|  |  |  |
| --- | --- | --- |
|  |  |  |
| 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 №6**

по дисциплине «Разработка серверных частей интернет-ресурсов»

**Студент группы** ИКБО-01-19 Кузин Д.О.

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

(подпись студента)

**Руководитель практической работы** преподаватель Волков М.Ю.

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

(подпись руководителя)

Работа представлена «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Москва 2021

**Оглавление**

[Цель работы 3](#_Toc88877274)

[Задание 3](#_Toc88877275)

[Ход работы 3](#_Toc88877276)

[Вывод 12](#_Toc88877277)

[Ответы на вопросы к практической работе 13](#_Toc88877278)

[Список использованной литературы 32](#_Toc88877279)

# Цель работы

Используя серверную конфигурацию из предыдущих работ, научиться работать различными расширениями и сторонними модулями PHP.

# Задание

1. Сгенерировать набор фикстур с помощью выбранного стороннего модуля. Каждая фикстура должна содержать не менее 5 полей. Всего фикстур должно быть не менее 50.
2. На основе этих фикстур построить минимум 3 графика различного типа с помощью сторонних модулей PHP.
3. График превратить в картинку и добавить на него ваш личный полупрозрачный водяной знак с помощью библиотеки GD(строго рекомендуется) или другой библиотеки для работы с изображениями в PHP.
4. Отобразить полученные графики на странице статистики.

# Ход работы

1. На основе серверной конфигурации, разработанной в первой работе, были добавлены настройки и код для работы с созданием фикстур и графиков (Листинг 1 – 11).

Листинг 1 – изменения в сервисе web в docker-compose.yml

 web:

build:

context: .

volumes:

- ./src/dynamic:/var/www/html/php/

- ./src/static/files:/var/www/files/

restart: always

Листинг 2 – Dockerfile

FROM php:7.4-apache

RUN php -r "readfile('http://getcomposer.org/installer');" | php -- --install-dir=/usr/local/bin/ --filename=composer

RUN alias composer='php /usr/local/bin/composer'

RUN apt-get update && \

pecl uninstall redis && \

pecl install -f redis && \

docker-php-ext-enable redis && \

docker-php-ext-install mysqli

RUN apt-get install -y zip

RUN composer require --prefer-dist nelmio/alice && \

composer update

RUN apt-get update && \

apt-get install -y libfreetype6-dev libjpeg62-turbo-dev libpng-dev && \

docker-php-ext-configure gd --with-freetype=/usr/include/ --with-jpeg=/usr/include/ && \

docker-php-ext-install gd

RUN composer require szymach/c-pchart

Листинг 3 – index.php в diagrams

<?php

include '../login.php';

?>

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Графики</title>

</head>

<body>

<h1>Графики</h1>

<div>

<a href="/php/trade/index.php">Панель администратора</a>

</div>

<div>

<img src="/php/trade/diagrams/diagramDarkAndMilkProfit.php"/>

</div>

<div>

<img src="/php/trade/diagrams/diagramMilkRating.php"/>

<img src="/php/trade/diagrams/diagramDarkRating.php"/>

</div>

<div>

<img src="/php/trade/diagrams/diagramCountInOrder.php"/>

</div>

</body>

</html>

Листинг 4 – diagramCountInOrder.php

<?php

include '../login.php';

include './getFixtures.php';

require '../../../vendor/autoload.php';

use CpChart\Chart\Pie;

use CpChart\Data;

use CpChart\Image;

include './drawImage.php';

$points = [];

$labels = [];

$uniqueCount = [];

foreach ($fixtures->getObjects() as $fixture) {

$count = floor($fixture->count\_in\_order/100);

$index = array\_search($count, $uniqueCount);

if ($index !== false) {

$points[$index]++;

} else {

array\_push($points, 1);

array\_push($labels, $count\*100 . " - " . ($count\*100 + 100));

array\_push($uniqueCount, $count);

}

}

$serie\_abcissa = "serie\_abcissa";

$serie\_count = "serie\_count";

// Create and populate data

$data = new Data();

$data->addPoints($points, $serie\_count);

// Define the absissa serie

$data->addPoints($labels, $serie\_abcissa);

$data->setAbscissa($serie\_abcissa);

// Create the image

$image = new Image(400, 230, $data);

// Add a border to the picture

$image->drawRectangle(0, 0, 399, 229, ["R" => 0, "G" => 0, "B" => 0]);

// Write the picture title

$image->setFontProperties(["FontName" => "Silkscreen.ttf", "FontSize" => 14]);

$image->drawText(200, 13, "Count in order", ["R" => 0, "G" => 0, "B" => 0, "Align" => TEXT\_ALIGN\_TOPMIDDLE]);

// Set the default font properties

$image->setFontProperties(["FontName" => "Forgotte.ttf", "FontSize" => 10, "R" => 80, "G" => 80, "B" => 80]);

// Create and draw the chart

$pieChart = new Pie($image, $data);

$pieChart->draw2DPie(200, 125, ["DrawLabels" => true, "Border" => true]);

drawImage($image);

Листинг 5 – diagramDarkAndMilkProfit.php

<?php

include '../login.php';

include './getFixtures.php';

require '../../../vendor/autoload.php';

use CpChart\Data;

use CpChart\Image;

include './drawImage.php';

$data = new Data();

$serie\_dark = "dark";

$serie\_milk = "milk";

$serie\_days = "days";

$day = 0;

foreach ($fixtures->getObjects() as $fixture) {

$data->addPoints($fixture->dark\_chocolate\_profit, $serie\_dark);

$data->addPoints($fixture->milk\_chocolate\_profit, $serie\_milk);

$data->addPoints($day, $serie\_days);

$day++;

}

$data->setAbscissa($serie\_days);

$data->setAbscissaName("Days");

//SERIE\_SHAPE\_FILLEDTRIANGLE - triangle

//SERIE\_SHAPE\_FILLEDSQUARE - square

$data->setSerieShape($serie\_milk, SERIE\_SHAPE\_FILLEDTRIANGLE);

$data->setSerieShape($serie\_dark, SERIE\_SHAPE\_FILLEDSQUARE);

$data->setSerieOnAxis($serie\_milk, 0);

$data->setSerieOnAxis($serie\_dark, 1);

$data->setAxisName(0, "Milk Profit");

$data->setAxisName(1, "Dark Profit");

$data->setAxisPosition(1, AXIS\_POSITION\_RIGHT);

/\* Create the Image object \*/

$image = new Image(900, 300, $data);

/\* Turn off Antialiasing \*/

$image->Antialias = false;

/\* Add a border to the picture \*/

$image->drawRectangle(0, 0, 899, 299, ["R" => 0, "G" => 0, "B" => 0]);

/\* Write the chart title \*/

$image->setFontProperties(["FontName" => "Forgotte.ttf", "FontSize" => 11]);

$image->drawText(200, 35, "Milk and dark chocolate profit", ["FontSize" => 20, "Align" => TEXT\_ALIGN\_BOTTOMMIDDLE]);

/\* Set the default font \*/

$image->setFontProperties(["FontName" => "pf\_arma\_five.ttf", "FontSize" => 6]);

/\* Define the chart area \*/

$image->setGraphArea(60, 40, 850, 250);

/\* Draw the scale \*/

$scaleSettings = [

"XMargin" => 10,

"YMargin" => 10,

"Floating" => true,

"GridR" => 200,

"GridG" => 200,

"GridB" => 200,

"DrawSubTicks" => true,

"CycleBackground" => true

];

$image->drawScale($scaleSettings);

/\* Turn on Antialiasing \*/

$image->Antialias = true;

$image->setShadow(true, ["X" => 1, "Y" => 1, "R" => 0, "G" => 0, "B" => 0, "Alpha" => 10]);

/\* Draw the line chart \*/

$image->drawPlotChart();

$image->drawLineChart();

/\* Write the chart legend \*/

$image->drawLegend(680, 20, ["Style" => LEGEND\_NOBORDER, "Mode" => LEGEND\_HORIZONTAL]);

drawImage($image);

Листинг 6 – diagramDarkRating.php

<?php

include '../login.php';

include './getFixtures.php';

include './drawRatingDiagram.php';

$points = [0, 0, 0, 0, 0]; // 0 - <=$norm; 1 - >$norm

foreach ($fixtures->getObjects() as $fixture) {

$points[$fixture->dark\_rating\_from\_users - 1]++;

}

drawRatingDiagram($points, "Dark", "Dark rating", 0, max($points) + 1);

Листинг 7 – diagramMilkRating.php

<?php

include '../login.php';

include './getFixtures.php';

include './drawRatingDiagram.php';

$points = [0, 0, 0, 0, 0]; // 0 - <=$norm; 1 - >$norm

foreach ($fixtures->getObjects() as $fixture) {

$points[$fixture->milk\_rating\_from\_users - 1]++;

}

drawRatingDiagram($points, "Milk", "Milk rating", 0, max($points) + 1);

Листинг 8 – drawRatingDiagram.php

<?php

include '../login.php';

require '../../../vendor/autoload.php';

use CpChart\Data;

use CpChart\Image;

include './drawImage.php';

function drawRatingDiagram($points, $axisName, $title, $min, $max)

{

$data = new Data();

$serie\_rating = "serie\_rating";

$serie\_user\_count = "serie\_user\_count";

$data->addPoints($points, $serie\_user\_count);

$data->setAxisName(0, $axisName);

$data->addPoints(["1", "2", "3", "4", "5"], $serie\_rating);

$data->setAbscissa($serie\_rating);

/\* Create the Image object \*/

$image = new Image(500, 500, $data);

$image->setFontProperties(["FontName" => "pf\_arma\_five.ttf", "FontSize" => 6]);

/\* Draw the chart scale \*/

$image->setGraphArea(30, 80, 480, 480);

$image->drawText(250, 55, $title, array("FontSize" => 20, "Align" => TEXT\_ALIGN\_BOTTOMMIDDLE));

/\* Turn on shadow computing \*/

$image->setShadow(true, ["X" => 1, "Y" => 1, "R" => 0, "G" => 0, "B" => 0, "Alpha" => 10]);

/\* Draw the chart \*/

$image->drawScale([

"Mode" => SCALE\_MODE\_MANUAL,

"ManualScale" => [

0 => [

"Min" => $min,

"Max" => $max,

],

],

]);

$image->drawBarChart([

"DisplayValues" => true,

"Rounded" => true,

"Surrounding" => 30,

]);

drawImage($image);

}

Листинг 9 – getFixtures.php

<?php

include '../login.php';

require '../../../vendor/autoload.php';

use Nelmio\Alice\Loader\NativeLoader;

// данные для ящичка интернет-вещей

class Dump {

public $dark\_chocolate\_profit;

public $milk\_chocolate\_profit;

public $milk\_rating\_from\_users;

public $dark\_rating\_from\_users;

public $count\_in\_order;

}

$loader = new Nelmio\Alice\Loader\NativeLoader();

$fixtures = $loader->loadFile('./fixtures.yml');

?>

Листинг 10 – fixtures.yml

Dump:

dump{0..59}:

milk\_chocolate\_profit: '<randomFloat(2, 100, 1000)>'

dark\_chocolate\_profit: '<randomFloat(2, 100, 1000)>'

milk\_rating\_from\_users: '<numberBetween(1, 5)>'

dark\_rating\_from\_users: '<numberBetween(1, 5)>'

count\_in\_order: '<numberBetween(1, 1000)>'

Листинг 11 – drawImage.php

<?php

function drawImage($image) {

$filename1 = "example.createFunctionSerie.scatter.png";

$filename\_whater\_mark = "whater\_mark.png";

$image->render($filename1);

$stamp = imagecreatefrompng($filename\_whater\_mark);

$im = imagecreatefrompng($filename1);

$marge\_right = 10;

$marge\_bottom = 10;

$sx = imagesx($stamp);

$sy = imagesy($stamp);

imagecopy($im, $stamp, imagesx($im) - $sx - $marge\_right, imagesy($im) - $sy - $marge\_bottom, 0, 0, imagesx($stamp), imagesy($stamp));

header('Content-type: image/png');

imagepng($im);

imagedestroy($im);

unlink($filename1);

}

1. Программа была запущена и протестирована (Рисунок 1 - 3).

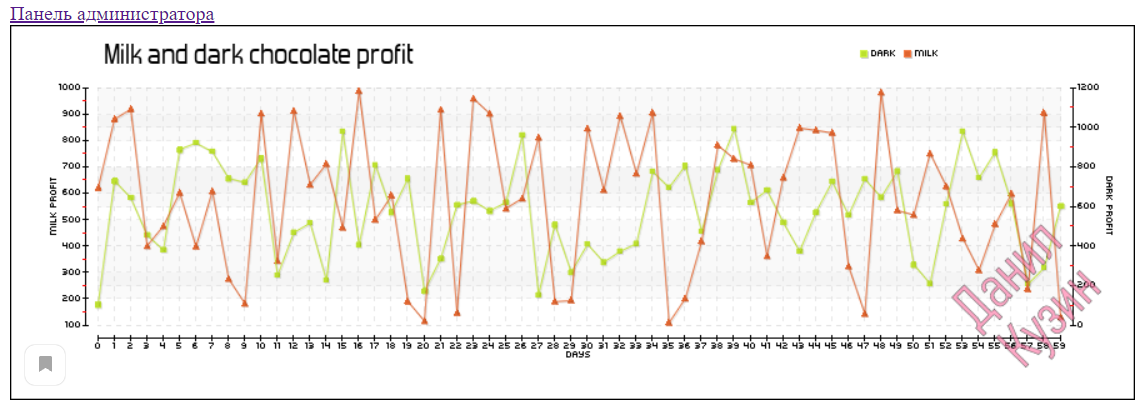


Рисунок 1 – линейная диаграмма

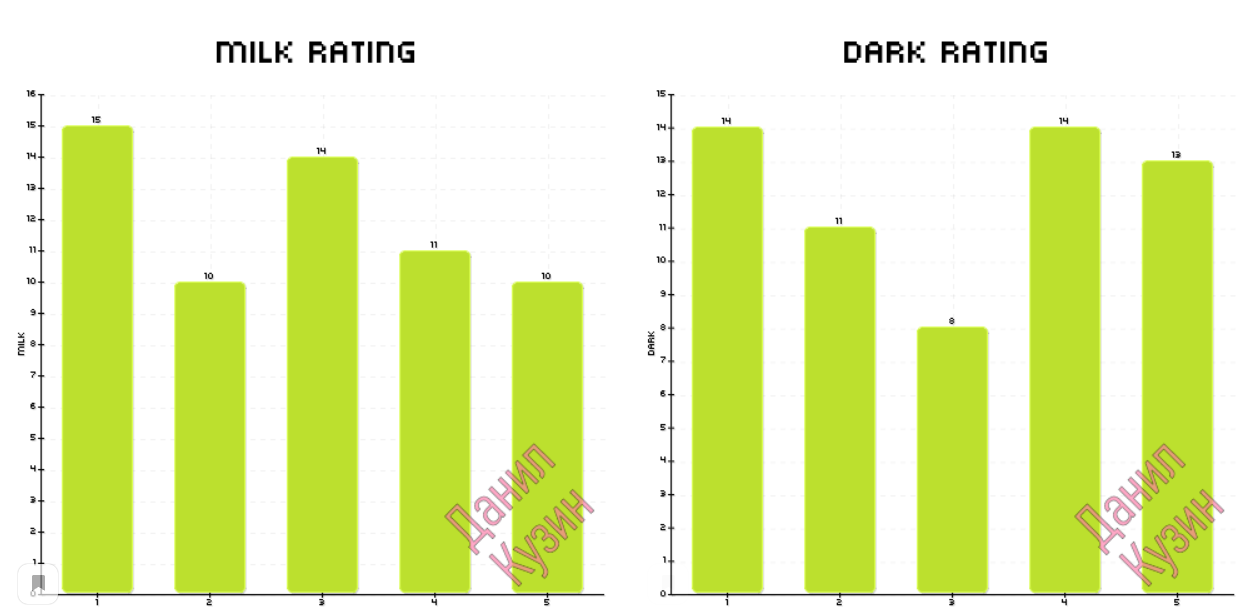


Рисунок 2 – столбчатая диаграмма

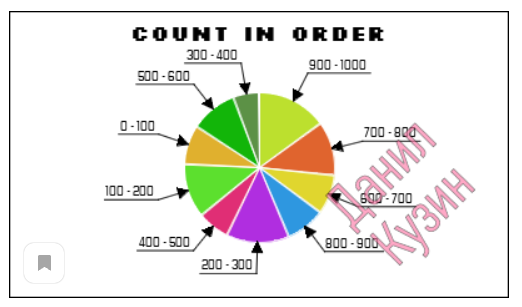


Рисунок 3 – круговая диаграмма

# Вывод

В результате работы была изучена работа с различными расширениями и сторонними модулями PHP.

# Ответы на вопросы к практической работе

1. Что такое Composer и для чего он используется?

Composer - это менеджер для подключения и управления сторонними библиотеками или пакетам в PHP. Его называют пакетный менеджер.

Зачем же в таком случае нужен composer?

1. Много рутинных операций при установке. Нужно производить настройки, подстраивать автозагрузку компонентов и.т.д.

2. Трудность в обновлении библиотек на новые версии. Библиотеки имеют такое свойство расширяться и обновляться. Если вы вручную будете обновлять каждый пакет в вашем проекте, это работа довольно трудная и большая. В composer для обновления пакетов достаточно выполнить одну команду и все пакеты успешно обновляться.

3. Одна библиотека может требовать для своей работы другую, другая третью и.т.д. В итоге, может получаться ряд зависимостей, которые вы должны подключать. Бывают такие библиотеки, которые для своей работы могут требовать десяток и даже сотни других библиотек. Если это все скачивать и подключать вручную на это может уйти месяцы работы. В случае с composer вы просто выполняете команду установки какой-либо библиотеки и он уже автоматически подключит все необходимые для этого другие библиотеки. Пожалуй, это самое важное преимущество почему стоит использовать composer в своей работе.

4. Трудность переноса проекта на рабочий сервер из-за большого объема библиотек. В вашем проекте код самого проекта может занимать всего десятки килобайт, а библиотеки, которые подключены к этому проекту могут занимать сотни и 10-ки сотен мегабайт. Для того, чтобы перенести изменения на рабочем сервере, вам каждый раз нужно будет передавать такой большой объем данных.

Решение этой проблемы с composer довольно простое. В программе composer есть файл настроек, который вы устанавливаете на вашем домашнем компьютере и файл настроек, который вы устанавливаете на удаленном компьютер

2. Опишите работу с календарем в PHP: опишите преобразование дат из одного календаря в другой.

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

jdtogregorian(int $julian\_day): string

Переводит число дней в юлианском летоисчислении в строку, содержащую григорианскую дату в формате "месяц/день/год".

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

cal\_from\_jd(int $julian\_day, int $calendar): array

cal\_from\_jd() преобразует дату юлианского календаря, заданную в julian\_day, в дату указанного календаря calendar. Поддерживаемые значения calendar: CAL\_GREGORIAN, CAL\_JULIAN, CAL\_JEWISH и CAL\_FRENCH.



Рисунок 4 - Пример из лекции о дате и времени

3. Опишите понятие серверное время и процесс форматирования времени для разных часовых поясов.

Чтобы получить время сервера, используйте функции time() или date() (есть и другие методы, но их должно быть достаточно). Однако time() может вызывать медленно, как и другие функции, которые получают текущее время (поскольку для получения времени требуются системные вызовы). Вы также должны знать, что он будет меняться от вызова к вызову, если выполнение сценария между вызовами занимает более 1 секунды. Это может вызвать проблемы с согласованностью. Обходной путь заключается в использовании $\_SERVER['REQUEST\_TIME'], если это возможно, и устранит ошибки постоянства и проблемы со скоростью, которые могут быть вызваны повторным вызовом time(), date() и т. Д., Которые активно получают текущее системное время.

* checkdate — *Проверяет правильность даты по грегорианскому календарю*
* date(gmdate) — *Форматирует системную дату/время (по Гринвичу)*
* getdate — *Возвращает информацию о дате/времени*
* gettimeofday — *Возвращает текущее время*
* localtime — Возвращает системное время
* mktime(gmmktime) — Возвращает метку времени для заданной даты (по Гринвичу)
* strftime(gmstrftime) — *Форматирует дату/время с учетом текущей локали (по Гринвичу)*
* strtotime — *Преобразует текстовое представление даты на английском языке в метку времени Unix*

4. Опишите способ получения времени заката и рассвета с использованием языка программирования PHP.

date\_sun\_info ( int $time , float $latitude , float $longitude )

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

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.

«**Фреймворк**» **отличается** от понятия **библиотеки** тем, что **библиотека** может быть использована в программном продукте просто как набор подпрограмм близкой функциональности, не влияя на архитектуру программного продукта и не накладывая на неё никаких ограничений.

**Yii** - достаточно "возрастной" фреймворк, который продолжает обновляться в наши дни. Отличается удобным функционалом - кэширование, высокая производительность, полная обработка ошибок, возможность переноса (миграции) существующих баз данных, использование jQuery и другое. Фреймворк Yii отличается своей простотой, можно быстро освоить его основы, нет никаких сложностей в работе и использовании основного функционала. Начните [изучать Yii2 фрейморк на нашем курсе](https://fructcode.com/ru/courses/yii2/?utm_source=blog_fc&utm_medium=blog_fc_frameworks&utm_campaign=blog_fc_frameworks). Данный PHP-фреймворк часто советуют людям, которые делают первые шаги в понимании PHP-программирования.

Официальная страница [yiiframework.com](http://www.yiiframework.com/)

**CodeIgniter** - еще один "возрастной" фреймворк, появившийся в начале 2006 года. Именно тогда состоялся его публичный релиз. Среди основных преимуществ этого фреймворка:

- Хорошая документация;

- Небольшой вес и быстрая установка;

- Простота использования.

Многие используют [CodeIgniter в качестве базы для обучения](https://fructcode.com/ru/courses/codeigniter-mvc/) - его простота действительно считается наиболее значимым преимуществом. Постоянно появляются новые версии, каждая из которых отличается большим количеством нововведений, исправленных багов.

Официальная страница [codeigniter.com](https://www.codeigniter.com/)

**Symfony** - невероятно стабильный, мощный фреймворк, который специалисты рекомендуют применять для созданиях крупных проектов. Значительный функционал, гибкость в настройках - популярность этого фреймворка обусловлена его преимуществами. Присутствует огромное количество полезных, многоразовых компонентов, которые можно использовать для создания большого сайта. Сюда можно отнести шаблоны, настройки форм, безопасность.

Официальная страница [symfony.com](https://symfony.com/)

**Laravel** - частый лидер разнообразных опросов и рейтингов, посвященных php-фреймворкам. Проект является действительно многообещающим, получил признание достаточно опытных специалистов. Фреймворк просто освоить, является идеальным вариантом для небольших, а также средних по сложности проектов. Подойдет для быстрого, удобного написания требуемого кода.

Официальная страница [laravel.com](https://laravel.com/)

**Phalcon PHP** - отличается открытым кодом (языки программирования C, C++, PHP), поддержкой практически всех современных ОС. Производительность этого фреймворка находится на высоком уровне - это подтверждено множеством специализированных тестирований, и, как следствие, его популярностью. Есть возможность использования на собственном сервере.

Официальная страница [phalconphp.com](https://phalconphp.com/ru/)

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

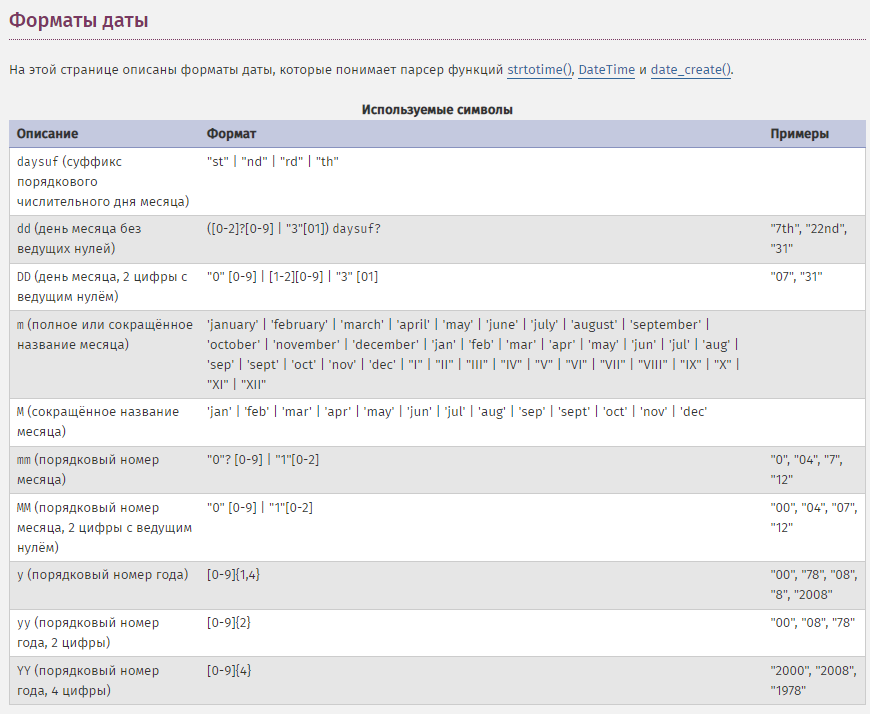


Рисунок - Форматы даты php



Рисунок - Форматы даты и времени php

date( ... )

* d - номер дня в месяце. Если меньше 10, то с нулём: "09", "05"
* m - номер месяца. Если меньше 10, то с нулём: "09", "05"
* Y - год, 4 цифры.
* y - год, две цифры.
* n - номер месяца. Без первого нуля, если меньше 10
* j - номер дня в месяце. Без первого нуля, если меньше 10
* H - часы в 24-часовом формате. Если меньше 10, то с нулём: "09", "05"
* s - секунды. Если меньше 10, то с нулём: "09", "05"
* i - минуты. Если меньше 10, то с нулём: "09", "05"
* z - номер дня от начала года.
* w - день недели (0 - воскресенье, 1 - понедельник и т.д.).
* h - часы в 12-часовом формате
* L - 1, если високосный год, 0, если не високосный.
* W - порядковый номер недели года.
* U - количество секунд, прошедших с 1 января 1970 года (то есть timestamp).

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

[PHP-расширение Date / Time](https://translate.googleusercontent.com/translate_c?depth=1&rurl=translate.google.ru&sl=en&sp=nmt4&tl=ru&u=http://www.php.net/manual/en/book.datetime.php&xid=17259,1500004,15700021,15700186,15700190,15700259,15700271,15700302&usg=ALkJrhiSYcq-Id_gD1IVUVeHe0wsnyWusw) представляет собой набор классов, которые позволяют вам работать практически со всеми задачами, связанными с датой и временем. Он был доступен с момента выпуска PHP 5.2, и расширение представило несколько новых классов, все из которых сопоставлены с реальными сценариями:

* Дата или время представлены объектом DateTime .
* Часовой пояс мира представлен объектом DateTimeZone .
* Объекты DateInterval представляют интервал. Например, когда мы говорим «два дня», «два дня» — это интервал. Объект DateInterval не зависит от конкретной даты или времени.
* Объекты DatePeriod представляют период между двумя датами.

10.Опишите использование класса DateTime.

Листинг - Примеры методов DateTime

/\* Методы \*/

public [\_\_construct](https://www.php.net/manual/ru/datetime.construct.php)(string $datetime = "now", ?[DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php) $timezone = null)

public [add](https://www.php.net/manual/ru/datetime.add.php)([DateInterval](https://www.php.net/manual/ru/class.dateinterval.php) $interval): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public static [createFromFormat](https://www.php.net/manual/ru/datetime.createfromformat.php)(string $format, string $datetime, ?[DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php) $timezone = null): [DateTime](https://www.php.net/manual/ru/class.datetime.php)|false

public static [createFromImmutable](https://www.php.net/manual/ru/datetime.createfromimmutable.php)([DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php) $object): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public static [createFromInterface](https://www.php.net/manual/ru/datetime.createfrominterface.php)([DateTimeInterface](https://www.php.net/manual/ru/class.datetimeinterface.php) $object): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public static [getLastErrors](https://www.php.net/manual/ru/datetime.getlasterrors.php)(): array|false

public [modify](https://www.php.net/manual/ru/datetime.modify.php)(string $modifier): [DateTime](https://www.php.net/manual/ru/class.datetime.php)|false

public static [\_\_set\_state](https://www.php.net/manual/ru/datetime.set-state.php)(array $array): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public [setDate](https://www.php.net/manual/ru/datetime.setdate.php)(int $year, int $month, int $day): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public [setISODate](https://www.php.net/manual/ru/datetime.setisodate.php)(int $year, int $week, int $dayOfWeek = 1): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public [setTime](https://www.php.net/manual/ru/datetime.settime.php)(

int $hour,

int $minute,

int $second = 0,

int $microsecond = 0

): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public [setTimestamp](https://www.php.net/manual/ru/datetime.settimestamp.php)(int $timestamp): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public [setTimezone](https://www.php.net/manual/ru/datetime.settimezone.php)([DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php) $timezone): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public [sub](https://www.php.net/manual/ru/datetime.sub.php)([DateInterval](https://www.php.net/manual/ru/class.dateinterval.php) $interval): [DateTime](https://www.php.net/manual/ru/class.datetime.php)

public [diff](https://www.php.net/manual/ru/datetime.diff.php)([DateTimeInterface](https://www.php.net/manual/ru/class.datetimeinterface.php) $targetObject, bool $absolute = false): [DateInterval](https://www.php.net/manual/ru/class.dateinterval.php)

public [format](https://www.php.net/manual/ru/datetime.format.php)(string $format): string

public [getOffset](https://www.php.net/manual/ru/datetime.getoffset.php)(): int

public [getTimestamp](https://www.php.net/manual/ru/datetime.gettimestamp.php)(): int

public [getTimezone](https://www.php.net/manual/ru/datetime.gettimezone.php)(): [DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php)|false

public [\_\_wakeup](https://www.php.net/manual/ru/datetime.wakeup.php)(): void

11.Опишите использование класса DateTimeImmutable.

Листинг - Примеры констант и методов DateTimeImmutable

class DateTimeImmutable implements [DateTimeInterface](https://www.php.net/manual/ru/class.datetimeinterface.php) {

/\* Наследуемые константы \*/

const string [DateTimeInterface::ATOM](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.atom) = "Y-m-d\TH:i:sP";

const string [DateTimeInterface::COOKIE](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.cookie) = "l, d-M-Y H:i:s T";

const string [DateTimeInterface::ISO8601](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.iso8601) = "Y-m-d\TH:i:sO";

const string [DateTimeInterface::RFC822](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc822) = "D, d M y H:i:s O";

const string [DateTimeInterface::RFC850](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc850) = "l, d-M-y H:i:s T";

const string [DateTimeInterface::RFC1036](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc1036) = "D, d M y H:i:s O";

const string [DateTimeInterface::RFC1123](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc1123) = "D, d M Y H:i:s O";

const string [DateTimeInterface::RFC7231](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc7231) = "D, d M Y H:i:s \G\M\T";

const string [DateTimeInterface::RFC2822](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc2822) = "D, d M Y H:i:s O";

const string [DateTimeInterface::RFC3339](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc3339) = "Y-m-d\TH:i:sP";

const string [DateTimeInterface::RFC3339\_EXTENDED](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rfc3339_extended) = "Y-m-d\TH:i:s.vP";

const string [DateTimeInterface::RSS](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.rss) = "D, d M Y H:i:s O";

const string [DateTimeInterface::W3C](https://www.php.net/manual/ru/class.datetimeinterface.php#datetime.constants.w3c) = "Y-m-d\TH:i:sP";

/\* Методы \*/

public [\_\_construct](https://www.php.net/manual/ru/datetimeimmutable.construct.php)(string $datetime = "now", ?[DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php) $timezone = null)

public [add](https://www.php.net/manual/ru/datetimeimmutable.add.php)([DateInterval](https://www.php.net/manual/ru/class.dateinterval.php) $interval): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public static [createFromFormat](https://www.php.net/manual/ru/datetimeimmutable.createfromformat.php)(string $format, string $datetime, ?[DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php) $timezone = null): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)|false

public static [createFromInterface](https://www.php.net/manual/ru/datetimeimmutable.createfrominterface.php)([DateTimeInterface](https://www.php.net/manual/ru/class.datetimeinterface.php) $object): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public static [createFromMutable](https://www.php.net/manual/ru/datetimeimmutable.createfrommutable.php)([DateTime](https://www.php.net/manual/ru/class.datetime.php) $object): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public static [getLastErrors](https://www.php.net/manual/ru/datetimeimmutable.getlasterrors.php)(): array|false

public [modify](https://www.php.net/manual/ru/datetimeimmutable.modify.php)(string $modifier): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)|false

public static [\_\_set\_state](https://www.php.net/manual/ru/datetimeimmutable.set-state.php)(array $array): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public [setDate](https://www.php.net/manual/ru/datetimeimmutable.setdate.php)(int $year, int $month, int $day): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public [setISODate](https://www.php.net/manual/ru/datetimeimmutable.setisodate.php)(int $year, int $week, int $dayOfWeek = 1): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public [setTime](https://www.php.net/manual/ru/datetimeimmutable.settime.php)(

int $hour,

int $minute,

int $second = 0,

int $microsecond = 0

): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public [setTimestamp](https://www.php.net/manual/ru/datetimeimmutable.settimestamp.php)(int $timestamp): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public [setTimezone](https://www.php.net/manual/ru/datetimeimmutable.settimezone.php)([DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php) $timezone): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public [sub](https://www.php.net/manual/ru/datetimeimmutable.sub.php)([DateInterval](https://www.php.net/manual/ru/class.dateinterval.php) $interval): [DateTimeImmutable](https://www.php.net/manual/ru/class.datetimeimmutable.php)

public [diff](https://www.php.net/manual/ru/datetime.diff.php)([DateTimeInterface](https://www.php.net/manual/ru/class.datetimeinterface.php) $targetObject, bool $absolute = false): [DateInterval](https://www.php.net/manual/ru/class.dateinterval.php)

public [format](https://www.php.net/manual/ru/datetime.format.php)(string $format): string

public [getOffset](https://www.php.net/manual/ru/datetime.getoffset.php)(): int

public [getTimestamp](https://www.php.net/manual/ru/datetime.gettimestamp.php)(): int

public [getTimezone](https://www.php.net/manual/ru/datetime.gettimezone.php)(): [DateTimeZone](https://www.php.net/manual/ru/class.datetimezone.php)|false

public [\_\_wakeup](https://www.php.net/manual/ru/datetime.wakeup.php)(): void

}

12.Опишите использование класса DateTimeZone.

Листинг - Примеры констант и методов DateTimeZone

class DateTimeZone {

/\* Константы \*/

const int [AFRICA](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.africa) = 1;

const int [AMERICA](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.america) = 2;

const int [ANTARCTICA](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.antarctica) = 4;

const int [ARCTIC](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.arctic) = 8;

const int [ASIA](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.asia) = 16;

const int [ATLANTIC](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.atlantic) = 32;

const int [AUSTRALIA](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.australia) = 64;

const int [EUROPE](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.europe) = 128;

const int [INDIAN](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.indian) = 256;

const int [PACIFIC](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.pacific) = 512;

const int [UTC](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.utc) = 1024;

const int [ALL](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.all) = 2047;

const int [ALL\_WITH\_BC](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.all-with-bc) = 4095;

const int [PER\_COUNTRY](https://www.php.net/manual/ru/class.datetimezone.php" \l "datetimezone.constants.per-country) = 4096;

/\* Методы \*/

public [\_\_construct](https://www.php.net/manual/ru/datetimezone.construct.php)(string $timezone)

public [getLocation](https://www.php.net/manual/ru/datetimezone.getlocation.php)(): array|false

public [getName](https://www.php.net/manual/ru/datetimezone.getname.php)(): string

public [getOffset](https://www.php.net/manual/ru/datetimezone.getoffset.php)([DateTimeInterface](https://www.php.net/manual/ru/class.datetimeinterface.php) $datetime): int

public [getTransitions](https://www.php.net/manual/ru/datetimezone.gettransitions.php)(int $timestampBegin = PHP\_INT\_MIN, int $timestampEnd = PHP\_INT\_MAX): array|false

public static [listAbbreviations](https://www.php.net/manual/ru/datetimezone.listabbreviations.php)(): array

public static [listIdentifiers](https://www.php.net/manual/ru/datetimezone.listidentifiers.php)(int $timezoneGroup = DateTimeZone::ALL, ?string $countryCode = null): array

}

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

1. Goutte
2. GoogChart
3. JpGraph
4. Imagine
5. Php Graphic Works
6. Zebra Image
7. Php5 Image Manipulation
8. Dynamic Dummy Image Generator
9. WideImage
10. Image Cache

14.Опишите основные возможности библиотеки GD.

GD Graphics Library (GD) — программная библиотека, написанная Томасом Баутелом (Thomas Boutell) и другими разработчиками для динамической работы с изображениями.

В PHP библиотека GD значительно расширена. Начиная с версии PHP 4.3 входит в стандартную поставку интерпретатора. До этой версии могла подключаться как отдельная библиотека. Поддерживает почти все существующие форматы графики для использования в веб: PNG, JPEG, GIF, ICO и различные методы работы с графическими файлами (применение фильтров, текст, изменение размера, и прочее).

Функции GD и функции для работы с изображениями:

* getimagesize — Получение размера изображения
* image\_type\_to\_extension — Получение расширения файла для типа изображения
* mage2wbmp — Выводит изображение в браузер или пишет в файл
* imagealphablending — Задание режима сопряжения цветов для изображения
* imagearc — Рисование дуги
* imageavif — Выводит изображение в браузер или пишет в файл
* imagebmp — Вывести BMP-изображение в браузер или файл
* imagechar — Рисование символа по горизонтали
* imagecharup — Рисование символа вертикально
* imagecolorallocate — Создание цвета для изображения
* imagecolorallocatealpha — Создание цвета для изображения
* imagecolorat — Получение индекса цвета пиксела
* imagecolorexact — Получение индекса заданного цвета
* imagecolorset — Установка набора цветов для заданного индекса палитры
* imagecolorsforindex — Получение цветов, соответствующих индексу
* imagecolorstotal — Определение количества цветов в палитре изображения
* imagecopy — Копирование части изображения
* imagecopyresized — Копирование и изменение размера части изображения
* imagecreate — Создание нового палитрового изображения
* imagecreatefromavif — Создаёт новое изображение из файла или URL
* imagecreatefrombmp — Создаёт новое изображение из файла или URL
* imagecreatetruecolor — Создание нового полноцветного изображения
* imagecrop — Обрезать изображение до заданного прямоугольника
* imagedashedline — Рисование пунктирной линии
* imagedestroy — Уничтожение изображения
* imageellipse — Рисование эллипса
* imagefill — Заливка
* imagefilledarc — Рисование и заливка дуги
* imagefilledellipse — Рисование закрашенного эллипса
* imagefilledpolygon — Рисование закрашенного многоугольника
* imagefilledrectangle — Рисование закрашенного прямоугольника
* imagefilltoborder — Заливка цветом
* и т.д.

15.Как использовать Composer для подключения библиотек к проекту?

Composer скачивает и устанавливает библиотеки по их имени. Это означает, что сначала нужно найти нужную библиотеку, перейти на её сайт, и найти там в описании её имя. Например, название библиотеки может быть таким: fzaninotto/faker.

Теперь мы можем попросить composer установить библиотеку. Для этого введите команду composer require <имя библиотеки>. Composer загрузит и установит библиотеку в папку vendor. Останется подключить установленную библиотеку в сценарии и можно её использовать.

Пример для библиотеки для валидации форм — GUMP внутри php:

<?php

require 'vendor/autoload.php';

$rules = [

'email' => 'required|valid\_email',

'password' => 'required|min\_len,8',

'login' => 'required|alpha\_numeric',

'phone' => 'phone\_number'

];

$gump = new GUMP('ru');

$gump->validation\_rules($rules);

$validated\_data = $gump->run($\_POST);

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

PEAR - это аббревиатура от "PHP Extension and Application Repository" (Репозиторий приложений и модулей PHP).

PEAR - это:

* структурированная библиотека открытого кода, созданная для пользователей PHP;
* система управления пакетами и распространения кода среди разработчиков;
* стандарт написания PHP-кода (подробнее о стандарте см. здесь);
* базовые классы PHP-кода (подробнее о базовых классах см. здесь;
* библиотека дополнительных модулей для PHP (The PHP Extension Code Library, PECL), подробную информацию о PECL можно узнать здесь;
* веб-сайт, листы рассылки и зеркала для загрузки - все это предназначено для поддержания и развития сообщества разработчиков PHP/PEAR.

Различия PEAR и Composer:

* Невозможно установить пакет в PEAR для одного проекта, все пакеты устанавливаются глобально, тогда как в Composer можно выбирать;
* Отсутствие управления зависимостями в PEAR;
* В PEAR лучше совместимость с PECL. Хотя, существует pickle , который использует Composer и позволяет вам определять собственные зависимости расширений PHP в вашем файле composer.json . Проект pickle в настоящее время находится в стадии разработки и пока не должен считаться зрелым.

17.Как использовать PEAR для установки библиотек?

pear install <имя библиотеки>

18.Как использовать Composer для обработки зависимостей PEAR?

Флаги для установки модулей с зависимостями:

* all deps - необходимые и необязательные зависимости
* only req deps - только необходимые зависимости [я не ебу, какое второе слово...]

19.Что такое PECL?

PECL — это репозиторий модулей для PHP, написанных на C, доступных через систему пакетов PEAR. PECL был создан, когда возникла проблема удаления некоторых модулей из стандартной поставки PHP. Модули PECL разработаны в соответствии со стандартами кодирования, которые приняты командой разработчиков PHP.

20.Как называется репозиторий, содержащий Composer-совместимые библиотеки?

Pickle

21.С помощью какой библиотеки можно получить детальный отчет о работе приложения?

PHP BenchMark

22.С помощью какой библиотеки можно упростить себе работу с регулярными выражениями в PHP?

RegExp Builder

23.С помощью какой библиотеки возможны быстрые и эффективные запросы на PHP, данная библиотека является аналогом jQuery.

phpQuery

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

mPDF

25.С помощью какой библиотеки возможен эффективный парсинг HTML/XML?

Goutte

26.С помощью какой библиотеки возможно создание диаграмм на движке GOOGLE?

GoogChart

27.С помощью какой библиотеки возможно сканирование конфигурационного файла PHP на предмет безопасности?

IniScan

28.С помощью какой библиотеки возможен анализ HTML и удаление вредоносного кода для защиты от XSS атак.

HTML Purifier

29.С помощью какой библиотеки возможно построение графиков, диаграмм и другого структурированного контента на PHP?

JpGraph

30.С помощью какой библиотеки облегчается процесс загрузки и валидации файлов на PHP?

Upload

# Список использованной литературы

1. Документация Docker — Текст : электронный // Docker docs : [Электронный ресурс]. — URL: https://docs.docker.com/ (дата обращения: 04.09.2021).
2. Руководство по php — Текст : электронный // php : [Электронный ресурс]. — URL: https://www.php.net/ (дата обращения: 22.09.2021).
3. Docker Hub — Текст : электронный // Docker Hub : [Электронный ресурс]. — URL: https://hub.docker.com/ (дата обращения: 04.09.2021).
4. Справочник по php — Текст : электронный // Docker Hub : [Электронный ресурс]. — URL: https://php.ru/manual/ (дата обращения: 22.09.2021).