Building and Installing:

Prerequisites:

OMI Script Provider depends on the following software. Be sure these are installed on your system before building.

- GNU make
- Native C and C++ compiler
- OpenSSL headers and libraries
- PAM headers and libraries
- python2.7-dev

OMI and OMIScriptProvider build and install. This will build, link, modify conf files, and install libraries and binaries. In the build-omi-script-provider path:

```
> cd omi/Unix
> ./configure --dev
> make
> make install
> cd ../../scriptprovider
> make
> sudo make install
```

Developing a Python Provider:

This is a very quick overview for how to develop an OMI Provider in Python. It shows the minimum steps for building a simple Python provider. More details will be included in the next section.

Defining the “schema.mof”:

First, we define the class schema shown in the “schema.mof” file below.

```
class XYZ_Frog
{
    [Key] String Name;
    Uint32 Weight;
    String Color;
};
```

Generate the Provider Sources:

Next, we generate the provider sources using the command below:

```
> omigen_py schema.mof XYZ_Frog
Creating schema.py
Creating mi_main.py
```

The omigen_py tool will generate the files schema.py and mi_main.py.

schema.py:

```
# @migen@
```

```
##==============================================================================
=
##
## WARNING: THIS FILE WAS AUTOMATICALLY GENERATED. PLEASE DO NOT EDIT.
##
##==============================================================================
=
import omi
from omi import *

##==============================================================================
=
##
## Qualifier declarations
##
##==============================================================================
=
qualifierDecls = [
    ]

##==============================================================================
=
##
## XYZ_Frog
##
##==============================================================================
=
XYZ_Frog_quals = [
    ]

XYZ_Frog_Name_quals = [
    ]

XYZ_Frog_Name_prop = MI_PropertyDecl (
    MI_FLAG_PROPERTY | MI_FLAG_KEY, # flags
    'Name', # name
    XYZ_Frog_Name_quals, # qualifiers
    MI_STRING, # type
    None, # className
    'XYZ_Frog', # origin
    'XYZ_Frog', # propagator
    None # value
    )

XYZ_Frog_Weight_quals = [
    ]

XYZ_Frog_Weight_prop = MI_PropertyDecl (
    MI_FLAG_PROPERTY, # flags
    'Weight', # name
    XYZ_Frog_Weight_quals, # qualifiers
    MI_UINT32, # type
    None, # className
    'XYZ_Frog', # origin
    'XYZ_Frog', # propagator
    None # value
    )

XYZ_Frog_Color_quals = [
    ]

XYZ_Frog_Color_prop = MI_PropertyDecl (
    MI_FLAG_PROPERTY, # flags
    'Color', # name
    XYZ_Frog_Color_quals, # qualifiers
    MI_STRING, # type
    None, # className
    'XYZ_Frog', # origin
    'XYZ_Frog', # propagator
    None # value
    )

XYZ_Frog_properties = [
    XYZ_Frog_Name_prop,
    XYZ_Frog_Weight_prop,
    XYZ_Frog_Color_prop,
    ]

XYZ_Frog_methods = [
    ]

XYZ_Frog_functions = MI_FunctionTable (
    'XYZ_Frog_Load',
    'XYZ_Frog_Unload',
    'XYZ_Frog_GetInstance',
    'XYZ_Frog_EnumerateInstances',
    'XYZ_Frog_CreateInstance',
    'XYZ_Frog_ModifyInstance',
    'XYZ_Frog_DeleteInstance',
    None,
    None,
    None,
    None,
    None,
    None,
    None
    )

XYZ_Frog_class = MI_ClassDecl (
    MI_FLAG_CLASS, # flags
    'XYZ_Frog', # name
    XYZ_Frog_quals, # qualifiers
    XYZ_Frog_properties, # properties
    None, # superclass
    XYZ_Frog_methods, # method
    XYZ_Frog_functions, # FunctionTable
    None # owningclass
    )

classDecls = [
    XYZ_Frog_class,
    ]

schema = MI_SchemaDecl (
    qualifierDecls,
    classDecls
    )
```

mi_main.py:

```
import omi
from omi import *

import schema
```

```
def Load (module, context):
    context.PostResult (MI_RESULT_OK)


def Unload (module, context):
    context.PostResult (MI_RESULT_OK)


def XYZ_Frog_Load (
    module, context):
    context.PostResult (MI_RESULT_OK)


def XYZ_Frog_Unload (
    context):
    context.PostResult (MI_RESULT_OK)


def XYZ_Frog_EnumerateInstances (
    context, nameSpace, className, propertySet, keysOnly):
    context.PostResult (MI_RESULT_NOT_SUPPORTED)


def XYZ_Frog_GetInstance (
    context, nameSpace, className, instanceName, propertySet):
    context.PostResult (MI_RESULT_NOT_SUPPORTED)


def XYZ_Frog_CreateInstance (
    context, nameSpace, className, instance):
    context.PostResult (MI_RESULT_NOT_SUPPORTED)


def XYZ_Frog_ModifyInstance (
    context, nameSpace, className, instance, propertySet):
    context.PostResult (MI_RESULT_NOT_SUPPORTED)


def XYZ_Frog_DeleteInstance (
    context, nameSpace, className, instanceName):
    context.PostResult (MI_RESULT_NOT_SUPPORTED)


def mi_main ():
    r = MI_Module (schema.schema, Load, Unload)
    return r
```

Implementing the “EnumerateInstances” Method:

Make the following changes to the XYZ_Frog_EnumerateInstances method in mi_main.py:

```
def XYZ_Frog_EnumerateInstances (
    context, nameSpace, className, propertySet, keysOnly):
    frog1 = context.NewInstance ('XYZ_Frog')
    frog1.SetValue ('Name', MI_String ('Fred'))
    frog1.SetValue ('Weight', MI_Uint32 (55))
    frog1.SetValue ('Color', MI_String ('Green'))
    context.PostInstance (frog1)
    frog2 = context.NewInstance ('XYZ_Frog')
    frog2.SetValue ('Name', MI_String ('Sam'))
    frog2.SetValue ('Weight', MI_Uint32 (65))
    frog2.SetValue ('Color', MI_String ('Blue'))
    context.PostInstance (frog2)
    context.PostResult (MI_RESULT_NOT_SUPPORTED)
    context.PostResult (MI_RESULT_OK)
```

Registering the Provider:

Next, register the provider as follows:

```
> omireg --Python XYZ_Frog
Created /opt/omi/etc/omiregister/root-cimv2/XYZ_Frog.reg
```

Testing the Provider:

To test the provider, send an enumerate request to the provider as shown:

```
> omicli ei root/cimv2 XYZ_Frog
instance of XYZ_Frog
{
    [Key] Name=Fred
    Weight=55
    Color=Green
}
instance of XYZ_Frog
{
    [Key] Name=Sam
    Weight=65
    Color=Blue
}
```

Going Further:

This provides a brief overview of the provider development process. The next section contains a deeper discussion.