# Mule Project and Domain Project

# Domain Project

It is used to share resources between mule application

Why use it?

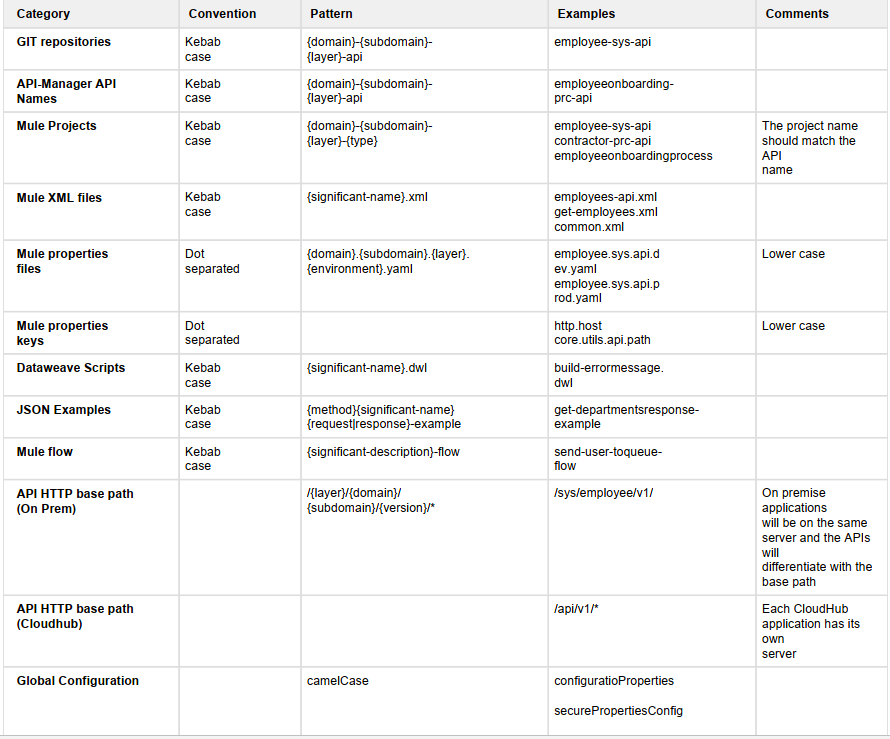
* Mule supports the ability to define selected connectors as common resources and expose them to all applications deployed under a same domain. These resources are known as shared resources, to host these you must create a Mule Domain Project
* **It is mainly used in customer hosted environment**

Why not supported in CLoudHub

* Mule project is a MuleSoft project which have all the dependencies defined in itself, it doesn’t             share its resources or use shared resources.
* In cloud hub we get application isolation so we don’t share resources.

In customer hosted you deploy multiple application in the same runtime. i.e one runtime can have multiple application

# Naming Conventions



# Logging

* Logging helps you to monitor and troubleshoot your application and server.
* Every application has a log4j file associated with it
* Mule application logging is done using log4j file
* It uses appenders to write a log. Appenders specify how you want to write a log. Example if you want to write a log in a text file, or write a log in a db or write a log in a separate file. You can customize it

**There are two types of log in Mule**

* + Application log
  + Audit log
* .
* **Audit logging**
  + In access management there is a separate log
  + It stores the log information for 6 years
  + It stores all the activity information you are performing using anypoint platform. Example when you deploy an application, download an application, restart servers
  + **Why?**
    - Mainly used to identify access violations with anypoint platform

**Application logging**

Log4j file is for **application logging**. Anything and everything you do in application are logged using Application logger

There are different levels of logging. They are :

* Info (default)
* Debug
* Error
* Trace
* Warn

**Synchronous/ asynchronous logging**

* By **default logging** is asynchronous. It is executed in a separate thread
* But when you have any **error type** of logging it synchronous. The thread will wait for the entire flow to complete

**System Log/ Application log**

* With in application also there are two types of logging
* 1. System Log
* It contains all the log information related to runtime, domain project information
* Example when the application was started and stopped
* By default is cannot be externalized. You need to configure it to externalize the log
* 2. Application Log
* It stores information about the application
* Example log in log4j file, seen in console

**Correlation ID**

* When you perform logging it is important to maintain the correlation ID
* **Why? Correlation ID is used to uniquely track a request from one system to another**
* Trace end to end request

**Externalization of logs**

* Log files can be externalized using system like SPLUNK, ELK
* Why?
* If you go to Anypoint platform Runtime Manager you have a LOGS control to see log. In this the log is kept for 30 days by default and the size is 100 MB. If any of the criteria exceeds then it start truncating your log
* To retain the log for more than that we externalize the log

SPLUNK and ELK are log management system that are specialized in log

* It can be done in different ways
* 1. Use cloud hub API for log management
  + In this call the API and download the log and store in the SPLUNK
* 2. Anypoint CLI
  + It is a command line interface for log externalization
* 3. Log4j Appender
  + As soon as you deploy your application the log4j file is replaced by cloud hub log
  + So whatever is written in log4j file is overwritten by cloud hub log4j
  + To see the studio log we need to raise a ticket so that cloud hub support will create an enable/ disable feature in your account. Then you can use the feature
  + After enabling it, if any logs are lost it is the responsibility of a customers to maintain the logs

# Semantic Versioning

* Semantic Versioning is a industry standard which is followed to do versioning of software system.
* Format: **MAJOR.MINOR.PATCH**
* Example: 4.3.0
* It is having four components: Major release number, minor release number, patch

release number and an optional service pack number.

* We should make all the changes backward compatible
* If there are backward incompatible change. Then change **major version. Example** Mule Major version changes from 3 to 4 because there are several functionalities introduced which were not compatible
* If there are backward compatible change. Then change **minor version**
* Bug fixer are **patch version**
* Versioning should be done only through url – **add only the major version**
  + **Example:** http:// localhost:8081/api/**v1**/employees

# Control Flows

The main task of Flow Control component is to take the input Mule event and route it to one or more separate sequences of components. It is basically routing the input Mule event to other sequence(s) of components. Therefore, it is also called as Routers. Choice and Scatter-Gather routers are the most used routers under Flow Control component.

1. Choice Router

The Choice router dynamically routes messages through a flow according to a set of DataWeave expressions that evaluate message content.

The Choice router enables [content-based routing](http://www.enterpriseintegrationpatterns.com/ContentBasedRouter.html), which is a common way to introduce routing logic based on content of the current message.

1. First Successful

The First Successful router iterates through a list of configured processing routes until one of the routes executes successfully. If any processing route fails execution (throws an error), the router executes the next configured route. If none of the configured routes execute successfully, the First Successful router throws an error.

1. Round Robin

A round robin is a router which iterates through two or more routes in order. During execution it routes to only one route. It maintain the track of the previously selected route and thus avoid the simultaneous execution of the same route .

1. Scatter Gather

The routing message processor Scatter-Gather sends a request message to multiple targets concurrently. It collects the responses from all routes, and aggregates them into a single message.

Error in scatter gather:  Mule: COMPOSITE\_ERROR

This error will be thrown by the S-G component only after every route either fails or completes

To handle this error type, a try scope can be used in each route of Scatter-Gather component. If the error is successfully handled by try scope, then the route will be able to generate a Mule event, for sure.

# Dataweave

DataWeave is basically a MuleSoft expression language. It is mainly used for accessing and transforming the data received through a Mule application. Mule runtime is responsible for running the script and expressions in our Mule application,

1. Data Transformation

By default dataweave code expression is placed inline in the mule configuration file

* 1. Reusing dw code: ${file::filename} syntax
  2. Create multiple transformation: multiple transformations using one single transform component

1. Simple Literals
2. Objects: Represented as the collection of key value pair
3. Arrays: you can use various content type: Represented as the sequence of comma separated values

Two types of dataweave errors:

* **Scripting errors:** A problem with the syntax
* **Formatting errors:** A problem with how the transformation from one format to another is written

If you get an error transform the input to **application/dw**-If the transform is successful then the error is like formatting error

* Internally dataweave uses

**Dataweave Operators**

1. Map operators
2. mapObject
3. FlatMap
4. Pluck

**Dataweave playground: sometime the studio get hang for complex dw so you ma use it**

# Exception handling:

1. Global Handler
2. Try Catch
3. Flow Level
   1. Error continue: used for committing previous statement
   2. Error propagate: rollback previous steps

There are broadly two ways to define the errors in Mulesoft as below.

* **Message Level Errors –** Message level error occurs when mule message is involved. e.g. error occurred when calling a database or error occurred (http 401 unauthorized or 503 service unavailable etc.) while calling an external API

**System Level Error –** System level error occurs when there is no mule message involved. E.g. connectivity issue to a JMS provider, connectivity issue to a Database. You cannot handle system level errors.