|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА** – **Российский технологический университет»**  **РТУ МИРЭА** | | | | |
| Институт информационных технологий | |
| Кафедра Инструментального и прикладного программного обеспечения (ИиППО) | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ПРАКТИЧЕСКАЯ РАБОТА №6** | | | |
| по дисциплине «Разработка серверных частей интернет-ресурсов» | | | |
| Сторонние модули PHP | | | |
| **Студент группы ИКБО-16-19** | | Григорян А.С.  ­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) | |
|  | |  | |
| Руководитель практической работы | | Преподаватель Волков М.Ю.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| Работа представлена | «\_\_\_»\_\_\_\_\_\_\_2021 г. | | (подпись руководителя) |
| Допущен к работе | «\_\_\_»\_\_\_\_\_\_\_2021 г. | |  |
|  |  | |  |

Москва 2021

Ход работы

[Задание 3](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%206\Babich_IKBO-16-19_Otchet_6_po_RSChIR.docx#_Toc87971777)

[Спецификация 3](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%206\Babich_IKBO-16-19_Otchet_6_po_RSChIR.docx#_Toc87971778)

[Упражнение 3](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%206\Babich_IKBO-16-19_Otchet_6_po_RSChIR.docx#_Toc87971779)

[Вывод 9](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%206\Babich_IKBO-16-19_Otchet_6_po_RSChIR.docx#_Toc87971780)

[Вопросы 9](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%206\Babich_IKBO-16-19_Otchet_6_po_RSChIR.docx#_Toc87971781)

[Список использованной литературы: 24](file:///C:\Users\AramK\OneDrive\Документы\Учеба\РСЧИР\Практика%206\Babich_IKBO-16-19_Otchet_6_po_RSChIR.docx#_Toc87971782)

## Цель работы:

В ходе выполнения добавить стороннюю библиотеку faker. Для тестирования данной библиотеки были сгенерировать наборы фикстур. Для отрисовки результатов тестирования были создать графики с помощью библиотеки PhpChart.

## Задание

Для выполнения каждой части задания предполагается использование различных расширений и сторонних модулей PHP. Предполагается выполнить модификацию разрабатываемого в процессе первых 5 практических работ интернет-ресурса: сгенерировать набор фикстур с помощью выбранного стороннего модуля (каждая фикстура должна содержать не менее 5 полей. Всего фикстур должно быть не менее 50.); на основе этих фикстур построить минимум 3 графика различного типа с помощью сторонних модулей PHP; График превратить в картинку и добавить на него ваш личный полупрозрачный водяной знак с помощью библиотеки GD.

## Спецификация

Все файлы для работы по данной практической работе в папке html. В файле charts.php хранится функция вывода отрисованных графиков на экран. В файле drawcharts.php функции отрисовки диаграмм и добавления на скрины водяных знаков. В файле generatefixtures.php хранится функция генерации фикстур. Все файлы связаны между собой и чтобы вывести диаграммы.

## Упражнение

Код charts.php

<html>

<?php

if (!isset($\_GET["q"]))

die();

$array = unserialize($\_GET["q"]);

for ($i = 0; $i < count($array); $i++)

{

echo $array[$i][0]."<br>";

$data = serialize($array[$i][1]);

echo '<iframe src="/html/drawchart.php?q='.$data.'" width="600" height="420" style="border:none;" scrolling="no">Browser not compatible.</iframe>';

echo "<br>";

} ?></html>

Код drawcharts.php

<html>

<?php

require\_once("phpChart\_Lite/conf.php");

if (!isset($\_GET["q"]))

die();

$data = unserialize($\_GET["q"]);

$pc = new C\_PhpChartX($data,'basic\_chart');

$pc->draw();

?>

<script>

for (let i = 1; i < document.getElementsByClassName("plot jqplot-target").length+1; i++) {

const targetNode = document.getElementById('\_\_chart'+i);

// Options for the observer (which mutations to observe)

const config = { attributes: false, childList: true, subtree: false };

// Callback function to execute when mutations are observed

const callback = function(mutationsList, observer) {

// Use traditional 'for loops' for IE 11

for(const mutation of mutationsList) {

if (mutation.type === 'childList') {

if (mutation.addedNodes.length > 0 && mutation.addedNodes[0].style.zIndex == 9999)

{

if (mutation.addedNodes[0].class != "watermark")

{

console.log(mutation.addedNodes[0]);

mutation.addedNodes[0].style.visibility = "hidden";

}

}

}

}

};

// Create an observer instance linked to the callback function

const observer = new MutationObserver(callback);

// Start observing the target node for configured mutations

observer.observe(targetNode, config);

}

jQuery(document).ready(function (jQuery) {

for (let i = 1; i < document.getElementsByClassName("plot jqplot-target").length+1; i++) {

var chart = document.getElementById("\_\_chart" + i);

var chartcanvas = jQuery("#\_\_chart" + i + " > canvas").first();

var newcanvas = document.createElement("canvas");

newcanvas.width = chartcanvas.width();

newcanvas.height = chartcanvas.height();

newcanvas.class = "watermark";

newcanvas.setAttribute("style", "position:absolute;z-index:9999");

if (chart.firstChild) {

chart.insertBefore(newcanvas, chart.firstChild);

} else {

chart.appendChild(newcanvas);

}

;

var context = newcanvas.getContext("2d");

context.globalAlpha = .4;

context.font = "50px arial";

context.fillStyle = "lightgray";

var textsize = context.measureText("LizzieGri");

var textwidth = textsize.width;

var textheight = 72;

context.translate(chartcanvas.width() / 2, chartcanvas.height() / 2);

context.rotate(-Math.atan(chartcanvas.height() / chartcanvas.width()));

context.fillText("LizzieGri", -textwidth / 2, textheight / 2);

}

});

</script>

</html>

Код generatefixtures.php

<?php

require\_once '/var/www/html/vendor/autoload.php';

$faker = Faker\Factory::create();

$fixtures = array();

for ($i = 0; $i < 50; $i++)

{

$fixtures[$i] = new ProductSellDay($faker);

}

$chartdata = array();

$chartdata[0][0] = "Cost";

$chartdata[0][1] = array();

for ($i = 0; $i < 50; $i++)

{

$chartdata[0][1][$i/10][$i%10] = $fixtures[$i]->Cost;

}

$chartdata[1][0] = "SoldTotal";

$chartdata[1][1] = array();

for ($i = 0; $i < 50; $i++)

{

$chartdata[1][1][$i/10][$i%10] = $fixtures[$i]->SoldTotal;

}

$chartdata[2][0] = "Refunds";

$chartdata[2][1] = array();

for ($i = 0; $i < 50; $i++)

{

$chartdata[2][1][$i/10][$i%10] = $fixtures[$i]->Refunds;

}

$chartdata[3][0] = "TotalIncome";

$chartdata[3][1] = array();

for ($i = 0; $i < 50; $i++)

{

$chartdata[3][1][$i/10][$i%10] = $fixtures[$i]->TotalIncome;

}

$chartdata[4][0] = "TotalIncomeFromAllProducts";

$chartdata[4][1] = array();

for ($i = 0; $i < 50; $i++)

{

$chartdata[4][1][$i%10] += $fixtures[$i]->TotalIncome;

}

$chartdata[4][1] = array($chartdata[4][1]);

$data = serialize($chartdata);

echo '<meta http-equiv="refresh" content="0;url=charts.php?q='.htmlspecialchars($data).'">';

class ProductSellDay

{

public $ProductName;

public $Date;

public $Cost;

public $SoldTotal;

public $Refunds;

public $TotalIncome;

function \_\_construct($faker)

{

$this->ProductName = $faker->word;

$this->Date = $faker->dateTimeBetween('-50 month', '+50 month');

$this->Cost = $faker->numberBetween(1, 1000);

$this->SoldTotal = $faker->numberBetween(1, 10);

$this->Refunds = $faker->numberBetween(0, $this->SoldTotal);

$this->TotalIncome = $this->Cost \* ($this->SoldTotal - $this->Refunds);

}

}

?>

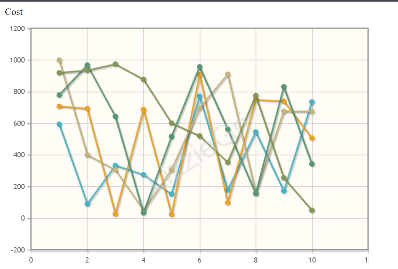


Рис.1-вывод диаграмм

## Вывод

В ходе выполнения была подключена сторонняя библиотека faker. Для тестирования данной библиотеки были сгенерированы наборы фикстур. Для отрисовки результатов тестирования были созданы 5 графиков с помощью библиотеки PhpChart. Также для добавления уникальности в работы, на каждый из графиков был добавлен личный водяной знак.

## Вопросы:

1. Composer - это менеджер для подключения и управления этими сторонними библиотеками или пакетами в PHP-проекте. Еще его называют пакетный менеджер. Composer управляет этими пакетами, чтобы они подключились и хорошо работали.

2. Календарь:

· cal\_days\_in\_month — Возвращает количество дней в месяце для заданного года и календаря

· cal\_from\_jd — Преобразует дату, заданную в юлианском календаре, в дату указанного календаря

· cal\_info — Возвращает информацию о заданном календаре

· cal\_to\_jd — Преобразует заданную дату в юлианскую

· easter\_date — Получить метку времени Unix, соответствующую полуночи на Пасху в заданном году

· easter\_days — Получить количество дней между 21 марта и Пасхой в заданном году

· frenchtojd — Преобразует дату Французского республиканского календаря в количество дней в Юлианском летоисчислении

· gregoriantojd — Преобразует дату по григорианскому календарю в количество дней в юлианском летоисчислении

· jddayofweek — Возвращает день недели

· jdmonthname — Возвращает название месяца

· jdtofrench — Переводит число дней в юлианском летоисчислении в дату по французскому республиканскому календарю

· jdtogregorian — Переводит число дней в юлианском летоисчислении в дату по Григорианскому календарю

· jdtojewish — Переводит количество дней из юлианского календаря в дату по еврейскому календарю

· jdtojulian — Переводит число дней в юлианском летоисчислении в дату по юлианскому календарю

· jdtounix — Переводит число дней в юлианском летоисчислении в метку времени Unix

· jewishtojd — Переводит дату по еврейскому календарю в число дней в юлианском летоисчислении

· juliantojd — Переводит дату по юлианскому календарю в число дней в юлианском летоисчислении

· unixtojd — Переводит метку времени Unix в юлианский день

3. Серверное время - это время региона (часового пояса), где находится сервер.

Dater — определяет часовой пояс, локализует и форматирует время в PHP.

Dater, и его основные возможности:

· Биндинг форматов

· Локализация текстов и форматов

· Расширение списка опций форматирования

· Автоопределение часового пояса

· Конвертация времени с учётом часового пояса

· Автоматическая конвертация времени в $\_GET, $\_POST, $\_REQUEST с учётом часового пояса

· Автоматическая конвертация часового пояса в шаблоне отправляемых данных

4. Опишите способ получения времени заката и рассвета с

использованием языка программирования PHP:

date\_sunrise — Возвращает время рассвета для заданных дня и местоположения

date\_sunset — Возвращает время захода солнца для заданных дня и местоположения

5. Опишите константы, используемые в модуле “Время и дата”:

SUNFUNCS\_RET\_TIMESTAMP (int)

Время в секундах с начала эпохи Unix

SUNFUNCS\_RET\_STRING (int)

Часы:минуты (например: 08:02)

SUNFUNCS\_RET\_DOUBLE (int)

Часы как число с плавающей точкой (например: 8.75)

DateTime::ATOM

DATE\_ATOM

Atom (пример: 2005-08-15T15:52:01+00:00)

DateTime::COOKIE

DATE\_COOKIE

HTTP Cookies (пример: Monday, 15-Aug-05 15:52:01 UTC)

DateTime::ISO8601

DATE\_ISO8601

ISO-8601 (пример: 2005-08-15T15:52:01+0000)

Замечание: Этот формат не совместим с ISO-8601, но остаётся для обратной совместимости. Вместо него используйте DateTime::ATOM или DATE\_ATOM для совместимости с ISO-8601.

DateTime::RFC822

DATE\_RFC822

RFC 822 (пример: Mon, 15 Aug 05 15:52:01 +0000)

DateTime::RFC850

DATE\_RFC850

RFC 850 (пример: Monday, 15-Aug-05 15:52:01 UTC)

DateTime::RFC1036

DATE\_RFC1036

RFC 1036 (пример: Mon, 15 Aug 05 15:52:01 +0000)

DateTime::RFC1123

DATE\_RFC1123

RFC 1123 (пример: Mon, 15 Aug 2005 15:52:01 +0000)

DateTimeInterface::RFC7231

DATE\_RFC7231

RFC 7231 (с версии PHP 7.0.19 и 7.1.5) (пример: Sat, 30 Apr 2016 17:52:13 GMT)

DateTime::RFC2822

DATE\_RFC2822

RFC 2822 (пример: Mon, 15 Aug 2005 15:52:01 +0000)

DateTime::RFC3339

DATE\_RFC3339

Тоже, что и DATE\_ATOM

DateTime::RFC3339\_EXTENDED

DATE\_RFC3339\_EXTENDED

Формат RFC 3339 EXTENDED (пример: 2005-08-15T15:52:01.000+00:00)

DateTime::RSS

DATE\_RSS

RSS (пример: Mon, 15 Aug 2005 15:52:01 +0000)

DateTime::W3C

DATE\_W3C

World Wide Web Consortium (пример: 2005-08-15T15:52:01+00:00)

6. Приведите принципы арифметики даты и времени

Пример #1 DateTimeImmutable::add/sub добавляет интервалы, охватывающие прошедшее время

Добавление PT24H через переход DST приведёт к добавлению 23/25 часов (для большинства часовых поясов).

<?php $dt = new DateTimeImmutable("2015-11-01 00:00:00", new DateTimeZone("America/New\_York")); echo "Начало: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; $dt = $dt->add(new DateInterval("PT3H")); echo "Конец: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; ?>

Результат выполнения данного примера:

Начало: 2015-11-01 00:00:00 -04:00

Конец: 2015-11-01 02:00:00 -05:00

Пример #2 DateTimeImmutable::modify и strtotime увеличит или уменьшит значения индивидуальных компонентов

Добавление +24 часов через переход DST добавит точно 24 часов (вместо учёта перехода на зимнее или летнее время).

<?php $dt = new DateTimeImmutable("2015-11-01 00:00:00", new DateTimeZone("America/New\_York")); echo "Начало: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; $dt = $dt->modify("+24 hours"); echo "Конец: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; ?>

Результат выполнения данного примера:

Начало: 2015-11-01 00:00:00 -04:00

Конец: 2015-11-02 00:00:00 -05:00

Пример #3 Добавление или вычитание времени может уменьшить или увеличить дату

Например, 31 января + 1 месяц вернёт 2 марта (високосный год) или 3 марта (обычный год).

<?php echo "Обычный год:\n"; // В феврале 28 дней $dt = new DateTimeImmutable("2015-01-31 00:00:00", new DateTimeZone("America/New\_York")); echo "Начало: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; $dt = $dt->modify("+1 month"); echo "Конец: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; echo "Високосный год:\n"; // В феврале 29 дней $dt = new DateTimeImmutable("2016-01-31 00:00:00", new DateTimeZone("America/New\_York"));

echo "Начало: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; $dt = $dt->modify("+1 month"); echo "Конец: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; ?>

Результат выполнения данного примера:

Обычный год:

Начало: 2015-01-31 00:00:00 -05:00

Конец: 2015-03-03 00:00:00 -05:00

Високосный год:

Начало: 2016-01-31 00:00:00 -05:00

Конец: 2016-03-02 00:00:00 -05:00

Для получения последнего дня следующего месяца (то есть чтобы предотвратить переполнение) существует директива last day of.

<?php echo "Обычный год:\n"; // Февраль содержит 28 дней $dt = new DateTimeImmutable("2015-01-31 00:00:00", new DateTimeZone("America/New\_York")); echo "Начало: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; $dt = $dt->modify("last day of next month"); echo "Конец: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; echo "Високосный год:\n"; // Февраль содержит 29 дней $dt = new DateTimeImmutable("2016-01-31 00:00:00", new DateTimeZone("America/New\_York")); echo "Начало: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; $dt = $dt->modify("last day of next month"); echo "Конец: ", $dt->format("Y-m-d H:i:s P"), PHP\_EOL; ?>

Результат выполнения данного примера:

Обычный год:

Начало: 2015-01-31 00:00:00 -05:00

Конец: 2015-02-28 00:00:00 -05:00

Високосный год:

Начало: 2016-01-31 00:00:00 -05:00

Конец: 2016-02-29 00:00:00 -05:00

7. Чем отличается фреймворк от библиотеки? Приведите пример

В отличие от библиотеки, которая объединяет в себе набор близкой функциональности, — «фреймворк» может содержать в себе большое число разных по тематике библиотек.

Другим ключевым отличием «фреймворка» от библиотеки может быть инверсия управления: пользовательский код вызывает функции библиотеки (или классы) и получает управление после вызова.

8. Опишите возможные форматы даты и времени и примеры их использования

Данные даты (date)

Дата при использовании типа date определяется в формате "ГГГГ-ММ-ДД", где:

ГГГГ – год; ММ — месяц; ДД – день

Данные времени (time)

Время определяется в формате "чч:мм:сс", где:

чч – часы; мм – минуты; сс – секунды

Тип данных dateTime

Тип данных dateTime используется для определения даты и времени.

Значения типа dateTime имеют формат "ГГГГ-ММ-ДДTчч:мм:сс", где:

ГГГГ – год; ММ – месяц; ДД – день; T — указывает на начало данных времени; чч – час; мм – минуты; сс – секунды

Данные о продолжительности

Типы данных о продолжительности используются для определения интервалов времени.

Интервал времени определяется в формате "PnYnMnDTnHnMnS", где:

P указывает период (обязателен); nY указывает число лет; nM указывает число месяцев; nD указывает число дней; T указывает на начало раздела с временем (обязателен, если будут определяться часы, минуты или секунды); nH указывает количество часов; nM указывает количество минут; nS указывает количество секунд.

9. Опишите работу с датой и временем в подходе ООП.

PHP предоставляет специализированный класс DateTime для работы с датой и временем.

Вот несколько причин, почему предпочтительнее использовать класс DateTime вместо strtotime и date:

Класс DateTime может работать с большим числом форматов даты и времени по сравнению с strtotime.

Работать с объектами легче, чем с функциями. Даты, являющиеся объектами класса DateTime можно сравнивать напрямую, как обычные числа. Тогда как для сравнения двух дат с помощью функции strtotimе нам необходимо сначала преобразовать их во временные метки и только затем сравнить.

Объектно-ориентированный интерфейс DateTime скрывает много внутренней логики работы с датой и обеспечивает понятный и однозначный интерфейс.

10. Опишите использование класса DateTime

Класс DateTime может работать с большим числом форматов даты и времени. Даты, являющиеся объектами класса DateTime можно сравнивать напрямую, как обычные числа. Объектно-ориентированный интерфейс DateTime скрывает много внутренней логики работы с датой и обеспечивает понятный и однозначный интерфейс.

DateTime::add — Добавляет заданное количество дней, месяцев, лет, часов, минут и секунд к объекту DateTime

DateTime::\_\_construct — Конструктор класса DateTime

DateTime::createFromFormat — Разбирает строку с датой согласно указанному формату

DateTime::createFromImmutable — Возвращает объект DateTime инкапсулирующий заданный объект DateTimeImmutable

DateTime::createFromInterface — Возвращает новый объект DateTime, созданный из переданного объекта, реализующего интерфейс DateTimeInterface

DateTime::getLastErrors — Возвращает предупреждения и ошибки

DateTime::modify — Изменение временной метки

DateTime::\_\_set\_state — Обработчик \_\_set\_state

DateTime::setDate — Устанавливает дату

DateTime::setISODate — Устанавливает дату в формате ISO

DateTime::setTime — Устанавливает время

DateTime::setTimestamp — Устанавливает дату и время на основе метки времени Unix

DateTime::setTimezone — Устанавливает часовой пояс для объекта класса DateTime

DateTime::sub — Вычитает заданное количество дней, месяцев, лет, часов, минут и секунд из времени объекта DateTime

Пример: $now = new DateTime();

11. Опишите использование класса DateTimeImmutable

Представление даты и времени. Данный класс ведет себя аналогично классу DateTime, за исключением того, что он никогда не изменяет себя и всегда возвращает новый объект.

DateTime::diff — Возвращает разницу между двумя объектами DateTime

DateTime::format — Возвращает дату, отформатированную согласно переданному формату

DateTime::getOffset — Возвращает смещение часового пояса

DateTime::getTimestamp — Возвращает временную метку Unix

DateTime::getTimezone — Возвращает часовой пояс относительно текущего значения DateTime

DateTime::\_\_wakeup — Обработчик \_\_wakeup

Пример: $date = new DateTimeImmutable();

12. Опишите использование класса DateTimeZone

Представление часового пояса.

DateTimeZone :: \_\_ construct - Создает новый объект DateTimeZone

DateTimeZone :: getLocation - Возвращает информацию о местоположении для часового пояса

DateTimeZone :: getName - Возвращает название часового пояса

DateTimeZone :: getOffset - Возвращает смещение часового пояса от GMT

DateTimeZone :: getTransitions - Возвращает все переходы для часового пояса

DateTimeZone :: listAbbreviations - Возвращает ассоциативный массив, содержащий dst, смещение и имя часового пояса

DateTimeZone :: listIdentifiers - Возвращает числовой индексный массив со всеми идентификаторами часовых поясов.

const integer DateTimeZone::AFRICA = 1 ;

const integer DateTimeZone::AMERICA = 2 ;

const integer DateTimeZone::ANTARCTICA = 4 ;

const integer DateTimeZone::ARCTIC = 8 ;

const integer DateTimeZone::ASIA = 16 ;

const integer DateTimeZone::ATLANTIC = 32 ;

const integer DateTimeZone::AUSTRALIA = 64 ;

const integer DateTimeZone::EUROPE = 128 ;

const integer DateTimeZone::INDIAN = 256 ;

const integer DateTimeZone::PACIFIC = 512 ;

const integer DateTimeZone::UTC = 1024 ;

const integer DateTimeZone::ALL = 2047 ;

const integer DateTimeZone::ALL\_WITH\_BC = 4095 ;

const integer DateTimeZone::PER\_COUNTRY = 4096 ;

Пример: date\_default\_timezone\_set(«Europe/Oslo»);

13.Какие библиотеки используются для работы с изображениями в PHP.

Imagine, Php Graphic Works, Zebra Image, PhpThumb,GD.

14.Опишите основные возможности библиотеки GD.

Библиотека GD, позволяет создавать новые изображения, редактировать уже существующие, копировать одни изображения на другие, изменять размеры, а также наносить текст на изображения.

15.Как использовать Composer для подключения библиотек к проекту?

Composer упрощает не только установку библиотек, но и их использование. Он берёт на себя подключение всех необходимых файлов классов библиотеки. За это отвечает специальный сценарий autoload.php.

Сценарий autoload.php — единственный файл, который необходимо подключить для использования любых библиотек

16.Что такое PEAR? В чём разница работы PEAR и Composer?

PEAR — это библиотека классов PHP с открытым исходным кодом, распространяемых через одноименный пакетный менеджер. В стандартную поставку PHP входит система управления классами PEAR, которая позволяет легко скачивать и обновлять их.

Использование PEAR более обременительно для сопровождающих пакетов. Поэтому большая часть кода на PEAR устарела. Разработчику необходимо получить пакет "PEAR-reviewed", прежде чем он может быть опубликован на PEAR, поэтому количество доступных пакетов мало по сравнению с количеством пакетов, доступных в Composer. Кроме того, в PEAR нет возможности установить пакет для одного проекта. Все пакеты устанавливаются глобально. В Composer вы можете устанавливать пакеты для каждого проекта или глобально. Ну и еще в PEAR отсутствует управление зависимостями, что, откровенно говоря, должно быть единственной вещью, которую менеджер пакетов делает хорошо.

17.Как использовать PEAR для установки библиотек?

Сам пакетный менеджер pear не входит в состав дистрибутива PHP, поэтому необходимо, чтобы он был предварительно установлен у хостера.

Основная проблема — это чтобы php-скрипт «видел» откуда ему брать тот или иной компонент (PHP как всегда идёт «своим» путём). Для этого должна быть определена настройка include\_path, например, для пользователя vasya, так:

include\_path=".:/home/vasya/pear"

Перед установкой модулей PEAR следует сообщить утилите pear, что мы хотим ставить компоненты в свой домашний каталог командой:

pear create-config $HOME .pearrc

Будет создан конфигурационный файл, используемый pear в дальнейшем. Конечно же вместо $HOME можно выбрать любое другое место, не забыв отразить это в значении include\_path.

Теперь, если пакет существует в списке пакетов PEAR, можно просто устанавливать требуемые пакеты. Например:

pear install -o PEAR

установит базовый компонент системы PEAR с зависимостями.

Если пакет выложен на другом канале, вам нужно сначало сделать discover этого канала и затем указать его во время установки.

18.Как использовать Composer для обработки зависимостей PEAR?

Если вы уже используете Composer и желаете установить какой-то код из PEAR, вы можете использовать Composer для обработки зависимостей PEAR. Этот пример установит код из pear2.php.net:

{

"repositories": [

{

"type": "pear",

"url": "http://pear2.php.net"

}

],

"require": {

"pear-pear2/PEAR2\_Text\_Markdown": "\*",

"pear-pear2/PEAR2\_HTTP\_Request": "\*"

}

}

Первый раздел "repositories" даст понять Composer, что он должен сделать "initialise" (или "discover" в терминологии PEAR) репозиторий pear. Затем секция require укажет именам пакетов префикс, как ниже:

pear-channel/Package

Префикс "pear" жестко ограничен, чтобы избежать любых конфликтов, так как каналы Pear могут быть схожи с другими поставщиками пакетов,

например, вместо короткого имени (или полного URL) может быть использовано для объявления в каком канале находится пакет.

Когда код будет установлен он будет доступен в вашей папке vendor и автоматически доступен через автозагрузчик (файл Autoload) Composer.

vendor/pear-pear2.php.net/PEAR2\_HTTP\_Request/pear2/HTTP/Request.php

Чтобы использовать этот пакет PEAR просто объявите, как ниже:

$request = new pear2\HTTP\Request();

19. Что такое PECL?

PECL - это репозиторий нативных расширений, написанных на C. Обычно из используют, когда что-то нельзя реализовать на голом PHP, например перегрузку функций или операторов. 20. Как декодировать строку JSON и вернуть JSON-представление данных?

Метод JSONDecoder.raw\_decode() декодирует JSON-документ из строки s в формате JSON и возвращает двойной кортеж представление данной строки в Python и индекс в строки s, где документ закончился. Метод JSONDecoder.raw\_decode() может быть использован для декодирования документа JSON из строки, которая в конце содержит посторонние данные.

20. Как называется репозиторий, содержащий Composer-совместимые

библиотеки?

21. С помощью какой библиотеки можно получить детальный отчет о

работе приложения?

PHP Benchmark

\PHPBenchmark\Monitor::instance()->snapshot('Plugins loaded');

22. С помощью какой библиотеки можно упростить себе работу с

регулярными выражениями в PHP?  
  
PCRE  
23. С помощью какой библиотеки возможны быстрые и эффективные

запросы на PHP, данная библиотека является аналогом jQuery.  
Simple HTML DOM — PHP-библиотека, позволяющая парсить HTML-код с помощью удобных jQuery-подобных селекторов. Она лишена главного недостатка XPath — библиотека умеет работать даже с невалидным HTML-кодом, что значительно упрощает работу.  
24. С помощью какой библиотеки возможно простое и эффективное

генерирование документов в формате PDF?  
С помощью какой библиотеки возможно простое и эффективное генерирование документов в формате PDF

pdftk, DOMPDF

25. С помощью какой библиотеки возможен эффективный парсинг

HTML/XML?

C помощью какой библиотеки возможен эффективный парсинг HTML/XML

phpQuery, PHP Simple HTML DOM Parser, Nokogiri.

26. С помощью какой библиотеки возможно создание диаграмм на

движке GOOGLE?

C помощью какой библиотеки возможно сканирование конфигурационного файла PHP на предмет безопасности

SaltStack

27. С помощью какой библиотеки возможно сканирование

конфигурационного файла PHP на предмет безопасности?

RIPS

28. С помощью какой библиотеки возможен анализ HTML и удаление

вредоносного кода для защиты от XSS атак. HTMLPurifier

29. С помощью какой библиотеки возможно построение графиков,

диаграмм и другого структурированного контента на PHP?

PChart — это PHP-библиотека для создания графиков, гистограмм и диаграмм.

30. С помощью какой библиотеки облегчается процесс загрузки и

валидации файлов на PHP? Upload

## Список использованной литературы:

1. Конспект лекций по дисциплине «Разработка сервисных частей интернет ресурсов», РТУ МИРЭА, лектор –преподаватель Куликов А.А.

2. Docker – URL: https://www.docker.com/ (Дата обращения – 16.11.2021)

3. Faker: https://fakerphp.github.io/ (Дата обращения – 16.11.2021)

4. Полное практическое руководство Docker – URL: https://habr.com/ru/post/310460/ (Дата обращения – 16.11.2021)

5. Тестирование fixtures – URL: https://www.yiiframework.com/doc/guide/2.0/ru/test-fixtures (Дата обращения – 16.11.2021)

## 6. PHP Line Charts описание – URL:

https://habr.com/ru/post/30202/ (Дата обращения – 16.11.2021)