YangIDE

Welcome to YangIDE

Introduction

The YangIDE subproject will provide an Eclipse plugin that can be used to create, edit, and view Yang model files.

Who Would Want to Use This?

The target audience of this tool is anyone who needs to work with Yang models, either to create new models, make changes to existing models, or view existing models, perhaps to use for integrating these models into new services and custom appliances.

It is particularly appropriate for Java developers who are already used to the Eclipse ecosystem, but the value it provides in the areas of immediate validation and ease of use will be very enjoyable for network interface designers who do not work primarily with Java, or at all, if they previously have only used plain text editors or minimally enhanced Yang editors.

In terms of the feature set that Yang IDE provides for editing Yang model files, it is very comparable to other advanced programming language editors in the Eclipse ecosystem. These features not only provide considerable value, they are expected. A programming language editor in this ecosystem that did not provide features comparable to this would be seen as insufficient.

The Yang IDE uses components of OpenDaylight internally, but it's not a requirement that the Yang models developed with it should be used with OpenDaylight. The models produced by Yang IDE can be used with any system that supports version 1.0 of the Yang specification (updating to 1.1 will happen eventually).

File and Project Support

- Create Yang file
- Create Yang Maven project

Code Editing

- While editing or viewing a Yang model file, symbols and keywords will be color-coded to provide visual differentiation.
- While entering new Yang predefined keywords, completion will be available on those symbols.
- While entering new user-defined symbols (module names, type names, et cetera), completion will be limited by the scope of the current keyword (completion on a grouping name will only offer grouping names).
- While entering new user-defined symbols with an existing prefix, the prefix will be considered as a namespace, so only symbols in that namespace will be offered.
- The editing view will immediately notify the user when the text they have entered is either syntactically or semantically invalid. Syntactic checking will be based on compliance to the Yang 1.0 specification. Semantic checking will report issues that would only be seen by a full Yang compilation, like mistyped module or type names.
- While viewing the reference to an imported module or type reference, a keyboard shortcut (or menu item) will change the editor view to display the defined module or type definition. Another keyboard shortcut will return the user to the original file. This view of defined modules or types will work whether the module is present in source form in the project itself, or provided by an external import source, like Maven dependencies, Github sources, or local project dependencies.
- The editor will provide “refactoring shortcuts”. This will include renaming symbols causing all the references to effect the same change, along with advanced refactoring features like selecting

Project Facts

Project Creation Date: Nov 19th, 2015

Lifecycle State: Incubation

Primary Contact: David Karr <dk068x@att.com>

Project Lead: David Karr <dk068x@att.com>

Committers:

- David Karr <dk068x@att.com>

Mailing List: yangide-dev@lists.opendaylight.org
Archives: mailing list archives

Meetings: See Community Meetings

Repository: git clone https://git.opendaylight.org/gerrit/yangide

Jenkins: jenkins silo

Gerrit Patches: code patches /reviews

Bugs:

- open bugs

multiple leaf items and replacing it with the definition of a grouping and the reference to that grouping. Refactoring will consider components with the same name in a different namespace to be different components.
- Hovering the mouse over a reference to a symbol will display in a tooltip the description of that symbol.
- While viewing a large Yang model file, hierarchical constructs can be “folded” to provide a wider view of the entire file. A separate “Outline view” will provide a high-level list of the components in the file.
- To save time entering common constructs, users can define templates that are inserted by entering a single symbol along with a keyboard shortcut. The plugin will come with a full set of these, and the user can easily add to them.
- Preferences for the plugin can specify properties of the Yang code formatter, with options like presenting imports on a single line, or wrapping long description strings, along with formatting the entire file with reasonable indenting.

Diagram Model Editor

- In addition to the textual Yang model view, a graphical “UML-like” parallel editor will be available, allowing navigation and editing of Yang models in a “point-and-click” fashion. This will be integrated directly with the text editor view, so changes made in the text view will be immediately visible in the diagram view, and vice versa.

Export Features

- Export Yin (XML) file
- Export UML view of Yang model

Maven integration

- The editor will provide a wizard to use the “Maven Yang Plugin” to generate Java code associated with the Yang model.
- Import Maven project.
- Yang module dependencies will be resolved through Maven dependencies specified in project.

non-Maven integration

- Use Eclipse project dependencies for additional import sources

Documentation

Project Information

Project Proposal

Release	Release Plan	Release Notes	Release Review	Installation Guide	User Guide	Developer Guide
Boron	Release Plan	Release Notes	Release Review	Installation Guide	User Guide	Developer Guide

Requirements

Release Planning

Release Notes