# Reactive Streams

**Reactive Streams** is an initiative to provide a standard for asynchronous stream processing with non-blocking back pressure.[1]

## Reactive Streams

| | |
|---|---|
| **Initial release** | 15 May 2015 |
| **Stable release** | 1.0.3 / 23 August 2019 |
| **Repository** | github.com/reactive-streams/reactive-streams-jvm (https://github.com/reactive-streams/reactive-streams-jvm) |
| **Written in** | Java (Scala, Groovy, Kotlin), .NET |
| **Website** | reactive-streams.org (http://reactive-streams.org) |

## Contents

## Origin

Reactive Streams started as an initiative in late 2013 between engineers at Netflix, Pivotal and Lightbend. Some of the earliest discussions began in 2013 between the Play and Akka teams at Lightbend.[2][3] Lightbend is one of the main contributors of Reactive Streams.[4] Other contributors include Red Hat, Oracle, Twitter and spray.io.[5]

## Goals

The main goal of Reactive Streams is to govern the exchange of stream data across an asynchronous boundary – like passing elements on to another thread or thread-pool – while ensuring that the receiving side is not forced to buffer arbitrary amounts of data. In other words, back pressure is an integral part of this model in order to allow the queues which mediate between threads to be bounded.

The intention of the specification is to allow the creation of many conforming implementations, which by virtue of abiding by the rules will be able to interoperate smoothly, preserving the mentioned benefits and characteristics across the whole processing graph of a stream application. A freely-available Technology Compatibility Kit[6] was developed alongside the specification that allows implementors of the specification to verify if they covered all rules and requirements, including checks for potential race conditions.

The scope of Reactive Streams is a minimal set of interfaces, methods and protocols that describe the necessary operations and entities to achieve the asynchronous streams of data with non-blocking back pressure. End-user DSLs or protocol binding APIs have purposefully been left out of the scope to encourage and enable different implementations that potentially use different programming languages to stay as true as possible to the idioms of their platform.

## Inclusion in Java standard

The specification developed with the intent of future inclusion in the official Java standard library, if proven successful and adopted by enough libraries and vendors.

Reactive Streams were proposed to become part of Java 9 by Doug Lea, leader of JSR 166[7] as a new Flow class[8] that would include the interfaces currently provided by Reactive Streams.[4][9] After a successful 1.0 release of Reactive Streams and growing adoption, the proposal was accepted and Reactive Streams was included in JDK9 via the JEP-266.[9]

## Adoption

On April 30, 2015 version 1.0.0 of Reactive Streams for the JVM was released,[4][5][10] including Java API,[11] a textual specification,[12] a TCK and implementation examples. It comes with a multitude of compliant implementations verified by the TCK for 1.0.0, listed in alphabetical order:[10]

- Akka Streams[13][14]
- MongoDB[15]
- Ratpack[16]
- Reactive Rabbit – driver for RabbitMQ/AMQP
- Spring and Pivotal Project Reactor[17]
- Netflix RxJava[18]
- Slick 3.0[19][20]
- Vert.x 3.0[21]

Other implementations include Cassandra,[22] Elasticsearch,[23] Apache Kafka,[24] Parallel Universe Quasar,[25] Play Framework,[26] Armeria.[27]

Spring 5 is announced to be built upon Reactive Streams compatible Reactor Core.[28]

Amazon announced that its Amazon Web Services SDK would support Reactive Streams to provide streaming capabilities in its client libraries in version 2.0.[29]

Reactive Streams 1.0.1 is released on August 9, 2017, including various improvements in specification preciseness, TCK improvements and other clarifications. The specification as well as interfaces remained fully backwards compatible with the 1.0.0 version, however aimed to streamline the adoption for future implementors as well as align with some additional requirements set by the OpenJDK.[30]

## Ports and influences

- A direct port of the specification, interfaces and TCK was made available under the same working group for the .NET platform.[31]
- When the Elixir language introduced its streaming API called GenStage, the authors extended a thanks "[to] akka-streams and reactive-streams projects which provided us guidance in implementing the demand-driven exchange between stages".[32]

## References

1. reactive-streams.org (http://www.reactive-streams.org/)
2. A Journey into Reactive Streams (https://medium.com/@kvnwbbr/a-journey-into-reactive-streams-5ee2a9cd7e29#.2wqcc3cja)

3. Reactive Streams 1.0.0 interview (https://medium.com/@viktorklang/reactive-streams-1-0-0-interview-faaca2c00bec#.3gh42y3gd)
4. Reactive Streams Releases First Stable Version for JVM (http://www.infoq.com/news/2015/06/Reactive-Streams-JVM-Version)
5. Reactive Streams 1.0.0 – a new standard in reactive data processing (https://jaxenter.com/reactive-streams-1-0-0-a-new-standard-in-reactive-data-processing-117123.html)
6. "Reactive Streams TCK" (https://github.com/reactive-streams/reactive-streams-jvm/tree/master/tck).
7. jdk9 Candidate classes Flow and SubmissionPublisher (http://cs.oswego.edu/pipermail/concurrency-interest/2015-January/013641.html)
8. java.util.concurrent.Flow (http://gee.cs.oswego.edu/dl/jsr166/dist/docs/java/util/concurrent/Flow.html)
9. JEP 266: More Concurrency Updates (http://openjdk.java.net/jeps/266)
10. Reactive Streams 1.0.0 is here! (http://www.reactive-streams.org/announce-1.0.0)
11. Java API (http://www.reactive-streams.org/reactive-streams-1.0.0-javadoc/)
12. Reactive Streams for the JVM specification (https://github.com/reactive-streams/reactive-streams-jvm/blob/v1.0.0/README.md#specification)
13. InfoQ: Reactive Streams with Akka Streams (http://www.infoq.com/news/2014/04/reactive-streams-akka-streams)
14. Design Principles behind Akka Streams (http://doc.akka.io/docs/akka-stream-and-http-experimental/2.0.1/stream-design.html)
15. MongoDB Reactive Streams Java Driver (https://mongodb.github.io/mongo-java-driver-reactivestreams/)
16. Ratpack: The Reactive Streams API (https://ratpack.io/manual/current/streams.html#the_reactive_streams_api)
17. Reactor 2.0.0.RC1 with native Reactive Streams support now available! (https://spring.io/blog/2015/02/18/reactor-2-0-0-rc1-with-native-reactive-streams-support-now-available)
18. Advanced RxJava: The Reactive-Streams API (part 1) (http://akarnokd.blogspot.com/2015/06/the-reactive-streams-api-part-1.html)
19. Slick 3: Reactive Streams for Asynchronous Database Access in Scala (http://www.infoq.com/news/2015/05/slick3)
20. Slick 3.0.0 (http://slick.typesafe.com/news/2015/04/29/slick-3.0.0-released.html)
21. Vert.x Reactive Streams Integration (http://vertx.io/docs/vertx-reactive-streams/java/)
22. Accessing Cassandra in a Reactive Way (http://www.planetcassandra.org/blog/accessing-cassandra-in-a-reactive-way/)
23. elastic4s — Non blocking, type safe DSL and Scala client for Elasticsearch (https://github.com/sksamuel/elastic4s)
24. Reactive Streams for Apache Kafka (https://softwaremill.com/reactive-kafka/)
25. Quasar and Reactive Streams (http://blog.paralleluniverse.co/2015/06/25/reactive-streams/)
26. Play Framework — Reactive Streams integration (experimental) (https://www.playframework.com/documentation/2.4.x/ReactiveStreamsIntegration)
27. Armeria - Completely asynchronous and reactive (https://line.github.io/armeria/index.html#completely-asynchronous-and-reactive)
28. Reactive Spring (http://spring.io/blog/2016/02/09/reactive-spring)
29. "AWS SDK for Java 2.0 developer preview announcement" (https://aws.amazon.com/blogs/developer/aws-sdk-for-java-2-0-developer-preview/).
30. "Reactive Streams 1.0.1 Release Announcement" (http://www.reactive-streams.org/announce-1.0.1).
31. "Reactive Streams .NET" (https://github.com/reactive-streams/reactive-streams-dotnet).

32. "Elixir blog: Announcing GenStage" (https://elixir-lang.org/blog/2016/07/14/announcing-genstage/).

- *This article incorporates text from www.reactive-streams.org (http://www.reactive-streams.org/), which is released under a CC0 1.0 Universal (CC0 1.0) Public Domain Dedication*.

Retrieved from "https://en.wikipedia.org/w/index.php?title=Reactive_Streams&oldid=963451847"