THE DEVELOPER'S CONFERENCE

Java & Reactive Streams

Ironi Jr. Medina
Software Engineer @SAP

Agenda

THE DEVELOPER'S CONFERENCE

- Conceitos importantes
  - Reactive Systems
- Reactive Streams
  - Conceito
  - Como funciona
  - Java 9+
  - Implementações
- Considerações Finais

Reactive Systems

Reactive Systems

THE DEVELOPER'S CONFERENCE

Antigas aplicações

- Segundos de response time
- Horas de downtime
- Gigabytes de dados
- Pouca importância pra TCO

Aplicações atuais

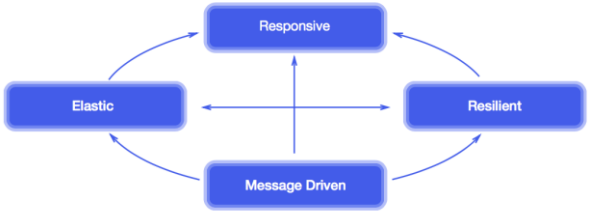
- Milissegundos de response time
- Zero downtime
- Petabytes de dados
- Low TCO é a lei

Diferentes usuários

Reactive Systems

THE DEVELOPER'S CONFERENCE

- Responsive
  - resposta rápida + consistente
- Resilient
  - se mantém responsivo, qualquer em circunstância
- Elastic
  - escalabilidade de recursos e instâncias
- Message-Driven
  - mensageria assíncrona
  - diminuir acoplamento, separando responsabilidades



THE
DEVELOPER'S
CONFERENCE

REACTIVE STREAMS

O que é?

THE DEVELOPER'S CONFERENCE

- Especificação que define um padrão para o processamento de streams de forma **assíncrona** e com **back pressure não bloqueante**

- **Assíncrona:** não espera um retorno imediato; processamento paralelo
- **Back pressure:** consumidor pode dizer pro produtor ir mais devagar
- **Não bloqueante:** threads principais sempre liberadas

https://reactive-streams.org

O que é?

THE DEVELOPER'S CONFERENCE

- reactive-streams.jar
- 4 interfaces
  - **Publisher**
  - **Subscriber**
  - **Subscription**
  - **Processor**
- JDK 9+
  - java.util.concurrent.Flow.*

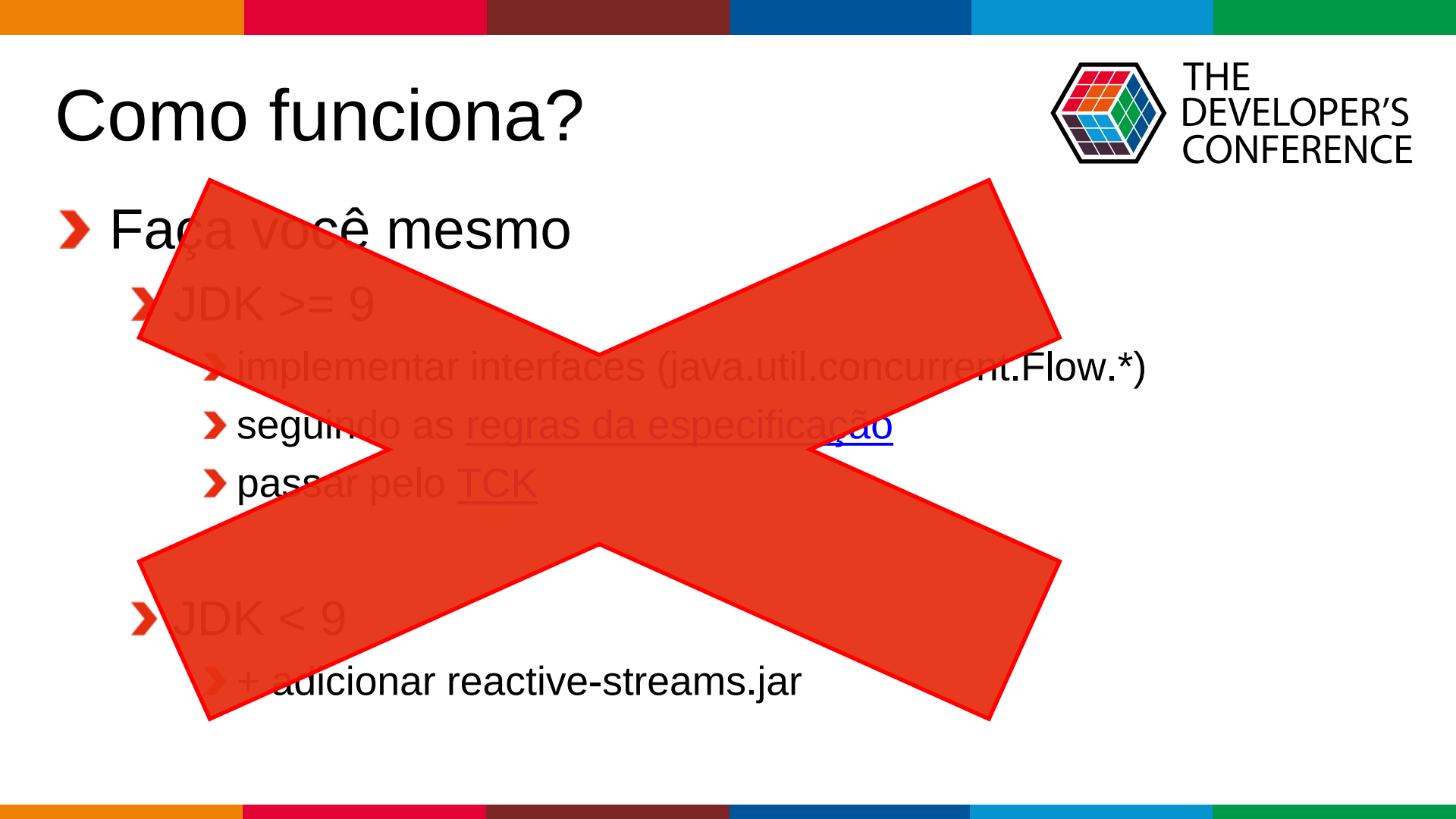
```
public interface Publisher<T> {
    public void subscribe(Subscriber<? super T> s);
}

public interface Subscriber<T> {
    public void onSubscribe(Subscription s);
    public void onNext(T t);
    public void onError(Throwable t);
    public void onComplete();
}

public interface Subscription {
    public void request(long n);
    public void cancel();
}

public interface Processor<T, R> extends Subscriber<T>, Publisher<R> {
}
```

Como funciona?



- Faça você mesmo
  - JDK >= 9
    - implementar interfaces (java.util.concurrent.Flow.*)
    - seguindo as regras da especificação
    - passar pelo TCK
  - JDK < 9
    - + adicionar reactive-streams.jar

Como funciona?

THE DEVELOPER'S CONFERENCE

- Bibliotecas de terceiros
  - Project Reactor
    - Pivotal; Spring 5; Java 8+
  - RxJava 2 & 3
    - Netflix; Java 6+
  - Akka-Streams
    - Lightbend; Java 8+
  - Vert.x
    - Eclipse; Quarkus; Java 8+


I HAVE NO IDEA
WHAT I'M DOING
DEMO

Reactor 3

THE DEVELOPER'S CONFERENCE

- Flux -> impl. de Publisher -> 0 ~ n
- Mono -> impl. de Publisher -> 0 ~ 1
- Operadores para aplicar processamento nos dados (map(), filter(), etc)
- Lazy evaluation

Desvantagens

THE DEVELOPER'S CONFERENCE

- Difícil de debbugar (mesmo com libs)
- Muitas libs ainda bloqueantes
- Pouca/nenhuma opção de banco de dados reativo*
- **Muito fácil fazer código bloqueante**

*somente connection

Resumo

THE DEVELOPER'S CONFERENCE

- O futuro é reativo, low tco
- Reactive Streams = SPI
- Usar libs ao invés de “reinventar a roda”
- Mudança de paradigma

  (imperative -> functional -> reactive)

THANK YOU

Ironi Júnior Medina

Software Engineer @ SAP

/ironijunior

/in/ironi-junior-medina

ironimedina@gmail.com

THE DEVELOPER'S CONFERENCE