```



# Вопросы для самопроверки:

1. Какие действия предпринимает сервер, если для обработки поступившего запроса требуется запуск CGI-программы?
2. Почему CGI-программа должна выдавать заголовок "Content-Type:"?
3. Что позволяет делать модуль `mod_perl`?
4. В чем заключается основная особенность CGI-программирования?
5. В чем состоит проблема сохранения состояния в CGI-программировании и какие существуют способы ее решения? Укажите их достоинства и недостатки.
6. Почему язык Perl считается наиболее удобным для написания CGI-приложений?



# Рекомендуемая литература:

1. Мамаев М., Петренко С. Технологии защиты информации в Интернете. Специальный справочник. – СПб: "Питер", 2005.
2. UNIX для системных администраторов: Энциклопедия пользователя/ Пер.с англ. – Киев: ДиаСофт, 2008.
3. Д.Р.Левин, К.Бароди. Секреты Интернет. – К.: Диалектика, ICE, 2005.
4. S.Spainbour, V.Quercia. Webmaster in a Nutshell. – O'Reily & Associates, Inc., 2003.



- **Использование материалов презентации**

- Использование данной презентации, может осуществляться только при условии соблюдения требований законов РФ об авторском праве и интеллектуальной собственности, а также с учетом требований настоящего Заявления.
- Презентация является собственностью авторов. Разрешается распечатывать копию любой части презентации для личного некоммерческого использования, однако не допускается распечатывать какую-либо часть презентации с любой иной целью или по каким-либо причинам вносить изменения в любую часть презентации. Использование любой части презентации в другом произведении, как в печатной, электронной, так и иной форме, а также использование любой части презентации в другой презентации посредством ссылки или иным образом допускается только после получения письменного согласия авторов.

