Refactoring

Refactoring support is a very important part of the IDE functionality for every mature language. MPS provides several facilities to simplify development of refactoring actions.

Simple refactorings can be written using the jetbrains.mps.lang.refactoring language. After writing a refactoring using this language you should make it available to the users and invokeable by using the following steps:

1. Create a UI action for the refactoring. This is a simple action written using the plugin language. You can check the Rena me action from jetbrains.mps.ide.platform.actions.core as an example of proper refactoring action registration.
2. Create ActionGroupDeclaration for the refactoring and make it modify the jetbrains.mps.ide.platform.actions.NodeRefac toring action group at the default position.
3. Add a refactoring keystroke that triggers the newly created action to a KeymapChangesDeclaration.
4. Add an `isApplicable` clause to the action created; usually it is just an **`is applicable refactoring< >`**`()` call.
5. Add an `execute` clause to the action created. All user interactions should be performed here. At the end it is necessary to execute the refactoring with the prepared parameters by calling **`execute refactoring< >`**`()`.

Your refactorings may additionally log information about the performed changes and update the usages in other project that depend on the refactored project. In such cases you should use the Refactoring Participant framework. This way of creating refactorings consist of the following steps:

1. Create a UI action for the refactoring. Here you can show dialogs and retrieve any necessary data from the user, register the action in an action group and optionally create a keymap.
2. In the execute clause of the action call RefactoringProcessor.performRefactoringInProject() and pass all the data related to all the suggested changes.
3. For each kind of the possible usages create a subclass of RefactoringParticipant that handles its kind of usages, i.e. searches for usages and updates them
4. Create an extension point and register all the participants in that extension point.
5. To support updating usages in other dependent projects, participants should be additionally registered in the Persistent RefactoringParticipantsEP extension point. A special participant for creating the refactoring log is required (see MoveNod eRefactoringLogParticipant for implementation details).