20.1. Evoluindo a API com gestão de mudanças

quarta-feira, 19 de abril de 2023 21:10

Retrocompatibilidade vs Quebra de compatibilidade

20.2. Evitando quebrar os clientes: nova propriedade no modelo

quarta-feira, 19 de abril de 2023 21:23

Modelo de representação (saída)

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

Exemplo: adicionar uma mudança para avaliação de restaurantes

Modelo de representação (saída)

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true,
    "nota": 98
}
```

✅ Retrocompatibilidade

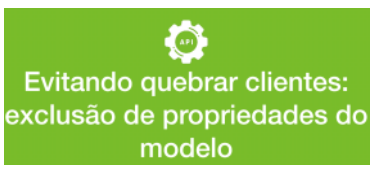
Os consumidores da api irão ignorar a propriedade nova por desconhecer e eventualmente não quebrar a api pela adição da mudança, no caso, é uma Retrocompatibilidade, uma mudança que é possível novos e antigos clientes terem compatibilidade com a versão atual.

Ou adicionar uma propriedade opcional

Modelo de representação (entrada)

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    },
    "dataFundacao": "1981-01-01"
}
```

❌ Quebra compatibilidade ✅ Retrocompatibilidade



Tanto na representação do recurso de entrada e saída da requisição

Modelo de representação (saída)

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

Modelo de representação (saída)

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

❌ Quebra compatibilidade

Modelo de representação (entrada)

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    }
}
```

❌ Quebra compatibilidade ✅ Retrocompatibilidade

20.03. Evitando ...

Modelo de representação (saída)

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

Modelo de representação (saída)

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

❌ Quebra compatibilidade

Modelo de representação (entrada)

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    }
}
```

Modelo de representação (entrada)

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    }
}
```

❌ Quebra compatibilidade ✅ Retrocompatibilidade

RestauranteInput ⌄ {

cozinha*	CozinhaIdInput › {...}
endereco*	EnderecoInput › {...}
nome*	string *example: Thai Gourmet*
taxaFrete*	number *example: 12*

}

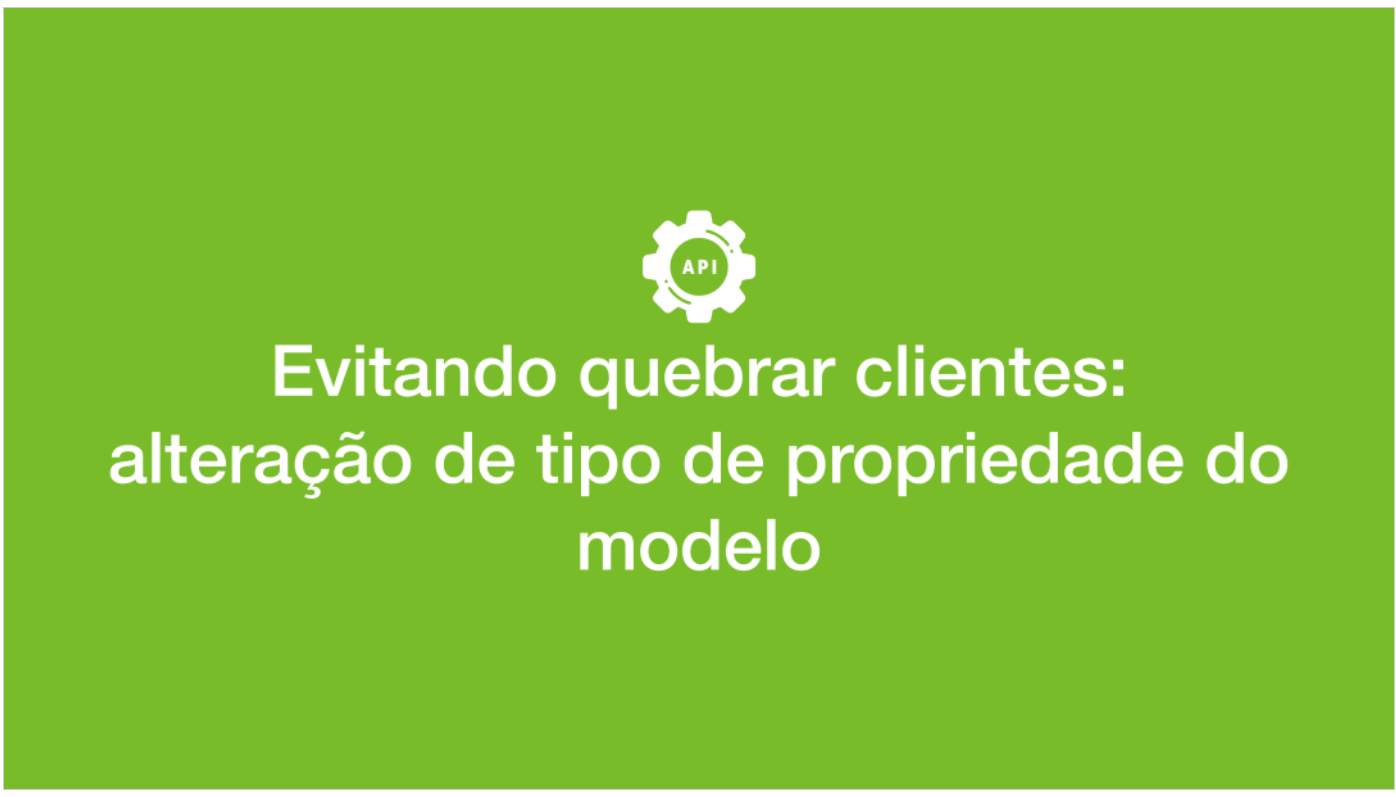
RestauranteInput ⌄ {

cozinha*	CozinhaIdInput › {...}
endereco*	EnderecoInput › {...}
nome*	string *example: Thai Gourmet*

}

```
public class RestauranteInput {

    @ApiModelProperty(hidden = true)
//  @NotNull
//  @PositiveOrZero
    private BigDecimal taxaFrete;
```

20.04. Evitando ...



Modelo de representação de saída

Alteração de um tipo amplo para específico

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": "10.00",
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

✅ Retrocompatibilidade

Alteração de um tipo específico para amplo

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": "10.00",
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

❌ Quebra compatibilidade

Uma solução: adicione uma nova propriedade

```
{
    "id": 1,
    "nome": "Thai Gourmet”,
    "taxaFrete": 10.00,
    "taxaFreteTexto": "10.00",
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

Uma solução: adicione uma nova propriedade

RestauranteModel ⌄ {

_links	Links › {...}
aberto	boolean
ativo	boolean
cozinha	CozinhaModel › {...}
endereco	EnderecoModel › {...}
id	integer($int64) *example: 1*
nome	string *example: Thai Gourmet*
taxaFrete	number *example: 12* Depreciada. Use propriedade taxaFreteTexto
taxaFreteTexto	string *example: 12.00*

}

Uma solução: adicione uma nova propriedade

```
public class RestauranteModel {

    @ApiModelProperty(example = "12.00",
            value = "Depreciada. Use propriedade taxaFreteTexto")
    private BigDecimal taxaFrete;

    @ApiModelProperty(example = "'12.00'")
    private String taxaFreteTexto;
```

Modelo de representação de entrada

Alteração de um tipo amplo para específico

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": "10.00",
    "cozinha": {
        "id": 1
    }
}
```

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    }
}
```

❌ Quebra compatibilidade

Alteração de um tipo específico para amplo

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    }
}
```

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": "10.00",
    "cozinha": {
        "id": 1
    }
}
```

✅ Retrocompatibilidade

Evitando quebrar clientes: alteração na estrutura de dados do modelo

Modelo de representação (saída)

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinhaId": 1,
    "cozinhaNome": "Tailandesa",
    "ativo": true,
    "aberto": true
}
```

❌ Quebra compatibilidade

Solução: retorne as duas coisas

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "ativo": true,
    "aberto": true
}
```

```
{
    "id": 1,
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1,
        "nome": "Tailandesa"
    },
    "cozinhaId": 1,
    "cozinhaNome": "Tailandesa",
    "ativo": true,
    "aberto": true
}
```

✅ Retrocompatibilidade

Modelo de representação (entrada)

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    }
}
```

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinhaId": 1
}
```

❌ Quebra compatibilidade

Solução: permita as duas coisas

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinha": {
        "id": 1
    }
}
```

```
{
    "nome": "Thai Gourmet",
    "taxaFrete": 10.00,
    "cozinhaId": 1
}
```

✅ Retrocompatibilidade

20.6. Evitando quebrar os clientes: alteração de URL de recurso

quinta-feira, 20 de abril de 2023 07:51

Podemos adicionar duas urls em um mesmo endpoint quando há necessidade de alterar a URL do recurso, mantendo a compatibilidade dos novos e antigos consumidores da api.

Quando usamos HATEOAS, é necessário alterar também o link do Root Entry Point para a nova URL. Uma alternativa é manter o antigo e o novo recurso apontando para a mesma URL.

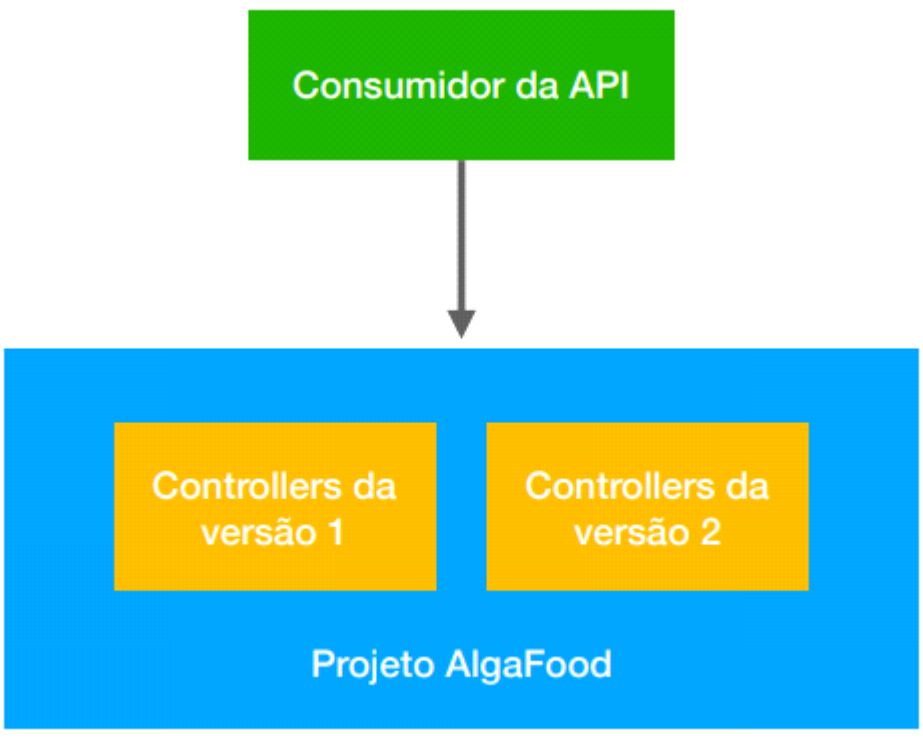
20.7. O que é e quando versionar uma API?

quinta-feira, 20 de abril de 2023 13:35

- Evitar quebrar compatibilidade
- Quando precisar quebrar compatibilidade, versione a API
- Utilizar versões em apis
    - Rodar simultaneamente a versão antiga e a nova
- Evitar versionamento de API

20.8. As principais técnicas de versionamento de APIs

quinta-feira, 20 de abril de 2023 13:42

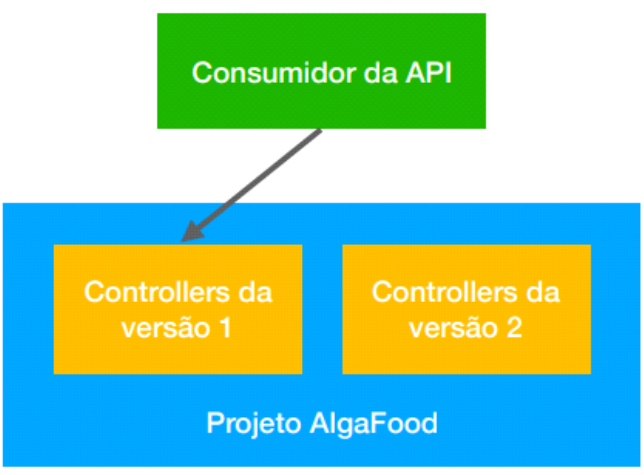


É como uma nova aplicação backend rodando. Podemos rodar duas aplicações simultaneamente ou uma aplicação com controladores diferentes para cada versão

Como controlar o acesso a versões diferentes em uma mesma aplicação?

- Versionamento por Media Type

Versionamento por Media Type



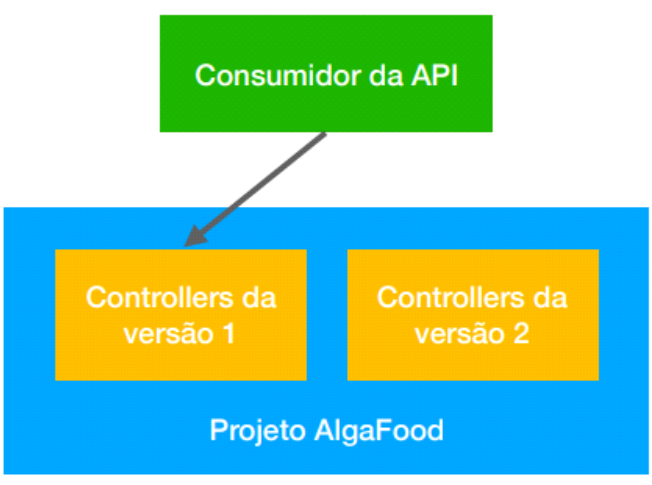
REQUISIÇÃO

```
GET /cozinhas
Accept: application/vnd.algafood.v1+json
```

Media Types customizados para atender aos consumidores da api para quais endpoints das versões querem consumir. O header Accept com o valor application/vnd.algafood.v1+json

o nvd é um prefixo para padrão definido para customização de media types.

Versionamento por URI



REQUISIÇÃO

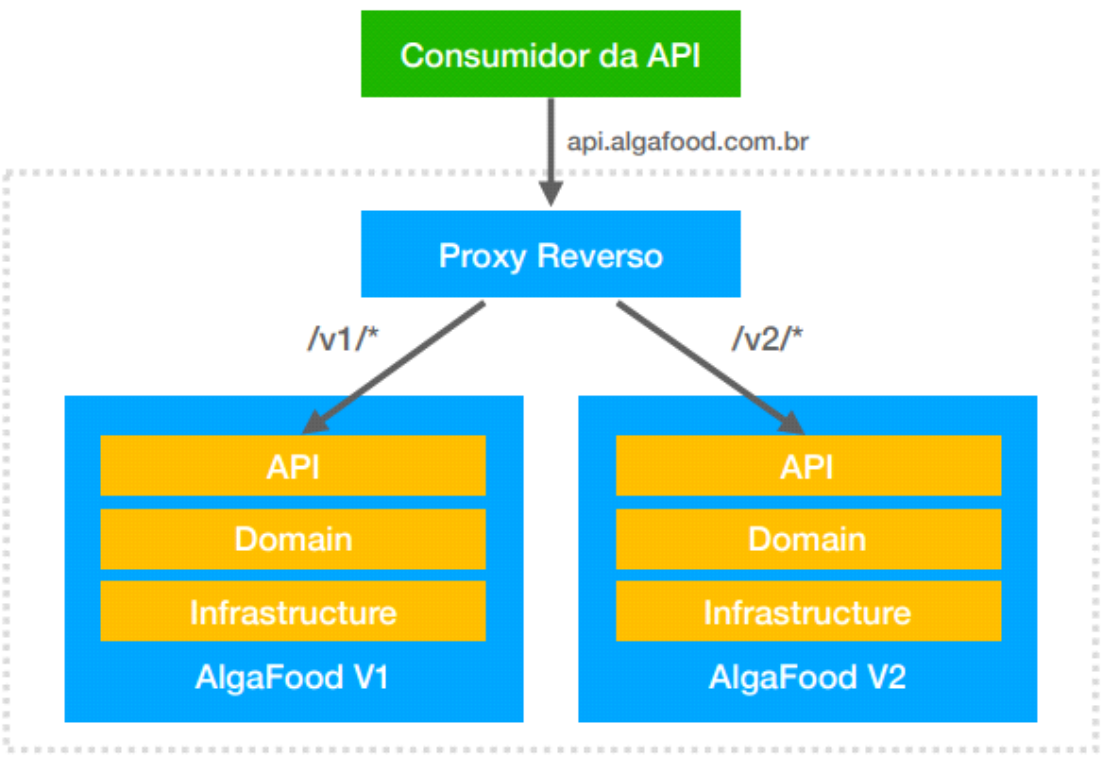
```
GET /v1/cozinhas
Accept: application/json
```

20.9. As principais abordagens para manter a base de código de APIs versionadas

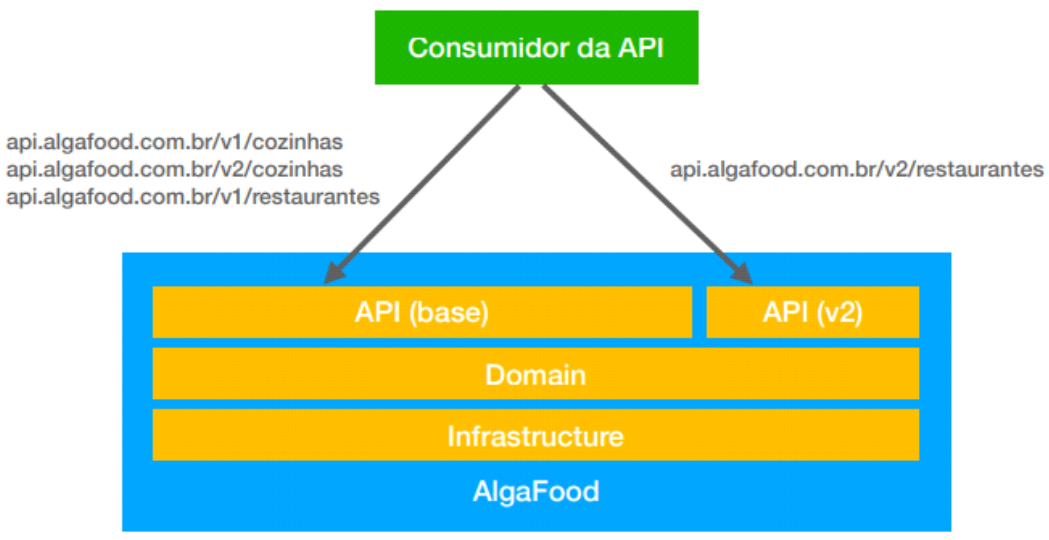
segunda-feira, 24 de abril de 2023 17:58

Durante o desenvolvimento de outra versão da API, as duas versões terão que estar disponíveis, nessa aula veremos abordagens para a separação de cada versão.

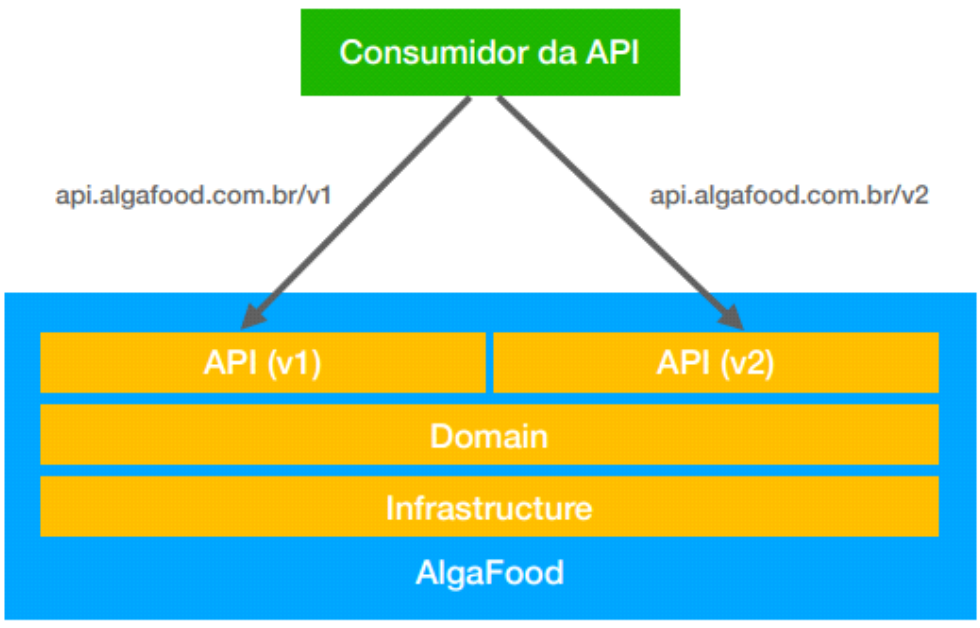
- Projeto separado para cada versão



- Mesmo projeto com reaproveitamento da camada de API



- Mesmo projeto com separação total da camada da API



20.10. Preparando o projeto para versionamento da API por Media Type

segunda-feira, 24 de abril de 2023 18:12

- Adotamos a estratégia com o mesmo projeto reaproveitamento a camada da API versionando para V1 e V2
- Versionar apenas o recurso de cidades

- Alterar o Media Type aceito pela URI

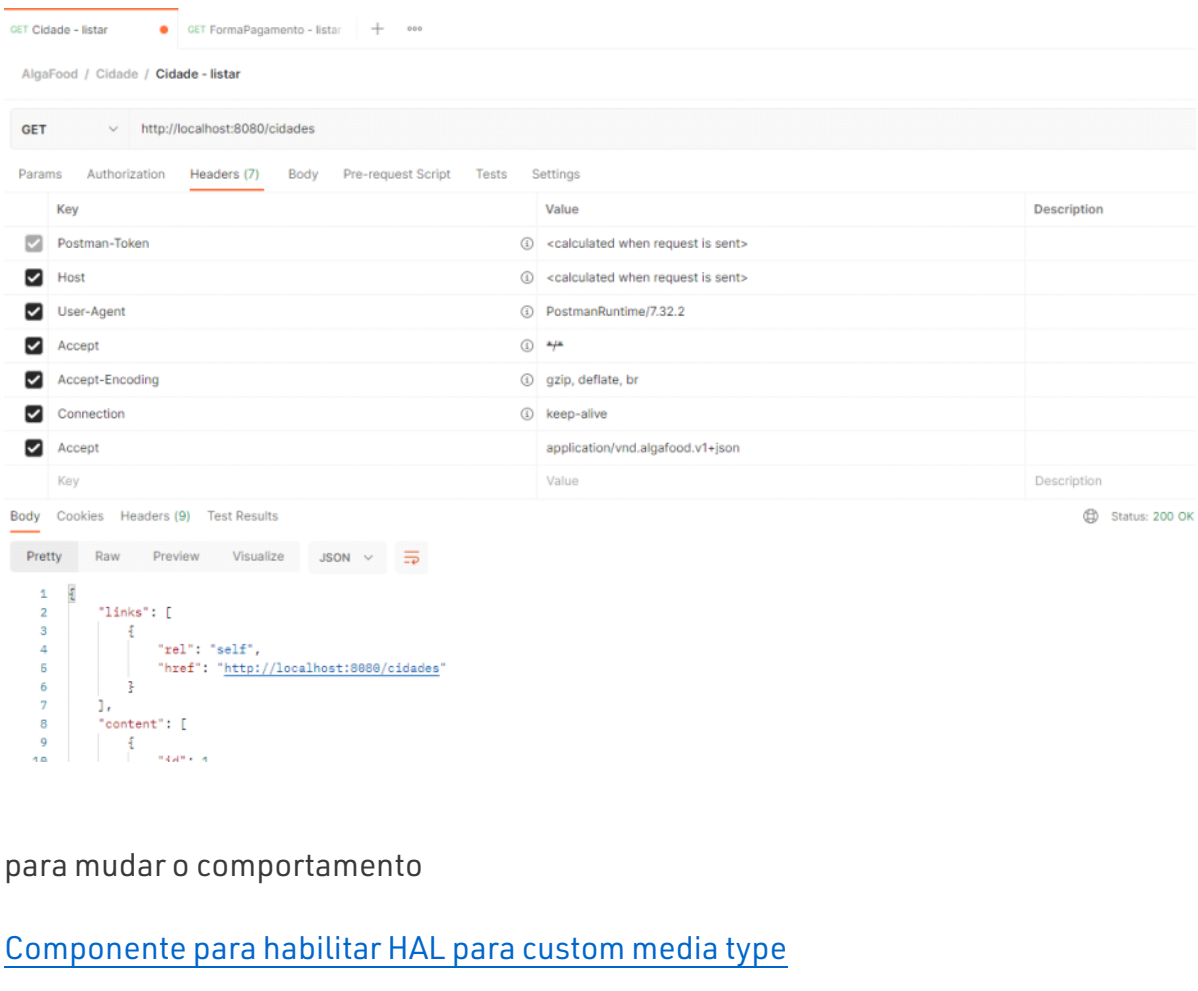
```
4 usages
@Api(tags = "Cidades")
@RestController
@RequestMapping(path = "/cidades", produces = MediaType.APPLICATION_JSON_VALUE)
public class CidadeController implements CidadeControllerOpenApi {
```

para

```
4 usages
@Api(tags = "Cidades")
@RestController
@RequestMapping(path = "/cidades", produces = "application/vnd.algafood.v1+json")
public class CidadeController implements CidadeControllerOpenApi {
```

a ideia é separar as versões do recurso ou toda a camada de controller das duas versões.

Ao adicionar um mediatype customizado, a resposta serializada não vem mais no formato HAL



para mudar o comportamento

- Componente para habilitar HAL para custom media type

  https://gist.github.com/thiagofa/8952d7fee7650c94bb116917d63c9ae6

```
package com.algaworks.algafood.core.web;

import ...

// Fonte: https://github.com/spring-projects/spring-hateoas/issues/263#issuecomment-358969098
@Component
public class HalCustomMediaTypeEnabler {

    2 usages
    private final RequestMappingHandlerAdapter requestMappingHandlerAdapter;

    @Autowired
    public HalCustomMediaTypeEnabler(RequestMappingHandlerAdapter requestMappingHandlerAdapter) {
        this.requestMappingHandlerAdapter = requestMappingHandlerAdapter;
    }

    @PostConstruct
    public void enableVndHalJson() {
        for (HttpMessageConverter<?> converter : requestMappingHandlerAdapter.getMessageConverters()) {
            if (converter instanceof MappingJackson2HttpMessageConverter
                    && converter.getSupportedMediaTypes().contains(MediaTypes.HAL_JSON)) {

                MappingJackson2HttpMessageConverter messageConverter = (MappingJackson2HttpMessageConverter) converter;
                messageConverter.setSupportedMediaTypes(Arrays.asList(MediaTypes.HAL_JSON,
                        MediaType.valueOf("application/vnd.algafood.v1+json")));
            }
        }
    }
}
```

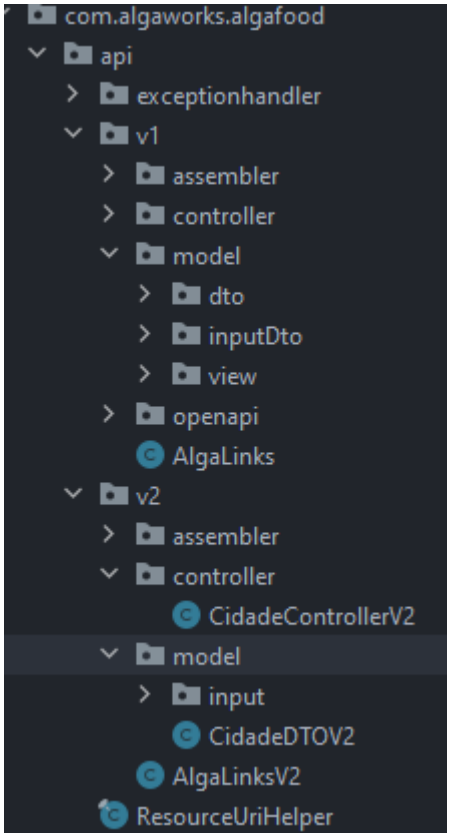
- Criar uma constante para utilizar o Media Type customizado

```
3 usages
public class AlgaMediaTypes {

    1 usage
    public static final String V1_APPLICATION_JSON_VALUE = "application/vnd.algafood.v1+json";

    1 usage
    public static final MediaType V1_APPLICATION_JSON = MediaType.valueOf("application/vnd.algafood.v1+json");
}
```

```
4 usages
@Api(tags = "Cidades")
@RestController
@RequestMapping(path = "/cidades", produces = AlgaMediaTypes.V1_APPLICATION_JSON_VALUE)
public class CidadeController implements CidadeControllerOpenApi {

    6 usages
```

20.11. Implementando o versionamento da API por Media Type

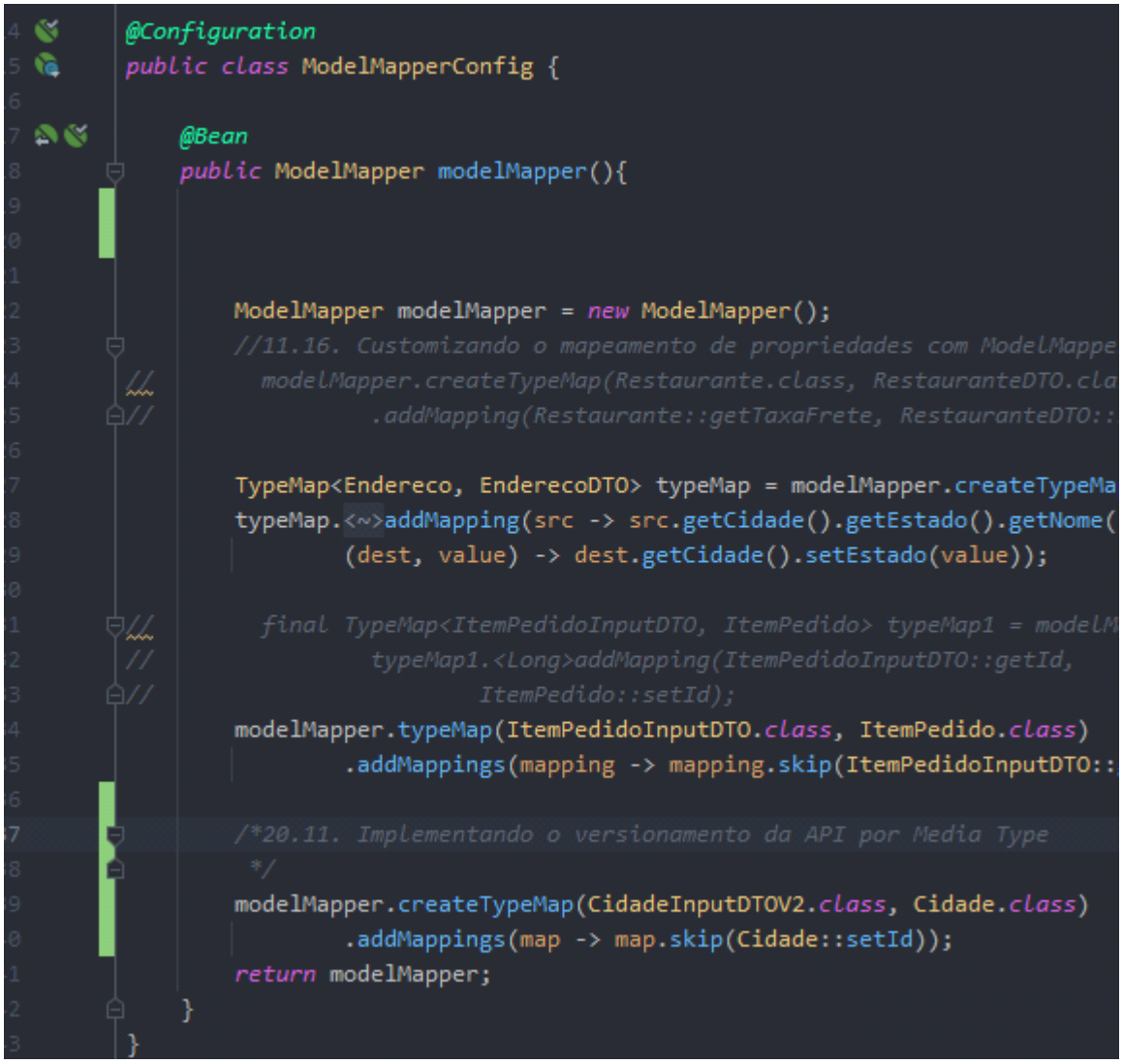
segunda-feira, 24 de abril de 2023 21:56



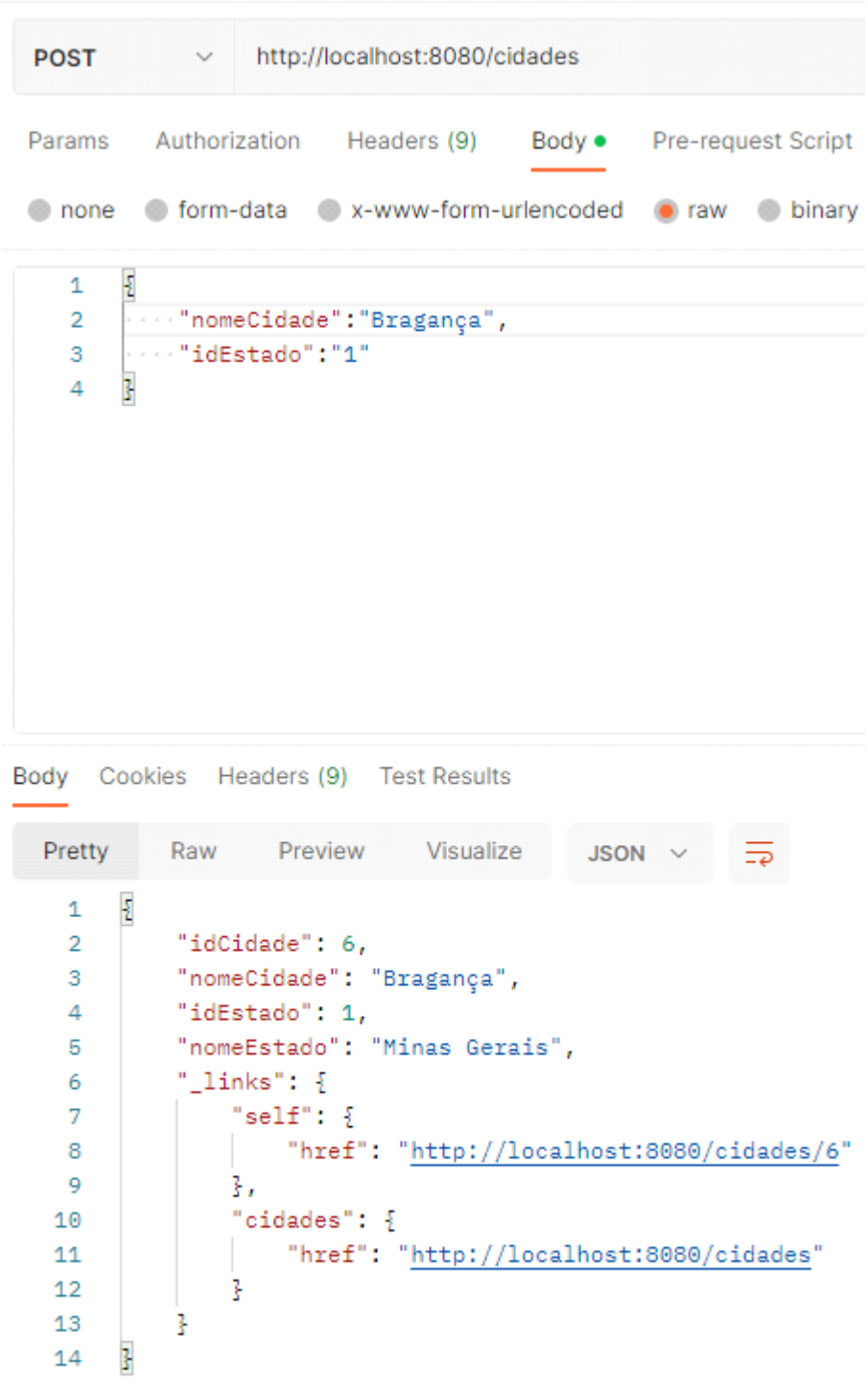
Criamos a camada controladores responsável pela entidade Cidade. A diferença é apenas no DTO de cidade

```java
package com.algaworks.algafood.api.v2.model.input;

import ...

//@ApiModel(value = "Cidade input", description = "Representa uma cidade")
6 usages
@Getter
@Setter
public class CidadeInputDTOV2 {

    @ApiModelProperty(example = "Uberlândia", required = true   )
    @NotBlank(message = "Não é permitido um campo vazio")
    private String nomeCidade;

    @ApiModelProperty(example = "1", required = true)
    @Valid
    @NotNull
    private Long idEstado;
}
```

Por conta do modelMapper, ele atribui o ID do estado ao ID da entidade que está sendo persistida, e o hibernate acaba atualizando a entidade de id 1



```java
@Configuration
public class ModelMapperConfig {

    @Bean
    public ModelMapper modelMapper(){



        ModelMapper modelMapper = new ModelMapper();
        //11.16. Customizando o mapeamento de propriedades com ModelMappe
        //  modelMapper.createTypeMap(Restaurante.class, RestauranteDTO.cla
        //          .addMapping(Restaurante::getTaxaFrete, RestauranteDTO::

        TypeMap<Endereco, EnderecoDTO> typeMap = modelMapper.createTypeMa
        typeMap.addMapping(src -> src.getCidade().getEstado().getNome(
                (dest, value) -> dest.getCidade().setEstado(value));

        //  final TypeMap<ItemPedidoInputDTO, ItemPedido> typeMap1 = modelM
        //          typeMap1.<Long>addMapping(ItemPedidoInputDTO::getId,
        //                  ItemPedido::setId);
        modelMapper.typeMap(ItemPedidoInputDTO.class, ItemPedido.class)
                .addMappings(mapping -> mapping.skip(ItemPedidoInputDTO::

        /*20.11. Implementando o versionamento da API por Media Type
         */
        modelMapper.createTypeMap(CidadeInputDTOV2.class, Cidade.class)
                .addMappings(map -> map.skip(Cidade::setId));
        return modelMapper;
    }
}
```


POST
http://localhost:8080/cidades
Params
Authorization
Headers (9)
Body
Pre-request Script
none
form-data
x-www-form-urlencoded
raw
binary
{
"nomeCidade":"Bragança",
"idEstado":"1"
}
Body
Cookies
Headers (9)
Test Results
Pretty
Raw
Preview
Visualize
JSON
{
"idCidade": 6,
"nomeCidade": "Bragança",
"idEstado": 1,
"nomeEstado": "Minas Gerais",
"_links": {
"self": {
"href": "http://localhost:8080/cidades/6"
},
"cidades": {
"href": "http://localhost:8080/cidades"
}
}
}

20.12. Definindo a versão padrão da API quando o Media Type não é informado

quinta-feira, 27 de abril de 2023 18:10

```java
package com.algaworks.algafood.core.web;

import ...

@Configuration
public class WebConfig implements WebMvcConfigurer {

    @Override
    public void addCorsMappings(CorsRegistry registry) {
        registry.addMapping( pathPattern: "/**") //todos os endpoints
                .allowedMethods("*"); //todos os métodos http
    }

    3 usages
    @Override
    public void configureContentNegotiation(ContentNegotiationConfigurer configurer) {
        configurer.defaultContentType(AlgaMediaTypes.V2_APPLICATION_JSON);
    }

    @Bean
    public Filter shallowEtagHeaderFilter() { return new ShallowEtagHeaderFilter(); }
}
```

Ao fazer a requisição, o content type por padrão, é da versão 2

20.13. Implementando o versionamento da API por URI

quinta-feira, 27 de abril de 2023 18:19

Alterar o caminho da URI para indicar qual recurso está na versão 1 e 2 para "/v1/cidades" e retirar o content negotiation

20.14. Desafio: Refatorando controladores para adicionar /v1 nas URIs

sexta-feira, 28 de abril de 2023 10:21

Esse desafio será bem simples, iremos apenas adicionar o prefixo "/v1" na anotação @RequestMapping de nossos Controllers.

Ficará mais ou menos assim:

@RequestMapping(path = "/v1/<O path que já existia>")

20.15. Desafio: adicionando os recursos de cozinhas na v2 da API

sexta-feira, 28 de abril de 2023 23:55

20.16. Gerando documentação das versões da API com SpringFox e Swagger UI

sexta-feira, 28 de abril de 2023 23:56

20.17. Desafio: revisando documentação da v2 da API

sexta-feira, 28 de abril de 2023 23:56

20.18. Depreciando uma versão da API

sexta-feira, 28 de abril de 2023 18:03

```java
@Deprecated
@Override
@GetMapping
public CollectionModel<CidadeDTO> listar() {
    final List<Cidade> cidades = cidadeService.listar();

    return cAssembler.toCollectionModel(cidades);
}
```

AlgaFood API 1 OAS3

http://localhost:8080/v3/api-docs?group=V1

API aberta para clientes e restaurantes.
Essa versão da API está depreciada e deixará de existir a partir de 01/01/2021. Use a versão mais atual da API.

Wenderson Gustavo - Website
Send email to Wenderson Gustavo

Servers
http://localhost:8080 - Inferred Url

Cidades Gerencia as cidades

GET ~~/v1/cidades~~ Lista as cidades

```java
package com.algaworks.algafood.core.web;

import org.springframework.stereotype.Component;
import org.springframework.web.servlet.HandlerInterceptor;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

1 usage
@Component
public class ApiDeprecationHandler implements HandlerInterceptor {

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {

        if (request.getRequestURI().startsWith("/v1/")) {
            response.addHeader( name: "X-AlgaFood-Deprecated",
                    value: "Essa versão da API está depreciada e deixará de existir a partir de 01/01/2021."
                            + "Use a versão mais atual da API.");
        }

        return true;
    }
}
```

Vary	ⓘ	Access-Control-Request-Headers
X-AlgaFood-Deprecated	ⓘ	Essa versão da API está depreciada e deixará de existir a partir de 01/01/2021.Use a versão mais atua
ETag	ⓘ	"09cf850c8b080a43af30809e14bc68e7c"

20.19. Desligando uma versão da API

sexta-feira, 28 de abril de 2023 19:01

```java
@Component
public class ApiRentiramentHandler implements HandlerInterceptor { //ApiDepracetationHandler

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {

        if (request.getRequestURI().startsWith("/v1/")) {
            response.setStatus(HttpStatus.GONE.value());
            return false;
        }

        return true;
    }
}
```

```java
@Configuration
@Import(BeanValidatorPluginsConfiguration.class)
public class SpringFoxConfig {

    3 usages
    TypeResolver typeResolver = new TypeResolver();

//    @Bean
//    public Docket apiDocketV1() {
//
//        return new Docket(DocumentationType.OAS_30)
//                .groupName("V1")
//                .ignoredParameterTypes(ignoredParameterTypes())
//                .select()
//                //.apis(RequestHandlerSelectors.any())/*Todos os endpointds*/
//                .apis(RequestHandlerSelectors.basePackage("com.algaworks.algafood.api"))
//                .paths(PathSelectors.ant("/v1/**"))/*Caminho padrão*/
////                  .paths(PathSelectors.ant("/restaurantes/*"))
//                .build()
//                .apiInfo(apiInfoV1())
//                .tags(tags()[0], tags())
//                .useDefaultResponseMessages(false)
//                .globalResponses(HttpMethod.GET, globalGetResponses())
//                .globalResponses(HttpMethod.POST, globalPutResponses())
//                .globalResponses(HttpMethod.PUT, globalPutResponses())
//                .globalResponses(HttpMethod.DELETE, globalDeleteResponses())
```

Necessário também retirar a documentação da api desligada