Refactoring Tools

Fabio B. De Oliveira e
Flávia M. B. Greggio

Smalltalk

Refactoring Browser

The *original* refactoring power tool

Refactoring & Smalltalk

- Refactoring has been a part of Smalltalk culture since the beginning.
- Simple syntax.
- Dynamic typing.
- Ease of Interpretation/Compilation.
- All source are avaiable and searchable.

Features

- **Refactorings:** Automatically perform some behavior preserving transformations such as abstracting references to an instance variable.
- **Old methods:** No more accidentally accepting changed methods. Whenever you change a method in one window, all other browsers viewing the methods will turn their code tools red until you update their code or accept the method.

Features

- **Lint:** Automatically look for over 60 types of common Smalltalk bugs in your program.
- **Rewrite tool:** Specify new rules to make source to source transformations. For example, you can create a rule to automatically convert uses of "aCollection size == 0" to be "aCollection isEmpty".

Java

JRefactory

- This tool comes as a command line option with GUI or without, and as a plugin for the JBuilder, NetBeans, and Elixir IDEs.

Features

- The UML diagrams are reverse engineered from the .java files, so there is no fear that they will get out of date. The class diagrams are the main interface to the system. The user inteface for the refactory is described here.

Refactorings

- Move class between packages (repackage)
- Rename class
- Move method
- Extract method
- Rename Parameter

Refactorings

- Add abstract parent class
- Add child class
- Remove empty class
- Extract interface
- Push up field
- Push down field
- Rename Field

Refactorings

- Push up method
- Push up abstract method
- Push down method

JFactor

- A refactoring plug-in for Visual Age and JBuilder

Method Refactorings

- **Extract Method**
- **Rename Method Variables**
- **Introduce Explaining Variable**
- **Inline Temp**
- **Replace Magic Number with Symbolic Constant**

Method Refactorings

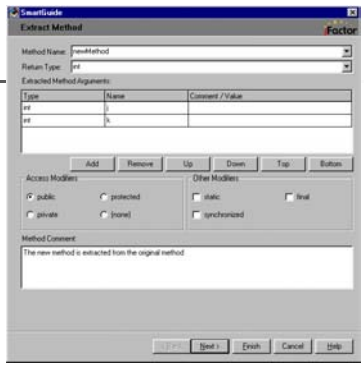
- **Inline Method**
- **Rename Method**
- **Safe Delete Method**
- **Pull Up Method**
- **Push Down Method**
- **Introduce Foreign Method**

Field Refactorings

- **Rename Field**
- **Pull Up Field**
- **Push Down Field**
- **Encapsulate Field**

Class Refactorings

- **Extract Superclass**
- **Extract Interface**



XRefactory

Features

- Full integration with ***Emacs*** and ***XEmacs***.
- Working on ***Unix*** (*Linux, SunOs-Solaris, HP-UX, ...*) as well as ***Windows*** platforms.
- Support for both ***C*** and ***Java*** languages (including JNI).

Features

- ***Source browsing*** based on our own tag implementation taking care about multiple preprocessings, scopes, classes, accessibility, overloading and polymorphism.
- Designed to work with ***huge projects***. Xrefactory is indexing complete jdk 1.4 classes (1 million lines of code) in 2 minutes and Linux 2.4.7 sources (3 millions lines) in 15 minutes. It is able to update index by reparsing only modified files.

Features

- ***Refactorings*** for method (function) extraction; renaming of packages, classes, parameters, variables, fields (structure records) and methods (functions); insertion, deletion, shift and exchange of parameters; field and method moving; pushing down and pulling up fields and methods; encapsulate field; and more. Refactorings are safe with detection of possible conflicts.

Features

- Very solid implementation of intellisense ***code completition***. Recognizing around 15 different contexts with completions auto-updated after each modification of source code without need of recompilation.
- ***Multiple projects*** support with project autodetection.

Features

- Interfacing ***Emacs IDE*** functions from *compile.el* and *comint.el* packages.
- ***Symbol retrieving*** functions.
- Ability to index whole ***jar archives***.
- ***Javadoc documentation*** available at a fingertip.
- Possibility to generate ***HTML documentation***.
- Full ***undo***.

IntelliJ IDEA

- IntelliJ IDEA is a full-featured Java IDE with a high level of usability and outstanding advanced code editing and refactoring support .

Refactorings-rename:

- Package
- Class
- Method
- Field
- Parameter
- Local Variable

Refactorings- move:

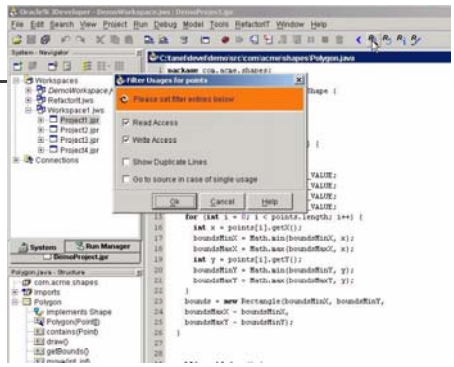
- Class
- Package
- Static Field
- Id
- Static Method
- Static Inner Class
- Inner Class to Upper Level

Refactorings

- Change Method Signature (add, remove,rename,reorder method parameters)
- Extract Method
- Introduce Variable
- Introduce Field
- Inline Method

Refactorings

- Inline Local Variable
- Replace Temp With Query
- Extract Interface
- Extract Superclass
- Encapsulate Fields
- Convert Local Variable to Field
- Convert Anonymous Class to Inner



Other languages

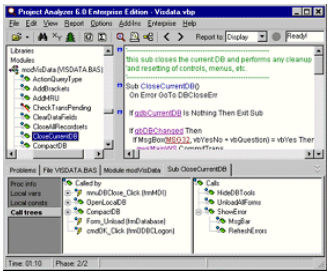
Visual Basic- Aivosto Project Analyzer

- A Visual Basic code optimization utility. While not strictly a refactoring tool, it does include a lot of related functionality. For example, it can automatically do Encapsulate Field, Remove Parameter, and Hide Method.

Automatic code review

- Dead code detection. Remove unused procedures, variables, constants etc. Decrease your .exe by up to 100s of kB .
- Optimization. Find inefficient code. View suggestions for better syntax.
- Style. Enforce programming standards and make that spaghetti code readable again.
- Functionality. Detect run-time issues, such as form resizing, error handling and tab orders .

Screenshot



Phyton- The Bicycle Repair Man

Named after...

- The Bicycle Repair Man was a superhero in a Monty Python skit, his special power was repairing bicycles. The Python Refactoring Browser is supposed to be good at only one thing as well.

Refactoring

- The Refactoring Browser is smart enough to rename *every* reference to your class, method or variable. If you've ever renamed a variable and broken classes in widely scattered parts of your system, you might be happier using a Refactoring Browser.
  A Refactoring Browser operates on any of method, function, class, or variable. It can add, delete, rename, move up down or sideways, inline and abstract. There are some operations that are specific to one of the three types, such as abstracting a variable into accessors, or turing several lines of code into a separate method.