18.22. Resolução do desafio

1 - Alterando SpringFoxConfig

Primeiro, vamos alterar o método apiDocket, adicionando novas Tags

```
.tags(new Tag("Cidades", "Gerencia as cidades"),
        new Tag("Grupos", "Gerencia os grupos de usuários"),
        new Tag("Cozinhas", "Gerencia as cozinhas"));
```

2 - Alterando CozinhaInput

Vamos adicionar a anotação @ApiModelProperty acima da propriedade "nome"

```
@ApiModelProperty(example = "Brasileira", required = true)
@NotBlank
private String nome;
```

3 - Alterando CozinhaModel

Agora, vamos adicionar a mesma anotação nas propriedades de nosso modelo

```
@ApiModelProperty(example = "1")
@JsonView(RestauranteView.Resumo.class)
private Long id;

@ApiModelProperty(example = "Brasileira")
@JsonView(RestauranteView.Resumo.class)
private String nome;
```

4 - Criando CozinhaControllerOpenApi

```
@Api(tags = "Cozinhas")
public interface CozinhaControllerOpenApi {

    @ApiOperation("Lista as cozinhas com paginação")
    public Page<CozinhaModel> listar(Pageable pageable);

    @ApiOperation("Busca uma cozinha por ID")
    @ApiResponses({
          @ApiResponse(code = 400, message = "ID da cozinha inválido", response =
Problem.class),
          @ApiResponse(code = 404, message = "Cozinha não encontrada", response =
Problem.class)
    })
    public CozinhaModel buscar(
            @ApiParam(value = "ID de uma cozinha", example = "1")
            Long cozinhaId);

    @ApiOperation("Cadastra uma cozinha")
    @ApiResponses({
        @ApiResponse(code = 201, message = "Cozinha cadastrada"),
    })
    public CozinhaModel adicionar(
            @ApiParam(name = "corpo", value = "Representação de uma nova cozinha")
            CozinhaInput cozinhaInput);

    @ApiOperation("Atualiza uma cozinha por ID")
```

```
    @ApiResponses({
        @ApiResponse(code = 200, message = "Cozinha atualizada"),
          @ApiResponse(code = 404, message = "Cozinha não encontrada", response =
Problem.class)
    })
    public CozinhaModel atualizar(
            @ApiParam(value = "ID de uma cozinha", example = "1")
            Long cozinhaId,

             @ApiParam(name = "corpo", value = "Representação de uma cozinha com os
novos dados")
            CozinhaInput cozinhaInput);

    @ApiOperation("Exclui uma cozinha por ID")
    @ApiResponses({
        @ApiResponse(code = 204, message = "Cozinha excluída"),
          @ApiResponse(code = 404, message = "Cozinha não encontrada", response =
Problem.class)
    })
    public void remover(
            @ApiParam(value = "ID de uma cozinha", example = "1")
            Long cozinhaId);
}
```

5 - Alterando CozinhaController

Vamos alterar CozinhaController para que o mesmo implemente a interface CozinhaControllerOpenApi

```
@RestController
@RequestMapping(value = "/cozinhas", produces = MediaType.APPLICATION_JSON_VALUE)
public class CozinhaController implements CozinhaControllerOpenApi {
    //...
}
```

Atualizando para o SpringFox 3.0 e Open API 3

Este documento irá te auxiliar a fazer esta aula com a versão 3.0.0 do Spring Fox e suas dependências.

Observação: Este documento é apenas para os que querem usar as versões mais recentes das dependências contidas nesta aula.

Evitando um NullPointerException

Ao adicionar o `produces = MediaType.APPLICATION_JSON_VALUE` na Annotation `@RequestMapping` nos Controllers de Grupo ou Cidade, o SpringFox 3 nos apresentará um NullPointerException. Isso é um problema conhecido que ainda não foi corrigido nessa biblioteca. Acontece devido a alguns métodos das Controllers terem o retorno `void`, como os de `DELETE`, que somado ao `produces = MediaType.APPLICATION_JSON_VALUE` gera um NullPointerException.

Sendo assim, é necessário adicionar o `produces` em cada um dos métodos, com exceção daqueles que retornam `void`, ao invés de adicioná-los na Controller.

Deixando as implementações da seguinte forma:

```java
package com.algaworks.algafood.api.controller;

import com.algaworks.algafood.api.assembler.CozinhaInputDisassembler;
import com.algaworks.algafood.api.assembler.CozinhaModelAssembler;
import com.algaworks.algafood.api.model.CozinhaModel;
import com.algaworks.algafood.api.model.input.CozinhaInput;
import com.algaworks.algafood.api.openapi.controller.CozinhaControllerOpenApi;
import com.algaworks.algafood.domain.model.Cozinha;
import com.algaworks.algafood.domain.repository.CozinhaRepository;
import com.algaworks.algafood.domain.service.CadastroCozinhaService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.data.web.PageableDefault;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseStatus;
import org.springframework.web.bind.annotation.RestController;

import javax.validation.Valid;
import java.util.List;

@RestController
@RequestMapping(value = "/cozinhas")
public class CozinhaController implements CozinhaControllerOpenApi {

	@Autowired
	private CozinhaRepository cozinhaRepository;

	@Autowired
	private CadastroCozinhaService cadastroCozinha;

	@Autowired
	private CozinhaModelAssembler cozinhaModelAssembler;

	@Autowired
	private CozinhaInputDisassembler cozinhaInputDisassembler;

	@GetMapping(produces = MediaType.APPLICATION_JSON_VALUE)
	public Page<CozinhaModel> listar(@PageableDefault(size = 10) Pageable
pageable) {
		Page<Cozinha> cozinhasPage = cozinhaRepository.findAll(pageable);

		List<CozinhaModel> cozinhasModel = cozinhaModelAssembler
				.toCollectionModel(cozinhasPage.getContent());

		Page<CozinhaModel> cozinhasModelPage = new
PageImpl<>(cozinhasModel, pageable,
				cozinhasPage.getTotalElements());

		return cozinhasModelPage;
	}

	@GetMapping(value = "/{cozinhaId}", produces =
MediaType.APPLICATION_JSON_VALUE)
	public CozinhaModel buscar(@PathVariable Long cozinhaId) {
		Cozinha cozinha = cadastroCozinha.buscarOuFalhar(cozinhaId);
```

```
                return cozinhaModelAssembler.toModel(cozinha);
        }

        @PostMapping(produces = MediaType.APPLICATION_JSON_VALUE)
        @ResponseStatus(HttpStatus.CREATED)
                public CozinhaModel adicionar(@RequestBody @Valid CozinhaInput
cozinhaInput) {
                                                        Cozinha cozinha =
cozinhaInputDisassembler.toDomainObject(cozinhaInput);
                cozinha = cadastroCozinha.salvar(cozinha);

                return cozinhaModelAssembler.toModel(cozinha);
        }

                            @PutMapping(value = "/{cozinhaId}", produces =
MediaType.APPLICATION_JSON_VALUE)
        public CozinhaModel atualizar(@PathVariable Long cozinhaId,
                                      @RequestBody @Valid CozinhaInput cozinhaInput)
{
                Cozinha cozinhaAtual = cadastroCozinha.buscarOuFalhar(cozinhaId);
                        cozinhaInputDisassembler.copyToDomainObject(cozinhaInput,
cozinhaAtual);
                cozinhaAtual = cadastroCozinha.salvar(cozinhaAtual);

                return cozinhaModelAssembler.toModel(cozinhaAtual);
        }

        @DeleteMapping(value = "/{cozinhaId}", produces = {})
        @ResponseStatus(HttpStatus.NO_CONTENT)
        public void remover(@PathVariable Long cozinhaId) {
                cadastroCozinha.excluir(cozinhaId);
        }

}
```

Código-fonte atualizado

Código-fonte da aula com as dependências atualizadas