Урок 1. Основы PHP

Знакомимся с основами веб-программирования

- Что представляет из себя Интернет
  - Что такое IP-адрес?
  - Запрос → ответ
  - Где здесь PHP?
- PHP
  - Программное обеспечение
  - Первая программа
  - Переменные и константы
  - Типы данных
  - Операции со строками
  - Приведение к типу
  - Математические операции
  - Константы
- Итоги
  - Пример: Задача 1. Написать программу, вычисляющую факториал.
- Домашнее задание
  - Обязательная часть
  - Дополнительные задания
- Литература

Что представляет из себя Интернет

Первоначально, ещё в 60ых годах прошлого века Интернет назывался ARPANet – сеть, предназначенная для военных целей: а именно, для координации работы командных центров в случае, если какой-либо из них вышел из строя. Это предполагало, что каждый узел должен быть связан с другим. Позднее, Интернет появился в образовательных учреждениях США и, далее, на рубеже 80ых и 90ых годов распространился по всему миру.

Однако, одной лишь кабельной сети недостаточно для того, чтобы называться глобальной сетью: необходимо, чтобы один узел сети мог «понять», что ему отправляет другой узел. Набор правил пересылки данных называется протоколом. Их великое множество, но, самый главный протокол передачи данных через сеть называется TCP/IP – Transmission Control Protocol / Internet Protocol.

Именно с помощью данного протокола происходит обмен данными: получение веб-страницы, видео-потока, картинки, сетевые игры и многое другое. Подробнее об устройстве протокола вы можете прочитать в Википедии. Самое основное, что нам важно знать — что такое IP-адрес и для чего он нужен.

Что такое IP-адрес?

Для того, чтобы, например, отправить письмо или зайти на веб-страницу, нам достаточно лишь знать, собственно, сам email-адрес или адрес этой веб-страницы. Но что же происходит «под капотом»? Как именно наши программы узнают, что мы хотим попасть именно на этот сайт? Откуда Интернет знает, где располагается тот или иной сайт?

У каждого компьютера, подключенного к глобальной сети, есть свой собственный адрес — IP-адрес, который представляет из себя последовательность чисел (4 байта) от 0 до 255 и разделённые точкой. Примеры:

192.168.0.21

213.141.199.33

10.10.4.87

Каждый «разряд» - соответствует определённой подсети. Компьютеры в рамках одной подсети соединены со шлюзом и могут общаться друг с другом. Шлюз, в свою очередь, может быть таким же компьютером, соединённым с другими соседями, которые являются шлюзом для своих подсетей. Именно таким образом строится глобальная сеть. При этом, любое звено в этой сети может быть соединено не с одним шлюзом, а с несколькими. Это обеспечивает сети глобальную устойчивость.

Запрос → ответ

И так, мы узнали, что чтобы получить те или иные данные с другого компьютера, нам необходимо знать его адрес в сети. Даже в том случае, когда мы хотим зайти на какой-нибудь сайт, нам нужно знать IP-адрес компьютера — сервера, на котором эта веб-страница находится. Но из практики мы знаем, что никаких ip-адресов мы в адресной строке не вводим.

В первую очередь, нам необходимо узнать несколько понятий:

Сервер — специальный компьютер, подключённый к сети. Используется, как правило, для приёма и обработки тех или иных запросов, в т.ч. - веб-страниц, почты, игровой трафик и тому подобное.

DNS – Служба доменных имён. Специальная база данных, содержащая информацию о соответствии домена (адреса страницы) с IP-адресом.

Порт — условно, одна из «дверей» на сервере, в которую можно «зайти» и получить (или отправить) нужную информацию. Например, так исторически сложилось, что веб-страницы можно получить на 80ом порту. А доступ к файлам сервера (если разрешено) — можно получить через 21 порт. Почта отправляется через порт 25. Всего их может быть 65535 на одном IP-адресе в рамках одного протокола.

HTTP / FTP / SMTP (и другие) – протоколы передачи данных прикладного уровня. Своеобразные наборы правил, по которым оформляются пакеты для запросов и ответов. Работают поверх базового протокола TCP/IP.

На самом деле, когда вы вводите адрес веб-страницы в браузере, происходит следующее:

1. После нажатия кнопки enter браузер смотрит настройки вашей сети и узнаёт адреса DNS-серверов.
2. Спрашивает у DNS-сервера: знаешь ли ты IP-адрес сайта anysite.com?
3. Если DNS-сервер вашей сети отвечает отрицательно, ваш браузер опрашивает корневые DNS-сервера. На них содержится база соответствий вида «домен — ip-адрес».
4. После ответа от корневого DNS-сервера, эта запись возвращается сразу и вам, и вашему местному DNS-серверу. Теперь вам известен IP-адрес.
5. После того, как ваш браузер получил IP-адрес, он подключается по данному IP-адресу через порт 80 или 443 (соответствует протоколам HTTP и HTTPS), специальным образом оформляет запрос, где указывает IP-адрес запрашиваемого сервера, домен, и адрес самой страницы на сайте. Вместе с этими данными браузер может переслать и другие данные: язык вашей системы, местное время, страница, с которой вы пришли, и так далее.
6. Составленный запрос отправляется веб-серверу. Веб-сервер проверяет его правильность, смотрит, есть ли у него такая страница, считывает её содержимое и, отправляет обратно браузеру. Браузер читает html-код веб-страницы (должен быть знаком вам по курсу HTML и CSS) и отражает уже готовую страницу на экране.

Попробуйте запустить командную строку и написать

```
telnet 37.140.192.198 80
GET /index.html HTTP/1.0
Host: pogoda.ru
```

Каждый раз, когда вы переходите по ссылке или открываете картинку, действия повторяются. Для каждого отдельно загружаемого элемента на странице (стили, картинка, видео, javascript) – выполняется то же самое. Но все эти процессы скрыты от простого пользователя.

Где здесь PHP?

Изначально, в Интернете были только статичные веб-страницы, вроде тех, что вы делали (или будете делать) на курсе HTML и CSS. Чтобы посетитель мог пообщаться с автором страницы, ему предлагать пройти по специально сформированной ссылке (mailto:) и написать письмо с помощью программы-клиента. Со временем, стало понятно, что этого недостаточно: нужен какой-то интерактив — заказ товаров, отправка сообщений, автоматически обновляемые данные на страницах.

Для этих целей необходимы были программы, исполняемые на сервере. Как выполняется

подобная программа?

1. Веб-сервер получает запрос на получение той или иной веб-страницы
2. Веб-сервер сравнивает свои настройки с характером запрашиваемых данных
3. Если мы, например, запрашиваем веб-страницу index.html, веб сервер отдаёт её в том же виде, в каком эта страница на нём сохранена. Если была запрошена страница page.php, то:
4. Веб-сервер запускает интерпретатор PHP, и перенаправляет туда все данные, полученные от пользователя, мешая их с некоторыми своими.
5. Интерпретатор выполняет заданную программу и формирует содержимое веб-страницы
6. После завершения программы, результат её выполнения отправляется веб-серверу
7. Веб-сервер пересылает запрос пользователю.

PHP

Язык программирования PHP был разработан в 1995 году, и изначально планировался именно для разработки веб-страниц. Получил очень широкое распространение на счёт своей простоты и открытого исходного кода интерпретатора, благодаря чему, его доработкой могли заниматься энтузиасты и коммерческие компании.

Язык PHP – интерпретируемый, то есть программа-интерпретатор выполняет написанные на нём инструкции построчно.

Исходный код программ на PHP исполняется непосредственно на веб-сервере, в отличие от JavaScript – пользователь не может посмотреть исходный код программы и узнать, что в ней происходит. То же касается любого другого серверного языка программирования (даже JS в составе NodeJS!).

Программное обеспечение

Для выполнения команд языка нам необходимо подготовить окружение.

Компоненты, которые нам понадобятся:

– Веб-сервер
– Интерпретатор PHP
– База данных MySQL

Можно собрать комплект самостоятельно, можно воспользоваться готовой сборкой.

Кроме того, нам понадобится среда разработки. Программы на PHP можно писать в любом текстовом редакторе, даже в том, что по-умолчанию входит в стандартный комплект вашей ОС. Однако, это неудобно. Существуют различные редакторы с подсветкой синтаксиса:

– Notepad++
– Atom
– Visual Studio Code

Можно воспользоваться профессиональными IDE: PhpStorm или Netbeans.

Первая программа

Откройте ваш редактор и напишите следующее:

```
<?php
echo 'Hello, world!';
?>
```

Любая программа на языке PHP должна начинаться с «<?php». Допустим сокращённый вариант: «<?», но использовать его не рекомендуется.

Команда «echo» выводит на экран строку, заключённую в кавычки. Любая команда должна заканчиваться с помощью «;».

Завершение программы обозначается с помощью «?>», но это необязательно.

Сохраните ваш код в файл index.php в директории, которая является корнем веб-сервера. Откройте браузер и напишите в адресной строке: http://localhost

Посмотрите исходный код открывшейся страницы: в ней нет ничего, кроме «Hello, world!». Отображается только результат. Инструкции, которые вы описали в программе, никому не видны.

Переменные и константы

Наша первая программа — вывод заранее заданной строки на экран. В конечном счёте, это ничем не отличается от статической веб-страницы. Усложним нашу программу, добавив в неё переменные.

```
<?php
$name = "Alex";
echo "Hello, $name!";
?>
```

В контексте языков программирования, переменная — размеченная область в памяти, в которой записано какое-либо значение, либо не записано никакого. У любой переменной есть определённый числовой адрес - шестнадцатеричное число. Машина среди таких переменных ориентируется без проблем, однако человеку это достаточно трудно. В языках программирования переменным можно задавать синоним — например, $name. В данном случае в этой переменной записано имя.

Любая переменная в PHP обозначается знаком «$». Переменная в PHP должна состоять из латинских символов (хотя, язык допускает имя переменной на других языках) и цифр, не должно содержать спецсимволов, кроме «_», и не может начинаться с числа.

Правильно:

- $variable
- $myVariable
- $_variableWithDigits

Неправильно:

- $1stvariable
- $Переменная (сработает, но так нельзя!)
- $any%other/variable-withSymbols

Откройте ваш скрипт в браузере. Измените значение переменной и откройте снова. Добавьте новые переменные.

Типы данных

Мы можем присваивать значения переменных не только в виде строк, но и в виде массивов, чисел, булевых значений, объектов.

```
<?php
$a = "Hello"; // Строка
$b = true; // Булевое значение (true/false)
```

```php
$c = 45; // Целочисленное число
$d = 4.2; // Число с плавающей точкой

echo $a;
echo $b;
echo $c;
echo $d;
?>
```

Выведите результат работы скрипта в браузер и внимательно посмотрите на результат.

Строка вывелась без изменений, а true превратился в единицу. Остальные числа также вывелись без изменений.

Дело в том, что тип boolean – преобразовался из одного типа в другой. В данном случае он был преобразован сначала в числовой тип — 1 (так, как true = 1, false = 0), а затем — в строчный.

PHP – язык с динамической типизацией. Это означает, что любая наша переменная в процессе выполнения программы может изменить свой тип данных. В языках со строгой типизацией присвоить, например, переменной с типом bool строчное или числовое значение не получится.

Операции со строками

Объявлять переменные строкового типа можно двумя способами:

```php
<?php
$a = 'Hello,';
$b = "world";
?>
```

С первого взгляда разницы никакой. Однако, если выполнить следующий пример:

```php
<?php
$a = 'Hello';
$b = "world";

$c = "$a, $b";
$d = '$a, $b';
echo $c;
echo $d;
```

```
?>
```

Мы увидим, что переменные, указанные в одинарных кавычках, не считываются интерпретатором, и отображаются, ровно так, как они записаны внутри переменной.

Мы можем объединять несколько строковых переменных в одну. Такая операция называется — конкатенация. Пример:

```
<?php
$a = 'Hello,';
$b = 'world';
$c = $a . $b;
echo $c;
?>
```

Операция соединения строк — конкатенация — в языке PHP выполняется с помощью символа «.». Кроме того, при выполнении данной операции, все переменные других типов, если это возможно, будут тоже приведены к строковому типу. Обратите внимание, что, согласно стандарту PSR-2, при конкатенации двух строк справа и слева от точки должны быть пробелы.

Приведение к типу

Приведение переменной одного типа к другому типу можно использовать встроенные функции PHP, такие как intval(), boolval(), stringval() (о функциях поговорим позже), так и их сокращённые варианты:

```
<?php
$a = (int)'05';
$b = (string)145.3;
$c = (bool)-4;



echo 'Строка -> число -> строка : ' . $a . '<br>';
echo 'Число -> строка : ' . $b . '<br>';
echo 'Число -> булев -> строка : ' . $c . '<br>';
?>
```

Попробуйте объяснить, почему те или иные переменные были выведены именно так.

Математические операции

Помимо стандартных математических операций:

```
<?php
$a = 4;
$b = 5;
echo $a + $b . '<br>'; // сложение
echo $a * $b . '<br>'; // умножение
echo $a - $b . '<br>'; // вычитание
echo $a / $b . '<br>'; // деление
echo $a % $b . '<br>'; // остаток от деления
echo $a ** $b . '<br>'; // возведение в степень
?>
```

Существуют и другие, менее очевидные: *=, /=, +=, -= - эти операторы позволяют выполнить математическое действие и сразу же присвоить значение переменной. Пример:

```
<?php
$a = 4;
$b = 5;

$a += $b;
echo 'a = ' . $a;
?>
```

Ещё:

```
<?php
$a = 0;
echo $a++; // Постинкремент
echo ++$a; // Преинкремент
```

```
echo $a--; // Постдекремент
echo --$a; // Предекремент
?>
```

Операции сравнения:

```
<?php
$a = 4;
$b = 5;

var_dump($a == $b); // Сравнение с приведением типов
var_dump($a === $b); // Сравнение без приведения типов
var_dump($a > $b); // Больше
var_dump($a < $b); // Меньше
var_dump($a <> $b); // Не равно
var_dump($a != $b); // Не равно
var_dump($a !== $b); // Не равно без приведения типов
var_dump($a <= $b); // Меньше или равно
var_dump($a >= $b); // Больше или равно
?>
```

Функция var_dump позволяет вывести тип переменной из её значение. Попробуйте выполнить код. Обратите внимание, что знак «=» - не знак равенства, а знак присваивания.

Чем отличается сравнение с приведением типов и без? Попробуйте выполнить код:

```
<?php
$a = 4;
$b = '4';

var_dump($a == $b);
var_dump($a === $b);
?>
```

Попробуйте объяснить результат.

Константы

Константы мало чем отличаются от переменных, кроме одной важной детали: им нельзя присвоить другое значение. Константы используются, как некое подобие системных переменных, которые устанавливает разработчик, но обычный пользователь никак повлиять на них не может. Константы устанавливаются с помощью функции define():

```
<?php
define('MY_CONST', 100);
echo MY_CONST;
?>
```

Обратите внимание, что обращение к константе осуществляется без знака «$». Правила написания названий констант такие же, как и для переменных, однако, согласно текущим стандартам, константы должны быть написаны заглавными буквами.

Итоги

В рамках данного скучного урока мы с вами изучили: как осуществляется доступ к веб-сайтам в интернете, изучили самые основы PHP: переменные, их типы, простейшие операции над ними. Для чего все эти числа и значки нужны? Любая программа состоит из условий, циклов, обработки данных из одного типа в другой. Без понимания механизмов сравнения и типизации, мы не сможем правильно описывать даже самые простые условия.

Домашнее задание

Обязательная часть

1. Установить программное обеспечение: веб-сервер, базу данных, интерпретатор, текстовый редактор и проверить, что всё работает правильно.
2. Выполнить примеры из методички, разобраться, как это работает.
3. Объясните, как работает данный код:

```php
<?php
// --- //
$a = 5;
$b = '05';
var_dump($a == $b); // Почему true?
// --- //
var_dump((int)'012345'); // Почему 12345?
// --- //
var_dump((float)123.0 === (int)123.0); // Почему false?
// --- //
var_dump((int)0 === (int)'hello, world'); // Почему true?
?>
```

Дополнительные задания

Используя только две переменные, поменяйте их значение местами. Например, если a = 1, b = 2, надо, чтобы получилось: b = 1, a = 2. Дополнительные переменные использовать нельзя.

Литература

- Котеров Д.: PHP 5 в подлиннике
- O. Kirch: Linux Network Administrator's Guide, 2nd Edition

Дополнительные материалы

- http://php.net/manual/ru/language.basic-syntax.comments.php
- http://php.net/manual/ru/language.expressions.php
- https://ru.wikipedia.org/wiki/Telnet
- https://ru.wikipedia.org/wiki/HTTP