Welcome to MDT

The MDT (Magik Development Tools) is a flexible and intuitive visual Integrated Development Environment (IDE) for Smallworld™ Magik developers. MDT is the fastest and smartest way to deliver your Smallworld™ solutions.

Vision
The Smallworld™ platform grows and Smallworld™ applications become more complex and involve various technologies. Developing, integrating and maintaining such systems require a flexible development environment for Smallworld™ Magik-based solutions. The main objective of the new IDE for Smallworld™ Magik was to optimize the development process for Smallworld™ projects using an industry open source standard Eclipse (www.eclipse.org).

Powerful Suite
The MDT suite is a powerful combination of existing approaches and modern programming features. The MDT framework supports business needs such as efficient and rapid Smallworld™ application development as well as scalability and ability to build extensions and advanced tools based on the Eclipse plug-in technology. MDT maintains and supports the whole Smallworld™ project lifecycle starting from designing through coding, debugging as well as integrating, testing and deploying.

Introduction
Dear Readers,

We are proud to present to you the updated MDT White Paper for the year 2012. It has been refreshed and reviewed to match the newest release of MDT 3.1. In autumn last year we have published MDT 3.0 which introduced a couple of most wanted functionalities extending our tool with support for MUnit testing and the ability to create models and architectures in accordance to UML2 language standards. With the addition of Mylyn Context Connector extension, MDT transformed from a solely development tool into an enterprise multitasking machine dedicated to entire development teams. Hopefully you will find our efforts to be of great value to You, Your projects and the overall quality of Your Smallworld applications.

With the ongoing support coming from GE Energy, Your kind comments and words of appreciation we find our work very rewarding and ourselves motivated with heads full of ideas for the future.

With best wishes,
MDT Team

Table of contents

Welcome to MDT ............................................. 2
Architecture .................................................. 3
Working with Magik Projects .......................... 4
Magik Code Editing ........................................ 8
Interacting with Smallworld™ GIS .................. 11
Browsing Magik Code .................................... 13
Debugging Magik Programs ............................ 15
Extending MDT .............................................. 18
Architecture

Core
Core plug-in mostly contains two fundamental MDT components: Magik language parser and Smallworld™ GIS platform meta-model.

The Magik parser was created on the basis of the Magik language specification invented and written in ANTLR language (www.antlr.org). It covers syntax of all supported Smallworld™ GIS versions and has the ability to detect syntax errors in Magik code.

The Smallworld™ GIS platform meta-model was created with the use of Eclipse Modeling Framework (www.eclipse.org/emf). Meta-model can model basic platform elements such as: product, module etc. and elements of Magik files such as class, method etc. Meta-model has mechanisms for validating of result models.

User Interface
User Interface plug-in provides visual elements of MDT environment that allow working with resources of Magik projects. It contains:

- editors for important Smallworld™ GIS files such as: Magik files, load lists, product and module definitions and messages files,
- views that can navigate a hierarchy of information or display properties for any available element in Magik project,
- wizards that simplify and automate typical developer’s activity.

One of the most important features of this plug-in is depicting logic structure of Magik projects and presentation of the validation results for elements and files of Magik projects and GIS session configurations.

Editor, views and wizards are grouped in perspectives which support work organization with Magik projects. The perspective defines initial set and layout of user interface windows and provides a set of functionalities that allows user to execute tasks.

Launcher
Launcher plug-in provides functionalities that allow the user to work with Smallworld™ GIS sessions. Implementation enables defining, simultaneous starting and working with many sessions at the same time. Expanded validation mechanism enables the detection of errors in session’s configuration in the defining stage and during session start.

The plug-in provides specialized GIS console that allows the control of GIS sessions. Additionally, the plug-in provides series of views, wizards and editors supporting session management.

Search
The basic task of the Search plug-in is to search the Magik code both in Magik projects and in GIS sessions. Additionally, this plug-in also makes available information from other important Smallworld™ GIS files: load lists, product and module definitions and messages files. Just as in the Magik code case, information is delivered both from Magik projects and GIS sessions.

Search results can be presented in many available perspectives and views which enable easy and effective analysis of these results.

Debugger
The most advanced feature of the MDT is its Magik debugger which enables to detect and diagnose errors in running GIS sessions and thus incredibly facilitates the development process of Smallworld™ applications. Debugger provides all standard debugging functionalities including the ability to:

- run GIS session in debug mode,
- manage breakpoints (which includes adding, removing, disabling breakpoints),
- suspend and resume program execution at a breakpoint,
- perform program execution step by step,
- inspect variables and values,
- modify the state of the program and values during execution.
Working with Magik Projects

Magik Development Perspective

Magik Development Perspective is a fundamental perspective provided by MDT. This perspective groups all the functionalities designed for working with Magik projects. Within this perspective the following set of windows is available:

- **Product Explorer view** - shows Magik projects which a user is working on,
- **Magik editor** - used for editing Magik files,
- **Problems view** - gathers and shows problems detected in files and elements of Smallworld™ GIS platform, contained in Magik projects and in GIS session configuration,
- **Outline view** - a view connected with Magik editor which shows Magik code elements present in edited Magik file,
- **Sessions view** - shows GIS sessions available in the environment,
- **Console view** - GIS console which allows controlling the running GIS session,
- **Class Browser view** - allows the search of Magik code available in GIS session.
**Product Explorer**

Product Explorer view, shown by default in the Magik Development perspective, is the most important navigation view available in MDT. In comparison to other standard navigation views (Navigator and Project Explorer), this view provides a Magik-specific view of the resources that the user is working on. The element hierarchy displayed in the view is derived from physical folders and files structure. It also contains additional information and settings for Magik project.

The view shows the Smallworld™ GIS platform element hierarchy of the Magik projects. For each project, its products, modules, source folders, Magik files, referenced products and other important resources are shown in the tree. The user can browse the tree structure and edit the files.

Each element presented in the hierarchy possesses its own set of specific commands. In this way you can easily: load a GIS module to the session, configure a load list or transmit a single method from Magik file to the session etc. These operations can be executed directly from the view without the need to open additional windows (e.g. Module Manager) or editors.

**Transmitting code to any predefined session**

MDT Professional Edition now allows to transmit the code to sessions that are not assigned to current project. Source can be sent to one or multiple running sessions. Appropriate commands are accessible from the main menu, Product Explorer view, Magik editor and Outline view.

**Magik Projects Builders**

Magik project builders are used to detect problems in Magik project resources. There are two kinds of builds:

- **Incremental build**: uses a “last build state,” maintained internally by the builder, to do an optimized build based on the changes in the project since the last build. That kind of a build is usually automatically triggered after making changes in the project resources (saving changed file, removing or adding resources).

- **Full build**: performs a build from scratch. It treats all resources in a project as if they have never been seen by the builder. It is usually triggered on user demand after invoking one of the build or clean commands.

When resources are built, the progress indicator is shown which allows to observe the details of building process in the Progress view. Builders can be, if necessary, freely switched on or off in selected Magik projects.
Import/Export of Resources

Import wizard is available in MDT environment and allows importing and working with existing resources: Magik projects, products, modules, Magik files etc. Wizard allows to import resources directly from files system and from Archive files.

Export wizard is available in MDT environment and allows the export of Magik projects, products, modules, Magik files etc. from resources environment that user is working on. Wizard allows export resources to files system and to Archive files.

Project Dependencies

GIS session image may contain Magik code that comes from different sources. That is why it is hard to collect all the sources that user can work with in the environment all at once.

MDT provides External File wizard that helps you to get access to Magik files which are not present in your workspace but contain important code indispensable for your work. When accessing external files with the wizard you can decide whether you want to simply open them or open and associate them with a Magik project for further use.

Local Edit History

A local edit history of a file is maintained when a file is created or modified. Each time a user edits and saves a file, a copy is also saved so that the user can replace the current file with a previous edit or even restore a deleted file. A user can also compare the contents of all the local edits. Each edit in the local history is uniquely represented by the date and time the file was saved.

Only files have local history; projects and folders do not.
Comparing Files

MDT allows easy and comfortable comparison of two files and two versions of the same file. File versions can be stored in the local edit history or in any VMS (Version Management System).

When a comparison is performed, compare editor appears in the editor area. The differences between files are highlighted allowing the user to browse and copy changes between the compared resources.

Wizards new

Wizards are used to guide the user through a sequenced set of tasks. MDT provides a number of wizards for creating most of important elements of Smallworld™ GIS platform (products, modules, Magik files etc.), exporting and importing Magik project resources (file and folders), exporting and importing MDT configuration data (general preferences, GIS session configurations etc). Special set of wizards helps to work with the Magik code for example in defining new code element (exemplars, methods etc.), code refactoring and many others.

With MDT 3.0 new wizards have been added for easy and semiautomatic creation of Magik elements such as Mixins and Slotted Exemplars. With those new wizards you can now create entire blocks of code including automatic creation of method stubs, pragma statements and comments.
Magik Code Editing

Magik Editor
Magik editor is a handy and powerful tool for editing Magik files. It makes code development much easier and faster than in any other available editor for Magik. It is supported by many functionalities which accelerate code edition and navigation.

The major editor features are:

Syntax Colorizing
Aesthetic colorizing of Magik lexical elements for better code readability.

Templates
The ability for defining and reusing frequently used Magik code chunks.

Emacs-like Auto Indentation
Code formatting consistent with Emacs editor rules. Changes in Emacs-Like Auto Indentation mechanisms now allow to customize the formatting style so it is identical with Emacs and recognize the differences in code formatting depending on the version of Smallworld GIS.

Magik Occurrences
Functionality responsible for searching and marking occurrences of selected symbol or identifier in Magik source file.

Smart Caret Positioning
Ctrl + ‹ and Ctrl + › stop at the beginning/end of Magik words. Their 'shifted' equivalents extend the selection in the same way.

Code Folding
Enables folding/unfolding of Magik compound statements.

One of the functionalities of the Magik editor are code problem markers. The contents of a file are checked on the fly, during code editing and detected problems (e.g. syntax errors) are instantly marked in the editor. The problems are marked in a few different ways, so they can be easily found even in case of big files. The code checking is independent of the Smallworld™ GIS platform and does not demand running of the GIS session. Magik editor also offers Quick Fix functionality for the frequently occurring problems, which presents one or more possible fixes that can be automatically applied in order to resolve particular problem.

Outline
The Outline view displays the structure of the file that is currently open in Magik file editor, and lists structural elements that can be accessed fast and easily. The content of the Outline view is editor-specific. For a Magik source file, the structural elements are exemplars, slots, methods, shared variables, shared constants, procedures and blocks. Every element is shown in outline with a specific set of icons which give user information about its type and modifiers. With the use of Outline, user can navigate the source code much faster.
Code Assist

Code Assist is a ticket for easier and more flexible style of programming. It is a feature that prompts the user with a list of valid alternatives for completing current line of codes. Code Assist can be accessed by the user or triggered automatically after specific conditions. The resulting dialog provides a context sensitive coding help for a user by making available a listing of all applicable Magik code elements for the location where code assist was activated. For a Magik editor, the code elements are methods, templates, keywords etc. Types of code elements are grouped and displayed separately. There is a possibility of cyclic switching between groups. User can choose which groups of proposals should be displayed and in what order.

Code Templates

Code Templates are used by actions that generate code. It is a very powerful functionality which improves the level of coding style. There are two kinds of code templates in Magik editor - “Comments” and “Pragmas”. User can define which template structure, will be applied to specific code element like method, exemplar definition, block, procedure etc. Templates may contain variables that are substituted when the template is applied. All code templates can be defined separately for each of the code element types. Additionally, each Magik project can possess specific set of code templates. With use of code templates during Magik code developing, the programmer is able to add defined templates for any given element very quickly.

Transmitting Magik Code

During work with Magik editor, user can, at any time, transmit code to any running GIS session. A user can transmit whole file, selected region or any code element in a file. Even modules and projects can be transmitted with one mouse click. For safety reasons user may choose to be warned when origin Magik file is dirty. Each problem in transmitted code is automatically reported to user via special window which shows up after transmitting operation.
Type Overview

Type Overview is a kind of read-only editor that allows to see the source code of an entire class in case it is spread between many files - it occurs quite often if a particular class is frequently patched. Type Overview collects all information directly from the session, so the shown code is always up-to-date. It groups the methods coming from the same files and marks them with different background color so they are easily distinguished. It is not possible to make changes directly in the Type Overview but you can easily jump to the source file of any method. Moreover Type Overview is equipped with features like Outline and hyperlinks for simpler navigation.

Messages File Editor

Messages File Editor is dedicated to handle .msg files. It has all common editor features like Outline, syntax colorizing, smart caret positioning, on the fly content validation mechanism etc.

Draft Editor

Draft Editor is a different kind of common Magik editor created to safeguard files from unwanted changes. Draft Editor is used by default to open files from Project Dependencies or write-protected files. In case you would like to save your changes anyway a local copy of the file will be made without affecting its original source. Files opened in Draft Editor are easily distinguished by the “glasses” icon overlay shown in editor tab.
Managing GIS Sessions

Managing GIS sessions in MDT is one of the greatest enhancements of the environment. The most important feature is the ability to handle multiple GIS sessions based on defined runtimes. The architecture of runtimes and sessions introduced in the MDT provides a very flexible way of managing access to the Smallworld™ GIS environment. Runtime in the MDT is a definition of the Smallworld™ GIS runtime environment i.e. core product directory and some supporting executables required by the MDT. The MDT may support many runtimes at the same time.

Session in the MDT is a definition of the Smallworld™ GIS image and assigned runtime used to run it. Each Magik project may have assigned single session in which all code held in the project is executed. GIS sessions and runtimes are liable to validation process which is responsible for finding possible problems related to their configurations. Problems are being collected and reported in appropriate views. The MDT may support many sessions at the same time. Each session may be run independently of others in standard Run mode or in Debug mode which gives user ability for debugging Magik code related to the given session.

Session and Runtime configurations can be easily exported/imported to single files and shared between team members for better teamwork synchronization.

The MDT provides three kinds of sessions:

Session
An ordinary session which configuration (image file location, session launcher parameters, environment variables) is entirely stored in the MDT.

Session Link
It links to the configuration stored within existing environment and GIS aliases files. The MDT stores only the locations of both files and name of a particular alias to run. This way of session managing gives opportunity for simultaneously work in both MDT and Emacs. Thus it makes also session administration easier for both environments.

Command Line Session
It allows to run complex GIS session configurations based on executable files such as batch files or gis.exe. Command Line Session simply starts executable file with declared working directory and parameters. MDT provides a dedicated wizard that helps to create new session type configuration and a session editor which allows you to modify created configurations.

Session view

The Sessions view shows all sessions and runtimes defined in the MDT. It contains two tabs with tree structures of available sessions and runtimes. The Sessions tab presents information about the session type, associated runtime and projects, as well as current session status and image file location. The Runtimes tab presents information about runtime type, associated session, core product and session launcher executable locations. If there are any problems after the validation of a given session or runtime configuration, then the appropriate error markers are being presented. With the use of Sessions view, user can add or delete a session, start or stop selected session or open it for editing. Sessions can also be run in Debug mode which enables Magik code debugging mechanism for further use.
GIS Console

The GIS Console provided by MDT is a complex view which fully reflects all known Emacs’ console functionalities and introduces many new ones.

It has many great editing features such as:

- Automatic keywords underscoring – automatic underscoring of Magik keywords while typing a code,
- Automatic "$" appending – functionality for automatic adding of "$" execute sign after complete Magik statement,
- Double-click strategy – double-click strategy for selecting Magik identifiers, symbols etc.,
- Multi-line statements edition – ability for edition multi-line Magik statements in console,
- Code browsing – ability for browsing for selected code.

Persistent Command History

One of the extra features provided by the GIS console is persistent command history. Each defined Smallworld™ GIS session may have its own command history. The history stores set of commands which have been executed within the console. All histories are persistent - they are restored with every started session. The MDT provides also a special view which allows user to move/copy particular commands among the histories.

The commands may be accessed within the console in three ways:

- by means of Command History dialog,
- by means of a set of shortcuts,
- by means of Code Assist.

Code Assist

The Code Assist in GIS Console has all the features of Code Assist of the Magik Editor. Additionally, it provides an easy access to commands stored in commands history of a console.

By means of Code Assist a user can easily complete line of code, insert predefined template or a command from history into console.
Browsing Magik Code

Magik Search

The Magik Search offers all searching options available in an Emacs’ Class Browser. The Magik Search is fully integrated with Eclipse platform and thus it is familiar to Eclipse users.

The Magik Search is more “static” in the way of making queries than the Class Browser. It also differs in the way of presenting searching results, which is more aesthetic and readable. One of the great advantages of the Magik Search is a searching history. The history gives the ability of remaking queries without the need of entering all searching options again.

The Magik Search allows user to make queries within any running GIS session.

Class Browser

The Class Browser view is very similar to the Emacs’ Class Browser and thus is more familiar to Emacs users. The Class Browser view offers all searching options available in Emacs’ Class Browser.

The MDT supports the Class Browser with some extra functionalities such as: switching between multiple running sessions, distinguishing search results without accessible source files and configurable regular expression template for search queries.

Magik File Search

The File Search is a very advanced textual search facility. It supports many options for search queries such as: regular expressions, file name patterns, case sensitive etc. The searching scope may be limited to the: whole workspace, enclosing projects and selected files. You can choose to include project dependencies in your search. Magik File Search provides a searching history.
Browsing Magik Code

Magik Browsing

The code in Magik Project or GIS session can be searched using special Magik Browsing perspective. The perspective offers all searching options available in Emacs’ Class Browser. The perspective combines the ability to quickly search and easily access search results thanks to the use of graphical user interface.

Navigation Dialogs

To navigate the code you can quickly jump to other locations using a few dialogs. Open Declaration dialog allows you to filter element declarations and open selected one in Magik editor. Open Member dialog and Open Type dialog can be additional help to navigate swiftly.

Hierarchy View

Hierarchy view in MDT Professional Edition supports now both children and parents hierarchy. In addition to standard tree layout, the view has also graphical layouts that can be convenient in some situations.

Also new actions have been implemented allowing to display the hierarchy in all MDT views and editors. Navigation dialogs have been extended with ‘Open Type in Hierarchy’ action that allows you to choose the type to display.

Type Hierarchy

Magik Type Hierarchy perspective is designed for exploring all types of hierarchies. The perspective consists of the Hierarchy view and Magik editor. It can be opened on exemplars and exemplars members (methods, slots etc.) from various MDT views.
Debugging Magik Programs

Debug Perspective

Debug perspective is designed for debugging Magik programs. It includes a Magik editor area and a set of views which present aspects of the program being debugged. In standard configuration, when GIS session is started in debug mode and program is in breakpoint stage, the program execution is suspended and user is asked if he wants to open Debug perspective.

Starting from MDT version 2.0 debugging of Smallworld™ 4.2 applications is also possible. Also a couple of significant changes and improvements have been made to Debugger mechanism in recent MDT releases.
Magik Stack Trace Console

A new type of a console has been added to MDT. The Magik Stacktrace Console displays a Magik stacktrace in a nicely formatted manner, with hyperlinks, which lead to the definitions of the stack trace line elements. You can paste stack trace from GIS console and follow created hyperlinks. A stacktrace content can also be edited.

Debug View

Debug View allows user to manage the debugging or running a program in the environment.

It displays the stack frame for the suspended threads for each target that the user is debugging. Each thread in the debugged program appears as a node in the tree. The view displays the process for each target the user is running. If the thread is suspended, its stack frames are shown as child elements.

Breakpoints View

Breakpoints View lists all the breakpoints that the user currently has set in his workspace. User can double-click a breakpoint to display its location in the editor (if applicable). User can also enable or disable breakpoints, delete them, add new ones, group them by working set, or set hit counts.
Debugging Magik Programs

Variables View

Variables View displays information about the variables associated with the stack frame selected in the Debug View. While debugging a Magik program, variables can be selected to have more detailed information displayed in the Detail Pane. In addition, objects can be expanded to show slots that the variable contains.

Displaying logical structures in Debugger

New debugger Logical Structures feature allows to present alternate structures for common types in the debugger views. It is very useful in case of collection objects, you can easily access their contents without having to look into their implementations. Showing logical structures is accessible from different views and dialogs.

By default MDT provides logical views for basic collections. You can define more logical views for other types using Logical Structures preference page.

Expressions View

Data can be inspected in the Expressions View. User can inspect data from a scrapbook page, a stack frame of a suspended thread, and other places. The Expressions View opens automatically when an item is added to the view. Entries in the Expressions View can be selected to have more detailed information displayed in the Detail Pane. When debugging a Magik program, data that contain variables can be expanded to show the variables and the slots the variables contain.
Extending MDT

The abilities of MDT environment can be extended by functionalities supplied by ASTEC or by other producers. Third party plug-ins have to be available as Eclipse plug-ins in accordance to specifications of Eclipse platform.

ASTEC has developed a few dedicated plug-ins supporting MUnit testing, UML modeling and task-oriented work with Mylyn. We also recommend third party plug-ins provided by Eclipse community that allow for integration with various Version Control Systems, XML editor etc. You can find detailed information and files to download on www.mdt.net pages.

MUnit 3 Support new

MUnit 3 Support plug-in provides modern mechanism that allows to write and run tests for applications written in Magik language. It is responsible for automatic generation of test cases, running tests and presenting their results. Thanks to MUnit tests, specific parts of code can be automatically tested. It is very helpful in discovering errors and mistakes already in code production phase.

MUnit 3 Support is a dedicated add-on to MDT and cannot function without it.

UML2 Modeler new

UML2 Modeler plug-in is a sophisticated tool to be used for modeling of whole systems with the use of UML language. Its main functionality is to model system architectures in MDT environment as well as dynamic generation of Magik source code according to previously created models. It helps to combine the work of architects working with UML language with the tasks given to Magik developers. Architects can extend existing systems by new elements by creating specific artifacts for developers directly from MDT environment.

UML2 Modeler is a dedicated add-on to MDT and cannot function without it.
Mylyn Context Connector

Mylyn Context Connector, also known as Mylyn Bridge allows for integration of Mylyn with MDT. Mylyn is an integrated Task Management system that reduces information overload and makes multi-tasking easy. Mylyn Context Connector introduces all benefits of task-focused style of work provided by Mylyn into MDT environment.

Version Control Systems

MDT can be extended with plug-ins provided by Eclipse Community that allow for integration with various Version Control Systems. The two most popular systems are Concurrent Versions System (CVS) and Subversion (SVN).

Eclipse CVS Client provided by Eclipse Community is the most common extension for CVS. It fully integrates with Team Synchronizing perspective and provides mechanisms to synchronize resources with CVS repositories. The extension also provides an additional CVS Repository Exploring Perspective, used to manage and browse CVS repositories.

SVN extensions also synchronize well with Eclipse’s Team Synchronizing perspective. There are two basic extensions that support Subversion (SVN): Subclipse (by Tigris.org) and Subversive (by Eclipse Community). They both provide additional perspectives and views for managing and browsing SVN repositories.

MDT can be also extended to cooperate with many other Version Control Systems, like: IBM Rational Clear Case, Visual SourceSafe, Git and many more.

Team Synchronizing perspective is dedicated to help managing all incoming, outgoing or conflicting changes between the resources and a repository.