Rational Robot 基础使用手册

目录

第一章 绪论

一、概述

Rational Robot 可开发三种测试脚本：用于功能测试的 GUI 脚本、用于性能测试的 VU 以及 VB 脚本。

Rational Robot 作用

1、 执行完整的功能测试。记录和回放遍历应用程序的脚本，以及测试在查证点（verification points）处的对象状态。

2、 执行完整的性能测试。Robot 和 Test Manager 协作可以记录和回放脚本，这些脚本有助于你断定多客户系统在不同负载情况下是否能够按照用户定义标准运行。

3、 在 SQA Basic、VB、VU 环境下创建并编辑脚本。Robot 编辑器提供有色代码命令，并且在强大的集成脚本开发阶段提供键盘帮助。

4、 测试 IDE 下 Visual Basic、Oracle Forms、Power Builder、HTML、Java 开发的应用程序。甚至可测试用户界面上不可见对象。

5、 脚本回放阶段收集应用程序诊断信息，Robot 同 Rational Purify、Quantify、Pure Coverage 集成，可以通过诊断工具回放脚本，在日志中察看结果。

Robot 使用面向对象记录技术：记录对象内部名称，而非屏幕坐标。若对象改变位置或者窗口文本发生变化，Robot 仍然可以找到对象并回放。

同其他组件集成使用 Robot

1、 Rational Administrator：用于集中管理 Rational 项目。

2、 Rational Test Manager (日志)和 Comparators：用于回顾和分析测试结果。

3、 Rational Site Check：用于管理互联网和企业互联网网页站点。

同其他 Rational 产品集成使用 Robot

1、 用 Rational TestFactory 测试应用程序；

2、 用 Rational ClearQuest 管理缺陷；

3、 在回放期间收集诊断信息；

4、 用 Rational TestManager 做性能测试；

5、 用 Rational RequisitePro 做需求管理。

二、基本概念

VU 和 GUI 脚本组成部分

1、 由 Robot 或者 Test Manager Suite 生成的可运行文件。
2、 脚本属性集，例如类型和脚本目标。

VU 和 GUI 脚本的异同

方面	GUI 脚本	VU 脚本
并发性	在一台计算机上同时只能执行一个 GUI 脚本。	在一台计算机上同时可以执行多个 VU 脚本。
语言	包括对 GUI 对象的键盘敲击以及鼠标点击行为，脚本用 SQA Basic 语言写成。	包括客户端发送到服务器的要求，脚本用 VU 语言写成。
测试领域	用于功能测试和性能测试。	通常用于加入用户负载的性能测试，例如：测试不同负载下服务器响应时间。
查证点	可以包括查证点，用于比较记录回放时捕获的信息。	不支持查证点。
执行	既可在 Robot 中执行，也可以作为 Test Manager Suite 的一部分执行。	作为 Test Manager Suite 的部分执行。

在同一脚本中，不能混合 SQA Basic 和 VU 代码。

Rational Test 中的两种模拟用户

1、 GUI 用户：单用户，模拟前台的实际用户操作。
2、 虚拟测试者：多用户，虚拟测试者模拟发送到数据库、Tuxedo 或者 Web 服务器的请求，Robot 记录网络流量等后台，忽略前台 GUI 操作。

Rational Test 中的两种测试类型

1、 功能测试：Robot 是一种用于功能测试的计划、开发、执行和分析工具；
2、 性能测试：Robot 和 TestManager 结合用于性能测试。

第二章 使用

一、GUI 脚本

（一）、设置以及预定义

应该先在应用程序开发和测试过程早期制定计划使用 Robot。如果在应用程序初始版本中存在任何 Windows GUI 对象（比如菜单、对话框），可以使用 Robot 来记录相应的查证点。

编写脚本之前的准备工作

1、 为脚本建立可预计的起始和结束状态；
2、 安装测试环境；
3、 创建模块脚本；
4、 转换应用程序使其可测试。

加载 IDE Extensions

加载 IDE Extensions。始终加载对 C++应用程序的支持。

设置 GUI 记录选项

GUI 记录选项提供如何记录和产生 GUI 脚本的 Robot 指令。可以在记录之前，也可以在记录过程早期设置这些选项。

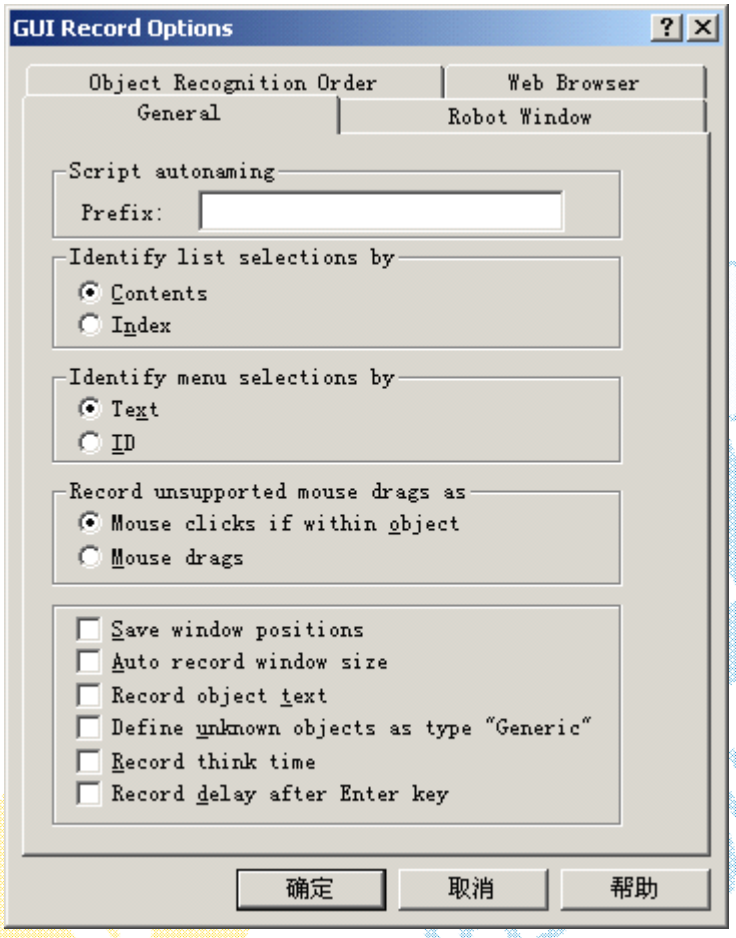
1、 按照如下任一步骤打开 GUI Record Options 对话框：
   - 在开始记录之前，点击 Tools 菜单下 GUI Record Options；
   - 在快捷栏上点击 Record GUI Script 按钮开始记录，在 Record GUI Script 对话框上点击“Opitions…”按扭；
2、 在每页选项卡上设置选项，需要细节帮助可以点击对话框顶部“?”按钮，再单击项目；
3、 单击确定按钮。

选项设置中的一些重要特性

- 脚本自动命名；
- 控制 Robot 响应未知对象；
- 选择对象参考顺序。

脚本自动命名

1、 打开 GUI Record Options 对话框；
2、 在 General 页面，在 Prefix 框中输入前缀，如果不希望有前缀，则清空该编辑框，以后每次记录新脚本都需要输入名称；
3、 点击确定按钮。



控制 Robot 响应未知对象

1、 打开 GUI Record Options 对话框；
2、 在 General 页面对 Define unknown objects as type "Generic"操作，选中表示 Robot 遇上未知对象时，将其作为通用对象处理，否则在记录时 Robot 挂起，打开 Define Object 对话框来关联对象到类；
3、 点击确定。

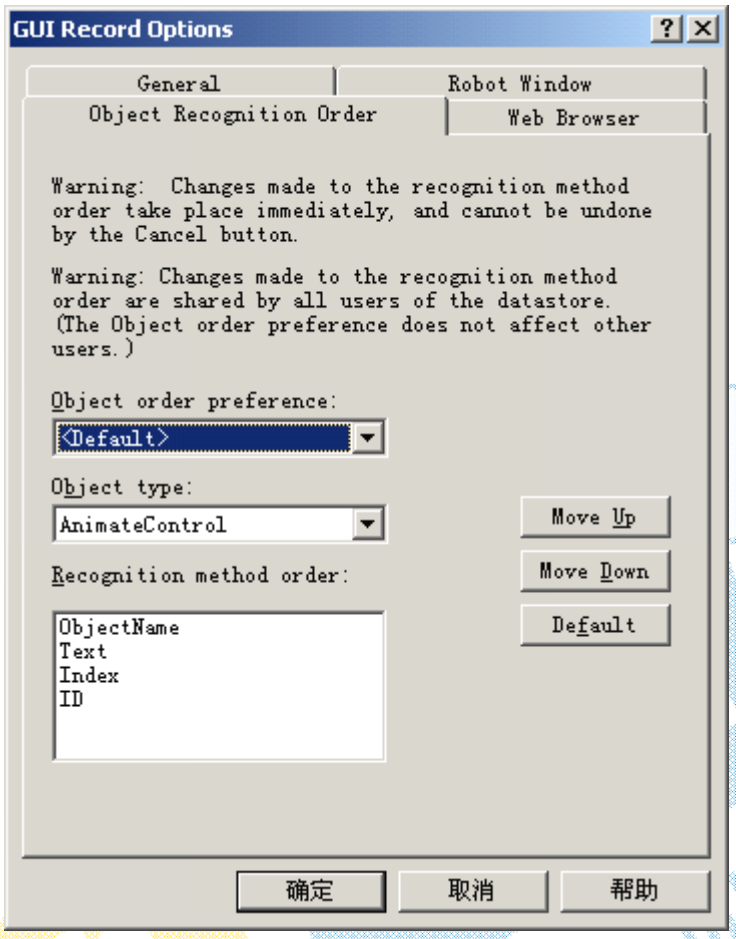
也可以在开始纪录之前映射对象类型以及类。

类名称到对象类型的定制映射关系对于项目的所有用户都是共享的。

选择对象参考顺序

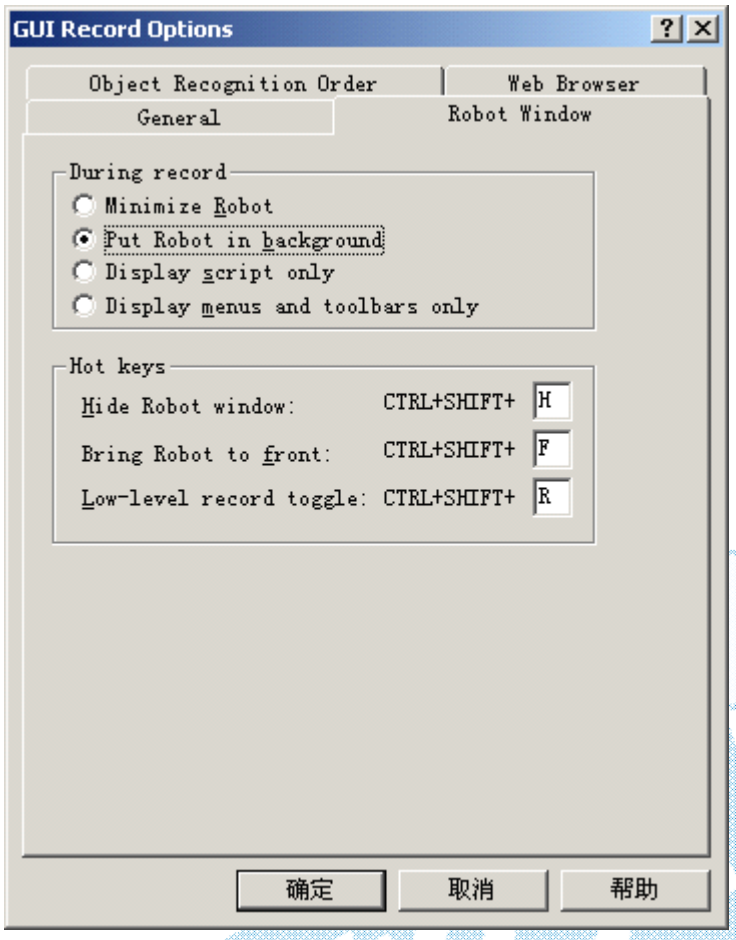
Robot 有两种识别标准对象类型方法顺序的预定义参考，缺省识别顺序和 C++识别顺序（用于测试 C++应用程序）。改变对象参考顺序步骤如下：

1、 打开 GUI Record Options 对话框；
2、 点击 Object Recognition Order 页面；
3、 更改对象识别参考顺序；
4、 点击确定按钮。



设置 Robot 窗口选项

需要在记录期间改变 Robot 主窗口缺省行为，按照以下步骤配置：
1、 打开 GUI Record Options 对话框；
2、 选中 Robot Window 页面；
3、 在 During Record 下点击选项；
4、 点击确定按钮。

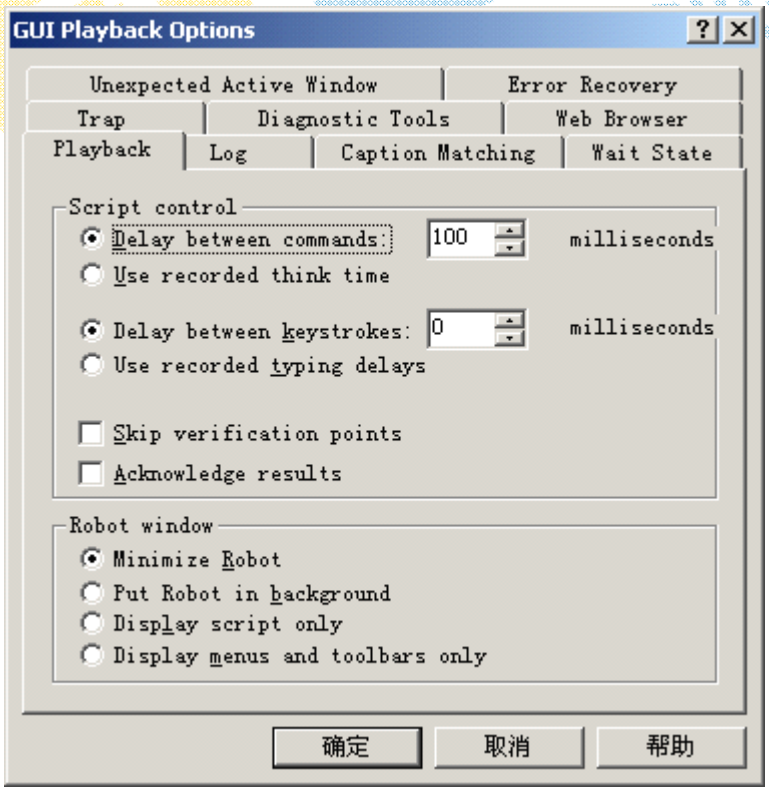


在回放时改变 Robot 主窗口缺省行为：

1、 打开 GUI Playback Options 对话框；

2、 在 Playback 页面，点击 Robot Window 下的选项按钮；

3、 点击确认按钮。



改变热键

可以利用热键隐藏和显示 Robot Window，并在面向对象和低级录制之间切换。Robot 热键在录制期间激活，并且不记入脚本，热键有缺省值。

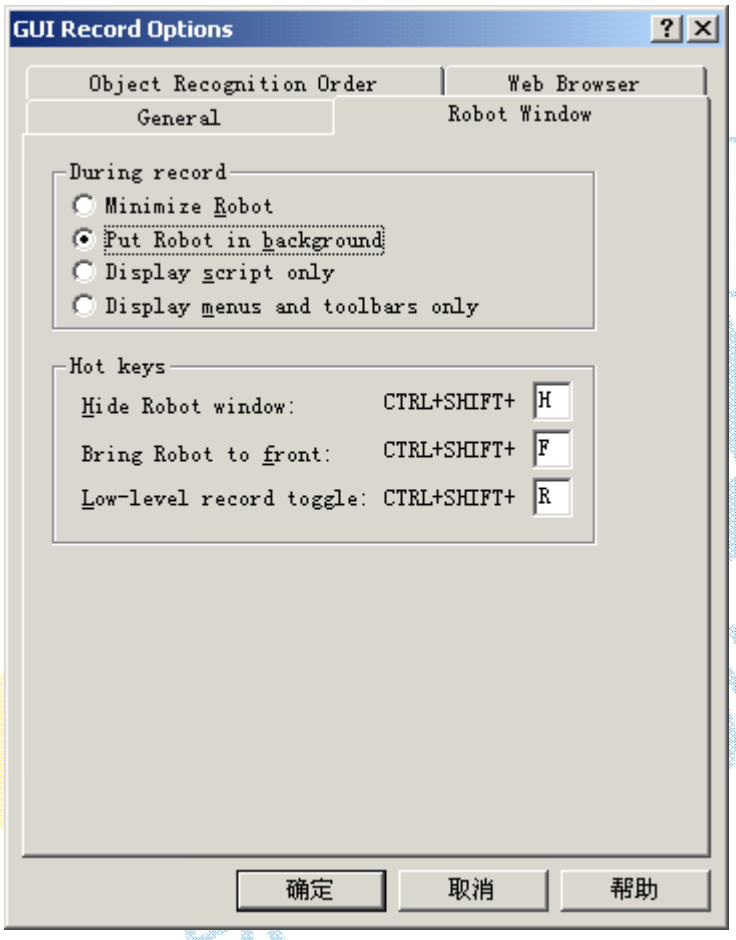
改变热键步骤如下：

1、 打开 GUI Record Options 对话框；

2、 点击 Robot Window 页面；

3、 改变热键；

4、 点击确定按钮。



（二）、记录 GUI 脚本

除了面向对象记录技术，Robot 还支持底层记录技术，该技术在需要跟踪鼠标行为细节的功能测试时很有用，比如绘制应用程序。

GUI 记录工作流程

1、 按照指导，为脚本确定可预测的起始状态和结束状态，安装测试环境，创建模块脚本，并且使应用程序可测。

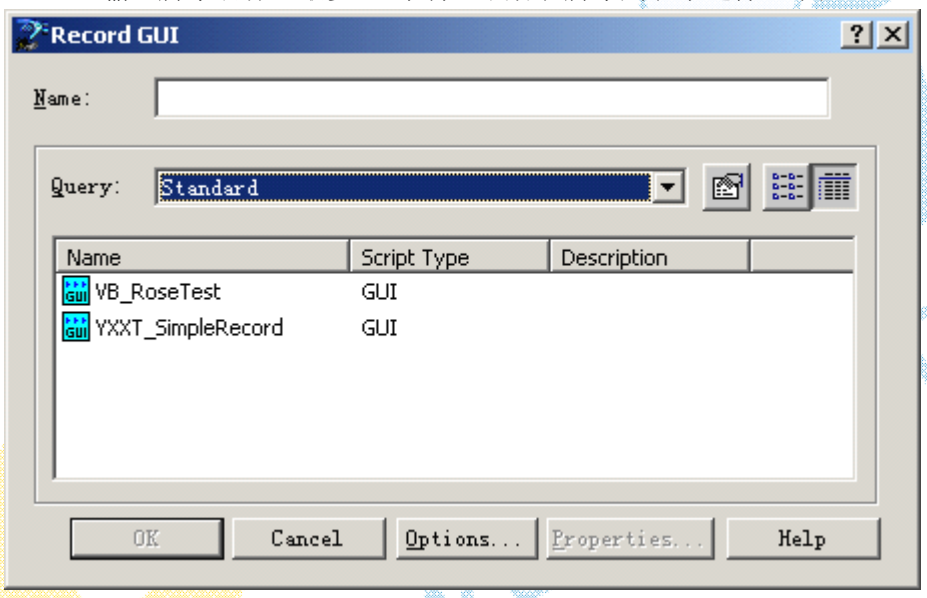
2、 设置记录选项，也可在记录开始后设置；

3、 开始记录；

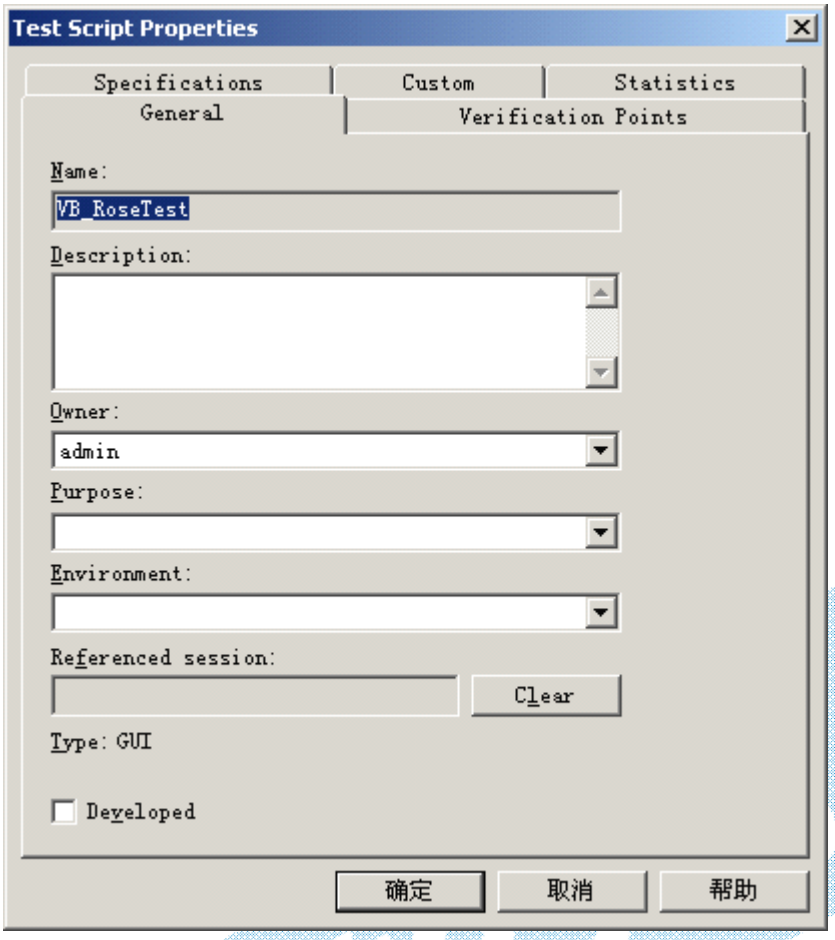
4、 在测试环境下启动应用程序，必须按照期望回放的方式正确启动应用程序；
5、 在应用程序中执行系列行为；
6、 加入必要的特写，例如查证点、注释以及计时器；
7、 如有必要，将面向对象记录切换至底层记录方式；
8、 结束记录会话；
9、 可选操作，通过文件菜单下属性菜单项定义脚本属性，在 Test Manager 中也可以定义脚本属性。

记录新的 GUI 脚本

1、 按照指导，为脚本确定可预测的起始状态和结束状态，安装测试环境，创建模块脚本，在 Test Manager 中建立脚本计划，并且使应用程序可测；
2、 如果可能，使应用程序可测，加载 IDE Extensions；
3、 记录之前设置记录选项，也可以在开始后设置；
4、 点击快捷栏上的“Record GUI Script”快捷按钮；
5、 输入脚本名称（最多 40 字符）或者从脚本列表中选择一个；



6、 要改变记录设置，点击“Options…”按钮，完成设置后点击确定按钮；
7、 如果选中一个预定义或者已记录脚本，你可以通过 Properties 菜单项设置脚本属性，设置完成后确认退出；



8、 开始记录，以下事件一次发生，

- 如果选中已存在脚本，Robot 询问是否覆盖，
- 缺省情况下，Robot 最小化，
- 出现浮动 GUI Record 快捷栏，可以通过它暂停或者停止记录、重新显示 Robot，在脚本中插入特写；

9、 按照以下步骤启动测试环境下的应用程序：

- 点击 GUI Record 快捷栏的 Display GUI Insert Toolbar 按钮，
- 点击 GUI Insert Toolbar 上适当的起始按钮，
  - 启动应用程序按钮：用于启动应用程序（除用 Quantify 或者 Pure Coverage 回放的 HTML、Java 应用程序），
  - 启动 Java 应用程序按钮：用于启动由 Quantify 或者 Pure Coverage 回放的 Java 应用程序，
  - 启动浏览器按钮：用于启动 HTML 应用程序；

10、 在应用程序中执行系列行为；

11、 如果需要则插入特写，可以插入查证点、注释、计时器等；

12、 如有必要，将面向对象记录模式切换至底层记录模式；

13、 记录完成，点击 GUI Record 快捷栏上 Stop Recording 按钮，Robot 主窗口显示如下信息：

- 查证点和底层脚本显示在左侧的 Asset 窗格，
- 文本和脚本显示在右侧的脚本窗格，

编译或者回放脚本时，编译结果显示在 Output 窗口的 Build 页面上；

14、 可选操作：设置脚本属性；

记录期间恢复 Robot 主窗口

1、 点击 GUI Record 快捷栏上 Open Robot Window 按钮；
2、 点击 Windows 任务栏上 Robot 按钮；
3、 使用 CTRL+SHIFT+F 热键显示窗口，CTRL+SHIFT+H 隐藏窗口；

暂停和唤醒 GUI 脚本记录

暂停记录：点击 GUI Record 快捷栏上 Pause 按钮，Robot 指示操作暂停。

- 点击暂停按钮；
- 状态条显示“Recording Suspended”；
- 在 Record 菜单项 Pause 菜单项左侧出现选中标志。

唤醒记录：再次点击 Pause 菜单项。
唤醒操作和暂停操作时，应该在应用程序中处于同一状态。

在 GUI 记录期间定义未知对象

记录期间，Robot 只识别标准的 Windows GUI 对象和一些定制对象。你可以设置记录选项这样 Robot 自动和具有通用类型的不能识别的对象连接。如果未设置选项，如果你点击 Robot 不能识别的对，Robot 打开“Define Object”对话框，用该对话框把该对象映射成一种已知对象。
记录期间定义未知对象
1、 在定义对象对话框中从“**Type**”列表中点击一种与未知关联的对象类型
2、 点击“**ok**”按扭继续记录。

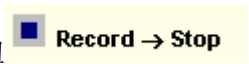
切换至底层记录

1、 按下 CTRL+SHIFT+R 键；
2、 点击 GUI Record 快捷栏上 Open Robot Window 按钮（或者按下 CTRL+SHIFT+F 按钮），将 Robot 置于前台，点击 Record 菜单下 Turn Low-Level Recording On/Off 菜单项；

切换至底层记录方式之后，Robot 进行如下操作：
1、 在不可编辑的二进制脚本中记录底层行为，并在项目中保存；
2、 给底层脚本分配连续数字，在 Script 窗体的 Assert 窗格中显示，数字位于 Low-Level Scripts 下方；
3、 在引用底层脚本文件的脚本中加入“PlayJrnl”命令；

回放期间，PlayJrnl 命令调用底层文件，该文件回放记录的实时行为，这不同于面向对象记录，面向对象记录方式检查测试环境下应用程序的 Windows 对象，而不依赖于精确的时间和屏幕坐标。

结束 GUI 脚本记录

通过点击 GUI Record 工具条上的 **Stop Recording** 按扭 

结束录制 GUI 脚本。

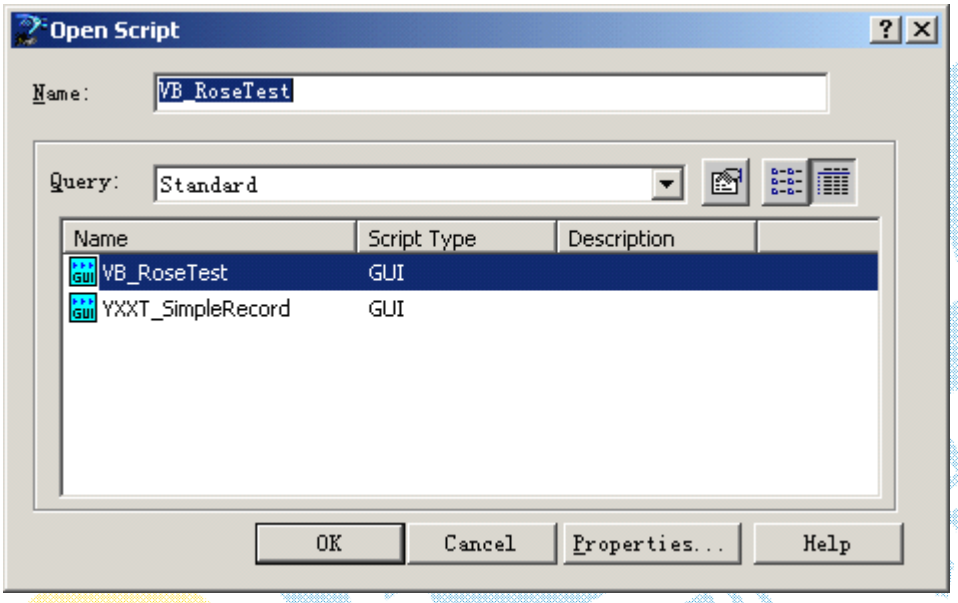
记录结束时，应该使测试下应用程序和开始记录时的状态一致。这样，可以不必人工重置环境就能回放脚本。

如果从 Windows 桌面启动应用程序，应该在桌面停止记录。若从主窗口启动记录，则在主窗口停止记录，确定主窗口状态相同。例如：如果应用程序使一个编辑器，记录开始时启动应用程序没有任何文本，则在结束录制时确认没有开启任何文档。

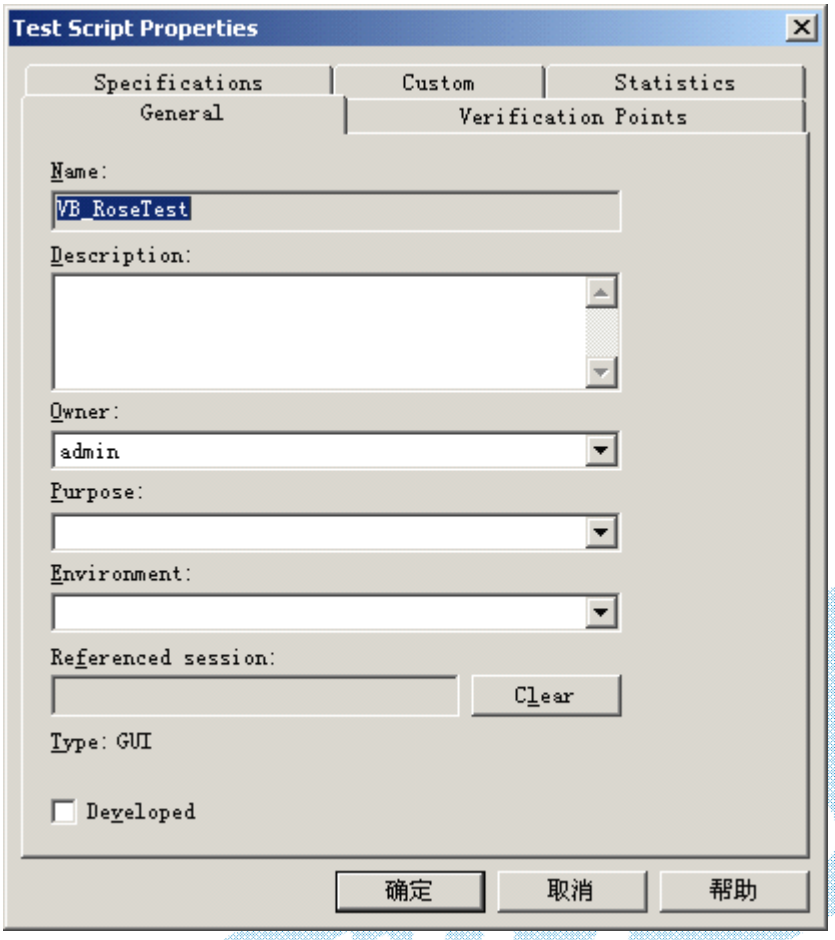
定义脚本属性

在 Robot 中记录脚本之后，可以定义脚本属性：

1、 如果已经打开脚本，点击 File 菜单下 Properties 菜单项，如果脚本没打开，点击 File 菜单下 Open 菜单 Script 菜单项，选中脚本，。点击 Properties 按钮；



2、 在 Script Properties 对话框中，定义属性；



3、 确认退出。

如果记录覆盖了一个已经存在的脚本，则已经存在的属性将应用到新的脚本上。

手工 GUI Script 编码

到目前为止，最快的 GUI 脚本生成方式是利用 Robot 记录行为并自动生成脚本，然而，也可以使用 SQA Basic 脚本语言编写 GUI 脚本。

手工编写脚本的步骤如下：

1、 在 Robot 中，点击 File 菜单下的 New 子菜单的 Script 菜单项；

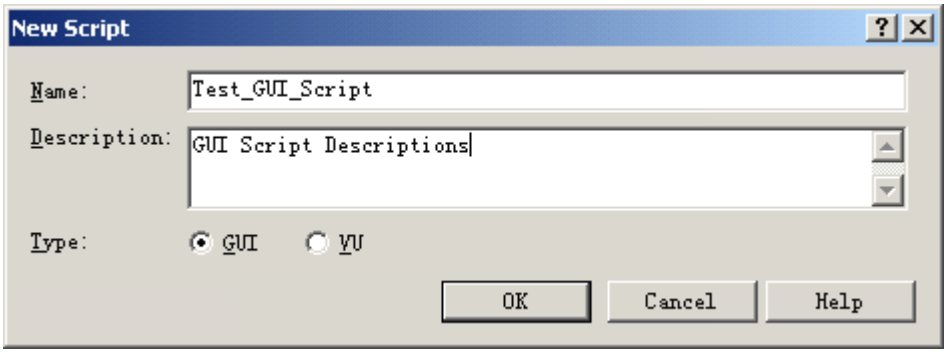
2、 输入脚本名称（最多 40 字符），可以加入脚本描述；

3、 点击 GUI；

4、 点击确认，Robot 产生一个带主程序头的空脚本；

5、 开始 GUI 脚本编码。

SQA Basic 语言参考见帮助。



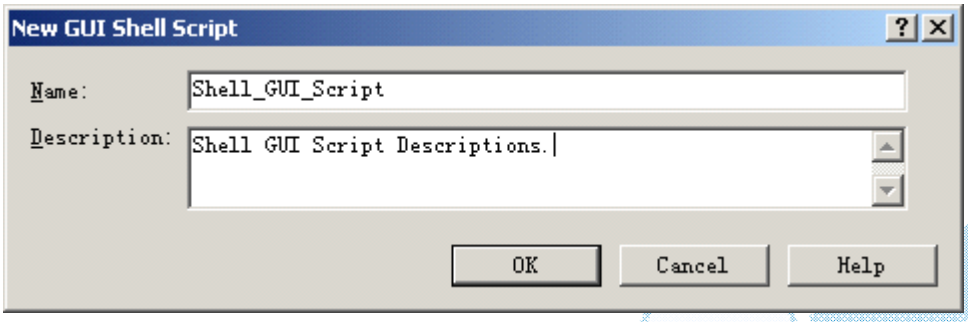
创建 Shell Scripts 顺序回放 GUI Scripts

创建外壳脚本之前，应该先记录需要引用的独立脚本。

建立外壳脚本的顺序：

1、 点击 File 菜单的 New 子菜单的 GUI Shell Script 菜单项；

2、 输入脚本名字（最多 40 字符）；

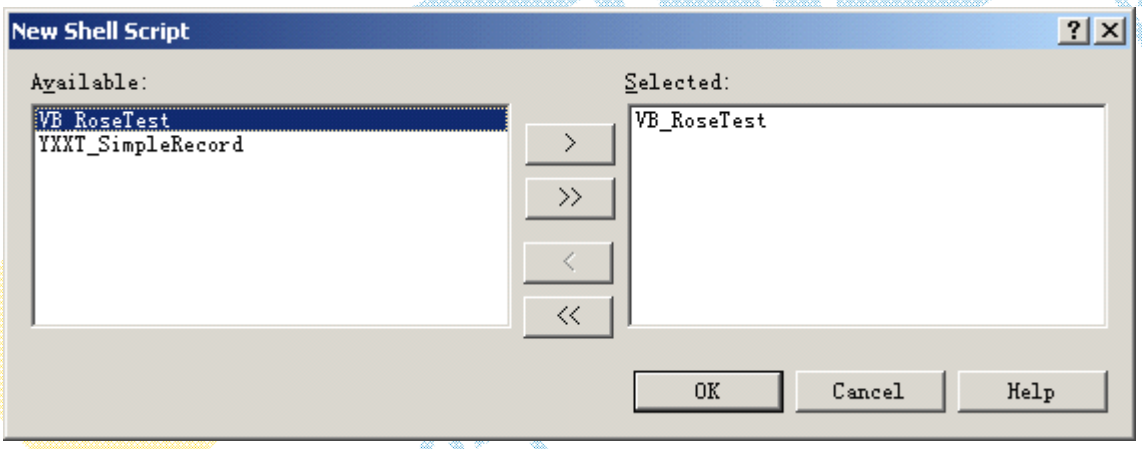


3、 可选操作：输入脚本描述；

4、 确定；

5、 要增加脚本，在 Available 列表中选中一个或者多个脚本，点击“>”或者“>>”按钮，Robot 按照 Selected 列表中的脚本顺序回放脚本；

6、 确定。



在外壳脚本中，用“Call Script+脚本名字”引用包含的脚本。

（三）、在 GUI Script 中加入特写

在 GUI 脚本中启动应用程序

要成功测试 Oracle Forms、HTML、Java、Delphi、C++和 Visual Basic 4.00 应用程序中的对象，应该在开始记录脚本之前允许应用程序。

启动应用程序时，可以特别说明回放时需要该应用程序在 Rational 诊断工具环境下启动。

在脚本中启动应用程序的步骤如下：

1、 记录时，点击 GUI Record 快捷栏上 Display GUI Insert Toolbar 按钮，编辑时，定位脚本光标，点击 Standard 快捷栏的 Display GUI Insert Toolbar 按钮；

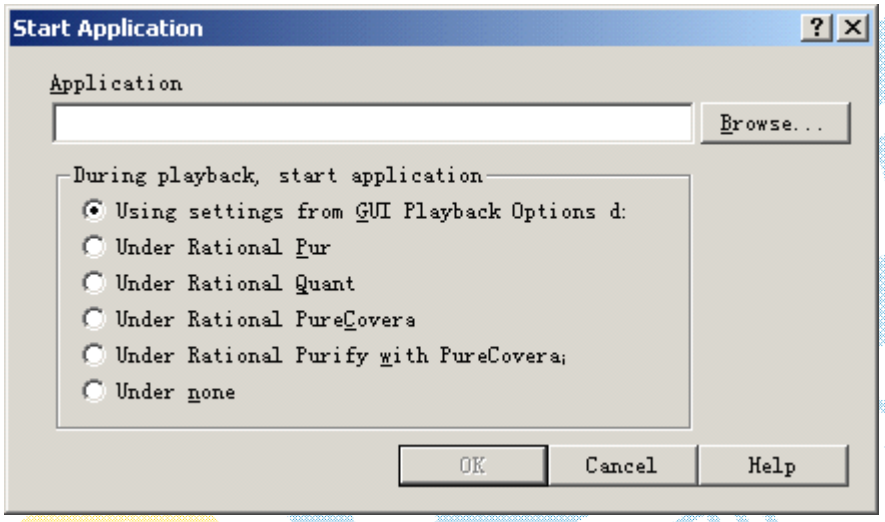
2、 点击 GUI Insert 工具条上适当的启动按钮（启动应用程序、启动 Java 应用程序、启动浏览器）；

启动应用程序

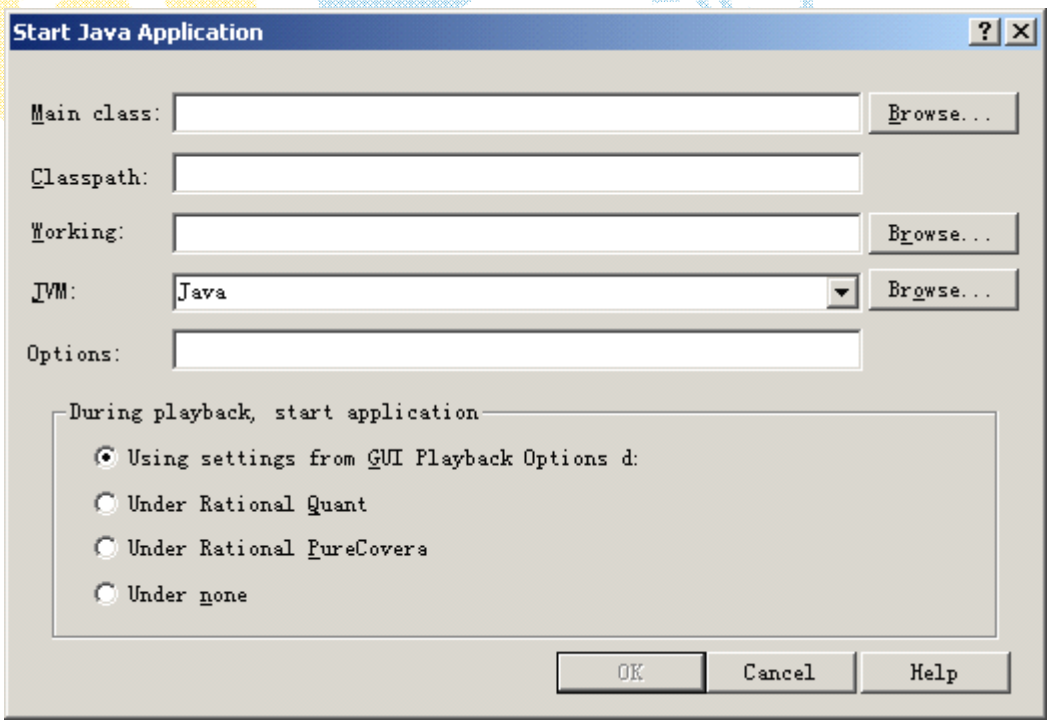
启动 Java 应用程序

启动浏览器

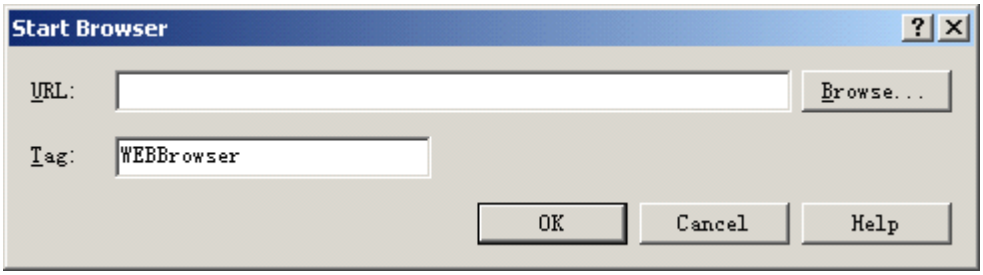
3、 填写对话框并确定；



启动应用程序



启动 Java 应用程序



启动浏览器

4、 开始记录并且编辑脚本。

回放过程中，Robot 运行到脚本中响应的命令时就启动特定的应用程序。

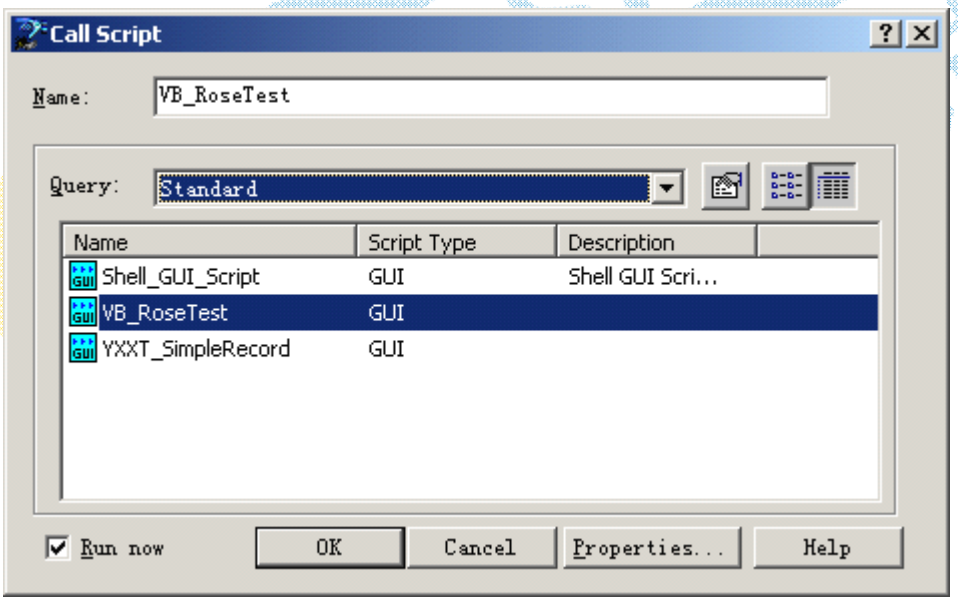
在其它脚本中插入调用

在记录或者编辑 GUI 脚本的状态，可以插入已有的 GUI 脚本的调用，避免了重复的应用程序行为。

1、 如果处于记录状态，点击 GUI Record 快捷栏上的 Display GUI Insert Toolbar 按钮，如果处于编辑状态，在 Standard 快捷栏上点击 Display GUI Insert Toolbar 按钮；

2、 点击 GUI Insert 快捷栏上的 Call Script 按钮；

3、 从列表中选择 GUI 脚本，要改变脚本列表，选择 Query 列表；



4、 如果测试环境依据脚本的执行结果则选中 Run now，如果脚本执行不改变应用程序状态则清空 Run now，无论选中与否，Robot 都将对该脚本的调用加入脚本中，选中则立即执行；

5、 确定以继续录制或者编辑。

在 GUI 脚本中插入计时器

1、 如果在记录状态，点击 GUI Record 快捷栏的 Display GUI Insert Toolbar 按钮，如果在编辑状态，点击 Standard 快捷栏的 Display GUI Insert Toolbar 按钮；

2、 在 GUI Insert 工具栏上点击 Start Timer 按钮；

3、 输入计时器名称（最多 40 字符）后确定，如果要启用多个计时器，确定每个计时器有不同的名字；

4、 执行计时行为；

5、 执行完计时行为，立即点击 GUI Insert 工具栏上 Stop Timer；

6、 在计时器列表中选择一个开启的计时器，确认。

在 GUI 脚本中插入注释

记录或者编辑时可以插入注释，以利于文档和脚本编辑。

1、 记录期间，单击 GUI Record 工具栏上的 Display GUI Insert Toolbar 按钮，编辑期间，单击 Standard 工具栏上 Display GUI Insert Toolbar 按钮；

2、 点击注释按钮；

3、 输入注释（最多 60 字符）；

4、 确认之后继续记录或者编辑。

Robot 在单引号之后加入注释，缺省为绿色。将注释改为非注释，选中文本，单击 Edit 菜单下 Comment Line 或者 Uncomment Line 菜单项。

在 GUI 脚本中插入 Log Message

在记录或者编辑状态，可以在 GUI 脚本中插入日志消息、描述和结果。回放阶段，Robot 在日志中插入这些信息。可以利用这些日志消息文档化回放的脚本。

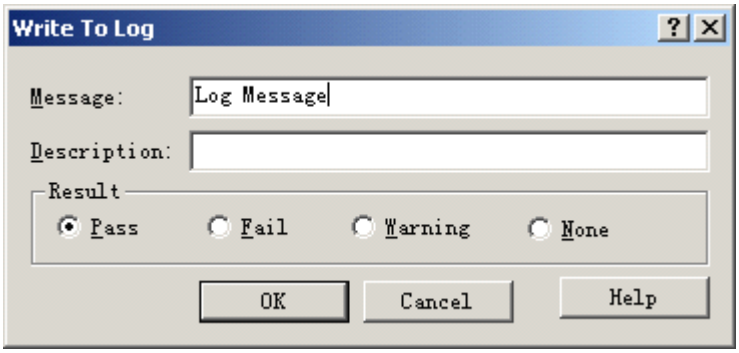
1、 记录状态下，单击 GUI Record 工具栏上的 Display GUI Insert Toolbar 按钮，编辑状态下，单击 Standard 工具栏上 Display GUI Insert Toolbar 按钮；

2、 单击 GUI Insert 工具栏上 Write to Log 按钮；

3、 输入消息（最多 60 字符）；

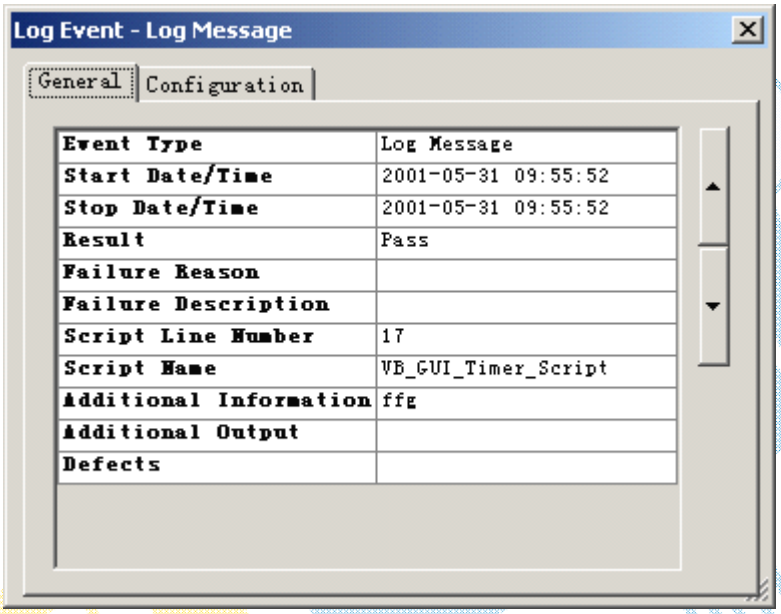
4、 可选操作：输入描述（最多 60 字符）；

5、 选择一个结果：Pass、Fail、Warning、None；



6、 确定继续记录或者编辑。

回放之后，在 Test Manager 日志中查看日志和消息。Event Type 列中显示消息，Result 列中显示结果。查看相关描述，选中日志事件，单击 View 菜单下 Properties 菜单项，打开 Result 页面。



事件日志

在 GUI 脚本中插入延迟

1、 记录阶段，单击 GUI Record 工具栏上 Open Robot Window 按钮；

2、 在脚本中定位光标；

3、 单击 Insert 菜单下 Delay 菜单项；

4、 输入延迟毫秒数；

5、 确认之后继续录制或者编辑。

（四）、使用查证点

在创建查证点时的任务

创建查证点

选择测试对象

选择查证方法

选择验证方法

在数据网格中使用数据

在查证点查看基线文件

复制查证点

重新命名查证点

删除查证点

（五）、使用 **Datapools**

如何理解 Datapools

Datapool 是一个测试数据集。它为脚本回放期间提供数据值给脚本变量。
Datapool 让你自动在大数据量的情况下（潜在的包含数个虚拟测试人执行上千条事务）提取测试数据给虚拟测试人。

Datapool 作用：

1、每个虚拟测试人能在脚本运行时发送实际数据（独一的数据）给服务器。
2、单一的虚拟测试人多次执行相同的事务，能在每次执行事务发送实际数据给服务器。
如果在回放脚本期间不用数据源，每个虚拟测试人会发送相同的数据给服务器（此数据是记录脚本捕获下的数据）。
例如：假使你在记录 vu 脚本时发命令数 53328 给数据库服务器，若有 100 个虚拟测试人在运行这个脚本，则命令数 53328 会给服务器发送 100 次。如果运用 Datapool，每个虚拟测试人会发送不同命令数给服务器。

Datapool 结构：

Datapool 用.csv 扩展名存文件，此文件有如下特征：
1、 每行包含一项记录。
2、 每项记录包含被 separator character 限定的 datapool 值域，象（,）
3、 datapool 值域可包含脚本。
4、 datapool 文件的每个 column 包含 datapool 值域的列表。
5、 如果值是附载双引号内，这单一的值包含一个 separator character 域，如："jones,Robert"在记录中是单一的值，不是两个。当值被存储在 datapool 文件中才用引号。引号不是供给应用程序的值的一部分。
6、 一个单一的值可包含内含行。例如："jones,robert"bob""是一个记录的单一值，不是两个。
.csv 和.spc 是存储在 Robot 工程的 datapool 目录中。
下面是一个有三行数据的 datapool 文件的事例：
John,Sullivan,238 Tuckerman St,Andover,MA,01810
Peter,Hahn,512 Lewiston Rd,Malden,MA,02148
Sally,Sutherland,8 Upper Woodland Highway,Revere,MA,02151
注意：如果 datapool 包含复杂的值（如，内含行，datapoo 值包含 field separator characters），应在 datapool editor 观察（或其他文本编辑器如 Microsoft Excel）并使之成为自己期望的确切的 datapool columns

datapool 编辑器：

当 Robot 编辑 datapool 值，用 Configure Datapool in Script 对话框编辑。
观察或编辑现有 Datapool：
1、 如果 Datapool 将编辑的脚本未打开，击 File --Open ---Script 打开
2、 击 Edit --Datapool Information 打开在脚对话框的 Datapool 设置。
3、 可接受脚对话框的 Datapool 默认设置，也可做些调整。可查看帮助。
4、 完成设置，按确定。
5、 按 Edit Existing Data.
6、 在 Datapool 编辑对话框，适当校正 Datapool 值。
7、 完成校正 Datapool 值，按保存，关闭。

如何使用 Datapools

1、 GUI 脚本中增加 Datapool 命令：
记录会话时向应用程序赋了值，记录结束后，编辑脚本并执行以下基本操作
1）参考 SQAUTIL.SBH 头文件；

2）用记录时提供的值替换变量；

3）增加 Datapool 命令打开 Datapool，从 Datapool 中取一行数据，从该行中找到个体值，将每个值赋给脚本变量。

2、 建立及合成 Datapool：

1）点击菜单 **File → Open → Script** 打开脚本

2）点击菜单 **Edit → Datapool Information** 在脚本对话框中打开 Datapool 配置选项；

3）采用缺省配置，或作适当的改变。需要帮助击对话框顶部的?，再点击需帮助的条目；

4）修改完后点击按扭 **Save**；

5）做以下任一操作：

- 击 **Create** 定义及组成一个新的 Datapool，此时出现 Datapool Specification 对话框，若 datapool 已经存在，则没有 **Create** 按扭，而是 **Edit Specification** 按扭；
- 若此时不想定义生成 Datapool 则击 **Close;**

6）在 Datapool Specification 对话框中，用 Datapool 字段定义 Datapool 栏；

7）要往 datapool 中插入新列：

a、点击要插入的 datapool 列的行；

b、根据要插入的 datapool 列点击 **Insert before** 或 **Insert after**；

c、输入新列的名称（最大为 40 个字符）；

d、该新 datapool 列赋予数据类型。

8）定义完 datapool 栏后，在 No. of records to generate.中输入一个数字；

9）点击 **Generate Data** 生成数据；

10）点击 **Yes** 可看到生成数据的摘要。

3、 编辑 Datapool 定义的列：

操作基本同 2 项，区别：第五步为点击 **Edit Specification** 打开 Datapool Specification 对话框，在此可以修改 datapool 列的定义。无第 6）步；

4、 编辑 Datapool 值：

1） 点击菜单 **File → Open → Script** 打开脚本；

2） 点击菜单 **Edit → Datapool Information** 打开 Configure Datapool in Script 对话框；

3） 采用缺省配置，或作适当的改变。需要帮助击对话框顶部的?，再点击需帮助的条目；

4） 修改完后点击 **Save**；

5） 点击 **Edit Existing Data**；

6） 在 Edit Datapool 对话框中，修改 datapool 的值；

7） 编辑完后，击 **Save**，然后击 **Close**。

5、 编辑 Datapool 配置

1） 点击菜单 **File → Open → Script** 打开脚本；

2） 点击菜单 **Edit → Datapool Information** 打开 Configure Datapool in Script 对话框；

3） 在 Configure Datapool in Script 对话框中修改字段和列；

4） 修改完后点击 **Save**；

5） 做以下任一操作：

- 点击 **Create** 定义组成新的 datapool；
- 点击 **Edit Specification** 修改已有 datapool 的列定义；
- 点击 **Edit Existing Data** 修改已有 datapool 的值；
- 点击 **Close**。

6、 设置 Datapool 指针

1） 击菜单 **File → Open → Script** 打开脚本；

2） 点击菜单 **Edit → Datapool Information** 打开 Configure Datapool in Script 对话框；

3） 选中 **Persistent** 复选框，将 **Access Order** 设置为 **Sequential** 或 **Shuffle**；

4） 在 **Row Number** 指定在下次测试时首次要访问的 datapool 行；

5） 点击 **Set Cursor**。

7、 产生及找回唯一值

至少指定一列唯一数据

生成足够的 datapool 行

不能隐藏指针

使用有序或混乱的访问顺序

测试时不能指针

（六）、编辑 GUI 脚本

在 GUI 脚本中增加行为

在已有的 GUI 脚本中增加用户行为而不覆盖已有代码：

1、 打开一个已有的脚本；

2、 如果处于 Debug 状态，停止 Debug；

3、 将光标移至需要加入行为的位置，确认当前的应用程序与光标位置的应用程序状态一致；

4、 单击 Standard 工具栏的 Insert At Cursor 按钮；

5、 继续用户行为的记录。

在 GUI 脚本中增加特写

在已有的 GUI 脚本中增加用户行为而不覆盖已有代码：

1、 开一个已有的脚本；

2、 如果处于 Debug 状态，停止 Debug；

3、 将光标移至需要加入特写的位置，确认当前的应用程序与光标位置的应用程序状态一致；

4、 以下两种操作需要任选其一：

- 不进入记录模式增加特写，单击 Standard 工具栏上的 Display GUI Insert Toolbar 按钮，Robot 窗口依然开启；
- 进入记录模式增加特写，单击 Standard 工具栏上 Insert Recording 按钮，单击 GUI Record 工具栏的 Display GUI Insert Toolbar 按钮。 To start recording and add the feature

5、 单击 GUI Insert 工具栏上合适的按钮，File Comparison、File Existence、Module Existence 和 Delay 特写没有出现在 GUI Insert 工具栏中，要增加这些特写，单击 GUI Record 工具栏上 Open Robot Window 按钮，在 Insert 菜单下选择合适的菜单项；

6、 继续增加特写。

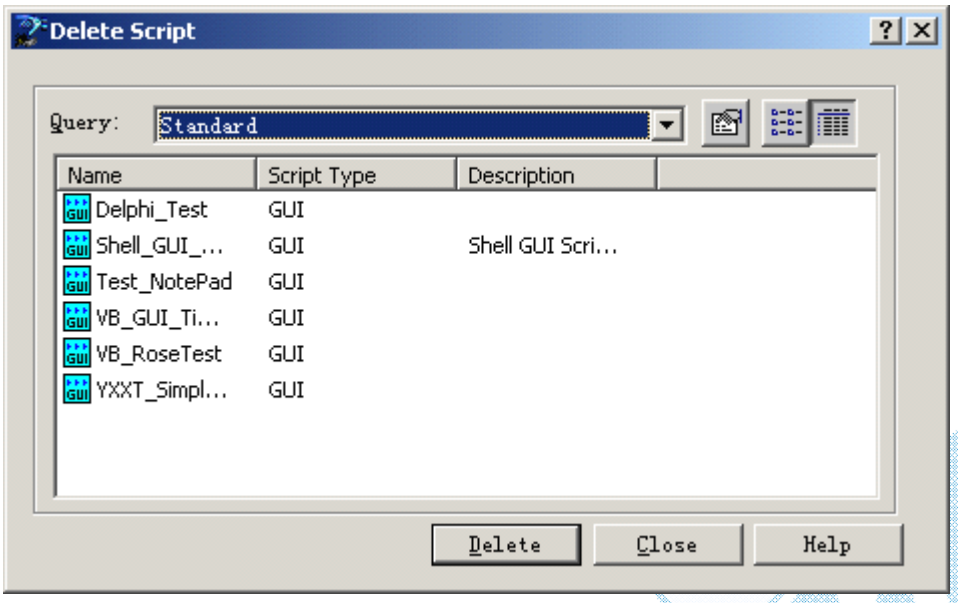
删除 GUI 脚本

1、 单击 File 菜单下 Delete 菜单项；

2、 从列表中选中一个或者更多的脚本，要改变脚本列表，从 Query 列表中选取不同的项目；

3、 点击删除按钮；

4、 关闭对话框；



从项目中删除 GUI 脚本同时删除了对应的脚本文件（.rec）、可执行文件（.sbx）、查证点和底层脚本。

（七）、编译 GUI 脚本

编译脚本和库文件

回放或者调试 GUI 脚本时，一旦发生改变，Robot 自动编译脚本。也可以手工编译 SQA Basic 库文件。

编译对象	操作	备注
当前脚本或者库文件	File 菜单下 Compile 子菜单	
当前项目的所有脚本和库文件	File 菜单下 Compile All 子菜单	改变影响所有 SQA Basic 文件的全局定义。

编译过程中，输出窗口显示脚本和库文件的编译结果以及错误消息。

批编译脚本和库文件

批编译步骤：

1、 单击 File 菜单下 Batch Compile 菜单项；

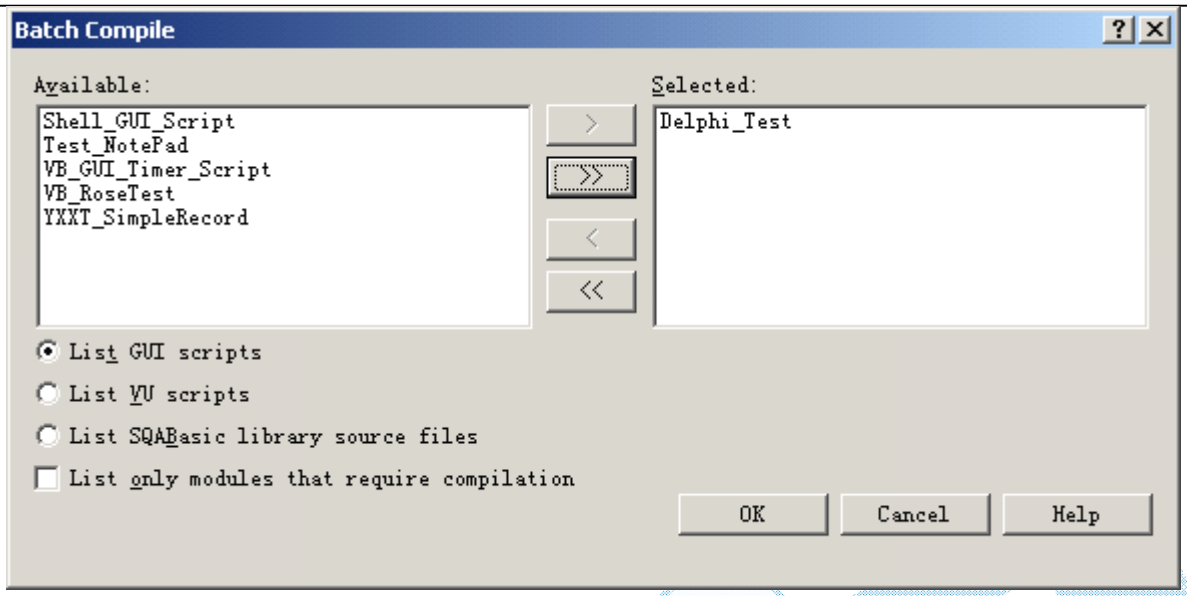
2、 在 GUI 脚本、VU 脚本或者 SQA Basic 库文件中选择一个，相应类型的脚本列表显示在 Available 列表中；

3、 可选操作：选中 List only modules that require compilation 则只显示需要编译的文件；

4、 选中需要编译的文件将其加入 Selected 列表中；

5、 确认编译选中的文件。

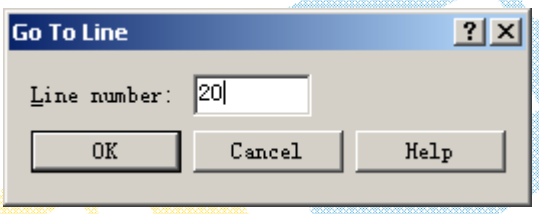
编译过程中，输出窗口显示脚本和库文件的编译结果以及错误消息。



定位编译错误

需要在 Script 窗口中定位编译错误，需要做以下操作之一：

- 在 Build 页面中双击错误，Robot 自动定位错误；
- 单击 Edit 菜单下 Next Error 或者 Previous Error 菜单项，Robot 自动查找错误；
- 单击 Edit 菜单下 Go to Line 菜单项，输入行号并确定，Robot 移动光标到该行首。



（八）、调试 GUI 脚本

调试前必须打开一个 GUI 脚本，若上次运行后，脚本被修改，则在调试前，Robot 会自动编译该脚本。

调试 GUI 脚本

1、 点击菜单“**File → Open → Script...**”打开脚本；
2、 点击 **Debug** 菜单命令或快捷按扭。

设置和清除断点

1、 点击菜单“**File → Open → Script...**”打开脚本；
2、 将光标移到要新增或清除断点的行；
3、 点击一次插入一个闪烁的光标；
4、 点击菜单“**Debug → Set or Clear Breakpoint**”设置或清除断点；

5、 若设置了断点，点击菜单“**Debug → Go**”开始调试。

执行选中行

没有设置断点时，在脚本中选中行处停止执行，需要
1、 点击菜单“**File → Open → Script...**”打开脚本；
2、 将指针放在要停止执行的行处；
3、 点击一次插入一个闪烁的光标；
4、 点击菜单“**Debug → Go Until Cursor**”开始调试。

在 Animation 模式下运行

1、 点击菜单“**File → Open → Script...**”打开脚本；
2、 移动及更改 Robot 窗口的大小以便不覆盖测试的应用程序，可以看到脚本窗口；
3、 点击菜单“**Debug → Animate.**”

（九）、回放 GUI 脚本

回放前恢复测试环境

设置 GUI 回放选项

1、用以下任一方法打开 GUI　回放选项对话框；
- 回放前，点击菜单“**Tools → GUI Playback Options**.”
- 点击快捷栏上的“Playback Script”按扭，在回放对话框中点击 **Options** 按扭；

2、一页上设置选项；可通过点击对话框右上角的“ ？ ”后再点击项目来获得该项目的详细帮助信息；
3、 点击 **ok** 按扭。

回放 GUI 脚本

1、 回放前恢复测试环境；
2、 设置回放选项；
3、 点击工具栏上的 **Playback Script** 按扭；
4、 输入名称或从列表中选择名称；
5、 点击 **Options** 按扭改变回放选项，完成后击 **ok** 按扭；
6、 击 **ok** 继续；
7、 要出现 Specify Log Information 对话框对话框，需做

a、 从列表中选择一种 Build；（点击右边的 **Build Button** 按扭创建一个新的 Build）；
b、 从列表中选择一个日志文件夹；（点击右边的 **Log Folder** 按扭创建一个新的日志文件夹）；
c、 接受缺省的日志文件名（与脚本文件相同）或输入一个新的名称；
d、 击 **ok** 按扭.

8、 若出现了提示询问你是否覆盖日志，做以下任一操作：
- 击 **Yes** 覆盖日志；
- 击 **No** 返回 Specify Log Information 对话框,更改 Build、日志文件夹及 And/or 日志信息；
- 击 **Cancel** 取消回放；

在 LogViewer 中查看结果

回放结束后可以用 TestManger log 查看回放结果，包括查证点失败、程序失败、异常中断及附加的回放信息。控制日志信息及显示日志，需要在 GUI Playback Options 对话框的 Log 页中设置选项：
- 选择 **Output playback results to log** 更新项目的回放结果；
- 选择 **View log after playback** 回放后自动打开日志文件，若未选择，回放后可通过点击菜单“**Tools → Rational Test → Rational TestManager**”.打开日志文件

在 Comparators 中查看查证点结果

在 TestManger log 中打开 Comparator：在日志文件的 Event Type 栏，选择一个查证点，点击“**View → Verification Point**.”

结束回放

人为结束回放：按功能键 **F11.**

（十）、工具条操作

Robot 工具条

Standard——记录、回放、打开、保存、编辑、编译、调试及显示帮助信息；
Tools——启动其他的 Rational 测试产品和组件；
GUI Record——暂停或停止记录，打开 Robot 窗口，显示 GUI 插入工具栏；
GUI Playback——回放及调试 GUI 脚本；
GUI Insert——往 GUI 脚本中插入特写；
Session Record——停止记录会话（VU 脚本），打开 Robot 窗口，分离脚本，显示会话插入工具栏；
Session Insert——往 VU 脚本中插入特写；

工具条按扭查看信息

有几种方式查看工具条按扭信息及相应的菜单命令：

- 查看按扭名称：指到按扭并停顿会看见在黄色的工具条上提示按扭名称；
- 查看简述：指到按扭或菜单命令在状态条上看到一条简短的描述；
- 查看详细信息：通过以下任一操作可以查看按扭或菜单命令的详细信息
  - 指向按扭或菜单命令，按 F1 键；
  - 在标准工具条的右边，击 **Help Pointer icon**  **Help Pointer** 按扭，指向按扭或菜单并击鼠标；

显示和隐藏工具条

击菜单 **View** → **Toolbars**，显示或隐藏工具条的名称

锁定和浮动工具条

锁定浮动工具条：

1、 指向工具条的标题栏或指向按扭外的的任意位置；
2、 拖动工具条到标题栏、菜单栏或一个存在的工具条；

使浮动工具条锁定：

8、 指向工具条，但不能在任何按扭上；
9、 拖动工具条到另一位置；

设置工具条选项

1、 击菜单 **View** → **Toolbars** → **Customize** 或用鼠标右击工具条，点击 **Customize**；
2、 在 **Toolbars** 页，选中或清除适当的复选框：
   Show ToolTips——指向按扭并暂停时显示工具条提示；
   Cool Look——改变工具条按扭的外观，让其没有边框。不会改变按扭的功能；
   Large Buttons——改变工具条按扭的大小；
3、 击 **ok**。

增加、删除、移动工具条按扭

1、 击菜单 **View** → **Toolbars** → **Customize** 或用鼠标右击工具条，点击 **Customize**；
2、 击 **Commands** 页；
3、 增加按扭：从 **Categories** 列中点击菜单名；
4、 删除按扭：拖动按扭到工具条外的任意位置；
5、 移动按扭：拖动按扭到工具条上的另一位置；
6、 击 **ok**；

创建自己的工具条

1、击菜单 **View → Toolbars → Customize** 或用鼠标右击工具条，点击 **Customize**；
2、击 **Toolbars** 页；
3、击 **New;**
4、 输入新建工具条的名称，击 **ok**；
5、 击 **Commands** 页；
6、 从 **Categories** 中点击菜单名；
7、 点击按扭看其描述。拖动按扭到新建的工具条；
8、 重复步骤 6、7；
9、 击 **ok**;

重置及删除工具条

1、 击菜单 **View → Toolbars → Customize** 或用鼠标右击工具条，点击 **Customize**；
2、 在 **Toolbars** 页做以下任一操作：
- 重置缺省工具条，恢复其最初配置，在列表中突出该工具条，击 **Reset**;
- 删除定制的工具条，在列表中突出该工具条，击 **Delete**。

3、击 ok。

二、VU 脚本

（一）、设置以及预定义

设置记录选项

（二）、记录 VU 脚本

1.在工具栏击'record session' 

按钮。

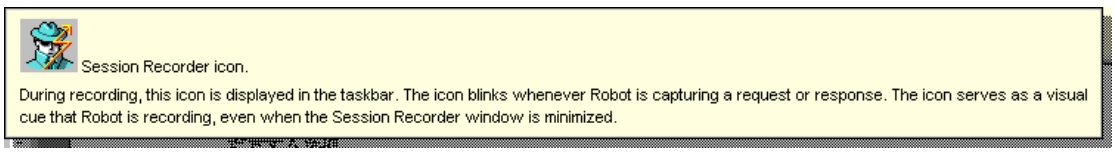
2.输入 session 名称（不超过 40 字节），或接受默认名。当完成录入脚本将指定脚本名称。如果没有 session recording 选择权，可以击权限按钮，下一步就进行权限设置。

3.在 session recording 界面按确定，弹出 session 名称对话框，接着进行：

--robot 最小化（默认行为）

--出现不固定的 session record 工具栏（默认行为）。应用工具栏停止录入，重现 robot,展开一个脚本，可在此脚本加入内容。

-- session record 图标



出现在工具栏上。

--如果客户应用程序在运行， session record 窗口会出现正常或最小化状态。在录入期间，窗口会显现统计客户端或服务器进行的每一步骤。

--如果客户端应用程序停止运行，则 Start Application 对话框（此对话框只出现在执行 API 录入时，如果是执行网络或代理录入则应在会话记录选项对话框的 GENERAL 标号下选择 **Prompt for application name on start recording** ）出现在 session record 窗口出现之前。

4.若最初的应用程序对话框出现，并提供以下信息，按 ok 键：

--数据库应用程序提供执行路径.

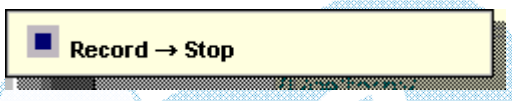
--一些构成工作目录(如 DLLS)的客户应用程序运行时间.

--一些你想客户应用程序通过的建议.

5.完成一个或多个事务的记录.

6.插入主要内容如通过浮动工具栏的 Session Insert 或 robot insert 菜单插入定时器和板块.

7.完成处理事务的录入后,关闭客户端应用程序.

8.在 Session Record 浮动工具栏上击 Stop Recording 按钮. 

9.用录入的脚本对话框可为刚完成的录入脚本选择脚本名称或默认名称.

10.击 ok 键

一个产生脚本的对话框出现,它反映了脚本自动生成过程,一段时间后,脚本生成结束成功出现在状态栏内.ok 键被击活.

在脚本生成期间,可以看到:

--缺少口令的对话框

--手动过滤对话框

11.在生成脚本对话框击 ok 键,已录入的脚本会出现在 robot 的窗口里.

（三）、回放 VU 脚本

播放 VU 脚本可用下列任意方法：

--可用运行 TestManager suite 连同其他脚本播放 VU 脚本。因信息在 TestManager suite 上，可查看 Rational TestManager 帮助。

--录入或编辑 VU 脚本之后，可自行播放，正好可测试自己所做的记录和编辑。

从 ROBOT 启动脚本播放：

1.在 ROBOT，按 File --Playback.

2.选择要播放的 VU 脚本名称。

3.击确定键。

TestManager 出现，准备播放选择的脚本。

4.在 TestManager，按 Run--Suite.

5.在这打开的对话框中击确定键。

（四）、重录 VU 脚本

覆盖记录一个 session 会影响这个 session 中的所有脚本。

如果单想重新记录某一个脚本，仅仅选择 ROBOT（在分离脚本或停止记录的对话框中）提示你正在记录脚本的脚本名称。

同样，若你在 TestManage 中设计一个脚本，它的名称会出现在现存脚本列表中，当在 ROBOT 中记录脚本可从

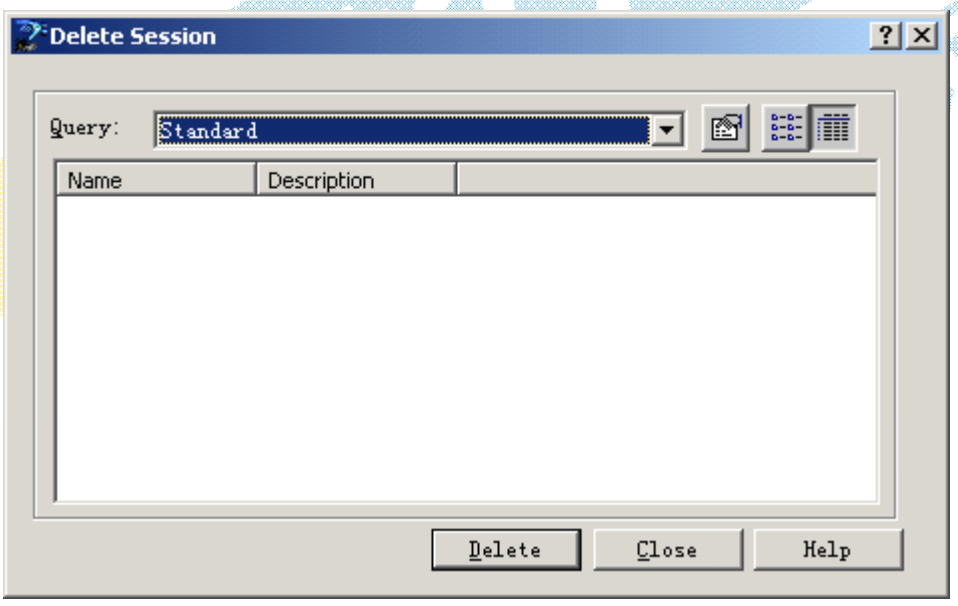
中选择脚本名。
想知道编写或现存的脚本的结果，见下列：
1.脚本已在 TestManage 设计了，还未记录：这个脚本的道具已用在新的脚本中，这个脚本在记录之前不能被 robot 迅速确认，因为此脚本是空的。
2.现存脚本是会话的一部分：能被 robot 迅速确认想编写的脚本。
---击 no 是选择或定义其他脚本名。
---击 yes 是编写此脚本，最初脚本的道具已用在新的脚本，同样，这个脚本从最初的会话移到另一个会话中。
3. 现存脚本不是会话的一部分：最初的脚本不能被 robot 迅速确认，此脚本的道具已用在新的脚本

（五）、复制 VU 脚本

1.按 File --Open --Script.
2.选择要复制的脚本名，按 ok 键。
3.按 File --Save As.
4.定义新的脚本名，按 ok 键。
新的脚本不能保存最初脚本的道具，并不能关联其它会话。

（六）、删除 VU 脚本

删除 VU 脚本时删除了.s 文件及其属性，但是不删除关联的会话文件（.wch）。



（七）、编译 VU 脚本

回放 VU 脚本时，如果脚本改变，则自动编译。手工编译方式操作与 GUI 脚本编译操作相同。批编译 VU 脚本操作方式与批编译 GUI 脚本操作一致。

（八）、查询会话中的脚本列表

脚本如何包含在一个会话中
查询一个会话中的脚本列表：

1. 在 TestManager，按 View --Test Asset Workspace.
2. 双击 Session Queries.的 ALL
3. 双击想查看脚本所在的会话名称。

按 Contained Scripts.

（九）、用会话生成脚本

一个会话再生成脚本
因多种原因你想再生成一个会话脚本，例如：你想放弃最初编辑的脚本或想再生脚本用于不同的 script generation options.
当在一个会话中再生成脚本，最初的脚本被再新脚本覆盖，最初脚本的道具被应用在再生脚本中
任何时候，可用下列方法再生脚本：

1. 在 ROBOT，按 Tools ---Regenerate Test Scripts from Session.
2. 选择要再生脚本的会话名。
3. 按确定键
   生成脚本对话框出现。它反映自动再生脚本的进程。一段时间后，脚本再生结束，成功完成信息出现在状态栏中。Ok 键被击活。
   注意：当从会话对话框的再生测试脚本击确定键，在会话中的这个脚本被毁灭。如果在再生脚本之前，在再生脚本对话框击 CANCEL，Robot 将生成一些空脚本。
4. 按 OK 确认脚本再生运行完成。

（十）、将 VU 脚本融入会话

一个会话融入多个脚本

1. 在会话录入期间，你想结束一个脚本并开始新的脚本，击 

Split Script 对话框出现。
2. 结束时定义或选择脚本名称或接受默认名。可放弃你的要求直到开始录入当前脚本，击 Ignore just-recorded information。此动作只影响当前脚本。对同个会话先前录入的脚本无效。
3. 按 ok
4. 多次重复以上步骤。
5. 按 

后结束录入会话，定义或选择脚本名称或接受默认名。

（十一）、手工 VU 脚本编码

迄今为止，最快最简单的方法生成 vu 脚本是让 robot 记录客户要求并自动生成脚本。

也可以打开空的 vu 脚本，并给他添加代码。如：你是手工编码，或从其他脚本复制编码。
步骤：
1. 在 robot 击 File ---New ---Script.
2. 定义或选择脚本名称或接受默认名
3. 击 VU
4. 击确定键，Robot 生成空脚本如下：

```
#include <VU.h>
{
}
```

5.为 vu 脚本添加代码。

会话记录选项对话框---生成标志
这个标志能设置多种脚本生成选项
打开：按 Tools--Session Record Options. 按 the Generator 标志
能做：
A 在脚本里自动增加 datapool 命令与 DATAPOOL_CONFIG 声明（击 Tools ---Session Record Options. 按 Generator，选择 Use datapools.按确定键。）
B 分派 a command ID 前缀（不能用这个选项为一个会话记录定义多个 command ID 前缀，停止记录之前分派的最后 command ID 前缀用于 robot 生成的脚本。）
C 插入返回行到脚本。选择 Display recorded rows 任意值。
----**None.**不插入任何返回行到脚本。
---- **First**.从服务器插入刚开始返回数据指定数目的字节或行。用邻近域指定字节或行的数目。
---**last** 从重新设置记录的结尾插入行的数目。
---**ALL** 插入所有返回数目到脚本里。
D 受 SQL statement 或网络服务器影响插入字节或行的数目。按 Tools--Session Record Options. 按 the Generator 标志，选择 Verify playback row counts.
E **插入 SQL statement 返回的代码。选择 Verify playback return codes.**
F 捆绑脚本表达输出参数。选择 Bind output parameters to VU variables 复选框。当选择后，它将生成 vu 脚本所需包含的输出参数返回变量。这个申请只支持输出捆绑参数。
G 控制 vu 脚本回放速度。
--- per command . vu 脚本回放速度基于每个 emulation command.所需记录和处理的实际时间。
---per scrip. vu 脚本回放速度基于所有 emulation command. .所需记录和处理的平均时间。所有 emulation command 用相同的平均延迟思考时间。
---none. vu 脚本回放速度基于各自脚本速度，默认值为 5 秒。
H 设置 CPU/用户端，选择 CPU / User threshold (ms)复选框，用毫秒指定 CPU/用户端
I. 设置最大思考时间，用毫秒指定最大思考时间。

三、VB 脚本

删除 VB 脚本

从项目中删除 VB 脚本同时删除了对应的类模块文件（.cls），项目文件（.vbp），资源脚本文件（.rc），资源

文件（.res）和 Windows 动态链接库文件（.dll）。

四、SQA Basic

（一）、定制 SQA Basic 脚本

定制 SQA Basic 脚本的途径

- 直接在脚本文件中加入自定义 SQA Basic 子程序、函数或者包含库文件，其它文件（脚本或者其它库文件）可调用库文件的自定义子程序；
- 用 SQA Basic 头文件申明自定义子程序、常量和变量，SQA Basic 头文件申明的项可被其它脚本和库文件所用；
- 通过脚本模版包含每个新脚本和.rec 库文件都需要的信息。

库文件（Library Source Files）

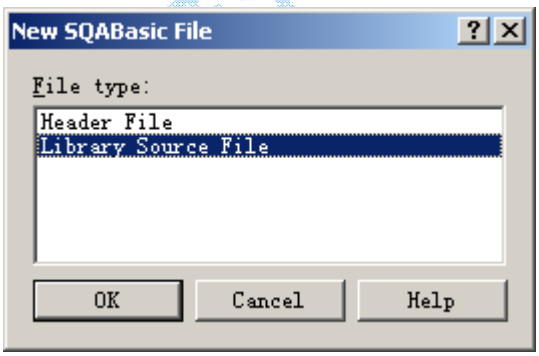
- .sbl 文件：项目范围的库文件，但是不支持查证点，存储于项目的 SQABas32 文件夹下；
- .rec 文件：项目范围的库文件，支持查证点，存储于项目的 Script 文件夹下。

.rec 文件常被用作 GUI 脚本。

库文件用于保存多个脚本需要访问的自定义子程序。如果子程序只有一个脚本访问，建议将其放入该脚本中。也可以在 SQA Basic 脚本和库文件中调用动态链接库中的子程序。然而，Robot 不能像.sbl 和.rec 创建或者编辑动态链接库。

创建、编辑.sbl 库文件

创建.sbl 文件：单击 File 菜单下 New 子菜单的 SQA Basic File 菜单项，在下图对话框中选择 Library Source File，确认即可。

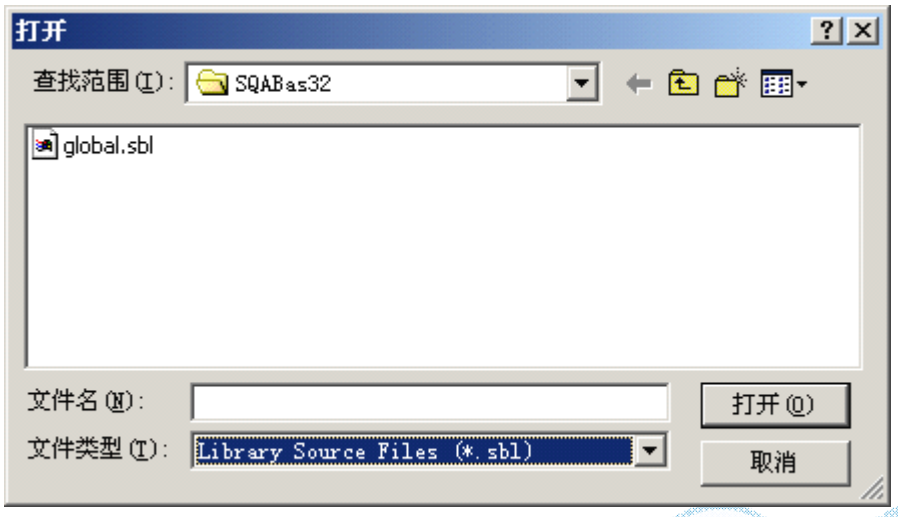


库文件不能与调用它的脚本文件同名。

为了方便，Robot 提供了空白的库文件（Global.sbl），可以将子定义子程序加入到该文件中，或者创建新的库。

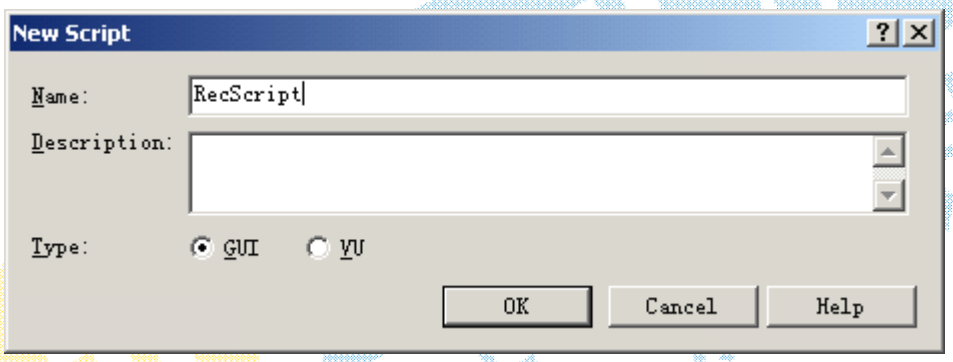
编辑.sbl 库文件：单击 File 菜单下 Open 子菜单的 SQA Basic File 菜单项，在下图的打开文件对话框中，选

择 Library Source Files (*.sbl)，选中一个要编辑的文件，单击“打开(O) ”。

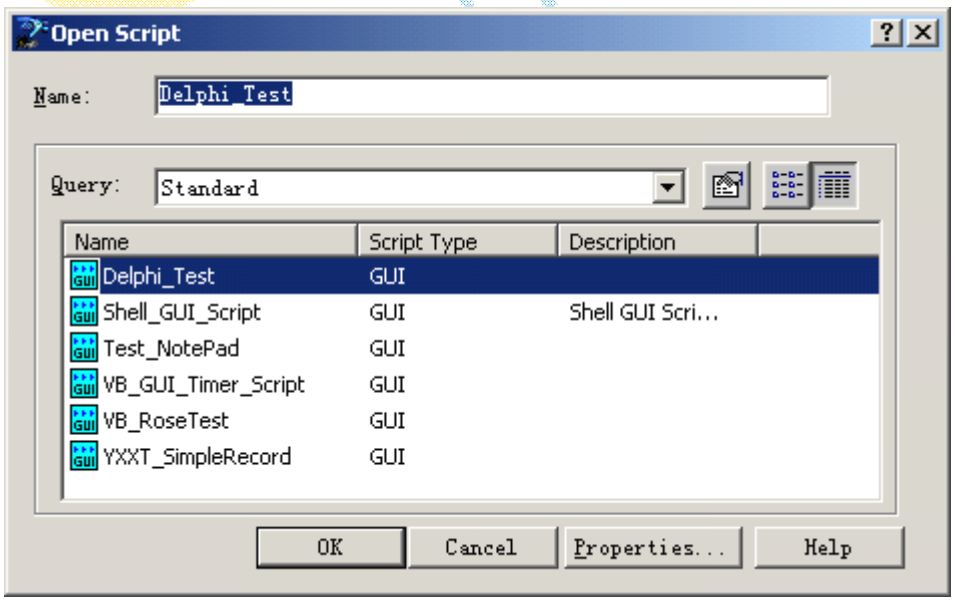


创建、编辑.rec 库文件

创建.rec 库文件：单击 File 菜单下 New 子菜单的 Script 菜单项，在下图所示的对话框中输入文件名称，选中 GUI，确认。



编辑.rec 库文件：单击 File 菜单下 Open 子菜单的 Script 菜单项，在下图所示的对话框中选定文件名称，确定。



在 Global.sbl 中加入子程序

用编辑.sbl 库文件的方法打开 Global.sbl 文件，编辑即可。

使用库文件

在运行时使用 SQA Basic 库文件。

1、 在库文件中增加自定义子程序；

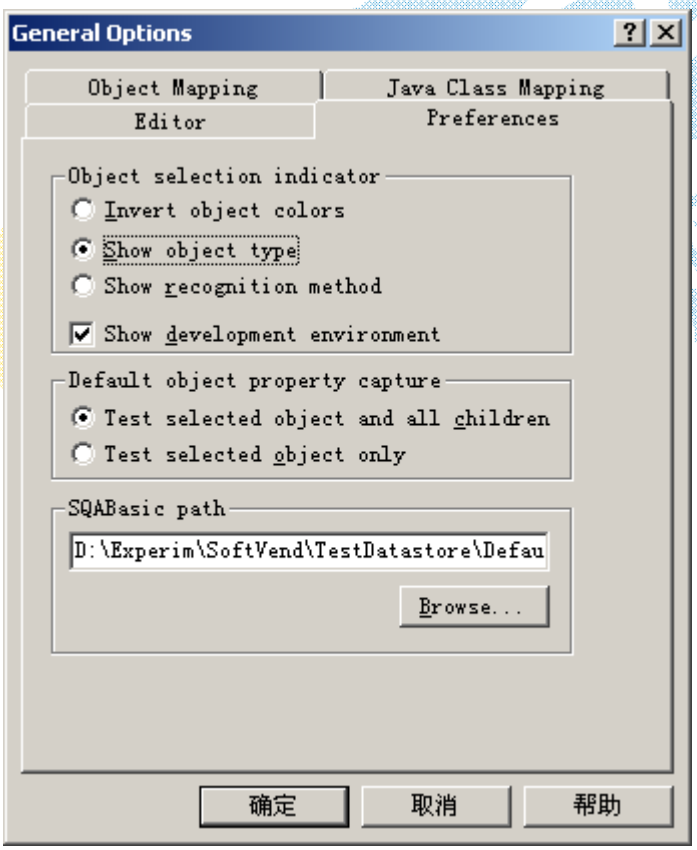
2、 编译文件：两种 SQA Basic 库文件（.sbl 库文件和.rec 库文件）在运行时被编译成.sbx 文件；

3、 在 SQA Basic 头文件或者脚本或者库文件中申明需要调用的自定义子程序所在的文件。SQA Basic 支持.sbx 库文件和.dll 库文件。

SQA Basic 头文件

在 SQA Basic 头文件中可以申明自定义子程序、常量和变量。缺省时，SQA Basic 文件存储于 SQABas32 文件夹，可以在 Tools 菜单的 General Options 菜单项中设置，如下图所示。在 Preferences 页面，使用 Browse 按钮指定新文件夹。如果文件不在此文件夹下，则在 SQABas32 目录下。

SQA Basic 头文件的扩展名是.sbh。

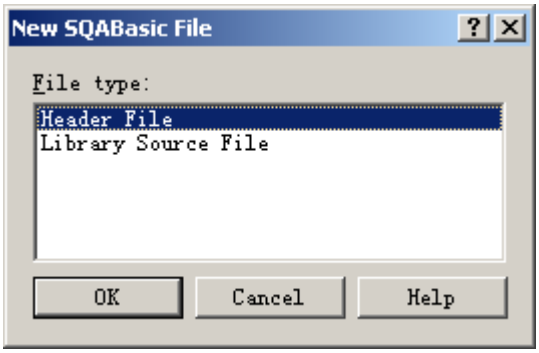


创建、编辑仓库（Repository）范围的头文件

创建一个项目中的仓库均可访问的头文件：

1、 单击 File 菜单下 New 子菜单的 SQA Basic File 菜单项；

2、 点击 Header File，确定。



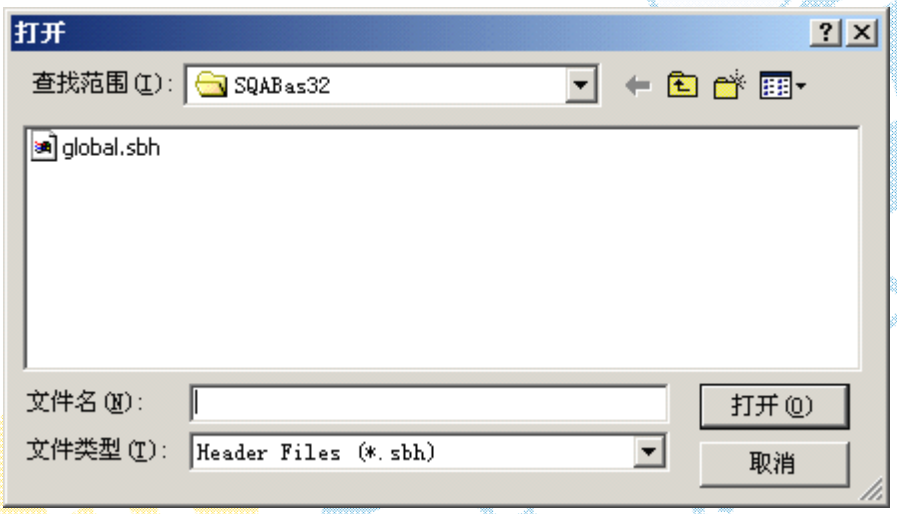
为了方便，Robot 有一个 Global.sbh 头文件。

编辑一个已有的头文件：

1、 单击 File 菜单下 Open 子菜单的 SQA Basic File 菜单项；

2、 选择文件类型为 Header Files(*.sbh)；

3、 选中要编辑的文件，打开即可。



创建、编辑项目（Project）头文件

创建一个新的项目头文件：单击 File 菜单下 New 子菜单的 Project Header File 菜单项。

编辑一个已有项目头文件：单击 File 菜单下 Open 子菜单的 Project Header File 菜单项，选中要编辑的文件确定即可。在 Global.sbh 文件中加入申明的方法一样。

使用头文件

应该先保存头文件，再编译引用了头文件的脚本或者库文件。

头文件不能单独编译。

使用模板（Template）文件

Robot 提供了 TestProc.tpl 模板文件，用于在新的 GUI 脚本中自动增加注释和语句。编辑模板文件步骤如下：

1、 单击 File 菜单下 Open 子菜单的 SQA Basic File 菜单项；
2、 设置文件类型为 Template Files(*.tpl)；
3、 选中 TestProc.tpl 文件，打开；
4、 输入相应的语句并且保存。

五、测试应用程序

（一）、测试 Delphi 应用程序

Rational Robot 全面支持 Delphi 应用程序。Robot 支持用 Delphi 3.0、4.0 和 5.0 在 Windows NT、Windows 95、Windows 98 和 Windows 2000 平台下编写的程序。

用 Robot 测试第三方组件，包括：

- VCL 组件；
- Win32 控件；
- ActiveX 控件；
- Data-aware 控件；
- 不可见控件；
- 支持 Internet 的控件；
- 可视的继承窗体。

要测试 Delphi 应用程序，必须运行 Delphi Enabler 并且安装 Rational Object Testing Library for Delphi。

安装 Rational Object Testing Library for Delphi 步骤

1、 装 Rational Object Testing Library for Delphi。
2、 安装 Rational Test Delphi Enabler。
3、 运行 Enabler，在工程中产生一行代码（SQA Server），然后在 Delphi 中重新编译工程即可。

安装 Rational Object Testing Library for Delphi 步骤

1、 Rational Suite CD-ROM 放入光驱；
2、 进入运行对话框；
3、 键入“（光驱盘符）:\Setup.exe”，开始 Rational Setup Wizard 时点击 OK；
4、 Rational Test Enablers 列表出现时，选择 Rational Test Delphi Enabler；
5、 根据屏幕指示完成安装。

加入 Rational Object Testing Library

1、 在开始菜单下启动 Borland Delphi 菜单下的 Rational Test Delphi Enabler；
2、 在 Delphi Project 下点击 Browse，选中需要 Robot 测试的项目，点击 OK 按钮；
3、 根据计算机上已经安装的 Delphi 选择正确的版本；
4、 选择增加 Rational Object Testing Library；
5、 可选操作：如果在工程文件转换之前不希望备份，不选中 Backup Project File；
6、 可选操作：如果在转换完成之后不希望启动 Delphi，不选中 Launch Delphi After Conversion；
7、 点击转换按钮，开始转换（在 uses 中加入“SQASrvr”）；
8、 点击 Enabler 的 Close 按钮，如果探出一个信息对话框，单击 Yes 来重新载入工程；
9、 在 Delphi 中，重新编译工程。

注意：Rational Object Testing Library 不可见，也不可插入，并且没有许可证限制。因此，可在应用程序分布时保留它。但是，若想从工程中删除它，依据以下步骤删除 Rational Object Test Library。

删除 Rational Object Test Library

1、 在开始菜单下启动 Borland Delphi 菜单下的 Rational Test Delphi Enabler；
2、 在 Delphi Project 下点击 Browse，选中需要删除 Robot 测试的项目，点击 OK 按钮；
3、 根据计算机上已经安装的 Delphi 选择正确的版本；
4、 选择删除 Rational Object Testing Library；
5、 可选操作：如果在工程文件转换之前不希望备份，不选中 Backup Project File；
6、 可选操作：如果在转换完成之后不希望启动 Delphi，不选中 Launch Delphi After Conversion；
7、 点击转换按钮，开始转换（在 uses 中删除“SQASrvr”）；
8、 点击 Enabler 的 Close 按钮，如果探出一个信息对话框，单击 Yes 来重新载入工程；
9、 在 Delphi 中，重新编译工程。

测试 Delphi 组件属性

测试 Delphi 组件属性有两种方法，可以通过方法测试 Delphi Object Inspector 中显示的所有属性

1、 对象属性查证点（Object Properties Verification Point）：用于在记录或者编辑脚本时测试对象属性，关于测试对象属性的指令，参看 Creating an Object Properties Verification Point；
2、 对象脚本命令（Object Scripting Commands）：用于在编辑脚本的同时测试可编程属性，相关命令见 SQA Basic 帮助。

（二）、测试 Visual Basic 应用程序

Robot 全面支持 Visual Basic 4.00 及更高版本编译的 32 位应用程序，支持对 Visual Basic 版本移植的应用程序，并且允许重用 Windows NT 4.00、Windows 2000、Windows 98 和 Windows 95 的脚本。Robot 使用对象测试技术检查用户不可见的数据和属性，可以完成以下工作：

- 识别所有的 Visual Basic 对象，包括带窗口对象（例如 Edit Box）和“画”在容器表单上的对象（例如 Label）；
- 获取程序中的对象名称（在 Visual Basic 源代码中写定），并且使用这些对象名称来识别对象；
- 使用对象属性查证点捕获 Visual Basic 对象属性；

- 使用对象属性查证点捕获潜在的 Visual Basic 数据控件；

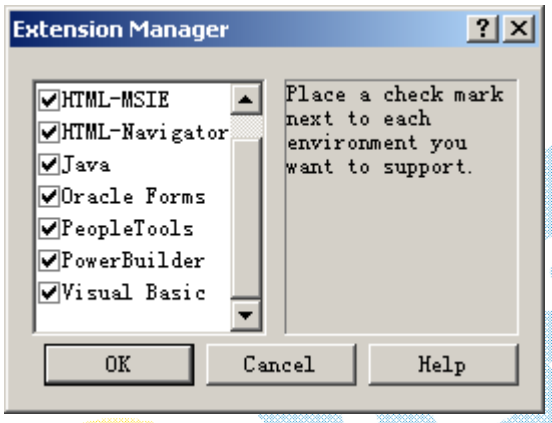
举例而言，假设在 Visual Basic form.上有一个标签（label）。如果在 Robot 录制期间点击标签，则标签的名称会出现在 Robot 脚本中。如果在标签上设置了对象属性查证点，标签的名称被捕获。Robot 定义的名称与 Visual Basic 属性窗口中显示的名称相同。

测试 Visual Basic 4.00 的应用程序，需要在 Visual Basic 表单上增加 Rational ActiveX Test Control。

加载 IDE Extensions

要测试 Visual Basic 应用程序，应该先验证在 Robot 中 Visual Basic Extension 是否已经加载。

1、 启动 Robot；
2、 选择工具菜单下 Extension Manager；
3、 确认 Visual Basic 是否被选中；
4、 为了提高 Robot 性能，可以禁止不需要支持的环境；
5、 退出 Robot。



重新进入 Robot 之后，只加载被选中的环境。

关于 Visual Basic 4.00 的支持，要做特殊处理，详情请见帮助。

第三章 参考

（一）查证点

Alphanumeric —捕获及比较字母或数字的值；
Clipboard — 捕获及比较复制到剪贴板的字母数字的数据；
File Comparison —比较两个文件的内容；
File Existence —检查一个指定的文件是否存在；
Menu — 捕获及比较菜单的文本、快捷键及状态，能够捕捉到第五级子菜单；
Module Existence —检查连接到指定上下文（过程）或内存的任意地方的模块是否存在；
Object Data —捕获及比较目标数据；
Object Properties —捕获及比较对象的属性；
Region Image —捕获及比较位图的屏幕区域；
Web Site Compare —捕获 Web 站点的基线，并及时与另一 Web 站点比较；
Web Site Scan —检查每次修改后 Web 站点的内容，确保这些变化不会有差错；

Window Existence —检查继续回放前指定的窗口是否显示；
Window Image — 捕获及比较位图（菜单、标题栏和未捕获的边框）窗口的客户区域。

（二）查证方法

Case-Sensitive —校验记录时捕获的文本与回放时捕获的是否完全匹配；
Case-Insensitive —校验记录时捕获的文本与回放时捕获的是否匹配（不区分大小写）；
Find Sub String Case-Sensitive —核实记录时捕获的文本是否是回放时捕获的子串（区分大小写）
Find Sub String Case-Insensitive—核实记录时捕获的文本是否是回放时捕获的子串（不区分大小写）；
Numeric Equivalence —核实记录时的数据值与回放时是否相等；
Numeric Range —核实数字值的范围；
User-Defined/Apply a User-Defined DLL test function—将文本传给动态连接库中的函数以便运行定制的测试；
Verify that selected field is blank —校验选中的字段是否为空。

（三）鉴别方法

（四）标准数据类型

（五）Rational Robot 命令行选项

（六）Rational Robot 窗口

（七）菜单