# Back-end

# Back-end

Бэкенд – серверное приложение.

Основные задачи:

- сетевой ввод-вывод – это общение с одной стороны с proxy – прием http-запроса и ответ на него, а с другой стороны общение со всевозможными сервисами, которые хранят данные – это могут быть БД, очереди, memcached и т.п.
- склеивание строк – сериализовать данные в JSON, сформировать шаблон на основе html, посчитать sh1 или md5? выполнить сжатие данных.

# Параллелизм запросов

Если мы говорим о бэкенде, а его производительность во многом будет определять производительность в целом нашего продукта, то у нас может быть 2 цели по оптимизации:

- заставить его "переваривать" все большее количество запросов в секунду, т.е. увеличить его производительность,
- вторая цель – продуктовая – это уменьшение времени отклика, т.е., чтобы каждый запрос выполнялся намного быстрее, для пользователя результат появлялся быстрее и т.п.

# PHP

**PHP** — скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений.

В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.

# PHP

Синтаксис **PHP** подобен синтаксису языка **Си**. Некоторые элементы, такие как ассоциативные массивы и цикл foreach, заимствованы из Perl.

Для работы программы не требуется описывать какие-либо переменные, используемые модули и т. п. Любая программа может начинаться непосредственно с оператора **PHP**.

Простейшая программа Hello world на **PHP** выглядит следующим образом:

```
<?php
  echo 'Hello, world!';
?>
```

# PHP

**PHP** исполняет код, находящийся внутри ограничителей, таких как <?php ?>. Всё, что находится вне ограничителей, выводится без изменений. В основном это используется для вставки **PHP**-кода в **HTML**-документ.

# PHP

Имена переменных начинаются с символа $, тип переменной объявлять не нужно. Имена переменных и констант чувствительны к регистру символов. Имена классов, методов классов и функций к регистру символов не чувствительны.

Переменные обрабатываются в строках, заключённых в двойные кавычки, и heredoc-строках (строках, созданных при помощи оператора <<<). Переменные в строках, заключённых в одинарные кавычки, не обрабатываются.

# PHP

**PHP** рассматривает переход на новую строку как пробел, так же как **HTML** и другие языки со свободным форматом. Инструкции разделяются с помощью точки с запятой (;), за исключением некоторых случаев, после объявления конструкции if/else и циклов.

Переменные в функцию можно передавать как по значению, так и по ссылке (используется знак &).

# PHP

**PHP** является языком программирования с динамической типизацией, не требующим указания типа при объявлении переменных, равно как и самого объявления переменных.

Преобразования между скалярными типами зачастую осуществляются неявно без дополнительных усилий (впрочем, **PHP** предоставляет широкие возможности и для явного преобразования типов).

К скалярным типам данных относятся:

- целочисленный тип (integer)
- число с плавающей точкой (float, double)
- логический тип (boolean)
- строковый тип (string)

# PHP

К нескалярным типам относятся:

- массив (array)
- объект (object)
- внешний ресурс (resource)
- неопределенное значение (null)
- К псевдотипам[26] относятся:
- mixed любой тип
- number число (integer либо float)
- callback (string или анонимная функция)
- void отсутствие параметров

# PHP

Обращение к переменным осуществляется с помощью символа $, за которым следует имя переменной. Данная конструкция может быть применена также для создания динамических переменных и функций. Например:

- $a = 'I am a'; // Запись значения в переменную $a
- echo $a; // Вывод переменной $a
- $b = 'a';
- echo $$b; // Вывод переменной $a (дополнительный $ перед переменной $b)
- echo ${'a'}; // Вывод переменной $a
- function_name(); // Вызов функции function_name
- $c = 'function_name';
- $c(); // Вызов функции function_name
- $d = 'Class_name';

# PHP

**PHP**-скрипты обычно обрабатываются интерпретатором в порядке, обеспечивающем кроссплатформенность разработанного приложения:

- лексический анализ исходного кода и генерация лексем,
- синтаксический анализ полученных лексем,
- генерация байт-кода,
- выполнение байт-кода интерпретатором (без создания исполняемого файла).

# 8 of the Best PHP Frameworks for Web Development

1. Laravel
2. CodeIgniter
3. Symfony
4. Zend
5. Phalcon
6. CakePHP
7. Yii
8. FuelPHP

Source https://www.hostinger.com/tutorials/best-php-framework

# LAMP

LAMP — акроним, обозначающий набор (комплекс) серверного программного обеспечения, широко используемый во Всемирной паутине. LAMP назван по первым буквам входящих в его состав компонентов:

- Linux — операционная система Linux;
- Apache — веб-сервер;
- MariaDB / MySQL — СУБД;
- PHP — язык программирования, используемый для создания веб-приложений (помимо PHP могут подразумеваться другие языки, такие как Perl и Python).

# LAMP

Акроним LAMP может использоваться для обозначения:

- Инфраструктуры веб-сервера;
- Парадигмы программирования;
- Пакета программ.

# NODEJS

**Node** или **Node.js** — программная платформа, основанная на движке V8 (транслирующем **JavaScript** в машинный код), превращающая **JavaScript** из узкоспециализированного языка в язык общего назначения.

**Node.js** добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода.

# NODEJS

**Node.js** применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на **Node.js** и десктопные оконные приложения и даже программировать микроконтроллеры.

В основе **Node.js** лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.

# NPM

NPM — это менеджер пакетов, используемый Node.js приложениями (https://npmjs.com)

// **package.json** after "**npm init**"

```
{
        "name": "useless-app",
        "version": "1.0.0",
        "description": "",
        "main": "index.js",
        "scripts": {
                "test": "echo \"Error: no test specified\" && exit 1",
                "start": "node index.js"
        },
        "author": "", "license": "ISC"
}
```

# 14 Best NodeJS Frameworks for Developers in 2019

- Express.JS
- Socket.io
- Meteor.JS
- Koa.JS
- Sails.js
- MEAN.io
- Nest.JS
- Loopback.io
- Keystone.JS
- Feathers.JS
- Hapi.JS
- Strapi.io
- Restify.JS
- Adonis.JS

Source https://www.tecmint.com/best-nodejs-frameworks-for-developers/

All http://nodeframework.com/

# Гайды

PHP

- https://www.php.net/manual/ru/tutorial.php
- http://www.php.su/lessons/
- https://phptherightway.com/

Nodejs

- https://nodeguide.ru/
- https://medium.com/devschacht/node-hero-6a07ef8d822d
- https://medium.freecodecamp.org/the-definitive-node-js-handbook-6912378afc6e (есть частичный перевод от RUVDS на хабре)

# Примеры

- PHP http://phpfiddle.org/main/code/w991-4nym
- PHP https://repl.it/@terra_ra/PHP-Step-3-calculator
- PHP+Laravel https://implode.io/5eTSOO
- Nodejs https://repl.it/@zanedb/nodejs-memory-example
- Nodejs+Expressjs https://repl.it/@protonbobby/express-on-nodejs
- Nodejs+Expressjs https://repl.it/@MikeShi42/competitive-2048-demo-final