ATT-YANG DESIGN STUDIO
User Guide

6/28/2015

Version 1.5

Eclipse Public License -v 1.0

THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT.

1. DEFINITIONS

"Contribution" means:

a) in the case of the initial Contributor, the initial code and documentation distributed under this Agreement, and

b) in the case of each subsequent Contributor:

i) changes to the Program, and

ii) additions to the Program;

where such changes and/or additions to the Program originate from and are distributed by that particular Contributor. A Contribution 'originates' from a Contributor if it was added to the Program by such Contributor itself or anyone acting on such Contributor's behalf. Contributions do not include additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program.

"Contributor" means any person or entity that distributes the Program.

"Licensed Patents " mean patent claims licensable by a Contributor which are necessarily infringed by the use or sale of its Contribution alone or when combined with the Program.

"Program" means the Contributions distributed in accordance with this Agreement.

"Recipient" means anyone who receives the Program under this Agreement, including all Contributors.

2. GRANT OF RIGHTS

a) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form.

b) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents. The patent license shall not apply to any other combinations which include the Contribution. No hardware per se is licensed hereunder.

c) Recipient understands that although each Contributor grants the licenses to its Contributions set forth herein, no assurances are provided by any Contributor that the Program does not infringe the patent or other intellectual property rights of any other entity. Each Contributor disclaims any liability to Recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise. As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole responsibility to secure any other intellectual property rights needed, if any. For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program.

d) Each Contributor represents that to its knowledge it has sufficient copyright rights in its Contribution, if any, to grant the copyright license set forth in this Agreement.

3. REQUIREMENTS

A Contributor may choose to distribute the Program in object code form under its own license agreement, provided that:

a) it complies with the terms and conditions of this Agreement; and

b) its license agreement:

i) effectively disclaims on behalf of all Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose;

ii) effectively excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits;

iii) states that any provisions which differ from this Agreement are offered by that Contributor alone and not by any other party; and

iv) states that source code for the Program is available from such Contributor, and informs licensees how to obtain it in a reasonable manner on or through a medium customarily used for software exchange.

When the Program is made available in source code form:

a) it must be made available under this Agreement; and

b) a copy of this Agreement must be included with each copy of the Program.

Contributors may not remove or alter any copyright notices contained within the Program.

Each Contributor must identify itself as the originator of its Contribution, if any, in a manner that reasonably allows subsequent Recipients to identify the originator of the Contribution.

4. COMMERCIAL DISTRIBUTION

Commercial distributors of software may accept certain responsibilities with respect to end users, business partners and the like. While this license is intended to facilitate the commercial use of the Program, the Contributor who includes the Program in a commercial product offering should do so in a manner which does not create potential liability for other Contributors. Therefore, if a Contributor includes the Program in a commercial product offering, such Contributor ("Commercial Contributor") hereby agrees to defend and indemnify every other Contributor ("Indemnified Contributor") against any losses, damages and costs (collectively "Losses") arising from claims, lawsuits and other legal actions brought by a third party against the Indemnified Contributor to the extent caused by the acts or omissions of such Commercial Contributor in connection with its distribution of the Program in a commercial product offering. The obligations in this section do not apply to any claims or Losses relating to any actual or alleged intellectual property infringement. In order to qualify, an Indemnified Contributor must: a) promptly notify the Commercial Contributor in writing of such claim, and b) allow the Commercial Contributor to control, and cooperate with the Commercial Contributor in, the defense and any related settlement negotiations. The Indemnified Contributor may participate in any such claim at its own expense.

For example, a Contributor might include the Program in a commercial product offering, Product X. That Contributor is then a Commercial Contributor. If that Commercial Contributor then makes performance claims, or offers warranties related to Product X, those performance claims and warranties are such Commercial Contributor's responsibility alone. Under this section, the Commercial Contributor would have to defend claims against the other Contributors related to those performance claims and warranties, and if a court requires any other Contributor to pay any damages as a result, the Commercial Contributor must pay those damages.

5. NO WARRANTY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the appropriateness of using and distributing the Program and assumes all risks associated with its exercise of rights under this Agreement , including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and unavailability or interruption of operations.

6. DISCLAIMER OF LIABILITY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, NEITHER RECIPIENT NOR ANY CONTRIBUTORS SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE PROGRAM OR THE EXERCISE OF ANY RIGHTS GRANTED HEREUNDER, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. GENERAL

If any provision of this Agreement is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this Agreement, and without further action by the parties hereto, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.

If Recipient institutes patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Program itself (excluding combinations of the Program with other software or hardware) infringes such Recipient's patent(s), then such Recipient's rights granted under Section 2(b) shall terminate as of the date such litigation is filed.

All Recipient's rights under this Agreement shall terminate if it fails to comply with any of the material terms or conditions of this Agreement and does not cure such failure in a reasonable period of time after becoming aware of such noncompliance. If all Recipient's rights under this Agreement terminate, Recipient agrees to cease use and distribution of the Program as soon as reasonably practicable. However, Recipient's obligations under this Agreement and any licenses granted by Recipient relating to the Program shall continue and survive.

Everyone is permitted to copy and distribute copies of this Agreement, but in order to avoid inconsistency the Agreement is copyrighted and may only be modified in the following manner. The Agreement Steward reserves the right to publish new versions (including revisions) of this Agreement from time to time. No one other than the Agreement Steward has the right to modify this Agreement. The Eclipse Foundation is the initial Agreement Steward. The Eclipse Foundation may assign the responsibility to serve as the Agreement Steward to a suitable separate entity. Each new version of the Agreement will be given a distinguishing version number. The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received. In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version. Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppel or otherwise. All rights in the Program not expressly granted under this Agreement are reserved.

This Agreement is governed by the laws of the State of New York and the intellectual property laws of the United States of America. No party to this Agreement will bring a legal action under this Agreement more than one year after the cause of action arose. Each party waives its rights to a jury trial in any resulting litigation.

ATT-YANG DESIGN STUDIO User Guide

Create a New YANG Project:

1. File->New->Other
2. Search for Yang
3. Select New Yang Project (under Yang Wizard)
4. Enter a Project Name
5. Select Finish
6. [If Finish is not selectable, you most likely have a problem with the Project Name you entered]
7. If successful, you will see the new Project in the Package Explorer Window with a "Yang-Modules" folder under it

Create a New YANG Module (in this example illustration it is created under the "Yang-Modules" folder):

1. Right-click on Yang-Modules folder name
2. New->Other
3. Search for Yang
4. Select New Yang File (under Yang Wizard)
5. [Alternatively you can also create via File->New->Other->(Yang Wizard) New Yang File]
6. Enter a New Yang Module Name
7. Select Finish
8. [If Finish is not selectable, you most likely have a problem with the Module Name you entered]
9. If successful, you will see the new Module in the Package Explorer Window under the folder you selected for its creation

WYSWYG Editor (using XText):

1. Click on any YANG Module in the Package Explorer Window
2. The model should show up in the Presentation Space with its YANG elements recognized (e.g., highlighted appropriately)
3. If there is a syntax error that is recognized by the XText grammar, it will be visually indicated
4. For context-specific help, position the cursor within the context and use <CONTROL><SPACE> to get the appropriate help
5. For Hover-Over help, position the cursor over context and see visual help based
6. Several other WYSWYG features are available, including:
    a. Auto-matching on quotes, brackets, etc.
    b. Presentation of all groupings for inclusion in "uses"
        - Position cursor after uses and use <CONTROL><SPACE> to see all available groupings
    c.

Note 1: When you bring up a YANG file for the very first time, you will need to associate it with XText when presented with the option to do so.

Note 2: Not that the WYSIWYG Editor does find a majority of the syntax errors, however not all YANG syntax is coded into it. For a deeper check, please run the Check Yang Syntax option below.

Note 3: The WYSWYG editor is not very good at following the PATH to find imported modules, so you might need to copy the imported modules into the local workspace directory if you would like to resolve dependency issues that this editor highlights. Alternatively, run the Check Yang Syntax option below after setting the YANG_MODPATH to validate imports.

Selectable features supported by ATT-Yang:

1. Check Yang Syntax
   - The results are displayed in the Eclipse Console and Problems Window
2. Create YIN/XML File
   - The results are displayed in a new presentation window and saved to file
3. Create Tree File (with selectable tree depth
   - The results are displayed in a new presentation window and saved to file
4. Create UML file
   - The results are displayed in a new presentation window and saved to file
5. Create PNG from UML
   - The UML output is displayed in a new presentation window
   - The PNG is displayed in a new pop-up window created by the image viewer and saved to the <img> directory under the Eclipse Workspace directory for the Yang Model
6. Create XSD file
   - The results are displayed in a new presentation window and saved to file
7. Create JS Tree
   - The results are displayed in a new presentation window and saved to file.
   - The user can interact with the JSTree output by expanding or contracting the depth of the YANG model namespace
8. Create Skeleton XML File
   - The results are displayed in a new presentation window and saved to file
   - The user can manipulate the XML output file, such as adding instance data, to illustrate a sample instantiation of the YANG model
9. Create DSDL File
   - The results are displayed in a new presentation window and saved to file
10. Create XSLT File
    - The results are displayed in a new presentation window and saved to file
11. About ATT-YANG IDE (aka YANG Design Studio)
    - A display of developer, version and dependency information

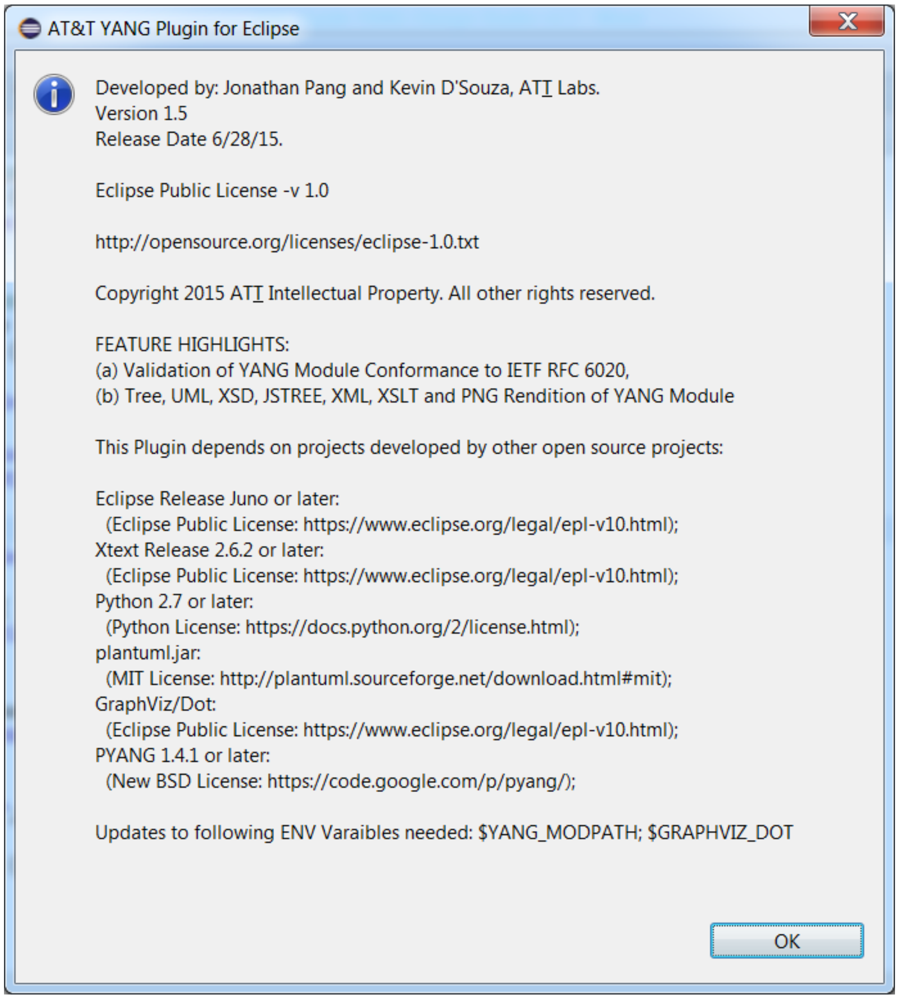
All File Output options give the user the choice of where to save the file, with the default being the Eclipse Workspace directory.

NOTE: When you select any of the above options, it might seem that Eclipse has hung. Please be patient as it takes a several seconds to complete the task.

Known Issues:

- Automatic syntax validation using the WYSIWYG Editor sometimes marks valid statements with errors. The selectable ATT-Yang->Check Yang Syntax is more precise. A fix will require enhancement to the XText module.
- Caching of path names to dependency commands is not implemented, so execution times of the ATT-Yang selectable options is dependent on the speed at which the machine can find the dependencies
- Some of the rendering options might not work with all YANG models as the tool relies on pyang for rendering YANG models. A fix might require developing an alternate rending method.

About ATT-YANG IDE Screen Shot


AT&T YANG Plugin for Eclipse

Developed by: Jonathan Pang and Kevin D'Souza, ATT Labs.
Version 1.5
Release Date 6/28/15.

Eclipse Public License -v 1.0

http://opensource.org/licenses/eclipse-1.0.txt

Copyright 2015 ATT Intellectual Property. All other rights reserved.

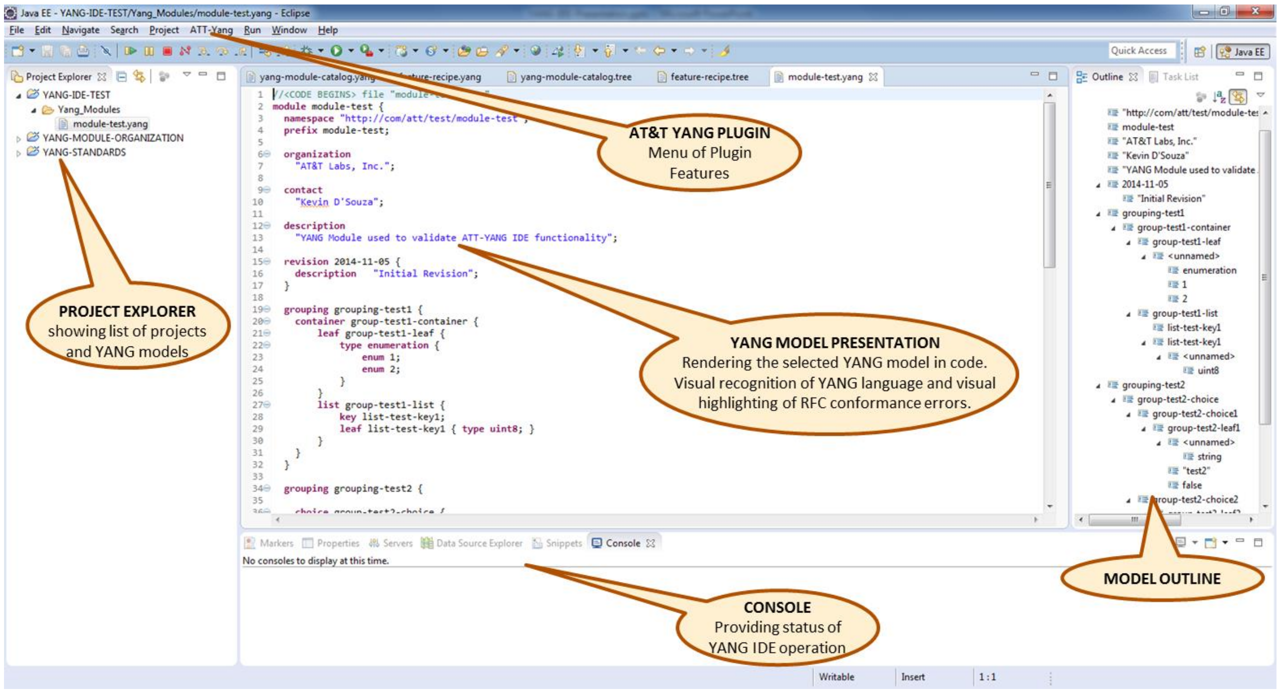
FEATURE HIGHLIGHTS:
(a) Validation of YANG Module Conformance to IETF RFC 6020,
(b) Tree, UML, XSD, JSTREE, XML, XSLT and PNG Rendition of YANG Module

This Plugin depends on projects developed by other open source projects:

Eclipse Release Juno or later:
  (Eclipse Public License: https://www.eclipse.org/legal/epl-v10.html);
Xtext Release 2.6.2 or later:
  (Eclipse Public License: https://www.eclipse.org/legal/epl-v10.html);
Python 2.7 or later:
  (Python License: https://docs.python.org/2/license.html);
plantuml.jar:
  (MIT License: http://plantuml.sourceforge.net/download.html#mit);
GraphViz/Dot:
  (Eclipse Public License: https://www.eclipse.org/legal/epl-v10.html);
PYANG 1.4.1 or later:
  (New BSD License: https://code.google.com/p/pyang/);

Updates to following ENV Varaibles needed: $YANG_MODPATH; $GRAPHVIZ_DOT

ATT-YANG DESIGN STUDIO ECLIPSE PLUGIN OVERVIEW

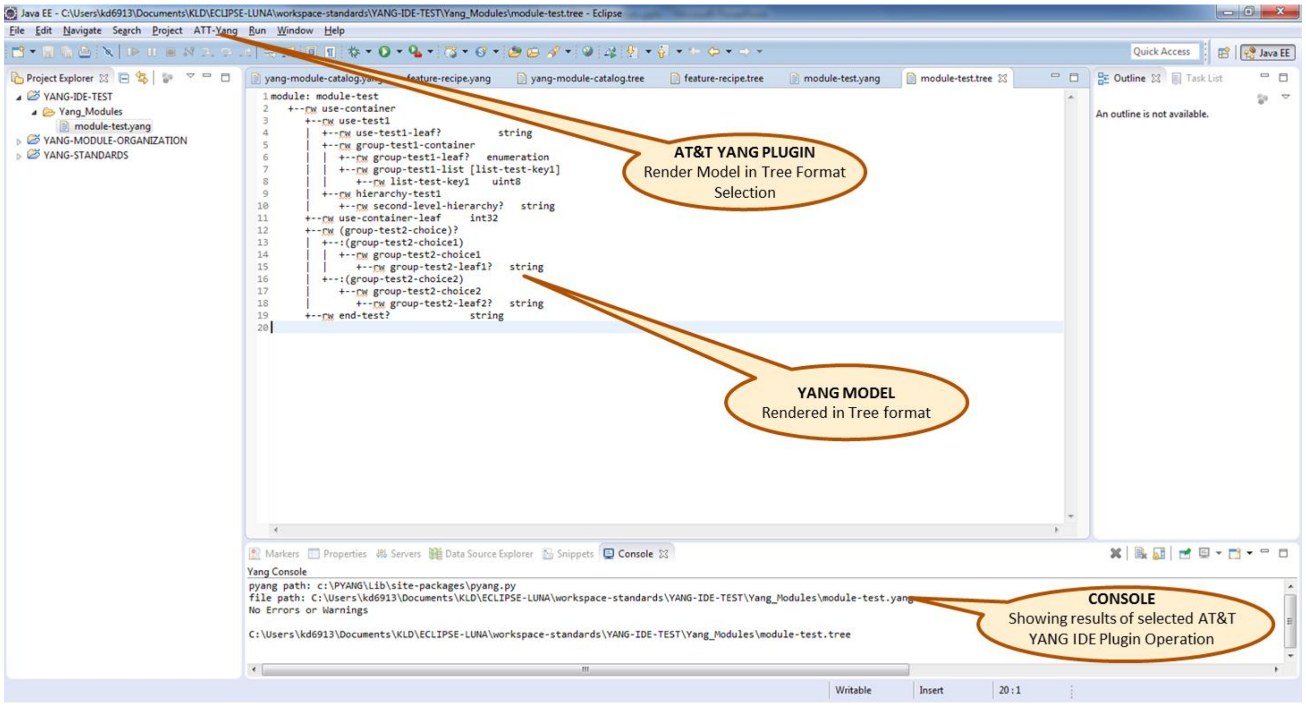


```
//<CODE BEGINS> file "module-
module module-test {
  namespace "http://com/att/test/module-test";
  prefix module-test;

  organization
    "AT&T Labs, Inc.";

  contact
    "Kevin D'Souza";

  description
    "YANG Module used to validate ATT-YANG IDE functionality";

  revision 2014-11-05 {
    description   "Initial Revision";
  }

  grouping grouping-test1 {
    container group-test1-container {
        leaf group-test1-leaf {
            type enumeration {
                enum 1;
                enum 2;
            }
        }
        list group-test1-list {
            key list-test-key1;
            leaf list-test-key1 { type uint8; }
        }
    }
  }

  grouping grouping-test2 {

    choice group-test2-choice {
```

```
"http://com/att/test/module-te
module-test
"AT&T Labs, Inc."
"Kevin D'Souza"
"YANG Module used to validate
2014-11-05
  "Initial Revision"
grouping-test1
  group-test1-container
    group-test1-leaf
      <unnamed>
        enumeration
        1
        2
    group-test1-list
      list-test-key1
      list-test-key1
        <unnamed>
          uint8
grouping-test2
  group-test2-choice
    group-test2-choice1
      group-test2-leaf1
        <unnamed>
          string
        "test2"
        false
    group-test2-choice2
```

No consoles to display at this time.

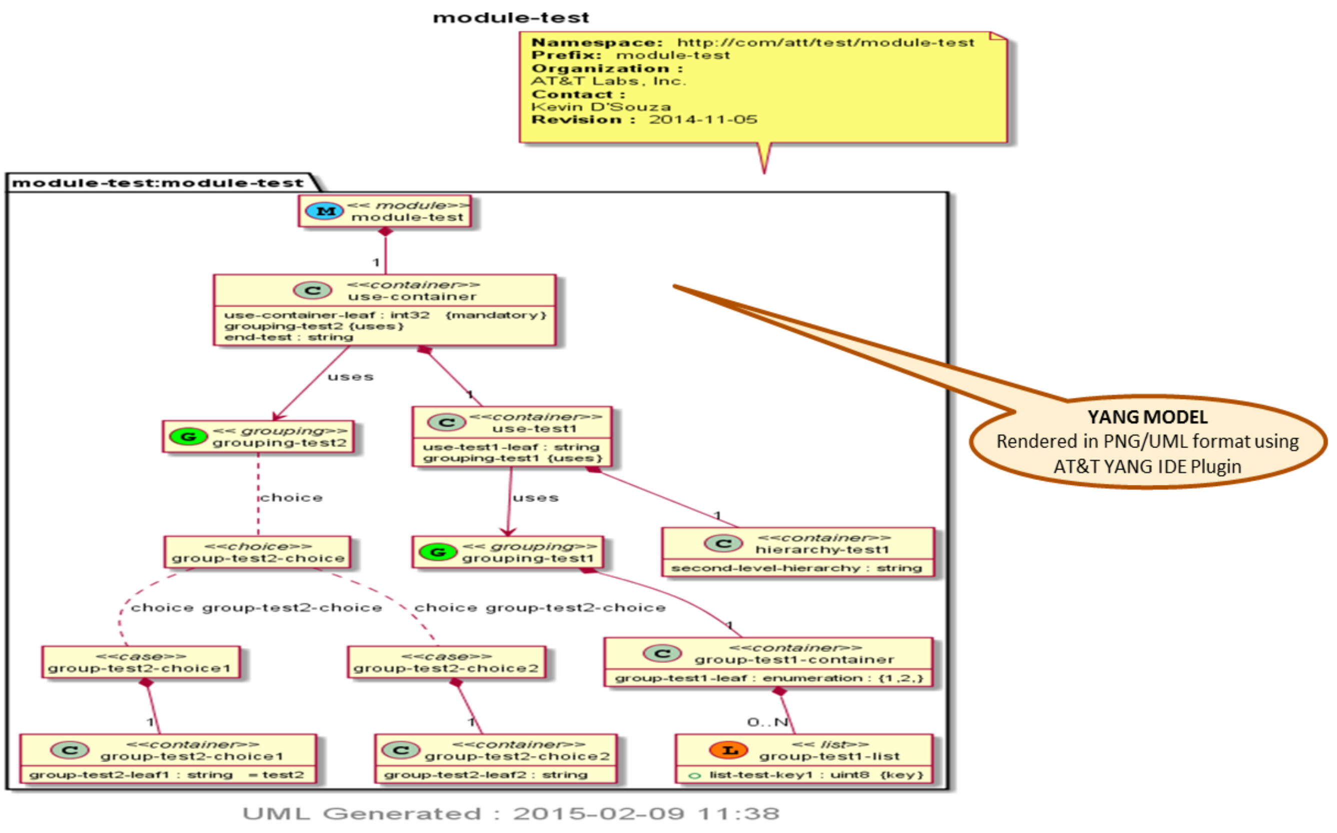
ATT-YANG DESIGN STUDIO – EXAMPLE RENDERING OF A TREE


Java EE - C:\Users\kd6913\Documents\KLD\ECLIPSE-LUNA\workspace-standards\YANG-IDE-TEST\Yang_Modules\module-test.tree - Eclipse
File Edit Navigate Search Project ATT-Yang Run Window Help
Project Explorer
YANG-IDE-TEST
Yang_Modules
module-test.yang
YANG-MODULE-ORGANIZATION
YANG-STANDARDS
yang-module-catalog.yang
feature-recipe.yang
yang-module-catalog.tree
feature-recipe.tree
module-test.yang
module-test.tree
Outline
Task List
An outline is not available.
AT&T YANG PLUGIN
Render Model in Tree Format Selection
YANG MODEL
Rendered in Tree format
Markers Properties Servers Data Source Explorer Snippets Console
Yang Console
CONSOLE
Showing results of selected AT&T YANG IDE Plugin Operation
Writable
Insert
20 : 1

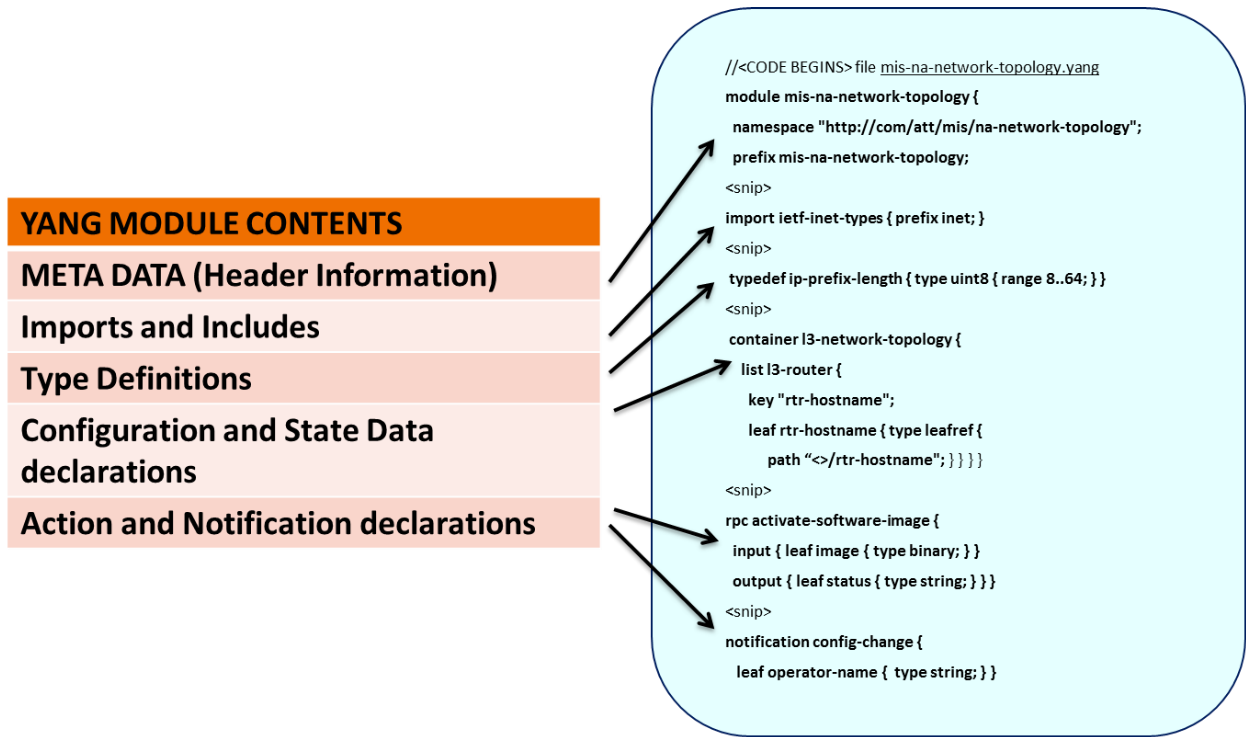
```
module: module-test
   +--rw use-container
      +--rw use-test1
      |  +--rw use-test1-leaf?          string
      |  +--rw group-test1-container
      |  |  +--rw group-test1-leaf?   enumeration
      |  |  +--rw group-test1-list [list-test-key1]
      |  |     +--rw list-test-key1    uint8
      |  +--rw hierarchy-test1
      |     +--rw second-level-hierarchy?   string
      +--rw use-container-leaf     int32
      +--rw (group-test2-choice)?
      |  +--:(group-test2-choice1)
      |  |  +--rw group-test2-choice1
      |  |     +--rw group-test2-leaf1?   string
      |  +--:(group-test2-choice2)
      |     +--rw group-test2-choice2
      |        +--rw group-test2-leaf2?   string
      +--rw end-test?             string
```

```
pyang path: c:\PYANG\Lib\site-packages\pyang.py
file path: C:\Users\kd6913\Documents\KLD\ECLIPSE-LUNA\workspace-standards\YANG-IDE-TEST\Yang_Modules\module-test.yang
No Errors or Warnings

C:\Users\kd6913\Documents\KLD\ECLIPSE-LUNA\workspace-standards\YANG-IDE-TEST\Yang_Modules\module-test.tree
```

ATT-YANG DESIGN STUDIO – EXAMPLE RENDERING OF UML


module-test
Namespace: http://com/att/test/module-test
Prefix: module-test
Organization :
AT&T Labs, Inc.
Contact :
Kevin D'Souza
Revision : 2014-11-05
module-test:module-test
<< module>> module-test
1
<<container>> use-container
use-container-leaf : int32 {mandatory}
grouping-test2 {uses}
end-test : string
uses
1
<< grouping>> grouping-test2
<<container>> use-test1
use-test1-leaf : string
grouping-test1 {uses}
choice
uses
1
<<choice>> group-test2-choice
<< grouping>> grouping-test1
<<container>> hierarchy-test1
second-level-hierarchy : string
choice group-test2-choice
choice group-test2-choice
1
<<case>> group-test2-choice1
<<case>> group-test2-choice2
<<container>> group-test1-container
group-test1-leaf : enumeration : {1,2,}
1
1
0..N
<<container>> group-test2-choice1
group-test2-leaf1 : string = test2
<<container>> group-test2-choice2
group-test2-leaf2 : string
<< list>> group-test1-list
list-test-key1 : uint8 {key}
YANG MODEL
Rendered in PNG/UML format using
AT&T YANG IDE Plugin
UML Generated : 2015-02-09 11:38

YANG MODULE OVERVIEW


YANG MODULE CONTENTS
META DATA (Header Information)
Imports and Includes
Type Definitions
Configuration and State Data declarations
Action and Notification declarations

```
//<CODE BEGINS> file mis-na-network-topology.yang
module mis-na-network-topology {
 namespace "http://com/att/mis/na-network-topology";
 prefix mis-na-network-topology;
<snip>
import ietf-inet-types { prefix inet; }
<snip>
 typedef ip-prefix-length { type uint8 { range 8..64; } }
<snip>
 container l3-network-topology {
   list l3-router {
     key "rtr-hostname";
     leaf rtr-hostname { type leafref {
          path “<>/rtr-hostname"; } } } }
<snip>
rpc activate-software-image {
 input { leaf image { type binary; } }
 output { leaf status { type string; } } }
<snip>
notification config-change {
  leaf operator-name {  type string; } }
```