|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №5**

по дисциплине «Разработка серверных частей интернет-ресурсов»

**Студент группы** ИКБО-30-20 Бессмертных И.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель практической работы** преподаватель Благирев М.М.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Москва 2022г.

# Оглавление

[Оглавление 2](#_Toc87704203)

[Цель работы 3](#_Toc87704204)

[Ход работы 3](#_Toc87704205)

[Вывод 16](#_Toc87704206)

[Ответы на вопросы к практической работе 16](#_Toc87704207)

[Список использованной литературы 54](#_Toc87704208)

# Цель работы

Используя серверную конфигурацию, разработанную в первой работе, научиться работать с сессией, файлами и БД Redis.

# Ход работы

На основе серверной конфигурации, разработанной в первой работе, были добавлены настройки и код для работы с сессией, файлами и БД Redis.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1 – Redis

В результате программа способна обеспечить следующую функциональность:

* Хранение данных сессий в БД Redis;
* Формирование индивидуального контента пользователя, включающее запоминание и использование логина пользователя, выбор одной из двух тем и выбор одного из двух доступных языков интерфейса;
* Загрузка файлов на сервер, хранение их в не реляционной БД и выдача их пользователю по запросу.

Программа была запущена и протестирована (Рисунок 3 – 6).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 – страница настроек

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4 – смена языка и темная тема

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5 – страница для загрузки файлов

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 6 – успешная загрузка файла

Файлы работы находятся на GitHub:

https://github.com/Ivanbessmertnykh/Data-Servers

# Вывод

В результате работы была изучена работы с сессией, файлами и БД Redis.

# Ответы на вопросы к практической работе

1. **Что такое сессия в рамках веб-разработки?**

В web-разработке под сессией понимается промежуток времени, в течении которого пользователь находится на сайте. Сессия начинается в момент захода на сайт и заканчивается при закрытии вкладки браузера или при переходе в пределах текущей вкладки на другой ресурс, и позволяет сохранять, например, данные в действиях пользователя, которые не теряются при переходе на другую страницу.

1. Что такое cookie в рамках веб-разработки?

**Ку́ки** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *cookie*, букв. — «печенье») — небольшой фрагмент данных, отправленный [веб-сервером](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) и хранимый на [компьютере](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) пользователя. Веб-клиент (обычно [веб-браузер](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80)) всякий раз при попытке открыть страницу соответствующего сайта пересылает этот фрагмент данных веб-серверу в составе [HTTP](https://ru.wikipedia.org/wiki/HTTP)-запроса.

1. **Опишите механизм использования cookies.**

 Применяется для сохранения данных на стороне пользователя, на практике обычно используется для[]](https://ru.wikipedia.org/wiki/Cookie#cite_note-microsoft-1):

* [аутентификации](https://ru.wikipedia.org/wiki/%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F) пользователя;
* хранения персональных предпочтений и настроек пользователя;
* отслеживания состояния [сеанса](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D0%B0%D0%BD%D1%81_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) доступа пользователя;
* сведения статистики о [пользователях](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C).

Поддержки браузерами cookie (приём, сохранение и последующая пересылка серверу сохранённых cookie) требуют многие сайты с ограничениями доступа, большинство [интернет-магазинов](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD). Настройка оформления и поведения многих [веб-сайтов](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82) по индивидуальным предпочтениям пользователя тоже основана на cookie.

Cookie легко [перехватить](https://ru.wikipedia.org/wiki/%D0%A1%D0%BD%D0%B8%D1%84%D1%84%D0%B5%D1%80) и подменить (например, для получения доступа к учётной записи), если пользователь использует нешифрованное соединение с сервером. В группе риска пользователи, выходящие в интернет при помощи публичных точек доступа [Wi-Fi](https://ru.wikipedia.org/wiki/Wi-Fi) и не использующие при этом таких механизмов, как [SSL](https://ru.wikipedia.org/wiki/SSL) и [TLS](https://ru.wikipedia.org/wiki/TLS). Шифрование позволяет также решить и другие проблемы, связанные с безопасностью передаваемых данных.

Большинство современных браузеров позволяет пользователям выбрать — принимать cookie или нет, но их отключение делает невозможной работу с некоторыми сайтами. Кроме того, по законам некоторых стран (например, согласно постановлению [Евросоюза](https://ru.wikipedia.org/wiki/%D0%95%D0%B2%D1%80%D0%BE%D0%BF%D0%B5%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%81%D0%BE%D1%8E%D0%B7) от 2016 года, см. [общий регламент по защите данных](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%89%D0%B8%D0%B9_%D1%80%D0%B5%D0%B3%D0%BB%D0%B0%D0%BC%D0%B5%D0%BD%D1%82_%D0%BF%D0%BE_%D0%B7%D0%B0%D1%89%D0%B8%D1%82%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)) сайты должны в обязательном порядке запрашивать согласие пользователя перед установкой cookie.

1. **Опишите простой пример работы сессий в PHP.**

Сессии являются простым способом хранения информации для отдельных пользователей с уникальным идентификатором сессии. Это может использоваться для сохранения состояния между запросами страниц. Идентификаторы сессий обычно отправляются браузеру через сессионный cookie и используются для получения имеющихся данных сессии. Отсутствие идентификатора сессии или сессионного cookie сообщает PHP о том, что необходимо создать новую сессию и сгенерировать новый идентификатор сессии.

Сессии используют простую технологию. Когда сессия создана, PHP будет либо получать существующую сессию, используя переданный идентификатор (обычно из сессионного cookie) или, если ничего не передавалось, будет создана новая сессия. PHP заполнит суперглобальную переменную [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php) сессионной информацией после того, как будет запущена сессия. Когда PHP завершает работу, он автоматически сериализует содержимое суперглобальной переменной [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php) и отправит для сохранения, используя сессионный обработчик для записи сессии.

По умолчанию PHP использует внутренний обработчик files для сохранения сессий, который установлен в INI-переменной [session.save\_handler](https://www.php.net/manual/ru/session.configuration.php#ini.session.save-handler). Этот обработчик сохраняет данные на сервере в директории, указанной в конфигурационной директиве [session.save\_path](https://www.php.net/manual/ru/session.configuration.php#ini.session.save-path).

Сессии могут запускаться вручную с помощью функции [session\_start()](https://www.php.net/manual/ru/function.session-start.php). Если директива [session.auto\_start](https://www.php.net/manual/ru/session.configuration.php#ini.session.auto-start) установлена в 1, сессия автоматически запустится, в начале запроса.

Сессия обычно завершает свою работу, когда PHP заканчивает исполнять скрипт, но может быть завершена и вручную с помощью функции [session\_write\_close()](https://www.php.net/manual/ru/function.session-write-close.php).

Пример #1 Регистрация переменной с помощью [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php).

<?php

session\_start();

if (!isset($\_SESSION['count'])) {

$\_SESSION['count'] = 0;

} else {

$\_SESSION['count']++;

}

?>

Пример #2 Отмена объявления переменной с помощью [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php).

<?php

session\_start();

unset($\_SESSION['count']);

?>

НЕ ОЧИЩАЙТЕ [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php) целиком, используя unset($\_SESSION), так как это отключит возможность регистрации сессионных переменных через суперглобальную переменную [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php).

Сессии, использующие файлы (по умолчанию в PHP), блокируют файл сессии сразу при открытии сессии функцией [session\_start()](https://www.php.net/manual/ru/function.session-start.php) или косвенно при указании [session.auto\_start](https://www.php.net/manual/ru/session.configuration.php#ini.session.auto-start). После блокировки, ни один другой скрипт не может получить доступ к этому же файлу сессии, пока он не будет закрыт или при завершении скрипта или при вызове функции [session\_write\_close()](https://www.php.net/manual/ru/function.session-write-close.php).

Скорее всего это станет проблемой для сайтов, которые активно используют AJAX и делают несколько одновременных запросов. Простейшим путём решить эту проблему будет вызов функции [session\_write\_close()](https://www.php.net/manual/ru/function.session-write-close.php) сразу же как только все требуемые изменения в сессии будут сделаны, предпочтительно ближе к началу работы скрипта. Также можно использовать другой механизм сессии, который поддерживает конкурентный доступ.

1. **Опишите способы защиты сессии пользователя.**

Предотвращение — это первая линия защиты: должны быть приняты все возможные меры, чтобы снизить шанс воровства. Однако аутентификационные токены относительно часто подвержены краже из-за того, что передаются ненадёжной стороне (интерфейсу приложения), поэтому обнаружение кражи играет важную роль в безопасности. Существующие методы обнаружения основаны на эвристических алгоритмах вроде отслеживания внезапных изменений IP-адресов и следов браузера (или мобильного устройства), а также маркировка «необычного поведения пользователя». К несчастью, эти методы могут быть неточными, их легко подделать и сложно реализовать. Однако есть [надёжный способ](https://hackernoon.com/the-best-way-to-securely-manage-user-sessions-91f27eeef460) интегрировать обнаружение кражи в поток управления сеансом.

Когда уязвимости сессий публично раскрываются, компании могут заявлять, что нет никаких признаков использования уязвимости. Но они не упоминают, насколько обширно их система способна обнаруживать кражу токенов.

Самый простой способ защиты — использовать HTTPS и защищённые файлы cookie во всех приложениях. Но это не предотвращает абсолютно все атаки. Можно предпринять дополнительные меры, используя открытые/закрытые ключи, которые фиксируются на устройстве. Фронтенд и бэкенд будут обмениваться этими ключами во время инициализации, до того, как пользователь войдёт в систему. Для последующей связи данные токена могут быть зашифрованы с использованием открытых ключей. Это уменьшает уязвимость к транзитным атакам, делая их возможными только при начальном обмене открытым ключом.

Злоумышленник с достаточными ресурсами может непрерывно «угадывать» токены аутентификации, пока одна из попыток не окажется успешной. Это даст ему доступ ко всем украденным токенам. Защититься можно, используя длинные токены с высокой энтропией.

1. **Верно ли, что можно хранить данные сессии в БД?**

Не надо **хранить сессии в базе данных**. Для большинства сайтов достаточно использовать стандартные методы **хранения** PHP, в файлах. На некоторых высоконагруженных проектах используется многосерверная конфигурация.

1. **Опишите жизненный цикл сессии (Рисунок 5).**

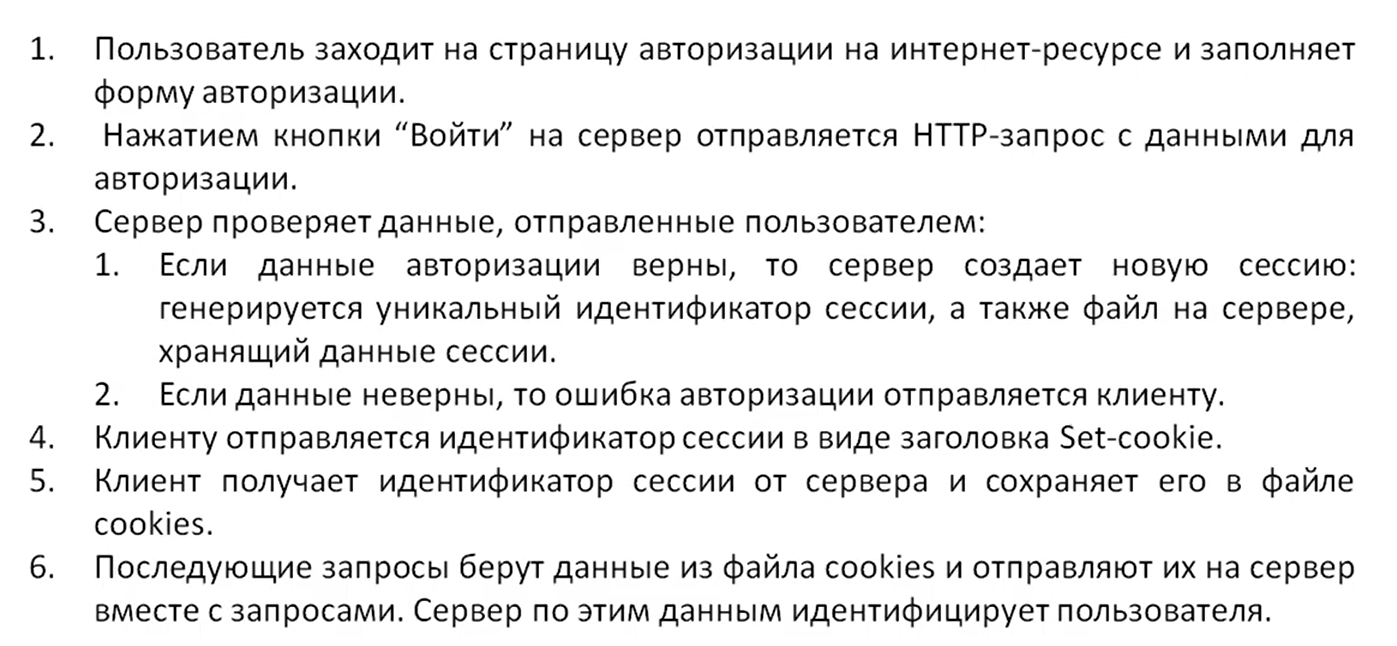


Рисунок 5 - Демонстрация этапов жизненного цикла

1. **Верно ли, что можно убрать механизм обработки сессий?**

Можно. Для работы с сессией сервер должен выдать браузеру некий id, чтобы распознавать его при обработке других запросов. Получается, чтобы это не происходило, нужно просто отключить модуль, отвечающий за это(как например в php), или даже не подлкючать(как например это происходит в Java Spring).

1. **Опишите примеры настройки сессии во время выполнения.**

session.save\_handler string

session.save\_handler определяет имя обработчика, который используется для хранения и извлечения данных, связанных с сессией. По умолчанию имеет значение files. Следует обратить внимание, что некоторые модули могут зарегистрировать собственные обработчики (save\_handler). Текущие зарегистрированные обработчики отображаются в [phpinfo()](https://www.php.net/manual/ru/function.phpinfo.php). Смотрите также [session\_set\_save\_handler()](https://www.php.net/manual/ru/function.session-set-save-handler.php).

session.save\_path string

session.save\_path определяет аргумент, который передается в обработчик сохранения. При установленном по умолчанию обработчике files, аргумент содержит путь, где будут создаваться файлы. Смотрите также [session\_save\_path()](https://www.php.net/manual/ru/function.session-save-path.php).

У этой директивы также существует дополнительный аргумент N, определяющий глубину размещения файлов сессии относительно указанной директории. Например, указание '5;/tmp' может в конечном итоге привести к такому размещению файла сессии: /tmp/4/b/1/e/3/sess\_4b1e384ad74619bd212e236e52a5a174If. Для того, чтобы использовать аргумент N, необходимо предварительно создать все эти директории. Помочь в этом может небольшой скрипт, расположенный в ext/session. Версия для bash называется mod\_files.sh, а Windows-версия - mod\_files.bat. Также следует учитывать, что если N определён и больше 0, то автоматическая сборка мусора не выполняется, подробнее смотрите информацию в файле php.ini. Кроме того, если используется N, необходимо удостовериться, что значение session.save\_path указано в кавычках, поскольку разделитель (;) в php.ini используется как знак комментария.

Модуль хранения файлов создаёт файлы с правами 600 по умолчанию. Это можно изменить с помощью необязательного аргумента MODE: N;MODE;/path, где MODE - восьмеричное представление режима доступа к файлу. Установка MODE не затрагивает umask.

Внимание

Если установить в качестве значения общедоступную для чтения директорию, например, /tmp (по умолчанию), остальные пользователи сервера получат возможность перехватить сессию пользователя, получив список файлов такой директории.

Предостережение

При использовании необязательного аргумента N уровня директорий, как описано выше, учтите, что использование значений выше чем 1 или 2 недопустимо для большинства сайтов, так как требуется очень большое количество директорий: например, значение 3 приводит к (2 \*\* session.sid\_bits\_per\_character) \*\* 3 директориям в файловой системе, которые приводят к огромным потерям места и inode.

Используйте только N больше 2, если вы абсолютно уверены, что ваш сайт достаточно велик, чтобы требовать это.

session.name string

session.name определяет название сессии, используемое как название cookies. Может содержать только цифры и буквы. По умолчанию равно PHPSESSID.

session.auto\_start bool

session.auto\_start определяет, будет ли модуль сессии запускать сессию автоматически при старте. Значение по умолчанию 0 (отключено).

session.serialize\_handler string

session.serialize\_handler определяет имя обработчика, который используется для сериализации/десериализации данных. Поддерживаются формат сериализации PHP (наименование php\_serialize), внутренний формат PHP (наименование php и php\_binary) и WDDX (наименование wddx). WDDX доступен только в том случае, если PHP скомпилирован с [поддержкой WDDX](https://www.php.net/manual/ru/ref.wddx.php). php\_serialize использует простую функцию сериализации/десериализации для внутренних нужд и не имеет тех ограничений, какие есть у php и php\_binary. Старые обработчики сериализации не могут хранить ни числовые, ни строковые индексы, содержащие специальные символы (| и !) в $\_SESSION. Используйте php\_serialize, чтобы обойти ошибки числовых и строковых индексов при завершении скрипта. Значение по умолчанию php.

session.gc\_probability int

session.gc\_probability в сочетании с session.gc\_divisor определяет вероятность запуска функции сборщика мусора (gc, garbage collection). По умолчанию равен 1. Смотрите подробнее в [session.gc\_divisor](https://www.php.net/manual/ru/session.configuration.php#ini.session.gc-divisor).

session.gc\_divisor int

session.gc\_divisor в сочетании с session.gc\_probability определяет вероятность запуска функции сборщика мусора (gc, garbage collection) при каждой инициализации сессии. Вероятность рассчитывается как gc\_probability/gc\_divisor, то есть 1/100 означает, что функция gc запускается в одном случае из ста, или 1% при каждом запросе. session.gc\_divisor по умолчанию имеет значение 100.

session.gc\_maxlifetime int

session.gc\_maxlifetime задаёт отсрочку времени в секундах, после которой данные будут рассматриваться как "мусор" и потенциально будут удалены. Сбор мусора может произойти в течение старта сессии (в зависимости от значений [session.gc\_probability](https://www.php.net/manual/ru/session.configuration.php#ini.session.gc-probability) и [session.gc\_divisor](https://www.php.net/manual/ru/session.configuration.php#ini.session.gc-divisor)).

Замечание: Если разные скрипты имеют разные значения session.gc\_maxlifetime, но при этом одни и те же места для хранения данных сессии, то скрипт с минимальным значением уничтожит все данные. В таком случае следует использовать эту директиву вместе с [session.save\_path](https://www.php.net/manual/ru/session.configuration.php#ini.session.save-path).

session.referer\_check string

session.referer\_check содержит подстроку, которую можно использовать при проверке HTTP Referer. Если клиентом был послан referer и подстрока не была выявлена, то идентификатор сессии будет помечен как недействительный. По умолчанию используется пустая строка.

session.entropy\_file string

session.entropy\_file содержит путь к ресурсу (файлу), используемому как дополнительный источник энтропии в процессе создания идентификатора сессии. Например, /dev/random или /dev/urandom, которые доступны на многих Unix-системах. Эта возможность также поддерживается в Windows. Указание ненулевого значения в session.entropy\_length предписывает PHP использовать в качестве источника энтропии Windows Random API.

Замечание: Удалено в PHP 7.1.0. session.entropy\_file имеет значение по умолчанию равное /dev/urandom или /dev/arandom, если они доступны.

session.entropy\_length int

session.entropy\_length определяет количество байт, которые будут прочитаны из вышеуказанного файла. По умолчанию 32. Удалено в PHP 7.1.0.

session.use\_strict\_mode bool

session.use\_strict\_mode определяет будет ли модуль использовать режим строгого идентификатора (ID). Если от браузера получен неопределённый ID, то браузеру будет отправлен новый ID. Таким образом приложения защищаются от фиксации сессии с помощью строгого режима. По умолчанию 0 (отключено).

Замечание: Включение session.use\_strict\_mode является обязательным для общей безопасности сессии. Всем сайтам рекомендуется её включать. Смотрите примеры [session\_create\_id()](https://www.php.net/manual/ru/function.session-create-id.php).

Внимание

Если пользовательский обработчик сессии, зарегистрированный с помощью [session\_set\_save\_handler()](https://www.php.net/manual/ru/function.session-set-save-handler.php), не реализует SessionUpdateTimestampHandlerInterface::validateId() и не предоставляет callback-функцию validate\_sid, соответственно, режим строгого идентификатора сессии будет отключён, независимо от значения этой директивы. Особо обратите внимание, что [SessionHandler](https://www.php.net/manual/ru/class.sessionhandler.php) не реализует SessionHandler::validateId().

session.use\_cookies bool

session.use\_cookies определяет, будет ли модуль использовать cookies для хранения идентификатора сессии на стороне клиента. По умолчанию 1 (включено).

session.use\_only\_cookies bool

session.use\_only\_cookies определяет, будет ли модуль использовать *только* cookies для хранения идентификатора сессии на стороне клиента. Включение этого параметра предотвращает атаки с использованием идентификатора сессии, размещённого в URL. Значение по умолчанию 1 (включено).

session.cookie\_lifetime int

session.cookie\_lifetime указывает время жизни cookies, отправляемого в браузер клиента, в секундах. Значение 0 означает, что cookies будут валидны до закрытия браузера. По умолчанию равно 0. Смотрите также [session\_get\_cookie\_params()](https://www.php.net/manual/ru/function.session-get-cookie-params.php) и [session\_set\_cookie\_params()](https://www.php.net/manual/ru/function.session-set-cookie-params.php).

Замечание: Отметка окончания времени устанавливается по отношению к серверному времени, которое не обязательно совпадает с временем в браузере клиента.

session.cookie\_path string

session.cookie\_path определяет устанавливаемый путь в сессионной cookie. По умолчанию /. Смотрите также [session\_get\_cookie\_params()](https://www.php.net/manual/ru/function.session-get-cookie-params.php) и [session\_set\_cookie\_params()](https://www.php.net/manual/ru/function.session-set-cookie-params.php).

session.cookie\_domain string

session.cookie\_domain определяет устанавливаемый домен в сессионной cookie. В соответствии со спецификацией нет смысла дополнительно указывать имя хоста, который сгенерировал cookies. Смотрите также [session\_get\_cookie\_params()](https://www.php.net/manual/ru/function.session-get-cookie-params.php) и [session\_set\_cookie\_params()](https://www.php.net/manual/ru/function.session-set-cookie-params.php).

session.cookie\_secure bool

session.cookie\_secure указывает, должны ли cookies передаваться только через защищённое соединение. По умолчанию off. Смотрите также [session\_get\_cookie\_params()](https://www.php.net/manual/ru/function.session-get-cookie-params.php) и [session\_set\_cookie\_params()](https://www.php.net/manual/ru/function.session-set-cookie-params.php).

session.cookie\_httponly bool

Отметка, согласно которой доступ к cookie с идентификатором сессии может быть получен только через HTTP-протокол. Это означает, что cookie не будет доступна через скриптовые языки, такие как JavaScript. Данная настройка позволяет эффективно защитить от XSS-атак (к сожалению, эта функция поддерживается не всеми браузерами).

session.cookie\_samesite string

Позволяет серверам утверждать, что cookie не должен отправляться вместе с межсайтовыми запросами. Это утверждение позволяет браузерам пользователей снизить риск утечки информации из разных источников и обеспечивает определённую защиту от подделки кроссдоменных запросов. Обратите внимание, что это поддерживается не всеми браузерами. Пустое значение означает, что атрибут cookie сайта не будет установлен. Lax и Strict означают, что cookie не будет отправлено для кроссдоменных POST-запросов; Lax отправит cookie для междоменных GET-запросов, а Strict не будет делать этого.

session.cache\_limiter string

session.cache\_limiter определяет режим кеширования, используемого для страниц сессий. Может принимать одно из следующих значений: nocache, private, private\_no\_expire или public. По умолчанию nocache. Подробнее о данных значениях смотрите в [session\_cache\_limiter()](https://www.php.net/manual/ru/function.session-cache-limiter.php).

session.cache\_expire int

session.cache\_expire указывает время жизни кешированных страниц сессий в минутах, это никак не влияет на ограничитель nocache. По умолчанию 180. Смотрите также [session\_cache\_expire()](https://www.php.net/manual/ru/function.session-cache-expire.php).

session.use\_trans\_sid bool

session.use\_trans\_sid указывает, используется ли прозрачная поддержка sid или нет. По умолчанию 0 (отключено).

Замечание: Управление сессией на основе URL имеет дополнительные риски безопасности по сравнению с управлением на основе cookies. В качестве примера можно упомянуть такие ситуации, когда пользователи могут отправить URL, содержащий идентификатор активной сессии, своим друзьям по электронной почте или сохранить ссылку с идентификатором в закладках и все время посещать сайт с одним и тем же идентификатором. Начиная с PHP 7.1.0, полный путь URL, то есть https://php.net/, обрабатывается "trans sid". Раньше PHP обрабатывал только относительный URL-адреса. Перезапись целевого хоста задаётся [session.trans\_sid\_hosts](https://www.php.net/manual/ru/session.configuration.php#ini.session.trans-sid-hosts).

session.trans\_sid\_tags string

session.trans\_sid\_tags задаёт перезаписываемые теги HTML для включения идентификатора сессии когда включена поддержка прозрачных "sid". По умолчанию a=href,area=href,frame=src,input=src,form= form - специальных тег. <input hidden="session\_id" name="session\_name"> добавляется в форму.

Замечание: До PHP 7.1.0 для этого использовался [url\_rewriter.tags](https://www.php.net/manual/ru/outcontrol.configuration.php#ini.url-rewriter.tags). С PHP 7.1.0, fieldset больше не считается за специальный тег.

session.trans\_sid\_hosts string

session.trans\_sid\_hosts задаёт, какие хосты будут перезаписаны для включения идентификатора сессии, когда включена поддержка прозрачных "sid". По умолчанию $\_SERVER['HTTP\_HOST']. Несколько хостов можно указать через запятую. Не допускается вставлять пробелы между хостами. Так правильно: php.net,wiki.php.net,bugs.php.net.

session.sid\_length int

session.sid\_length позволяет указать длину идентификатора сессии. Это значение должно быть в диапазоне 22-256. По умолчанию 32. Если вам нужна совместимость, указывайте 32, 40 и т.д. Более длинные идентификаторы сложнее подобрать. Рекомендуется использовать длину не менее 32.

Подсказка

Заметки по совместимости: Используйте 32 для session.hash\_function=0 (MD5) и session.hash\_bits\_per\_character=4, session.hash\_function=1 (SHA1) и session.hash\_bits\_per\_character=6. 26 для session.hash\_function=0 (MD5) и session.hash\_bits\_per\_character=5. 22 для session.hash\_function=0 (MD5) и session.hash\_bits\_per\_character=6. Вы должны сконфигурировать INI-настройки таким образом, чтобы идентификатор сессии состоял как минимум из 128 бит. Не забудьте задать соответствующие значения для session.sid\_bits\_per\_character, иначе ваши идентификаторы будут слабыми.

Замечание: Эта настройка появилась в PHP 7.1.0.

session.sid\_bits\_per\_character int

session.sid\_per\_character позволяет задать количество бит в одном символе идентификатора сессии. Доступные значения '4' (0-9, a-f), '5' (0-9, a-v), и '6' (0-9, a-z, A-Z, "-", ","). По умолчанию 4. Чем больше бит, тем сильнее идентификатор сессии. В большинстве окружений рекомендуется 5.

Замечание: Эта настройка появилась в PHP 7.1.0.

session.hash\_function [mixed](https://www.php.net/manual/ru/language.types.declarations.php#language.types.declarations.mixed)

session.hash\_function позволяет указать алгоритм хеширования, используемый для генерации идентификатора сессии. '0' означает MD5 (128 bits), а '1' означает SHA-1 (160 bits).

Возможно указать любой из алгоритмов, предусмотренных [модулем hash](https://www.php.net/manual/ru/ref.hash.php) (если он доступен), например, sha512 или whirlpool. Полный список алгоритмов может быть получен с помощью функции [hash\_algos()](https://www.php.net/manual/ru/function.hash-algos.php).

Замечание: Удалена в PHP 7.1.0.

session.hash\_bits\_per\_character int

session.hash\_bits\_per\_character позволяет указать сколько бит хранится в каждом символе при преобразовании бинарного представления во что-либо более удобочитаемое. Возможные значения: '4' (0-9, a-f), '5' (0-9, a-v) и '6' (0-9, a-z, A-Z, "-", ",").

Замечание: Удалена в PHP 7.1.0.

session.upload\_progress.enabled bool

Включает отслеживание прогресса загрузки файлов и заполнение соответствующей переменной в массиве [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php). По умолчанию 1, включено.

session.upload\_progress.cleanup bool

Чистка информации о прогрессе загрузки файлов по завершении обработки POST-данных (то есть когда загрузка завершена). По умолчанию 1, включено.

Замечание: Строго рекомендуется не отключать эту опцию.

session.upload\_progress.prefix string

Префикс, используемый для ключа прогресса загрузки в массиве [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php). Для обеспечения уникальности данный ключ будет присоединён к значению $\_POST[ini\_get("session.upload\_progress.name")]. По умолчанию равен "upload\_progress\_".

session.upload\_progress.name string

Имя ключа, используемого в массиве [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php), для хранения информации о прогрессе. Смотрите также директиву [session.upload\_progress.prefix](https://www.php.net/manual/ru/session.configuration.php#ini.session.upload-progress.prefix). Если элемент $\_POST[ini\_get("session.upload\_progress.name")] не передан, прогресс загрузки данного файла не будет отслеживаться. По умолчанию "PHP\_SESSION\_UPLOAD\_PROGRESS".

session.upload\_progress.freq [mixed](https://www.php.net/manual/ru/language.types.declarations.php#language.types.declarations.mixed)

Определяет частоту обновления информации о прогрессе загрузки. Можно указать значение в байтах (то есть "обновлять информацию о прогрессе каждые 100 байт") или в процентах (то есть "обновлять информацию о прогрессе после получения 1% данных от размера файла"). По умолчанию "1%".

session.upload\_progress.min\_freq int

Минимальная задержка между обновлениями, в секундах. По умолчанию "1" (одна секунда).

session.lazy\_write bool

Если session.lazy\_write установлен в 1, то при изменении данных сессии они будут только перезаписываться. По умолчанию 1, включено.

Прогресс загрузки файлов не будет обрабатываться, если не включена опция session.upload\_progress.enabled и не установлена переменная $\_POST[ini\_get("session.upload\_progress.name")].

1. **Опишите директивы конфигурации файловой системы и потоков в PHP (Рисунок 6).**

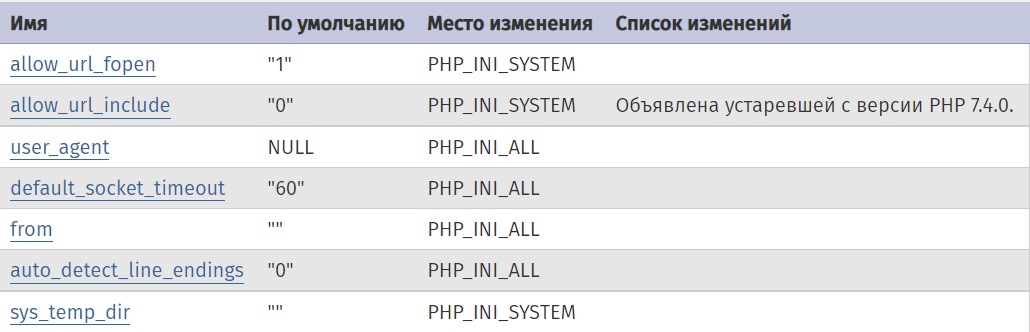


Рисунок 6 - Демонстрация директив конфигурации

Краткое разъяснение конфигурационных директив.

allow\_url\_fopen bool

Данная директива включает поддержку обёрток URL (URL wrappers), которые позволяют работать с объектами URL как с обычными файлами. Обёртки, доступные по умолчанию, служат для работы с [удалёнными файлами](https://www.php.net/manual/ru/features.remote-files.php) с использованием ftp или http протокола. Некоторые модули, например, [zlib](https://www.php.net/manual/ru/ref.zlib.php), могут регистрировать собственные обёртки.

allow\_url\_include bool

Эта опция позволяет использование обёрток fopen, которые поддерживают работу с URL, в функциях [include](https://www.php.net/manual/ru/function.include.php), [include\_once](https://www.php.net/manual/ru/function.include-once.php), [require](https://www.php.net/manual/ru/function.require.php), [require\_once](https://www.php.net/manual/ru/function.require-once.php).

Замечание:

Эта опция требует включения опции allow\_url\_fopen.

user\_agent string

Устанавливает отсылаемую PHP строку "User-Agent".

default\_socket\_timeout int

Значение тайм-аута по умолчанию (в секундах) для потоков, использующих сокеты. Отрицательное значения означает бесконечное время ожидания.

from string

Адрес email, используемый в соединениях FTP без авторизации, а также в качестве значения заголовка From в HTTP соединениях при использовании ftp и http обёрток, соответственно.

auto\_detect\_line\_endings bool

Когда данная директива включена, PHP проверяет данные, получаемые функциями [fgets()](https://www.php.net/manual/ru/function.fgets.php) и [file()](https://www.php.net/manual/ru/function.file.php) с тем, чтобы определить способ завершения строк (Unix, MS-Dos или Macintosh).

Данная директива позволяет PHP взаимодействовать с системами Macintosh, однако, по умолчанию эта директива выключена, поскольку при её использовании возникает (несущественная) потребность в дополнительных ресурсах для определения символа окончания первой строки, а также потому, что программисты, использующие в системах Unix символы возврата каретки в качестве разделителей, столкнутся с обратно-несовместимым поведением PHP.

sys\_temp\_dir string

1. **Какой тип ресурса использует файловая система. Опишите данный тип.**

Файловая система использует поток в качестве ресурса.

Поток — это последовательность байтов. В файловой системе NTFS потоки содержат данные, которые записываются в файл и предоставляют дополнительные сведения о файле, чем атрибуты и свойства. Например, можно создать поток, содержащий ключевые слова поиска, или удостоверение учетной записи пользователя, которая создает файл.

Каждый поток, связанный с файлом, имеет свой собственный размер выделения, фактический размер и допустимую длину данных:

1. **Размер выделения — это объем места на диске, зарезервированный для потока.**
2. **Фактический размер — это число байтов, используемых вызывающим объектом.**
3. **Допустимая длина данных (ВДЛ) — это число байтов, инициализированных из размера выделения для потока.**
4. **Как открыть и закрыть файл с помощью PHP.**

Самый лучший метод для открытия файлов является функция fopen. Функция fopen, даст вам больше вариантов, чем функция readfile().

Во время занятий, будем использовать текстовый файл, "webdictionary.txt",:

AJAX = Асинхронный JavaScript и XML

CSS = Каскадные Таблицы Стилей

HTML = Язык Разметки Гипер Текста

PHP = PHP Гипертекстовый Препроцессор

SQL = Язык Структурированных Запросов

SVG = Масштабируемая Векторная Графика

XML = Расширенный Язык Разметки

Первый параметр fopen содержит, имя открываемого файла, а второй параметр указывает, в каком режиме должен быть открыт файл. Следующий пример также генерирует сообщение, если функции fopen не удается открыть указанный файл:

Файл может быть открыт в одном из следующих режимов(Рисунок 6):

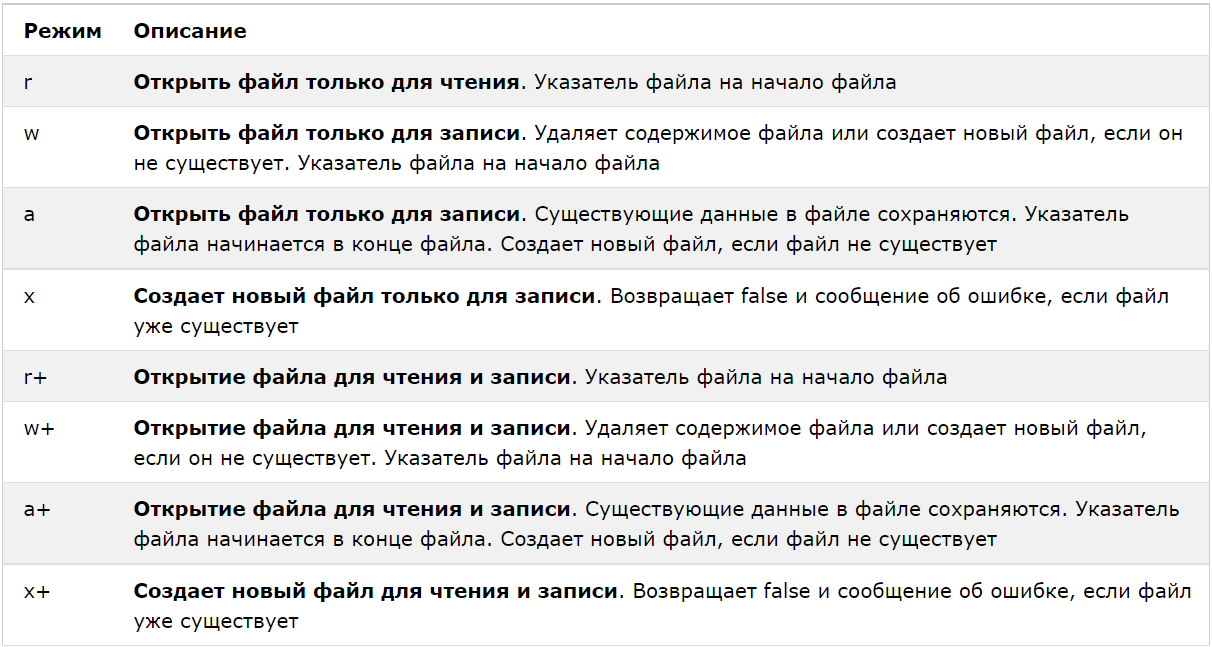


Рисунок 7 - Демонстрация режимов открытия файла

<?php

$myfile = fopen("webdictionary.txt", "r") or die("Не удается открыть файл!");

echo fread($myfile,filesize("webdictionary.txt"));

fclose($myfile);

?>

Функция fclose() используется для закрытия открытого файла.

Это хорошая практика программирования, чтобы закрыть все файлы, после того как закончили работу с ними. Вы не хотите открыть файл на ваш сервер принимает ресурсы!

Функция fclose() требует имя файла (или переменной, содержащей имя файла), который нужно закрыть:

<?php

$myfile = fopen("webdictionary.txt", "r");

// некоторый код для выполнения....

fclose($myfile);

?>

1. Как производится чтение и запись файлов в PHP.

fread() функция читает из открытого файла.

Первый параметр fread() содержит имя файла для чтения и второй параметр указывает максимальное число байтов для чтения.

Следующий код PHP читает файл "Dictionary.txt" до конца:

fread($myfile,filesize("webdictionary.txt"));

Один из самых простых способов записи данных в файл в PHP - с помощью функции file\_put\_contents($filename, $data, $flags, $context).

Параметр $filename определяет файл, в котором будут записываться данные. Второй параметр - это данные, которые вы хотите записать в файл. В большинстве случаев это будет строка, но она также может быть массивом или ресурсом потока.

Параметр $flags определяет, как содержимое будет записано в файл. Он может иметь любое или все из следующих трех значений:

* **FILE\_USE\_INCLUDE\_PATH** - Это указывает PHP на поиск имени файла в каталоге include.
* **FILE\_APPEND** - Это скажет PHP, чтобы добавить данные, которые вы передали функции, к существующим данным в файле. Это может быть полезно, если вы храните данные в файле, таком как журнал или личный дневник. Запись новых данных, таких как температура или события, которые произошли с вами сегодня, не будет перезаписывать то, что вы записали вчера.
* **LOCK\_EX** - Это скажет PHP, чтобы получить блокировку файла, прежде чем начинать писать в него содержимое. Это может предотвратить непредвиденные события, когда два разных сценария читают или записывают данные в один и тот же файл. С этим конкретным значением вы получите эксклюзивную блокировку файла. Вы можете узнать больше об этих блокировках в документации PHP о функции [flock()](https://php.net/manual/en/function.flock.php).

Эта функция возвращает количество байтов, которые были записаны в файл с успехом, и false при ошибке. Тем не менее, вы все равно должны использовать строгий оператор равенства, чтобы проверить, было ли успешным запись содержимого в файл. Это потому, что код, который записывает 0 байтов в файл, будет по-прежнему оцениваться как false.

1. **Опишите как считать только часть файла, как считывать файл последовательно и считать весь файл целиком.**

Для считывания содержимого файла в массив служит функция file(). Файл считывается построчно и каждая строка помещается в отдельный элемент массива. Функция принимает 2 параметра — имя файла и флаг. Флаг определяет режим чтения строк из файла. Например, при чтении можно игнорировать пустые строки и удалять символ перевода строки.

Например, прочитать и вывести в браузер все строки из файла можно следующим способом:

Если нужно просто читать и писать блоки текста, то проще всего воспользоваться парой функций file\_get\_contents() и file\_put\_contents(). Первая функция считывает и возвращает данные из файла в виде строки, а вторая — записывает блок текста в указанный файл. Пользоваться этими функциями очень просто:

В этом примере мы считываем всё содержимое файла myfile.txt, заменяем все вхождения строки «» на нормальное имя и отчество («Иван Петрович»), после чего сохраняем изменённые данные обратно в файл. Пользоваться этими функциями очень удобно, когда надо считать всё содержимое файла (например, фрагмент страницы или шаблон письма электронной почты).

1. **Как производится создание и удаление файлов с помощью PHP.**

* tempnam — Создаёт файл с уникальным именем
* tmpfile — Создаёт временный файл
* unlink — Удаляет файл

Также можно открыть требуемый файл, при отсутствии он создастся:

$f = fopen("file.txt", "w");

1. **С помощью каких функций и какую информацию о файле можно получить с помощью PHP?**

Функции файловой системы:

* basename — Возвращает последний компонент имени из указанного пути
* chgrp — Изменяет группу файла
* chmod — Изменяет режим доступа к файлу
* chown — Изменяет владельца файла
* copy — Копирует файл
* dirname — Возвращает имя родительского каталога из указанного пути
* disk\_free\_space — Возвращает размер доступного пространства в каталоге или файловой системе
* disk\_total\_space — Возвращает общий размер файловой системы или раздела диска
* fclose — Закрывает открытый дескриптор файла
* feof — Проверяет, достигнут ли конец файла
* fflush — Сбрасывает буфер вывода в файл
* fgetc — Считывает символ из файла
* fgets — Читает строку из файла
* file\_exists — Проверяет существование указанного файла или каталога
* file\_get\_contents — Читает содержимое файла в строку
* file\_put\_contents — Пишет данные в файл
* file — Читает содержимое файла и помещает его в массив
* filegroup — Получает идентификатор группы файла
* filemtime — Возвращает время последнего изменения файла
* fileowner — Возвращает идентификатор владельца файла
* fileperms — Возвращает информацию о правах на файл
* filesize — Возвращает размер файла
* filetype — Возвращает тип файла
* fopen — Открывает файл или URL
* ftruncate — Урезает файл до указанной длины
* is\_executable — Определяет, является ли файл исполняемым
* rename — Переименовывает файл или директорию
* stat — Возвращает информацию о файле
* и т.д.

1. **Что такое DOM?**

Браузер, когда запрашивает страницу и получает в ответе от сервера её исходный HTML-код, должен сначала его разобрать. В процессе анализа и разбора HTML-кода браузер строит на основе него DOM-дерево.

После выполнения этого действия и ряда других браузер приступает к отрисовке страницы. В этом процессе он, конечно, уже использует созданное им DOM-дерево, а не исходный HTML-код. Можно проиллюстрировать рисунком 7 ниже.

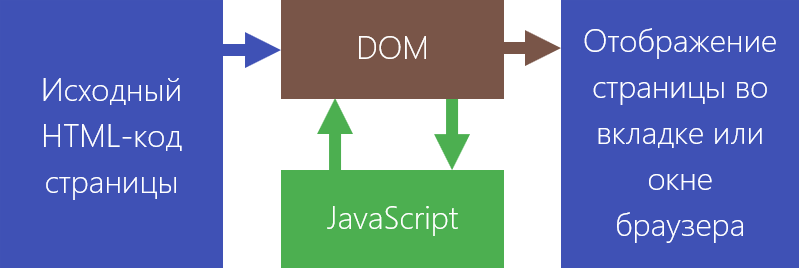


Рисунок 8

DOM – это объектная модель документа, которую браузер создаёт в памяти компьютера на основании HTML-кода, полученного им от сервера.

Если сказать по-простому, то HTML-код – это текст страницы, а DOM – это набор связанных объектов, созданных браузером при парсинге её текста.

1. **Как создать документ и работать с ним с помощью модуля DOM?**

При создании сценария с использованием элемента <script>, либо включая в веб страницу инструкцию для загрузки скрипта, вы можете немедленно приступить к использованию программного интерфейса (API), используя элементы document или window для взаимодействия с самим документом, либо для получения потомков этого документа, т.е. различных элементов на странице.

В соответствии с объектной моделью документа («Document Object Model», коротко DOM), каждый HTML-тег является объектом. Вложенные теги являются «детьми» родительского элемента. Текст, который находится внутри тега, также является объектом.

Все эти объекты доступны при помощи JavaScript, мы можем использовать их для изменения страницы.

Например, document.body – объект для тега <body>.

В следующем примере внутри элемента <script> определён код JavaScript, данный код устанавливает функцию при загрузке документа (когда весь DOM доступен для использования). Эта функция создаёт новый элемент H1, добавляет текст в данный элемент, а затем добавляет H1 в дерево документа:

<html>

<head>

<script>

// запуск данной функции при загрузке документа

window.onload = function() {

// создание нескольких элементов

// в пустой HTML странице

heading = document.createElement("h1");

heading\_text = document.createTextNode("Big Head!");

heading.appendChild(heading\_text);

document.body.appendChild(heading);

}

</script>

</head>

<body>

</body>

</html>

1. **Что такое JSON?**

JSON (англ. JavaScript Object Notation) — текстовый формат обмена данными, основанный на JavaScript. Но при этом формат независим от JS и может использоваться в любом языке программирования. JSON используется в REST API. JSON-объект — это неупорядоченное множество пар «ключ: значение».

1. **Как декодировать строку JSON и вернуть JSON-представление данных?**

mixed json\_decode ( string $json [, bool $assoc = false [, int $depth = 512 [, int $options = 0 ]]] ) — принимает закодированную в JSON строку и преобразует ее в переменную PHP.

string json\_encode ( mixed $value [, int $options = 0 [, int $depth = 512 ]] ) — возвращает строку, содержащую JSON-представление value.

1. **Как проанализировать и выявить ошибки при кодировании и декодировании JSON?**

Функция json\_last\_error возвращает последнюю ошибку, возникшую во время кодирования и декодирования JSON. Таким образом, самый быстрый способ проверить действительный JSON.

// decode the JSON data

// set second parameter boolean TRUE for associative array output.

$result = json\_decode($json);

if (json\_last\_error() === JSON\_ERROR\_NONE) {

// JSON is valid

}

// OR this is equivalent

if (json\_last\_error() === 0) {

// JSON is valid

}

Тестирование с помощью действительного JSON INPUT, покажет данные в виде массива, либо сообщит об ошибке:

$json = '[{"user\_id":13,"username":"stack"},{"user\_id":14,"username":"over"}]';

$output = json\_validate($json);

print\_r($output);

Полная программа для проверки точной ошибки:

function json\_validate($string)

{

// decode the JSON data

$result = json\_decode($string);

// switch and check possible JSON errors

switch (json\_last\_error()) {

case JSON\_ERROR\_NONE:

$error = ''; // JSON is valid // No error has occurred

break;

case JSON\_ERROR\_DEPTH:

$error = 'The maximum stack depth has been exceeded.';

break;

case JSON\_ERROR\_STATE\_MISMATCH:

$error = 'Invalid or malformed JSON.';

break;

case JSON\_ERROR\_CTRL\_CHAR:

$error = 'Control character error, possibly incorrectly encoded.';

break;

case JSON\_ERROR\_SYNTAX:

$error = 'Syntax error, malformed JSON.';

break;

// PHP >= 5.3.3

case JSON\_ERROR\_UTF8:

$error = 'Malformed UTF-8 characters, possibly incorrectly encoded.';

break;

// PHP >= 5.5.0

case JSON\_ERROR\_RECURSION:

$error = 'One or more recursive references in the value to be encoded.';

break;

// PHP >= 5.5.0

case JSON\_ERROR\_INF\_OR\_NAN:

$error = 'One or more NAN or INF values in the value to be encoded.';

break;

case JSON\_ERROR\_UNSUPPORTED\_TYPE:

$error = 'A value of a type that cannot be encoded was given.';

break;

default:

$error = 'Unknown JSON error occured.';

break;

}

if ($error !== '') {

// throw the Exception or exit // or whatever :)

exit($error);

}

// everything is OK

return $result;

}

1. **Опишите создание, сохранение, парсинг XML-документа c помощью PHP.**

Пусть XML хранится в строке. В этом случае из этой строки нужно создать объект с помощью new SimpleXMLElement:

$str = "<?xml version='1.0'?>

<worker>

<name>Коля</name>

<age>25</age>

<salary>1000</salary>

</worker>";

$xml = new SimpleXMLElement($str);

Если же XML хранится в файле или отдается по обращению к URL (что чаще всего и бывает), то следует использовать функцию simplexml\_load\_file, которая делает тот же объект $xml:

<?xml version='1.0'?>

<worker>

<name>Коля</name>

<age>25</age>

<salary>1000</salary>

</worker>

$xml = simplexml\_load\_file(путь к файлу или урл);

В примерах ниже наш XML хранится в файле или по URL.

Пусть дан следующий XML:

<?xml version='1.0'?>

<worker>

<name>Коля</name>

<age>25</age>

<salary>1000</salary>

</worker>

Давайте получим имя, возраст и зарплату работника:

И скрипт на PHP:

$xml = simplexml\_load\_file(путь к файлу или урл);

echo $xml->name; //выведет 'Коля'

echo $xml->age; //выведет 25

echo $xml->salary; //выведет 1000

Пусть некоторые данные хранятся в атрибутах:

<?xml version='1.0'?>

<root>

<worker name="Коля" age="25" salary="1000">Номер 1</worker>

</root>

Тогда обработка будет выглядеть так:

$xml = simplexml\_load\_file(путь к файлу или урл);

echo $xml->worker['name']; //выведет 'Коля'

echo $xml->worker['age']; //выведет 25

echo $xml->worker['salary']; //выведет 1000

echo $xml->worker; //выведет 'Номер 1'

Сохранение xml данных в файл осуществляется при помощи метода save.

<?php

$dom\_xml= new DomDocument();

$dom\_xml->loadXML('<car><model>volvo</model><model>Lada</model></car>');

$path="auto.xml";

echo $dom\_xml->save($path);

?>

1. **Что такое драйвер в рамках взаимодействия с СУБД?**

Такой способ манипуляции данными в приложении базируется на применении универсальных механизмов доступа к данным. Универсальный механизм доступа к данным обычно реализован в виде библиотек и дополнительных модулей, называемых драйверами или провайдерами. Библиотеки содержат некий стандартный набор функций или классов, нередко подчиняющийся той или иной спецификации. Дополнительные модули, специфичные для той или иной СУБД, реализуют непосредственное обращение к функциям клиентского API конкретных СУБД.

1. **Опишите добавление записи в рамках использования модуля mysqli для взаимодействия с БД MYSQL**

С использованием объектно-ориентированного подхода добавление будет выглядеть следующим образом:

$sql = "INSERT INTO Users (name, age) VALUES ('Tom', 37)";

if($conn->query($sql)){

echo "Данные успешно добавлены";

}

1. **Что такое постоянное соединение, опишите проблемы данного подхода и его решение в модуле mysqli.**

Постоянные соединения представляют собой связи с базами данных, которые не закрываются при завершении скрипта. При получении запроса на постоянное соединение PHP вначале проверяет, имеется ли идентичное постоянное соединение (которое было открыто при предыдущих обращениях) и, если таковое было найдено, использует его. В случае, если идентичного соединения нет, PHP создаёт новое. Под "идентичным" подразумевается соединение, открытое на том же хосте с таким же именем пользователя и паролем (если они указаны).

Постоянные соединения не предоставляют возможность открывать 'пользовательские сессии' в том же самом соединении, они не предоставляют возможность организовывать более эффективные транзакции, также они не предоставляют множества других полезных возможностей. Фактически, постоянные соединения не предоставляют никакой функциональности, которая была бы невозможна в непостоянных аналогичных соединениях.

В отличие от модуля mysql, в mysqli нет отдельной функции для создания постоянных соединений. Чтобы открыть постоянное соединение, при создании подключения к имени хоста нужно добавить префикс p:.

При использовании постоянных соединений можно столкнуться с проблемой, которая заключается в том, что клиенты могут оставлять такие подключения в непредсказуемом состоянии. Например, клиент ставит блокировку на таблицу, а затем аварийно завершает работу. То есть блокировка снята не будет. Новый клиентский процесс, использующий это подключение повторно, получит его как есть, и вынужден будет провести какую-то очистку подключения, прежде чем начать его использовать. Соответственно, в задачи программиста входит ещё и проверка подобных ситуаций и внедрение кода, осуществляющего такую очистку.

В модуле есть встроенный функционал, осуществляющий очистку соединений и переводящий их в состояние пригодное для использования. Код очистки, реализованный в mysqli включает следующие операции:

* Откат активных транзакций
* Закрытие и удаление временных таблиц
* Снятие блокировки с таблиц
* Сброс переменных сессии
* Закрытие подготовленных запросов (всегда происходит в PHP)
* Закрытие обработчиков
* Снятие блокировок, установленных функцией GET\_LOCK()

Это позволяет быть уверенным в том, что возвращённые из пула соединения готовы к использованию в клиентских процессах.

1. **Опишите основные особенности БД MongoDB**.

Главные особенности MongoDB:

* Это кроссплатформенная документо-ориентированная база данных NoSQL с открытым исходным кодом.
* Она не требует описания схемы таблиц, как в реляционных БД. Данные хранятся в виде коллекций и документов.
* Между коллекциями нет сложных соединений типа JOIN, как между таблицами реляционных БД. Обычно соединение производится при сохранении данных путем объединения документов.
* Данные хранятся в формате BSON (бинарные JSON-подобные документы).
* У коллекций не обязательно должна быть схожая структура. У одного документа может быть один набор полей, в то время как у другого документа — совершенно другой (как тип, так и количество полей).

1. **Опишите процесс добавления новой записи в СУБД MongoDB с помощью соответствующего драйвера.**

Для добавления в коллекцию могут использоваться три ее метода:

* insertOne(): добавляет один документ
* insertMany(): добавляет несколько документов
* insert(): может добавлять как один, так и несколько документов

Итак, добавим один документ:

> db.users.insertOne({"name": "Tom", "age": 28, languages: ["english", "spanish"]})

Если надо добавить ряд документов, то мы можем воспользоваться методом insertMany():

> db.users.insertMany([{"name": "Bob", "age": 26, languages: ["english", "french"]}, {"name": "Alice", "age": 31, languages:["german", "english"]}])

1. **Опишите процесс получения и обработки записей с помощью драйвера MongoDB.**

Наиболее простой способом получения содержимого БД представляет использование функции find. Действие этой функции во многом аналогично обычному запросу SELECT \* FROM Table, который применяется в SQL и который извлекает все строки. Например, чтобы извлечь все документы из коллекции users, созданной в прошлой теме, мы можем использовать команду:

> db.users.find()

Функция find() позволяет возвратить несколько документов. Еще одна функция findOne() работает похожим образом, только возвращает один документ:

> db.users.findOne()

1. **Опишите получение записей в рамках использования модуля mysqli для взаимодействия с БД MYSQL.**

При выполнении команды SELECT метод query() объекта mysqli возвращает набор полученных строк, который мы можем перебрать с помощью цикла. Например, перебор с помощью цикла foreach:

$sql = "SELECT \* FROM Users";

if($result = $conn->query($sql)){

foreach($result as $row){

$userid = $row["id"];

$username = $row["name"];

$userage = $row["age"];

}

}

1. **Опишите поиск записей, подсчет и ограничение выборки с помощью**

**драйвера MongoDB**

Выведем все документы, в которых name=Tom:

> db.users.find({name: "Tom"})

Теперь более сложный запрос: нам надо вывести те объекты, у которых name=Tom и одновременно age=32. То есть на языке SQL это могло бы выглядеть так: SELECT \* FROM Table WHERE Name='Tom' AND Age=32. Данному критерию у нас соответствует последний добавленный объект. Тогда мы можем написать следующий запрос:

> db.users.find({name: "Tom", age: 32})

Подсчет элементов в коллекции можно осуществлять с помощью функции:

> db.collection.count()

Но данная функция возвращает объект типа Promise, поэтому для получения целочисленного значения следует использовать примерно следующую конструкцию:

let coll = db.collection('collection\_name');

coll.count().then((count) => {

console.log(count);

});

MongoDB представляет ряд функций, которые помогают управлять выборкой из бд. Одна из них - функция limit. Она задает максимально допустимое количество получаемых документов. Количество передается в виде числового параметра. Например, ограничим выборку тремя документами:

> db.users.find().limit(3)

Но что, если мы хотим произвести выборку не сначала, а пропустив какое-то количество документов? В этом нам поможет функция skip. Например, пропустим первые три записи:

> db.users.find().skip(3)

MongoDB предоставляет возможности отсортировать полученный из бд набор данных с помощью функции sort. Передавая в эту функцию значения 1 или -1, мы можем указать в каком порядке сортировать: по возрастанию (1) или по убыванию (-1). Во многом эта функция по действию аналогична выражению ORDER BY в SQL. Например, сортировка по возрастанию по полю name:

> db.users.find().sort({name: 1})

Ну и в конце надо отметить, что мы можем совмещать все эти функции в одной цепочке:

> db.users.find().sort({name: 1}).skip(3).limit(3)

# Список использованной литературы

1. Документация Docker — Текст : электронный // Docker docs : [Электронный ресурс]. — URL: https://docs.docker.com/ (дата обращения: 21.11.2022).
2. Руководство по php — Текст : электронный // php : [Электронный ресурс]. — URL: https://www.php.net/ (дата обращения: 22.11.2022).
3. Docker Hub — Текст : электронный // Docker Hub : [Электронный ресурс]. — URL: https://hub.docker.com/ (дата обращения: 22.11.2022).
4. Справочник по php — Текст : электронный // Docker Hub : [Электронный ресурс]. — URL: https://php.ru/manual/ (дата обращения: 22.11.2022).