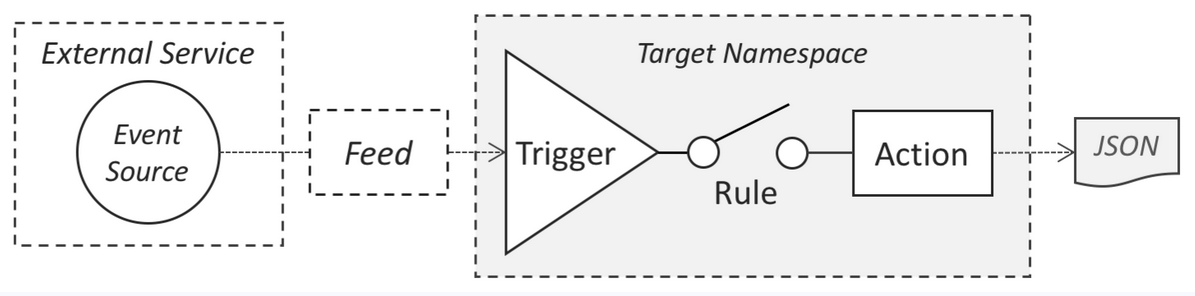
**OpenWhisk Programming Model**



The OpenWhisk programming model...

**Is Event-driven**

In the OpenWhisk programming model, Events drive the Serverless execution of functional code called Actions. Events can come from any Event Source or Feed service including:

* Datastores, Message Queues, Mobile and Web Applications, Sensors, Chatbots, Scheduled tasks (via Alarms), etc.

**Supports any functional programming language**

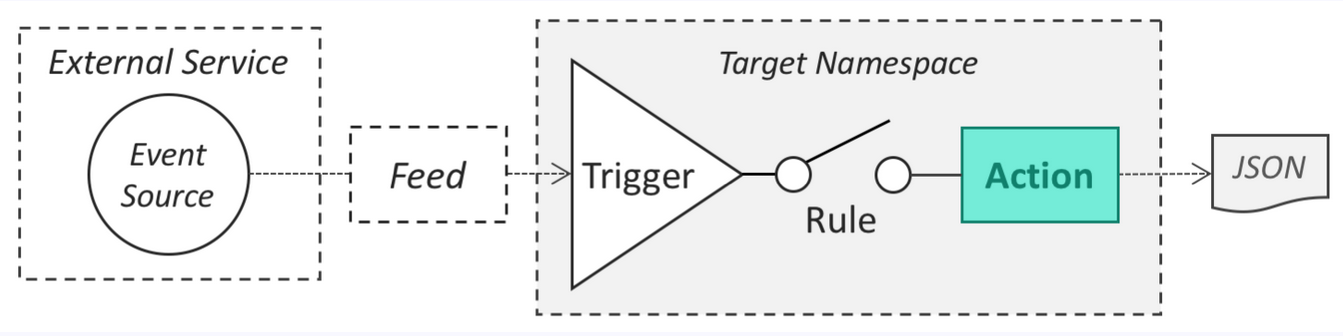
The OpenWhisk platform supports Action code written for any of its ever-growing, built-in [language runtimes](https://github.com/apache/openwhisk/blob/master/docs/actions.md#languages-and-runtimes).

The following is a list of runtimes the Apache OpenWhisk project has officially released and made available on our [Downloads](https://openwhisk.apache.org/downloads.html#component-releases) page.

* [.Net](https://github.com/apache/openwhisk-runtime-dotnet#readme) - OpenWhisk runtime for .Net Core 2.2
* [Go](https://github.com/apache/openwhisk-runtime-go#readme) - OpenWhisk runtime for Go
* [Java](https://github.com/apache/openwhisk-runtime-java#readme) - OpenWhisk runtime for Java 8 *(OpenJDK 8, JVM OpenJ9)*
* [JavaScript](https://github.com/apache/openwhisk-runtime-nodejs#readme) - OpenWhisk runtime for Node.js v6, v8 and v10
* [PHP](https://github.com/apache/openwhisk-runtime-php#readme) - OpenWhisk runtime for PHP 7.3, 7.2 and 7.1
* [Python](https://github.com/apache/openwhisk-runtime-python#readme) - OpenWhisk runtime for Python 2.7, 3 and a 3 runtime variant for AI/ML *(including packages for Tensorflow and PyTorch)*
* [Ruby](https://github.com/apache/openwhisk-runtime-ruby#readme) - OpenWhisk runtime for Ruby 2.5
* [Swift](https://github.com/apache/openwhisk-runtime-swift#readme) - OpenWhisk runtime for Swift 3.1.1, 4.1 and 4.2

If you need languages or libraries the current "out-of-the-box" runtimes do not support, you can create and customize your own executable that run "black box" [Docker Actions](https://github.com/apache/openwhisk-runtime-docker/blob/master/sdk/docker/README.md) using the Docker SDK which are run on the [Docker Runtime](https://github.com/apache/openwhisk-runtime-docker#readme).

**Actions**

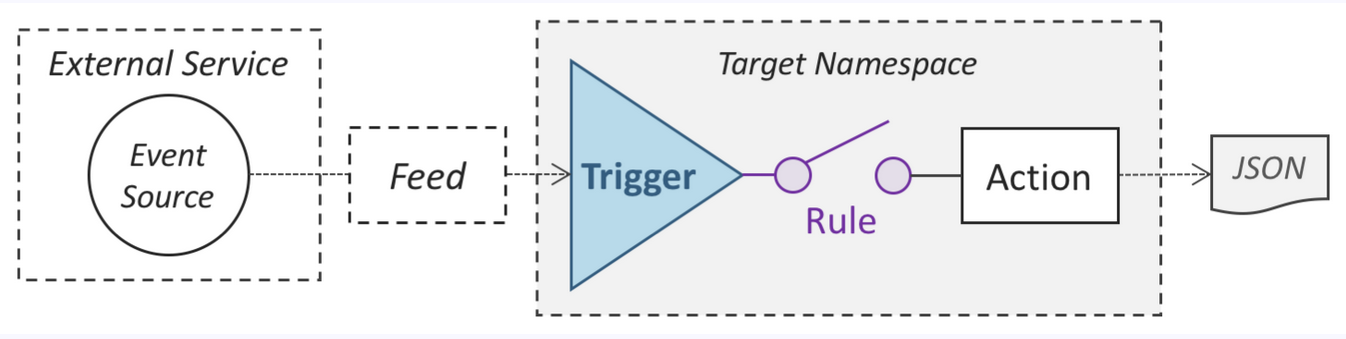


Actions are stateless functions (code snippets) that run on the OpenWhisk platform. Actions encapsulate application logic to be executed in response to events. Actions can be invoked manually by the OpenWhisk REST API, OpenWhisk CLI, simple and user-created APIs or automated via Triggers which we will discuss later.

**Sequence**

Multiple actions, even implemented in different languages, may be composed together to create a longer processing pipeline called a [sequence](https://github.com/apache/openwhisk/blob/master/docs/actions.md#creating-action-sequences). Sequence can be treated as a single action in terms of creation and invocation.

**Triggers and Rules**



What is a trigger? Triggers are named channels for classes or kinds of events sent from Event Sources.

What is a Rule? Rules are used to associate one trigger with one action. After this kind of association is created, each time a trigger event is fired, the action is invoked.

What are Event Sources? These are services that generate events that often indicate changes in data or carry data themselves. Some examples of common Event Sources include:

* Messages arriving on Message Queues
* Changes in Databases
* Changes in Document Stores
* Website or Web Application interactions
* Service APIs being invoked
* IoT Frameworks forwarding device sensor data
* etc.

Why do I need to connect actions to event sources? OpenWhisk is based on an event-driven architecture where most Actions are executed as events happen. The Trigger itself is "fired" with a dictionary of key-value pairs (i.e., the parameters) coming from the Event Source and allows the configuration of optional default values thus helping assure the data is compatible with any associated Actions. Rules allow for the same Trigger to be associated with multiple Actions, as well as allow for specific automation to be enabled or disabled dynamically without destroying the Trigger definition.