

# ОСНОВЫ ВЕБ-ТЕХНОЛОГИЙ

Принципы работы веб-технологий. Многоуровневая архитектура приложений. Принципы и особенности работы веб-приложений.

**Author: Svyatoslav Kulikov**  
**Training And Education Manager**  
**[svyatoslav\\_kulikov@epam.com](mailto:svyatoslav_kulikov@epam.com)**

## Содержание

1. **Обобщённая схема веб-технологий**
2. **Многоуровневая архитектура приложений**
3. **Принципы работы веб-приложений**
4. **Архитектура веб-приложений**
  - Java
  - ASP
  - PHP, Perl, Python, Ruby
  - Сравнение архитектур
5. **Процесс разработки веб-приложений**
6. **Рекомендуемые источники информации**

# ОБОБЩЁННАЯ СХЕМА ВЕБ-ТЕХНОЛОГИЙ

## Обобщённая схема веб-технологий

Веб-приложения

Веб-сервисы

Языки  
программирования

Языки разметки и  
оформления

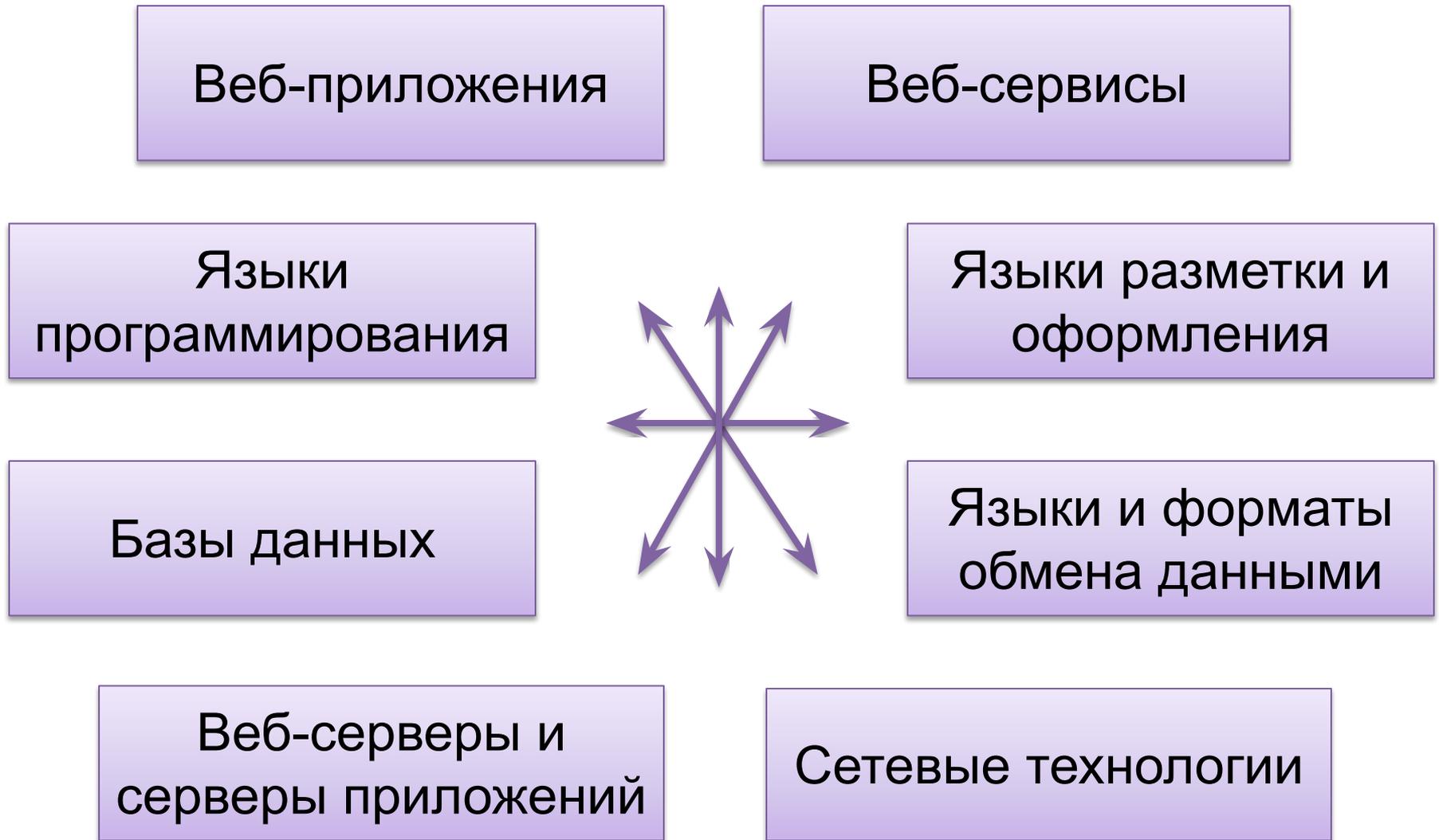
Базы данных

Языки и форматы  
обмена данными

Веб-серверы и  
серверы приложений

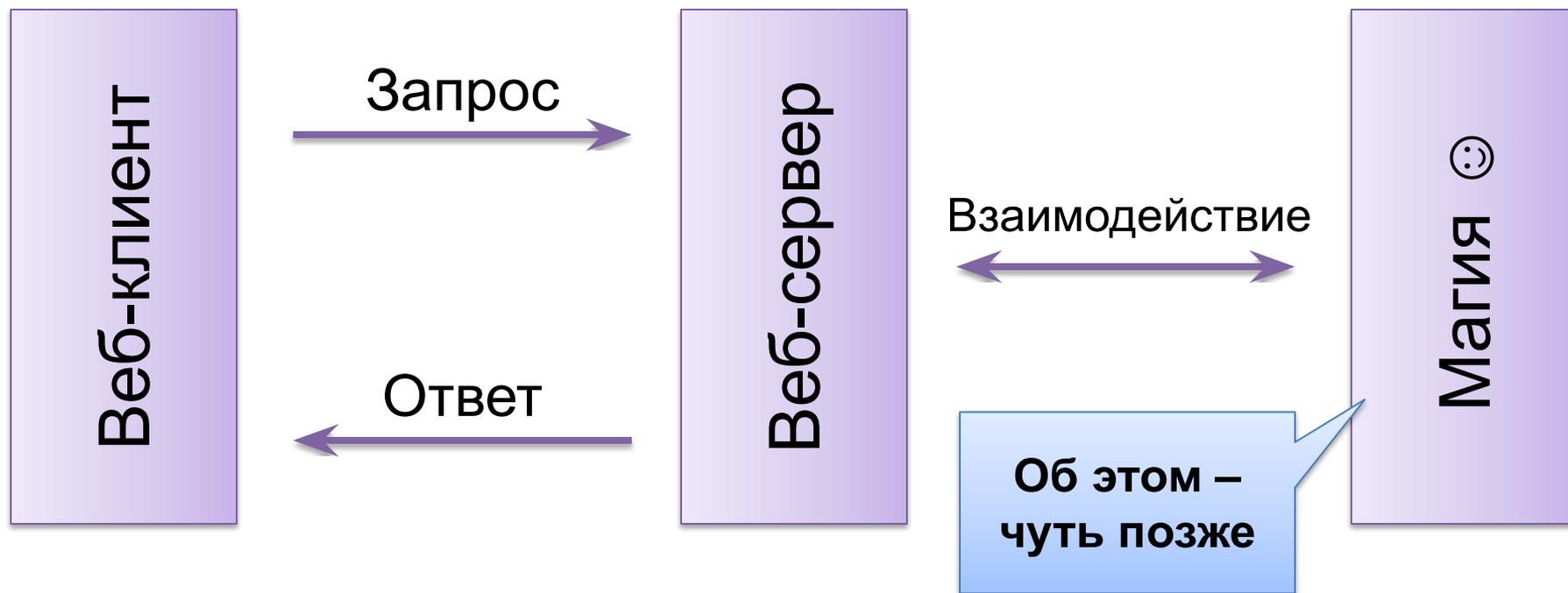
Сетевые технологии

## Обобщённая схема веб-технологий



## Веб-серверы и серверы приложений

**Веб-сервер** (web server) – специальное ПО, принимающее запросы (request) (как правило – HTTP) от веб-клиентов (web client), и генерирующее соответствующий ответ (response).



## Веб-серверы и серверы приложений

Для отображения одной страницы может выполняться много запросов: браузеру необходимо получить множество различных элементов – css-файлы, js-файлы, картинки и т.д.

Type	Status	Response Size	Time
GET watch.js	304 Not Modified	17 KB	17ms
GET tutby-ico.png	304 Not Modified	9.2 KB	2ms
GET socialbar~r20121025.css?227	200 OK	987 B	189ms
GET widgets.js	304 Not Modified	25.4 KB	49ms
GET all.js#xfbml=1	304 Not Modified	0 B	46ms
GET openapi.js?49	200 OK	18.6 KB	102ms
GET cnt?id=518&suid=5185382...%	200 OK	43 B	4ms
GET 17863687?rn=91392&wmode	200 OK	74 B	18ms
GET widget_subscribe.php?ap...w.t	200 OK	1.3 KB	149ms
GET upload.gif	304 Not Modified	265 B	96ms
GET ga.js	304 Not Modified	15.3 KB	36ms
GET vk.png	304 Not Modified	1.7 KB	2ms

## Веб-серверы и серверы приложений

Основные функции веб-сервера...



**КАКИЕ?**

## Веб-серверы и серверы приложений: функции веб-сервера

Обработка запросов и генерация ответов.

Обеспечение опосредованного доступа к ФС.

Обеспечение интерфейса к серверу приложений.

Контроль прав доступа.

Шифрование трафика.

Балансировка нагрузки.

Маршрутизация запросов (по виртуальным хостам).

Обработка ошибочных ситуаций.

Протоколирование.

Наиболее распространённые веб-серверы.

<b>Apache</b>	<b>47.3%</b>
<b>Nginx</b>	<b>37.1%</b>
<b>Microsoft-IIS</b>	<b>10.3%</b>
LiteSpeed	3.1%
Google Servers	1.0%
Tomcat	0.5%
Node.js	0.4%
Apache Traffic Server	0.3%
IdeaWebServer	0.3%
Tengine	0.2%
Cowboy	0.1%
Lighttpd	0.1%
Oracle Servers	0.1%
IBM Servers	0.1%
Kestrel	0.1%
Jetty	0.1%

По данным [http://w3techs.com/technologies/overview/web\\_server/all](http://w3techs.com/technologies/overview/web_server/all).

## Веб-серверы и серверы приложений

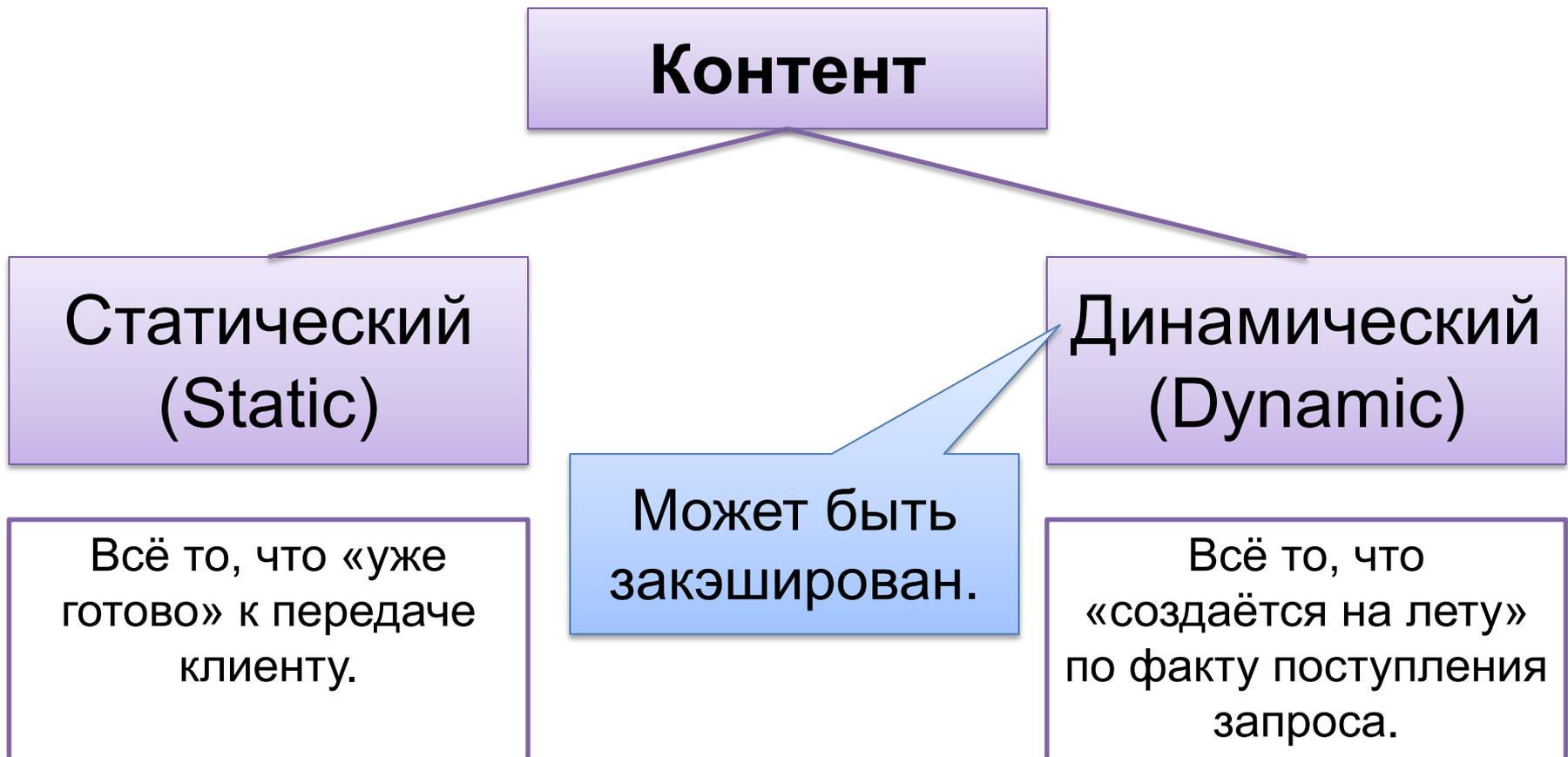
А ещё?

**Что ещё нам надо  
знать о работе веб-  
сервера?**

**Что может негативно  
сказаться на  
работоспособности  
сервера?**

## Веб-серверы и серверы приложений

Веб-сервер в ответ на запрос отдаёт «некий контент»:



## Веб-серверы и серверы приложений

Веб-сервер также может выполнять «преобразование URL».

**Преобразование URL** (URL rewriting) – способ представить динамические URL в максимально удобной для восприятия человеком форме.

[www.site.com/catalog/notebooks/hp/new/](http://www.site.com/catalog/notebooks/hp/new/)

VS

[www.site.com/index.php?page=catalog&category=notebooks  
&vendor=hp&mode=new](http://www.site.com/index.php?page=catalog&category=notebooks&vendor=hp&mode=new)

См. подробности здесь:

[http://httpd.apache.org/docs/current/mod/mod\\_rewrite.html](http://httpd.apache.org/docs/current/mod/mod_rewrite.html).

## Веб-серверы и серверы приложений

Один из способов реализации преобразования URL – использование модуля `mod_rewrite` веб-сервера Apache.

Не самый корректный, но самый простой пример. В файле `.htaccess` прописывается:

```
RewriteEngine On
RewriteBase /
RewriteRule .* index.php?url=$0 [QSA,L]
```

Теперь запрос к любому объекту будет переадресован на `index.php`.

См. подробности здесь:

[http://httpd.apache.org/docs/current/mod/mod\\_rewrite.html](http://httpd.apache.org/docs/current/mod/mod_rewrite.html).

## Веб-серверы и серверы приложений

**Сервер приложений** (application server) – специальное ПО, выполняющее для взаимодействующих с ним приложений ряд специфических функций, таких как управление производительностью, безопасностью, взаимодействие с операционной системой и т.п.

Частным случаем серверов приложений можно считать т.н. «среды исполнения» (.NET Framework, Java Runtime Environment, PHP и т.п.)



## Веб-серверы и серверы приложений: функции сервера приложений

Изолирование приложения от ОС и «железа».

Предоставление API для типичных действий.

Управление ресурсами.

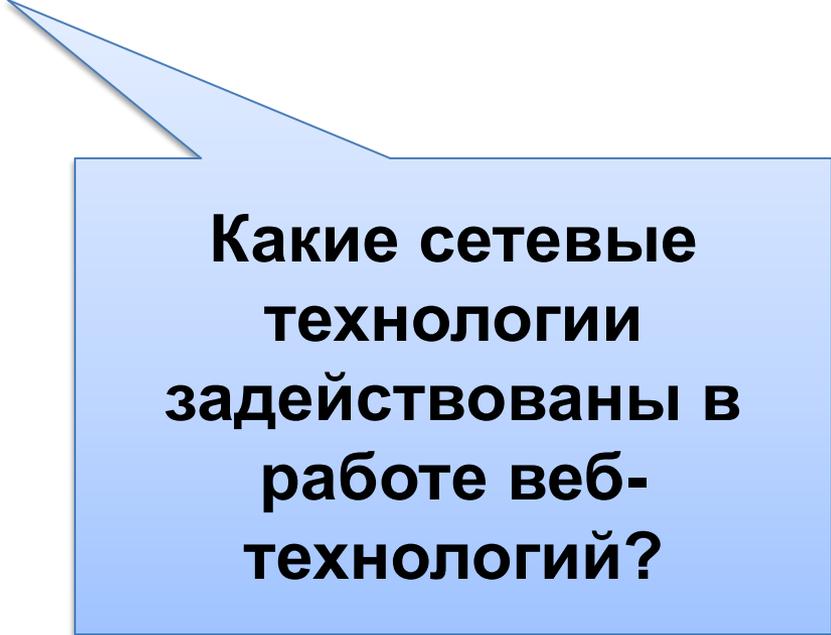
Протоколирование.

Оптимизация выполнения операций.

Балансировка нагрузки (в некоторых случаях).

Обработка ошибочных ситуаций.

**Сетевые технологии** (network technologies) – совокупность технологий, обеспечивающих функционирование компьютерных сетей и аппаратных и программных средств, использующих в своей работе компьютерные сети.



**Какие сетевые технологии задействованы в работе веб-технологий?**

## Сетевые технологии: что задействовано?

Для полноценного понимания сетевых технологий, задействованных в работе веб-приложений, рекомендуется ознакомиться с семиуровневой моделью ISO/OSI (особое внимание стоит уделить уровням 6-7).

См.: [http://en.wikipedia.org/wiki/OSI\\_model](http://en.wikipedia.org/wiki/OSI_model)

Если вкратце, то...

## Сетевые технологии: OSI-модель, уровни

Тип данных	Уровень	Функции
Данные	7. Прикладной	Доступ к сетевым службам
	6. Представления	Представление и кодирование данных
	5. Сеансовый	Управление сеансом связи
Сегменты	4. Транспортный	Прямая связь между конечными пунктами и надежность
Пакеты	3. Сетевой	Определение маршрута и логическая адресация
Кадры	2. Канальный	Физическая адресация
Биты	1. Физический	Работа со средой передачи, сигналами и двоичными данными

## Сетевые технологии: OSI-модель, протоколы

Тип данных	Уровень	Функции
Данные	7. Прикладной	http, https, ftp, ...

**HTTPS** — расширение протокола HTTP, поддерживающее шифрование. Данные, передаваемые по протоколу HTTPS, «упаковываются» в криптографический протокол SSL или TLS.

Метод HTTP — последовательность из любых символов, кроме управляющих и разделителей, указывающая на основную операцию над ресурсом.

**GET**

**POST**

**OPTIONS**

**HEAD**

**PUT**

**PATCH**

**DELETE**

**TRACE**

**CONNECT**



## Сетевые технологии: HTTP коды

**Код состояния HTTP** — часть первой строки ответа сервера при запросах по протоколу HTTP.

1xx Informational

2xx Success

3xx Redirection

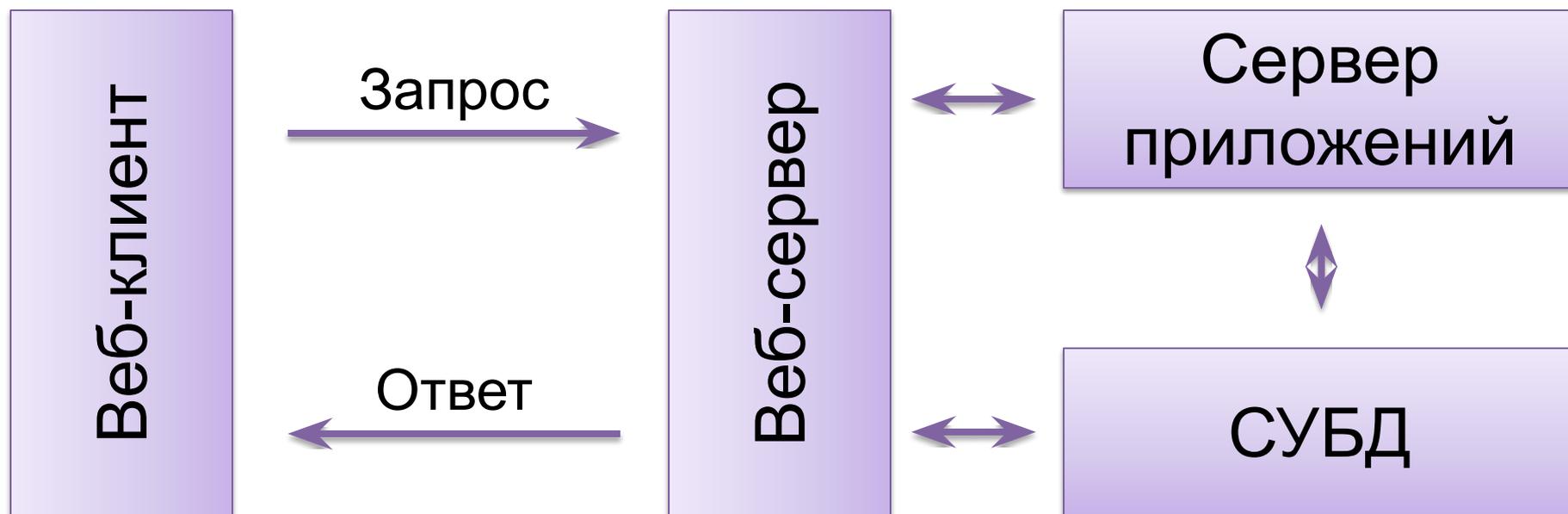
4xx Client Error

5xx Server Error

Hits by Response Code		
Code 200 - OK	15.45%	<b>7527</b>
Code 206 - Partial Content	0.13%	<b>64</b>
Code 301 - Moved Permanently	0.03%	<b>14</b>
Code 302 - Found	0.18%	<b>90</b>
Code 304 - Not Modified	83.46%	<b>40656</b>
Code 403 - Forbidden	0.00%	<b>1</b>
Code 404 - Not Found	0.72%	<b>353</b>
Code 500 - Internal Server Error	0.02%	<b>10</b>

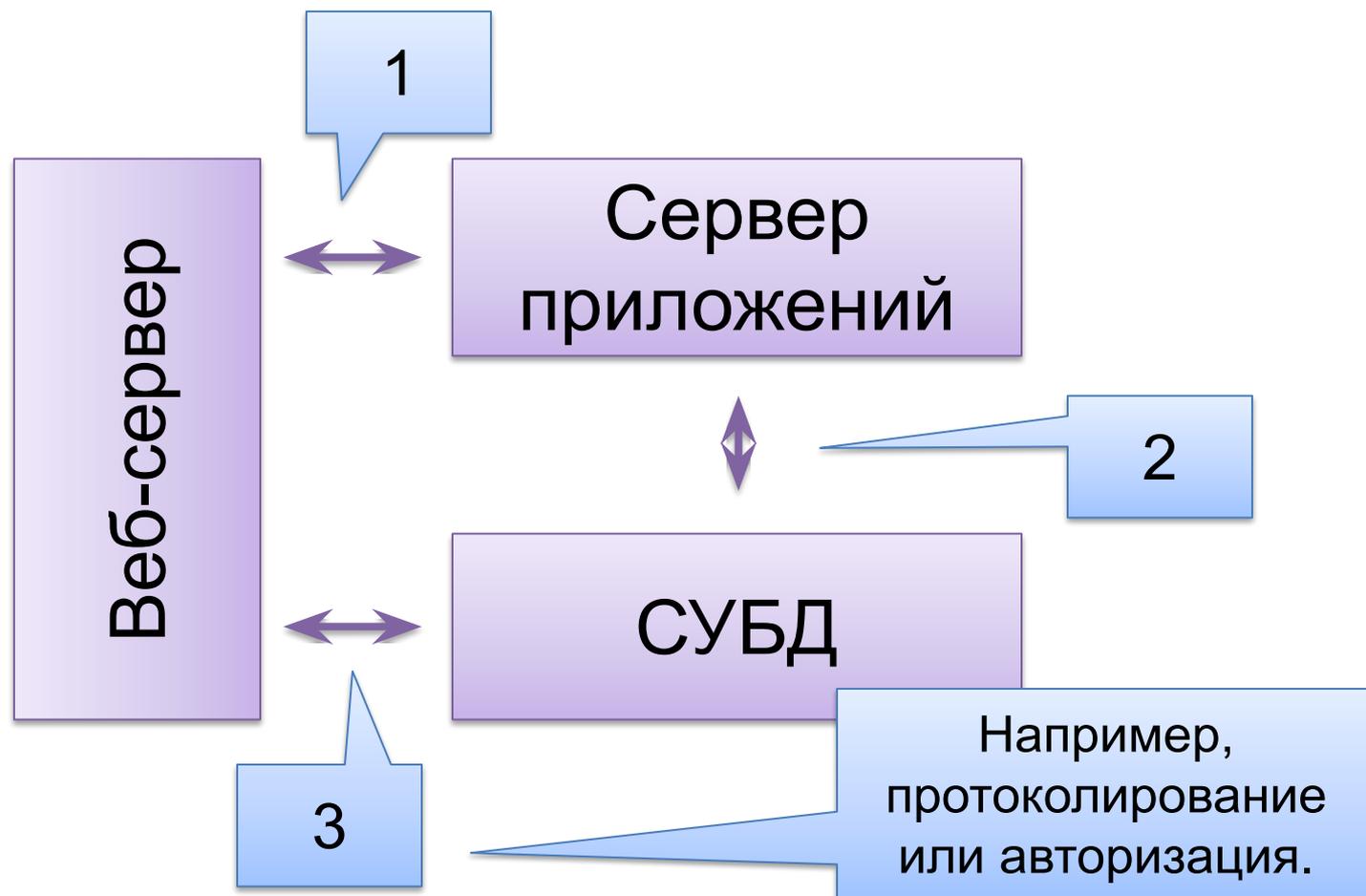
## Базы данных

**Базы данных** (databases) под управлением СУБД (DBMS, database management systems, системы управления базами данных) являются универсальным хранилищем информации для веб-приложений.



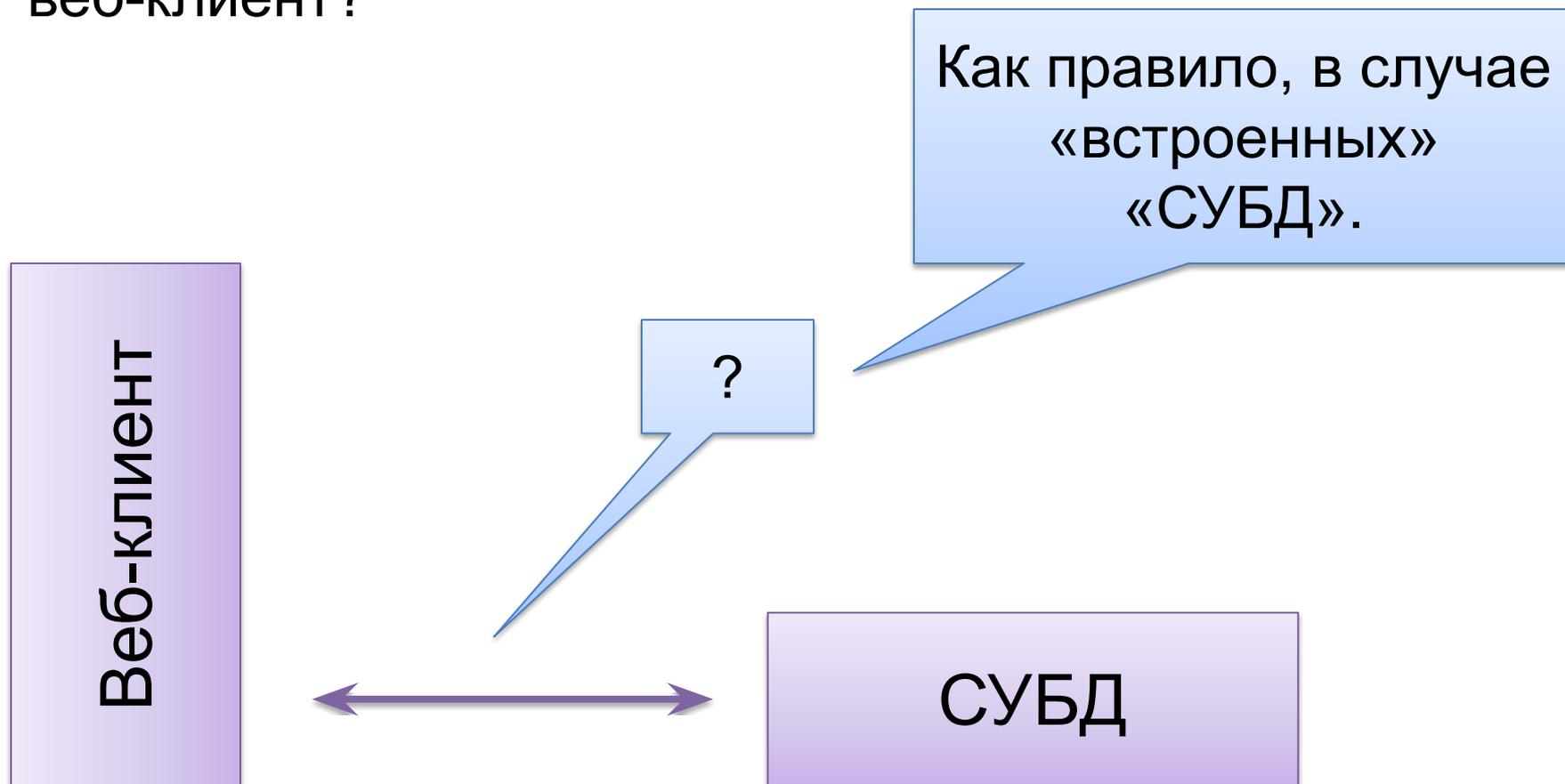
## Базы данных

В каких случаях происходят эти варианты взаимодействий?

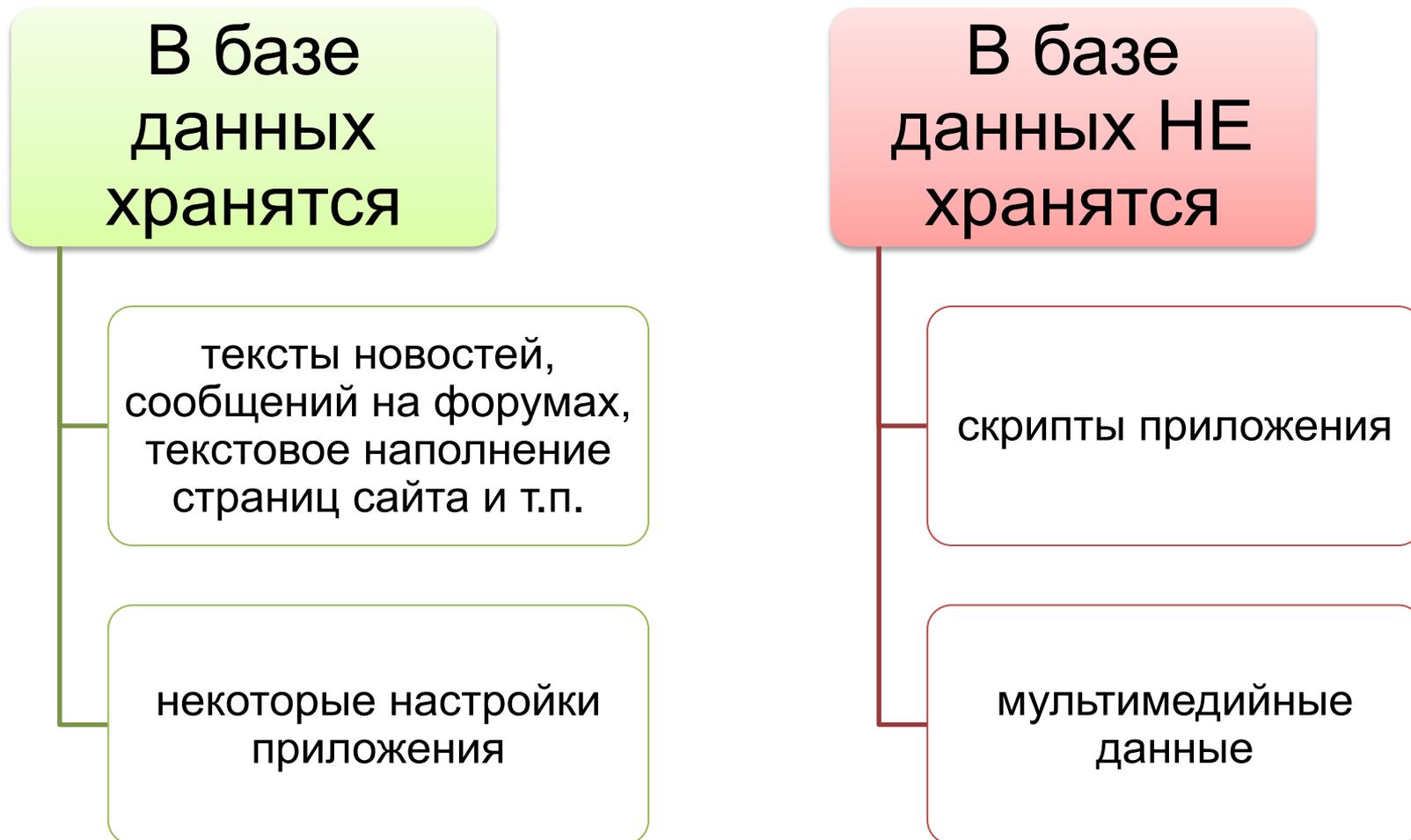


## Базы данных

А бывают ли ситуации, когда с СУБД взаимодействует веб-клиент?



Что хранится и НЕ хранится в БД:



## «Что где хранится» (расширенная версия)

**Хранится в БД**

**Хранится в файловой системе**

**Генерируется динамически**

Структура приложения (сайта)

Скрипты

Готовые страницы

Текстовое наполнение страниц

Мультимедийная информация

Текстовая информация

Некоторые настройки

Некоторые настройки

Мультимедийная информация

Кэш готовых страниц

Кэш готовых страниц

## Языки и форматы и обмена данными

XML

DTD

WSDL

SOAP

JSON

RSS

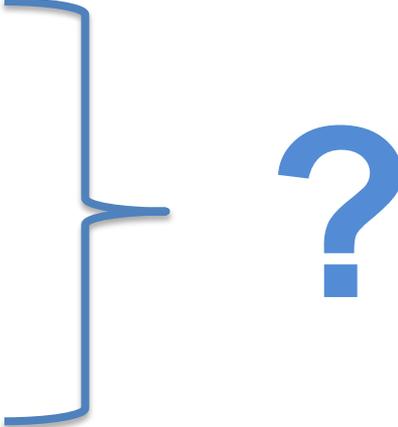
и т.д.

Загляните на <http://w3schools.com>

## Языки и форматы и обмена данными

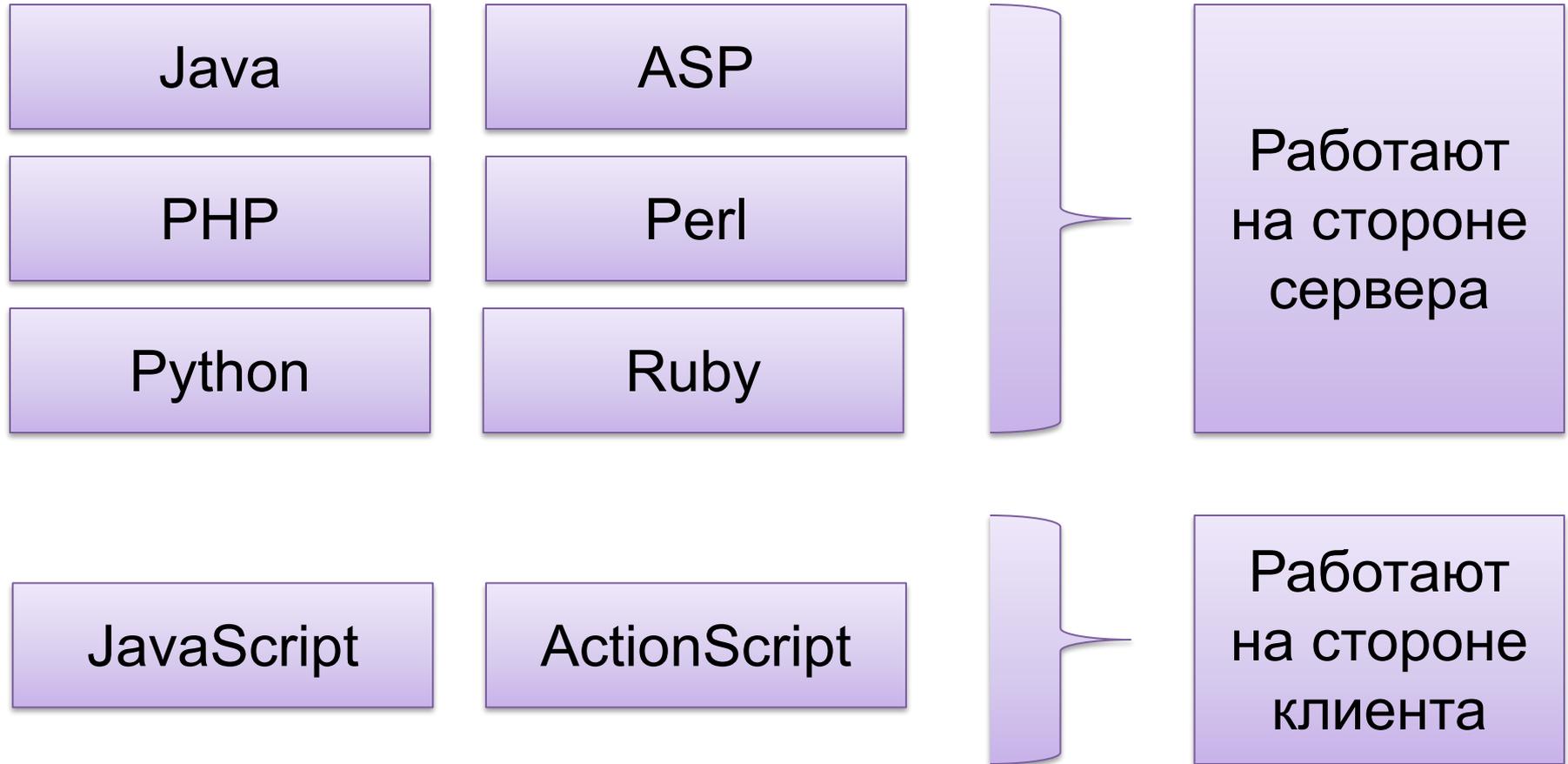
**XML** (eXtensible Markup Language) – формат хранения и обмена структурированными данными.

**DTD –**  
**WSDL –**  
**SOAP –**  
**JSON –**  
**RSS –**



Может быть, мы  
ещё что-то  
забыли?

## Языки программирования



Может быть, мы ещё что-то забыли?

## Языки разметки и оформления

HTML

HTML5

CSS

CSS3

Загляните на <http://w3schools.com>

## Языки разметки и оформления

**HTML** (Hypertext Markup Language, язык гипертекстовой разметки) – специальный язык РАЗМЕТКИ, используемый для описания структуры документа.

**XHTML** (Extensible Hypertext Markup Language, расширяемый язык гипертекстовой разметки) – более строгий «вариант» HTML, базирующийся на спецификациях XML.

**HTML5** – «улучшенная» версия HTML, добавляющая широкий спектр новых возможностей. Полный список нововведений в HTML5 можно увидеть здесь:

<http://www.w3.org/TR/html5-diff/>

## Языки разметки и оформления

**CSS** (Cascading Style Sheets, каскадные таблицы стилей) – язык описания внешнего вида документа (написанного с использованием HTML).

**CSS3 (CSS4)** – новые версии CSS, следующие той же логике развития, что и HTML5 по отношению к «классическому» HTML (XHTML).

## Веб-приложения

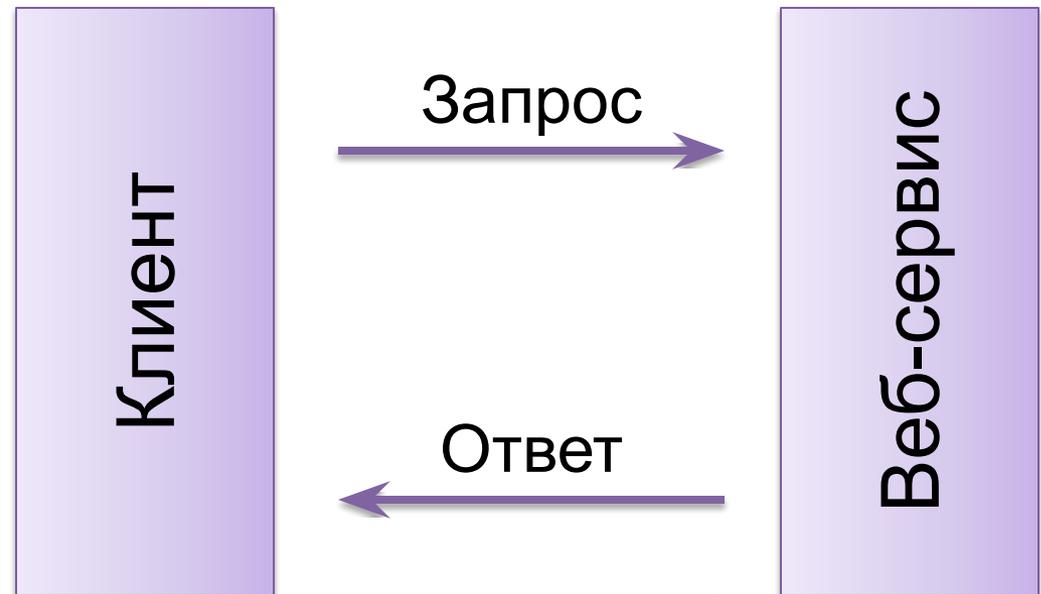
**Веб-приложение** (web application) – приложение, использующее для своей работы Интернет или интранет (локальную сеть). Как правило, работает по архитектуре: «веб-клиент + веб-сервер + сервер приложений + СУБД».

Часто веб-приложения строятся так, что для их работы необходим браузер, но это – **не обязательное условие**.

Давайте назовём несколько примеров веб-приложений разных видов.

## Веб-сервисы

**Веб-сервис** (web service) – ПО, предоставляющее возможность обмена данными по сети между приложениями, устройствами и другими веб-сервисами.



Давайте приведём примеры веб-сервисов.

# МНОГОУРОВНЕВАЯ АРХИТЕКТУРА ПРИЛОЖЕНИЙ

## Многоуровневая архитектура приложений

Уровень 1

Интерфейс 1

Уровень 2

Интерфейс 2

⋮

Уровень N

Уровень  
представления

Уровень  
бизнес-логики

Уровень  
данных

## Многоуровневая архитектура приложений

Отвечает за ввод-вывод данных, их форматирование и отображение.

Уровень  
представления

Выполняет непосредственно возложенные на приложение задачи.

Уровень  
бизнес-логики

Обеспечивает операции по сохранению, извлечению и некоторой обработке данных.

Уровень  
данных

## Многоуровневая архитектура приложений

Браузер

Уровень  
представления

Веб-сервер и  
сервер  
приложений

Уровень  
бизнес-логики

СУБД

Уровень  
данных

## Многоуровневая архитектура приложений

Всегда ли схема именно такая? Или есть другие варианты?

Браузер

Уровень  
представления

Веб-сервер и  
сервер  
приложений

Уровень  
бизнес-логики

СУБД

Уровень  
данных

# Многоуровневая архитектура приложений



Каковы преимущества и недостатки многоуровневой архитектуры?

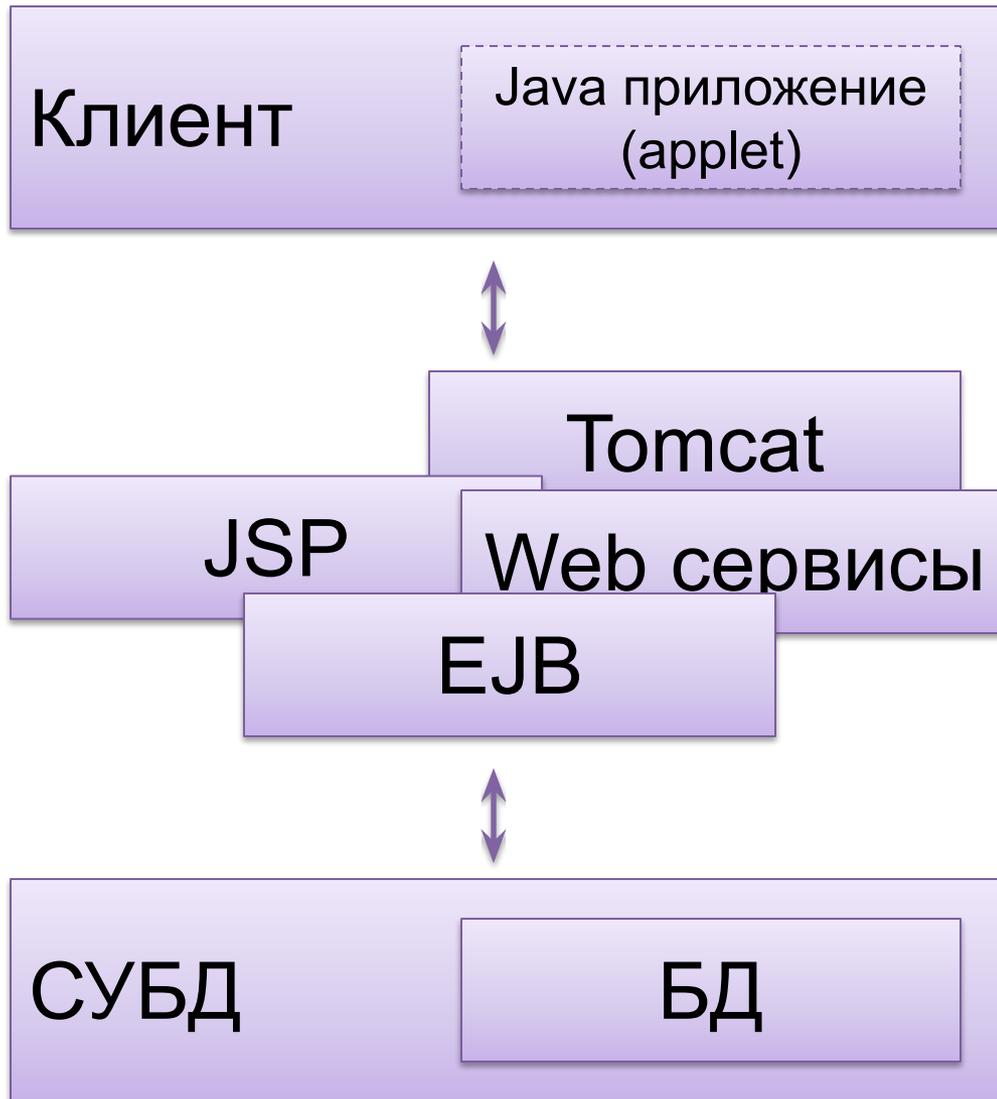
# ПРИНЦИПЫ РАБОТЫ ВЕБ-ПРИЛОЖЕНИЙ



# АРХИТЕКТУРА ВЕБ- ПРИЛОЖЕНИЙ

# JAVA

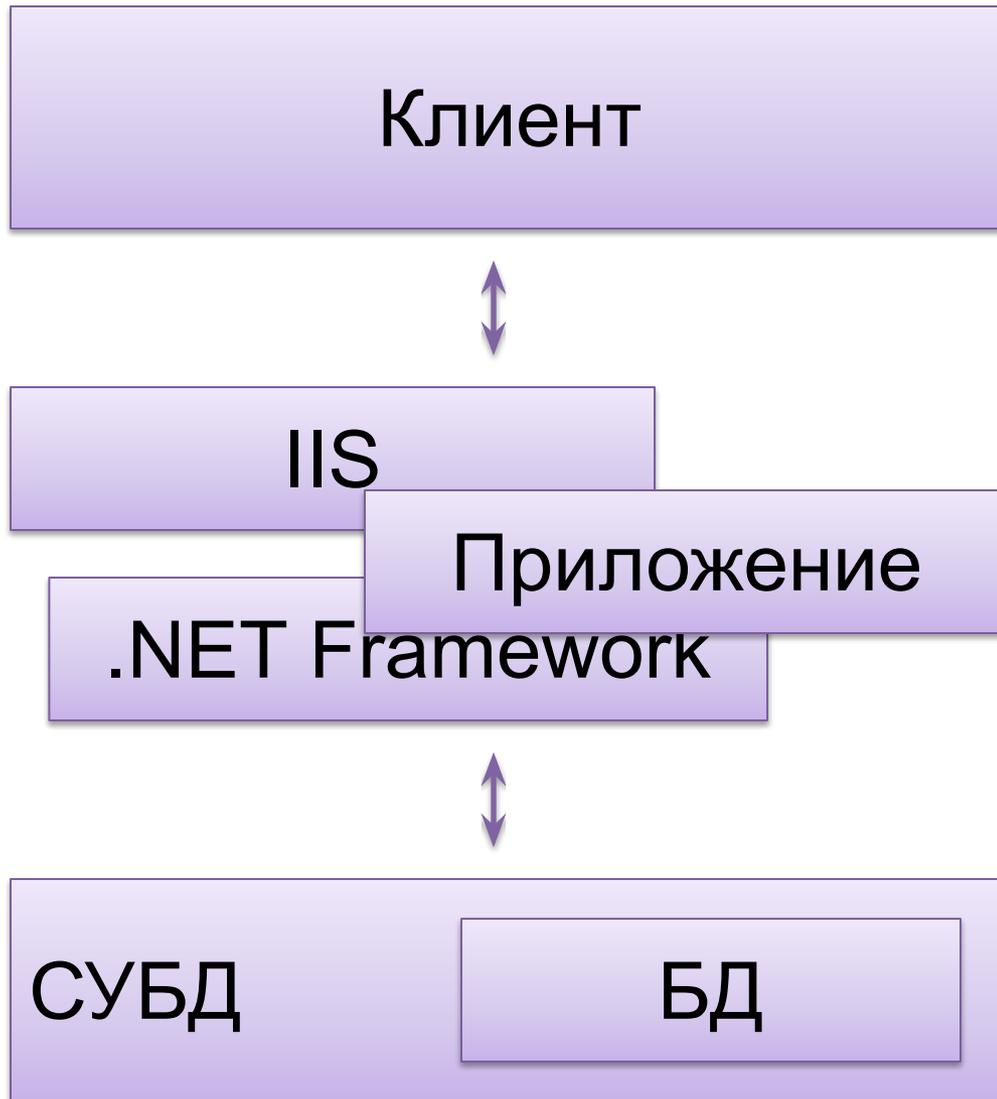
## Java (один из возможных вариантов архитектуры)



Всё верно? Ничего не забыли?

# ASP.NET

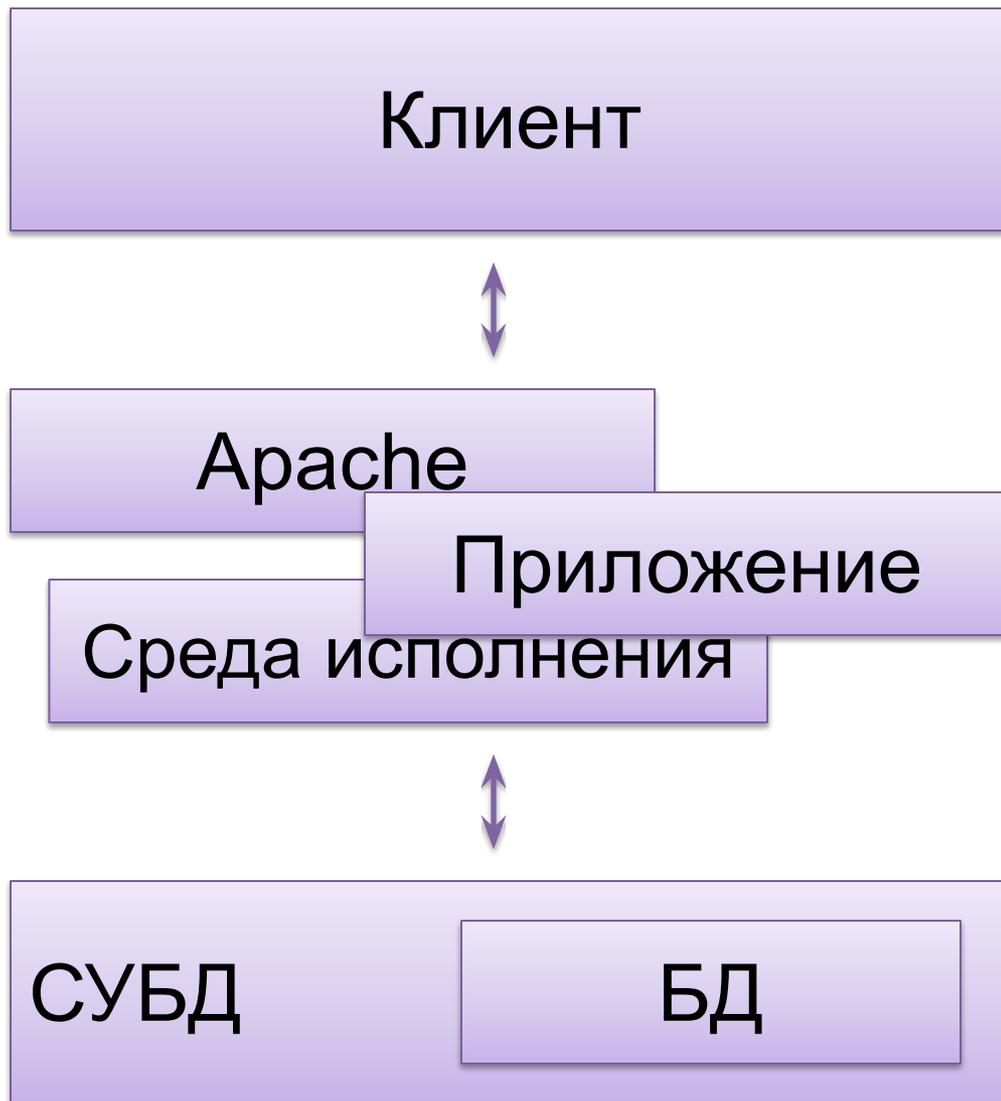
## ASP.NET (классический вариант архитектуры)



Всё верно? Ничего не забыли?

PHP, PERL, PYTHON, RUBY  
И Т.Д. И Т.П.

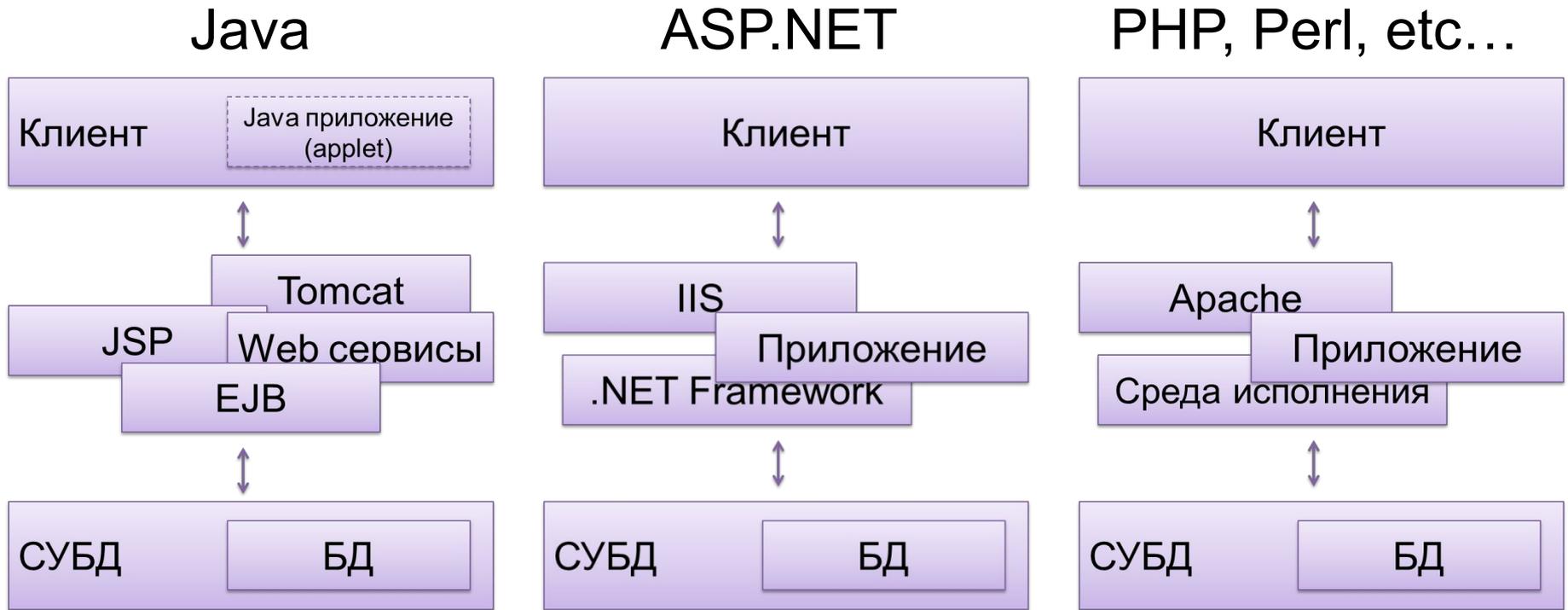
## PHP, Perl, Python, Ruby и т.д. (универсальная архитектура)



Всё верно? Ничего не забыли?

# СРАВНЕНИЕ АРХИТЕКТУР

## Сравнение архитектур



Что здесь выглядит похоже? А в чём отличия?

# ПРОЦЕСС РАЗРАБОТКИ ВЕБ-ПРИЛОЖЕНИЙ

# Процесс разработки веб-приложений

1

Разработка общей концепции приложения, создание структуры и т.д.

2

Создание графической концепции и графических макетов основных страниц приложения.

3

Вёрстка графического макета. Создание шаблонов основных страниц.

4

Разработка нового (или адаптация существующего) движка.

5

Разработка новой (или адаптация существующей) базы данных.

6

Финальное тестирование и размещение приложения на хостинге.

# WEB-BROWSER

**Браузер** или **веб-обозреватель** — прикладное программное обеспечение для просмотра веб-страниц; содержания веб-документов, компьютерных файлов и их каталогов; управления веб-приложениями; а также для решения других задач.

Cookie

Local Storage

Session Storage

## Браузеры, хранение данных

**Cookie** - небольшой фрагмент данных, отправленный веб-сервером и хранимый на компьютере пользователя.

Веб-клиент (обычно веб-браузер) всякий раз при попытке открыть страницу соответствующего сайта пересылает этот фрагмент данных веб-серверу в составе HTTP-запроса.

- Аутентификация пользователя
- Хранение настроек и предпочтений
- Отслеживание состояния сеанса доступа пользователя
- Ведение статистики о пользователях

**Куки легко перехватить и подменить**

**Интернет-хранилище** или **DOM-хранилище** — это программные методы и протоколы веб-приложения, используемые для хранения данных в веб-браузере. Интернет-хранилище представляет собой постоянное хранилище данных, похожее на куки, но со значительно расширенной ёмкостью и без хранения информации в заголовке запроса HTTP.

- локальное хранилище (localStorage)
- сессионное хранилище (sessionStorage)

# ХОСТИНГ

**Хóстинг** — услуга по предоставлению ресурсов для размещения информации на сервере, постоянно находящемся в сети (обычно Интернет).

- ОС
- поддержка CGI: Perl, PHP, Python, ASP, Ruby, JSP, Java
- поддержка .htaccess/.htpasswd (для Apache)
- поддержка баз данных

✓ Hosting provider

✓ Amazon AWS and Heroku, Docker, Microsoft Azure

✓ Собственный сервер

# РЕКОМЕНДУЕМЫЕ ИСТОЧНИКИ ИНФОРМАЦИИ

## Рекомендуемые источники информации

<http://google.com>

<http://w3schools.com>

<http://habrahabr.ru>

# СПАСИБО ЗА ВНИМАНИЕ!

## ВОПРОСЫ?

Основы веб-технологий

**Author: Svyatoslav Kulikov**  
**Training And Education Manager**  
**[svyatoslav\\_kulikov@epam.com](mailto:svyatoslav_kulikov@epam.com)**