Response File Editor
Technical Preview

Table of Contents

Table of Contents

Summary

The Response File Editor (RFE) is a command-line tool that you can use to create and edit response files for platforms that are supported by a package. This tool creates a scripted response file that uses a domain specific language (DSL), written in Groovy. Using Groovy provides more flexibility for administrators to manage their environments. You can then use the Response File Editor to run these scripted response files with IBM Installation Manager to manage packages.

Capabilities

The following capabilities are available in this technical preview:

Create or edit scripted response files

- Specify configurations for multiple platforms in one response file. For example, you can specify installation locations for a package for the Linux and Windows platforms.
- Create functions in the response file that are evaluated at run time. You can use one response file to handle different scenarios.
- It is not supported to edit the functions using the tool. To edit the functions, open the file and manually create or edit the function.

Generate Installation Manager response file based on a scripted response file

- Generate an Installation Manager response file based on a scripted response file.
- Run the Installation Manger response file on a machine.

Support the install and uninstall commands

- Support the install and uninstall commands in the scripted response file.

Support remote and local repository access

- Use data from repositories to generate a scripted response file.
- Support access to local or remote repositories.
- For remote repositories, use keyring files to connect to secure repositories.

Record a response file to install or uninstall multiple packages at the same time

Getting Started with the Response File Editor

Terminology

Many of the terms used for the Response File Editor are defined in the Installation Manager information center. For a list of Installation Manager information centers, see http://www-01.ibm.com/support/docview.wss?uid=swg27010911.

Terms that are specific to the Response File Editor:

- Context – A mode of operation that changes the available commands. Each of the contexts has different commands and behaviors. In the Response File Editor, there are three contexts:
  - base
  - create
  - edit
- Base context – The context that is entered when you start the Response File Editor. In the base context, you can create or edit scripted response files, run the scripted response file, and see the history of previous actions.
- Create context – You enter this context when you create a scripted response file. In the create context, you can add repositories, choose the packages to install or uninstall, and provide required information for the packages.
- Edit context – You enter the edit context when you edit a scripted response file. In the edit context, you can add repositories, choose the packages to install or uninstall, and provide required information for the packages.

Supported Platforms

The Response File Editor is currently supported on the same Microsoft® Windows® and Linux™ platforms that are supported by Installation Manager. To see a list of the supported Windows and Linux platforms, see the System Requirements for Installation Manager version 1.5.x: http://www-01.ibm.com/support/docview.wss?rs=0&uid=swg21498661

You can create or edit a response file for a different platform than the platform that you are currently on. To run a response file, you must be on the same platform.

For example, you create a response file for Linux by running the Response File Editor tool on Windows. To run the response file on Linux, you must run the Response File Editor tool on Linux.

The Response File Editor currently supports these packages:

- IBM® Rational® ClearCase® for Windows, versions 8.0.0.0 – 8.0.0.x
  - Does not support Rational ClearCase Remote Client WAN or Rational ClearCase Multisite

features.
- IBM Rational ClearCase for Linux x86, versions 8.0.0.0 – 8.0.0.x
  - Does not support Rational ClearCase Remote Client WAN or Rational ClearCase Multisite features.
- IBM Rational Team Concert™, versions 2.0.x - 3.0.x
- IBM Rational Application Developer, versions 8.0.0.0 - 8.0.1

Prerequisites

Before using the Response File Editor, you must have Installation Manager 1.5.x installed on your system. Also, you must set two environment variables before you start the Response File Editor:

- JRE_HOME – Set this variable to the location of a 32-bit Java Runtime Environment (JRE) that is installed on your system. You can use the JRE installed with Installation Manager that is located at *installation_manager_dir*\eclipse\jre*. NOTE: Enclose file paths that include spaces with double quotation marks.
- IM_TOOLS – Set this variable to the location of the Installation Manager tools directory that is located at *installation_manager_dir*\eclipse\tools. NOTE: Enclose file paths that include spaces with double quotation marks.

Navigating

The Response File Editor uses tab completion. You can press Tab at any point on the command line to see a list of valid commands and responses. If a long list of values are returned, you can enter the first few letters and then press Tab.

At the command line, default values are indicated by [ ]. You can press Enter to accept the default entry or enter a different value. For example, [`All`] indicates that the default value is `All`.

Starting the Response File Editor

1. Download the compressed file from the Installation Manager Enterprise DVD.
2. Extract the compressed files for the Response File Editor.
3. Open a command utility
4. Set the IM_TOOLS and JRE_HOME environment variables.

- JAVA_HOME: JRE installation directory
  You can use the JRE installed with Installation Manager or a different JRE. The JRE must be 32-bit.

  Windows: *install_dir*\Installation Manager\jre

Linux: *install_dir*/InstallationManager/jre

- IM_TOOLS: Installation Manager tools directory

  Windows: *install_dir*\Installation Manager\eclipse\tools

  Linux: *install_dir*/InstallationManager/eclipse/tools

5. To start the Response File Editor, run the command for your operating system:

   Windows: `rfe.bat`

   Linux: `./rfe.sh`

The Response File Editor opens in the base context.

In the command utility, you see this message:

```
Welcome to the Response File Editor console.  Type help
 for a list of commands.

>
```

Creating a Response File

1. To create a response file, enter the command: create *response_file*

You can provide a file name when entering the create command or provide a file name when you save the response file. In this example, the file name test.response is used. The response file is created in the same directory as the Response File Editor. To create the response file in a different location, use the full path with the file name.

After running the create command, you are in the create context. The command utility shows the prompt: create>

```
> create test.response

Response file will be written to test.response

Entering create context. Type help for a list of comman
ds.

create>
```

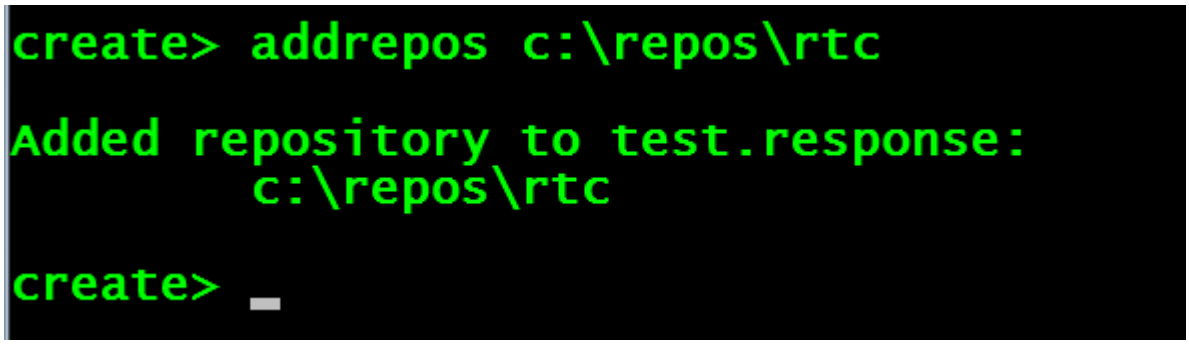
2. Add a repository by using the addrepos command: addrepos repository
NOTE: Enclose file paths that include spaces with double quotation marks.

Example:
Windows: addrepos C:\repository\packageA
Linux: addrepos /opt/repository/packageB

The Response File Editor uses repositories to gather information about the packages that are available to use when creating a response file. The information includes versions, supported platforms, and other information specific to a package.

To add more repositories, you must use the addrepos command for each repository. You can specify a repository by using FTP, HTTP, and HTTPS as well as authenticated repositories.

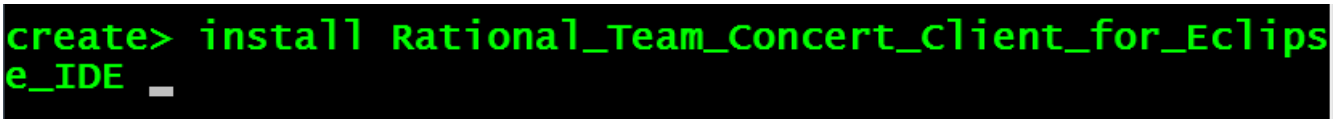
In this example, the repository is on the local machine.



3. Select the package to install by entering the `install` command. To see a list of available packages, press Tab after entering the `install` command. If a long list of values are returned, you can enter the first few letters and then press Tab. Select a package and press Enter. To select a package, either type the full name or use the Tab key to automatically complete the package name.

To uninstall a package, enter the `uninstall` command. Only the install and uninstall commands are supported. You cannot use the Response File editor to create a response file to modify, roll back or update a package.

In this example, the Rational Team Concert Client for Eclipse IDE package is selected.



4. If a package supports multiple platforms, you are prompted to select the platforms that you want to create a response file for. To see the list of platforms, press Tab at the prompt.

In this example, the Rational Team Concert package supports multiple platforms. The default value is `All`. To accept the default value of `All`, press Enter.

```
create> install Rational_Team_Concert_Client_for_Eclipse_IDE

Select platform [All]: _
```

5. You are prompted to enter advanced mode. The default value is `No`. If you do not enter advanced mode, default values are used in the response file. If you enter advance mode, you can customize these values for the response file. The values that you can customize are based on the package that you selected.

For some packages, an option might not have a default value. You are prompted to provide a value for these options even if you do not enter advanced mode.

In this example, the default values are used.

```
Do you want to enter advanced mode? [No]:

install command has been added successfully

create> _
```

6. Save the file by entering the `save` command.

If you did not provide a file name when using the create command, you must specify a file name:

`save` *`response_file`*

```
create> save

Save complete

create> _
```

7. To exit, first close the create content by using the close command. Then use the quit command to exit the Response File Editor.

```
create> close

> quit

Bye
```

As a result, you have created a response file.

Reading the response file

Open the response file that you created by using the Response File Editor. The file has five sections.

Script Version

The script version is listed at the top. For the technical preview, the value used is “1.0.0”.

```
rfe_script_version = '1.0.0'
```

Repositories

The repositories section lists the repositories that you provided by using the `addrepos` command. Each repository has a `addUrl` line.

```
repositories {
 addUrl 'c:\\repos\\rtc'
}
```

Environments

Environments are listed as env *env_name*. The environment sections are used to track platform specific information. For packages that support multiple platforms, an environment might consist of multiple platform entries. In this example, the response file lists the installation location and the shared resources directory:

```
env env1 {
 win32 {
  installLocation = '$
{defaultInstallRootLocation}\\IBM\\TeamConcert${beta}'
  installCommonLocation = '$
```

```
{defaultInstallRootLocation}\\IBM\\IBMIMShared${beta}'
 }
 linux {
  installLocation = '$
{defaultInstallRootLocation}/IBM/TeamConcert${beta}'
  installCommonLocation = '$
{defaultInstallRootLocation}/IBM/IBMIMShared${beta}'
 }
}
```

Packages

Packages are products to install or uninstall, and are listed as pkg *pkg_name*. The package sections contain information that identifies the package, the languages, and the features if the default values are not used. Other information specific to the package is included in the section. There is one package entry for each package that you selected to install or uninstall.

```
pkg RTCC4EIDE1 {
 id = 'com.ibm.team.install.rtc.client.eclipse'
 version = '3.0.1002.RTC-I20111208-1025-r3012'
 addProperty 'user.help.option', 'remote'
}
```

Command

The command section defines the values to use when you use the response file:

- `install` or uninstall action
- Package
- Environment

For multiple packages, you might have multiple commands that are defined.

Command syntax: *action package* using *environment*

Example command: `install RTCC4EIDE1 using env1`

where `install` is the action, `RTCC4EIDE1` is the package, and `env1` is the environment.

Running the Response File

Use the response file to install or uninstall a package. There are two ways to use the response file:

- From the command line
- From the base context after starting the Response File Editor

You must have Installation Manager and the Response File Editor installed on the computer that you are running the response file on.

If you are connecting to secured repositories, you must have the keyring file generated when you created the scripted response file available in the same directory as the scripted response file before running the response file.

Running From Command Line

1. Run the command to use the response file:

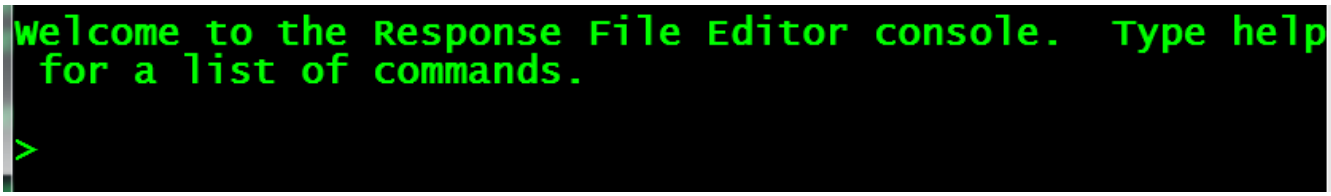
Windows: rfe.bat run *response_file* -opt *option1,option2,...,optionN*
Linux: ./rfe.sh run *response_file* -opt *option1,option2,...,optionN*

Running From The Tool

1. Start the Response File Editor by running the command for your operating system:

Windows: `rfe.bat`
Linux: `./rfe.sh`

The Response File Editor opens in the base context.



2. Enter the command to run the response file:

run *response_file* [-opt *options*]

Available options:

- `execute.response.verbose` – Execute the install or uninstall process in verbose mode. Use this option to maximizes the output to the console.
- `execute.response.skip.install` – Execute the install or uninstall process in skipInstall mode. Use this option to verify a response file. For information about the skipInstall mode, see the Installation Manager information center for your version of Installation Manager: http://www-

01.ibm.com/support/docview.wss?uid=swg27010911.

- `skip.execute.response` – Generates an Installation Manager XML response file, but does not execute the file.

If you do not use any options, the Response File Editor silently executes the response file based on the platform that you are running on.

Base Context Commands

Introduction

Information on the commands available in the base context of the Response File Editor.

Commands

The following commands are available in the base context.

Commands	Description
create [*response_file*]	Create a scripted response file. Use this command to enter the create context. You can add repositories, and install or uninstall packages in the create context. The *response_file* variable is the name of the response file that you are creating. If a path is not provided, the response file is saved to the Response File Editor directory. If you do not specify a *response_file* with the create command, you can provide the file name when the response file is saved.
edit *response_file*	Edit a scripted response file. Use this command to enter the edit context. You can add repositories, and install or uninstall packages in the edit context. NOTE: Manual changes that you make to a response file are removed when you save the response file in the Response File Editor.

	If you manually edit the response file, make a copy of the response file that contains the changes before you edit the file by using the Response File Editor. Manual changes to the response file are removed when you edit and save the response file in the edit context. After you save the response file in the Response File Editor, reapply the manual changes to the response file.
help [*command*]	List the available commands. When you use the help command with another command, a short description of the specified command is shown. When you use the help command without a command, descriptions of available commands are shown. Use the help command with the *command* option: • *command* : Specify a command to show information about the command.
history [-clear]	List the previously entered commands. Use the history command with the -clear option: • -clear: Clear the history for the current context.
quit	End the current session and exit the Response File Editor.
run *response_file* [-opt *option1, option2, option3*]	Run the scripted response file. This command generates an IBM Installation Manager XML response file then executes the XML response file by using the Installation Manager command line. For response files that contain information for multiple platforms, the platform information that is used is based on the platform that you are running on. For example, you can create one scripted response file that contains information for both the Windows and Linux platforms. When you run the

	scripted response file on Linux, the XML response file that is generated contains only the Linux information.
	
	For a response file that connects to a secured repository, a keyring file that matches the name of the scripted response file must be present in the same directory as the scripted response file. The Response File Editor will use that keyring file to connect to any secured repositories that have been specified. This keyring file will be created by the Response File Editor when you create the scripted response file.
	
	Use the run command with this options:
	
	• -opt: Additional commands to use when executing the scripted response file.
	  • execute.response.skip.install: Execute the install or uninstall process in skipInstall mode. Use this option to verify a response file. For information about the skipInstall mode, see the Installation Manager information center for your version of Installation Manager: http://www-01.ibm.com/support/docview.wss?uid=swg27010911.
	  • execute.response.verbose: Execute the install or uninstall process in verbose mode. Use this option to maximizes the output to the console.
	  • skip.execute.response: Generates an Installation Manager XML response file, but does not execute the response file.

Create & Edit Context Commands

Introduction

Information on the commands available in the create and edit contexts of the Response File Editor.

Commands

The following commands are available in the create and edit contexts.

Commands	**Description**
addrepos *repository* [-user *user_name* -password *password*]	Add a repository. Use this command to add a repository in the create context. You can then install or uninstall packages found in the given repository If you attempt to add a secured repository without providing the -user and -password options, you will be prompted for these credentials. The maximum number of attempts allowed is three. If you connect to one or more secured repositories, the Response File Editor will create a keyring file to store the user credentials for connecting to those repositories. This keyring file will have the same filename as the scripted response file you create. In order to run that scripted response file, you will need to have the keyring file in the same directory as the scripted response file. The *repository* variable is the path to the repository to add to this response file. The repository path can be a file path or a URL. The Response File Editor supports FTP, HTTP, and HTTPS. Use the `addrepos` command with the following options: • `user` : User name for a repository that requires authentication. • `password` : Password for a repository that requires authentication.

close	Exit the create context and return to the base context.
help [*command*]	List the available commands. When you use the help command with another command, a short description of the command is shown.
history [-clear]	List the previously entered commands. Use the history command with the -clear option: • -clear: Clear the history for the current context.
install *offering_id* [-version *version* -platform *platform*]	Add an install action to the response file for the given package. After you enter the install command, the Response File Editor prompts you to provide information about the package to install. The information that you provide includes the installation location, language and feature selections, and information specific to the product configuration. The *offffering_id* variable is the package to install. To see a list of available packages, press Tab after entering the install command. If a long list of values are returned, you can enter the first few letters and then press Tab. Select a package and press Enter. To select a package, either type the full name or use the Tab key to automatically complete the package name. Use the install command with the following options: • -version: The version of the package to install. You must enter the full package version. • -platform : The platform to install on. You use the `all` value to include all supported platforms or you specify a platform.
`list`	List the commands added to the scripted response file.
save [*response_file* -overwrite yes|no]	Save the scripted response file. If a file name is not provided, the Response File Editor prompts for a file name.

	Use the save command with this options: • -overwrite : Indicate if you want to overwrite the given file name if the file already exists. If you do not provide this option and the file does exist, the Response File Editor prompts you to overwrite the file.
uninstall *offering_id*	Add an uninstall action to the response file for the given package. After you enter the uninstall command, the Response File Editor prompts you to provide additional information about the package to uninstall. The information that you provide includes the installation location and information specific to the product configuration. The *offering_id* variable is the package to uninstall. To see a list of available packages, press Tab after entering the uninstall command. If a long list of values are returned, you can enter the first few letters and then press Tab. Select a package and press Enter. To select a package, either type the full name or use the Tab key to automatically complete the package name.

Sample Response Files

The following section contains sample response files with examples of user-defined functions.

Select the package to install based on the client location

```
rfe_script_version = '1.0.0'

//Function that selects repository to use based on client location
def checkLocationForCCSetting() {
        InetAddress inetadress = InetAddress.getLocalHost()
        if ( inetadress.getCanonicalHostName().contains('.location1.ibm.com') ) {
                "CCSetting1"
        }
        if ( inetadress.getCanonicalHostName().contains('.location2.ibm.com') ) {
                "CCSetting2"
        }
        else {
```

```
                "CCSettingDefault"
        }
}

repositories {
        addUrl 'https://someserver.com/cc/repository/'
}

env env1 {
        win32 {
                installLocation = '${specialFolder:PROGRAM_FILES}\\IBM\\RationalSDLC'
                installCommonLocation = '${specialFolder:PROGRAM_FILES}\\IBM\\IMShared'
        }
}

pkg RCCW1 {
        id = 'com.ibm.rational.clearcase.nt_i386'
        version = '8.0.2.0000-8-0-0-02-00-2012A-D120326'
        addProperty 'user.StopRunningProcesses', 'true'
        addProperty 'user.CC_DisableAccountConfiguration', 'true'
        addProperty 'user.CC_AccountDomain', 'TESTDOMAIN1'
        addProperty 'user.CC_ServerProcessUserID', 'testuser1'
        addProperty 'user.CC_ServerProcessPassword', 'fufgZbY47EfxLYarBAIxeQ=='
        addProperty 'user.CC_AdminGroup', 'cc_group1'
        addProperty 'user.CC_RegSvrHostName', 'testregserver1'
        addProperty 'user.CC_RegWinRegionName', 'win_reg1'
        addProperty 'user.CC_ScalingFactor', '1'
        addProperty 'user.CC_MaxNumMnodesKeepOnVOBFreeList', '1800'
        addProperty 'user.CC_MaxNumMnodesKeepOnCleartextFreeList', '1800'
}

pkg RCCW2 {
        id = 'com.ibm.rational.clearcase.nt_i386'
        version = '8.0.2.0000-8-0-0-02-00-2012A-D120326'
        addProperty 'user.StopRunningProcesses', 'true'
        addProperty 'user.CC_DisableAccountConfiguration', 'true'
        addProperty 'user.CC_AccountDomain', 'TESTDOMAIN2'
        addProperty 'user.CC_ServerProcessUserID', 'testuser2'
        addProperty 'user.CC_ServerProcessPassword', 'fufgZbY47EfxLYarBAIxeQ=='
        addProperty 'user.CC_AdminGroup', 'cc_group2'
        addProperty 'user.CC_RegSvrHostName', 'testregserver2'
        addProperty 'user.CC_RegWinRegionName', 'win_reg2'
        addProperty 'user.CC_ScalingFactor', '1'
        addProperty 'user.CC_MaxNumMnodesKeepOnVOBFreeList', '1800'
        addProperty 'user.CC_MaxNumMnodesKeepOnCleartextFreeList', '1800'
}

pkg RCCW3 {
        id = 'com.ibm.rational.clearcase.nt_i386'
```

```
        version = '8.0.2.0000-8-0-0-02-00-2012A-D120326'
        addProperty 'user.StopRunningProcesses', 'true'
        addProperty 'user.CC_DisableAccountConfiguration', 'true'
        addProperty 'user.CC_AccountDomain', 'TESTDOMAIN'
        addProperty 'user.CC_ServerProcessUserID', 'testuser'
        addProperty 'user.CC_ServerProcessPassword', 'fufgZbY47EfxLYarBAIxeQ=='
        addProperty 'user.CC_AdminGroup', 'cc_group'
        addProperty 'user.CC_RegSvrHostName', 'testregserver'
        addProperty 'user.CC_RegWinRegionName', 'win_reg'
        addProperty 'user.CC_ScalingFactor', '1'
        addProperty 'user.CC_MaxNumMnodesKeepOnVOBFreeList', '1800'
        addProperty 'user.CC_MaxNumMnodesKeepOnCleartextFreeList', '1800'
}

if ( checkLocationForCCSetting() == 'CCSetting1' ) install RCCW1 using env1
if ( checkLocationForCCSetting() == 'CCSetting2' ) install RCCW2 using env1
if ( checkLocationForCCSetting() == 'CCSettingDefault' ) install RCCW3 using env1
```

Select the language based on the system locale

```
rfe_script_version = '1.0.0'

//Sets the language required based on the client system locale
def getSystemLanguage() {
	Locale locale = Locale.getDefault()
	String currentLocale = locale.toString()

	if ( currentLocale.startsWith('fr' ) ) {
		'en,fr'
	} else if ( currentLocale.startsWith('de' ) ) {
		'en,de'
	} else if ( currentLocale.startsWith('ja') ) {
		'en,ja'
	} else {
		'en'
	}
}

repositories {
	addUrl 'https://someserver.com/rtc/repository'
	addUrl 'https://someserver.com/rad/repository/'
}

env env1 {
	win32 {
		installLocation = '${defaultInstallRootLocation}\\IBM\\SDP${beta}'
		installCommonLocation = '${defaultInstallRootLocation}\\IBM\\SDPShared${beta}'
	}
	linux {
		installLocation = '${defaultInstallRootLocation}/IBM/SDP${beta}'
		installCommonLocation = '${defaultInstallRootLocation}/IBM/SDPShared${beta}'
	}
}

env env2 {
	win32 {
		installLocation = '${defaultInstallRootLocation}\\IBM\\SDP${beta}'
		installCommonLocation = '${defaultInstallRootLocation}\\IBM\\SDPShared${beta}'
	}
	linux {
		installLocation = '${defaultInstallRootLocation}/IBM/SDP${beta}'
		installCommonLocation = '${defaultInstallRootLocation}/IBM/SDPShared${beta}'
	}
}
```

```
pkg RTCC4EIDE1 {
	id = 'com.ibm.team.install.rtc.client.eclipse'
	version = '3.0.1000.RTC-I20110602-0252'
	addProperty 'user.help.option', 'remote'
	languages = getSystemLanguage()
}

pkg RAD4WSS1 {
	id = 'com.ibm.rational.application.developer.v80'
	version = '8.0.1.20101027_1052'
	addProperty 'user.help.option', 'remote'
	languages = getSystemLanguage()
}

install RTCC4EIDE1 using env1
install RAD4WSS1 using env2
```

Select the repository based on the client location

```
rfe_script_version = '1.0.0'

//Function that selects repository to use based on client location
def useLocalRepo() {
	InetAddress inetadress = InetAddress.getLocalHost()
	println inetadress.getCanonicalHostName()
	if ( inetadress.getCanonicalHostName().contains('.location1.ibm.com') ) {
		"servername.location1.ibm.com"
	}
	else {
		"servername.location2.ibm.com"
	}
}

def printToScreen(value) {
	println value
}

repositories {
	addUrl 'https://' + useLocalRepo() + '/rad/repository/'
}

env env1 {
	win32 {
		installLocation = '${defaultInstallRootLocation}\\IBM\\SDP${beta}'
		installCommonLocation = '${defaultInstallRootLocation}\\IBM\\SDPShared${beta}'
	}
	linux {
		installLocation = '${defaultInstallRootLocation}/IBM/SDP${beta}'
		installCommonLocation = '${defaultInstallRootLocation}/IBM/SDPShared${beta}'
	}
}

pkg RAD4WSS1 {
	id = 'com.ibm.rational.application.developer.v80'
	version = '8.0.1.20101027_1052'
	addProperty 'user.help.option', 'remote'
}

install RAD4WSS1 using env1
```

Notices and trademarks

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this documentation in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this documentation. The furnishing of this documentation does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this documentation and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.