modelscope™

For Windows
Version 2.0

Getting Started

© 2003, 2004 Metamaxim Ltd

ALL RIGHTS RESERVED. NO PARTS OF THIS WORK MAY BE REPRODUCED IN ANY FORM OR BY ANY MEANS - GRAPHIC, ELECTRONIC, OR MECHANICAL, INCLUDING PHOTOCOPYING, RECORDING, TAPING, OR INFORMATION STORAGE AND RETRIEVAL SYSTEMS - WITHOUT THE WRITTEN PERMISSION OF THE PUBLISHER. PRODUCTS THAT ARE REFERRED TO IN THIS DOCUMENT MAY BE EITHER TRADEMARKS AND/OR REGISTERED TRADEMARKS OF THE RESPECTIVE OWNERS. THE PUBLISHER AND THE AUTHOR MAKE NO CLAIM TO THESE TRADEMARKS.

WHILE EVERY PRECAUTION HAS BEEN TAKEN IN THE PREPARATION OF THIS DOCUMENT, THE PUBLISHER AND THE AUTHOR ASSUME NO RESPONSIBILITY FOR ERRORS OR OMISSIONS, OR FOR DAMAGES RESULTING FROM THE USE OF INFORMATION CONTAINED IN THIS DOCUMENT OR FROM THE USE OF PROGRAMS AND SOURCE CODE THAT MAY ACCOMPANY IT. IN NO EVENT SHALL THE PUBLISHER AND THE AUTHOR BE LIABLE FOR ANY LOSS OF PROFIT OR ANY OTHER COMMERCIAL DAMAGE CAUSED OR ALLEGED TO HAVE BEEN CAUSED DIRECTLY OR INDIRECTLY BY THIS DOCUMENT.

PERMISSION IS HEREBY GRANTED TO PRINT THE ELECTRONIC FORM OF THIS DOCUMENT (PDF FILE) FOR THE SOLE PURPOSE OF USING IT TO WORK WITH THE MODELSCOPE SOFTWARE OBTAINED UNDER SEPARATE LICENCE AGREEMENT.

© 2003, 2004 METAMAXIM LTD.

CONTENTS

1 Introduction

ModelScope is a tool for exploring and defining the requirements and logical architecture of transactional business applications. Using ModelScope it is possible to create testable application models from simple logical descriptions of the objects in the application and how they behave.

The heart of ModelScope is a State Transition Diagram interpreter, which understands how events change the states of objects, what events an object can accept based on its state, and how events create and destroy relationships between objects. To provide a testable application model, ModelScope also provides a default user interface that allows events to be entered and processed, and a database that provides persistent storage of the created object instances.

ModelScope can be used to analyse a number of different kinds of application systems issues, for instance:

- To explore and define the logical scope and design of a new application or redesign of an existing application.
- To explore and define the behavioural requirements that a package must meet by customisation or parameterisation.
- To explore possible reconfiguration or wrapping of existing applications, e.g., to remove duplication of data or functionality, or to decouple workflow from transactional processing.
- To explore and define the requirements for the systems support required for new or redesigned business processes.
- To communicate aspects of application business purpose and logical design to a wider community

This Guide describes how to get started using ModelScope by building and running a simple model.

2 Organisation of this Guide

This guide is organised as follows:

- Section 3 lists the system requirements for ModelScope
- Section 4 describes how to obtain a User Key to run ModelScope
- Sections 5 and 6 describe how to resolve configuration problems that prevent ModelScope from running
- Section 7 describes how to get started using ModelScope by creating a simple model
- Section 8 describes the Instances File, which is used to make objects persistent between sessions
- Section 9 describes the Server Window and its menus
- Section 10 is a reference section covering configuration and customisation of ModelScope
- Section 11 lists known bugs.

3 System Requirements

ModelScope runs on a normal PC using Microsoft Windows. Although designed to run on a single PC, ModelScope is internally configured as a client/server system. The server is a Java program, and the client runs in a browser (Microsoft Internet Explorer).

To use ModelScope requires a PC with the following:

- MS Windows: 2000, ME or XP. (See Note 1 below)
- Java Software Development Kit: JSDK version 1.2.0 or later. (See Note 2 below)
- TCP/IP
- MS Internet Explorer: version 6 or later or Netscape Navigator: version 7.1 or later. (See Note 3 below)
- A text file editor, such as NotePad, WordPad, or TextPad.
- Adobe Acrobat Reader, for the documentation.

Note 1: ModelScope can be installed on other versions of Windows, but needs special installation. Please contact **enquiries@metamaxim.com** for further information.

Note 2: The Java Software Development kit is available from Sun Microsystems at **http://java.sun.com/j2se/downloads.html**

Note 3: ModelScope uses XSL processing, which is built into Microsoft Internet Explorer version 6 or later and Netscape Navigator version 7.1 or later.
Microsoft Internet Explorer version 6 is available from Microsoft at
http://www.microsoft.com/windows/ie/default.asp
Netscape Navigator is version 7.1 is available from Netscape at
http://channels.netscape.com/ns/browsers/download.jsp

4 User Key

4.1 Obtaining a User Key

In order to run ModelScope, you need a User Key, supplied by Metamaxim. To obtain a User Key, please email Metamaxim at: **enquiries@metamaxim.com**, putting “User Key” as the subject of the email and your name in the body of the email.

You will get an email that contains two lines (starting USER and ID) that need to be copied into the ModelScope configuration file (see Section 10).

4.2 Evaluation Mode

ModelScope is available for evaluation free of charge. For evaluation purposes, ModelScope runs in restricted mode, which means that the size of model that can be run is limited to 5 Objects and 15 Events. Otherwise the software is fully functional.

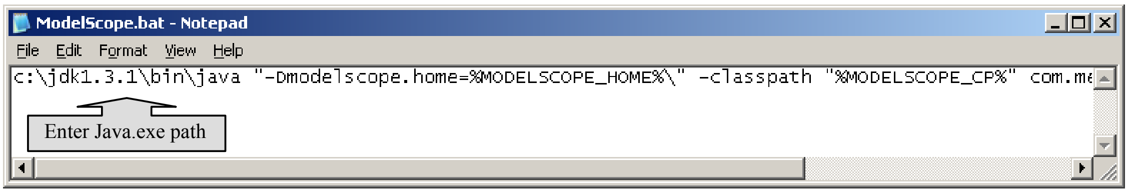
If this is not adequate for your evaluation, please contact **enquiries@metamaxim.com** .

5 Java Interpreter not Found

ModelScope is a Java program, and needs a Java interpreter to run. If you have the Java SDK installed, ModelScope should execute without any further action on your part.

In some cases, depending on the combination of Windows and Java versions installed on your machine, ModelScope may not be able to find Java virtual machine. In this case, an error alert “Java interpreter not found” will be displayed when you attempt to run ModelScope. If this happens, do the following:

1. Locate Java by using the Windows file search facility to look for “java.exe”. You may find there are several on your machine, but it is best to use the one located in c:\*java_dir*\bin, where *java_dir* is the directory where your Java SDK is installed (e.g., jdk1.3.1).
2. Find the file modelscope.bat, which is located in the directory where ModelScope is installed. Normally this will be c:\Program Files\ModelScope
3. Edit modelscope.bat by inserting the location of java.exe, as shown in Figure 1.



```
c:\jdk1.3.1\bin\java "-Dmodelscope.home=%MODELSCOPE_HOME%\" -classpath "%MODELSCOPE_CP%" com.me
```

Enter Java.exe path

Figure 1.

Note that if you use this mechanism and you upgrade to a new version of the Java SDK, you will need to re-edit modelscope.bat to point to the new version.

6 Cannot Start Internet Explorer

When ModelScope is started up, it should start both the server, a Java program, and the client, which runs in Microsoft Internet Explorer. Depending on which version of Windows is used, it is possible that ModelScope will not be able to start Internet Explorer automatically. In this case, a message “Unable to start Internet Explorer” will appear in the server messages window.

In this case, do the following:

1. With the server still running, navigate to the directory where ModelScope is installed. Normally this will be c:\Program Files\ModelScope.
2. Double-click on the file ModelScope.html.

This will open the client window.

To make this easy to do without navigating to the ModelScope directory every time, create a shortcut (by right-clicking on ModelScope.html) and place it on your desktop.

7 Creating and Running a Simple Model

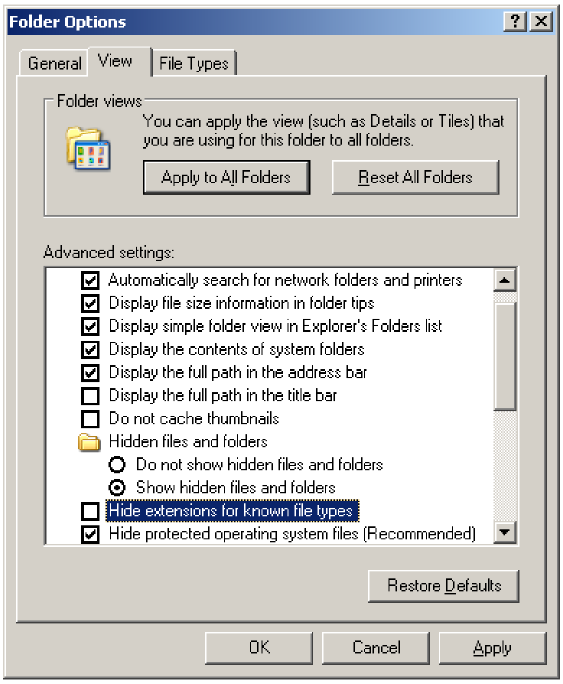
To get started with ModelScope, follow the instructions below to create and run a simple model. We will take the model through three versions.

7.1 First Version

First of all, you need to:

1. Create a folder anywhere on your system to hold your first model. Call this folder "ModelScope Test"
2. Using a text editor such as NotePad, create a text file within this folder called "Hollywood.bbm"

Note that text editors will create files with a default extension of ".txt". ModelScope uses the special extension ".bbm". If you uncheck the box "Hide extensions for known file types" (under the "Tools" menu of the "ModelScope Test" folder) you can see and edit extensions. See Figure 2.



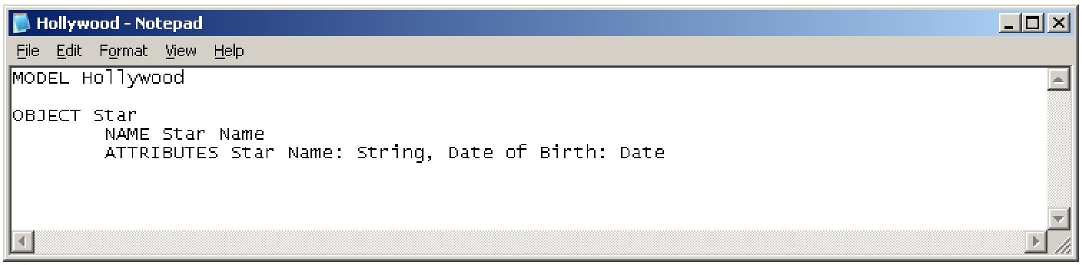
You can apply the view (such as Details or Tiles) that you are using for this folder to all folders.

Advanced settings:

☑ Automatically search for network folders and printers
☑ Display file size information in folder tips
☑ Display simple folder view in Explorer's Folders list
☑ Display the contents of system folders
☑ Display the full path in the address bar
☐ Display the full path in the title bar
☐ Do not cache thumbnails
Hidden files and folders
○ Do not show hidden files and folders
● Show hidden files and folders
☐ Hide extensions for known file types
☑ Hide protected operating system files (Recommended)

Figure 2.

In the file Hollywood.bbm, put the text shown in Figure 3.



```
MODEL Hollywood

OBJECT Star
        NAME Star Name
        ATTRIBUTES Star Name: String, Date of Birth: Date
```

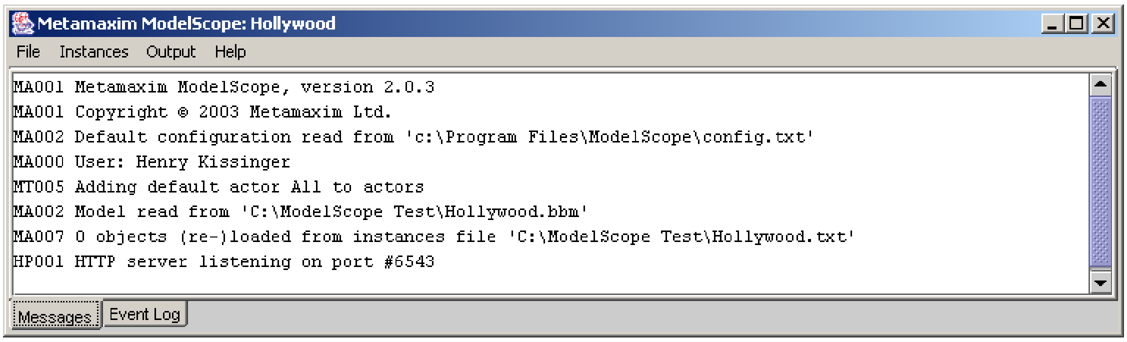
Figure 3.

This is your first model – the ModelScope equivalent of "Hello World". To check that this model executes:

1. Double-click on the ModelScope icon on your desktop
2. Select the file you have just created (Hollywood.bbm)
3. Click on “Open Model File”.

Note: It is also possible to start up ModelScope by double clicking on the file Hollywood.bbm.

This should cause the ModelScope Server Window to appear, as shown in Figure 4.

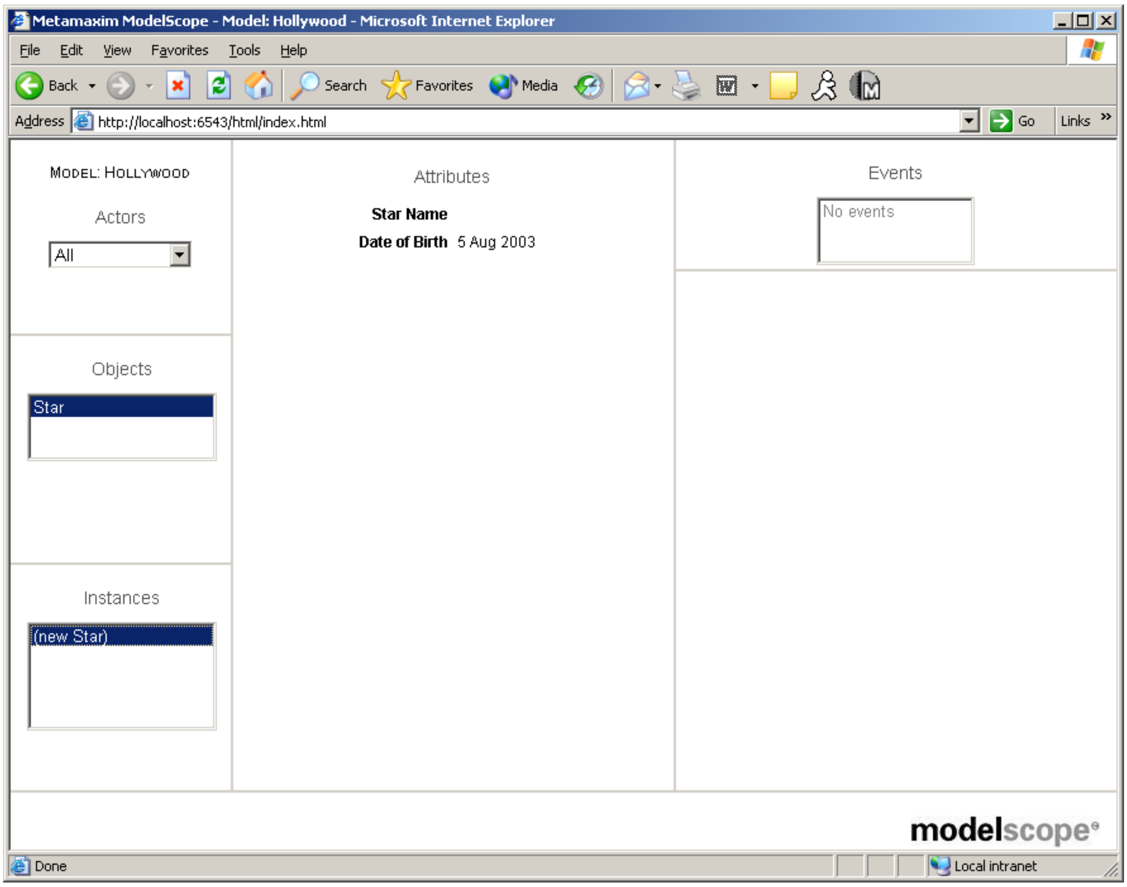


```
MA001 Metamaxim ModelScope, version 2.0.3
MA001 Copyright © 2003 Metamaxim Ltd.
MA002 Default configuration read from 'c:\Program Files\ModelScope\config.txt'
MA000 User: Henry Kissinger
MT005 Adding default actor All to actors
MA002 Model read from 'C:\ModelScope Test\Hollywood.bbm'
MA007 0 objects (re-)loaded from instances file 'C:\ModelScope Test\Hollywood.txt'
HP001 HTTP server listening on port #6543
```

Figure 4.

We will come back to discuss the messages that appear here later, in Section 7.

The client (Microsoft Internet Explorer) should also start up. If you select “All”, “Star” and “(new Star)” in the selection boxes on the right, it should appear as shown in Figure 5.



Star Name
Date of Birth 5 Aug 2003

Figure 5.

You have now created a model that has a single Object type: Star.

Table 1 shows how to use the ModelScope User Interface. The layout of the table (as three columns) corresponds to the layout of the user interface. To submit an event, follow the numbers.

Table 1.

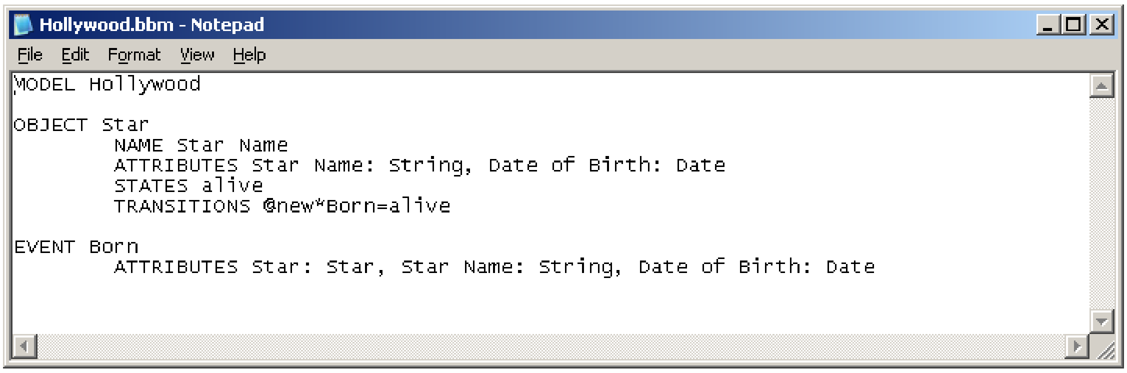
RIGHT HAND SIDE	CENTRE	LEFT HAND SIDE
Displays the Actors available in the model. **1. Select the Actor with which you want to work.**	Displays the attribute values of the selected Instance.	Displays the Events available for the selected Instance. **4. Select the Event you want to submit.**
Displays the Object Types available for the selected Actor. **2. Select the Object Type with which you want to work.**		Displays windows to enter the attributes of the slected Event. **5. Enter the desired values.**
Displays the Instances of the selected Object Type. **3. Select the Instance with which you want to work.**		Shows a button to submit the Event. **6. Press the button.**

(An "Actor", selected in step 1, corresponds to a user of the system. The example models discussed here do not use this feature of ModelScope, so only the default Actor "All" appears. *Please see the ModelScope Modellers' Guide for more details about Actors*).

In this version of the model, there are no Events to cause instances of the Star object to come into existence, so the message "No events" comes up when you perform step 3.

7.2 Second Version

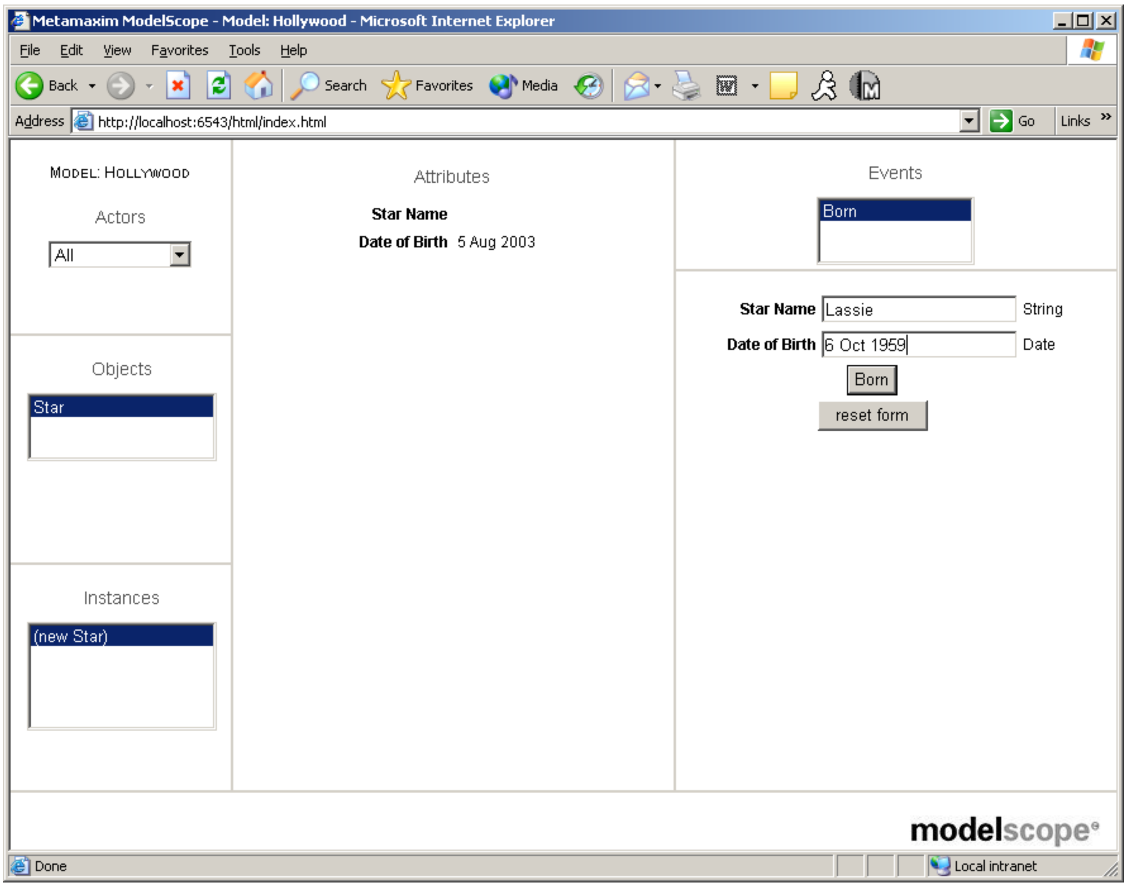
Close down ModelScope by closing the Browser and the Server Window, and amend the model as shown in Figure 6.



```
MODEL Hollywood

OBJECT Star
        NAME Star Name
        ATTRIBUTES Star Name: String, Date of Birth: Date
        STATES alive
        TRANSITIONS @new*Born=alive

EVENT Born
        ATTRIBUTES Star: Star, Star Name: String, Date of Birth: Date
```

Figure 6.

Save this file and restart ModelScope. You should now be able to create instances of Star by entering a Born event, as shown in Figure 7.


Metamaxim ModelScope - Model: Hollywood - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:6543/html/index.html
Go Links
Done
Local intranet
modelscope

MODEL: HOLLYWOOD

Actors: All

Objects: Star

Instances: (new Star)

Attributes

Star Name

Date of Birth 5 Aug 2003

Events: Born

Star Name: Lassie String

Date of Birth: 6 Oct 1959 Date

Born

reset form

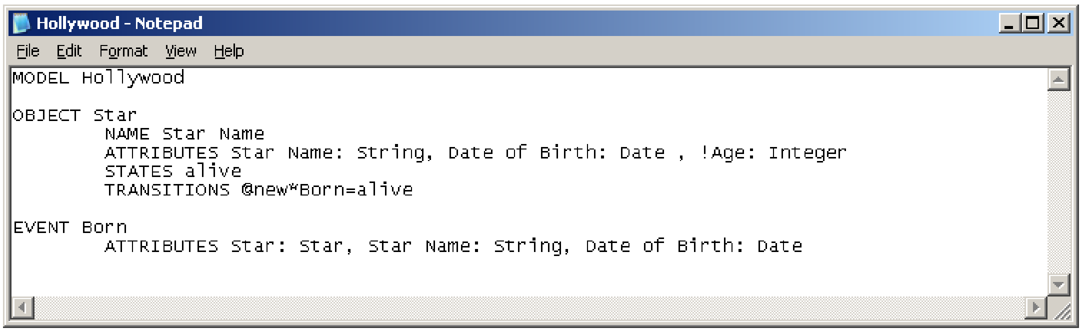
Figure 7.

7.3 Third Version

Finally, we will create a Java callback.

Callbacks are fragments of Java that supplement the basic model by performing calculations. In this example we shall create a callback to calculate the age of the Star. *(A full description of the different types of ModelScope callback can be found in the ModelScope Modellers' Guide.)*

First, add an Age attribute to the definition of Star, as shown in Figure 8.



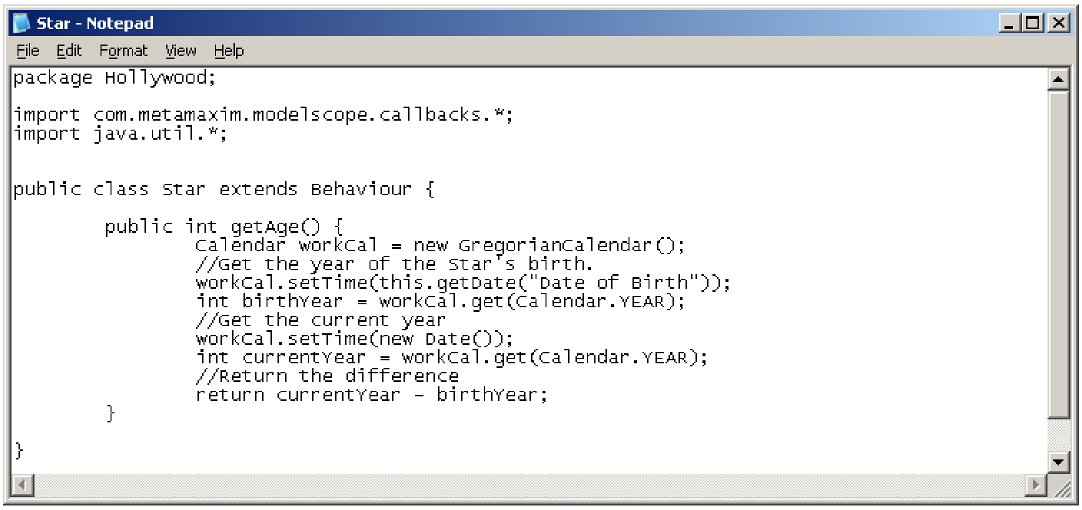
```
MODEL Hollywood

OBJECT Star
        NAME Star Name
        ATTRIBUTES Star Name: String, Date of Birth: Date , !Age: Integer
        STATES alive
        TRANSITIONS @new*Born=alive

EVENT Born
        ATTRIBUTES Star: Star, Star Name: String, Date of Birth: Date
```

Figure 8.

The "!" in front of the Age attribute indicates that it is calculated, rather than entered as part of an Event. This tells ModelScope that it has to invoke the callback to calculate the value of the attribute.

Next, create the callback by doing the following:

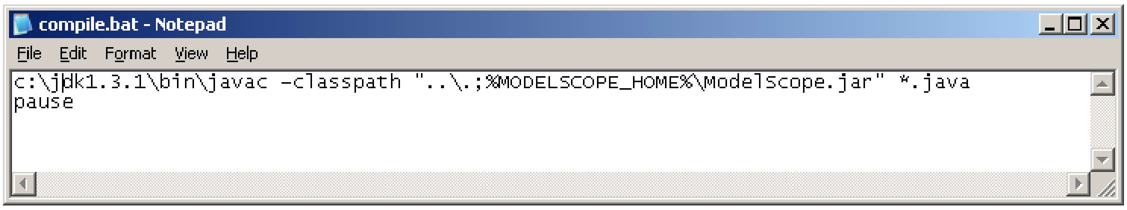
1. Create a new folder in “ModelScope Test” called “Hollywood”. This is the folder for callback files.
2. Create a file in this new folder called “Star.java”.
3. Put the text shown in Figure 9 in Star.java. This is the Java code required to calculate the Age in years from the Date of Birth.



```
package Hollywood;

import com.metamaxim.modelscope.callbacks.*;
import java.util.*;


public class Star extends Behaviour {

        public int getAge() {
                Calendar workCal = new GregorianCalendar();
                //Get the year of the Star's birth.
                workCal.setTime(this.getDate("Date of Birth"));
                int birthYear = workCal.get(Calendar.YEAR);
                //Get the current year
                workCal.setTime(new Date());
                int currentYear = workCal.get(Calendar.YEAR);
                //Return the difference
                return currentYear - birthYear;
        }

}
```

Figure 9.

4. Create a batch file, Compile.bat, to compile the callback, as shown in Figure 10. Replace `c:\jdk1.3.1` with your Java directory. (If you do know where this is, use the Windows file search facility to find javac.exe).
   The “pause” ensures that the window remains open so that you can see any compilation errors.



```
c:\jdk1.3.1\bin\javac -classpath "..\.;%MODELSCOPE_HOME%\ModelScope.jar" *.java
pause
```

Figure 10.

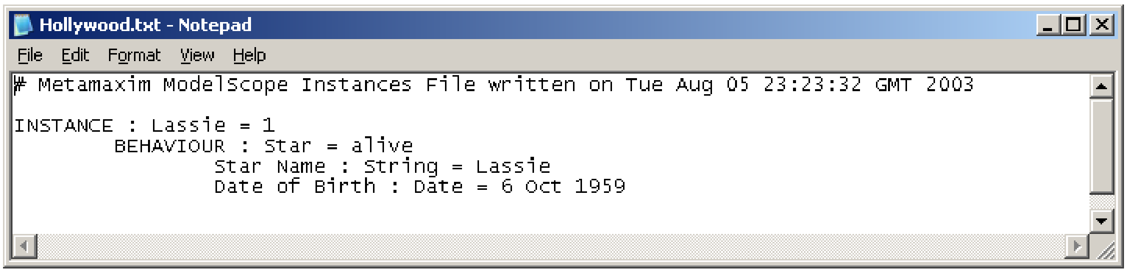
5. Run this batch file (double-click on it) to compile the callback.
6. Run ModelScope. The age should appear for each Star.

To find out more about how to build models, please refer to the ModelScope Modellers’ Guide.

8 The Instances File

When ModelScope runs, it creates a file to hold instances of the objects created by the session. This file is called “Hollywood.txt”, in other words the name of the model with extension “.txt”.

After creating a Star and closing ModelScope down., the instances file will be as shown in Figure 11



```
# Metamaxim ModelScope Instances File written on Tue Aug 05 23:23:32 GMT 2003

INSTANCE : Lassie = 1
        BEHAVIOUR : Star = alive
                Star Name : String = Lassie
                Date of Birth : Date = 6 Oct 1959
```

Figure 11.

This file makes instances of objects persistent. When you restart ModelScope, the instances created in the last session are read in from this file.

9 The Server Window

The Server Window (see Figure 4) is where ModelScope writes messages – both error messages when something has gone wrong, and information messages to help track what is going on during execution of a model.

The window has two tabs at the bottom, marked "Messages" and "Event Log". The first of these displays a pane showing error and information messages produced by ModelScope.

The second shows a pane for log messages that can be generated from a model using the event.log() command in a callback (*see ModelScope Modellers' Guide for more information)*

9.1 Error Messages

ModelScope error message are displayed in the Messages window. Error messages are designed to be self-explanatory.

9.1.1 Errors in the Model File

When an error is detected in the input model file, ModelScope gives a line number for the line where the error was detected. This is always the line on which the offending entry started. If the entry spans over several lines, it may not be the actual line where the error is located.

9.1.2 Errors in Callbacks

When an error is detected in a Java callback, ModelScope gives the name of the class and the name of the method that caused the error.

9.2 Menus

The menu options available from the Server Window are described in Table 2.

Table 2.

MENU	MENU OPTIONS	FUNCTION
File	Exit	Closes down the server (same as closing the Server Window).
Model (See Note below)	Reload Model	Reloads the model and associated callbacks into ModelScope. The effect is just the same as closing ModelScope, changing the model and/or callbacks and restarting ModelScope; but it is faster and more convenient.
Instances (See Note below)	Clear Instances	Deletes all the instances. This also clears out the Instance File.
	Reload Instances	Reloads the instances from the Instances File. Used after editing the Instances File while ModelScope is running.
	Load Instances …	Loads the instances from selected file. The new instances are added, so the old ones are still there. (If the old instances are not needed, use Clear Instances first.) ModelScope continues to write to the old Instances File.
Output	Copy Messages	Copies the contents of the Messages pane to the clipboard.
	Clear Messages	Clears the Messages pane.
	Copy Event Log	Copies the contents of the Event Log pane to the clipboard.
	Clear Event Log	Clears the Event Log pane.
Help	Getting Started	Opens the Getting Started document.
	Modellers Guide	Opens the Modellers Guide document.
	About Metamaxim ModelScope …	Displays version etc. information about ModelScope.

Note: After using some menu options, it is necessary to refresh the browser. ModelScope will generate an alert to advise that this is necessary.

In order to refresh the browser, use the refresh button. This appears as 🔄 in Microsoft Internet Explorer, and as Reload in Netscape.

10 Configuration File

A configuration file (config.txt) provides a means of configuring some aspects of the ModelScope installation.

This file is in the ModelScope folder containing the software . This will normally be located in the Program Files folder, but may be located elsewhere if requested at installation time.

Table 3 lists the parameters that may be specified in config.txt and the default value assumed by ModelScope if the parameter is omitted.

All the parameters are of the form KEY_WORD=VALUE. There must be no trailing spaces after the VALUE. KEY_WORDs must be in upper case.

Example:

MODEL_EXTENSION=mod

Lines beginning # in the parameter file are comments.

Table 3.

KEY WORD	DESCRIPTION	DEFAULT VALUE
USER KEY Parameters		
`USER`	The name to which the software is licensed.	`n/a`
`ID`	The encrypted license key provided by Metamaxim for this user.	`n/a`
FILE NAMING Parameters		
`MODEL_EXTENSION`	The file name extension for Model files.	`bbm`
`INSTANCES_EXTENSION`	The file name extension for Instances files. This must be different from the MODEL_EXTENSION value.	`txt`
METADATA Parameters		
`SPACE_REPLACEMENT_CHAR`	Identifies the character that is used to substitute for the space character when creating Java names (for callbacks) from names in the metadata. The only sensible choice (aside from the default) is the underscore character.	All space characters are removed from a name when translating to a Java name.
FORMAT Parameters		
`DATE_FORMAT`	Specifies the text format for displaying and storing date attribute values. See the documentation for the Java class `java.text.SimpleDateFormat` for details on the possible values for this parameter.	DD MMM YYYY (MMM is alpha, e.g., Jan)
HTTP Parameters		
`PORT_NUMBER`	The TCP/IP port number at which the server listens for HTTP requests.	`6543`

Continued on next page.

`HTTP_ROOT`	Names the directory root of the "web server" component of the ModelScope server. The paths to XSL stylesheets are relative to this directory, as are the paths to any CSS stylesheets and JavaScript source files that are referenced in the generated HTML.	The directory from which the ModelScope server is run.

11 Known Bugs

001 Metadata names (e.g., Attribute names. Behaviour names) which are the same as keywords (e.g., OBJECT, NAME, etc) and start a new line in the metadata file are processed incorrectly.

Work-around Avoid using keywords as metadata names, or ensure that they do not start a line.