

Engenharia de Softwares Escaláveis

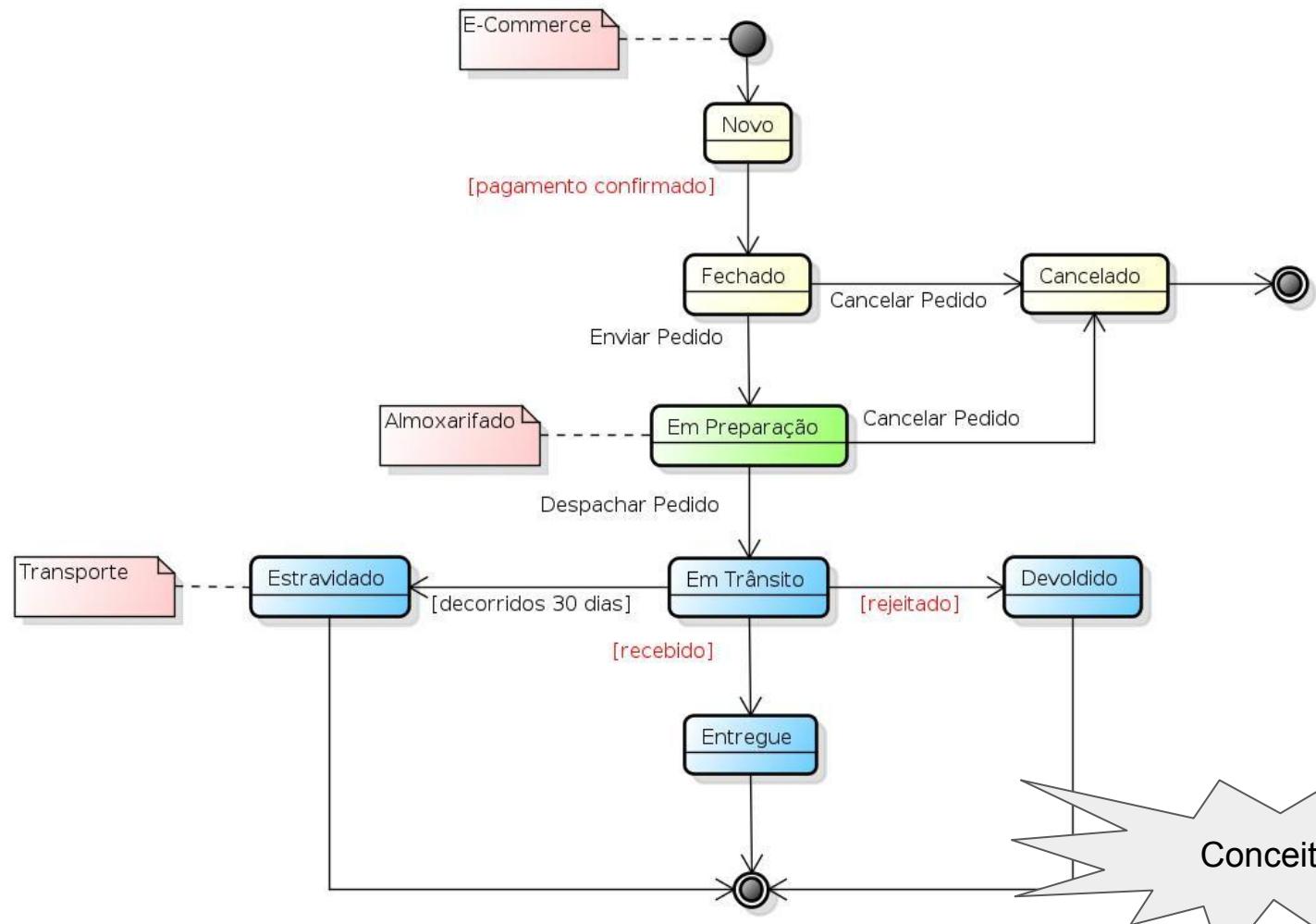
Domain-Driven Design (DDD) e Arquitetura de
Softwares Escaláveis com Java

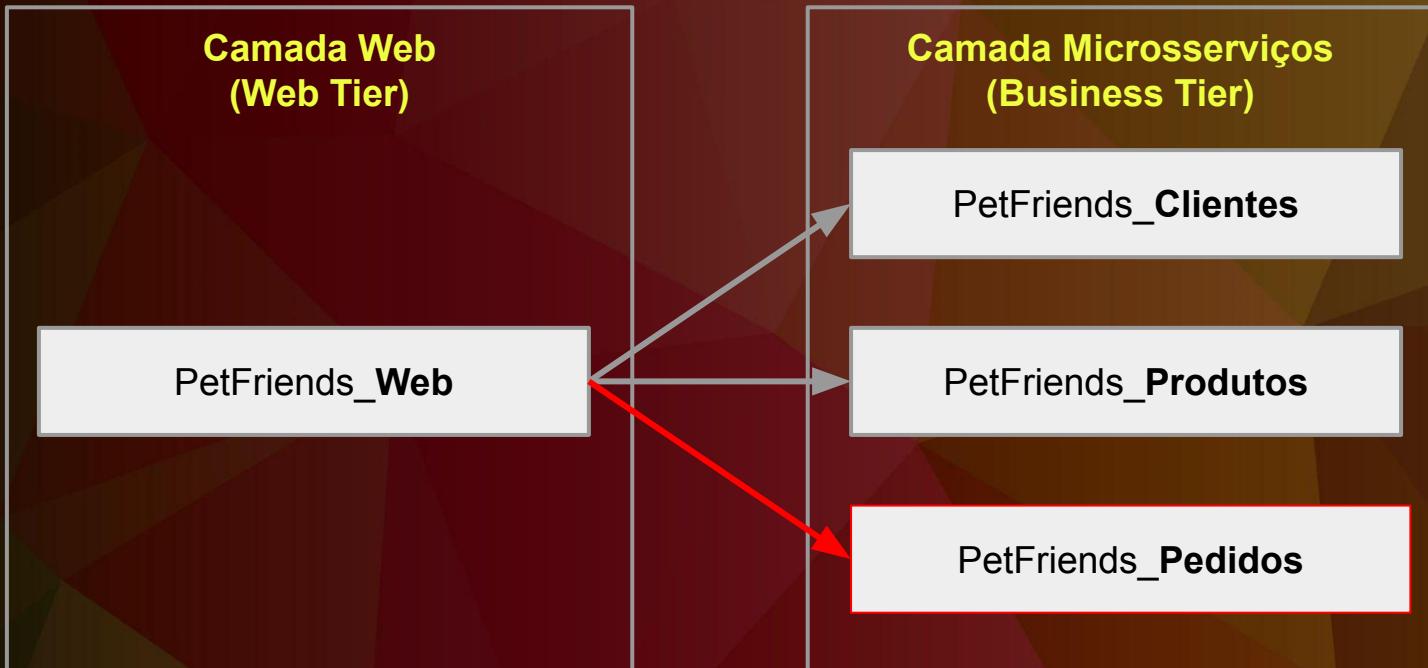
Agenda

Etapa 7: Prévia do AT.

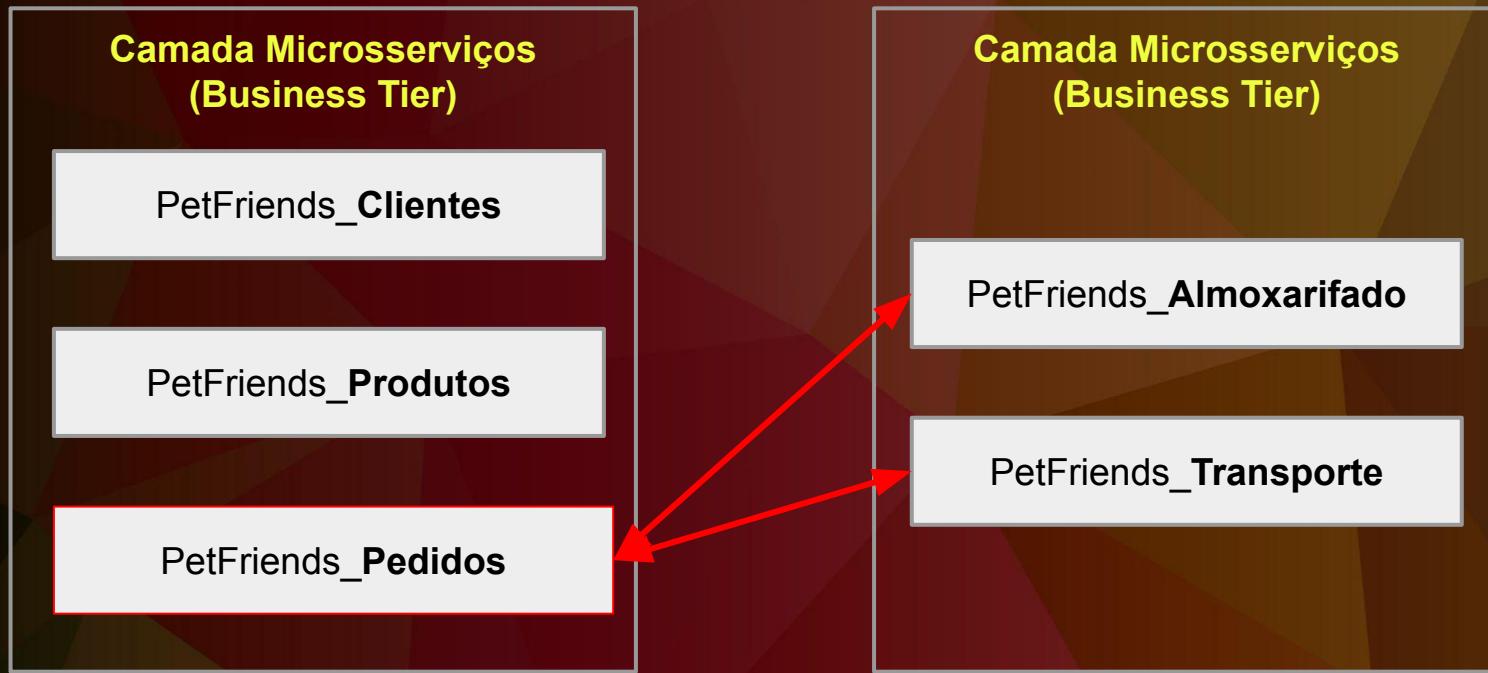


Prévia do AT





Conceitos



Conceitos

1. Transformar monólitos em microsserviços eficazes, aplicando princípios de DDD e técnicas de decomposição:
 - a. Implemente a classe entity e seu respectivo repository que represente o agregado mais representativo do microsserviço PetFriends_Almoçarifado.
 - b. Implemente um exemplo de Value Object a ser usado na classe entity da questão anterior.
 - c. Implemente a classe entity e seu respectivo repository que represente o agregado mais representativo do microsserviço PetFriends_Transporte.
 - d. Implemente um exemplo de Value Object a ser usado na classe entity da questão anterior.
2. Projetar softwares usando "domain events":
 - a. O módulo PetFriends_Web foi desenvolvido em ReactJS, acessando os microsserviços de Clientes, Produtos e Pedidos de forma síncrona, via REST API. Que funcionalidade síncrona executada pelo cliente é diretamente afetada pelos eventos de domínio?
 - b. Explique de forma sucinta qual é a diferença de enviar eventos somente com o ID do agregado e enviar um payload completo?
 - c. A partir da resposta dada na questão anterior, como você projetaria o evento a ser enviado pelo PetFriends_Pedido para o PetFriends_Almoçarifado?
 - d. A partir da resposta dada na questão anterior, como você projetaria o evento a ser enviado pelo PetFriends_Pedido para o PetFriends_Transporte?
3. Desenvolver microsserviços event-driven e com outros padrões de comunicação assíncrona:
 - a. No microsserviço PetFriends_Almoçarifado, implemente a classe de configuração para o tratamento de mensagens para receber os eventos do PetFriends_Pedidos.
 - b. No microsserviço PetFriends_Almoçarifado, implemente o serviço que receberá os eventos do PetFriends_Pedidos.
 - c. No microsserviço PetFriends_Transporte, implemente a classe de configuração para o tratamento de mensagens para receber os eventos do PetFriends_Pedidos.
 - d. No microsserviço PetFriends_Transporte, implemente o serviço que receberá os eventos do PetFriends_Pedidos.
4. Implementar testes e observabilidade em microsserviços com Zipkin, Spring Cloud Sleuth e ELK Stack:
 - a. Explique de forma sucinta o que é um Gateway de Serviço, suas vantagens e desvantagens.
 - b. O que é ID de Correlação e quais são os seus pré-requisitos?
 - c. Qual é a função do Spring Cloud Sleuth e sua relação com o serviço Zipkin?
 - d. Explique de forma sucinta o que é Agregador de Logs, suas vantagens e desvantagens.

Assim que terminar, salve seu trabalho em PDF nomeando o arquivo conforme a regra "nome_sobrenome_DR4_AT.PDF" e poste como resposta a este AT.

Assinaturas - Pub/Sub - Sof +

console.cloud.google.com/cloudpubsub/subscription/list?authuser=0&hl=pt&project=softwares-esca...

Administrativo Conteúdo Ferramentas Livros Localhost Todos os favoritos

Google Cloud Softwares Escaláveis top Pesquisa

Pub/Sub Assinuras CRIAR ASSINATURA EXCLUIR OCULTAR PAINEL DE INFORMAÇÕES

Pub/Sub Tópicos Assinuras Snapshots Esquemas Reservas do Lite Gerenciar recursos Notas de lançamento

Selecionar uma assinatura

PERMISSÕES MARCADORES

Selecionar pelo menos um recurso.

Setup

Estado	Código da assinatura	Tipo de envio	⋮	⋮
<input type="checkbox"/>	dr4_sub	Pull	⋮	⋮
<input checked="" type="checkbox"/>	PetFriends_Almoçarifado	Pull	⋮	⋮
<input checked="" type="checkbox"/>	PetFriends_Pedidos	Pull	⋮	⋮
<input checked="" type="checkbox"/>	PetFriends_Transporte	Pull	⋮	⋮



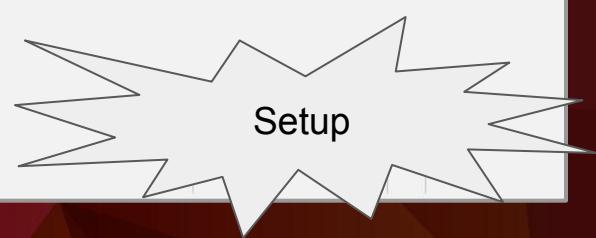
Projects - Infnet - Eng de Soft ... × Files Services

- > 🐾 DesignTatico
- > 🎨 DR4_E2_A1
- > 🎨 DR4_E2_A2
- > 🎨 DR4_E3_A2
- > 🎨 DR4_E4_A1
- > 🎨 DR4_E5_A1
- > 🎨 DR4_E6_A2 Stream

- 📁 PetFriends_Pedidos
 - Source Packages
 - > br.edu.infnet
 - > br.edu.infnet.controllers
 - > br.edu.infnet_pedidos.domain
 - > br.edu.infnet_pedidos.eventos
 - > br.edu.infnet_pedidos.infra.message
 - > br.edu.infnet_pedidos.infra.repository
 - > br.edu.infnet_pedidos.infra.service
 - > Test Packages
 - Other Sources
 - src/main/resources
 - 📁 <default package>
 - application.properties
 - softwares-escalaveis-a12fc70b8812.json
 - static
 - templates
 - > Dependencies
 - > Runtime Dependencies
 - > Test Dependencies
 - > Project Files



- New Project...
Ctrl+Shift+N
- Open Project...
Ctrl+Shift+O
- New File...
Ctrl+N
- Open File...
-
- Go to File...
Alt+Shift+O
- Show Dashboard
Alt+Shift+W



Setup

Source

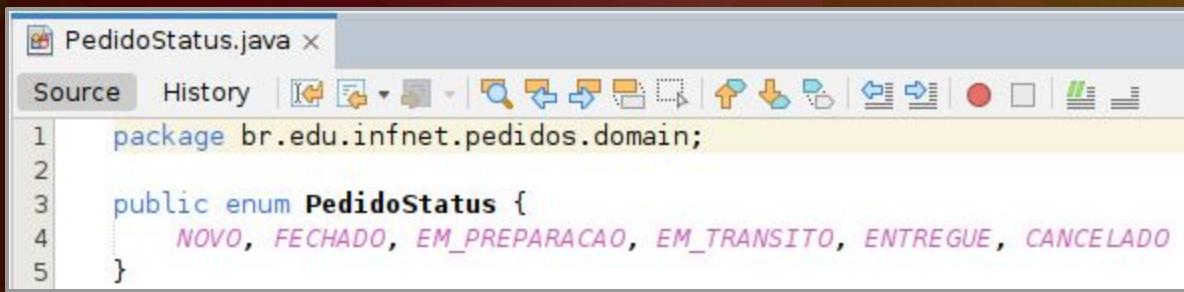
Graph

Effective

History

```
20 <dependency>
21   <groupId>org.springframework.boot</groupId>
22   <artifactId>spring-boot-starter-data-jpa</artifactId>
23 </dependency>
24 <dependency>
25   <groupId>org.springframework.boot</groupId>
26   <artifactId>spring-boot-starter-integration</artifactId>
27 </dependency>
28 <dependency>
29   <groupId>org.springframework.boot</groupId>
30   <artifactId>spring-boot-starter-web</artifactId>
31 </dependency>
32 <dependency>
33   <groupId>com.google.cloud</groupId>
34   <artifactId>spring-cloud-gcp-starter-pubsub</artifactId>
35 </dependency>
36 <dependency>
37   <groupId>org.springframework.integration</groupId>
38   <artifactId>spring-integration-http</artifactId>
39 </dependency>
40 <dependency>
41   <groupId>org.springframework.integration</groupId>
42   <artifactId>spring-integration-jpa</artifactId>
43 </dependency>
44 <dependency>
45   <groupId>com.h2database</groupId>
46   <artifactId>h2</artifactId>
47   <scope>runtime</scope>
48 </dependency>
49 <dependency>
50   <groupId>org.springdoc</groupId>
51   <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
52   <version>2.6.0</version>
53 </dependency>
```

Setup



```
1 package br.edu.infnet.pedidos.domain;
2
3 public enum PedidoStatus {
4     NOVO, FECHADO, EM_PREPARACAO, EM_TRANSITO, ENTREGUE, CANCELADO
5 }
```



Domínio

Domínio

```
1 package br.edu.infnet.pedidos.domain;
2
3 import ...19 lines
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23 @Entity
24 @Table(name = "pedidos", catalog = "DR4_1", schema = "PUBLIC")
25 public class Pedido implements Serializable {
26
27     private static final long serialVersionUID = 1L;
28     @Id
29     @GeneratedValue(strategy = GenerationType.IDENTITY)
30     @Column(nullable = false)
31     private Long id;
32     @Column(name = "ORDER_DATE")
33     @Temporal(TemporalType.DATE)
34     private Date orderDate;
35     @JsonIgnoreProperties(value = "orderId")
36     @OneToMany(mappedBy = "orderId", cascade = CascadeType.ALL)
37     private List<ItemPedido> itemList;
38     @Column(name = "CUSTOMER_ID")
39     private Long customerId;
40     @Convert(converter = PedidoStatusConverter.class)
41     @Column(name = "STATUS")
42     private PedidoStatus status;
43     @Convert(converter = ValorMonetarioConverter.class)
44     @Column(name = "VALOR_TOTAL")
45     private ValorMonetario valorTotal;
```

Domínio

Pedido.java x

Source History

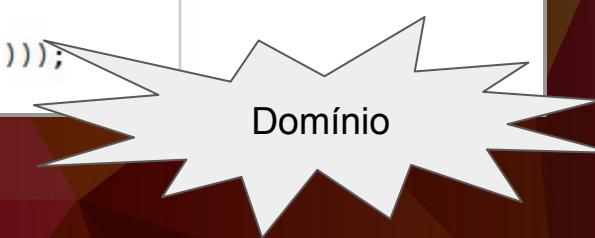
```
47     public Pedido() {
48         this.orderDate = new Date();
49         this.status = PedidoStatus.NOVO;
50         this.valorTotal = new ValorMonetario(BigDecimal.ZERO);
51     }
52
53     public Pedido(Long id) {
54         this.id = id;
55         this.orderDate = new Date();
56         this.status = PedidoStatus.NOVO;
57         this.valorTotal = new ValorMonetario(BigDecimal.ZERO);
58     }
59
60     public Pedido(Long id, Long customerId) {
61         this.id = id;
62         this.customerId = customerId;
63         this.orderDate = new Date();
64         this.status = PedidoStatus.NOVO;
65         this.valorTotal = new ValorMonetario(BigDecimal.ZERO);
66     }
```

Domínio

Pedido.java

Source History

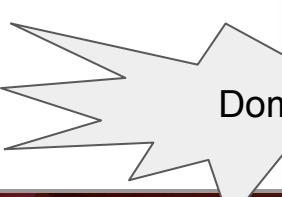
```
141     public void adicionarItem(Long productId, int quantidade, double unitValue) {  
142         if (productId == null) {  
143             throw new IllegalArgumentException("Produto inválido");  
144         }  
145         if (quantidade <= 0) {  
146             throw new IllegalArgumentException("Quantidade precisa ser positiva");  
147         }  
148         if (this.status != PedidoStatus.NOVO) {  
149             throw new IllegalStateException("Não é possível inserir itens em um pedido em andamento");  
150         }  
151         ItemPedido itemPedido = new ItemPedido();  
152         itemPedido.setOrderId(this);  
153         itemPedido.setProductId(productId);  
154         itemPedido.setQuantity(quantidade);  
155         itemPedido.setTotal(new ValorMonetario(new BigDecimal(unitValue * quantidade)));  
156         if (this.itemList == null) {  
157             this.itemList = new ArrayList<>();  
158         }  
159         this.itemList.add(itemPedido);  
160         this.setValorTotal(this.valorTotal.somar(itemPedido.getTotal()));  
161     }
```



Domínio

```
163     public void fecharPedido() {  
164         if (this.status != PedidoStatus.NOVO) {  
165             throw new IllegalStateException("Não é possível fechar um pedido que não é novo");  
166         }  
167         if (this.itemList.isEmpty()) {  
168             throw new IllegalStateException("Não é possível fechar um pedido vazio");  
169         }  
170         this.status = PedidoStatus.FECHADO;  
171     }  
172  
173     public void cancelarPedido() {  
174         if (this.status != PedidoStatus.FECHADO) {  
175             throw new IllegalStateException("Não é possível cancelar um pedido que não esteja fechado");  
176         }  
177         this.status = PedidoStatus.CANCELADO;  
178     }  
179  
180     public void enviarPedido() {  
181         if (this.status != PedidoStatus.FECHADO) {  
182             throw new IllegalStateException("Não é possível enviar um pedido que não esteja fechado");  
183         }  
184         this.status = PedidoStatus.EM_PREPARACAO;  
185     }  
186 }
```





ItemPedido.java

```
1 package br.edu.infnet.pedidos.domain;
2
3 import ...9 lines
4
5 @Entity
6 @Table(name = "itens_pedidos", catalog = "DR4_1", schema = "PUBLIC")
7 public class ItemPedido implements Serializable {
8
9     private static final long serialVersionUID = 1L;
10    @Id
11    @GeneratedValue(strategy = GenerationType.IDENTITY)
12    @Column(nullable = false)
13    private Long id;
14    private Integer quantity;
15    @Column(precision = 20, scale = 2)
16    private ValorMonetario total;
17    @Column(name = "PRODUCT_ID")
18    private Long productId;
19    @JoinColumn(name = "ORDER_ID", referencedColumnName = "ID")
20    @ManyToOne
21    private Pedido orderId;
22
23    public ItemPedido() {
24    }
25
26    public ItemPedido(Long id) {
27        this.id = id;
28    }
29
30
31
32
33
34
35
36
37
```

Domínio



EstadoPedidoMudou.java x

Source History

```
1 package br.edu.infnet.pedidos.eventos;
2
3 import ...3 lines
4
5
6 public class EstadoPedidoMudou implements Serializable {
7
8     private Long idPedido;
9     private PedidoStatus estado;
10    private Date momento;
11
12    public EstadoPedidoMudou() {
13    }
14
15
16    public EstadoPedidoMudou(Long idPedido, PedidoStatus estado) {
17        this.idPedido = idPedido;
18        this.estado = estado;
19        this.momento = new Date();
20    }
21
22    public EstadoPedidoMudou(Long idPedido, PedidoStatus estado, Date momento) {
23        this.idPedido = idPedido;
24        this.estado = estado;
25        this.momento = momento;
26    }
27
28    public Long getIdPedido() {
29        return idPedido;
30    }
31
32    public void setIdPedido(Long idPedido) {
33        this.idPedido = idPedido;
34    }
```



Eventos e
Mensageria

EstadoPedidoMudouSerializer.java x

Source History |

```
1 package br.edu.infnet.pedidos.infra.message;
2
3 import ...6 lines
4
5
6 public class EstadoPedidoMudouSerializer extends StdSerializer<EstadoPedidoMudou> {
7
8     public EstadoPedidoMudouSerializer() {
9         super(EstadoPedidoMudou.class);
10    }
11
12
13    @Override
14    public void serialize(EstadoPedidoMudou evento, JsonGenerator jgen, SerializerProvider provider) throws IOException {
15        jgen.writeStartObject();
16        jgen.writeNumberField("idPedido", evento.getIdPedido());
17        jgen.writeStringField("estado", evento.getEstado().toString());
18        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
19        String data = sdf.format(evento.getMomento());
20        jgen.writeStringField("momento", data);
21    }
22}
23
24
25}
```

Eventos e
Mensageria

EstadoPedidoMudouSerializer.java

EstadoPedidoMudouDeserializer.java

Source History

```
1 package br.edu.infnet.pedidos.infra.message;
2
3 import ...12 lines
4
5
6 public class EstadoPedidoMudouDeserializer extends StdDeserializer<EstadoPedidoMudou> {
7
8     public EstadoPedidoMudouDeserializer() {
9         super(EstadoPedidoMudou.class);
10    }
11
12    @Override
13    public EstadoPedidoMudou deserialize(JsonParser jp, DeserializationContext dc) throws IOException, JacksonException {
14        EstadoPedidoMudou evento = null;
15        JsonNode node = jp.getCodec().readTree(jp);
16        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
17        try {
18            evento = new EstadoPedidoMudou(
19                node.get("idPedido").asLong(),
20                PedidoStatus.valueOf(node.get("estado").asText()),
21                sdf.parse(node.get("momento").asText())
22            );
23        } catch (ParseException e) {
24            throw new IOException("Erro na data");
25        }
26        return evento;
27    }
28}
```



Eventos e
Mensageria

```
package br.edu.infnet.pedidos.infra.message;
import ...12 lines

@Configuration
public class PetFriendsPedidosMessageConfig {

    @Bean
    public JacksonPubSubMessageConverter estadoMudouConverter() {
        ObjectMapper objectMapper = new ObjectMapper();
        SimpleModule simpleModule = new SimpleModule();
        simpleModule.addSerializer(EstadoPedidoMudou.class, new EstadoPedidoMudouSerializer());
        simpleModule.addDeserializer(EstadoPedidoMudou.class, new EstadoPedidoMudouDeserializer());
        objectMapper.registerModule(simpleModule);
        return new JacksonPubSubMessageConverter(objectMapper);
    }
}
```

Eventos e Mensageria

```
29     @Bean
30     public MessageChannel inputMessageChannel() {
31         return new PublishSubscribeChannel();
32     }
33
34     @Bean
35     public PubSubInboundChannelAdapter inboundChannelAdapter(
36         @Qualifier("inputMessageChannel") MessageChannel messageChannel, PubSubTemplate pubSubTemplate) {
37
38         pubSubTemplate.setMessageConverter(estadoMudouConverter());
39         PubSubInboundChannelAdapter adapter = new PubSubInboundChannelAdapter(pubSubTemplate, "PetFriends_Pedidos");
40         adapter.setOutputChannel(messageChannel);
41         adapter.setAckMode(AckMode.MANUAL);
42         adapter.setPayloadType(EstadoPedidoMudou.class);
43         return adapter;
44     }
45 }
```

Eventos e Mensageria

The screenshot shows the IntelliJ IDEA interface with the file `PedidoRepository.java` open. The code defines a Spring Data JPA repository for the `Pedido` domain entity.

```
1 package br.edu.infnet.pedidos.infra.repository;
2
3 import br.edu.infnet.pedidos.domain.Pedido;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 public interface PedidoRepository extends JpaRepository<Pedido, Long> {
7 }
```

Persistência

```
1 package br.edu.infnet.pedidos.infra.repository;
2
3 import br.edu.infnet.pedidos.domain.PedidoStatus;
4 import jakarta.persistence.AttributeConverter;
5 import jakarta.persistence.Converter;
6
7 @Converter(autoApply = true)
8 public class PedidoStatusConverter implements AttributeConverter<PedidoStatus, String> {
9
10     @Override
11     public String convertToDatabaseColumn(PedidoStatus pedidoStatus) {
12         return pedidoStatus.toString();
13     }
14
15     @Override
16     public PedidoStatus convertToEntityAttribute(String pedidoStatus) {
17         return PedidoStatus.valueOf(pedidoStatus);
18     }
19 }
```

Persistência

Persistência

Assinaturas - Pub/Sub - Soft x H2 Console Swagger UI

localhost:8080/h2-console/login.do?jsessionid=0ad1f7dac2f40cfe14f0f389cc4c61c5

Administrativo Conteúdo Ferramentas Livros Localhost Todos os favoritos

Auto commit Max rows: 1000 Auto complete Off Auto select On

jdbc:h2:file:/home/armeniocardos Run Run Selected Auto complete Clear SQL statement:

+ ITENS_PEDIDOS
+ PEDIDOS
+ INFORMATION_SCHEMA
+ Users
H2 2.2.224 (2023-09-17)

SELECT * FROM PEDIDOS;

SELECT * FROM PEDIDOS;

ID	CUSTOMER_ID	ORDER_DATE	STATUS	VALOR_TOTAL
1	4321	2024-09-06	FECHADO	280.00

(1 row, 1 ms)

Edit



Persistência

Assinaturas - Pub/Sub - Soft x H2 Console Swagger UI

localhost:8080/h2-console/login.do?jsessionid=0ad1f7dac2f40cfe14f0f389cc4c61c5

Administrativo Conteúdo Ferramentas Livros Localhost Todos os favoritos

Auto commit Max rows: 1000 Auto complete Off Auto select On

jdbc:h2:file:/home/armeniocardos Run Run Selected Auto complete Clear SQL statement:

ITENS_PEDIDOS PESSOAS INFORMATION_SCHEMA Users H2 2.2.224 (2023-09-17)

SELECT * FROM ITENS_PEDIDOS;

ID	PRODUCT_ID	QUANTITY	TOTAL	ORDER_ID
1	1234	2	200.00	1
2	1235	1	50.00	1
3	1236	1	30.00	1

(3 rows, 2 ms)

Edit



Persistência

The screenshot shows the `PedidoService.java` file in a Java IDE. The code implements a service interface for managing orders. Several methods are highlighted with red boxes:

- `public Pedido obterPorId(long id)`
- `public Pedido fecharPedido(long id)`
- `public Pedido criarPedido(Pedido pedido)`

The code uses annotations like `@Service`, `@Autowired`, and `Logger` from the `infra.service` package. It interacts with a `PubSubTemplate` and a `JacksonPubSubMessageConverter` to handle messages, and a `PedidoRepository` to persist data.

```
1 package br.edu.infnet.pedidos.infra.service;
2
3 import ...14 lines
4
5 @Service
6 public class PedidoService {
7
8     private static final Logger LOG = LoggerFactory.getLogger(PedidoService.class)
9
10    @Autowired
11    private PubSubTemplate pubSubTemplate;
12    @Autowired
13    JacksonPubSubMessageConverter converter;
14    @Autowired
15    private PedidoRepository repository;
16
17    public Pedido obterPorId(long id) {
18        return repository.getReferenceById(id);
19    }
20
21    public Pedido fecharPedido(long id) {
22        Pedido pedido = repository.getReferenceById(id);
23        pedido.fecharPedido();
24        pedido = repository.save(pedido);
25        enviar(new EstadoPedidoMudou(pedido.getId(), PedidoStatus.FECHADO));
26        return pedido;
27    }
28
29    public Pedido criarPedido(Pedido pedido) {
30        Pedido retorno = null;
31        if (pedido != null) {
32            retorno = repository.save(pedido);
33            enviar(new EstadoPedidoMudou(pedido.getId(), PedidoStatus.NOVO));
34        }
35        return retorno;
36    }
37
38 }
```

Serviço

```
PedidoService.java x
34 public Pedido fecharPedido(long id) {
35
36     public void enviar(EstadoPedidoMudou estado) {
37         pubSubTemplate.setMessageConverter(converter);
38         pubSubTemplate.publish("dr4_topic", estado);
39         LOG.info("***** Mensagem Publicada ---> " + estado);
40     }
41
42     @ServiceActivator(inputChannel = "inputMessageChannel")
43     public void receber(EstadoPedidoMudou payload,
44                         @Header(GcpPubSubHeaders.ORIGINAL_MESSAGE) ConvertedBasicAcknowledgeablePubsubMessage<EstadoPedidoMudou> message) {
45
46         LOG.info("***** Mensagem Recebida ---> " + payload);
47         message.ack();
48     }
49
50 }
```

Serviço

```
1 package br.edu.infnet.controllers;
2
3 + import ...8 lines
11
12 @RestController
13 @RequestMapping(value = "/pedidos")
14 public class PedidoController {
15
16     @Autowired
17     private PedidoService service;
18
19     @GetMapping("/{id}")
20     public Pedido obtenerPorId(@PathVariable(value = "id") long id){
21         return service.obtenerPorId(id);
22     }
23
24     @PatchMapping("/fechar-pedido/{id}")
25     public Pedido fecharPedido(@PathVariable(value = "id") long id){
26         return service.fecharPedido(id);
27     }
28 }
```

Controller

Assinaturas - Pub/Sub - Soft x Swagger UI x

localhost:8080/swagger-ui/index.html

Administrativo Conteúdo Ferramentas Livros Localhost Todos os favoritos

Swagger Supported by SMARTBEAR /v3/api-docs Explore

OpenAPI definition v0 OAS 3.0

/v3/api-docs

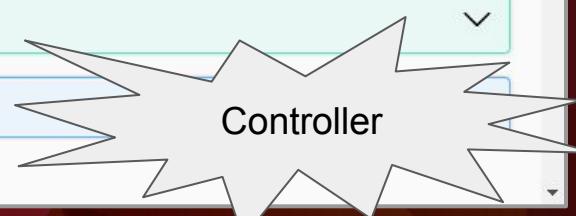
Servers

http://localhost:8080 - Generated server url ▾

pedido-controller

PATCH /pedidos/fechar-pedido/{id}

GET /pedidos/{id}



Assinaturas - Pub/Sub - Soft x Swagger UI x

localhost:8080/swagger-ui/index.html#/pedido-controller/obterPorId

Administrativo Conteúdo Ferramentas Livros Localhost Todos os favoritos

200 Response body

```
        "quantia": 200
    },
    "productId": 1234
},
{
    "id": 2,
    "quantity": 1,
    "total": {
        "quantia": 50
    },
    "productId": 1235
},
{
    "id": 3,
    "quantity": 1,
    "total": {
        "quantia": 30
    },
    "productId": 1236
}
],
{
    "customerId": 4321,
    "status": "FECHADO",
    "valorTotal": {
        "quantia": 280
    }
},
"hibernateLazyInitializer": {}
}
```

Response headers

```
connection: keep-alive
content-type: application/json
date: Fri, 06 Sep 2024 22:32:50 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Controller