Архитектура программных систем. Лекция 5
Фронтенд

Михаил Пожидаев

25 апреля 2025 г.

Фронтенд

Основная структура

Фронтенд состоит из трёх основных компонентов:

- браузерной части: HTML + JavaScript;
- балансирующего веб-сервера: NGinx;
- основного веб-сервера: Apache HTTPD, Jakarta Servlet и т. д.

Предлагает две формы представления:

- HTML для веб-страниц;
- RESTful API с XML и JSON для мобильных приложений.

RESTful API

Поддержка мобильных приложений

1. **Использование HTTP методов:** применение стандартных HTTP методов (GET, POST, PUT, DELETE) для обозначения операций над ресурсами.
2. **Ресурсы и идентификаторы:** каждый ресурс должен иметь уникальный идентификатор.
3. **Код состояния:** необходимо использовать стандартные коды состояния HTTP для обозначения результатов операций.
4. **Кэширование:** поддержка кэширования на стороне клиента и сервера, чтобы уменьшить нагрузку на сервер и ускорить обработку запросов.

React.js

Реактивные интерфейсы

React.js — это библиотека JavaScript для создания пользовательских веб-интерфейсов, особенно для динамичных и интерактивных приложений.

1. Преимущества React.js:
   - высокая производительность благодаря виртуальному DOM;
   - широкое сообщество разработчиков, большое количество доступных ресурсов и инструментов;
   - возможность повторного использования компонентов.
2. Основные концепции React.js:
   - компоненты: блоки приложения для повторного использования;
   - виртуальный DOM: представление реального DOM в памяти для оптимизации представления;
   - события и обратные вызовы: используются для взаимодействия между компонентами и обработки

WebSocket

Асинхронный двунаправленный протокол

1. **Двустороннее и полнодуплексное общение:** данные могут передаваться в обоих направлениях одновременно.
2. **Низкие задержки:** WebSocket использует постоянное соединение, что минимизирует задержки при обмене данными.
3. **Уведомление браузера:** возможность сообщить пользователю о событии на стороне сервера.

Подключение к WebSocket

Создание соединения с серверным приложением через W

```
const socket = new WebSocket({''}ws://localhost:8080{'
...
socket.onmessage = (event) => \{
const message = event.data;
console.log({''}Получено сообщение: {''} + message);
\};
socket.onclose = (event) => \{
if (event.wasClean)
console.log({''}Закрытие соединения было чистым.{''});
console.error({''}Закрытие соединения не было чистым.{
\};
socket.send(message);
```

Обратный прокси

Балансировка и отказоустойчивость

1. **Распределение нагрузки:** обратный прокси-сервер может распределять нагрузку между несколькими серверами, предотвращая перегрузку одного сервера.
2. **Отказоустойчивость:** при использовании обратного прокси-сервера можно настроить балансировку так, чтобы в случае сбоя одного из серверов запросы автоматически перенаправлялись на другие доступные серверы.
3. **Оптимизация производительности:** обратный прокси-сервер может кэшировать статические файлы, такие как изображения и CSS/JavaScript-файлы.
4. **Гибкость конфигурации:** обратный прокси-сервер позволяет гибко настраивать балансировку HTTP, что даёт возможность адаптировать систему под конкретные потребности и требования.

Nginx

Распространённый балансирующий сервер

Nginx — это бесплатный, открытый и высокопроизводительный веб-сервер, который также может выполнять функции прокси-сервера, обратного прокси-сервера и балансировщика нагрузки.

1. Способен обрабатывать большое количество одновременных соединений и эффективно управлять нагрузкой на сервер.
2. Совместим с различными операционными системами и серверами приложений. Он легко интегрируется с другими сервисами и может использоваться в различных конфигурациях для оптимизации работы веб-инфраструктуры.
3. Имеет активное сообщество разработчиков и пользователей, которые постоянно работают над улучшением продукта и предоставляют поддержку.

```
stream {
upstream lsocial {
server web1:80 weight=1;
server web2:80 weight=1;
}
server {
listen 80;
}
}
```

Jakarta Servlets

Стандарт разработки веб-приложений

1. **Архитектура.** Jakarta Servlet — это спецификация, которая определяет архитектуру веб-приложений на Java.
2. **Компоненты.** В архитектуре Jakarta Servlet есть несколько ключевых компонентов: сервлеты, фильтры, слушатели и другие.
3. **Жизненный цикл.** Сервлеты имеют определённый жизненный цикл, который включает в себя создание, инициализацию, обработку запросов и уничтожение.
4. **Обработка запросов.** Сервлеты могут обрабатывать различные типы запросов, такие как GET, POST, PUT и DELETE.
5. **Контейнеры.** Контейнеры сервлетов предоставляют среду выполнения для сервлетов и других компонентов.

Спасибо за внимание!

Всё о курсе: https://marigostra.ru/materials/arch.html
E-mail: msp@luwrain.org
Канал в Телеграм: @MarigostraRu