Context Assistant

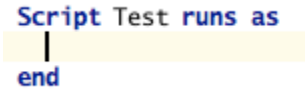
Overview

- Overview
- Using Context Assistant UI
- Using Context Assistant Framework
  - Placeholder Cells
  - Menu Lookup
  - Placeholder Lookup

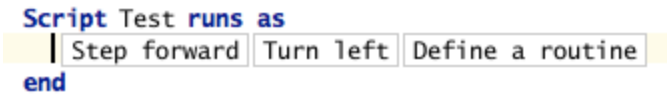
MPS provides several mechanisms for performing an action in a context: completion, intentions, refactorings, and various other popup menus. These mechanisms have in common that they are not immediately visible to new, inexperienced users. They also usually offer many possible choices and reveal the entire available functionality, which helps advanced users but may overwhelm the beginners.

To better guide the new users through the process of creating a script in your DSL, MPS 3.4 introduces a new UI mechanism, the context assistant. A context assistant shows a dynamically constructed menu with actions that are the most appropriate for a given context. The language author specifies where the menu should be shown by putting placeholders in the editor definition. The placeholders reserve screen space for the menu in advance so that the edited content does not shift around as the menu is being shown and hidden.

As an example consider the RobotKaja sample language which is bundled with MPS. The initial editor for a new script looks as follows (without a context assistant):



With a context assistant this initial UI might look like this:



and the menu now suggests several possible next steps to the user.

> We've also shot a short video illustrating use and definition of Context Assistant. You might also like to check out a screen-cast on how context assistant is being utilized for the language definition languages.

Using Context Assistant UI

- Jump to the context assistant by pressing Ctrl+Alt+Enter (Cmd+Option+Enter on Mac OS X).
- Navigate through the menu by using arrow keys.
- Invoke the selected menu item using Space or Enter.
- Press Escape to jump back to the editor.

Using Context Assistant Framework

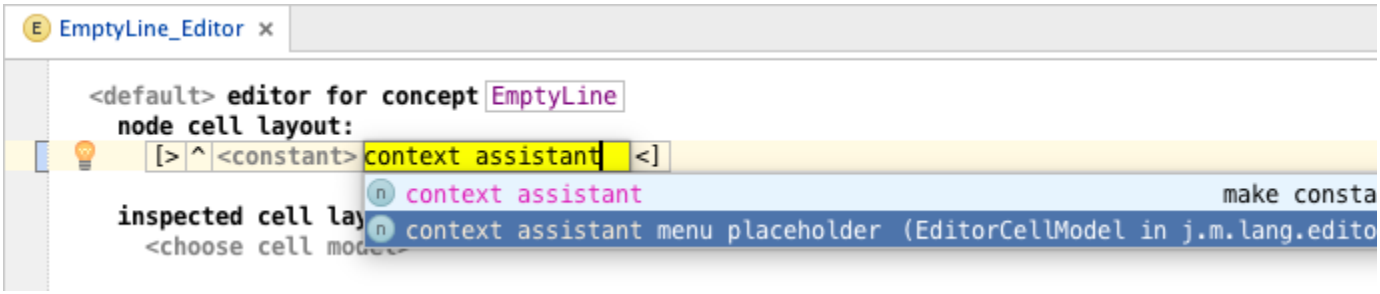
To add context assistant to a language, you as the language author have to do two things:

1. Place context assistant placeholders (a special kind of cell) at appropriate spots in the editor. The context assistant menus will be shown in these placeholders.
2. Define the menu hierarchy using the Transformation Menu Language (specifying location context assistant).

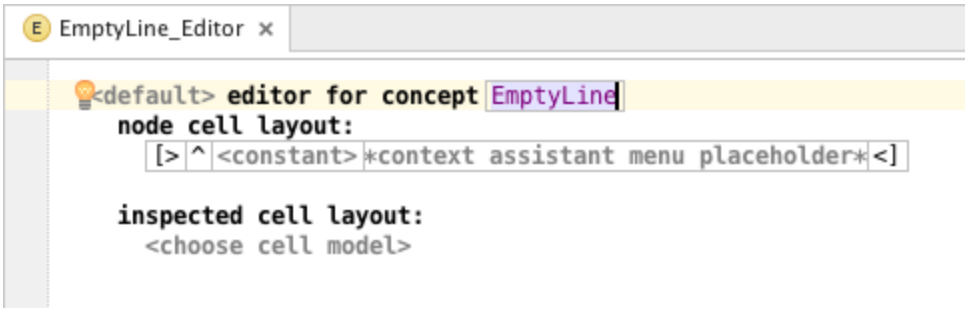
The MPS editor runtime will take care of building the appropriate menu at the appropriate point in time and showing it in the appropriate context assistant placeholder.

Placeholder Cells

Placeholder cells are added by choosing "context assistant menu placeholder" from the substitution menu when adding a new cell:



The placeholder cell reserves a certain amount of vertical screen space, about the size of one empty line, so that a menu can be shown in its place without shifting surrounding cells around. However, it doesn't reserve any horizontal space. It is therefore best to put the placeholder on a separate line or at the end of a short (or empty) line of text. For example, in the RobotKaja sample the placeholder is added after an empty line in the editor for the EmptyLine concept:



Menu Lookup

The menu to display is looked up by traversing the cell hierarchy from the currently selected cell to the top. You may specify the menu to show explicitly for a given cell (by setting the transformation menu property in the Inspector). In this case that menu is used. Otherwise, if the cell is a big cell (a cell that has no parent or whose parent is associated with a different node), an attempt is made to look up the menu based on the cell's node. The node's concept inheritance hierarchy is traversed in breadth-first order. If a non-empty menu is defined for the node's concept or one of its super-concepts and super-interfaces, this menu is used.

For example, consider a BaseLanguage `PlusExpression` which extends `BinaryOperation`, which in turn extends `Expression` and implements `IBinaryLike`. If during the traversal we reach the big cell of a `PlusExpression`, then menus of `PlusExpression`, `BinaryOperation`, `Expression`, `IBinaryLike`, and finally `BaseConcept` are checked, in that order, and the first non-empty menu definition is used. If all menu definitions are empty, the search continues from the parent cell of the big cell (if any).

Note that a non-empty menu definition, although chosen, may still produce an empty menu. This may happen if none of its menu parts produce any items (for example if no defined actions are currently applicable).

Placeholder Lookup

The place where a menu should be displayed is looked up by traversing the cell hierarchy from the currently selected cell to the root until a collection cell is reached that contains a context assistant placeholder cell, either directly or indirectly (but only belonging to the same node as the collection). The first cell found during this search is chosen and the menu is displayed in this placeholder cell.