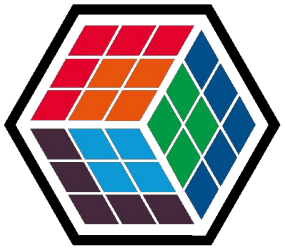


THE DEVELOPER'S CONFERENCE

Eventos de Aplicação com Spring

Lucas Farias

Sobre mim

THE DEVELOPER'S CONFERENCE

- Lucas Farias
- Backend Sênior no iFood (Logística)
- Graduado em Ciência da Computação
- Especialização em Sistemas Corporativos

@luksrn

ifood

O que será a palestra

THE DEVELOPER'S CONFERENCE

Spring Framework Documentation

Version 5.1.10.RELEASE

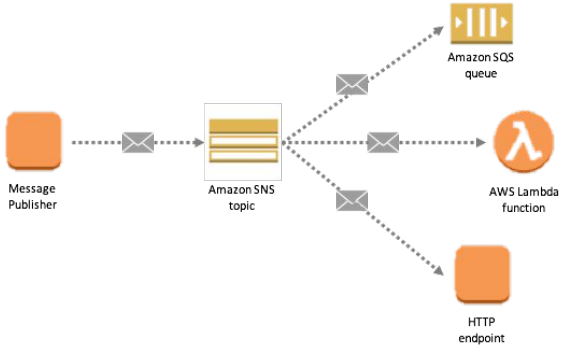
What's New, *Upgrade Notes*, *Supported Versions*, and other topics, independent of release cadence, are maintained externally on the project's Github Wiki.

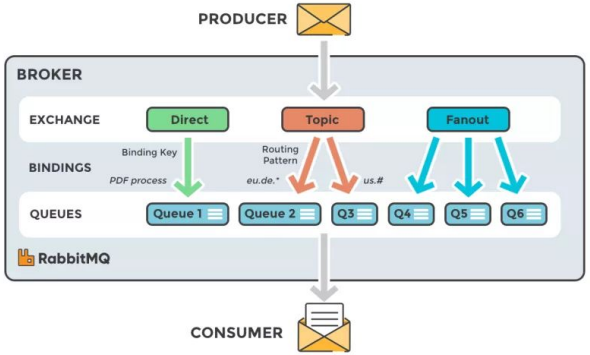
Overview	history, design philosophy, feedback, getting started.
Core	IoC Container, Events, Resources, i18n, Validation, Data Binding, Type Conversion, SpEL, AOP.
Testing	Mock Objects, TestContext Framework, Spring MVC Test, WebTestClient.
Data Access	Transactions, DAO Support, JDBC, O/R Mapping, XML Marshalling.
Web Servlet	Spring MVC, WebSocket, SockJS, STOMP Messaging.
Web Reactive	Spring WebFlux, WebClient, WebSocket.
Integration	Remoting, JMS, JCA, JMX, Email, Tasks, Scheduling, Caching.
Languages	Kotlin, Groovy, Dynamic Languages.

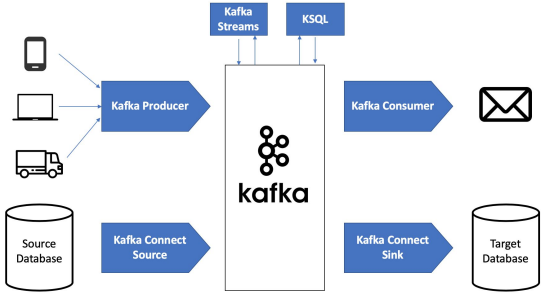
O que não será a palestra

Modelos de arquitetura baseada em eventos

- RabbitMQ
- SNS/SQS
- Kafka
- Pulsar
- etc...







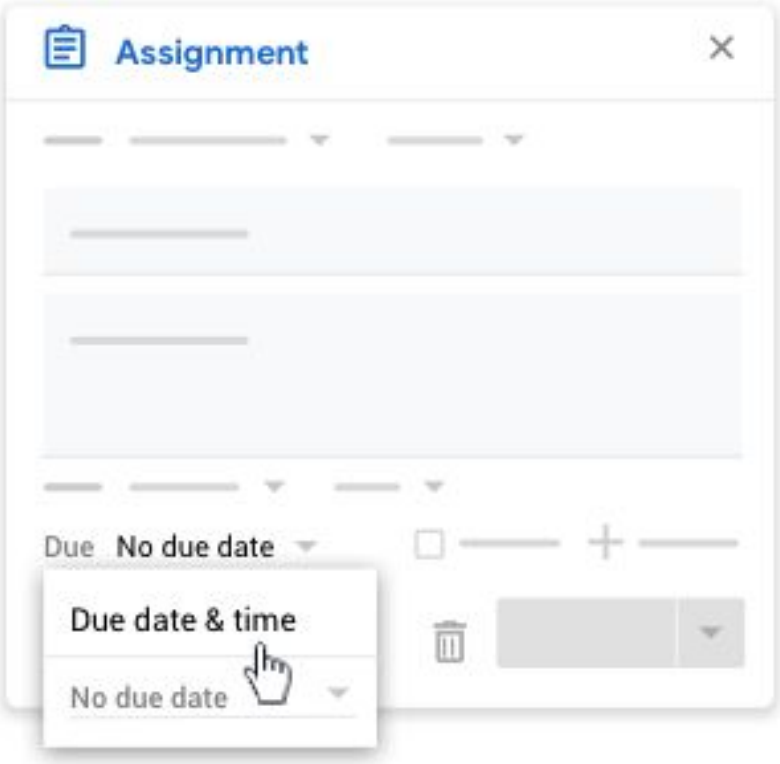
Como vou conduzir...

THE DEVELOPER'S CONFERENCE

- Dado um exemplo, será analisado um possível código
  - Possível cenário de Teste
  - Possível código para satisfazer Controller -> **Service**
- Discutir sobre alguns problemas do exemplo
- Conceitos de **Eventos de Aplicação**

Cenário de exemplo

THE DEVELOPER'S CONFERENCE



Os requisitos são simples:

- Apenas um **título** para aula, uma **descrição** detalhada e um **prazo** para entrega.
- Os alunos matriculados no curso devem ser **notificados via email**.

https://support.google.com/edu/classroom/answer/6020265?co=GENIE.Platform%3DDesktop&hl=en

O que estamos esperando...

THE DEVELOPER'S CONFERENCE

```
// Given {Hidden}...

CreateAssignmentRequest payload = new CreateAssignmentRequest(
        "Spring Application Events", "TDC talk about Spring event", Instant.now());

//when
MvcResult mvcResult = mvc.perform(post("/course/{courseId}/assignment/create", 1L)
        .contentType(MediaType.APPLICATION_JSON)
        .content(objectMapper.writeValueAsString(payload))
).andReturn();

// Then ...
```

THE DEVELOPER'S CONFERENCE

```
....

// then
SoftAssertions softly = new SoftAssertions();

softly.assertThat(mvcResult.getResponse().getStatus())
        .isEqualTo(HttpStatus.CREATED.value());

List<Assignment> assignments = assignmentService.findAllByCourse(new Course(1L));
softly.assertThat(assignments)
        .hasSize(1)
        .flatExtracting(Assignment::getTitle, Assignment::getInstructions)
        .contains("Spring Application Events", "TDC talk about Spring event");

softly.assertThat(mockingDetails(emailService).getInvocations().size()).isEqualTo(2);
softly.assertThat(emailCaptor.getAllValues())
        .hasSize(2)
        .containsExactlyInAnyOrder(emailToBenjamin, emailToZooey);

softly.assertAll();
```

THE DEVELOPER'S CONFERENCE

```
....

// then
SoftAssertions softly = new SoftAssertions();

softly.assertThat(mvcResult.getResponse().getStatus())
        .isEqualTo(HttpStatus.CREATED.value());

List<Assignment> assignments = assignmentService.findAllByCourse(new Course(1L));
softly.assertThat(assignments)
        .hasSize(1)
        .flatExtracting(Assignment::getTitle, Assignment::getInstructions)
        .contains("Spring Application Events", "TDC talk about Spring event");

softly.assertThat(mockingDetails(emailService).getInvocations().size()).isEqualTo(2);
softly.assertThat(emailCaptor.getAllValues())
        .hasSize(2)
        .containsExactlyInAnyOrder(emailToBenjamin, emailToZooey);

softly.assertAll();
```

THE DEVELOPER'S CONFERENCE

```
....

// then
SoftAssertions softly = new SoftAssertions();

softly.assertThat(mvcResult.getResponse().getStatus())
        .isEqualTo(HttpStatus.CREATED.value());

List<Assignment> assignments = assignmentService.findAllByCourse(new Course(1L));
softly.assertThat(assignments)
        .hasSize(1)
        .flatExtracting(Assignment::getTitle, Assignment::getInstructions)
        .contains("Spring Application Events", "TDC talk about Spring event");

softly.assertThat(mockingDetails(emailService).getInvocations().size()).isEqualTo(2);
softly.assertThat(emailCaptor.getAllValues())
        .hasSize(2)
        .containsExactlyInAnyOrder(emailToBenjamin, emailToZooey);

softly.assertAll();
```

Como poderia ser implementado...

THE DEVELOPER'S CONFERENCE

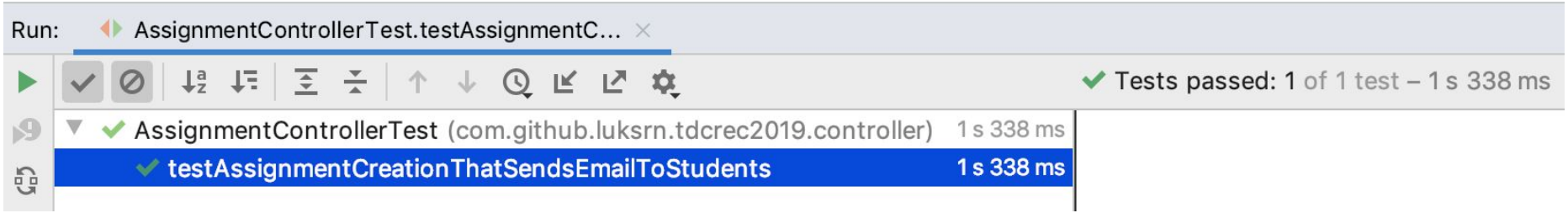
```
@PostMapping("/course/{courseId}/assignment/create")
public ResponseEntity<?> createAssignment(@PathVariable("courseId") Course course,
                                          @RequestBody CreateAssignmentRequest request){
    try {
        Assignment assignment = assignmentService.create(course,
                        conversionService.convert(request, Assignment.class));
        Resource<Assignment> resource = assembler.toResource(assignment);
        return ResponseEntity.created(
                    URI.create(resource.getId().expand().getHref())
                ).build();
    } catch (Exception e){
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
    }
}
```

THE DEVELOPER'S CONFERENCE

```
@Service
@AllArgsConstructor
public class AssignmentService {
   private AssignmentRepository assignmentRepository;
   private EmailService emailService;
   private CourseRepository courseRepository;
   @Transactional
   public Assignment create(Course course, Assignment assignment){
       // Any other validations,action here...
       assignmentRepository.save(assignment);

       courseRepository.findStudents(course).forEach( student ->  {
           Email email = new Email(student.getEmail(),
                   "New assignment created '" + assignment.getTitle() +"'",
                           assignment.getInstructions());
           emailService.send(email);
       });
       return assignment;
   }
}
```

Testes OK!?

THE DEVELOPER'S CONFERENCE



Qual o problema?

THE DEVELOPER'S CONFERENCE

- Design de código

THE DEVELOPER'S CONFERENCE

```
@Service
@AllArgsConstructor
public class AssignmentService {
    private AssignmentRepository assignmentRepository;
    private EmailService emailService;
    private CourseRepository courseRepository;
    @Transactional
    public Assignment create(Course course, Assignment assignment){
        // Any other validations,action here...
        assignment.setCourse(course);
        assignmentRepository.save(assignment);

        courseRepository.findStudents(course).forEach( student ->  {
            Email email = new Email(student.getEmail(),
                    "New assignment created '" + assignment.getTitle() +"'",
                            assignment.getInstructions());
            emailService.send(email);
        });
        return assignment;
    }
}
```

THE DEVELOPER'S CONFERENCE

```
@Service
@AllArgsConstructor
public class AssignmentService {
    private AssignmentRepository assignmentRepository;
    private EmailService emailService;
    private CourseRepository courseRepository;
    @Transactional
    public Assignment create(Course course, Assignment assignment){
        // Any other validations,action here...
        assignment.setCourse(course);
        assignmentRepository.save(assignment);

        courseRepository.findStudents(course).forEach( student ->  {
            final Email email = new Email(student.getEmail(),
                    "New assignment created '" + assignment.getTitle() +"'",
                            assignment.getInstructions());
            emailService.send(email);
        });
        return assignment;
    }
}
```

S
O
L
I
D

Qual o outro problema?

THE DEVELOPER'S CONFERENCE

Novo caso de teste

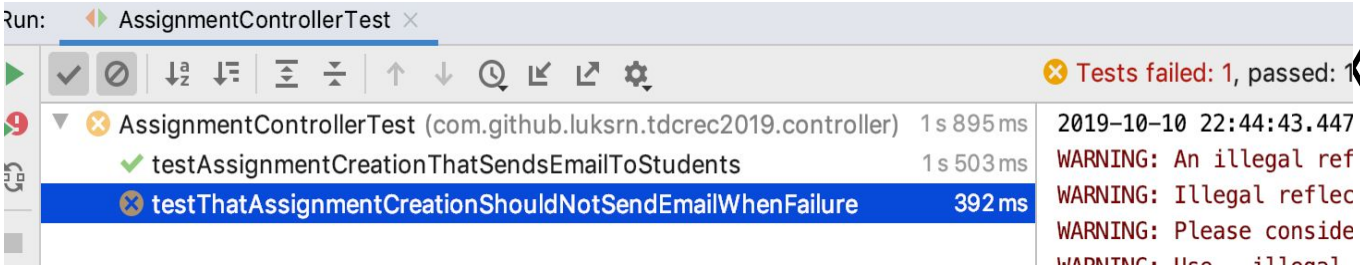
THE DEVELOPER'S CONFERENCE

```
CreateAssignmentRequest payload = new CreateAssignmentRequest(
        null, null, null); // ← Invalid payload

//when
MvcResult mvcResult = mvc.perform(post("/course/{courseId}/assignment/create",
1L)
        .contentType(MediaType.APPLICATION_JSON)
        .content(objectMapper.writeValueAsString(payload))
).andReturn();

// then
SoftAssertions softly = new SoftAssertions();
softly.assertThat(mvcResult.getResponse().getStatus())
        .isEqualTo(HttpStatus.BAD_REQUEST.value());
softly.assertThat(assignmentService.findAllByCourse(new Course(1L))).isEmpty();
softly.assertThat(mockingDetails(emailService).getInvocations())
    .as("Shouldn't interact with EmailService").isEmpty();
softly.assertAll();
```

Novo caso de teste

THE DEVELOPER'S CONFERENCE

```
CreateAssignmentRequest payload = new CreateAssignmentRequest(
        null, null, null); // ← Invalid payload

//when
MvcResult mvcResult = mvc.perform(post("/course/{courseId}/assignment/create",
1L)
        .contentType(MediaType.APPLICATION_JSON)
        .content(objectMapper.writeValueAsString(payload))
).andReturn();

// then
SoftAssertions softly = new SoftAssertions();
softly.assertThat(mvcResult.getResponse().getStatus())
        .isEqualTo(HttpStatus.BAD_REQUEST.value());
softly.assertThat(assignmentService.findAllByCourse(new Course(1L))).isEmpty();
softly.assertThat(mockingDetails(emailService).getInvocations())
    .as("Shouldn't interact with EmailService").isEmpty();
softly.assertAll();
```



THE DEVELOPER'S CONFERENCE

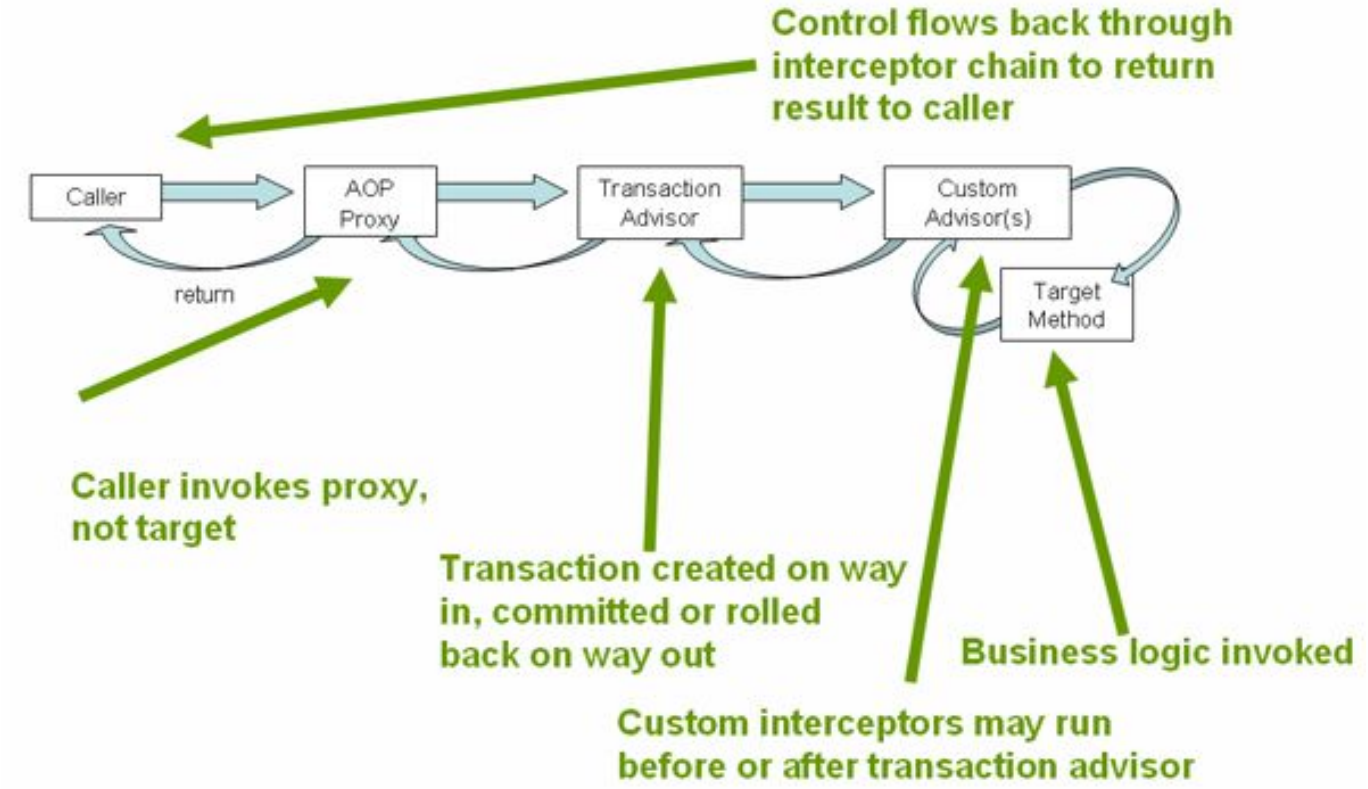
```
The following 2 assertions failed:
1) expected:<[4]00> but was:<[5]00>
2) [Shouldn't interact with EmailService]
Expecting empty but was:<[emailService bean.send(
    Email(to=benjamin@email.com, subject=New assignment created
'null', body=null)
);,
    emailService bean.send(
    Email(to=zoey@email.com, subject=New assignment created
'null', body=null)
);]>
```

THE DEVELOPER'S CONFERENCE

```
@Service
@AllArgsConstructor
public class AssignmentService {
    private AssignmentRepository assignmentRepository;
    private EmailService emailService;
    private CourseRepository courseRepository;

    @Transactional
    public Assignment create(Course course, Assignment assignment){
        // Any other validations,action here...
        assignment.setCourse(course);
        assignmentRepository.save(assignment);

        courseRepository.findStudents(course).forEach( student ->  {
            final Email email = new Email(student.getEmail(),
                    "New assignment created '" + assignment.getTitle() +"'",
                            assignment.getInstructions());
            emailService.send(email);
        });
        return assignment;
    }
```

Visão conceitual **@Transactional**

THE DEVELOPER'S CONFERENCE



https://docs.spring.io/spring/docs/current/spring-framework-reference/data-access.html#tx-decl-explained

Qual o problema?

THE DEVELOPER'S CONFERENCE

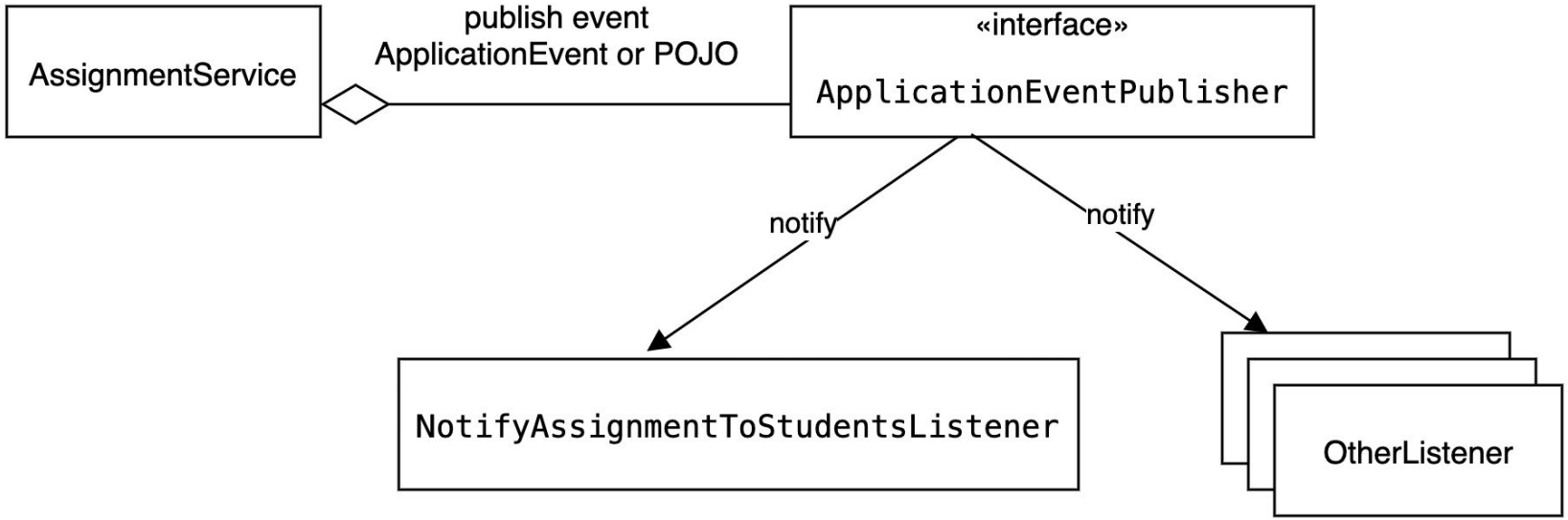
- Sincronização de Transação
  - Mais difícil de ser identificado
  - Extração de código para fora do Service?!
    - Alteração de assinatura
    - Como fica uma situação de composição?
  - E se… integração com:
    - SNS? RabbitMQ? Redis?...

Podemos fazer diferente?

THE DEVELOPER'S CONFERENCE

- E se...ao cadastrar a atividade fosse gerado um **evento** de "Atividade criada".
- Se a **notificação de alunos** fosse um **ouvinte** deste evento?
- Se uma ação fosse executada **após o Commit** da transação?

Eventos de Aplicação



O que seria nosso evento?

THE DEVELOPER'S CONFERENCE

```
@Value
public final class AssignmentCreated {
    private Assignment assignment;
}
```

THE DEVELOPER'S CONFERENCE

```
@Service
@AllArgsConstructor
public class AssignmentService {
    private AssignmentRepository assignmentRepository;
    private EmailService emailService;
    private CourseRepository courseRepository;
    @Transactional
    public Assignment create(Course course, Assignment assignment){
        // Any other validations,action here...
        assignment.setCourse(course);
        assignmentRepository.save(assignment);

        courseRepository.findStudents(course).forEach( student ->  {
            final Email email = new Email(student.getEmail(),
                    "New assignment created '" + assignment.getTitle() +"'",
                            assignment.getInstructions());
            emailService.send(email);
        });
        return assignment;
    }
}
```

THE DEVELOPER'S CONFERENCE

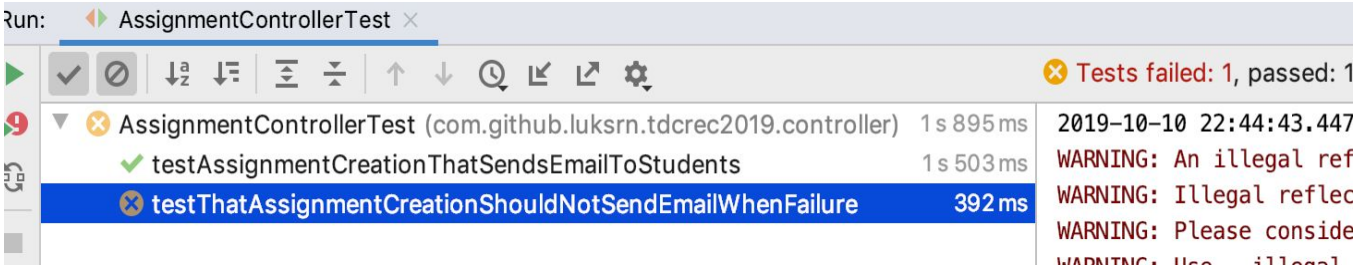
```
@Component
@AllArgsConstructor
public class NotifyAssignmentToStudentsListener {
    private EmailService emailService;
    private CourseRepository courseRepository;

    @EventListener
    public void onApplicationEvent(AssignmentCreated event){
        Assignment assignment = event.getAssignment();
        // course#getStudents is Lazy, so...
      courseRepository.findStudents(assignment.getCourse()).forEach(
          student ->  {
            Email email = new Email(student.getEmail(),
                    "New assignment created '" + assignment.getTitle() +"'",
                    assignment.getInstructions());
            emailService.send(email);
        });
    }}
```

THE DEVELOPER'S CONFERENCE

```
@Service
@AllArgsConstructor
public class AssignmentService {

    private AssignmentRepository assignmentRepository;
    private ApplicationEventPublisher publisher;

    @Transactional
    public Assignment create(final Course course, final Assignment assignment){
        // Any other validations,action here...
        assignment.setCourse(course);
        assignmentRepository.save(assignment);

        publisher.publishEvent(new AssignmentCreated(assignment));

        return assignment;
    }
```

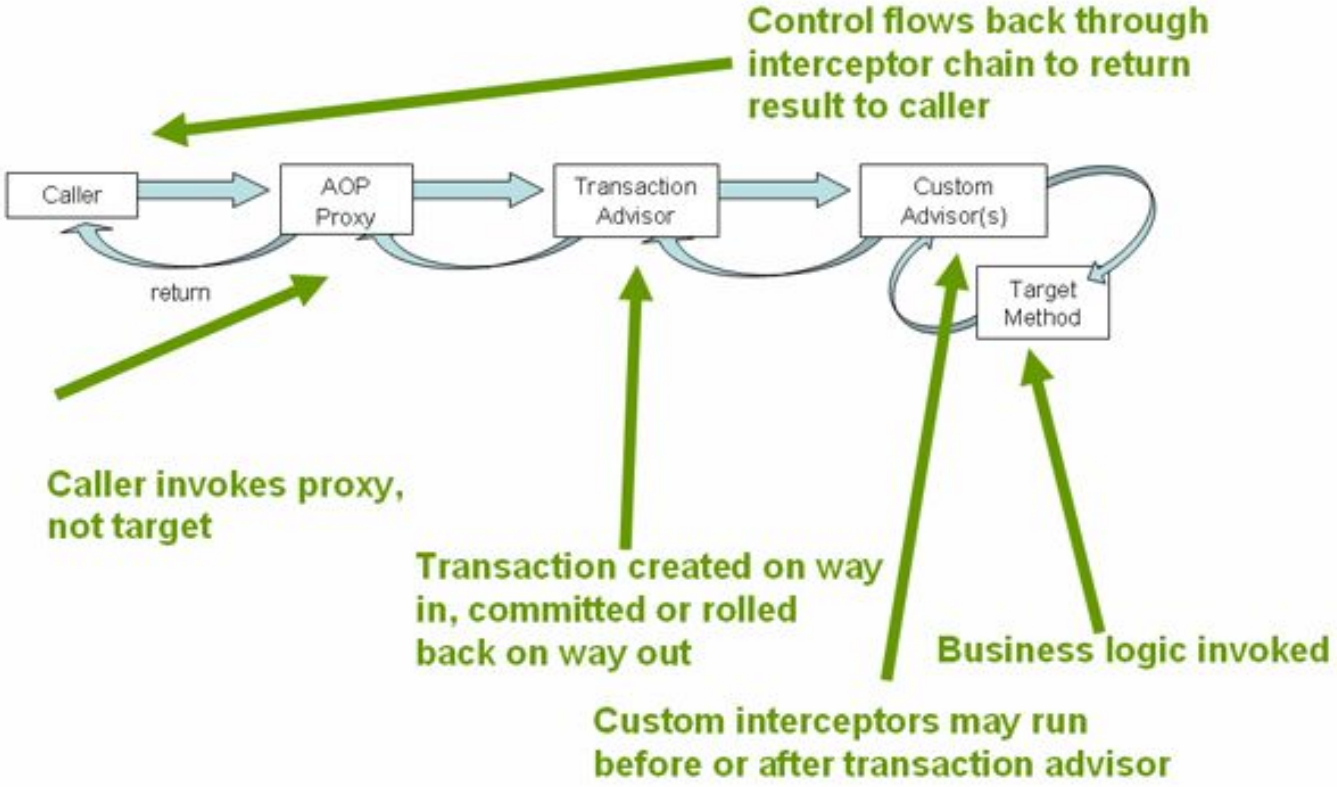
Mas espera… whaaat?

THE DEVELOPER'S CONFERENCE



```
The following 2 assertions failed:
1) expected:<[4]00> but was:<[5]00>
2) [Shouldn't interact with EmailService]
Expecting empty but was:<[emailService bean.send(
    Email(to=benjamin@email.com, subject=New assignment created
'null', body=null)
)...]>
```

Visão conceitual @Transactional



https://docs.spring.io/spring/docs/current/spring-framework-reference/data-access.html#tx-decl-explained

THE DEVELOPER'S CONFERENCE

```
@Component
@AllArgsConstructor
public class NotifyAssignmentToStudentsListener {
   private EmailService emailService;
   private CourseRepository courseRepository;

   @TransactionalEventListener(phase = TransactionPhase.AFTER_COMMIT)
   public void handle(AssignmentCreated event){
       Assignment assignment = event.getAssignment();
       // course#getStudents is Lazy, so...
     courseRepository.findStudents(assignment.getCourse()).forEach(
         student ->  {
           final Email email = new Email(student.getEmail(),
                   "New assignment created '" + assignment.getTitle() +"'",
                   assignment.getInstructions());
           emailService.send(email);
       });
   }}
```

THE DEVELOPER'S CONFERENCE

Run: AssignmentControllerTest

Tests passed: 2 of 2 tests

AssignmentControllerTest (com.github.luksrn.tdcrec2019.contr 1 s 695 ms

testAssignmentCreationThatSendsEmailToStudents 1 s 421 ms

testThatAssignmentCreationShouldNotSendEmailWhenFailure 274 ms

/Users/lucasfarias/.sdkma

23:36:51.505 [main] DEBUG

23:36:51.529 [main] DEBUG

23:36:51.541 [main] DEBUG

Considerações

- Código fica menos acoplado, seguindo o príncipio do **SO**LID
  - **S**ingle Responsibility
  - **O**pen Closed
- Boa opção para tratar **Sincronização de ações com Transação**
- Conheça outras funcionalidades
  - Eventos genéricos
  - Eventos Assíncronos
  - Conditional com SpEL

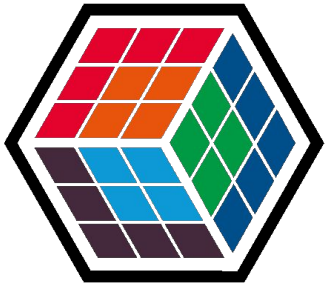
Considerações

- Como tratar consistência entre os Listeners?
  - **Spring Retry**
- Quando seu domínio possuir **diversos estados** que alteram em **eventos**, e que realiza **ações em cada transição** soluções melhores podem ser aplicadas.
  - **Spring State Machine**

Código da apresentação

THE DEVELOPER'S CONFERENCE

https://github.com/luksrn/java-eventos-tdcrecife2019



THE DEVELOPER'S CONFERENCE