**Schema Registry**

Schema Registry provides a serving layer for your metadata. It provides a RESTful interface for storing and retrieving Avro schemas. It stores a versioned history of all schemas, provides multiple compatibility settings and allows evolution of schemas according to the configured compatibility setting. It provides serializers that plug into Kafka clients that handle schema storage and retrieval for Kafka messages that are sent in the Avro format.

**Quickstart**

The following assumes you have Kafka and an instance of the Schema Registry running using the default settings.

# Register a new version of a schema under the subject "Kafka-key"

$ curl -X POST -H "Content-Type: application/vnd.schemaregistry.v1+json" \

--data '{"schema": "{\"type\": \"string\"}"}' \

http://localhost:8081/subjects/Kafka-key/versions

{"id":1}

# Register a new version of a schema under the subject "Kafka-value"

$ curl -X POST -H "Content-Type: application/vnd.schemaregistry.v1+json" \

--data '{"schema": "{\"type\": \"string\"}"}' \

http://localhost:8081/subjects/Kafka-value/versions

{"id":1}

# List all subjects

$ curl -X GET http://localhost:8081/subjects

["Kafka-value","Kafka-key"]

# List all schema versions registered under the subject "Kafka-value"

$ curl -X GET http://localhost:8081/subjects/Kafka-value/versions

[1]

# Fetch a schema by globally unique id 1

$ curl -X GET http://localhost:8081/schemas/ids/1

{"schema":"\"string\""}

# Fetch version 1 of the schema registered under subject "Kafka-value"

$ curl -X GET http://localhost:8081/subjects/Kafka-value/versions/1

{"subject":"Kafka-value","version":1,"id":1,"schema":"\"string\""}

# Fetch the most recently registered schema under subject "Kafka-value"

$ curl -X GET http://localhost:8081/subjects/Kafka-value/versions/latest

{"subject":"Kafka-value","version":1,"id":1,"schema":"\"string\""}

# Delete version 3 of the schema registered under subject "Kafka-value"

$ curl -X DELETE http://localhost:8081/subjects/Kafka-value/versions/3

3

# Delete all versions of the schema registered under subject "Kafka-value"

$ curl -X DELETE http://localhost:8081/subjects/Kafka-value

[1, 2, 3, 4, 5]

# Check whether a schema has been registered under subject "Kafka-key"

$ curl -X POST -H "Content-Type: application/vnd.schemaregistry.v1+json" \

--data '{"schema": "{\"type\": \"string\"}"}' \

http://localhost:8081/subjects/Kafka-key

{"subject":"Kafka-key","version":1,"id":1,"schema":"\"string\""}

# Test compatibility of a schema with the latest schema under subject "Kafka-value"

$ curl -X POST -H "Content-Type: application/vnd.schemaregistry.v1+json" \

--data '{"schema": "{\"type\": \"string\"}"}' \

http://localhost:8081/compatibility/subjects/Kafka-value/versions/latest

{"is\_compatible":true}

# Get top level config

$ curl -X GET http://localhost:8081/config

{"compatibilityLevel":"BACKWARD"}

# Update compatibility requirements globally

$ curl -X PUT -H "Content-Type: application/vnd.schemaregistry.v1+json" \

--data '{"compatibility": "NONE"}' \

http://localhost:8081/config

{"compatibility":"NONE"}

# Update compatibility requirements under the subject "Kafka-value"

$ curl -X PUT -H "Content-Type: application/vnd.schemaregistry.v1+json" \

--data '{"compatibility": "BACKWARD"}' \

http://localhost:8081/config/Kafka-value

{"compatibility":"BACKWARD"}

**Installation**

You can download prebuilt versions of the Kafka REST Proxy as part of the [Confluent Platform](http://confluent.io/downloads/). To install from source, follow the instructions in the Development section.

**Deployment**

The REST interface to schema registry includes a built-in Jetty server. The wrapper scripts bin/schema-registry-start and bin/schema-registry-stop are the recommended method of starting and stopping the service.