Клиент-Сервер

Часть II

Старков Дима


{ REST }

REST

Прост в понимании

```
DELETE /users/dimastark/posts/1
```

```
Удалить пост №1 пользователя dimastark
```

REST

Прост в понимании

```
DELETE /users/dimastark/posts/1
```

Удалить пост №1 пользователя dimastark

REST

Прост в понимании

```
DELETE /users/dimastark/posts/1
```

Удалить пост №1 пользователя dimastark

REST

Прост в понимании

```
DELETE /users/dimastark/posts/1
```

```
Удалить пост №1 пользователя dimastark
```

REST

Прост в понимании

```
DELETE /users/dimastark/posts/1
```

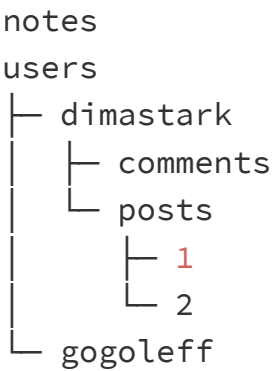
```
Удалить пост №1 пользователя dimastark
```

REST

Прост в понимании

```
DELETE /users/dimastark/posts/1
```

Удалить пост №1 пользователя dimastark



REST

Переиспользует HTTP

```
DELETE /users/dimastark/posts/1
```

REST

Переиспользует HTTP

```
// Семантичные методы
DELETE /users/dimastark/posts/1

// Семантичные коды ответа
200 OK
401 Unauthorized
403 Forbidden
404 Not found
500 Internal Server Error
```

REST

Переиспользует HTTP

Сжатие, кеширование

Балансировка и firewall

CORS, HTTPS

REST

Не спецификация, а набор рекомендаций

Код для ошибок валидации? *400 или 422?*

REST

Не спецификация, а набор рекомендаций

GET идемпотентен?

```
GET /search?text=идемпотентность
```

REST

Сложная фильтрация и поиск

```
GET /users?parameter1=value1&parameter2=value2&parameter3=...

POST /users/search
{
    "parameter1": "value1",
    "parameter2": "value2",
    "parameter3": "value3"
}
```

REST

Сложно мыслить ресурсами

Я создал (POST) себя в университете, обновил (PATCH) знания студентов лекцией и удалил (DELETE) себя из аудитории.

Я пришел в университет, рассказал лекцию студентам и вышел из аудитории.

REST

Нет валидации или схемы данных

Нет механизма версионирования

```
/v1/users/dimastark → /v2/users/dimastark
```

Нет инструментов в комплекте

Пишем клиенты для всех языков


GRPC

gRPC Remote Procedure Calls

gRPC Remote Procedure Calls

Схема в Protobuf формате

```
/* SearchRequest содержит поисковый запрос,
 * а также настройки пагинации */
message SearchRequest {
    string query = 1;
    int32 page_number = 2;
    int32 results_per_page = 3;
}
```

gRPC Remote Procedure Calls

Схема в Protobuf формате

```
/* SearchRequest содержит поисковый запрос,
 * а также настройки пагинации */
message SearchRequest {
    string query = 1;
    int32 page_number = 2;
    int32 results_per_page = 3;
}
```

```
double, float,
int32, int64, uint32, uint64, sint32, sint64,
bool, string, bytes
```

gRPC Remote Procedure Calls

Схема в Protobuf формате

```
/* SearchRequest содержит поисковый запрос,
 * а также настройки пагинации */
message SearchRequest {
    string query = 1;           // Идентификатор поля
    int32 page_number = 2;      // Уникален
    int32 results_per_page = 3; // Не меняется со временем
}
```

gRPC Remote Procedure Calls

Схема в Protobuf формате

```
message SearchReply {
    repeated Document items = 1;
}

message Document {
    map<string, int32> occurrence_counts = 1;
}
```

gRPC Remote Procedure Calls

Схема в Protobuf формате

```
message SearchReply {
    repeated Document items = 1;
}

message Document {
    map<string, int32> occurrence_counts = 1;
}
```

gRPC Remote Procedure Calls

Схема в Protobuf формате

```
message SearchReply {
    repeated Document items = 1;
}

message Document {
    map<string, int32> occurrence_counts = 1;
}
```

gRPC Remote Procedure Calls

Сервисы

```
service Searcher {
    rpc Search (SearchRequest) returns (SearchReply);
}
```

gRPC Remote Procedure Calls

Сервисы

```
service Searcher {
    rpc Search (SearchRequest) returns (SearchReply);
}
```

gRPC Remote Procedure Calls

Сервисы

```
service Searcher {
    rpc Search (SearchRequest) returns (SearchReply);
}
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем серверный и клиентский код
protoc
    --plugin=protoc-gen-grpc=grpc_tools_node_protoc_plugin
    --js_out="import_style=commonjs:./protos"
    --grpc_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем серверный и клиентский код
protoc
    --plugin=protoc-gen-grpc=grpc_tools_node_protoc_plugin
    --js_out="import_style=commonjs:./protos"
    --grpc_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем серверный и клиентский код
protoc
    --plugin=protoc-gen-grpc=grpc_tools_node_protoc_plugin
    --js_out="import_style=commonjs:./protos"
    --grpc_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем серверный и клиентский код
protoc
    --plugin=protoc-gen-grpc=grpc_tools_node_protoc_plugin
    --js_out="import_style=commonjs:./protos"
    --grpc_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем серверный и клиентский код
protoc
    --plugin=protoc-gen-grpc=grpc_tools_node_protoc_plugin
    --js_out="import_style=commonjs:./protos"
    --grpc_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем тайпинги
protoc
    --plugin=protoc-gen-ts=protoc-gen-ts
    --ts_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем тайпинги
protoc
    --plugin=protoc-gen-ts=protoc-gen-ts
    --ts_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Чуть-чуть магии автогенерации

```
# Генерируем тайпинги
protoc
    --plugin=protoc-gen-ts=protoc-gen-ts
    --ts_out=./protos
    search.proto
```

gRPC Remote Procedure Calls

Пишем реализацию методов

```
import grpc from 'grpc';


import { SearchRequest, SearchReply } from './protos/search_pb';
import { ISearcherServer } from './protos/search_grpc_pb';

class SearcherServiceImpl implements ISearcherServer {
    search(
        call: grpc.ServerUnaryCall<SearchRequest>,
        callback: grpc.sendUnaryData<SearchReply>
    ) {
        ...
    }
}
```

gRPC Remote Procedure Calls

Пишем реализацию методов

```
import grpc from 'grpc';


import { SearchRequest, SearchReply } from './protos/search_pb';
import { ISearcherServer } from './protos/search_grpc_pb';

class SearcherServiceImpl implements ISearcherServer {
    search(
        call: grpc.ServerUnaryCall<SearchRequest>,
        callback: grpc.sendUnaryData<SearchReply>
    ) {
        ...
    }
}
```

gRPC Remote Procedure Calls

Пишем реализацию методов

```
import grpc from 'grpc';


import { SearchRequest, SearchReply } from './protos/search_pb';
import { ISearcherServer } from './protos/search_grpc_pb';

class SearcherServiceImpl implements ISearcherServer {
    search(
        call: grpc.ServerUnaryCall<SearchRequest>,
        callback: grpc.sendUnaryData<SearchReply>
    ) {
        ...
    }
}
```

gRPC Remote Procedure Calls

Пишем реализацию методов

```
import grpc from 'grpc';


import { SearchRequest, SearchReply } from './protos/search_pb';
import { ISearcherServer } from './protos/search_grpc_pb';

class SearcherServiceImpl implements ISearcherServer {
    search(
        call: grpc.ServerUnaryCall<SearchRequest>,
        callback: grpc.sendUnaryData<SearchReply>
    ) {
        ...
    }
}
```

gRPC Remote Procedure Calls

Запускаем наш сервер

```
import grpc from 'grpc';


import { SearcherService } from './protos/search_grpc_pb';

const server = new grpc.Server();

server.addService(SearcherService, new SearcherServiceImpl());

server.bind(
    '0.0.0.0:50051',
    grpc.ServerCredentials.createInsecure()
);

server.start();
```

gRPC Remote Procedure Calls

Запускаем наш сервер

```
import grpc from 'grpc';


import { SearcherService } from './protos/search_grpc_pb';

const server = new grpc.Server();

server.addService(SearcherService, new SearcherServiceImpl());

server.bind(
    '0.0.0.0:50051',
    grpc.ServerCredentials.createInsecure()
);

server.start();
```

gRPC Remote Procedure Calls

Запускаем наш сервер

```
import grpc from 'grpc';


import { SearcherService } from './protos/search_grpc_pb';

const server = new grpc.Server();

server.addService(SearcherService, new SearcherServiceImpl());

server.bind(
    '0.0.0.0:50051',
    grpc.ServerCredentials.createInsecure()
);

server.start();
```

gRPC Remote Procedure Calls

Запускаем наш сервер

```
import grpc from 'grpc';


import { SearcherService } from './protos/search_grpc_pb';

const server = new grpc.Server();

server.addService(SearcherService, new SearcherServiceImpl());

server.bind(
    '0.0.0.0:50051',
    grpc.ServerCredentials.createInsecure()
);

server.start();
```

gRPC Remote Procedure Calls

Запускаем наш сервер

```
import grpc from 'grpc';


import { SearcherService } from './protos/search_grpc_pb';

const server = new grpc.Server();

server.addService(SearcherService, new SearcherServiceImpl());

server.bind(
    '0.0.0.0:50051',
    grpc.ServerCredentials.createInsecure()
);

server.start();
```

PROFIT?

gRPC Remote Procedure Calls

Использует HTTP/2

Полностью бинарный

Поточная передача данных

Авторизация, балансировка

gRPC Remote Procedure Calls

Описывает схему в protobuf формате

Валидация входных данных

gRPC Remote Procedure Calls

Описывает схему в protobuf формате

Суперкомпактность ~25%

```
message Person {
    int32  id    = 1;
    string name  = 2;
    string email = 3;
}
```

gRPC Remote Procedure Calls

Описывает схему в protobuf формате

Производительность десериализации

В 10 - 100 порядков быстрее, чем JSON

gRPC Remote Procedure Calls

Генерирует шаблонный код

Серверный, клиентский, документация

Генерирует идеоматичный языку код

gRPC Remote Procedure Calls

Генерирует шаблонный код

gRPC Remote Procedure Calls

Генерирует шаблонный код

Перерыв

Пишем код!

Ссылочки

Бенчмарки

gRPC и Protobuf

Varun Talwar – gRPC Overview

gRPC in 3 minutes

REST это новый SOAP

Штуки

grpc-protoc-ts – генерируем тайпинги

protoc-gen-doc – генерируем документацию

grpc-web – gRPC из браузера

twirp – простая альтернатива gRPC-Web

Вопросы?

Спасибо!