Core Working Group Meeting Notes (7-Feb-2019)

Attendees:

Old Business

- Plugins
  - Any possible updates from Beau's (Intel) contacts at Microsoft?
    - *Still waiting AFAIK*
- Modules
  - Decided on multiple-repo approach
  - Some items pending DevOps call
    - *If we haven't already finalized repo names by this meeting, do it here.*
      - *Repos have been created*
      - *Still need verify jobs (We may need to help DevOps)*
    - *Proposed format* github.com/<organization>/go-mod-<capability>
    - *Initial repos to include*
      - github.com/edgexfoundry/go-mod-registry
      - github.com/edgexfoundry/go-mod-messaging
      - github.com/edgexfoundry/go-mod-core-contracts
    - *Discuss if necessary*

New Business

- Resolution on PR 990 (Dockerfiles)
  - https://github.com/edgexfoundry/edgex-go/pull/990
- Float representations (via Tony Espy)
  - Seeking input from wider audience to build concensus
  - Options include
    - - change the default encoding of floats to use the C print floating point format { printf("%1.8e", 3.14159562) } which results in a string value like this: "3.14159265+e00"
      (note - for float64, the precision needs to be increased to 10+, which makes the logic a bit trickier)
    - - change the default encoding of floats to use some other literal representation of the actual binary value such a binary literal format as

mentioned here (note, we'd have to implement this as the link points to a proposal for Go)
        - keep the existing base64 encoding of floats and add an optional encoding attribute to the value property and value descriptor objects. This would allow alternate encodings to be added to handle floating point numbers
    - Preference is to keep base64
    - Possible impacts on application/export services
- Correlation ID demo if time permits

Edgex-go v1.0.0

Registry v1.0.0

funcA

Edgex-go v2.0.0

Registry v1.1.0

funcA

funcB

sdk-go → mod/core-contracts ← edgex-go

1.0	1.0	1.0 (Edinburgh)
2.0	1.1	1.5

Sdk-go → 2.0.0 (Fuji)

Mod/core-contracts 1.5.1

Mod/go-registry 1.3.2

Mod/go-messaging 2.0.0

App-func-sdk-go → x.0.0 (??)

Mod/core-contracts 1.5.1

Mod/go-registry 1.3.2

Mod/go-messaging 2.0.0

Released artifacts

Core-command (v2.0.0)

Core-data

Device-mqtt

Snap (v2.0.0)

Registry master SHA. ←-- Edgex-go

Edinburgh 1.x.x. ←- Edinburgh

←- Fuji

Summary

- We need to look at how 6 month monolithic releases affects the more flexible versioning capabilities possible with modules
- Distinguish actual released artifacts from code dependencies
- Do modules need their own release branches (Delhi, Edinburgh)
    - If no major version, breaking change, why do we need a release branch?
- Major version (Edinburgh) could reference modules versioned on their own via go.mod
- Do we publish SDKs independently? Do they need to be included in a major rev? Do we proclaim SDKs compatible with a major EdgeX release?
- TSC Agenda Item related to versioning, LTS, what does a release look like.