crystal_facet_uml user documentation

COLLABORATORS			
	TITLE : crystal_facet_uml user documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Andreas Warnke	2018-11-04	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Introduction

1.1 Goal

crystal_facet_uml creates a set of uml diagrams.

It ensures consistency of relationships and uml element names between different diagrams.

crystal_facet_uml exports diagrams in various vector and pixel-based image formats.

1.2 Features

crystal_facet_uml provides a graphical user interface to

- create diagrams
  (use-case, deployment, component, composite-structure, package, class, activity, state, timing, communication, sequence)
- manage diagrams in a tree-hierarchy
- create uml elements
  (actor, system-boundary, use-case, node, component, part, interface, package, class, activity, state, object, artifact, comment, requirement)
- move, modify and delete uml elements
- create, modify and delete relationships
  (dependency, association, aggregation, composition, generalization, realization, contains, sync-call, return-call, async-message, communication-path, control-flow, object-flow, deployment, manifest, include, extend)
- create, modify and delete features
  (port, field, operation)
- cut, copy, paste of uml elements between diagrams
- undo and redo are supported
- multiple windows can show different or same parts of the uml model

Diagrams are layouted part-automatically:

- The user chooses the relative location of uml elements towards others
- crystal_facet_uml selects the exact locations of uml elements
- The user controls the positions of messages/transisions in sequence and timing diagrams
- crystal_facet_uml auto-layouts relationships in other diagrams

crystal_facet_uml manages a meta model (this uml-model is stored in an sqlite database):

- Diagrams are organized as a tree, similar to a book's table-of-contents

- Uml elements exist only once even if shown in many diagrams
- Relationships and features are consistent between all diagrams
- Diagram-local messages/transitions are supported in scenario based diagrams
  (sequence, communication, timing)

crystal_facet_uml exports diagrams as

- vector graphics
  (pdf, ps, svg)
- pixel graphics
  (png)
- textual representation
  (utf8)

crystal_facet_uml can also be started from command line to check and repair database files.

1.3 Usage Overview

crystal_facet_uml can be started in graphical mode to

- manage a diagram structure
- create, modify and delete diagrams
- create, modify and delete uml elements in diagrams
- create, modify and delete relationships and features of uml elements
- export the uml diagrams to image file formats

crystal_facet_uml can also be started from command line to check and repair database files. Run

```
./crystal_facet_uml -h
```

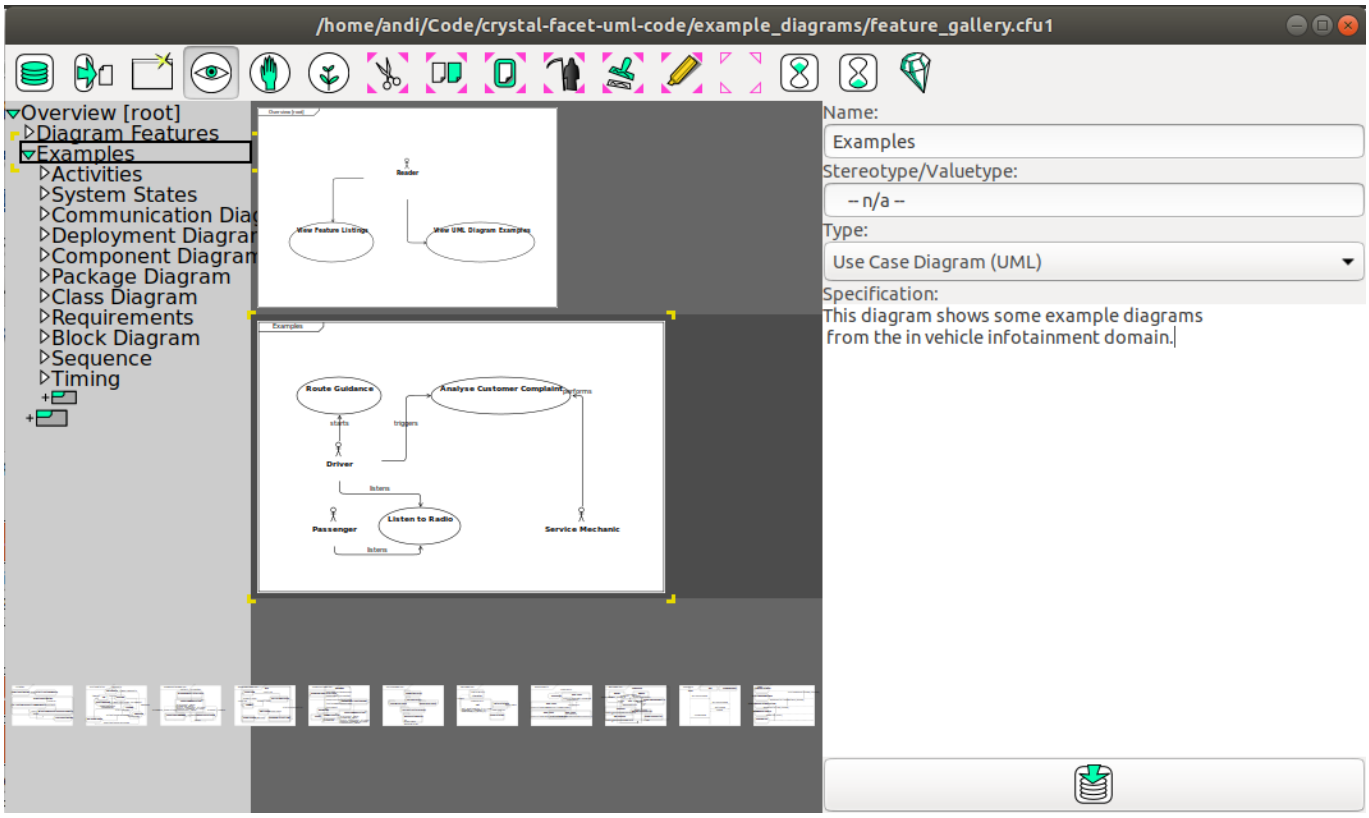
to get a list of supported parameters.

2 GUI

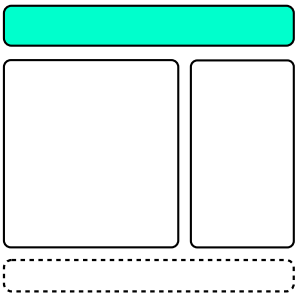
2.1 Overview

If started in graphical mode, crystal_facet_uml shows a window with

- toolbar on top,
- drawing area in the center,
- element configuration widgets to the right and
- an optional notification bar at the bottom.



2.1.1 Tool Bar



2.1.1.1 Create/Use DB

- Opens an existing database file or creates a new database file

2.1.1.2 Export

- Exports all diagrams to the selected folder (supported formats are txt, png, pdf, ps and svg)

2.1.1.3 New Window

- Opens another window on the same database.

2.1.1.4 Navigate

- Navigate to parent or child diagrams
- Create a new diagram

2.1.1.5 Edit

- Modify elements in the diagram

2.1.1.6 Create

- Create elements in the diagram

2.1.1.7 Cut

- Cut all pink-cornered elements to the clipboard (features of classifiers are cut independantly of their corner-colors)

2.1.1.8 Copy

- Copy all pink-cornered elements to the clipboard (features of classifiers are copied independantly of their corner-colors)

2.1.1.9 Paste

- Pastes diagrams and classifiers from the clipboard to the uml model. (Relationships are not pasted) If id and name are identical to an existing element, an instance of the existing element is pasted to the diagram. Otherwise a new element is created.

2.1.1.10 Delete

- Deletes all pink-cornered elements. This operation may fail if marked elements have unmarked children.

2.1.1.11 Instantiate

- Toggles the pink-cornered classifiers between classes and instances. (Does not work for relationships and features)

2.1.1.12 Highlight

- Toggles the pink-cornered classifiers between yellow-marked, greyed-out and normal. (Does not work for relationships and features)

2.1.1.13 Reset Selection

- Resets the pink-cornered selection

2.1.1.14 Undo

- Un-does the last operation (Opening a database and exporting files cannot be undone)

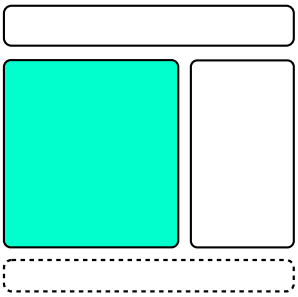
2.1.1.15 Redo

- Re-does the last un-done operation

2.1.1.16 About

- Shows version, license and copyrights

2.1.2 Drawing Area



Diagrams are layouted automatically. You can influence the locations of classifiers only. When adding too many classifiers or relations, auto layouting may not achieve the expected results. In many cases, splitting the diagram into two or more diagrams solves the layouting issues and at the same time improves understandability by focusing on one aspect/topic per diagram.

2.1.2.1 Navigate

- To navigate to the parent diagram, click on the parent diagram.
- To navigate to a child diagram, click on the child diagram.
- To create a diagram, click on the + sign.
- To restructure the diagram tree by shifting a child diagram up and the parent down, click on the child diagram and press F7.

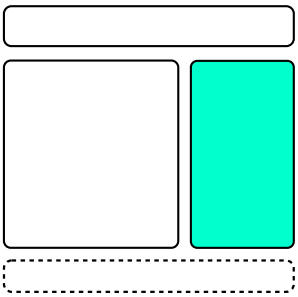
2.1.2.2 Edit

- To move classifiers within the diagram, press, drag and release the mouse button.
- To select the diagram or a classifier or a feature or a relationship with yellow corners, click on this object.
- To mark diagram and/or classifiers and/or features and/or relationships with pink corners, click on these objects twice.

2.1.2.3 Create

- To create a classifier, click at an empty space in the diagram.
- To create a child classifier, click into the white space of a classifier. Alternatively, create a classifier and a containment relation.
- To create a relationship, press on the source classifier and drag it to the destination classifier.
- To create a feature, click onto a classifier (name or border).

2.1.3 Element Configuration



Edit the properties of the yellow-cornered object.

- 0.: name of the focused object
- 1.: stereotype/valuetype of the focused object (deactivated depending on object-type)
- 2.: type of the focused object
- 3.: description of the focused object

2.1.3.1 Maximum stringlengths

All strings (names, descriptions, stereotypes) have a maximum length. Ascii characters require one, most other characters two bytes. Current sizes in bytes are: Classifiers:

- DATA_CLASSIFIER_MAX_NAME_LENGTH = 47,
- DATA_CLASSIFIER_MAX_STEREOTYPE_LENGTH = 47,
- DATA_CLASSIFIER_MAX_DESCRIPTION_LENGTH = 4095,

Features:

- DATA_FEATURE_MAX_KEY_LENGTH = 47, (name)
- DATA_FEATURE_MAX_VALUE_LENGTH = 255, (type)
- DATA_FEATURE_MAX_DESCRIPTION_LENGTH = 1023,

Relationships:

- DATA_RELATIONSHIP_MAX_NAME_LENGTH = 47,
- DATA_RELATIONSHIP_MAX_DESCRIPTION_LENGTH = 1023,

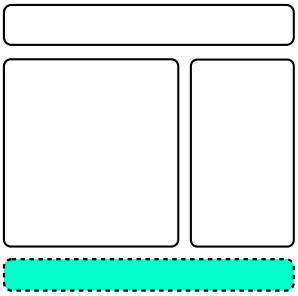
Diagrams:

- DATA_DIAGRAM_MAX_NAME_LENGTH = 47,
- DATA_DIAGRAM_MAX_DESCRIPTION_LENGTH = 8191,

2.1.3.2 Commit

- Stores the latest changes to the database immediately. This feature is optional, it is not necessary to explicitly save the file.

2.1.4 Notification Bar



2.1.4.1 Information

- Informs on success of an operation, e.g. an export

2.1.4.2 Warning

- Informs on a possible problem

2.1.4.3 Error

- Informs on an error

3 modelling guidelines

3.1 Modelling Introduction

This page lists remarks on creating a software architecture and design document in general and it lists hints on getting along with the tool crystal_facet_uml. As all tools, this program has its strengths and weaknesses. This page helps in making use of the strenghts.

3.1.1 crystal_facet_uml Hints

3.1.1.1 Tree Structure

Diagrams are organized as a tree. Start the root of the tree explaining the document scope and structure. At the second level of the tree, list the main areas to be shown, for example based on the arc42 template https://arc42.org/overview/ :

- Introduction and Goals
- Constraints

- Context and Scope (show the system boundary, what is outside, what use-cases exist)
- Solution Strategy
- Building Block View,
- Runtime View,
- Deployment View,
- Crosscutting Concepts,
- Architectural Decisions (show alternatives, give a rationale),
- Quality Requirements
- Risks and Technical Debt
- Glossary (table showing Context, Term and Description)

3.1.1.2 Focus

Put only few elements into each diagram. This increases understandability of the main purpuse of the diagram. Put further aspects of a topic into a separate diagram. Do not hesitate to copy an element from one diagram to the next. This is what crystal_facet_uml is good at: it keeps the model in sync.

3.1.2 General Hints on Architecture Documentation

3.1.2.1 Problem vs. Solution

Distinguish things that are

- given constraints (problem space),
- decisions, chosen and rejected alternatives and
- the designed solution

3.1.2.2 Names

Names of things are crucial: If the reader gets a wrong understanding by the name of an element, a hundred correct sentences of describing text cannot set this straight again.

3.1.2.3 Description

Every design element needs a description, maybe a list of responsibilities: What shall this element do, what is it for? Names alone cannot explain a system part.

3.1.2.4 Precise sentences

Be precise: Write in active form, e.g. The persistence component shall store and retrieve binary data records indentified by string-based keys.

3.1.2.5 Distinguish similar things

Things that are similar but not the same shall be different entities when modelling. E.g. The process in which an example application runs may be different from the storage location and may be different from the software-component. These are three things: Example_App_Process (Type: Node), Example_App_ObjectFile (Type:Artifact) and Example_App_SWComponent (Type:Component).

A Further Information

A.1 Download

Find the latest version at:

- https://sourceforge.net/projects/crystal-facet-uml/
- https://github.com/awarnke/crystal_facet_uml
- https://build.opensuse.org/package/show/home:awarnke/crystal_facet_uml

User documentation is available here:

- http://www.andreaswarnke.de/crystal_facet_uml/crystal_facet_uml_user_documentation.pdf
- https://github.com/awarnke/crystal_facet_uml/blob/master/user_doc/crystal_facet_uml_user_documentation.pdf

A.2 License