Building Reactive Applications

Freddy Pena

- ➔ *Java Developer.*
- ➔ *+10 años de experiencia en desarrollo bajo tecnología Java, y arquitectura de Software.*
- ➔ *Profesor **PUCMM** y el **TEP**.*
- ➔ *Miembro de **Java Dominicano**.*
- ➔ *Amante del Open Source, Cafe y el Rock.*

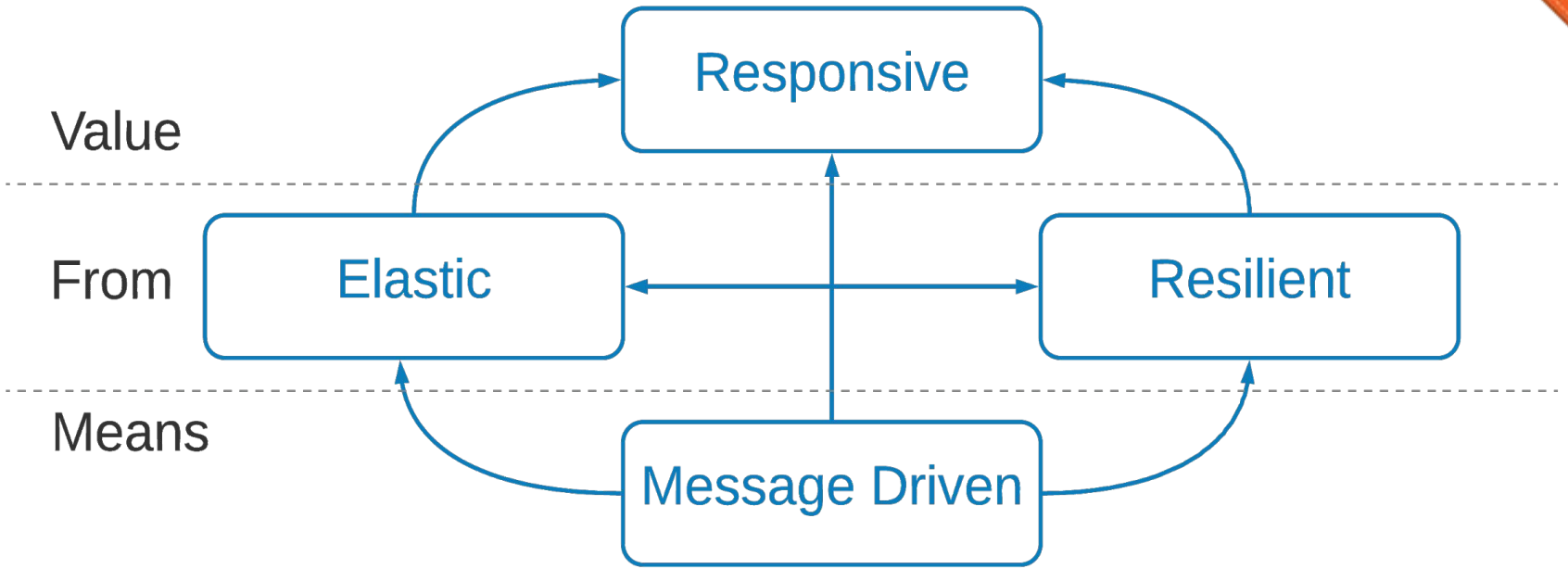
Agenda

- ➔ Que son sistemas reactivos?
- ➔ El manifiesto de los sistemas reactivos.
- ➔ Vert.x, qué es?, qué no es?
- ➔ Conceptos claves en Vert.x
- ➔ API RESTful, qué es?
- ➔ Open API y Swagger

Que son sistemas Reactivos?

- Más flexibles
- Débilmente acoplados y escalables
- Más fáciles de desarrollar y susceptibles de cambio
- Tolerantes a fallas y manejo elegante de ellas.
- Altamente receptivos.

https://www.reactivemanifesto.org/



Reactive System

https://www.reactivemanifesto.org/

Qué son sistemas Reactivos?

1. Responsive:

Responde de manera oportuna, si es posible. Brindar tiempos de respuesta rápidos y consistentes, estableciendo límites superiores confiables para que brinden una calidad de servicio constante.

https://www.reactivemanifesto.org/

Qué son sistemas Reactivos?

2. Resilient:

Es la capacidad que un sistema tiene de seguir funcionando en caso de errores, es decir, la aplicación tiene que estar diseñada para tratar con errores y seguir funcionando correctamente aunque ocurran desastres y perturbaciones.

https://www.reactivemanifesto.org/

Qué son sistemas Reactivos?

3. Elastic:

El sistema debe permanecer Responsive bajo una carga de trabajo variable, lo que implica que tiene que estar preparado para poder escalar.

https://www.reactivemanifesto.org/

Qué son sistemas Reactivos?

4. Message Driven:

Comunicación mediantes mensajes/eventos asíncronos, tanto entre ellos como entre componentes de terceros, esto incluye errores, y transparencia en la localización de los componentes, todo estos altamentes desacoplados.

https://www.reactivemanifesto.org/

Porqué Sistemas Reactivos?

	Hace 10 años	Presente y prueba a futuro
Alojamiento	10’s Servidores	1000’s Nodos o Contenedores
Tiempo de Respuesta	Segundos	Milisegundos, <0.200s
Tiempo de Mantenimiento	Horas	Sin pausa, en caliente.
Volumen de data	GBs	TBs -> PTs

https://www.reactivemanifesto.org/

Eclipse Vert.x es un tool-kit para construir aplicaciones **reactivas** en la **JVM**.

https://vertx.io/

Eclipse Vert.x: Es una plataforma de aplicaciones ligera y de alto rendimiento, diseñado para aplicaciones móviles, web y empresariales modernas

https://vertx.io/

Vert.x es un proyecto Open Source bajo ***Eclipse Foundation***. Iniciado en 2012 por Tim Fox.

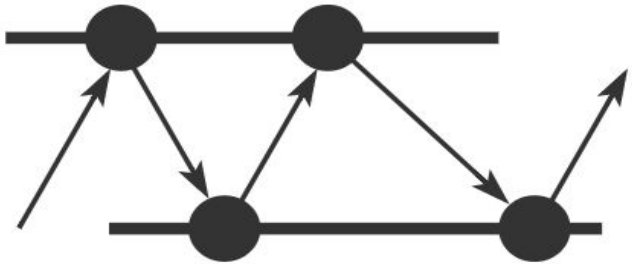
https://vertx.io/

VERT.X

Vert.x no es un servidor de aplicaciones, ni un framework, son simplemente componentes que pueden añadirse a una aplicación sin la necesidad de modificar la estructura de la misma

https://vertx.io/

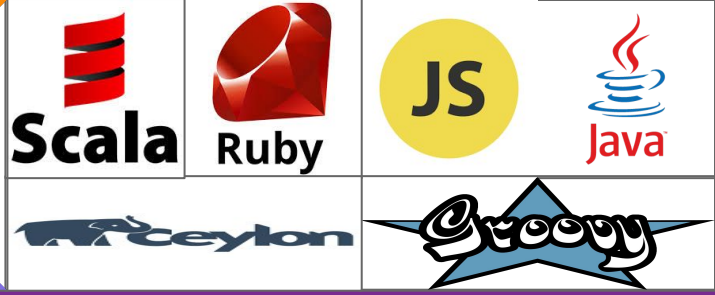
Escalable



Eclipse ***Vert.x*** es ***event driven*** y ***non blocking***. Esto quiere decir que nuestras apps pueden procesar mucha concurrencia usando un número pequeño de hilos del kernel. ***Vert.x*** nos permite escalar con hardware mínimos.

https://vertx.io/

Poliglota



Podemos usar ***Vert.x*** con múltiples lenguajes incluyendo **Java**, **JavaScript**, **Groovy**, **Ruby**, **Ceylon**, **Scala** y **Kotlin**.
Vert.x no predica sobre qué lenguaje es el mejor, usted elige el lenguaje que desea en función de la tarea en cuestión y el conjunto de habilidades de su equipo.

https://vertx.io/

Poliglota:

```java
//Java
import io.vertx.core.AbstractVerticle;

public class Server extends AbstractVerticle {
  public void start() {
    vertx.createHttpServer().requestHandler(req -> {
      req.response()
        .putHeader("content-type", "text/plain")
        .end("Hello from Vert.x!");
    }).listen(8080);
  }
}
```

```javascript
//JavaScript
vertx.createHttpServer()
  .requestHandler(function (req) {
    req.response()
      .putHeader("content-type", "text/plain")
      .end("Hello from Vert.x!");
}).listen(8080);
```

https://vertx.io/

Poliglota:

```groovy
//Groovy
vertx.createHttpServer().requestHandler({ req ->
  req.response()
    .putHeader("content-type", "text/plain")
    .end("Hello from Vert.x!")
}).listen(8080)
```

```ruby
//Ruby
$vertx.create_http_server.request_handler { |req|
  req.response
    .put_header("content-type", "text/plain")
    .end("Hello from Vert.x!")
}.listen(8080)
```

https://vertx.io/

Poliglota:

```ceylon
//Ceylon
import io.vertx.ceylon.core { ... }
import io.vertx.ceylon.core.http { ... }

shared class Server() extends Verticle() {
  start() ⇒ vertx.createHttpServer()
    .requestHandler((req) ⇒
      req.response()
        .putHeader("content-type", "text/plain")
        .end("Hello from Vert.x!")
    ).listen(8080);
}
```

```scala
//Scala
import io.vertx.lang.scala.ScalaVerticle

class Server extends ScalaVerticle {
  override def start(): Unit = {
    vertx
      .createHttpServer()
     .requestHandler(_.response()
       .putHeader("content-type", "text/plain")
       .end("Hello from Vert.x"))
     .listen(8080)
  }
}}
```

https://vertx.io/

Poliglota:

```kotlin
//Kotlin
import io.vertx.core.AbstractVerticle

class Server : AbstractVerticle() {
  override fun start() {
    vertx.createHttpServer()
      .requestHandler { req ->
        req.response()
          .putHeader("content-type", "text/plain")
          .end("Hello from Vert.x")
      }.listen(8080)
  }
}
```

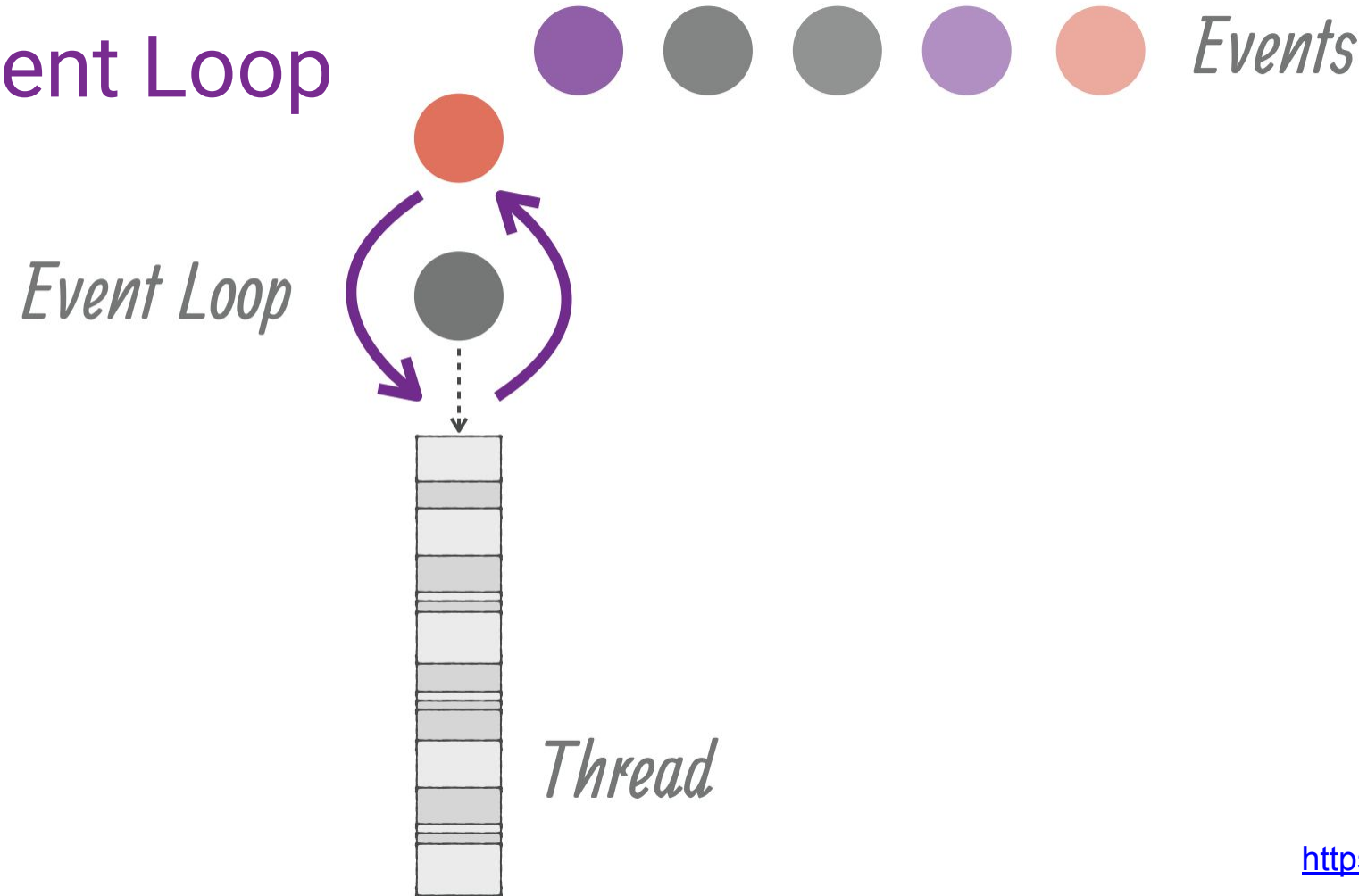
https://vertx.io/

Conceptos claves en Vertx

- Event Loop
- Verticle
- Event Bus
- Router
- Service Proxy

https://vertx.io/

Event Loop



https://vertx.io/

Nunca bloquees el Event Loop.

Thread.sleep(3000);

https://vertx.io/

Verticle

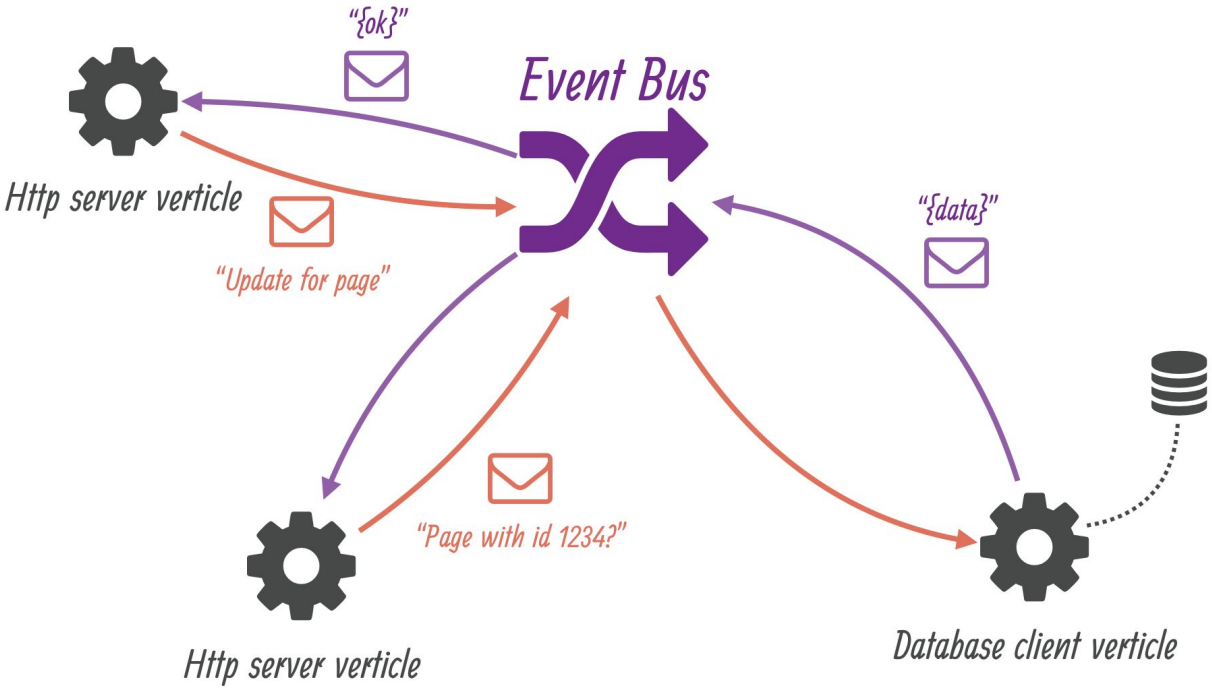
```java
import io.vertx.core.AbstractVerticle;

public class ServerVerticle extends AbstractVerticle {

  @Override
    public void start(Future<Void> future) throws Exception {
        super.start();
      // TODO: Add some code here
    }
}
```

Tipos de Verticles:

- Standard Verticle
- Worker Verticle

https://vertx.io/

Event Bus



Point to Point
Publish / Subscribe
Request / Response

https://vertx.io/

Router

Este es uno de los componentes core de **Vert.x Web**. Este se encarga de tener registradas las rutas a las que nuestra aplicación va a responder, para cuando reciba una petición sobre una de ella, así llamar a su manejador asociado.

https://vertx.io/

Service Proxy

Permite exponer un servicio en el even bus, por lo que cualquier otro componente Vert.x puede consumirlo, tan pronto como sepan la dirección en la que se publica el servicio.

El objetivo principal de los Service Proxy es aislar una funcionalidad en algún lugar y ponerla a disposición del resto de su aplicación.

https://vertx.io/

{RESTful API}

REST (Representational State Transfer), transferencia de estado representacional. Es un **Estilo Arquitectónico** de desarrollo web que se apoya totalmente en el estándar HTTP.

Open API

OpenAPI Specification (anteriormente conocida como, Swagger Specification) es un formato de descripción API para las API REST.

Open API

Permite describir su API completa, que incluye:

- Endpoints disponible (**/users**) y operaciones en cada Endpoints (**GET / users, POST / users**)
- Parámetros de operación Entrada y salida para cada operación.
- Métodos de autenticación

Open API

- Las especificaciones de API se pueden escribir en YAML o JSON. El formato es fácil de aprender y legible tanto para humanos como para máquinas.
- La especificación completa de OpenAPI se puede encontrar en GitHub:

https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md

Por qué? Open API

La capacidad de que un API pueda describir su propia estructura, es la raíz de toda la genialidad en OpenAPI. Una vez escrito, una especificación OpenAPI y Swagger pueden impulsar su desarrollo de API de varias maneras

Vert.x Web API Contract admite **OpenAPI 3**, ofreciéndole una interfaz simple para construir su enrutador y montar el controlador de seguridad y validación.

Swagger

Swagger es un conjunto de herramientas de código abierto creadas alrededor de la especificación OpenAPI que puede ayudarlo a diseñar, construir, documentar y consumir API REST. Las principales herramientas de Swagger incluyen:

- Swagger Editor
- Swagger UI
- Swagger Codegen

https://editor.swagger.io

keep calm and I'll show you the code

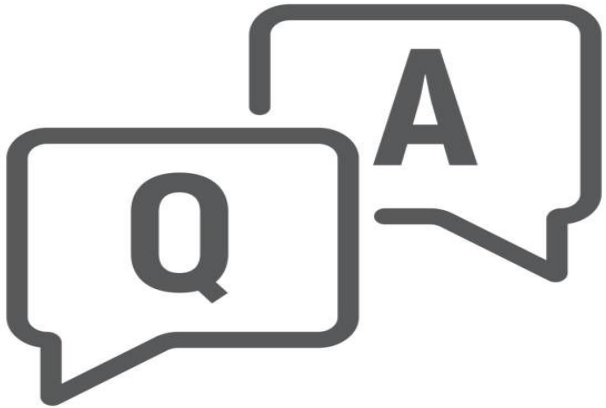
Vert.x RESTful Lab

https://github.com/fredpena/vertx-restful-lab

Google Slide

https://docs.google.com/presentation/d/1qqchUjkM0_ZI52I9RExH8ZDgcUKrV17IDSLjle3uTVY/edit?usp=sharing

Gracias por su tiempo!



@fantpena

@fredpena

@fred_pena