

# Встроенные ВОЗМОЖНОСТИ ЯЗЫКА

Работа с формами, сеансы, работа с файловой системой, HTTP-заголовки



# Суперглобальные массивы

`$GLOBALS` - ссылки на все переменные глобальной области видимости

`$_ENV` - переменные окружения

`$_SERVER` - информация о сервере и среде исполнения

`$_COOKIE` - HTTP Куки

`$_SESSION` - переменные сессии

`$_FILES` - переменные файлов, загруженных по HTTP

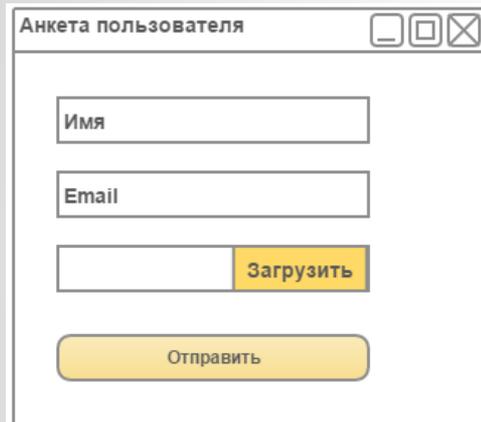
`$_GET` - HTTP GET переменные

`$_POST` - HTTP POST переменные

`$_REQUEST` - ассоциативный массив (array), который по умолчанию содержит данные переменных `$_GET`, `$_POST` и `$_COOKIE`



# Передача данных формами



Анкета пользователя

Имя

Email

Загрузить

Отправить

## Методы

- | GET
- | POST

```
<form action="..." method="...">
```

Логин:

```
<input name="login" type="text">
```

Пароль:

```
<input name="pwd" type="password">
```

```
<input type="submit">
```

```
</form>
```



# Прием данных

Для данных, переданных в строке запроса

- | `$_GET['login'];`
- | `$_GET['pwd'];`

Для данных, переданных в теле запроса

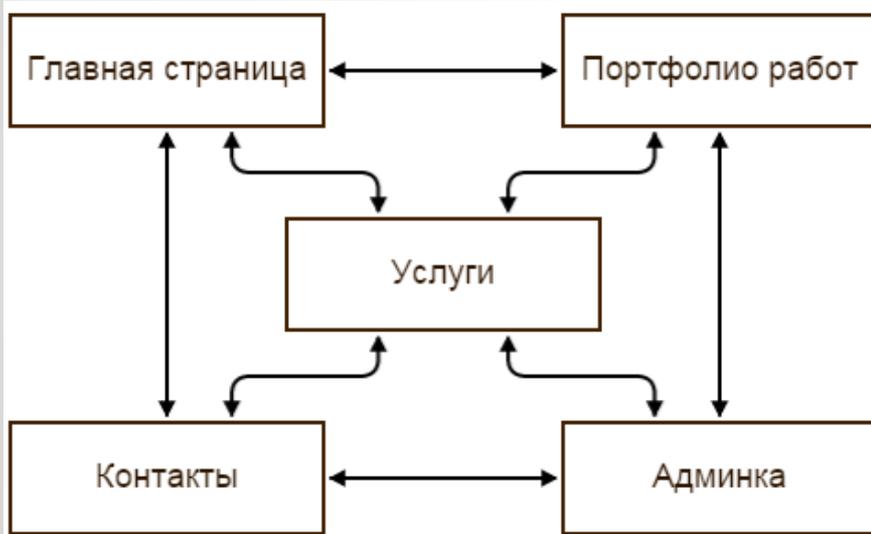
- | `$_POST['login'];`
- | `$_POST['pwd'];`

# Демонстрация



**LoftSchool**  
ОТ МЫСЛИТЕЛЯ К СОЗДАТЕЛЮ

# Работа с сессиями



# Работа с сессиями

```
<?php
```

```
// открытие сессии  
session_start();
```

```
// добавить данные в сессию  
$_SESSION['name'] = "Dmitry";
```

страница index.php

```
<?php
```

```
// открытие сессии  
session_start();
```

```
// взять данные из сессии  
echo $_SESSION['name'];
```

```
// удалить из сессии  
unset($_SESSION['name']);
```

страница about.php

```
<?php
```

```
// открытие сессии  
session_start();
```

```
// очищаем сессию  
session_destroy();
```

```
//Удаляем сессионные куки  
//для полного удаления  
сессии  
setcookie(session_name(), "");
```

страница logout.php



# Демонстрация



**LoftSchool**  
ОТ МЫСЛИТЕЛЯ К СОЗДАТЕЛЮ

# Работа с HTTP заголовками

- | **header** — отправка HTTP заголовка
- | **header\_remove** — удаляет предварительно установленные заголовки
- | **headers\_list** - возвращает список переданных (или готовых к передаче) заголовков
- | **http\_response\_code** — получает или устанавливает код ответа HTTP



# HTTP заголовки - перенаправления

```
header("Location: http://loftschool.com");  
exit;
```

```
header("HTTP/1.1 301 Moved Permanently");  
header("Location: http://loftschool.com");
```

# HTTP заголовки - перезапрос

// Перенаправление через определенное время - перезапрос  
`header('Refresh: 3; url=http://loftschool.com');`



# HTTP заголовки - установка типа содержимого

- | `header("Content-Type: text/xml");`
- | `header("Content-Type: text/html; charset=utf-8");`
- | `header("Content-Type: text/plain");`
- | `header("Content-Disposition: attachment; filename=\"myfile.txt\")`;



# HTTP заголовки - запрет и разрешение кеширования

// Запрет кеширования

```
header("Cache-Control: no-store; max-age=0");
```

// Разрешение кеширования

```
header("Cache-Control: max-age=3600");
```



# HTTP заголовки - опции кеширования

**max-age=[секунды]** — описывает максимальный период времени, в течение которого контент остается свежим. [секунды] — количество секунд от момента запроса, в течение которых вы хотите, чтобы контент трактовался как свежий.

**s-maxage=[секунды]** — подобен max-age, отличаясь тем, что применяется только к общему кэшу (т.е. прокси).

**public** — помечает авторизованные запросы, как кэшируемые; это нормально, если требуется HTTP-аутентификация, ответы автоматически становятся приватными.

**private** — позволяет кэшу, который действует для определенного пользователя (т.е. в браузере) хранить ответ; общему кэшу (т.е. прокси) — нет.

**no-cache** — принуждает кэш отправлять запрос на исходный сервер каждый раз для валидации, прежде чем выдать кэшированную копию. Это полезно, когда необходимо гарантировать, что аутентификация принята во внимание (в сочетании с public) или для поддержания жесткой свежести без потери преимуществ кэширования.



# HTTP заголовки - опции кеширования

**no-store** — указывает кэшу не сохранять копию контента, ни при каких условиях.

**must-revalidate** — сообщает кэшу, что он должен подчиниться любой свежей информации, что вы ему предоставляете о контенте. HTTP позволяет кэшу хранить устаревший контент при определенных условиях; упомянув этот заголовок, вы сообщаете кэшу, что вы хотите, чтобы он строго следовал вашим правилам.

**proxy-revalidate** — подобен **must-revalidate**, кроме того, что применяется только к прокси.

# Демонстрация



**LoftSchool**  
ОТ МЫСЛИТЕЛЯ К СОЗДАТЕЛЮ

# Работа с файловой системой

- | **bool file\_exists** (string \$filename) - Проверить наличие указанного файла или каталога
- | **int filesize** (string \$filename) - Получить размер файла
- | **int fileatime** (string \$filename) - Получить время последнего доступа к файлу
- | **int filemtime** (string \$filename) - Получить время последнего изменения файла
- | **bool is\_readable** (string \$filename) - Определяет, доступен ли файл для чтения
- | **bool is\_writable** (string \$filename) - Определяет, доступен ли файл для записи



# Файлы - режимы работы

- | **'r'** - Открывает файл только для чтения; помещает указатель в начало файла.
- | **'r+'** - Открывает файл для чтения и записи; помещает указатель в начало файла.
- | **'w'** - Открывает файл только для записи; помещает указатель в начало файла и обрезает файл до нулевой длины. Если файл не существует - пробует его создать.
- | **'w+'** - Открывает файл для чтения и записи; помещает указатель в начало файла и обрезает файл до нулевой длины. Если файл не существует - пытается его создать.



# Файлы - режимы работы

- | **'a'** - открывает файл только для записи; помещает указатель в конец файла. Если файл не существует - пытаются его создать.
- | **'a+'** - открывает файл для чтения и записи; помещает указатель в конец файла. Если файл не существует - пытаются его создать.
- | **'x'** - создаёт и открывает только для записи; помещает указатель в начало файла. Если файл уже существует, вызов `fopen()` закончится неудачей, вернёт `FALSE` и выдаст ошибку уровня `E_WARNING`. Если файл не существует, попытается его создать.
- | **'x+'** - создаёт и открывает для чтения и записи; иначе имеет то же поведение что и `'x'`.



# Файлы - открываем и закрываем файлы

// Открываем файл

```
resource fopen ( string $filename , string $mode )
```

```
$handle = fopen("file.txt", "r");
```

```
$handle = fopen("/docs/file.gif", "wb");
```

```
$handle = fopen("http://www.site.ru/", "r");
```

```
$handle = fopen("ftp://user:a@b.ru/file.txt", "w");
```

// Закрываем файл

```
fclose($handle);
```



# Файлы - читаем из файла

- | `string fread ( resource $handle , int $length )` - читает до `length` байтов из файлового указателя `handle`
- | `int fpassthru ( resource $handle )` - читает указанный файловый указатель с текущей позиции до конца файла и записывает результат в буфер вывода. **Возвращает количество прочтенных символов из файла**
- | `string fgets ( resource $handle [, int $length ] )` - читает строку из файла
- | `string fgetc ( resource $handle )` - считывает символ из файла



# Файлы - пишем в файл

- | `int fwrite ( resource $handle , string $string [, int $length ] )` - записывает содержимое `string` в файловый поток `handle`. Если передан аргумент `length`, запись остановится после того, как `length` байтов будут записаны или будет достигнут конец строки `string`, смотря что произойдёт первым. Возвращает количество записанных байтов или `FALSE` в случае ошибки
- | `fputs` - псевдоним `fwrite()`

# Файлы - манипуляции курсором

- | `int fseek ( resource $handle , int $offset [, int $whence ] )` - устанавливает смещение в файле
- | `int ftell ( resource $handle )` - сообщает текущее смещение чтения/записи файла
- | `bool rewind ( resource $handle )` - сбрасывает курсор у файлового указателя
- | `bool feof ( resource $handle )` - проверяет, достигнут ли конец файла



# Файлы - прямая работа с файлами

- | `int readfile ( string $filename )` - читает файл и записывает его в буфер вывода
- | `array file ( string $filename )` - читает содержимое файла и помещает его в массив построчно
- | `string file_get_contents ( string $filename )` - получить содержимое файла в виде одной строки
- | `int file_put_contents ( string $filename )` - записать строку в файл. Если необходимо дописать данные в файл - используйте константу `FILE_APPEND`

# Файлы - управление файлами

- | bool **copy** ( string \$source , string \$dest ) - копирует файл
- | bool **rename** ( string \$oldname , string \$newname ) - переименовывает файл или директорию
- | bool **unlink** ( string \$filename ) - удаляет файл



# Демонстрация



**LoftSchool**  
ОТ МЫСЛИТЕЛЯ К СОЗДАТЕЛЮ

# Работа с директориями

| **bool mkdir** ( string \$pathname [, int \$mode ] ) - пытается создать директорию, заданную в `pathname`. Аргумент `mode` необходимо задавать в виде восьмиричного числа. По умолчанию `mode` равен `0777`

| **bool rmdir** ( string \$dirname [, resource \$context ] ) - пытается удалить директорию с именем `dirname`. Директория должна быть пустой и должны иметься необходимые для этого права.

| **string getcwd** ( void ) - Возвращает имя текущего рабочего каталога.



# Работа с директориями

| **resource** opendir ( string \$path ) - возвращает дескриптор каталога

| **void** closedir ( resource \$dir\_handle ) - закрывает поток, связанный с каталогом

| **string** readdir ( resource \$dir\_handle ) - возвращает имя следующего по порядку элемента каталога.

| **bool** is\_file ( string \$filename ) - возвращает TRUE, если filename существует.

| **bool** is\_dir ( string \$filename ) - возвращает TRUE, если filename директория.

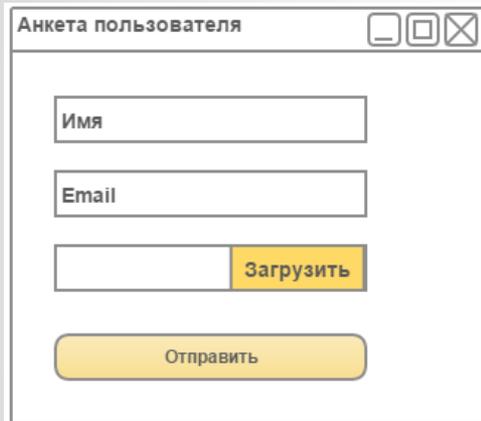


# Работа с директориями

- | `array scandir` ( `string $directory` [, `integer $sorting_order` ] ) - возвращает array, содержащий имена файлов и каталогов, расположенных по пути, переданном в параметре `directory`.
- | `array glob` ( `string $pattern` [, `int $flags = 0` ] ) - ищет все пути, совпадающие с шаблоном `pattern`.



# Загрузка файлов на сервер



Анкета пользователя

Имя

Email

Загрузить

Отправить

```
<form action="upload.php" enctype="multipart/form-data"  
method="post">  
  <input type="text" name="name" required/>  
  ...  
  <input type="email" name="email" required/>  
  ...  
  <input type="file" name="avatar"/>  
</form>
```



# Загрузка файлов на сервер

**`$_FILES['avatar']['name']`** - оригинальное имя файла, такое, каким его видел пользователь, выбирая файл;

**`$_FILES['avatar']['type']`** - mime/type файла, к примеру, может быть image/gif; это поле полезно сохранить, если Вы хотите предоставлять интерфейс для скачивания загруженных файлов;

**`$_FILES['avatar']['size']`** - размер загруженного файла;

**`$_FILES['avatar']['tmp_name']`** - полный путь к временному файлу на диске;

**`$_FILES['avatar']['error']`** - Код ошибки, который равен 0, если операция прошла успешно.



# Ошибки при загрузке файлов

**UPLOAD\_ERR\_OK [0]** - Ошибок нет.

**UPLOAD\_ERR\_INI\_SIZE [1]** - Размер принятого файла превысил максимально допустимый размер.

**UPLOAD\_ERR\_FORM\_SIZE [2]** - Размер загружаемого файла превысил значение `MAX_FILE_SIZE`.

**UPLOAD\_ERR\_PARTIAL [3]** - Загружаемый файл был получен только частично.

**UPLOAD\_ERR\_NO\_FILE [4]** - Файл не был загружен.

**UPLOAD\_ERR\_NO\_TMP\_DIR [6]** - Отсутствует временная папка.

**UPLOAD\_ERR\_CANT\_WRITE [7]** - Не удалось записать файл на диск.



# Перемещение загруженного файла

**bool** `move_uploaded_file ( string $filename , string $destination )` - проверяет, является ли файл `filename` загруженным на сервер (переданным по протоколу HTTP POST). Если файл действительно загружен на сервер, он будет перемещён в место, указанное в аргументе `destination`. Если `filename` не является загруженным файлом, никаких действий не предпринимается и функция возвращает `FALSE`. Если `filename` является загруженным файлом, но не может быть перемещён по каким-либо причинам, никакие действия не предпринимаются и функция возвращает `FALSE`.



# Демонстрация



**LoftSchool**  
ОТ МЫСЛИТЕЛЯ К СОЗДАТЕЛЮ

# Что нужно сделать после вебинара?

1. Пересмотреть запись вебинара
2. Прочитать методичку
3. Продолжить чтение книг из списка “[рекомендованной литературы](#)”
4. Задавать свои вопросы в общем чате
5. Ожидать следующий вебинар

