Getting Started

Table of Contents

To make it easier to see what kubernetes resources are being created as you create functions and flows lets use a separate namespace called `funky`.

```
kubectl create namespace funky
kubectl config set-context `kubectl config current-context` --namespace=funky
```

Now we'll install the runtimes and a couple of connectors into the `funky` namespace

```
funktion install runtime
funktion install connector http4 timer twitter
```

Create a function

```
funktion create fn -f example/hello.js
```

This should have created a new `function`. You can view it via

```
funktion get fn
```

If you wish to keep editing the source code of the function in your IDE and have funktion automatically update the running function use the `-w` argument to watch the source file(s):

```
funktion create fn -f example/hello.js -w
```

To be able to find the URL of the function type:

```
funktion url fn hello
```

which will output the URL to access your function. Or to open it in a browser:

```
funktion url fn hello -o
```

Create a flow

```
funktion create flow timer://bar?period=5000 http://hello/
```

You should now have created a flow. You can view the flow via:

```
funktion get flow
```

To view the output of the flow you can use the following:

```
funktion logs flow timer-bar1
```

You should eventually see the output of the timer events triggering your `hello` function.

To delete the flow:

```
funktion delete flow timer-bar1
```

Use an existing HTTP endpoint

Flows can work with any endpoints whether they are defined via a `function` or not.

e.g. this flow will use an existing endpoint on the internet

```
funktion create flow timer://bar?period=5000 http://ip.jsontest.com/
```

A more complex example

To see a more real world style example check out the blog splitting and counting example using functions and flows