

# RESTful API

## Мобильные Клиенты

Дмитрий Викторович Малыханов  
DataArt

# Почему REST?

```
@Path("/users")
@Produces({ "application/xml", "application/json" })
@RequestScoped
public class UsersRestfulService {
    @PersistenceContext
    private EntityManager em;
    @GET
    public List<User> list() {
        return em.createNamedQuery("User.findAll", User.class).getResultList();
    }
    @GET
    @Path("/{email}")
    public User findByEmail(@PathParam("email") String email) {
        return em.createNamedQuery("User.findByEmail", User.class)
            .setParameter("email", email).getSingleResult();
    }
}
```

## Потому что:

1. Быстро
2. Удобно
3. Понятно
4. Универсально

# Зачем оптимизировать?

Процессор == Заряд Аккумулятора



Объём данных == Деньги

## Ошибка №1

Уже есть замечательный RPC,  
работающий много лет с тысячами  
клиентов?

**Неправильно:**

Просто "обернуть" в JSON over HTTP

# Главный принцип



**Программа**

**=**

**Алгоритм**

**+**

**Структуры данных**



# Что оптимизируем?

Нужно:

Оптимальный код

Делаем:

Оптимизируем алгоритм



Нужно:

Оптимальный **интерфейс**

Делаем:

Оптимизируем **структуры данных**

## Рецепт №0

"Каждый делает своё дело"

Application Server **обрабатывает**  
данные

Web Server **передаёт** данные

## Рецепт №1

"Выжимайте все соки!"

Включите gzip сжатие при отсылке  
данных

**Важно:**

Не пытайтесь сжимать всё подряд  
(правило "256")

## Рецепт №2

"Не откусывайте больше, чем надо!"



Ограничивайте длину списков

## Рецепт №2: Пример

```
[  
  {field1:"value", ... },  
  // 200 записей  
  {field1:"value", ... }  
]
```

Итого:

**50 Kbytes**

```
[  
  {field1:"value", ... },  
  // 20 записей  
  {field1:"value", ... }  
]
```

Итого:

**5 Kbytes**

## Рецепт №3

Избавьтесь от лишнего "веса"!



Только те поля, которые реально  
используются на клиенте.

# Рецепт №3: Пример



**Joe Doe**

Engineering

```
{  
  first:"Joe",last:"Doe",  
  age:45, employer:"ACME",  
  department:25,  
  address:{country:"UK",  
    city:"Bristol",  
    street:"Main",  
    building:12},  
  photo:1945678,sex:'M'  
}
```

```
{  
  first:"Joe",last:"Doe",  
  department:25,  
  photo:1945678  
}
```

**на 70..90% меньше!**

## Рецепт №4

Соблюдайте правила движения!



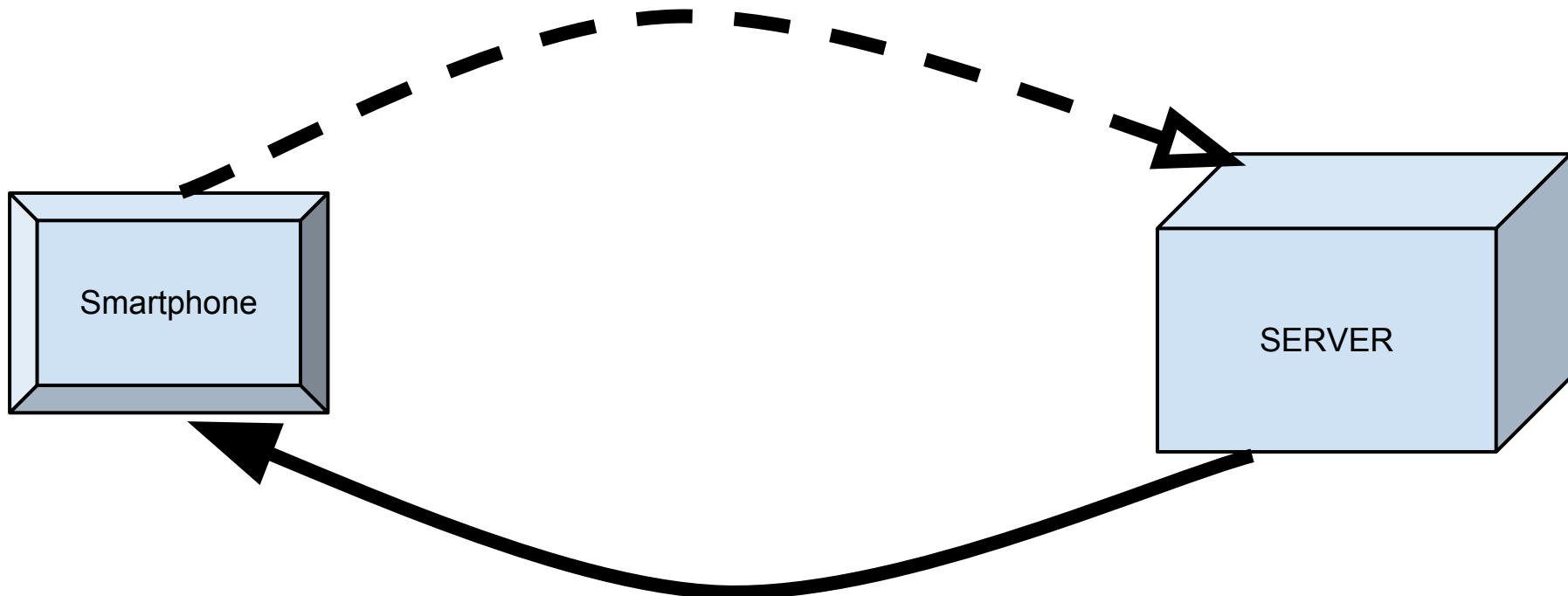
Используйте все возможности НТТР

# Рецепт №4: Пример 1

GET /rest/resource/path HTTP/1.1

Host: server.example.com

**If-Modified-Since: Sat, 1 Oct 2011 00:00 GMT**

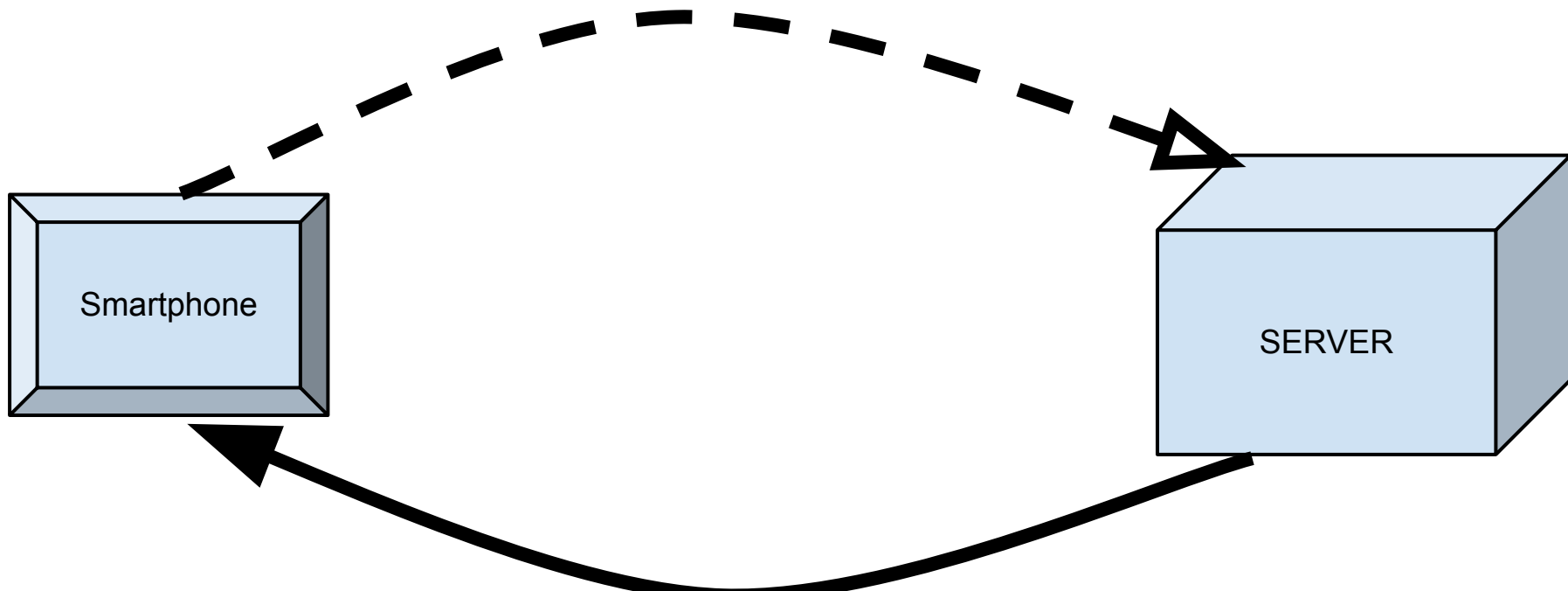


**HTTP/1.1 304 Not Modified**

## Рецепт №4: Пример 2

GET /rest/resource/path HTTP/1.1

Host: server.example.com



HTTP/1.1 200 OK

**Expires: Sat, 29 Oct 2011 06:00 GMT**

# Рецепт №5: XML

## "KISS"

Разбор потока XML обычно проще  
разбора потока JSON

Важно:

"Избыточность" XML легко  
устраняется

# Рецепт Последний

Универсального решения **НЕТ**



Всегда **АНАЛИЗИРУЙТЕ** требования  
и применяйте **КОМПЛЕКС** мер

# RESTful API

## Мобильные Клиенты

Вопросы?

Дмитрий Викторович Малыханов  
DataArt



**DATAART**<sup>®</sup>  
*Enjoy IT*