

Building Connectors

Conduit connectors can be built in any programming language that supports gRPC. To make it easier to write connectors we provide a [Connector SDK](#) written in Go. Using the SDK is the recommended way of writing a Conduit connector.

Conduit connector template

The easiest way to start implementing your own Conduit connector is by using the [Conduit connector template](#). It contains the basic project structure as well as some additional utilities like GitHub actions and a Makefile.

Find out more about the template and how to use it in the [readme](#).

Supported data types

There are no limitations when it comes to data types a source connector can read from a source. However, if a standalone source connector uses `record.StructuredData` in its key or any part of the payload, then there are certain limitations in the data types it can send to Conduit.

The following data types are supported:

- `bool`
- `int`
- `,int32`
- `,int64`
- `,uint`
- `,uint32`
- `,uint64`
- `float32`
- `,float64`
- `string`
- `[]byte`
- (stored as a string, base64-encoded)
- `map[string]interface{}`
- (a map of strings to any of the values that are supported)
- `[]interface{}`
- (a slice of any value that is supported)

A notable limitation is timestamps, i.e. `time.Time` values are not supported.

One way to support other values is to encode source data to a `[]byte` (e.g. using a JSON encoding) and then store the value as `record.RawData`. [Edit this page](#) [Previous Connector Behavior](#) [Next Output Formats](#)