

EXTENSION DEVELOPMENT

PROJECT FLOGO

EXTENSIONS

- ▶ Written in Go lang
- ▶ Interfaces are part of the core library “flogo-lib”
- ▶ Simple Interfaces
- ▶ Stubs generated by flogo CLI tool

EXTENSION POINTS – ACTIVITY

- ▶ An Activity is step in Flow
- ▶ Used to do a specific task in a Flow; examples:
 - ▶ Interact with external systems, i.e. send an message
 - ▶ Log an event
- ▶ Typically executed when its dependencies have been completed
- ▶ Example Activities: Log, REST Invoke, MQTT

EXTENSION POINTS – TRIGGER

- ▶ A Trigger is used to start a Flow
- ▶ Typically event based
- ▶ Examples: Timer, REST, MQTT

ACTIVITY

- ▶ Use “flogo” CLI to create stub
- ▶ Configure metadata
- ▶ Implement “Activity” interface
- ▶ Publish to GIT repository

ACTIVITY – CREATE STUB

- ▶ Setup tools
 - ▶ make sure Go is installed
 - ▶ install gb tool (used for building Go project)
 - ▶ go get [github.com/constabulary/gb/...](https://github.com/constabulary/gb/)
 - ▶ install flogo CLI
 - ▶ go get [github.com/TIBCOSoftware/flogo-cli/...](https://github.com/TIBCOSoftware/flogo-cli/)
- ▶ Create stub
 - ▶ flogo activity create myactivity
- ▶ Start editing
 - ▶ cd myactivity/src/rt

ACTIVITY – METADATA

- ▶ Describes the Activity
- ▶ Inputs/Outputs
 - ▶ Attribute: name, value, required
- ▶ located in `activity.json`, `activity_metadata.go`
 - ▶ currently needs to be synchronized manually
 - ▶ soon should have 'flogo activity sync' command
- ▶ Used by Web to generate configuration UIs

ACTIVITY – METADATA EXAMPLE

```
{
  "name": "tibco-log",
  "version": "0.0.1",
  "title": "Log Activity",
  "description": "Simple Log Activity",
  "inputs": [
    {
      "name": "message",
      "type": "string",
      "value": ""
    },
    {
      "name": "flowInfo",
      "type": "boolean",
      "value": "false"
    },
    {
      "name": "addToFlow",
      "type": "boolean",
      "value": "false"
    }
  ],
  "outputs": [
    {
      "name": "message",
      "type": "string"
    }
  ]
}
```

ACTIVITY – INTERFACE

```
// Activity is an interface for defining a custom Task Execution

type Activity interface {

    // Eval is called when an Activity is being evaluated. Returning true indicates
    // that the task is done.

    Eval(context Context) (done bool, evalError *Error)

    // ActivityMetadata returns the metadata of the activity

    Metadata() *Metadata
}
```

ACTIVITY – SUPPORT INTERFACE

```
// Context describes the execution context for an Activity.
// It provides access to attributes, task and Flow information.
type Context interface {

    // FlowInstanceID returns the ID of the Flow Instance
    FlowInstanceID() string

    // FlowName returns the name of the Flow
    FlowName() string

    // TaskName returns the name of the Task the Activity is currently executing
    TaskName() string

    // GetInput gets the value of the specified input attribute
    GetInput(name string) interface{}

    // SetOutput sets the value of the specified output attribute
    SetOutput(name string, value interface{})
}
```

TRIGGER

- ▶ Use “flogo” CLI to create stub
- ▶ Configure metadata
- ▶ Implement “Trigger” interface
- ▶ Publish to GIT repository

TRIGGER - METADATA

- ▶ Describes the TRIGGER
- ▶ Settings - Global Trigger configuration
- ▶ Outputs - Outputs created by the Trigger
- ▶ Endpoint - Describes a particular endpoint
- ▶ located in `trigger.json`, `trigger_metadata.go`
- ▶ Used by Web to generate configuration UIs

TRIGGER – METADATA EXAMPLE

```
{
  "name": "tibco-coap",
  "version": "0.0.1",
  "title": "CoAP Trigger",
  "description": "Simple CoAP Trigger",
  "settings": [
    {
      "name": "port", "type": "integer", "required": true
    }
  ],
  "outputs": [
    { "name": "queryParams", "type": "params" },
    { "name": "payload", "type": "string" }
  ],
  "endpoint": {
    "settings": [
      { "name": "method", "type": "string", "required" : true },
      { "name": "path", "type": "string", "required" : true },
      { "name": "autoIdReply", "type": "boolean" }
    ]
  }
}
```

TRIGGER - INTERFACE

```
// Trigger is object that triggers/starts flow instances and
// is managed by an engine

type Trigger interface {

    util.Managed

    // TriggerMetadata returns the metadata of the trigger
    Metadata() *Metadata

    // Init sets up the trigger, it is called before Start()
    Init(starter flowinst.Starter, config *Config)
}
```

TRIGGER - CREATE STUB

- ▶ Make sure flogo CLI is installed
- ▶ Create stub
 - ▶ `flogo trigger create my trigger`
- ▶ Start editing
 - ▶ `cd mytrigger/src/rt`

TRIGGER – SUPPORT INTERFACES

```
// Managed is an interface that is implemented by an object that needs to be
// managed via start/stop
type Managed interface {

    // Start starts the managed object
    Start() error

    // Stop stops the managed object
    Stop()
}

// Starter interface is used to start flow instances, used by Triggers
// to start instances
type Starter interface {

    // StartFlowInstance starts a flow instance using the provided information
    StartFlowInstance(flowURI string, startAttrs []*data.Attribute, replyHandler ReplyHandler,
        execOptions *ExecOptions) (instanceID string, startError error)
}

// ReplyHandler is used to reply back to whoever started the flow instance
type ReplyHandler interface {

    // Reply is used to reply with the results of the instance execution
    Reply(replyData map[string]string)
}
```

FLOGO RESOURCES

- ▶ Flogo CLI

- ▶ <https://github.com/TIBCOSoftware/flogo-cli>

- ▶ Activity and Trigger Examples

- ▶ <https://github.com/TIBCOSoftware/flogo-contrib>

- ▶ Flogo Core Library

- ▶ <https://github.com/TIBCOSoftware/flogo-lib>

- ▶ Extension interfaces located under core/ext

INTERNAL TECHNICAL PREVIEW AVAILABLE

- ▶ Fill up Formvine request for Internal Technical Preview
 - ▶ <https://formvine.tibco.com/fv/r/3555430693/5446>

COME TALK TO US

The screenshot shows the Tibbr social network interface for the 'Project Flogo' group. At the top, there is a navigation bar with the Tibbr logo, 'Home', and a user profile 'Rajeev Kozhikkattuthodi'. A search bar is located on the right. Below the navigation bar, the 'Project Flogo' group header is visible, featuring a menu icon, the group name, and options for 'Alerts', 'Actions', 'Manage', and 'Owner'. A horizontal menu below the header includes 'Wall Posts' (highlighted in blue), 'People', 'Links', 'Files', and 'Trends'. The main content area is divided into two columns. The left column contains a post creation section with options for 'Post', 'Poll', 'Event', and 'Question', and a text input field with the placeholder 'What's going on?'. Below this is a section for 'All' activity, showing a post by 'Kevin Bohan' titled 'Internal Webinar - Building Flogo Extensions' from June 28, 2016. The right column features a promotional banner for 'Project Flogo' with the text 'Ultra-light Open Source integration for IoT. Coming soon to a device near you!' and a link to 'Products > Project Flogo'. Below the banner is a 'Followers (66)' section displaying a row of user profile pictures, with a '6' icon indicating the total number of followers and a 'See All' link.

Tibbr: <https://tibco.tibbr.com/tibbr/#!/subjects/48414>