GUI_NotepadDefnDelete – Notepad Definition Delete

Robin Lamacraft 2017-03-26

SCOPE

This GUI module displays a screen to delete a Notepad Definition instance. This action only applies to Custom Notepads. In particular, it will detect whether another object instance will become corrupted if this Notepad Definition instance is deleted.

LOOK AND FEEL

The screen has 2 horizontal parts:

- The heading section shows
  - the focus Notepad Definition instance HRE-ID
  - the name of the Notepad Definition instance.
- A collection of buttons including “Configure”, “Delete”, “Apply”, “Cancel” and “Output”:
  - “Configure” opens a screen that allows the creation and re-use of screen content and layout
  - “Delete” starts a search to check that it is possible to delete Notepad Definition without causing inconsistency in the database:
    - This search lists on the screen the other objects that require the Notepad Definition to continue to exist
    - This scrollable table has columns HRE-Id, Object Type, Object Name (if it has one).
  - This search has 2 possible outcomes:
    - “Apply” is enabled on clicking if no dependent objects were found. “Apply” removes the Pattern
    - “Output” - If there are other objects that require the Definition to continue to exist, then “Output” can save this list for analysis as a file or to print it. There may be several reasons the Notepad Definition can’t be removed
    - “Cancel” does not delete the Notepad.

[Needs a mockup diagram here]

ACTIONS

The fundamental operations are:

1. Open Screen according to its saved Screen Layout (BR_PanelConfig)
2. Populate the property editor pane with values for the selected object
3. Populate the heading
4. Perform the search
5. Delete the Entity
6. Output the result of the search.

USED BY:

1. All links (not objects) that have “NotepadDefinition Edit” entries in the menu use this GUI_NotepadDefnDelete as their dependent coding
2. Almost any link type, either project-oriented or application-oriented has a GUI_NotepadDefnDelete variant. Because these GUI elements create mouse and keyboard events, each of these GUI screens must have unique identities. This means that the basic screen layout can be defined as an abstract class where each separate real class contains the object type specific code listening for its specific events to act upon.

DATA CONTROLLED BY THIS MODULE

1. None.

REQUIRED DATA CONTROLLED BY OTHER MODULES

1. HRE-ID.

REQUIRED SERVICES

1. GUI_PanelConfigEdit
2. GUI_FieldTranslationEdit
3. GUI_Output
4. BR_Setting
5. BR_PanelConfig
6. BR_Notepad
7. BR_EntityLink.

APPLICATION PROGRAMMING INTERFACE (API)

1. Need Details.

EVENT ACTIONS

1. Need details of event (keyboard or mouse) and the description of the action.

WARNING CONDITIONS

1. Need details of the condition that raised the warning, example message and possible next steps.

ERROR CONDITIONS

1. Need to record the condition that raised the error, example message and possible next steps.