REST

- *Create*—POST
- *Read*—GET
- *Update*—PUT or PATCH
- *Delete*—DELETE

ContentNegotiatingViewResolver

- ContentNegotiatingViewResolver - это ViewResolver который делегирует выбор представления другим резольверам, затем он формирует список всех доступных View и выбирает один из них исходя из запрашиваемого типа содержимого:
    1. Если имя ресурса заканчивается на .html .xml .json то выбирается соотв. View.
    2. View выбирается на основе заголовка Accept.
- ContentNegotiatingViewResolver конфигурируется с помощью ContentNegatiationManager.
- Можно например сконфигурировать ContentNegotiatingViewResolver так, чтобы по умолчанию он выбирал text/html и т.о. обслуживать веб страницы и Rest-сервисы.

- ContentNegotiatingViewResolver отобразит возвращаемые данные как будто они добавлены в модель, т.е.

```
List<Spittle> ->
{
    "spittleList": [
        {
            "id": 0,
            "message": "Some message 0",
            "date": 1488027200391
        }
      ]
}
```

MessageConverters

- Можно изменить сериализацию ответа рест-метода и не использовать для этого View, а использовать MessageConverters напрямую. Для этого нужно можно аннотировать:

```
@ResponseBody //аннотируем возвращаемое значение
@RequestBody //аннотируем тело запроса
@RestController //аннотируем весь класс, тогда
//аннотации @Response/RequestBody не нужны
```

- ResponseEntity позволяет вернуть данные клиенту и указать доп. параметры ответа - код состояния и заголовки. Можно использовать для обработки ошибок и указания url вновь созданного ресурса.

Обработка ошибок

- Можно выбросить исключение в ресурс методе и перехватить его в методе обрабатывающем ошибку, для этого метод-перехватчик аннотируется через:

```
@ExceptionHandler(SpittleNotFoundException.class)
```

Rest-клиент

- RestTemplate предназначен для посылки рест-запросов:

```
 <T> ResponseEntity<T> exchange(String url, HttpMethod method,
HttpEntity<?> requestEntity, Class<T> responseType,
Object... uriVariables) throws RestClientException;
```

- Response/RequestEntity содержат тело ответа/запроса и http-заголовки.
- uriVariables подставляются вместо соотв. plzctholderов в строке url.