Information Engineering Technology

Studio Developer Key New Features

Release 8.7

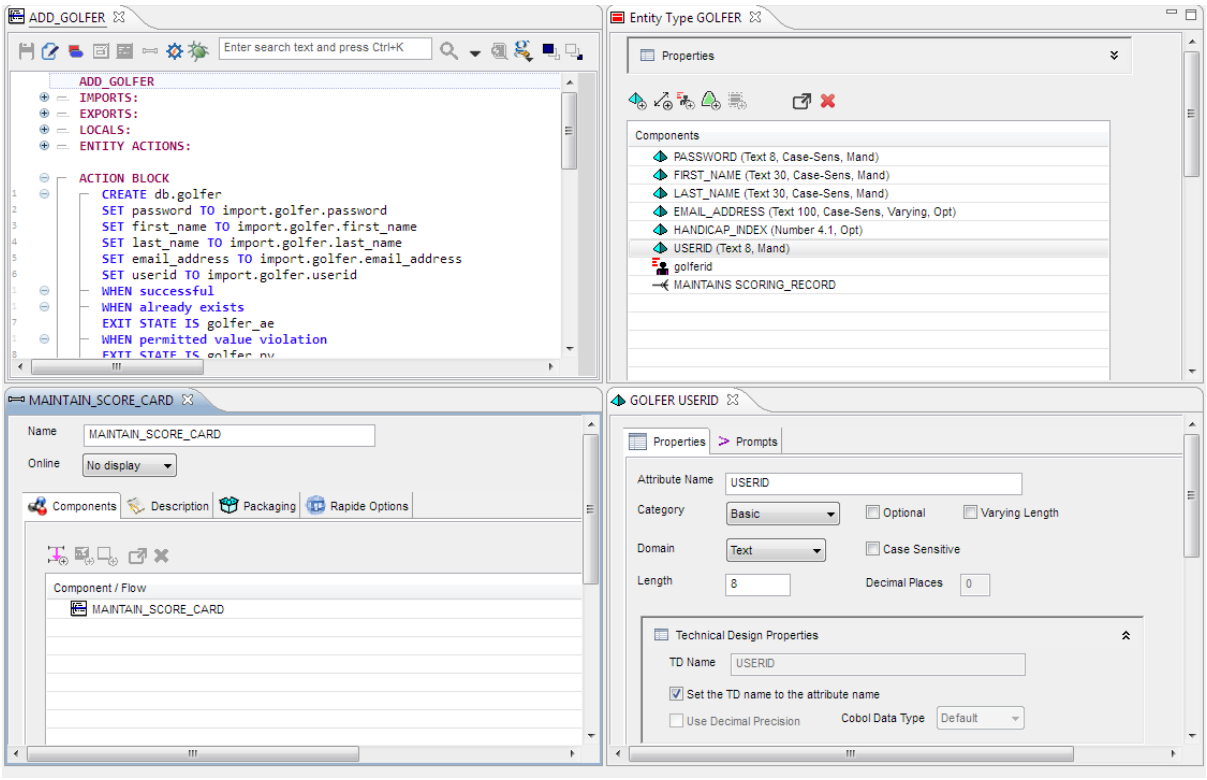
Key New Features

This document summaries the key new features available in Studio Developer.

General Features

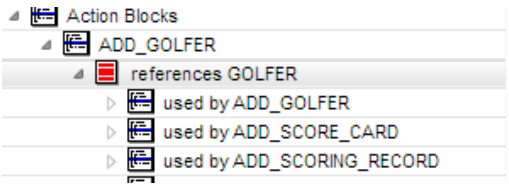
Multiple Editors

You can edit multiple objects simultaneously in independent editors.



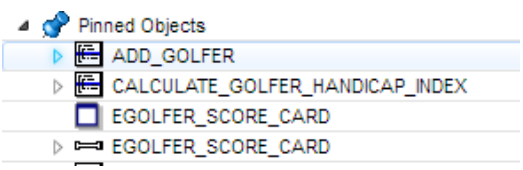
Model Explorer

The Model Explorer provides access to objects in the model via a tree structure with filtering and ‘where used’ and ‘contains’ functions.



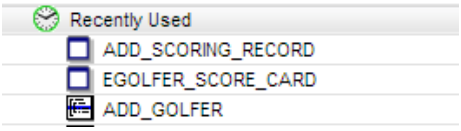
Pinned Objects

You can quickly access objects by ‘pinning’ them in the Model Explorer.



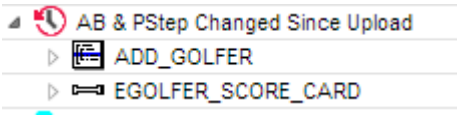
Recently Used Objects

You can quickly access objects that you have recently used by having them automatically added to the Recently Used list in the Model Explorer.



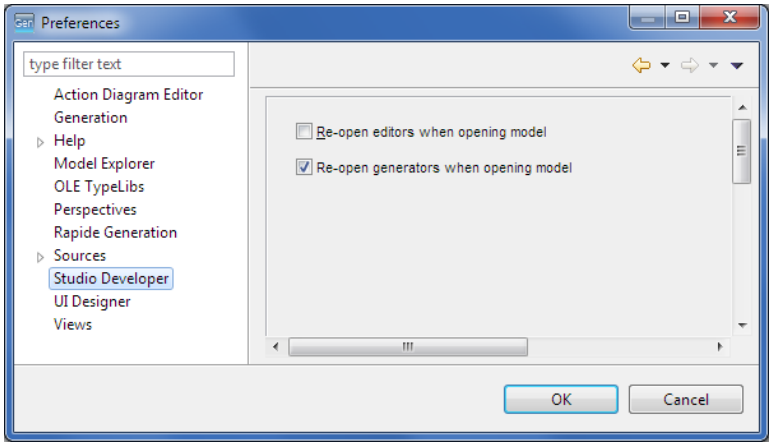
AB & Pstep Changed Since Upload

You can list action blocks and procedure steps that have changed since the last upload of the model to the encyclopaedia.



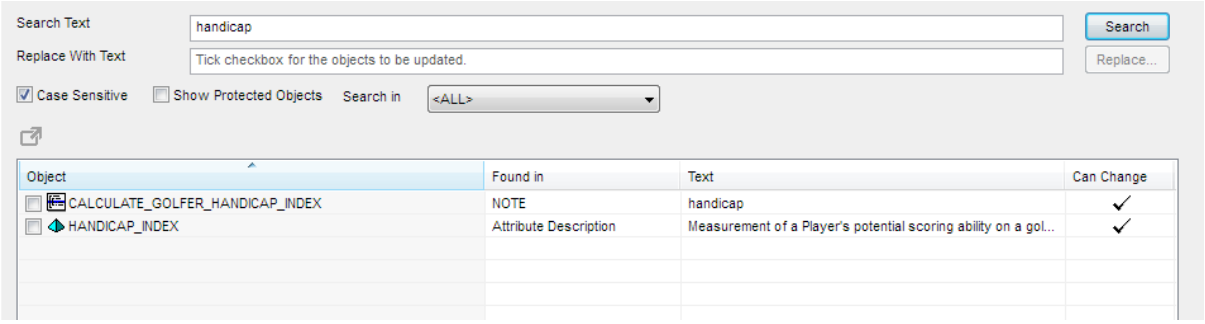
Re-open Editors

You can re-open editors and/or generators when opening the model by ticking the preference:



Search and Replace

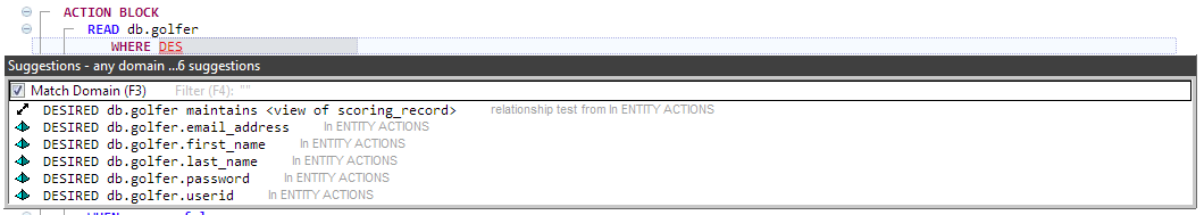
The model-wide Search and Replace tool accessed from the Model Explorer menu allows you to search the model for text strings in literals, notes and descriptions.



Action Diagram Features

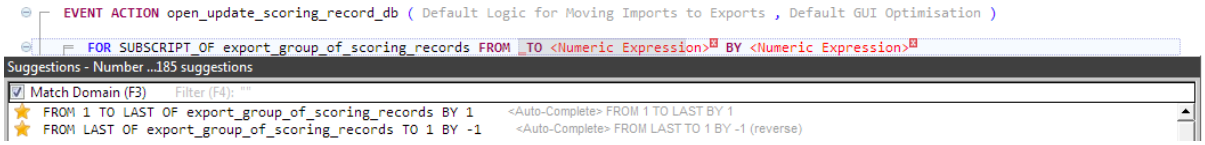
Keyboard Entry

The editor allows action diagram statements to be edited using a combination of keyboard entry and selection of objects using a mouse.



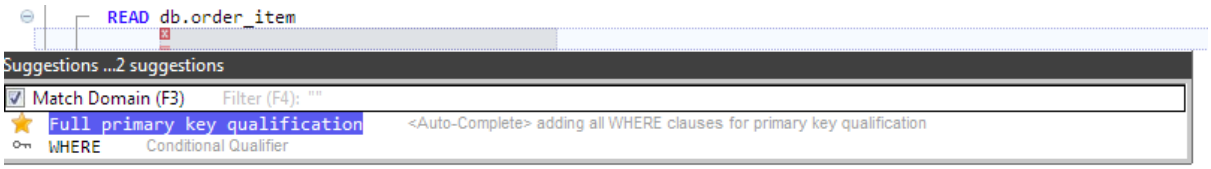
Auto Suggest / Complete

Code suggestion / auto-complete features are available to assist in selecting options and reduce the number of key strokes required. Some types of statements have an auto-complete feature.



READ qualified on the primary identifier

If you add a READ statement for a single entity action view, an auto-complete action is available to qualify the READ on the entity type's primary identifier.

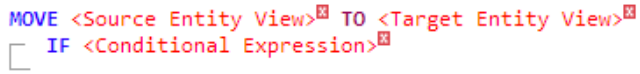


Undo/Redo

You can undo/redo statement editing back to the last save of the action diagram to the local model.

Flexible Editing

The diagram supports partially complete or invalid statements, which provides greater flexibility when editing statements since the diagram only has to be consistent when you want to save changes to the local model.



Copy & Paste

You can copy selected statements and paste back into the same action diagram or a different action diagram in the same or another model. Statements copied to the clipboard are also available in text and RTF (Rich Text Format) so they can be pasted into a text editor. If the text editor supports RTF, then the statement formatting and colouring is preserved.

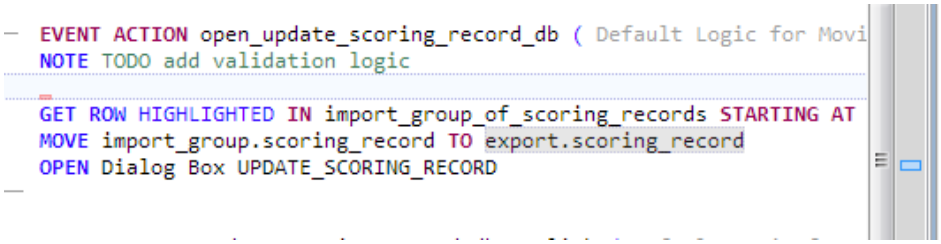
Copy & paste is more flexible than the legacy toolset. For example, you can copy a selection of ELSE_IF statements and paste into a new location with the first selected ELSE_IF becoming the initial IF statement.

Paste with substitution

You can paste with view substitution, which enables you to choose different views (of the same entity type / workset) or create new views.

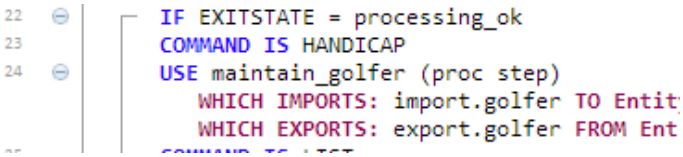
TODO Markers

If a NOTE statement contains the keyword “TODO” then the locator bar will indicate their presence with blue marker.



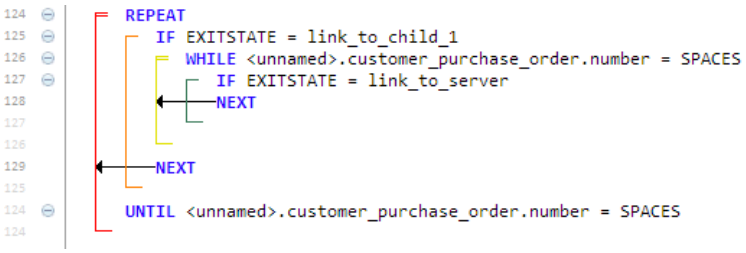
Statement Numbers

Statement numbers are displayed alongside the statements.



Block Line Colours

You can enable colouring of block lines with an action diagram preference. This makes it easier to locate the line for a specific statement when placing the end marker for an ESCAPE or NEXT pointer.

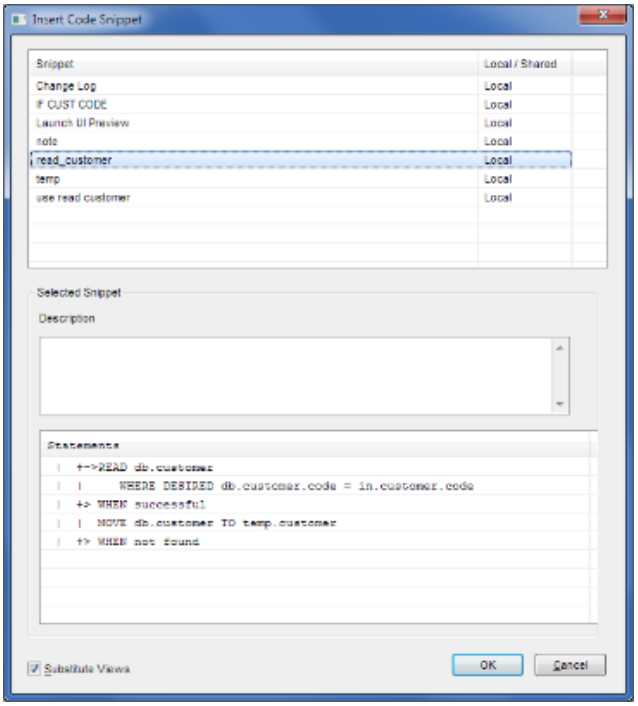


Code Snippets

Code snippets are one or more views or action diagram statements that can be saved in a file and then inserted into a different action diagram in the same or a different model.

They operate on the same principle as copy&paste with the same features and restrictions, for example automatic creation of views, exit states and commands, but other enabling objects like entity types, work sets or USEd action blocks must exist in the model.

You can tick the Substitute Views checkbox to specify alternate views when inserting the code snippet.



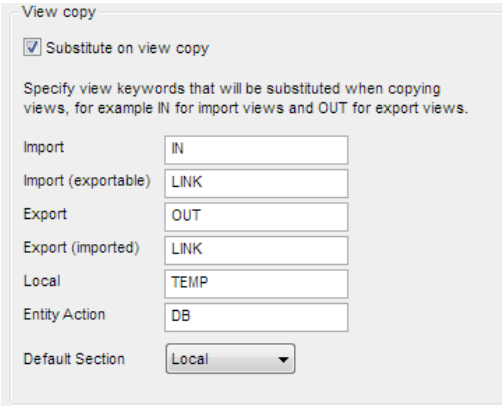
Inline Code Preferences

You can define inline code preferences which are then used to preset the values when adding a new INLINE code statement, for example the language.

View Maintenance Features

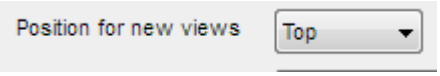
View Naming

The Views preferences allow you to specify view keywords for the various view sections. When matching or copying views, the keywords are used to format the copied view names. For example, when copying an import view named IN_HIDDEN to an export view, the view name would be pre-defined as OUT_HIDDEN.



View Position

The Views preferences allow you to specify whether new views are created at the top or bottom of the view section.



UI Design Features

Separate Editors

The windows and dialog boxes for each procedure step are edited in separate editors. This allows you to save the changes for each procedure step's UI design independently of other procedure steps.

Unlimited Size

The size of the window / dialog box being designed is not limited to the size of the developer's monitor. The window/dialog design area can be scrolled and zoomed in/out allowing the design of any size of window. This is especially useful when designing browser applications where the designed window can be scrolled vertically within the browser.

Undo/Redo

You can undo/redo editing of the UI design back to the last save to the local model.

Flexible Design

Controls can be added to the UI design before the views are created providing a more flexible approach to designing a new dialog.

Keyboard Control

You can use the keyboard to precisely position controls. Co-ordinates can also be entered in the property sheet.

◢ XY Co-ordinates	
XY Left	233
XY Right	320
XY Width	87
XY Top	226
XY Bottom	246
XY Height	20

Improved Interface

Tabbed controls and property sheets are used to specify all of the window and control properties, eliminating the need for most secondary pop-up dialog boxes.

Multiple control updates

When selecting multiple controls that share common properties, for example Margin Box or Video Properties, you can update the value of many of the common properties for the selected controls as one action rather than having to update each control separately.

Default Control Sizes

You can specify default height and width for push buttons and default height for fields in the Model Preferences so that newly created push buttons and fields have a standard size.

Copy Disabling Conditions

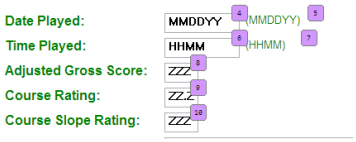
You can copy the disabling conditions from another control making it easier to define the same conditions for several controls.

UI Snippets

You can save standard controls (for example OK & cancel buttons) to a snippet file and insert them into a different window / dialog from a library of UI snippets.

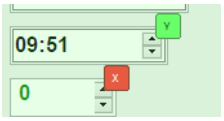
Tab Sequencing

When editing the tab sequence, visual indicators are shown on the UI design.



Mapping Visual Indicator

If the Mapping tab is selected in the UI Designer, a visual indicator is placed at the top right of controls that can have an import view mapping. The indicator is green for controls that are mapped and red where there is no import view mapped.

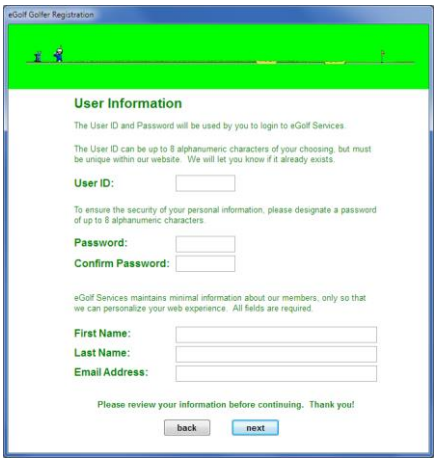


Integrated Rapide Designer

If you develop the user interface with Rapide, the additional Rapide specific control types and properties can designed in the same UI designer without the need for a separate plug-in.

UI Preview

You can preview windows and dialogs as a desktop or web page using the Preview feature. When using Rapide, this also allows you to preview how the responsive design features work in practice, for example re-sizing or moving controls based on the window size.



Data Structure Features

Custom Naming

With Studio Developer you can specify rules for how Studio Developer allocates names to attribute TD names, tablespaces, link tables, indexes, RI constraints and foreign key columns using Model Preferences.

Identifier Modification

If you modify the identifier of an entity type, you can synchronise the table's identifying index and associated foreign keys without needing to re-transform the entity type, thus retaining any customisations made to the table and its associated components.

Generation Features

Auto-including objects for generation

A feature of the code generation editors in Studio Developer including Rapide Generation is that whilst the generation editor is open it will detect changes to action blocks and procedure steps and automatically flag them for generation.

You can therefore open the required editors (for example client and server) and then when you have completed your changes to the model, switch to the editor and the objects will be pre-selected for generation.

Note that only the changed action blocks and procedure steps are flagged and a full impact analysis is not performed. For example, if you change the import view of an action block or change the length of an attribute, the impacted objects are not selected for generation. For these situations, you can use the 'Generate Used By' feature where you select an action block and it will flag all callers for generation.