**Вопросы**

*1. Что такое тип данных?*

Тип данных – это множество возможных значений и набор разрешенных операций над этими значениями.

*2. Какие есть типы данных?*

* Простые:
* логический тип (значения могут находиться только в состоянии «ложь» или «истина»);
* целочисленный тип;
* числа с плавающей запятой (вещественные числа);
* символьный тип;
* указатели (переменные, хранящие адресы ячеек памяти);
* перечисляемый тип.
* Составные:
* массив (индексированный набор элементов одного типа);
* строковый тип (последовательность символов);
* структура (группа однотипных и/или логически связанных данных);
* запись (состоит из фиксированного числа значений разного типа);
* стек ([абстрактный тип данных](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), [список элементов](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), организованных по принципу «последним пришёл – первым вышел»);
* очередь ([абстрактный тип данных](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), [список элементов](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), организованных по принципу «первым пришёл – первым вышел»);
* дек (абстрактный тип данных, в котором элементы можно добавлять и удалять как в начало, так и в конец).

*3. PHP строго или не строго типизированный?*

Не строго типизированный.

*4. В чем разница между этими типами?*

При строгой типизации невозможно сочетать в выражениях разные типы данных и осуществлять автоматически неявные преобразования типов.

При нестрогой типизации неявные преобразования осуществляются автоматически.

*5. Можно ли включить строгую типизацию?*

Да. Для включения строгой типизации используется оператор [declare](https://www.php.net/manual/ru/control-structures.declare.php) с объявлением strict\_types (declare(strict\_types=1))

*6. Разница между строгим и нестрогим сравнением?*

Нестрогое сравнение без учета типа данных, а строгое с учетом типа данных.

*7. Как можно защититься от SQL-инъекций*

Не соединяться с БД, используя учётную запись владельца БД. Использовать специально созданных пользователей с максимально ограниченными правами.

Использовать подготовленные выражения с привязанными переменными.

Проверять введённые данные на соответствие ожидаемому типу.

Если на уровне базы данных не поддерживаются привязанные переменные, то использовать экранирующих функций, например, mysql\_real\_escape\_string(), sqlite\_escape\_string().

Не выводить никакой информации о БД и её структуре.

Логировать запросы в скрипте для трассировки взломанного приложения.

*8. Что такое namespace-ы и зачем они нужны?*

Пространства имён используются для:

* избежания конфликта имён между вашим кодом и внутренними классами, функциями или константами.
* создания псевдонимов (сокращений) для длинных имён.

*9. Что такое static функция и чем она отличается от “обычной” (не static)?*

В отличие от обычной, функция static принадлежит классу и вызывается у класса, а не у объекта. К функциям static можно обращаться без создания экземпляра класса.

*10. Какая разница между require(), require once(), include() и include once()?*

include() включает один файл в другой так, если бы содержимое одного файла находилось внутри другого.

require() подключает файл также, как и include(), но при невозможности подключения файла include() покажет ошибку и продолжит работу, а require() покажет ошибку и остановит выполнение скрипта.

При использовании require\_once() и include\_once() запрещено повторное подключение скрипта.

При попытке повторного подключения include\_once() продолжит работу, а require\_once() остановит выполнение скрипта.

*11. Назови по памяти любые функции для работы со строками, массивами и файлами.*

echo – выводит одну или несколько строк

print – выводит строку

join – объединяет элементы массива в строку

str\_split – преобразование строки в массив

explode – разбивает строку на подстроки

array\_pop – извлечение последнего элемента массива

array\_push – добавляет один или несколько элементов в конец массива

sort – сортировка массива по возрастанию

rsort – сортировка массива по убыванию

array\_search – поиск данного значения в массиве

copy – копирование файла

file\_exists – проверка существования файла

fopen – открывает файл

fclose – закрывает файл

delete – удаление файла

*12. Можно ли в PHP создавать экземпляры классов и обращаться к их методам и свойствам зная только название/неймспейс класса?*

Нет

**OOP**

*1. Инкапсуляция/наследование/полиморфизм*

Инкапсуляция – это свойство системы, позволяющее объединить данные и методы в классе и скрыть детали реализации от пользователя.

Наследование – это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью.

Полиморфизм – это свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

*2. Поддерживает ли РНР множественное наследование?*

Нет

*3. Что такое абстрактные классы/методы?*

Абстрактный класс – это базовый класс, который не предполагает создания экземпляров.

Абстрактный метод – это метод класса, реализация для которого отсутствует.

*4. Что такое интерфейсы?*

Интерфейсы – структура программы, позволяющая создавать код, который указывает, какие методы должен реализовать класс, без необходимости определять, как именно они должны быть реализованы.

*5. Что такое traits?*

Трейт – это механизм обеспечения повторного использования кода.

*6. Какие есть области видимости в PHP?*

Область видимости свойств и методов:

* public (общедоступные)
* private (свойство или метод доступен только в собственном классе)
* protected (свойство или метод доступен в классах, расширяющих текущий класс).

*7. Что означает MVC и что делает каждый компонент?*

MVC (Model-View-Controller) – способ организации программного кода, который позволяет разделить логику приложения на три компонента: модель, представление и контроллер таким образом, что модификация каждого компонента может осуществляться независимо.

Модель предоставляет данные и реагирует на команды контроллера, изменяя своё состояние.

Представление отвечает за отображение данных модели пользователю, реагируя на изменения модели.

Контроллер управляет запросами пользователя, оповещая модель о необходимости изменений.

*8. Какие магические методы знаешь? Что это вообще такое?*

Магические методы – это специальные методы, которые переопределяют действие PHP по умолчанию, при выполнении над объектом определённых действий. Например:

[\_\_construct()](https://www.php.net/manual/ru/language.oop5.decon.php#object.construct) – классы, в которых объявлен метод-конструктор, будут вызывать этот метод при каждом создании нового объекта.

[\_\_destruct()](https://www.php.net/manual/ru/language.oop5.decon.php#object.destruct) – деструктор вызывается при освобождении всех ссылок на определённый объект или при завершении скрипта

[\_\_get()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.get) – выполняется при чтении данных из недоступных или несуществующих свойств

[\_\_set()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.set) – выполняется при записи данных в недоступные или несуществующие свойства.

*9. Список всех магических методов:*

[\_\_construct()](https://www.php.net/manual/ru/language.oop5.decon.php#object.construct), [\_\_destruct()](https://www.php.net/manual/ru/language.oop5.decon.php#object.destruct), [\_\_call()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.call), [\_\_callStatic()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.callstatic), [\_\_get()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.get), [\_\_set()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.set), [\_\_isset()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.isset), [\_\_unset()](https://www.php.net/manual/ru/language.oop5.overloading.php#object.unset), [\_\_sleep()](https://www.php.net/manual/ru/language.oop5.magic.php#object.sleep), [\_\_wakeup()](https://www.php.net/manual/ru/language.oop5.magic.php#object.wakeup), [\_\_serialize()](https://www.php.net/manual/ru/language.oop5.magic.php#object.serialize), [\_\_unserialize()](https://www.php.net/manual/ru/language.oop5.magic.php#object.unserialize), [\_\_toString()](https://www.php.net/manual/ru/language.oop5.magic.php#object.tostring), [\_\_invoke()](https://www.php.net/manual/ru/language.oop5.magic.php#object.invoke), [\_\_set\_state()](https://www.php.net/manual/ru/language.oop5.magic.php#object.set-state), [\_\_clone()](https://www.php.net/manual/ru/language.oop5.cloning.php#object.clone) и [\_\_debugInfo()](https://www.php.net/manual/ru/language.oop5.magic.php#object.debuginfo).

*10. В чем разница между абстрактным классом и интерфейсом?*

Абстрактный класс – это класс, у которого не реализован один или более методов.

Интерфейс – это абстрактный класс, у которого не реализован ни один метод.

*11. Самые популярные паттерны проектирования?*

Abstract Factory предоставляет интерфейс для создания семейств взаимосвязанных или взаимозависимых объектов, не специфицируя их конкретных классов.

Фабричный метод делегирует создание объектов наследникам родительского класса.

Builder отделяет конструирование сложного объекта от его представления, так что в результате одного и того же процесса конструирования могут получаться разные представления.

Singleton – гарантирует, что у класса есть только один экземпляр, и предоставляет к нему глобальную точку доступа.

Prototype описывает виды создаваемых объектов с помощью прототипа и создает новые объекты путем его копирования.

**Задание**

Для выполнения запросов API было использовано приложение Postman. Также при разработке использовался OpenServer.

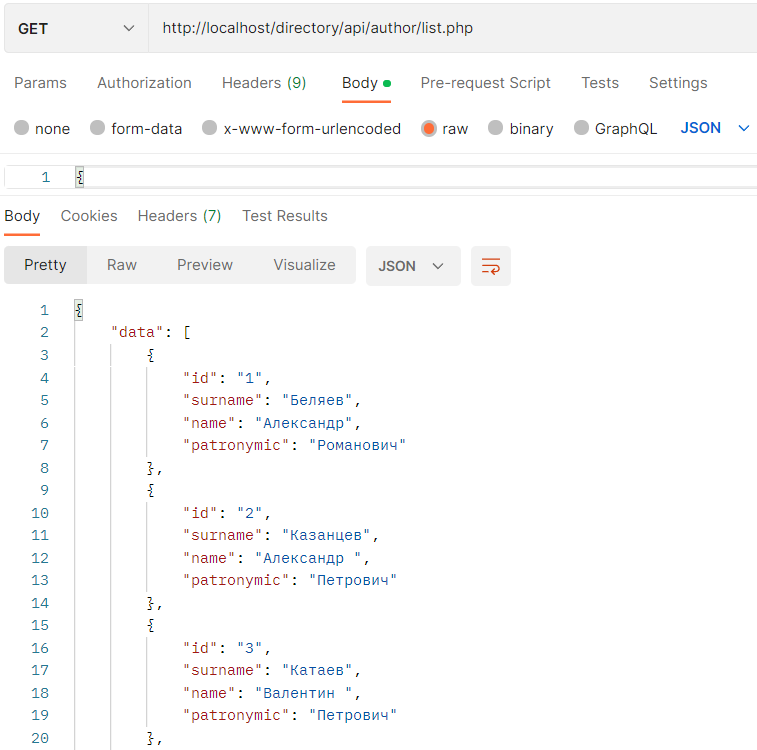
Было создано 2 класса Magazine и Author, соответствующие сущностям «Журнал» и «Автор», а также класс Database для подключения к локальной базе данных на PhpMyAdmin.

Как Magazine, так и Author включают в себя методы:

* list() – вывод всего списка;
* read() – получить одну запись по id;
* create() – создание;
* update() – изменение;
* delete() – удаление.

Чтобы отправить GET-запрос на вывод всего списка (например авторов), нужно обратиться к файлу api/author/list.php и из него будет вызван метод list().

<http://localhost/directory/api/author/list.php>



Чтобы отправить GET-запрос на вывод конкретного автора, нужно обратиться к файлу api/author/read.php и передать параметр id.

<http://localhost/directory/api/author/read.php?id=1>

Чтобы отправить POST-запрос на добавление автора, нужно обратиться к файлу api/author/create.php и в поле raw записать значения.

Изображение выглядит как текст

Автоматически созданное описание

Чтобы отправить POST-запрос на изменение автора, нужно обратиться к файлу api/author/update.php и в поле raw передать значения.

Чтобы отправить DELETE-запрос на удаление автора, нужно обратиться к файлу api/author/delete.php и передать параметр id.

<http://localhost/directory/api/author/delete.php?id=4>

Аналогичные действия с сущностью Журнал (Magazine).