

Rational Robot

A Test Automation Tool

What is Rational Robot?

- Rational Robot is a complete set of components for automating the testing of Microsoft Windows client/server and Internet applications. (Rational Robot user guide)
- Rational Robot is an ***automated functional regression*** testing tool.

Automated Functional Regression Testing

- **Functional Test:** Functional Tests are designed to make sure that the application performs as it was intended
- **Regression Test:** A regression test is a test where an application is subjected to a suite of functional tests at each build to ensure that everything that worked continues to work.

Components Of Rational Robot

- **Rational Administrator** - Create and manage Rational projects to store your testing information.
- **Rational TestManager** - Review and analyze test results.
- **Object Properties, Text, Grid, and Image Comparators** - View and analyze the results of verification point playback.
- **Rational SiteCheck** - Manage Internet and intranet Web sites.

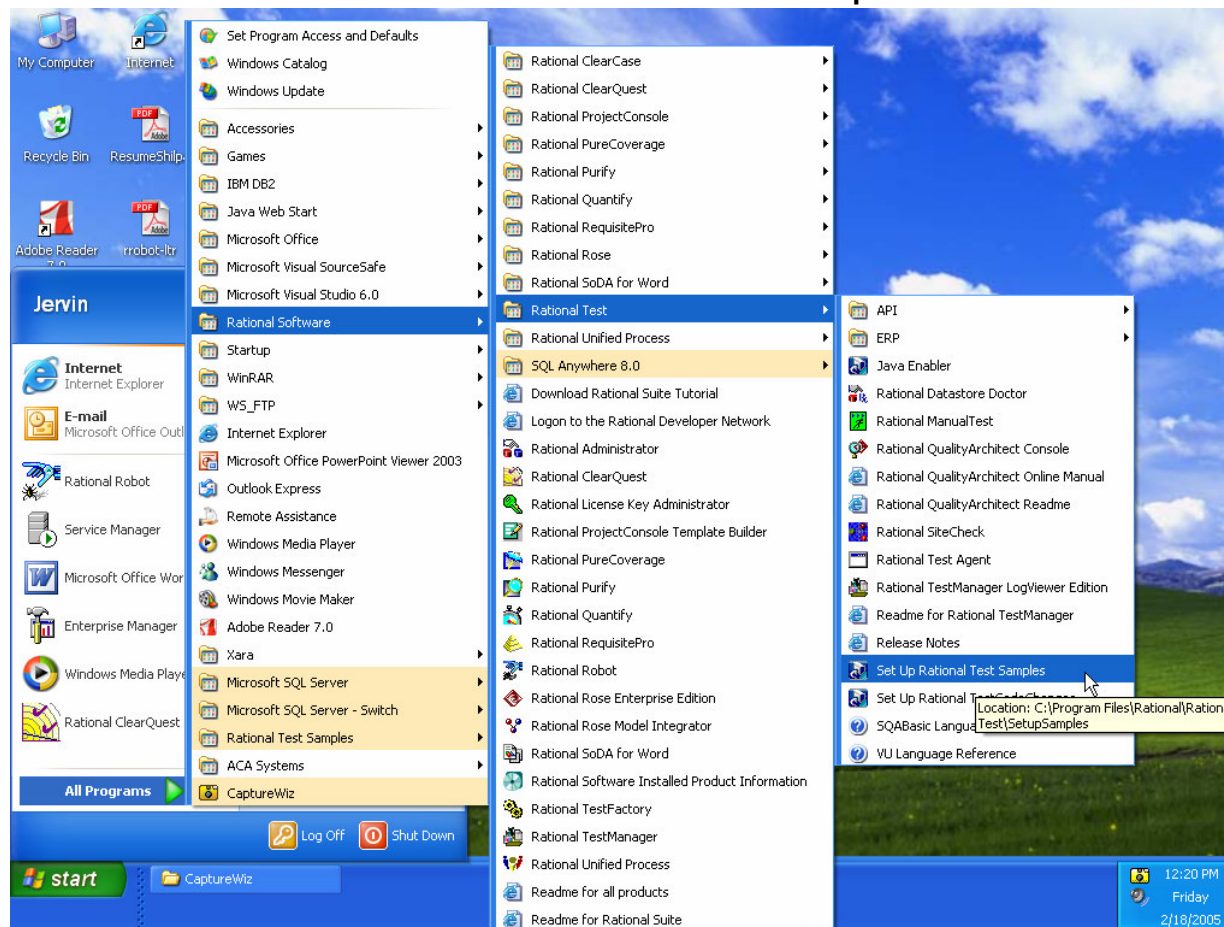
Rational Robot Tutorial

Setup For Tutorial

- Installation of Sample Applications
- Creating a Rational Administrator Project

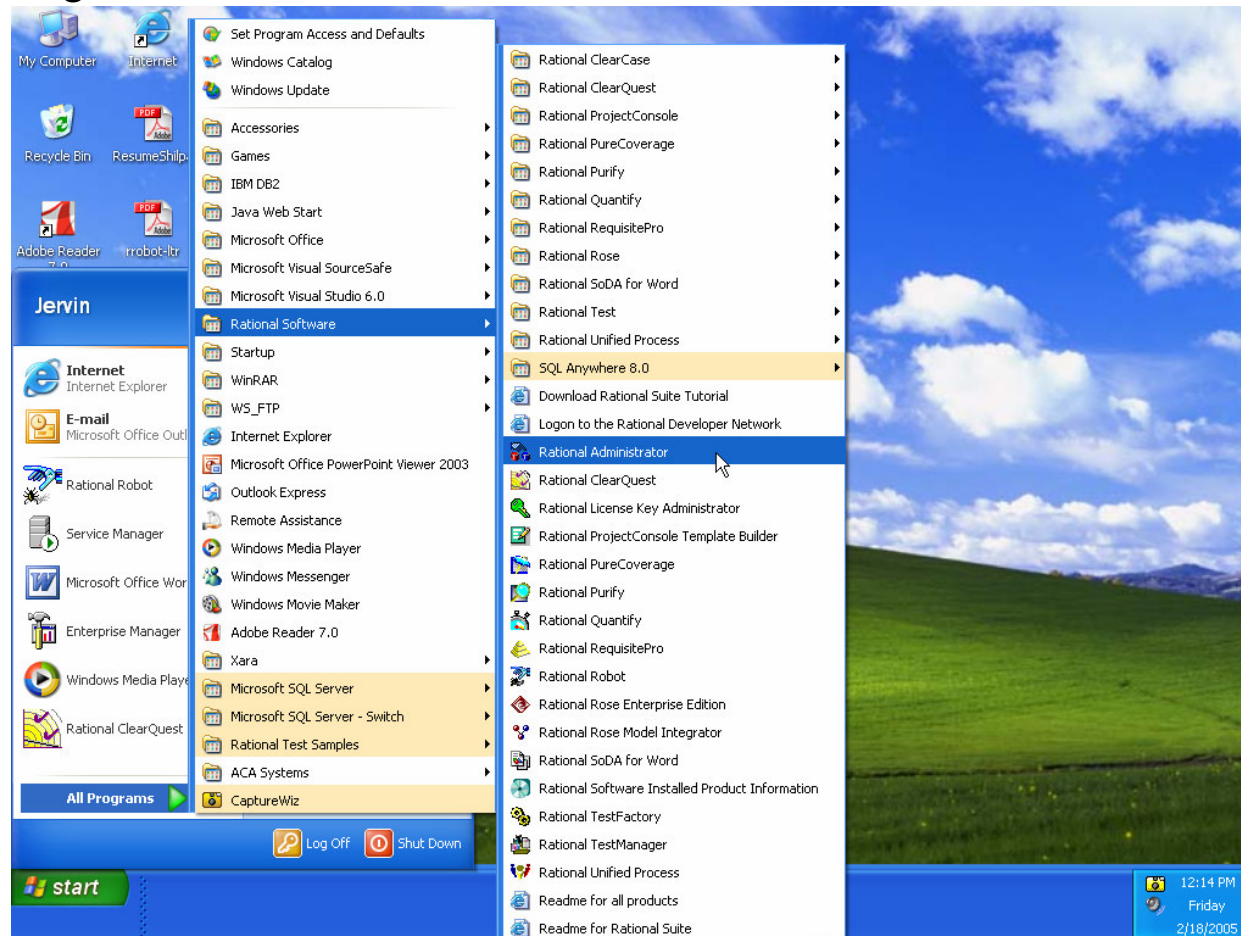
Installation of Sample Applications

- Install the Sample application by clicking:
Start > Rational Software > Rational Test > Set up Rational Test Samples



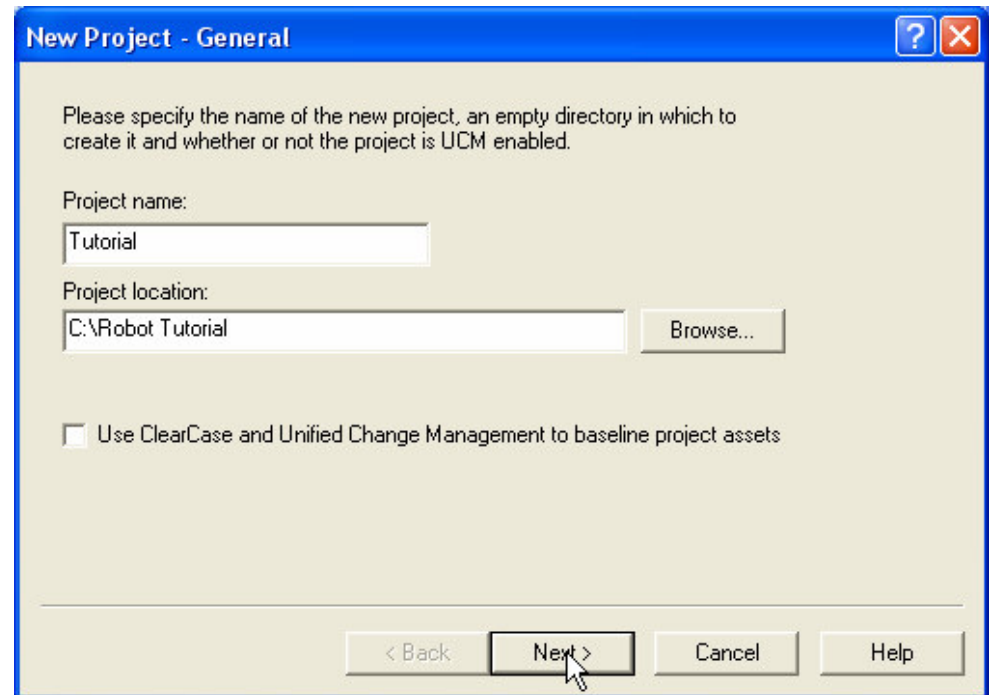
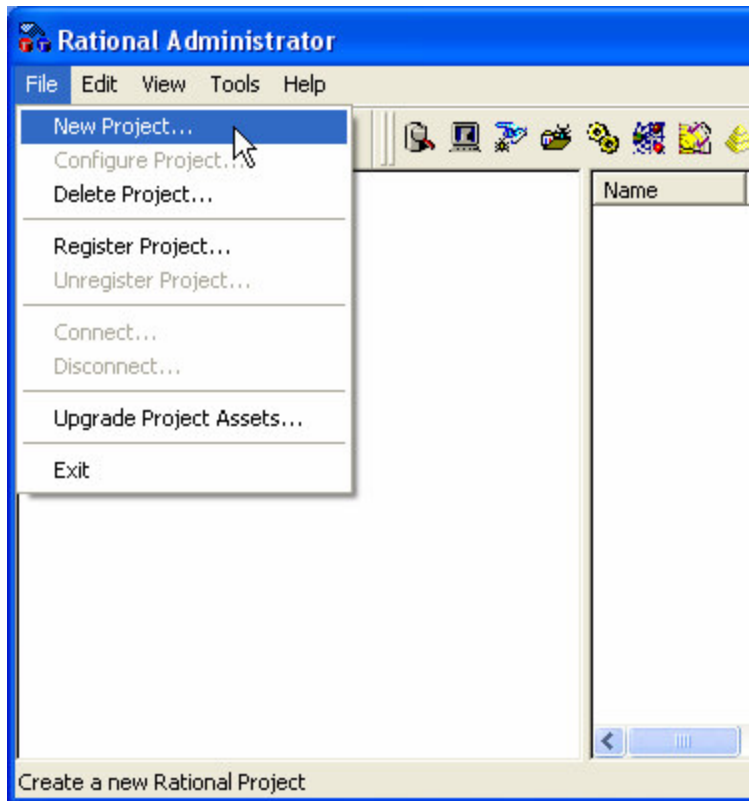
Create a Rational Administrator Project

- Start Rational Administrator by clicking:
Start > Programs > Rational Software > Rational Administrator



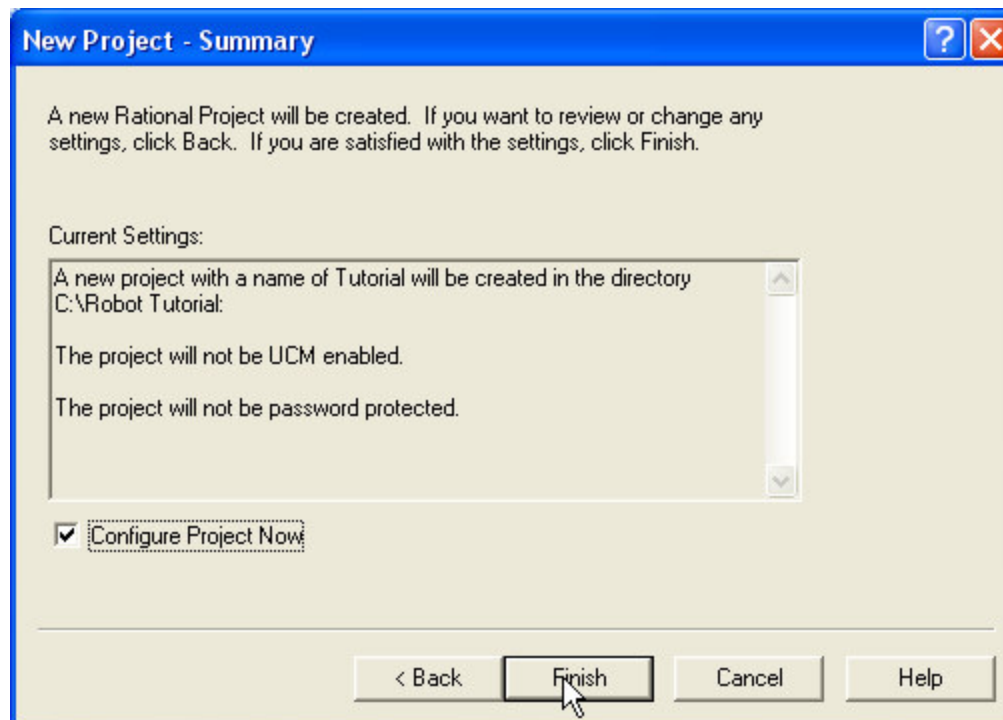
Create a Rational Administrator Project

- Create a new Project: Choose File > New Project
- Enter a project name (Tutorial) and path (C:\Robot Tutorial)



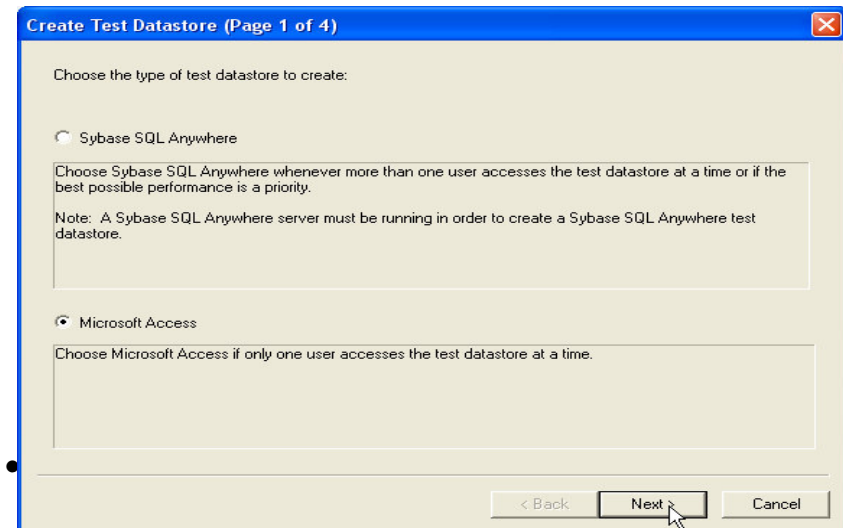
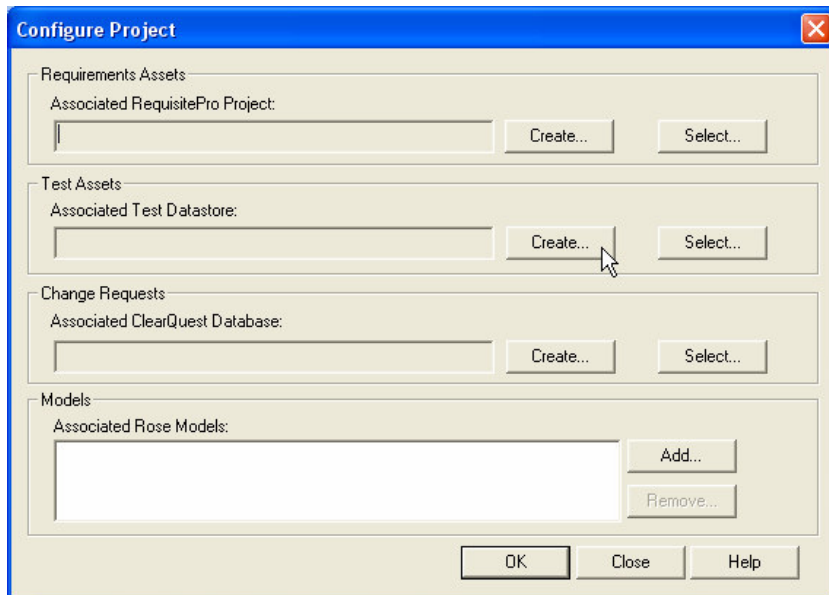
Create a Rational Administrator Project

- Click okay on the warning window and do not set a password, just click next.
- Ensure that **Configure Project Now** is checked on the summary page.



Create a Rational Administrator Project

- To configure the test datastore that will be part of your project, click **Create...** in the Test Assets group.
- For this Tutorial, MS Access will be sufficient, however for projects involving more than one person, Rational recommends using Sybase SQL Anywhere.



Overview of Sample Application

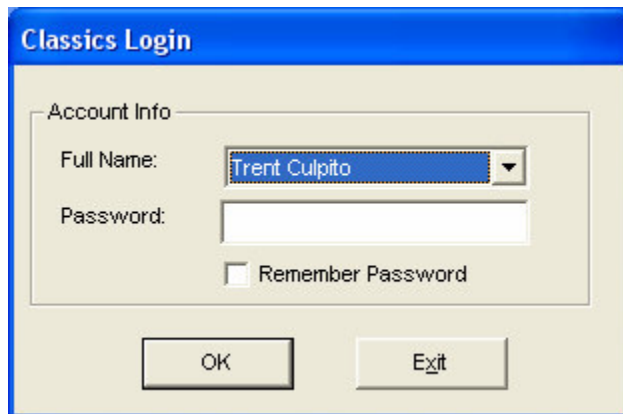
A Quick tour of Classics Online

A Quick tour of Classics Online

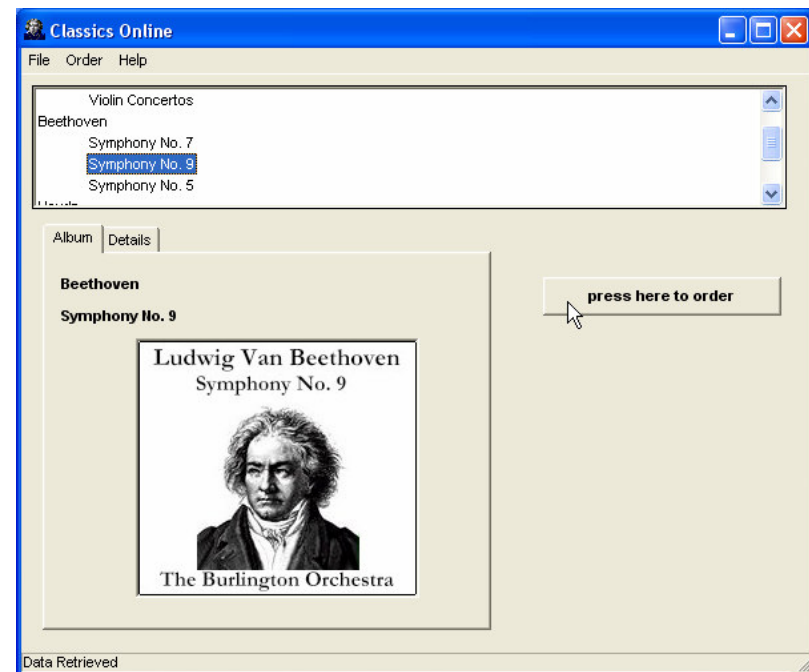
- Classics Online is a sample application written in Visual Basic that we will be using to run our automated testing on.
- There are 3 separate builds, each of which we will be performing regression tests on.

A Quick tour of Classics Online

- Classics Online is a simulation of an online store where you can buy classic CDs.
- Start Classics Online by clicking: Start > Rational Test Samples > ClassicsA (which is the first build)
- The password doesn't matter, just click OK.
- This brings up the main interface where you can select a CD to buy. Lets try it out.
- Double click on "Beethoven > Symphony No. 9" and click the "Press here to Order" Button.



The image shows a "Classics Login" dialog box with a blue title bar. It contains an "Account Info" section with a "Full Name:" label and a dropdown menu showing "Trent Culpito". Below that is a "Password:" label and an empty text box. A checkbox labeled "Remember Password" is also present. At the bottom are "OK" and "Exit" buttons.



A Quick tour of Classics Online

- The next screen prompts you for a credit card number and expiration date. Type any value and hit “Place Order”
- You can view a summary of your orders from the main window by clicking Order > View Existing Order Status...

Make An Order

Item: **Bach - Brandenburg Concertos Nos. 1_3** Sub-Total: \$ 16.99

S+H: \$ 2.00

Quantity: Total: \$ 18.99

Payment Information

Card Number (include the spaces):

Card Type: Expiration Date:

Your Information

Name:

Street:

City, State Zip:

Telephone:

View Existing Orders

Orders for Trent Culpito

ORDERID	STATUS	COMPOSER	COMPOSITION	QUANTIT
21	Order Initiated	Beethoven	Symphony No. 9	

- For setting up the tutorial, clear all order history before continuing and exit Classics Online.

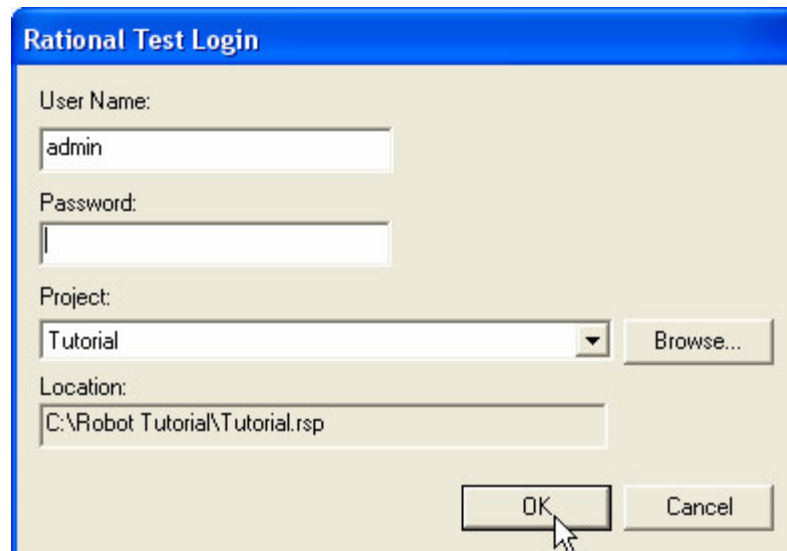
Rational Robot

Recording a Test Script

Recording a Test Script

Starting Rational Robot

- Start Rational Robot by choosing:
Start > Programs > Rational Software > Rational Robot
- You must log into a Rational Administrator Project to continue. Select the “Tutorial” Project we created earlier.
Projects are created with Admin user with a blank password. For this tutorial this will suffice, however on actual projects you will want to create a username and password for each member of the team.
Just click “OK” to use the Admin username



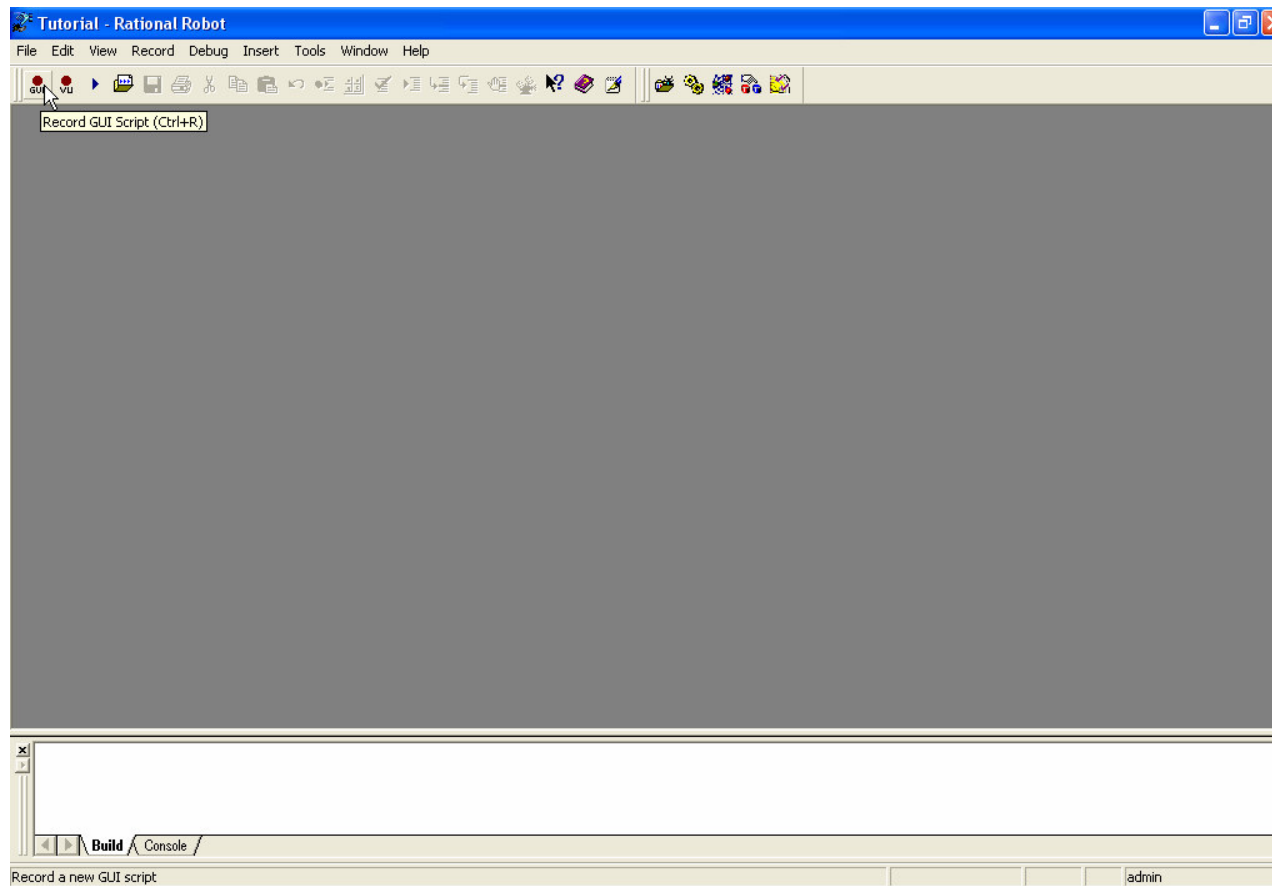
The image shows a screenshot of the "Rational Test Login" dialog box. The dialog has a blue title bar and a light beige background. It contains the following fields and controls:

- User Name:** A text box containing the text "admin".
- Password:** A text box that is currently empty.
- Project:** A dropdown menu showing "Tutorial" with a downward arrow. To the right of the dropdown is a "Browse..." button.
- Location:** A text box containing the path "C:\Robot Tutorial\Tutorial.rsp".
- Buttons:** At the bottom right, there are two buttons: "OK" and "Cancel". A mouse cursor is pointing at the "OK" button.

Recording a Test Script

Starting Rational Robot

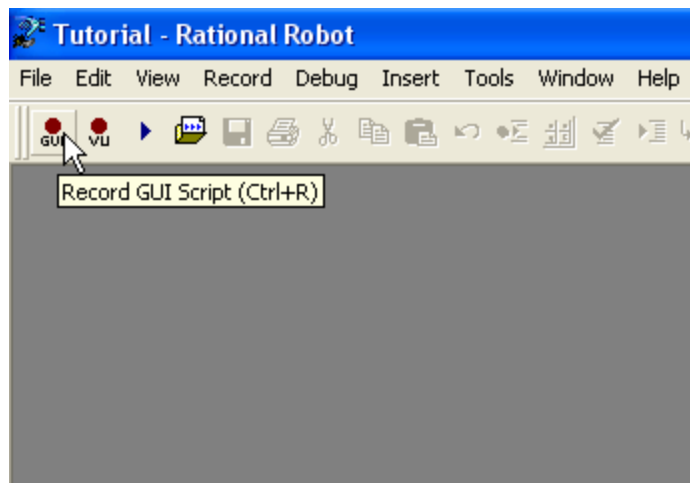
- This is the main screen of Rational Robot. Not much to see here until you have recorded a script.



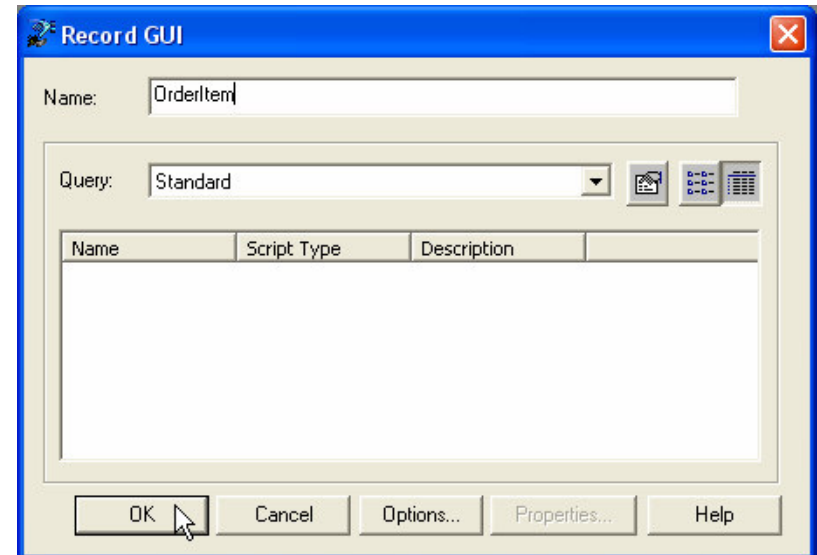
Recording a Test Script

Starting Rational Robot

- To start recording a GUI test script, click on the red button on the top-left called "Record GUI script"



- On the screen that pops up, enter a name of the script (OrderItem) and click OK.



- Robot minimizes and a GUI Record Toolbar appears. This toolbar lets you pause/stop recording the test script.



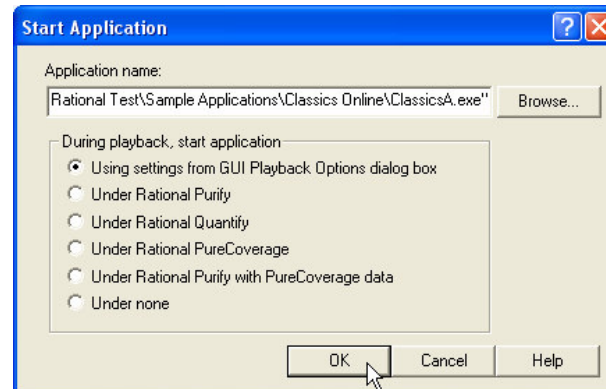
Recording a Test Script

Application Under Test

- When in Record Mode, the toolbar will be on top and will record all keystrokes and mouse clicks. If you want to do something that you don't want to record, remember to pause first.
- The first step in recording the script is to start the **Application Under Test (AUT)**. Click the last button on the toolbar – Display GUI Insert Toolbar.



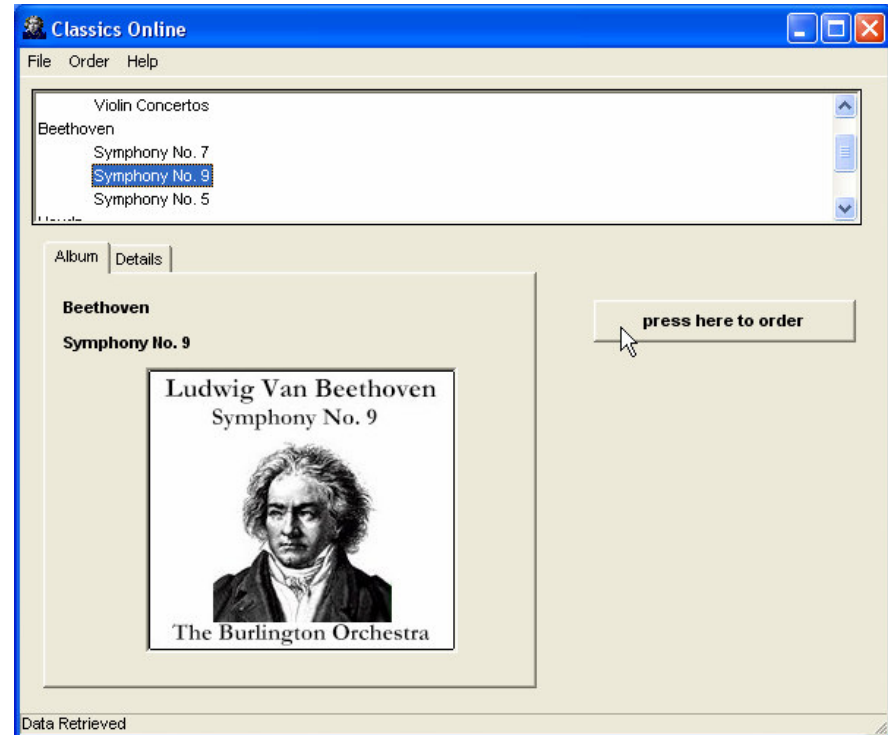
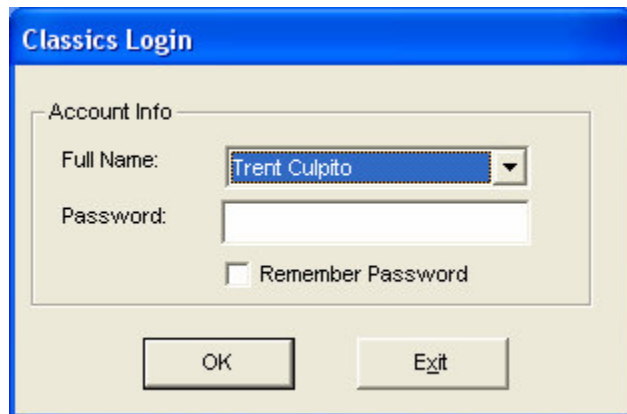
- This toolbar has many options you will use while recording, but for now select the “Start Application” button.
- Under Application Name enter “C:\Program Files\Rational\Rational Test\Sample Applications\Classics Online\ClassicsA.exe” or browse for it and hit OK.



Recording a Test Script

Placing an Order

- The login window for Classic Online Appears. Click OK as before.
- Once again, select “Beethoven Symphony No.9” and click press here to order.




Recording a Test Script

Placing an Order – Verification Points


- So far we've only had Robot record the navigation through the application, but have not actually tested anything.
- To test something, we set “Verification points” (VPs) on items we want to check against. The current state of the application is referred to as the *baseline*. This baseline is compared to the state during playback and if they do not match, the VP fails.
- On the GUI Record Toolbar, click the “Display GUI Insert” again. There are many buttons to set VPs. Here are some of the available VP types and their buttons:

 Alphanumeric Verifies alphanumeric data. Used for edit boxes, pushbuttons, l

 Clipboard Verifies the contents of the Windows clipboard.

 Menu Verifies the menu values and optionally their state (enabled or

 Object Data Tests data content of objects.

 Object Properties Tests object attributes such as color, font, position.

 Region Image Graphically compares an area of the screen you specify.

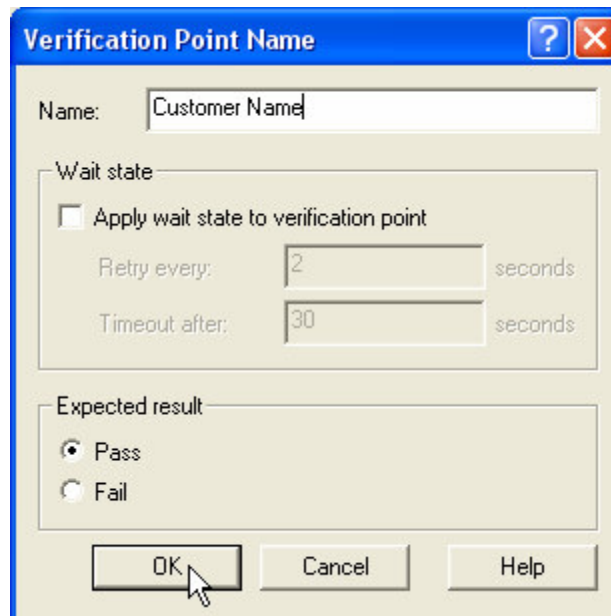
 Window Existence Tests to see if a particular window does or does not exist on th

 Window Image Graphically compares an entire window such as a window box.

Recording a Test Script

Placing an Order – Verification Points

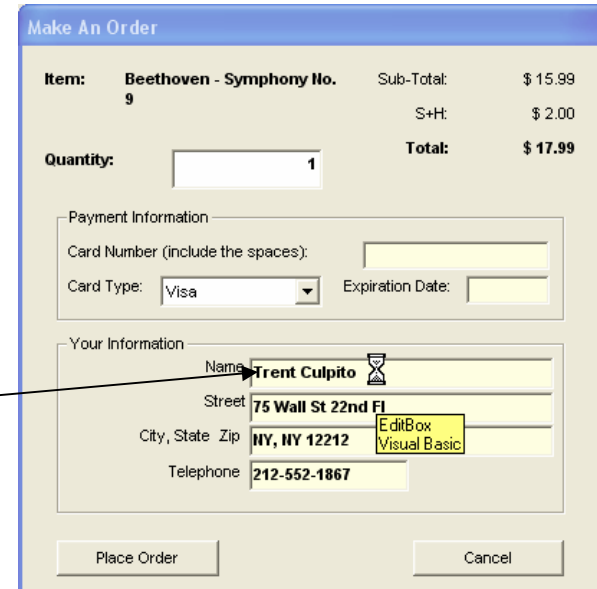
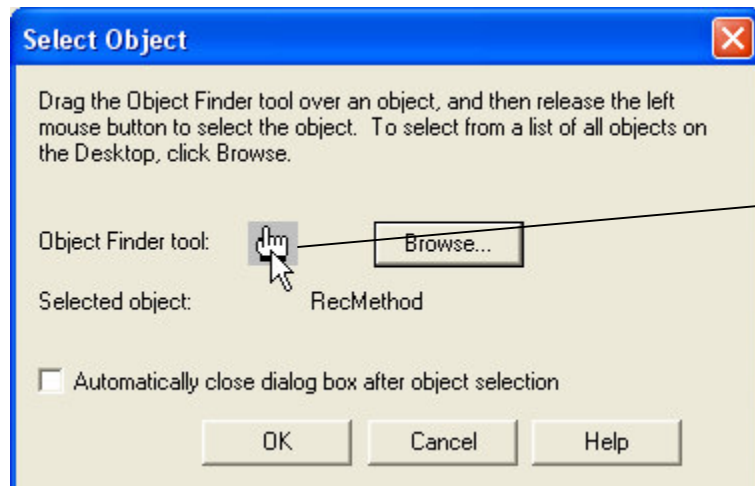
- Since we logged in as “Trent Culpito”, let’s verify that the name appears on the form. This will be our first VP.
- Click the Object Properties VP button on the toolbar. The following window lets you set up some options for the VP. We only need to set a name (“Customer Name”) and leave the rest at the default values.



Recording a Test Script

Placing an Order – Verification Points

- Now you must designate the object to be tested. You can do this either by dragging the “Object Finder Tool” over the object, or by browsing for it.
- In most cases, it is easier to drag and drop the tool, however for hidden items, you will have to browse for them.
- Drag and drop the tool over the Customer Name text field. Notice that the tool-top provides info about the object as Robot sees it.

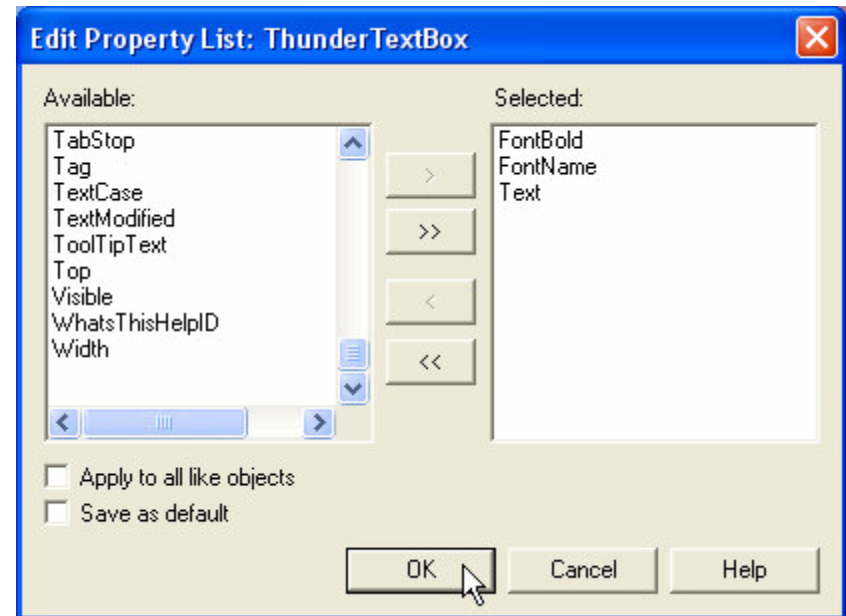
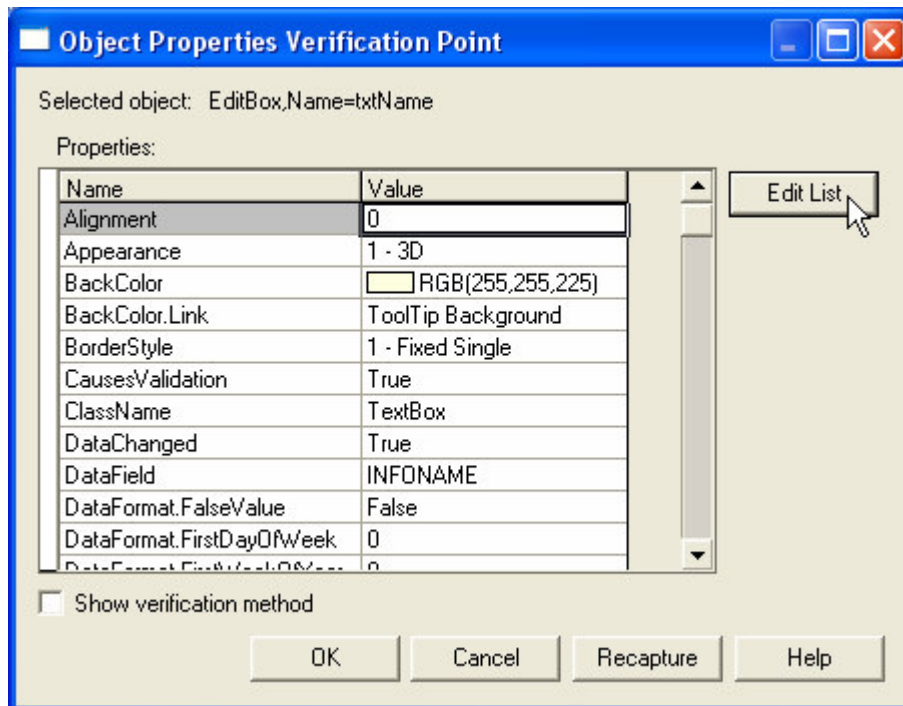


- Click okay when the Select Object window returns.

Recording a Test Script

Placing an Order – Verification Points

- The window that appears shows you what properties *can* be verified with this VP. We do not want to verify all of them, because most of them do not affect the functionality of the program.
- To select which properties we do want to test, click the “Edit List...” button.
- For example, let us just test **FontBold**, **FontName** and **Text** properties. Click OK.



Recording a Test Script

Placing an Order

- This brings us back to Classics Online. Enter any text into the credit card and expiration date text fields and hit Place Order

Make An Order

Item:	Beethoven - Symphony No. 9	Sub-Total:	\$ 15.99
		S+H:	\$ 2.00
Quantity:	<input type="text" value="1"/>	Total:	\$ 17.99

Payment Information

Card Number (include the spaces):

Card Type: Expiration Date:

Your Information

Name

Street

City, State, Zip

Telephone

Recording a Test Script

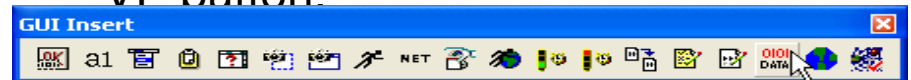
Verifying the Order

- One of the main things to check is whether or not the order was placed. To do this, select Order > View Existing Order Status...

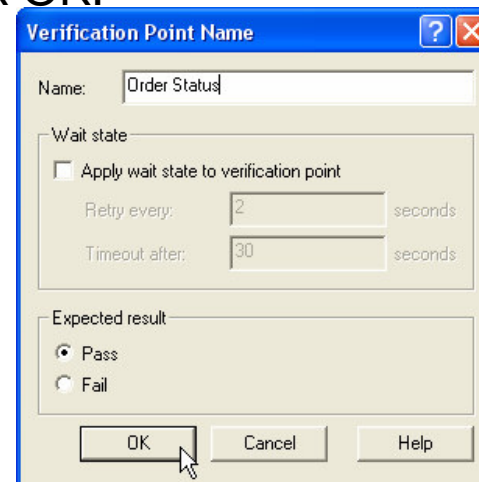


- This shows you a summary of past orders.

- Let us create a verification point for this.
- Click the "Display GUI Insert" Toolbar button on the GUI Record toolbar and click the Object Data VP button.



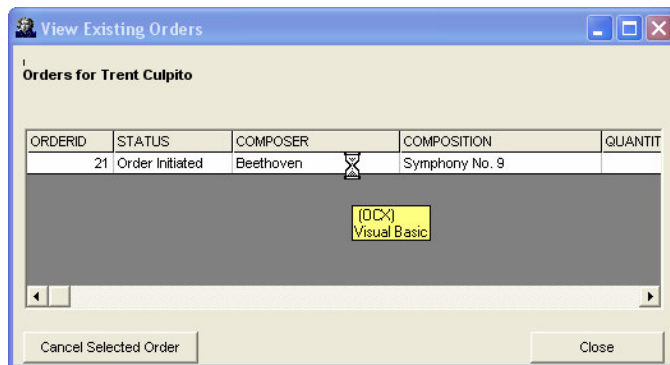
- Name this VP "Order Status" and click OK.



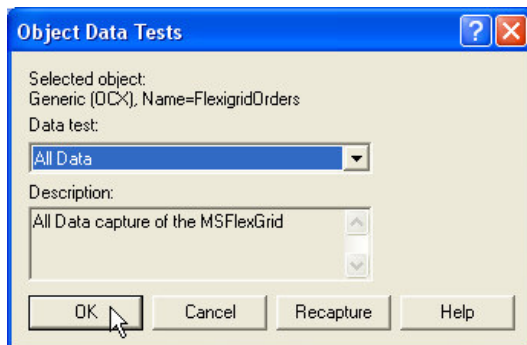
Recording a Test Script

Verifying the Order

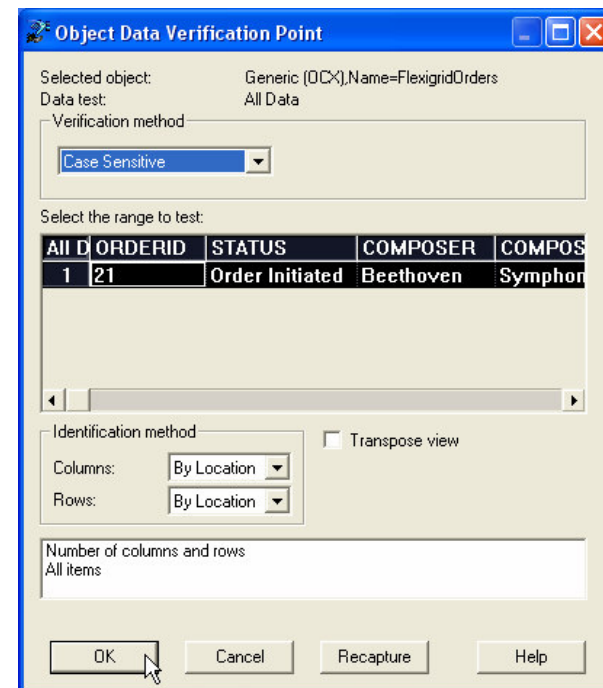
- As before, drag the Object Finder Tool over the data table on the “View Existing Orders” window and hit OK.



- For this test, capture “All Data” in the grid and click OK.



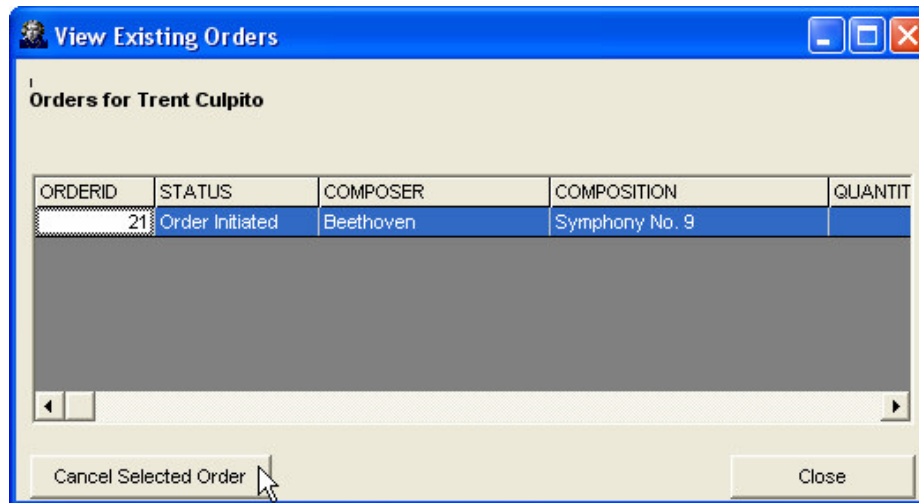
- The Object Data Verification Point window pops up where you can configure how you want to test the data.
- For this test, leave the default values and click OK.



Recording a Test Script

Restoring the state

- Finally, back at the Order Status window of Classics Online, select the row with the order you placed and hit the “Cancel Selected Order” button. This is to restore the application to its pre-test state.



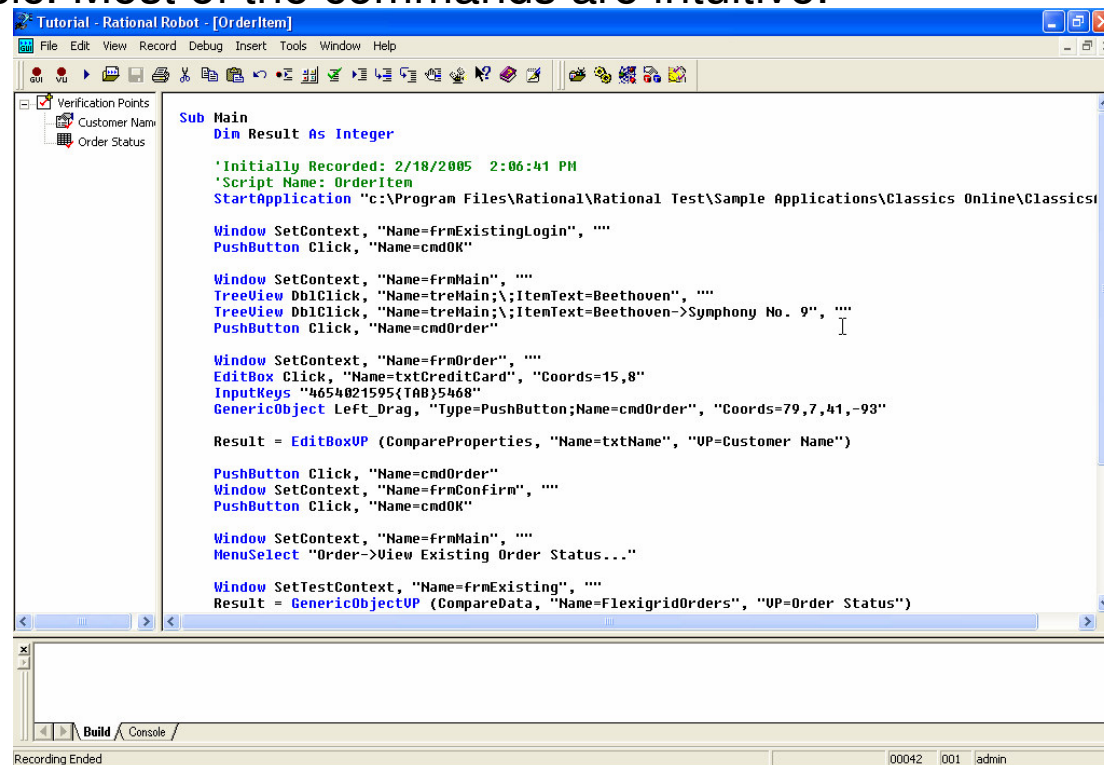
- Exit Classics online and stop recording by clicking the “Stop” icon on the GUI Record Toolbar.



Recording a Test Script

The Test Script

- Once you stop recording, Rational Robot restores itself. On the screen you can edit the script that was automatically generated, view and edit the VPs, etc.
- The script is written in a language called SQABasic which is an extension of Visual Basic. Most of the commands are intuitive.



The screenshot shows the Rational Robot interface with a test script for 'OrderItem'. The script is written in SQABasic and includes various actions like 'StartApplication', 'Window SetContext', 'PushButton Click', 'TreeView DblClick', 'EditBox Click', 'InputKeys', 'GenericObject Left_Drag', and 'Result = EditBoxUP'. The interface also shows a 'Verification Points' tree on the left and a 'Console' window at the bottom.

```
Sub Main
  Dim Result As Integer

  'Initially Recorded: 2/18/2005 2:06:41 PM
  'Script Name: OrderItem
  StartApplication "c:\Program Files\Rational\ Rational Test\Sample Applications\Classics Online\Classics

  Window SetContext, "Name=frmExistingLogin", ""
  PushButton Click, "Name=cmdOK"

  Window SetContext, "Name=frmMain", ""
  TreeView DblClick, "Name=treMain;ItemText=Beethoven", ""
  TreeView DblClick, "Name=treMain;ItemText=Beethoven->Symphony No. 9", ""
  PushButton Click, "Name=cmdOrder"

  Window SetContext, "Name=frmOrder", ""
  EditBox Click, "Name=txtCreditCard", "Coords=15,8"
  InputKeys "4654021595{TAB}5468"
  GenericObject Left_Drag, "Type=PushButton;Name=cmdOrder", "Coords=79,7,41,-93"

  Result = EditBoxUP (CompareProperties, "Name=txtName", "UP=Customer Name")

  PushButton Click, "Name=cmdOrder"
  Window SetContext, "Name=frmConfirm", ""
  PushButton Click, "Name=cmdOK"

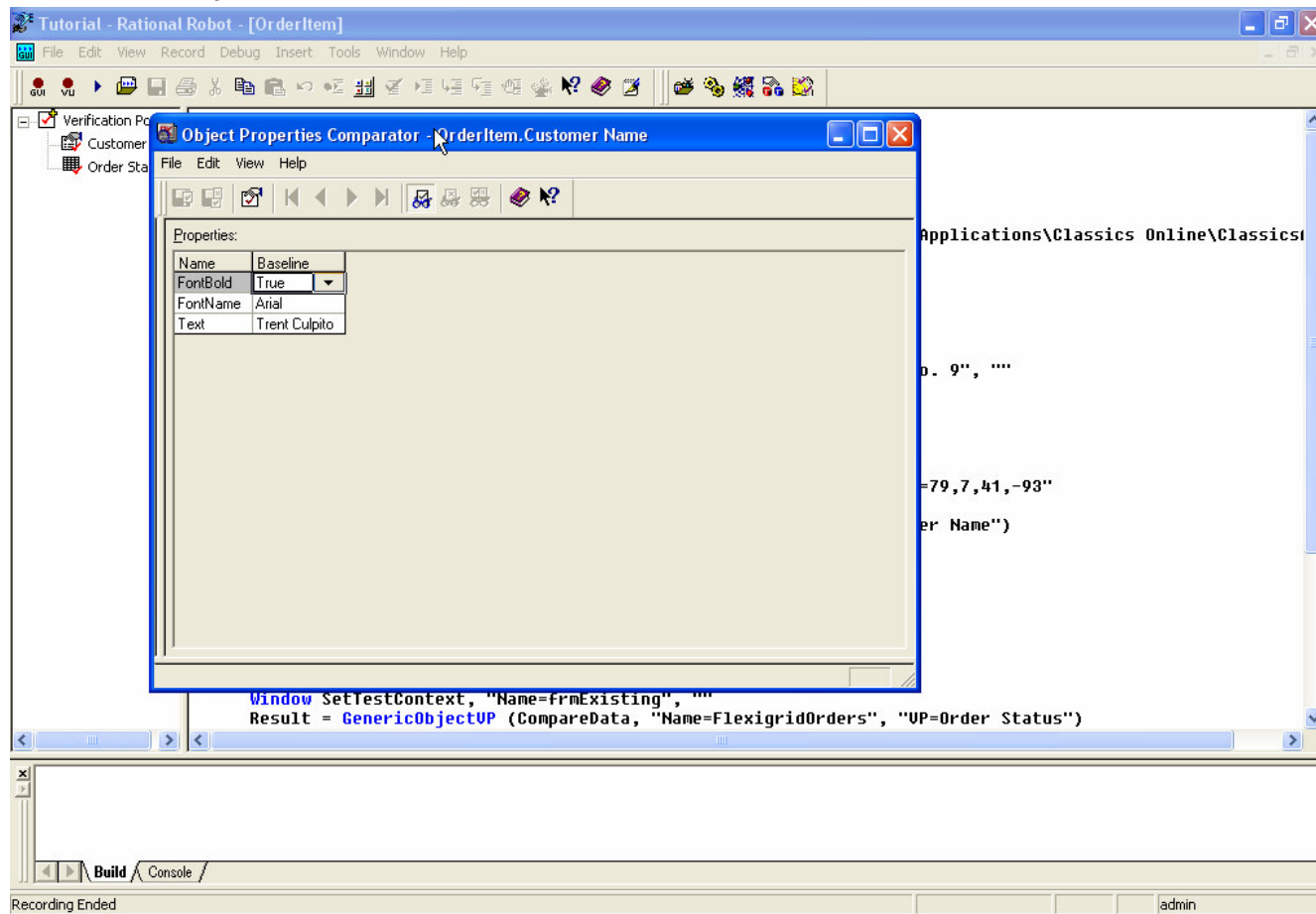
  Window SetContext, "Name=frmMain", ""
  MenuSelect "Order->View Existing Order Status..."

  Window SetTestContext, "Name=frmExisting", ""
  Result = GenericObjectUP (CompareData, "Name=FlexigridOrders", "UP=Order Status")
```

Recording a Test Script

The Test Script

- The panel on the left has a list of the VPs we created.
- Right click on any of them to view their details and edit them.



Rational Robot

Running the Test Script

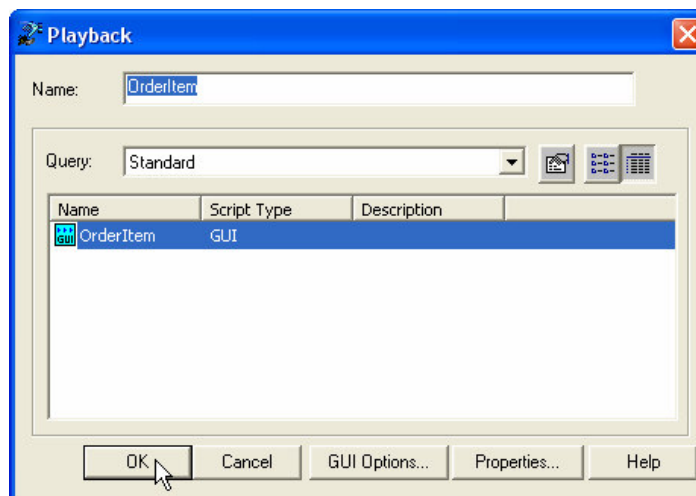
Running the Test Script

Playback on the same build

- To get a feel of how the test script is executed, let us run the test script on the same build that we recorded on.
- During playback, **Do NOT** touch the keyboard or mouse. Any keystroke/mouse-click could disrupt playback.
- To begin, click the Playback Script button on the Robot Toolbar.



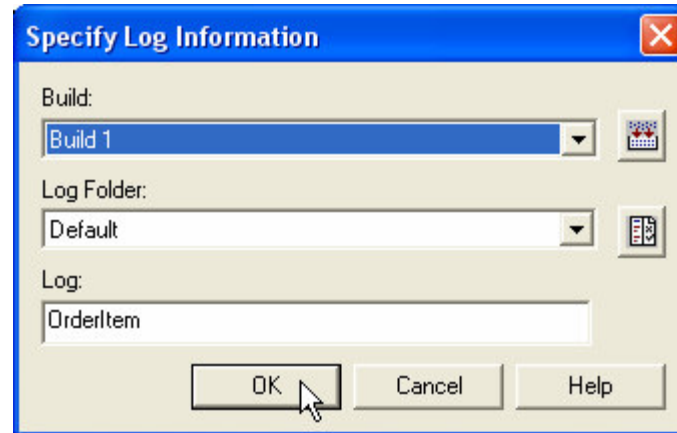
- Select the "Order Item" Script from the list that pops up and click OK.



Running the Test Script

Playback on the same build

- Next the Specify Log Information window pops up. Here you can specify the Build and the log to store to. Since this is our first build, leave the defaults and hit Okay.

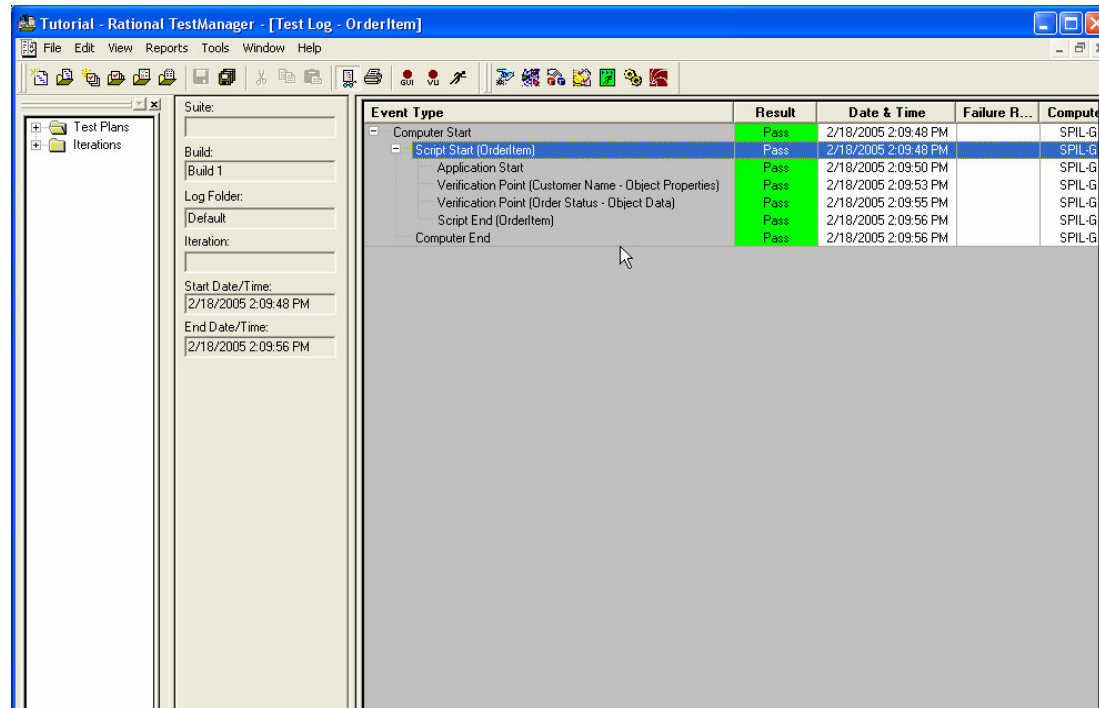


- Now sit back and watch the playback. Notice how much faster a computer can perform the playback compared to a manual playback.

Running the Test Script

Test Log - Viewing the Results

- After Completion of the playback, Rational TestManager opens up and displays the Test Log.



The screenshot shows the Rational TestManager interface with a test log for 'OrderItem'. The log is displayed in a table format with columns for Event Type, Result, Date & Time, Failure R..., and Computd. The log shows a successful execution of the test script, including application start, verification points, and script end.

Event Type	Result	Date & Time	Failure R...	Computd
Computer Start	Pass	2/18/2005 2:09:48 PM		SPIL-G
Script Start (OrderItem)	Pass	2/18/2005 2:09:48 PM		SPIL-G
Application Start	Pass	2/18/2005 2:09:50 PM		SPIL-G
Verification Point (Customer Name - Object Properties)	Pass	2/18/2005 2:09:53 PM		SPIL-G
Verification Point (Order Status - Object Data)	Pass	2/18/2005 2:09:55 PM		SPIL-G
Script End (OrderItem)	Pass	2/18/2005 2:09:56 PM		SPIL-G
Computer End	Pass	2/18/2005 2:09:56 PM		SPIL-G

- Since we ran the test on the same build, as we can expect, both the verification points passed.

Running the Test Script

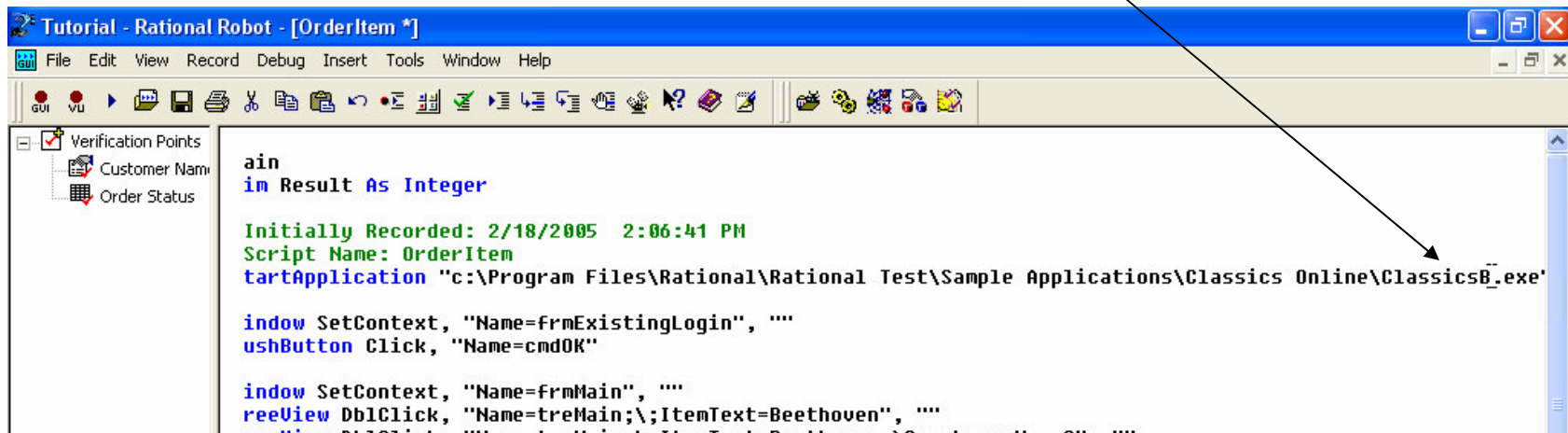
Playback on a faulty build

- Suppose after a week, we are given a new build of Classics Online to test. We want to be able to run the same script on the new build. This is known as **regression** testing.
- For the tutorial, we are provided with two other build, ClassicsB and ClassicsC.

Running the Test Script

Playback on a faulty build

- Let us take a quick look at ClassicsB:
ClassicsB is almost the same as our first build except that two bugs have been introduced. Without manually going through it, lets see if Robot catches them.
- Open up the script we had before, and edit the Start Application to ClassicsB.exe instead of ClassicsA.exe as shown.

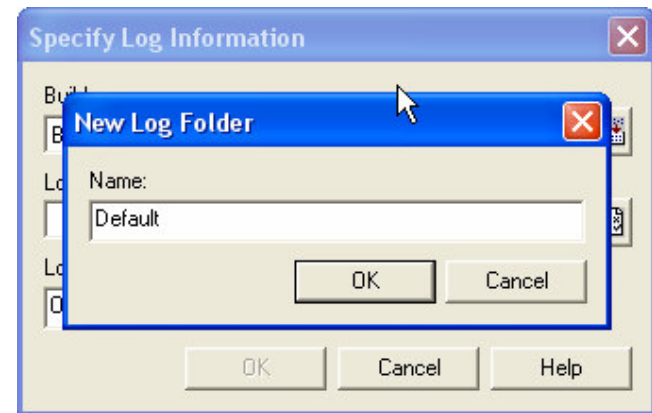
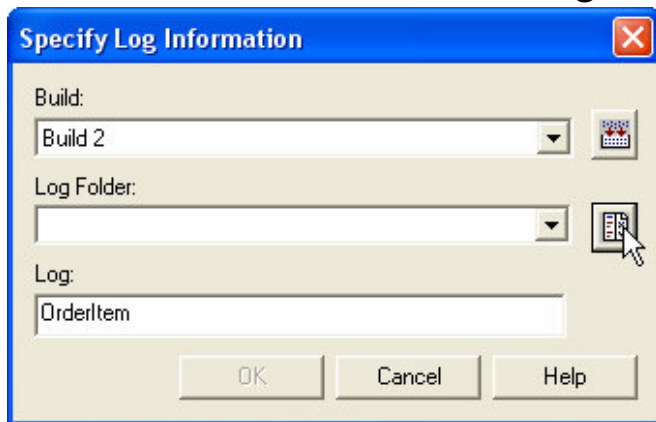


```
Tutorial - Rational Robot - [OrderItem *]  
File Edit View Record Debug Insert Tools Window Help  
gui vU [Icons]  
Verification Points  
  Customer Name  
  Order Status  
ain  
in Result As Integer  
Initially Recorded: 2/18/2005 2:06:41 PM  
Script Name: OrderItem  
StartApplication "c:\Program Files\Rational\Rational Test\Sample Applications\Classics Online\ClassicsB.exe"  
indw SetContext, "Name=frmExistingLogin", ""  
ushButton Click, "Name=cmdOK"  
  
indw SetContext, "Name=frmMain", ""  
reeView Db1Click, "Name=treeMain;\;ItemText=Beethoven", ""  
uselliew Db1Click, "Name=treeMain;\;ItemText=Beethoven \;Sunberry No. 81" ""
```

Running the Test Script

Playback on a faulty build

- Now run the script again by hitting the “Playback Script” button on the Robot toolbar. Select the OrderItem script and click OK.
- This time on the “Specify Log Information” window, change the name of the build to Build 2 and the Log Folder to Default.



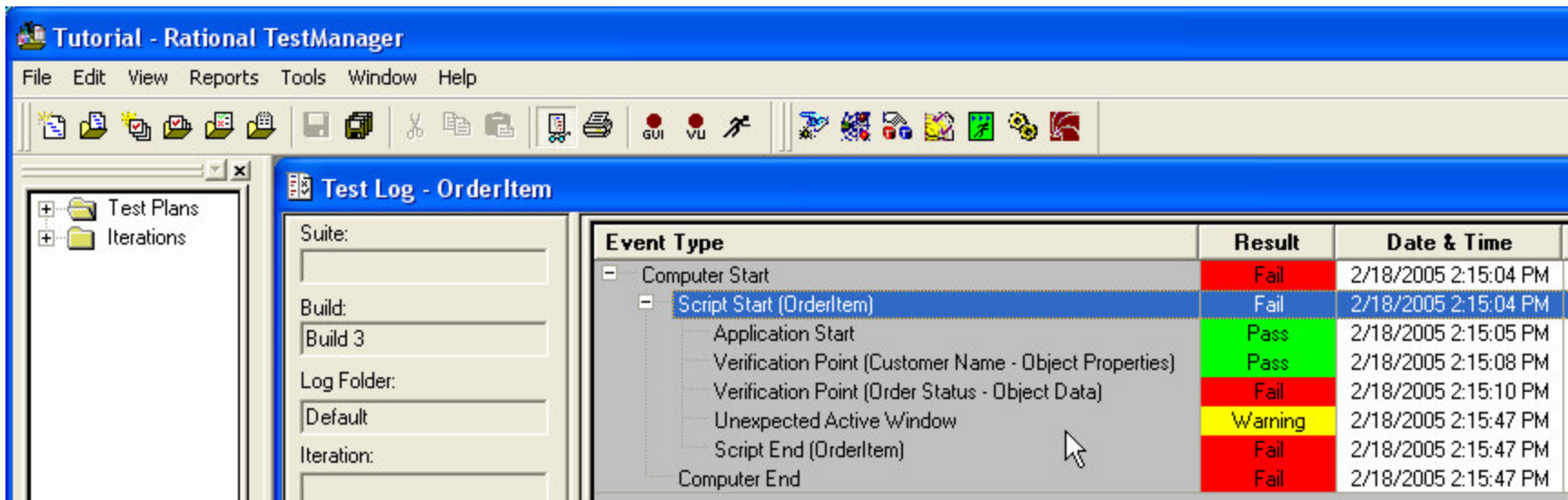
- Run the script. After a while a Network Error window pops up. This is one of the bugs mentioned earlier. Resist the urge to interact, and wait for Robot to take care of it.



Running the Test Script

Playback on a faulty build

- Once again TestManager pops open the Test Log.
- Notice that this time we have a warning called “Unexpected Active Window”. This is Robot’s way of telling us about the Network Error window that popped up. Double clicking on it brings up a screenshot.



The screenshot shows the Rational TestManager interface. The main window is titled "Test Log - OrderItem". On the left, there is a tree view with "Test Plans" and "Iterations". The main area displays a table of test events. The table has three columns: "Event Type", "Result", and "Date & Time". The events are as follows:

Event Type	Result	Date & Time
Computer Start	Fail	2/18/2005 2:15:04 PM
Script Start (OrderItem)	Fail	2/18/2005 2:15:04 PM
Application Start	Pass	2/18/2005 2:15:05 PM
Verification Point (Customer Name - Object Properties)	Pass	2/18/2005 2:15:08 PM
Verification Point (Order Status - Object Data)	Fail	2/18/2005 2:15:10 PM
Unexpected Active Window	Warning	2/18/2005 2:15:47 PM
Script End (OrderItem)	Fail	2/18/2005 2:15:47 PM
Computer End	Fail	2/18/2005 2:15:47 PM

- Also, notice that our second Verification Point failed! To see what caused this failure, double click on it.

Running the Test Script

Playback on a faulty build

- This brings up a window displaying Order Status Window. The baseline that we captured while recording our script as well as the new capture that took place while running the script are shown for comparison.

Grid Comparator - OrderItem.Order Status

File Edit View Help

Baseline

ORDERID	STATUS	COMPOSER	COMPOSITION	QUANTITY	TOTAL	
1	21	Order Initiated	Beethoven	Symphony No. 9	1	17.99

Actual

ORDERID	STATUS	COMPOSER	COMPOSITION	QUANTITY	TOTAL	
1	21	Order Initiated	Bach	Brandenburg Concertos Nos. 1 & 3	1	18.99

Comparison failed: 3,1
Comparison failed: 4,1
Comparison failed: 6,1

Ready

- As you can see, the program ordered a copy of Bach instead of Beethoven. This is a bug in the program that would have been hard to find using manual testing.

Running the Test Script

Resilience to Change

- What would happen if we changed the arrangement of the window in Classics Online? Like moved the panels around or gave new captions to the buttons?
- Would Robot still be able to locate the right buttons and run the script?
- The answer is YES! Robot actually keeps track of the object names, not their properties like location/text, so it will be able to run the same script even after such changes.
- This resilience to change is a key capability in regression testing that minimizes maintenance to test scripts.
- To test this out for yourself, run the script again on the third build, “ClassicsC.exe”. This is a bug-free build, with a slightly better looking interface and the buttons have different text on them.

Summary

This tutorial walked you through some of the basic functionalities of Robot and showed you how to:

- Record a script with Rational Robot
- Set up Verification points in the script
- Edit a script in the Editor Window
- Playback the script against future builds as part of regression testing
- Analyze the results of the playback using TestManager's Test Log.