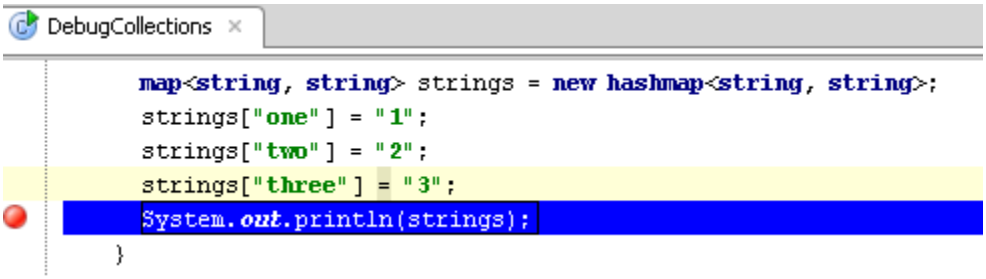
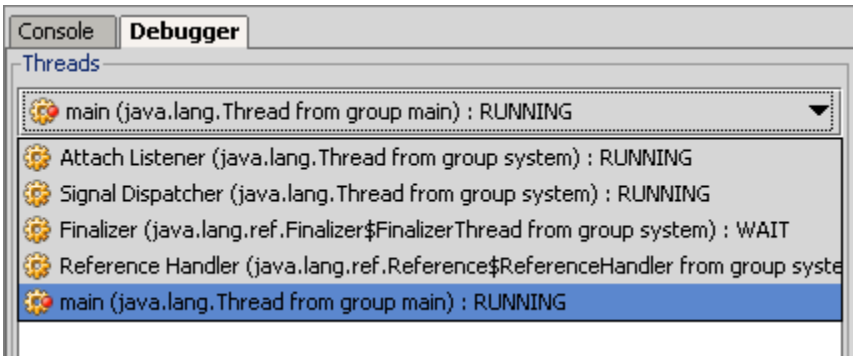
What's new in MPS 1.5

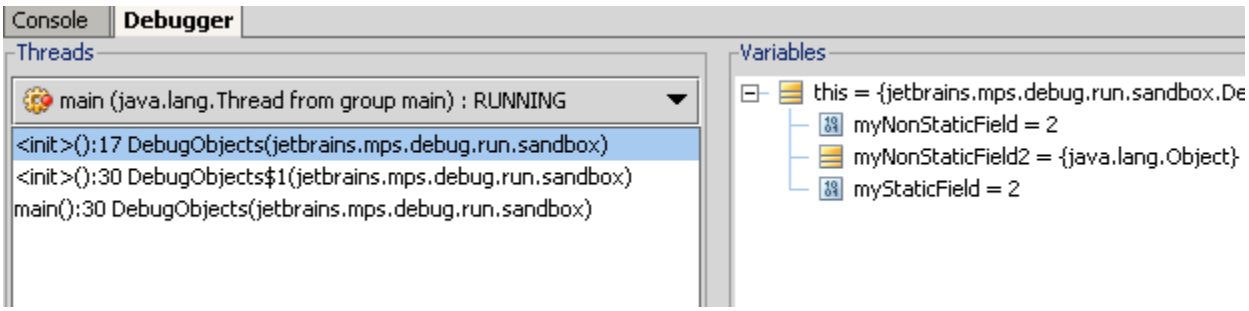
DSL debugger finally a part of MPS

One of the most important features in MPS 1.5 is a brand-new fully functioning DSL debugger. This debugger works with any languages that can be reduced to MPS Base Language (Java). Now you can start debugging your application running either locally or remotely with just a single mouse-click. You can also add breakpoints into DSL code by pressing Ctrl+F8 or by clicking onto editor margin to the left of the corresponding code line.

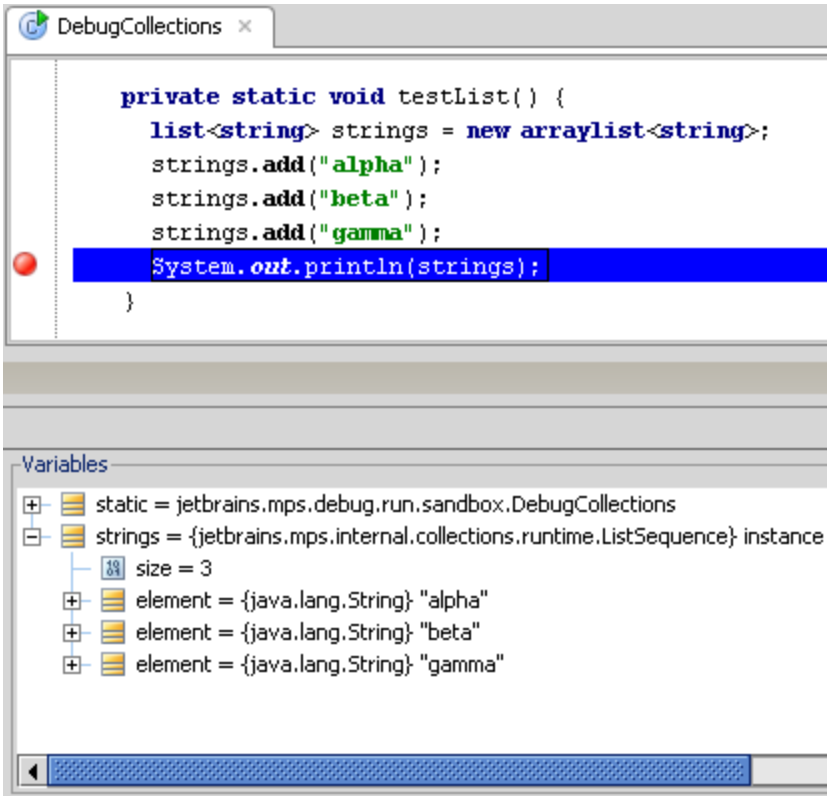


Like in a usual debugger session, you can make use of step over/into/out actions, exploring and exporting threads list and stack traces, or browsing variables.

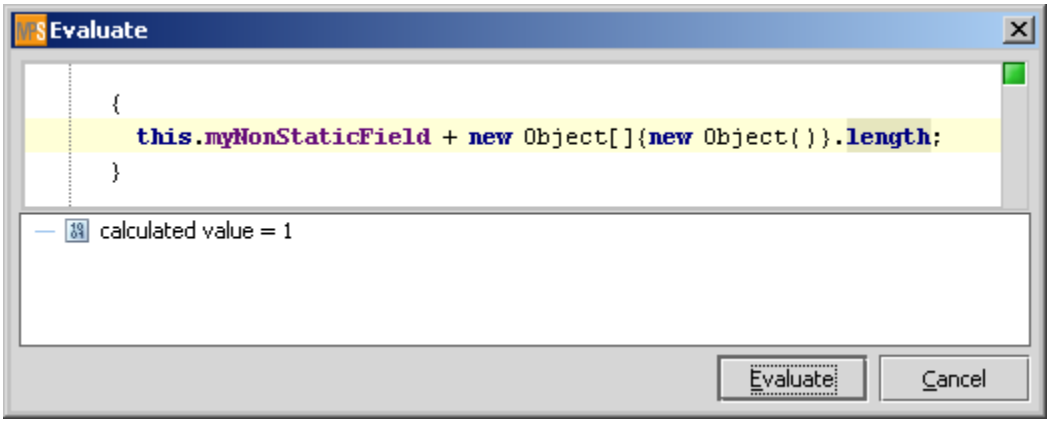




Now MPS offers custom viewers for when you want to view some variables' values in a more domain-specific and convenient way. Viewers specify which values they show in a specific manner and how to show them. MPS provides a special language for creating custom viewers.



The Evaluate Expression dialog window, available by pressing Alt+F8, can run any DSL queries based on context object instance.

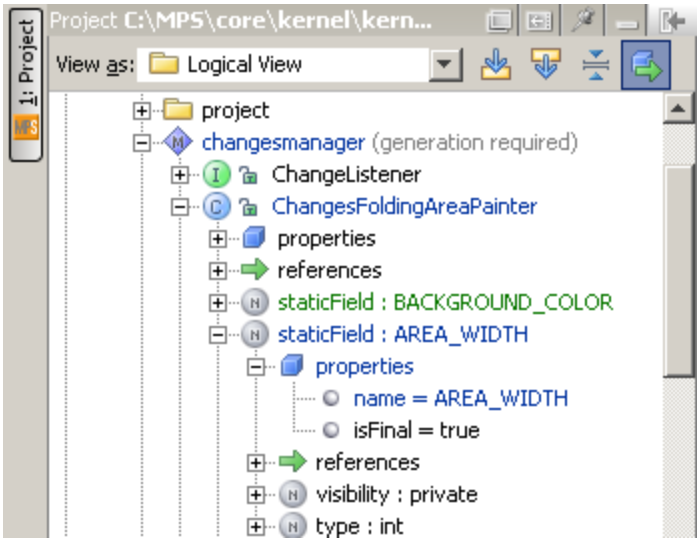


MPS 1.5 can debug any MPS Base Language (Java) code including various DSLs which reduce to MPS Base Language. In future versions we plan to finalize the API providing the possibility to debug any low-level languages other than Java.

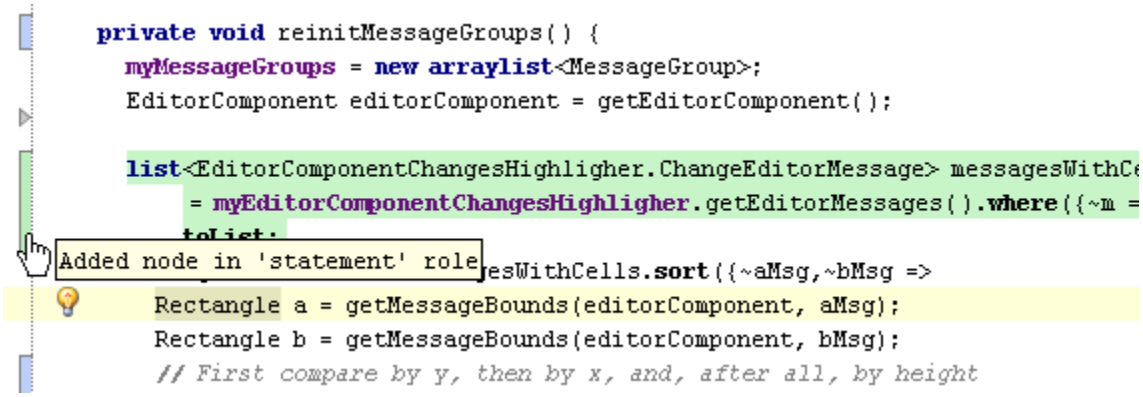
Improved version control integration

MPS 1.5 features enhanced version control integration, with many new highlightings for model changes:

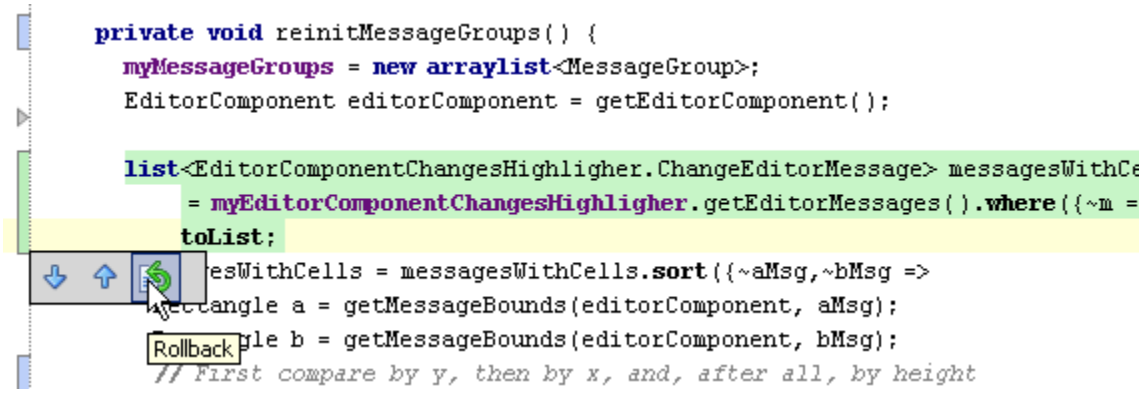
- You can distinguish between added/modified model elements and unchanged model parts by looking at different coloring of project explorer tree nodes and editor tab labels.



- Added and modified editor cells are highlighted with the background color.
- Special markers in left and right editor margins are associated with added/removed/changed cells.

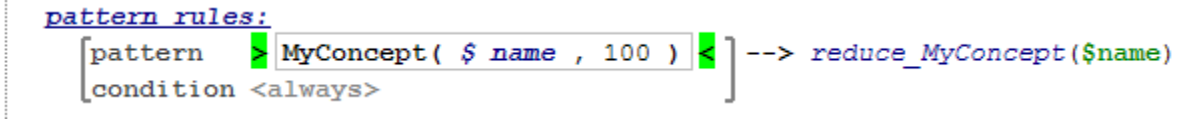


- Left editor margin markers are clickable, so you can navigate to the next/previous change or revert the current one.



Generator language improvements

MPS introduced a new possibility to create pattern rules in its generator mapping configuration description. In some cases it's much more convenient to specify a simple model pattern instead of using a reduction rule with complex conditions. Declared pattern variables are available in subsequent model queries through the generation context parameter.



Now you can declare template-specific parameters in the MPS template declaration header. These parameters should be specified when calling the template either from rules in mapping configurations or from another template. You can also pass pattern variables declared in pattern rule as template parameters.

main | reduce_MyConcept

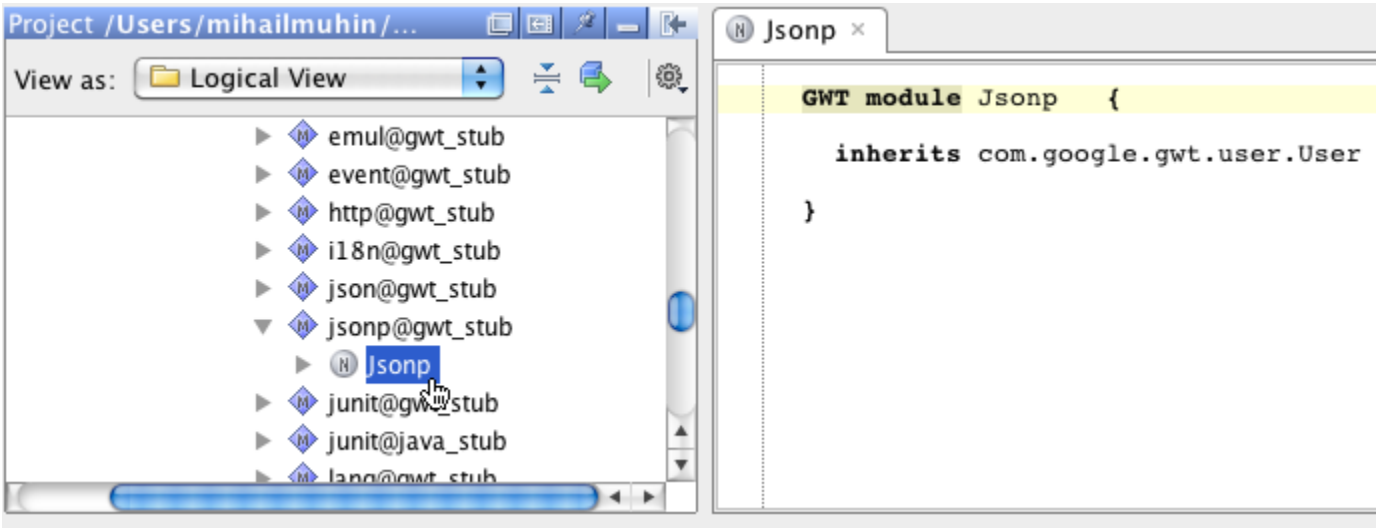
```
template reduce_MyConcept
input    MyConcept

parameters
p1 : string
```

Another useful generator feature is inline template with context: you can specify the template context for an inline template fragment just like in a standalone template declaration.

```
reduction rules:
  [concept    MyConcept ] --> content node:
  [inheritors false     ]     int i;
  [condition  <always>  ]     <TF [ int j = i + 2; ] TF>
```

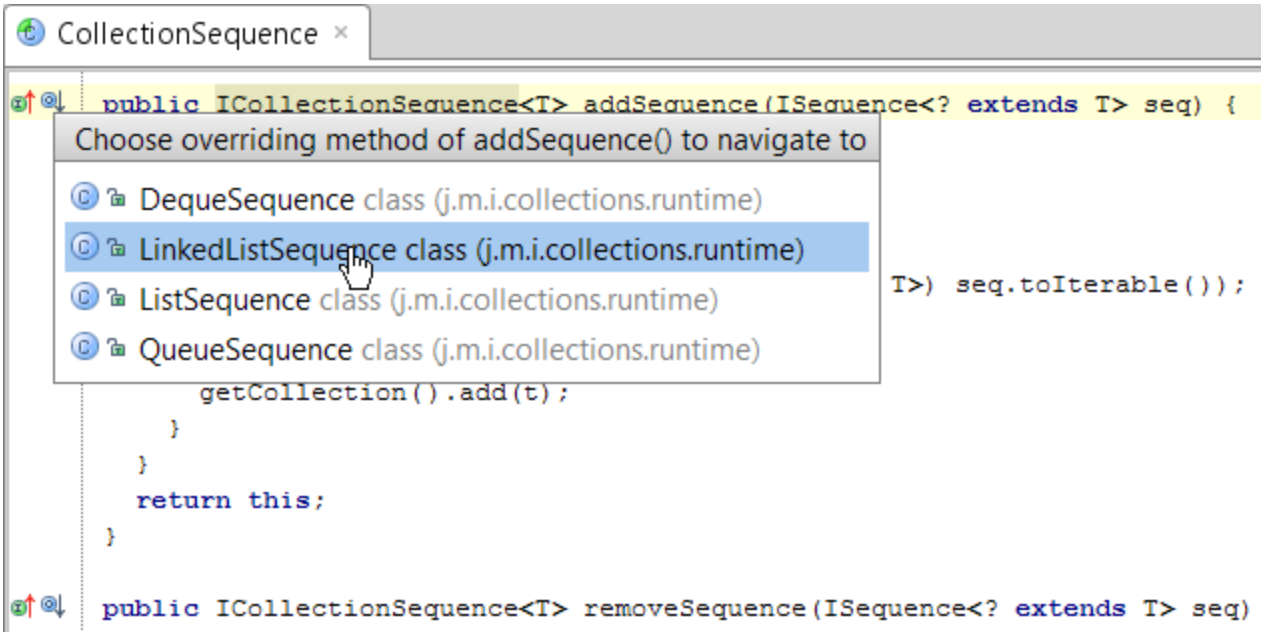
Integration with existing languages

MPS 1.5 features a new language aspect, "stubs." Use this aspect to describe how to load existing code written in any programming language and convert it to MPS model. You can then reference this code from any program written in MPS. This new language aspect was used in MPS to describe java stubs loading to MPS Base Language entities, as well as GWT stubs.



Editor features

Similar to an ordinary Java IDE, MPS 1.5 tracks extends/implements references in Base Language classes. Based on this information, it displays "implementing" and "overridden" markers in the left editor margin displayed next to the class/interface/method declaration. Click these icons to navigate to the corresponding method; a pop-up menu will be displayed if there is more than one available action.



Code completion is an essential feature of any MPS-generated editor. Usually you start typing and then press Ctrl+Space to see completion suggestions; by default, only those suggestions that match the typed prefix will be shown. Now, you can press Ctrl+Space again to explicitly show ALL possible completion suggestions.

The latest MPS Editor definition contains several minor improvements, including:

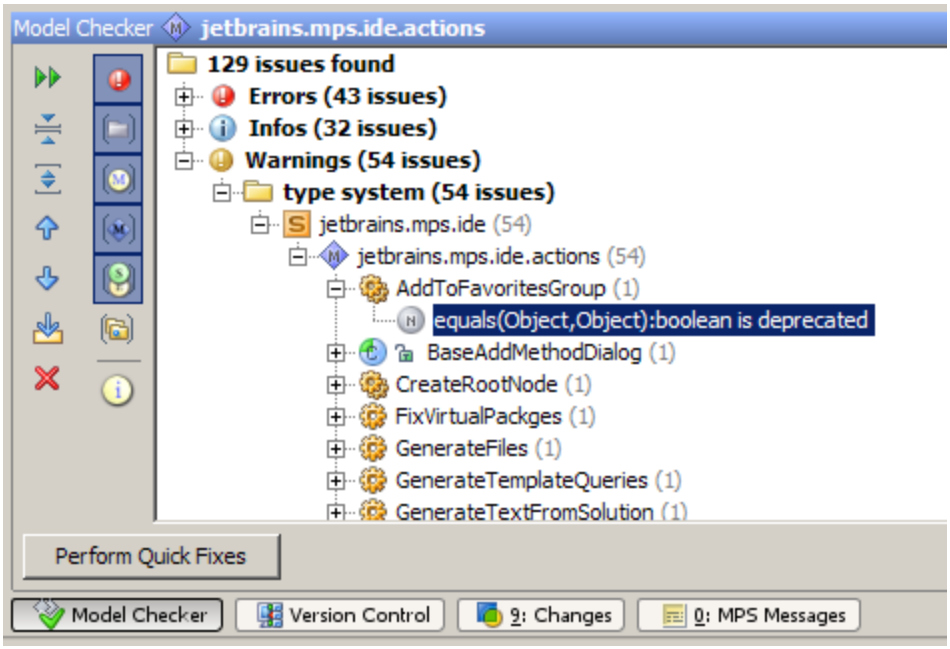
- Composite side-transform tag support. Use this when you need to associate more than one side-transform tag with the selected cell.
- New CopyPreProcessor/CopyPostProcessor for additionally handling model elements while copying/pasting. *Processors can be defined in the actions language aspect.

Model Checker

You can use newly added Model Checker functionality to invoke complete validation on the selected models or modules including:

- unresolved or invalid references detection
- typesystem-based validations
- model and module structure checks

Single view containing all validation error will be displayed in the end of check Models process. You can invoke Model checker manually by using pop-up menu on model/module or as a part of code generation and VCS commit changes process.



MPS core languages improvements

MPS 1.5 introduces some useful changes to different MPS core languages, listed below by language. Please refer to MPS 1.5 documentation for more details.

- SModel language
  - new implicit select operation
  - operations specific for LinkList type were deprecated - use collection operations instead
  - improved set of reflective operations
- Constraints
  - new can be ancestor constraint
  - new search scope validator in search scope constraint
- Collection language
  - alsoSortBy operation
  - reduceLeft/reduceRight operations
  - foldLeft/foldRight operations
  - Deque, Stack, Queue
  - Custom Containers
  - Primitive Containers
  - Collections Runtime moved to function types

Memory and performance improvements

Being a reasonably complex framework, MPS has high performance requirements which the MPS team is constantly working to lower. The current MPS upgrade contains a number of changes as well as some configuration options to achieve improved overall system performance, including:

- faster start-up time
- faster class reloading
- reduced memory consumption

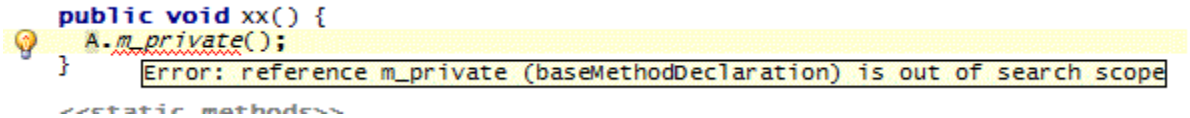
You can switch on/off the following code generator options:

- Generate in parallel (available in Strict mode only)
- Save generation dependencies (experimental) in this mode next code generation session will not include any model roots that weren't changed
- Model generation performance report

Other improvements

MPS 1.5 also provides other improvements:

- A DSL for creating and editing GWT client application manifests
- A DSL for adding Type Extension Methods
- Several Base language extensions
    - Checked dots
    - Overloaded operators
    - Custom constructors
- Improved error reporting - references out of search scope are marked as an error in the editor



MPS source code repository is publicly available

MPS is an open-source project, and starting form version 1.5 it has a publicly available Git repository. Use it to check out the current development state, browse history, stay up to date with the latest features and fixes, and of course contribute your patches.