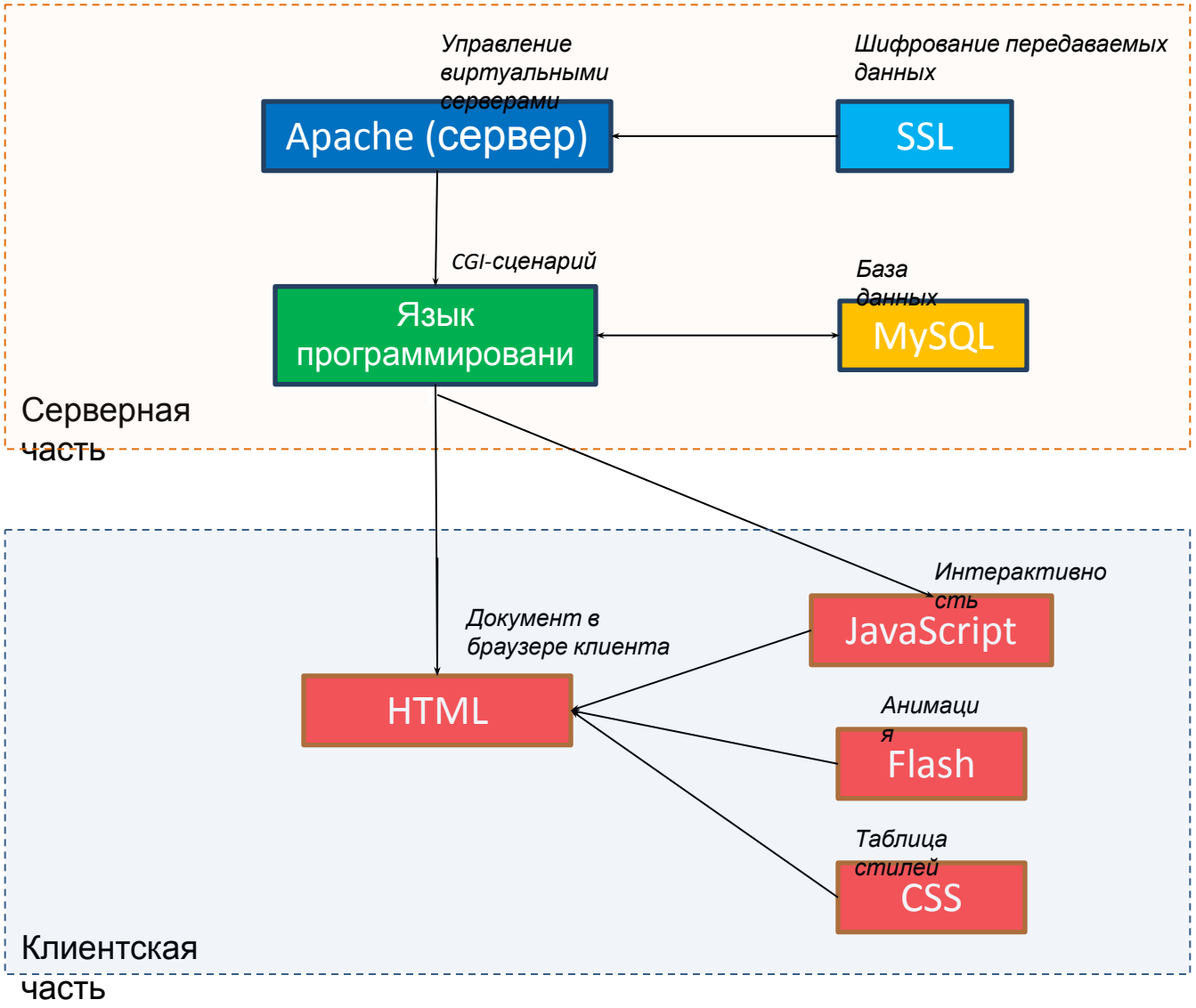
Лекция 1

Динамические и статические страницы

- Динамическая страница — Web-страница, сгенерированная с помощью логически построенной программы в зависимости от запрошенных пользователем данных.
- Статическая страница является простым файлом, лежащим на сервере.

Архитектура Web



Принципы получения данных динамической страницей

- Через HTML-формы методами GET и POST
- Через HTTP-Cookies
- Через переменные окружения Web-сервера

PHP

- “PHP: Hypertext Preprocessor” - «PHP: Препроцессор Гипертекста» интерпретируемый язык общего назначения с открытым исходным кодом.

Необходимые компоненты

- Web-сервер с поддержкой PHP
- Любой текстовый редактор
- Браузер

Пример PHP-скрипта

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Пример</title>
    </head>
    <body>

        <?php
        echo "Привет, я - скрипт PHP!";
        ?>

    </body>
</html>
```

Вывод строк

```
void echo ( string $arg1 [, string $... ] )
```

- Выводит одну или более строк
- Заключать аргументы в скобки необязательно
- Можно использовать HTML-теги для форматирования

Вывод строк. Примеры

```php
<?php
echo "Привет мир!";

echo "Это займет
несколько строк. Переводы строки тоже
выводятся";

echo "Это займет\nнесколько строк. Переводы строки тоже\nвыводятся";

echo "Экранирование символов делается \"Так\".";

// с echo можно использовать переменные ...
$foo = "foobar";
$bar = "barbaz";

echo "foo - это $foo"; // foo - это foobar

// При использовании одиночных кавычек выводится имя переменной, а
// не значение
echo 'foo - это $foo'; // foo - это $foo
?>
```

Комментарии

- Многострочные:

  `/* … */`
- Однострочные:

  `// …`

  `# …`

Переменные

- Представлены знаком доллара с последующим именем переменной. Имя переменной чувствительно к регистру.
- **`$var`** `= “Привет”`
- Имя переменной должно начинаться с буквы или символа подчеркивания и состоять из букв, цифр и символов подчеркивания в любом количестве

Типы данных

PHP поддерживает восемь простых типов.
Четыре скалярных типа:

- boolean – логический
- integer – целочисленный
- float – число с плавающей точкой
- string – строковый

Два смешанных типа:

- array – массив
- object – объект

Два специальных типа:

- resource – ресурс (файл)
- NULL – переменная без значения

Типы данных

- Тип переменной определяется на основе ее значения

См. также:

Таблица сравнения типов в PHP

http://ru2.php.net/manual/ru/types.comparisons.php

Математические операторы

- + Сумма двух чисел.
- - Разность чисел.
- * Произведение двух чисел.
- / Частное от деления двух чисел.
- % Остаток от деления

Операторы присвоения

Основным оператором присвоения является знак равенства (" = ")

Комбинированные операторы:

- **+=**
- **-=**
- **/=**
- **.=**
- **%=**
- **&=**
- **|=**
- **^=**
- **<=**
- **>=**

Инкремент, декремент

Оператор *++* называют инкрементом,
а *--* декрементом.

- ***++$a*** Пре-инкремент Увеличивает значение на единицу.
- ***$a++*** Пост-инкремент Возвращает текущее значение, после чего увеличивает его на единицу.
- ***--$a*** Пре-декремент Уменьшает значение на единицу.
- ***$a--*** Пост-декремент Возвращает текущее значение, после чего уменьшает его на единицу.

Приоритет операторов

new

[

! ~ ++ -- (int) (float) (string) (array) (object)

@

* / %

+ - .

< >

< <= > >=

&

^

|

&&

||

? :

= += -= *= /= .= %= &= |= ^= <= >=

print

and

xor

or

,

Строковые операторы

- Оператор конкатенации (" . "), который объединяет две строки в одну.
- Конкатенирующий оператор присвоения (" .= "), добавляет к строке нужное значение

Условный оператор IF

```
<?php
if ($a > $b)
   echo "a больше b";
?>
```

Оператор ELSE

```
<?php
if ($a > $b) {
   echo "a больше, чем b";
} else {
   echo "a НЕ больше, чем b";
}
?>
```

Оператор ELSEIF

```
<?php
if ($a > $b) {
    echo "a больше, чем b";
} elseif ($a == $b) {
    echo "a равен b";
} else {
    echo "a меньше, чем b";
}
?>
```

Операторы сравнения

==	Равенство	Истина, если $a равно $b
===	Идентичность	Истина, если $a равно $b, и они одного и того же типа
!=	Неравенство	Истина, если $a не равно $b
<>	Неравенство	Истина, если $a не равно $b
!==	Неидентичность	Истина, если $a не равно $b, или они не одного типа
<	Меньше	Истина, если $a меньше $b
>	Больше	Истина, если $a больше $b
<=	Меньше или равно	Истина, если $a меньше или равно $b
>=	Больше или равно	Истина, если $a больше или равно $b

Логические операторы PHP

and	Логическое "И"	Истина, если истинно $a и $b
&&	Логическое "И"	Истина, если истинно $a и $b
or	Логичесоке "ИЛИ"	Истина, если истинно $a или $b
||	Логическое "ИЛИ"	Истина, если истинно $a или $b
xor	Логическое "Исключающее ИЛИ"	Истина, если истинно $a или $b, но не оба одновременно
!	Логическое "НЕ"	Истина, если $a ложь

```
if (($speed > 35) and ($speed < 55))
    echo "Скорость в пределах нормы";
```

Оператор SWITCH

```
switch ($i) {
    case 0:
        echo "i равно 0";
        break;
    case 1:
        echo "i равно 1";
        break;
    case 2:
        echo "i равно 2";
        break;
}
```

Цикл FOR

```
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

Цикл While

```
$i = 1;
while ($i <= 10) {
    echo $i++;
}
```

Цикл Do-While

```
$i = 0;
do {
    echo $i;
} while ($i > 0);
```

foreach (вариант 1)

```
$myarray = array("BMW" => 2,
                 "Mercedes" => 12,
                 "Audi" => 0);
foreach ($myarray as $value){
    echo $value."<br>";
}
```

ВЫВОД:

2
10
0

foreach (вариант 2)

```
$myarray = array("BMW" => 2,
                 "Mercedes" => 12,
                 "Audi" => 0);
foreach ($myarray as $key => $value){
    echo $key." : ".$value."<br>";
}
```

ВЫВОД:

BMW : 2

Mercedes : 10

Audi : 0

Вывод строк при помощи print

Функции работы со временем

- time — Возвращает текущую метку времени Unix
- Возвращает количество секунд, прошедших с начала Эпохи Unix (The Unix Epoch, 1 января 1970 00:00:00 GMT) до текущего времени.

Функции работы со временем

- date — Форматирует вывод системной даты/времени

```
string date ( string $format [, int $timestamp = time() ] )
```

- Возвращает строку, отформатированную в соответствии с указанным шаблоном format. Используется метка времени, заданная аргументом timestamp, или текущее системное время, если timestamp не задан.

Подробнее о форматировании см.
http://ru2.php.net/manual/ru/function.date.php

Функции работы со временем

```
<p>Текущее время: <b><?php echo
time();?></b></p>
<?php $nextWeek = time()+(7*24*60*60); ?>
<p>Следующая неделя: <b>
<?php echo $nextWeek; ?></b></p>
<p>Человекопонятное текущее время: <b>
<?php echo date("d.m.Y G:i:s"); ?></b></p>
<p>Человекопонятное время через неделю:<b>
<?php echo date("d.m.Y G:i:s", $nextWeek);
?></b></p>
```

Создание массивов

- Задание пар «ключ-значение»
- `$arr[1] = "php";`
- `$arr[2] = "html";`
- `$arr[3] = "css";`
- Функция ***array()***

```
$arr1 = array('Яблоки', 'Груши', 'Сливы', 20, 10.2);
// Первый элемент получает индекс 0
$arr2 = array (1 => "php", "html", "css");
// Первый элемент получает индекс 1
```

Создание массивов

- Сокращенная запись

```
$arr2 = array();
$arr2[] = "Фрукты";
$arr2[] = "Овощи";
$arr2[] = "Ягоды";
```

Первый элемент получает индекс **0**

Создание массивов

- Ассоциативный массив

```
$assocArr = array(
   "lang" => "php",
   "group" => 97,
   "university" => "МАТИ");
```

Создание массивов

- Многомерный массив

```
$longArray = array(
    "foo" => "bar",
    42 => 24,
    "multi" => array(
        "dimensional" => array(
            "array" => "foo"
        )
    )
);
```

Удаление элементов массива

```php
<?php
$arr = array(5 => 1, 12 => 2);

$arr[] = 56;     // В этом месте скрипта это
                 // то же самое, что и $arr[13] = 56;

$arr["x"] = 42; // Это добавляет к массиву новый
                 // элемент с ключом "x"

unset($arr[5]); // Это удаляет элемент из массива

unset($arr);     // Это удаляет массив полностью
?>
```

Вывод элементов массива

- При помощи цикла for

```
$cnt = count($arr);
for ($i = 0; $i < $cnt; $i++) {
    echo $arr[$i],"<br />";
}
```

Вывод элементов массива

- При помощи цикла foreach

```
<?php

$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

foreach($arr as $value)
{
    echo $value, "<br>";
}

?>
```

Вывод элементов массива

- При помощи функции print_r

print_r() выводит информацию о переменной в удобочитаемом виде.

```
<?php

$arr[0] = "PHP";
$arr[1] = "HTML";
$arr[2] = "CSS";

print_r($arr);

?>
```

Функция isset

Определяет, была ли установлена переменная значением отличным от NULL

- Если переменная была удалена с помощью unset(), то она больше не считается установленной
- isset() вернет FALSE, если проверяемая переменная имеет значение NULL.
- Если были переданы несколько параметров, то isset() вернет TRUE только в том случае, если все параметры определены.

Функция empty

Проверяет, пуста ли переменная

Возвращает FALSE, если var содержит непустое и ненулевое значение.

Следующие значения воспринимаются как пустые:

- *""* (пустая строка)
- *0* (целое число)
- *0.0* (дробное число)
- *"0"* (строка)
- NULL
- FALSE
- *array()* (пустой массив)

Функции isset и empty

```php
<?php
$var = 0;

// Принимает значение true, потому что $var пусто
if (empty($var)) {
    echo '$var или 0, или пусто, или не определена';
}

// Принимает значение true, потому что $var определена
if (isset($var)) {
    echo '$var определена, даже если она пустая';
}
?>
```

Массив $_GET

http://example.com/my.php?id=1&lang=ru

Запрос

Ассоциативный массив параметров, переданных скрипту через URL.

```
if(isset($_GET['id'])) {
   echo $_GET["id"];// id index exists
}
```

Массив $_POST

Ассоциативный массив данных, переданных скрипту через HTTP метод POST

Тернарный оператор

```php
<?php
// Пример использования тернарного оператора
$action = (empty($_POST['action'])) ? 'default'
  : $_POST['action'];

// Приведенный выше код аналогичен следующему
блоку с использованием if/else
if (empty($_POST['action'])) {
     $action = 'default';
} else {
     $action = $_POST['action'];
}

?>
```

Функции

```
<?php
function foo($arg_1, $arg_2, /* ..., */
 $arg_n)
{
    echo "Example function.\n";
    return $retval;
}
?>
```

Использование значений по умолчанию

```
<?php
function makecoffee($type = "капуччино")
{
    return "Готовим чашку $type.\n";
}
echo makecoffee();
echo makecoffee(null);
echo makecoffee("эспрессо");
?>
```

При использовании нескольких аргументов, все аргументы, для которых установлены значения по умолчанию, должны находиться правее аргументов, для которых значения по умолчанию не заданы

Возврат нескольких значений

- Функция не может возвращать несколько значений, но аналогичного результата можно добиться, возвращая массив.

```
<?php
function small_numbers()
{
    return array (0, 1, 2);
}
list ($zero, $one, $two) = small_numbers();
?>
```