**Создание Web-приложений в среде Delphi**

Эпоха web-дизайна, когда наиважнейшим считалось возможность запихнуть на страницу как можно больше графики и поразить пользователя широтой фантазии дизайнера, канула в лету. В моде - информационный дизайн и всяческие удобности для пользователя. Плохим тоном уже считается одно только то, что на сайте нет форума или гостевой книги, и самое ужасное - о, боже - вы до сих пор верстаете каждую страницу вручную, вместо того, что бы поручить сеё скучное и утомительное занятие скрипту.

Время первопроходцев HTML в скором времени закончится и вовсе, и их место займут профессионалы-программисты. Именно программисты, ибо уже в большей степени от их умения создать удобный в использование сайт зависит его успех. Кроме прочего, теперь сайты создают совсем уж как программные продукты: тем кто занят наполнением содержанием, уже может не опасаться запутаться в html, java, cgi и прочих ипостасях интернет-технологий - администрирование сайта становится таким же привычным и удобным, как работа с текстовыми процессорами. Пока, правда, ощущается явный недостаток законченных продуктов, на плечи которых можно было бы возложить все функции по поддержанию сайта в актуальном состоянии, оставив себе лишь вопросы по наполнению его содержанием.

Пока же тяжкое бремя по созданию данных систем, на основе которых будет функционировать сайт, ложится на плечи программистов, только вот-вот успевших изумится возможностями языка Perl или Java. И многие из них, сказать без преувеличения, вышли, словно из гоголевской "Шинели", из Delphi. Действительно, данная среда разработчика, предоставляющая удобный интерфейс для визуального программирования и широкие возможности Object Pascal, столь мила сердцу российских программистов.

Но, похоже, что времена меняются, и Delphi из среды создания обычных настольных приложений, может сгодиться и в другом плане: для написания прикладных web-программ. И действительно, что может быть лучше: уже знакомая среда разработки, вдоль и поперек изученный язык, да и достаточно широкий круг специалистов по программированию в Delphi - это ли не плюсы создания web-приложений на Delphi. Есть конечно и минусы: созданные программы вряд ли смогут удовлетворить тех, кто считает, что лучший web-сервер, это сервер не от Microsoft. Но что поделаешь - версия Delphi под Unix отложена пока до лучших времен. Зато посудите сами: перенос программ в будущем светлом будущем можно будет совершить с малой кровью.

Но отложим пока теорию в сторону, и обратимся к практике. Итак, создание web-приложения в среде Delphi, что называется шаг за шагом.

**Пример из учебника**

В отличие от прочих аспектов создания приложений в Delphi, о создании приложений для web написано мало. Так ужасающе мало, что из книги в книгу, из учебника в учебник путешествует один и тот же пример. Не будем оригинальничать и мы - чем проще, тем лучше.

Вообще, простейшее web-приложение на Delphi мало чем отличается, а точнее ничем не отличается от создания программы для старой доброй DOS. Это - простейшее консольное приложение, запускаемое на стороне сервера и взаимодействующие с пользователем (в случае необходимости) через броузер.

Создайте в Delphi новый проект - ту самую уже давно забытую всем Console Application. Вы получите знакомый со школьной скамьи, текст обычной паскалевской программы:

program primer;

{$APPTYPE CONSOLE}

uses SysUtils;

begin

// Insert user code here

end.

Далее еще проще. Организуем вывод кода HTML с помощью команды writeln.

writeln ('CONTENT-TYPE: TEXT/HTML'); writeln; writeln ('<html>'); writeln ('<head>'); writeln ('<meta HTTP-EQUIV="Content-Type" Content="text-html; charset=windows-1251">'); writeln ('<title>Delphi the best facility for making web-publications!</title>'); writeln ('</head>'); writeln ('<body bgcolor="white">'); writeln ('Hello, world!'); writeln ('</body>'); writeln ('</html>');

Обратите внимание на строку CONTENT-TYPE: TEXT/HTML, которая определяет описание последующего содержимого, а именно кода HTML. После CONTENT-TYPE: TEXT/HTML, необходимо вывести пустую строку иначе броузер может выдать сообщение об ошибке.

Теперь, когда приложение закончено, осталось его скомпилировать и проверить. Для проверки работоспособности программы вам понадобиться веб-сервер. Можно особо не утруждаться, подойдет любой, даже стандартный домашний веб-сервер от Microsoft. Приложение надо будет разместить в папке публикаций сервера (обычно это - C:\Inetpub\ wwwroot) и запустить сам сервер. Теперь, если вы перейдете по адресу http://localhost/primer.exe в броузере, вы должны увидеть результат действия данной программы - строку Hello, world! Вот и всё, простейшее web-приложение на Delphi готово.

**Передача параметров**

На самом деле нам бы вряд ли понадобилось Delphi, для создания подобных программ. Конечно, можно генерировать страницы исходя из различных условий, но вопрос в том, как данные условия передать программе. И здесь оказывается не всё так сложно, достаточно вспомнить передачу параметров приложению с помощью командной строки и поступить соответствующе. Хотя это, разумеется, хитрость. Это для приложения Delphi мы оперируем командной строкой, для пользователя же это адресная строка в броузере, то есть url.

Попробуем на примере. Необходимо создать приложение, которое выдает различную информацию (к примеру, время, дату или то и другое вместе) в зависимости от параметров, указанных в адресной строке броузера. Как известно, за данные в параметрах отвечают такие функции, как ParamCount и ParamStr. Их-то мы и будем использовать.

program CgiDate;

{$APPTYPE CONSOLE}

uses SysUtils;

begin

writeln ('CONTENT-TYPE: TEXT/HTML');

writeln;

writeln ('<HTML><HEAD>');

writeln ('<TITLE>Cgidate</TITLE>');

writeln ('</HEAD><BODY>>');

writeln ('<H1>Пример передачи параметров</H1>');

writeln ('<HR>');

writeln ('<H4>

if ParamCount >0 then

begin

if ParamStr (1) = 'date' then

writeln (FormatDateTime('"Сегодня " dddd, mmmm d, yyyy', Now))

else if ParamStr (1) = 'time' then

writeln (FormatDateTime('"Время" hh:mm:ss AM/PM', Now))

else if ParamStr (1) = 'both' then

writeln (FormatDateTime('"Сегодня " dddd, mmmm d, yyyy,'

+ '"<p> и время" hh:mm:ss AM/PM', Now))

else

writeln ('Ошибка! Неверный параметр: ' + ParamStr (1) + '.')

end

else

writeln ('Параметр отсутствует.');

writeln ('</BODY></HTML>');

end.

Не правда ли просто? Теперь, если в адресной строке броузера вы наберете, например http://localhost/cgidate/exe?time, будет сгенерирована страница, отображающая текущее время, http://localhost/cgidate/exe?date - соответственно дата, а при передаче параметра both - текущая дата и время. В случае если никакой из параметров передан не был или он был ошибочен - возникнет сообщение об этом.

Данные адреса и параметры можно непосредственно указать в коде HTML и генерировать необходимые изменения на странице либо другие страницы переходя по соответствующим ссылкам.

Следует обратить внимание на то, как передавать данные через url. Знак вопроса отделяет параметр от адреса файла, с помощью знака равенства web-приложению передается значение данного параметра. Так как в адресной строке нельзя использовать пробел, он заменяется на шестнадцатеричный код в таблице ASCII, то есть %20.

Но на самом деле, если некие данные передаются от пользователя web-приложению, то обычно для этого используют формы, а не url (хотя одно другому не мешает). Попробуем и мы создать приложение, которое бы получало данные от пользователя, занесенные им в форму.

Для начала, естественно, нужно создать сам код HTML в котором бы присутствовала форма с полями ввода, кнопкой отправки и прочими необходимыми атрибутами. При этом form action должен содержать адрес программы, которая будет получать данные. Значение method может быть равно как GET, так и POST. На самом деле GET - это и есть передача параметров через url, добавляя их к адресной строке, так как POST передает их приложению посредством стандартного потока ввода. Какой из них лучше и удобней - решать вам, но чаще всего метод GET используется именно для генерации страниц (достаточно взглянуть на url который возникает при работе на поисковых серверах), тогда как второй для - передачи данных, отображать которые в адресной строке было бы весьма накладно.

Затем необходимо создать приложение, которое бы адекватно смогло воспринять все эти данные со стороны пользователя. В случае если данные передавались с помощью метода GET проблем не будет - можно действовать как в предыдущем примере. Если же приложение получает данные от пользователя с помощью POST, всё несколько иначе.

Необходимо будет считывать данные из переменной окружения, а для этого мы, естественно, должны знать данные переменные.

**Гюльчитай, открой личико**

Нет в мире тайн. Особенно их мало у пользователя от web-приложения. И если мы не знаем о пользователе кое-что личное, все прочее броузеры с легкостью отдают web-серверу, нисколько не заботясь о приватности и желаниях того самого пользователя. Это, конечно, нехорошо для пользователя, но хорошо для разработчика web-приложения, поскольку для него знания - великая вещь.

Итак, данные от пользователя web-приложению можно передать через переменные окружения. Вот список наиболее часто употребляемых:

GATEWAY\_INTERFACE

Поддерживаемая версия CGI.

REQUEST\_METHOD

Метод запроса, может быть как GET так и POST.

HTTP\_REFERER

Адрес страницы (url), активирующей текущее приложение на web-сервере.

PATH\_INFO

Путь переданный приложению расположенный между именем приложения и строкой запроса.

QUERY\_STRING

Строка запроса, если метод - GET, добавляеться к url.

REMOTE\_HOST

Имя хоста удаленного пользователя.

REMOTE\_USER

Имя удаленного пользователя.

REMOTE\_IDENT

IP-адрес удаленного пользователя.

HTTP\_USER\_AGENT

Имя и версия броузера удаленного пользователя.

С помощью данных переменных можно получить исчерпывающую информацию о пользователе и передаваемых данных для верного проектирования вашего web-приложения. Конечно, этого хватит в том случае, если вы не собираетесь подобно Большому Брату следить за каждым телодвижением пользователя.

Но вернемся к поставленной задаче - передаче данных приложению от пользователя через форму. Данные, которые передаются через QUERY\_STRING в приложение с помощью метода POST, достаточно просто извлечь для использования.

Ниже листинг программы, выдающий список некоторых переменных окружения и их значения. Узнайте кое-что о своем броузере и web-сервере.

program CgiVars;

{$APPTYPE CONSOLE}

uses

Windows;

сonst

VarList: array [1..17] of string [30] =

('SERVER\_NAME', 'SERVER\_PROTOCOL',

'SERVER\_PORT', 'SERVER\_SOFTWARE',

'GATEWAY\_INTERFACE', 'REQUEST\_METHOD',

'PATH\_TRANSLATED', 'HTTP\_REFERER',

'SCRIPT\_NAME', 'PATH\_INFO',

'QUERY\_STRING', 'HTTP\_ACCEPT',

'REMOTE\_HOST', 'REMOTE\_USER',

'REMOTE\_ADDR', 'REMOTE\_IDENT',

'HTTP\_USER\_AGENT');

var

I: Integer;

ReqVar: string;

VarValue: array [0..200] of Char;

begin

writeln('Content type: text/html');

writeln;

writeln('<HTML><HEAD>');

writeln('<TITLE>CGI Variables</TITLE>');

writeln('</HEAD><BODY>');

writeln('<H1>CGI Variables</H1>');

writeln('<HR><PRE>');

for I := Low (VarList) to High (VarList) do

begin

ReqVar := VarList[I];

if (GetEnvironmentVariable (PChar(ReqVar),

VarValue, 200) > 0) then

else

VarValue := '';

writeln (VarList[I] + ' = ' + VarValue);

end;

writeln('</PRE></BODY></HTML>');

end.

**За кадром**

В этой статье мы не коснулись другой и уж наверняка более обширной и сложной темы, как создание ISAPI-приложений на Delphi. Вышеприведенные способы создания приложений годны лишь в том случае, если вам необходимо быстрое, компактное и не слишком сложное web-приложение. Если же вам необходим, к примеру, доступ к базам данным, то подобный путь неприемлем.

Для создания полномасштабных приложений для интернета в Delphi существует специальный помощник - Web Server Application. С его помощью можно создать приложение генерируещее динамические web-страницы, основанные на CGI, NSAPI или ISAPI. Единственное накладываемое ограничение - непосредственно web-сервер должен работать на базе Windows.

Одним из главных преимуществ создания подобных приложений именно в среде Delphi является то, что вы продолжаете работать с визуальными компонентами - это значительно проще, чем создание приложений в других средах - возможность ошибки в больших проектах, где используется визуальное проектирование меньше, чем в тех, где всё описывается исключительно кодом. Кроме того, средства создания web-приложений позволяют импортировать уже существующие приложения в интернет-среду, что, согласитесь, немаловажно. Пока, конечно, Delphi не обладает большим набором компонентов для web-приложений, но, видимо, уже следующая версия этой среды обзаведется необходимыми. Но и сейчас Delphi можно считать достаточно удобным инструментом для создания приложений, взаимодействующих с интернетом.