# Mule Projects in Anypoint Studio

* Use Maven
* Include a .gitignore file
* Add APIKit components getting the RAML from Design Center

## Project Files Structure

* Create a separate xml for the global elements (e.g. configuration elements)
* Create a separate xml for each use-case or resource-implementation
* Create separate XMLs for common structures/logic
* Create different packages for the resources (dataweave, WSDLs, examples, etc.)

## XML identation and formatting

* Define a line width in your Anypoint Studio XML editor preferences, e.g. 140
* Indent all your xml (mule XMLs, pom.xml, log4j2.xml, etc.) files before committing to the source code repository

## Properties per environment

* Use property placeholders to externalize properties. Use a placeholder in the location-name of the file to identify the environment.

# Unit and Integration Testing

* Create significative Unit tests using MUnit.
* Create significative Integration tests using MUnit.

# Mule Deploy Maven Plugin

Include the Mule Maven Deploy Plugin in each application

# Debugging & Troubleshooting

Use Anypoint Studio Debugger to debug your Mule Applications.

## Wire Logging

To enable a more detailed logging to see all the HTTP requests and responses, configure the following loggers in src/main/resources/log4j2.xml

<AsyncLogger name=”org.mule.module.http.internal.HttpMessageLogger” level=”DEBUG” />

<AsyncLogger name=”com.ning.http” level=”DEBUG” />

# Mule Expression Language(MEL)

MEL allows dynamically defining addresses for http component. For this purpose, in the host field of http connector it is enough to use expressions of the following form:

* host="#[xpath3('/request/host')]" – in the case if different data come to the input of the http connector in the xml format.
* host=”#[payload]” – in the case if the address needs to be withdrawn from payload.
* host=”#[flowVars.host]” - in the case if the address needs to be withdrawn from flow variables.  
  Mule expressions can be used almost in all text-fields of the Mule components.

# Queues

Mule allows working with different queues, which settings have a lot of common (name of the queue, operation – read/record), but with its specific features:

* VM is an in-memory queue, support transactions, but it can be used only for the message exchange among threads of an application, and not among several applications.
* AnypointMQ is a queue, provided by Mulesoft for the commercial usage and integrated into the Anypoint platform. It does not require a specific server for installation, but it does not support transactions.
* ActiveMQ is a popular open-source solution, which supports JMS. It supports transactions and message exchange among applications but it requires a specific server for installation.

# Profiling service

A standard java/oracle utility jvisualvm.exe can be used for profiling. It can be found by the following way Windows C:\Program Files\Java\jdk1.8.0\_65\bin\jvisualvm.exe (in the bin catalogue of the installed jdk). This utility allows tracking resources which are used by java applications). The application can be loaded and resources spends can be tracked under the concrete loadings.

# Xml formatting

An extra component should be created for a smart/convenient text displaying in xml logs . When using this component it should be taking into consideration that it can change xml by adding to it spaces/inputs etc. that, as a result, might make xml invalid for web-services and various utilities for working with xml, so before logging one should save the source xml, and after logging one should return the previously saved xml.

Sometimes, in xml request there are symbols of &lt type instead <> brackets, for correct work with xml, it is necessary to do the operation of data symbols change to <>. In this regard an additional class may be used, which will change escape symbols.