Course Exercises

HCLSoftware
BigFix Relevance Course

Course code BESFND401R

HCL BigFix

For more information

To learn more about BigFix, contact your HCLSoftware representative, HCL Business Partner, or visit www.BigFix.com.

About HCLSoftware

HCLSoftware, a division of HCLTech, develops, markets, sells, and supports software for digital transformation, data, analytics and insights, AI and automation, and enterprise security. HCLSoftware is the cloud-native solution factory for enterprise software and powers millions of apps at more than 20,000 organizations, including more than half of the Fortune 1000 and Global 2000 companies. HCLSoftware's mission is to drive ultimate customer success through relentless product innovation. https://www.hcltechsw.com

© Copyright 2023 HCL

Produced in the United States of America.

All product names, trademarks and registered trademarks are property of their respective owners.

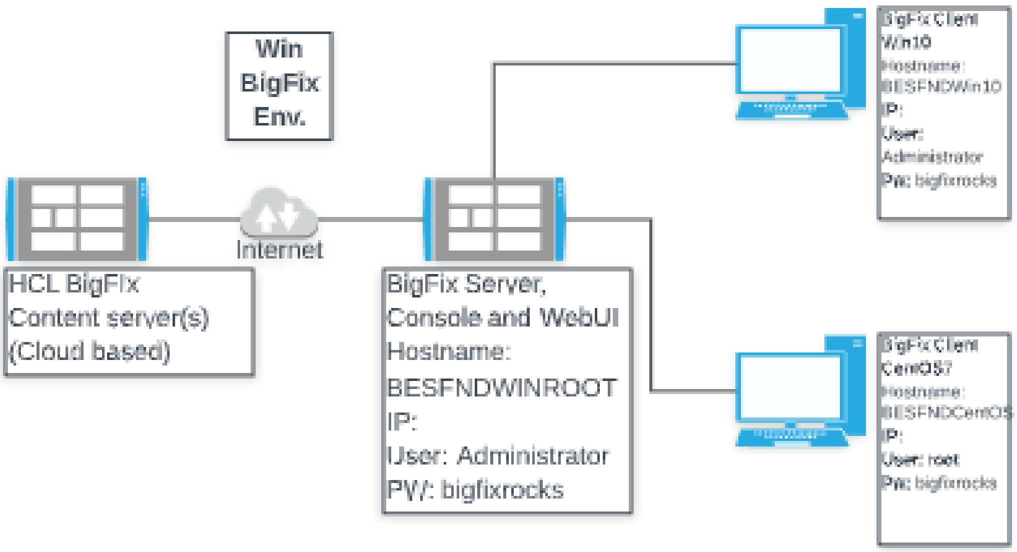
Contents

Release Date	Version	Authors	Comments
Aug 2019	2.0	Michael Stone	Initial version
May 2020	2.1	Michael Stone/ Mark Leaphart	Several edits after initial version.
July 2022	3.0.9.5	Kathy Nguyen	Updated Exercise 48 with steps to get to QnA via Terminal on BESFNDCENTOS
December 2022	3.1	Kathy Nguyen	Removed Exercise 27 and renumbered, updated Exercises 20 & 34, updated Exercise 37/38 to optional

About these exercises

Your lab environment consists of three virtual machine connected to a nat'd network with external Internet connectivity. The details for the virtual machines are as follows:

- **BESFNDWIN10** is a Windows 10 Enterprise system and is used as a BigFix client. The BigFix Console is also installed on this system. This system also has the Fixlet Debugger and a few other tools installed.
- Unless otherwise specified in the exercises, the operating system passwords for the `Administrator` user on the virtual machines are all set to `bigfixrocks`.



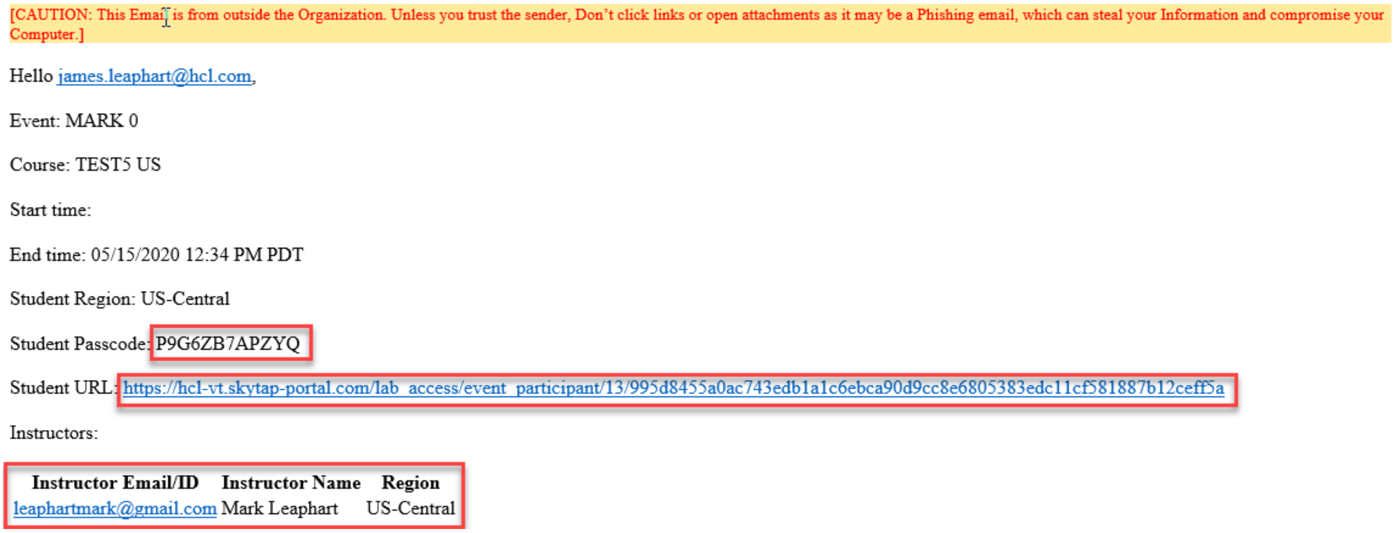
The below table contains a summary of the VM images used in this lab guide:

	Host Name	BigFix Components	OS	IP Address	Userid & Password
1	BESFNDWINROOT	BigFix Windows based Server, Console, WebUI,	Windows 2016	10.0.0.1	Administrator bigfixrocks
2	BESFNDRHROOT	BigFix Linux based Server, WebUI, Client	RHEL7	10.0.0.1	root bigfixrocks
3	BESFNDWIN10	BigFix Client, Console	Windows 10	10.0.0.2	Administrator bigfixrocks
4	BESFNDCENTOS	BigFix Client	CentOS7	10.0.0.3	root bigfixrocks
	All	BigFix Console creds			adminmo B1gfixrocks

Accessing Lab Environment

The BigFix Lab environment is currently being hosted in Skytap's (www.skytap.com). To access this environment, you will need the url, id, and password sent to your registered email address (this would be from Skytap.com). If you are a USA Federal customer – your instructor will provide you your credentials and access url(s).

Students will receive an email (this is the email address you provided when you registered for the course) from Skytap that contains the url to YOUR Skytap environment, the login id and password for this specific course. It will look something like this:



[CAUTION: This Email is from outside the Organization. Unless you trust the sender, Don't click links or open attachments as it may be a Phishing email, which can steal your Information and compromise your Computer.]

Hello james.leaphart@hcl.com,

Event: MARK 0

Course: TEST5 US

Start time:

End time: 05/15/2020 12:34 PM PDT

Student Region: US-Central

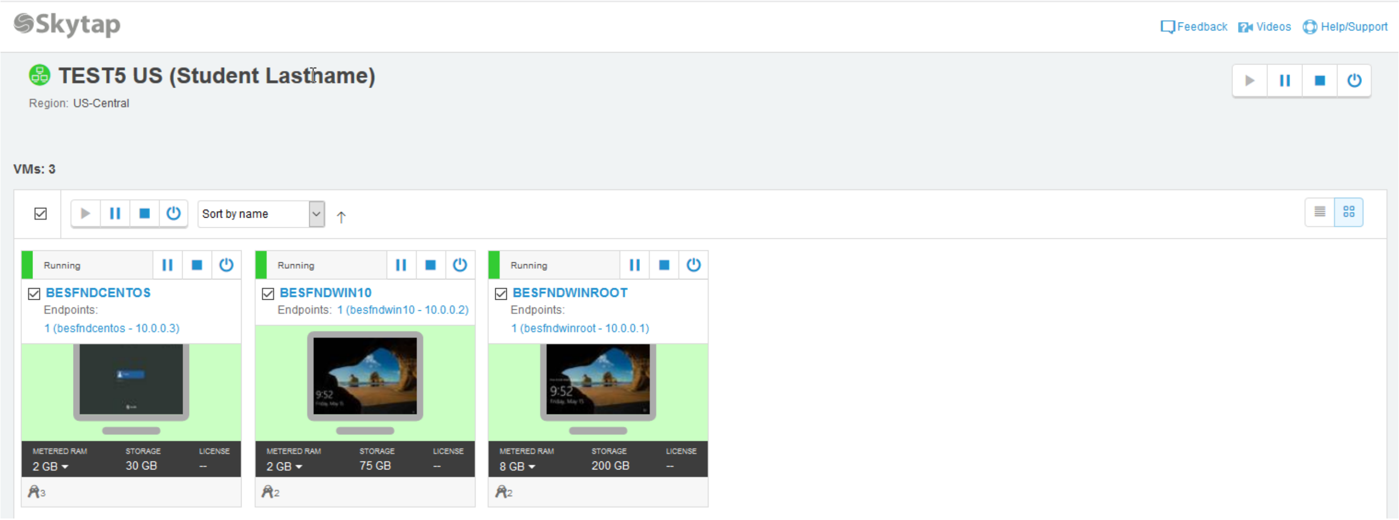
Student Passcode: P9G6ZB7APZYQ

Student URL: https://hcl-vt.skytap-portal.com/lab_access/event_participant/13/995d8455a0ac743edb1a1c6ebca90d9cc8e6805383edc11cf581887b12ceff5a

Instructors:

Instructor Email/ID	Instructor Name	Region
leaphartmark@gmail.com	Mark Leaphart	US-Central

Click on the url provided in your email and provide your credentials (if asked). You will be taken into Skytap and you will see your provisioned environment.



TEST5 US (Student Lastname)

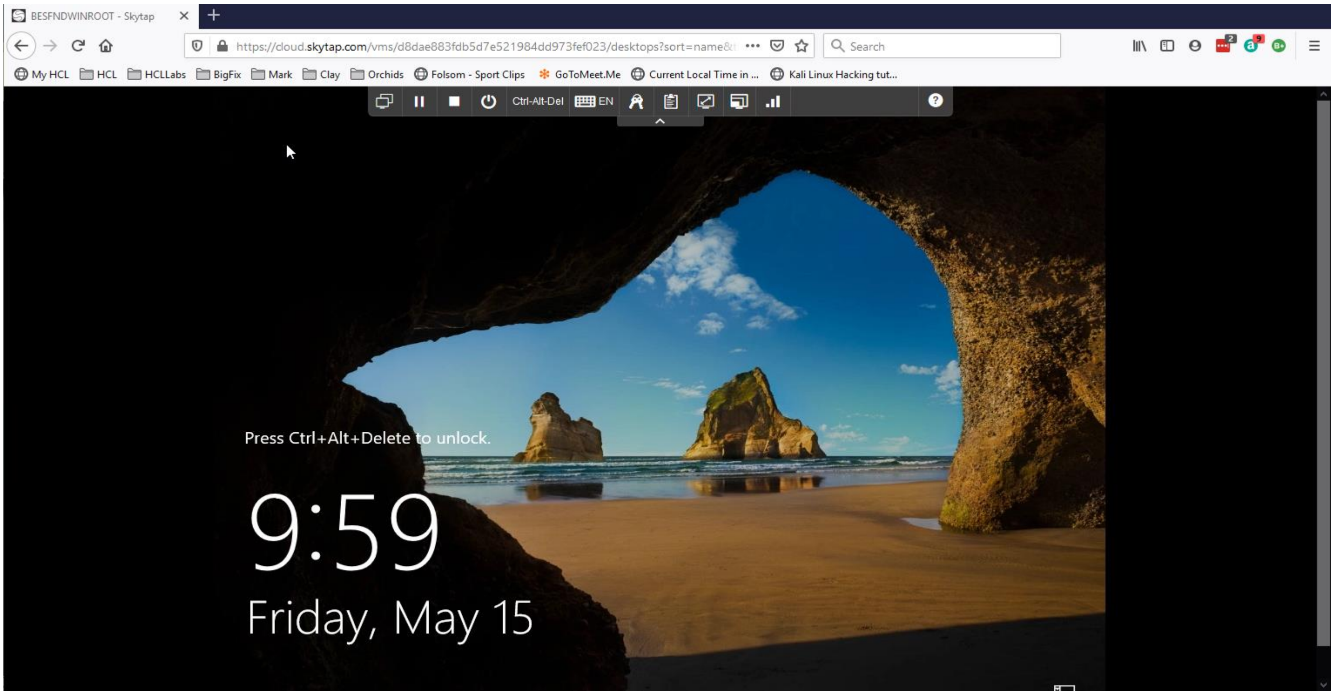
Region: US-Central

VMs: 3

Sort by name

Status	VM	Endpoints	Metered RAM	Storage	License
Running	BESFNDCENTOS	1 (besfndcentos - 10.0.0.3)	2 GB	30 GB	--
Running	BESFNDWIN10	1 (besfndwin10 - 10.0.0.2)	2 GB	75 GB	--
Running	BESFNDWINROOT	1 (besfndwinroot - 10.0.0.1)	8 GB	200 GB	--

The vm's provided here are accessible via your browser (rdp is not required). Click on a vm and your browser will present your vm:



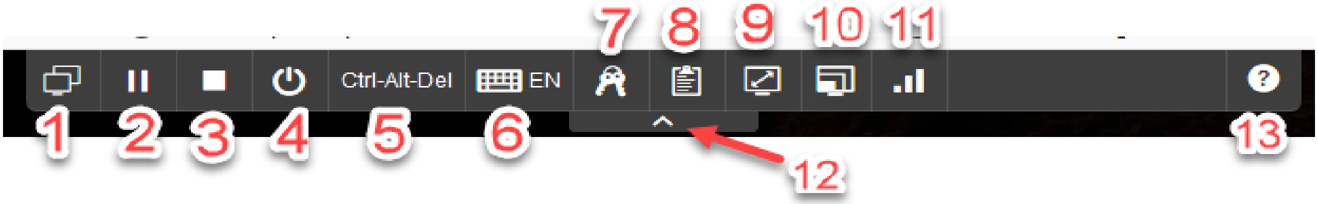
BESFNDWINROOT - Skytap
https://cloud.skytap.com/vms/d8dae883fdb5d7e521984dd973fef023/desktops?sort=name&
Search
My HCL
HCL
HCLLabs
BigFix
Mark
Clay
Orchids
Folsom - Sport Clips
GoToMeet.Me
Current Local Time in ...
Kali Linux Hacking tut...
Ctrl-Alt-Del
EN

Press Ctrl+Alt+Delete to unlock.

9:59

Friday, May 15

Now let's look at the controls in the browser for this vm.



1) Environment VM's: View all vm's in your environment or switch to another vm in your environment
2) Suspend this vm
3) Shutdown this vm
4) Power options for vm - a) shutdown, b) reset, c) power off
5) Ctrl-Alt-Del is passed to the vm
6) Keyboard layout and or inject key combinations
7) Credentials: operating system and applications in this vm
8) VM Clipboard
9) Fit to window
10) Change video resolution
11) Network Quality Indicator
12) Hide this tool bar
13) Help

When you open any of the Windows vm's, always answer YES to the network connection question.

Starting the environment –

Windows ROOT Server

In this exercise, you will install BigFix and start the configuration process.

____1) Verify that the following virtual machines are started:

- BigFix Server: **BESFNDWINROOT**
- BigFix Windows Client: **BESFNDWIN10**
- BigFix Linux Client: **BESFNDCENTOS**

____2) Switch to the BigFix Server virtual machine. If you are logged off, log in to the server as **Administrator** with a password of **bigfixrocks**

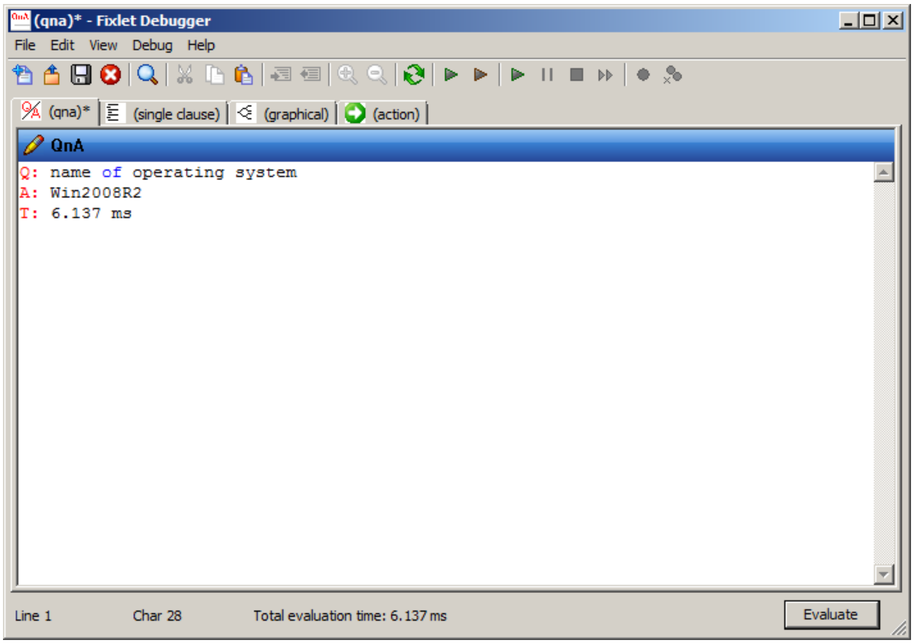
Unit 1 HCL BigFix Content Development: Introduction

This unit has no student exercises.

Unit 2 Basic Relevance Exercises

You perform these exercises by using the HCL BigFix Fixlet Debugger. This tool is installed on the **student** virtual machine desktop. To start the tool, click **Start > All Programs > BigFix > BigFix Fixlet Debugger**.

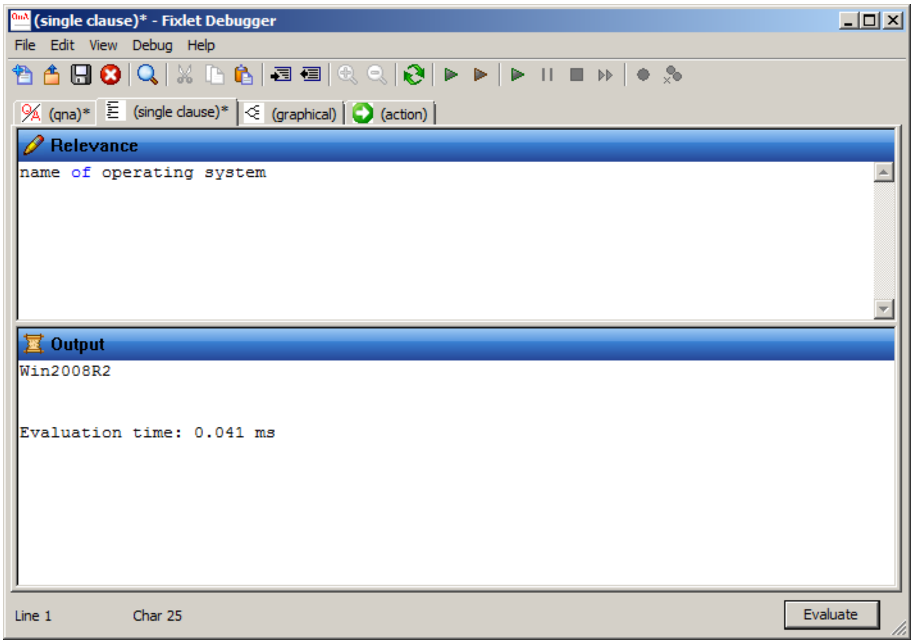
When you enter the code in the Fixlet Debugger on the query and answer **(qna)** tab, you must preface the query with a **Q:** to designate it as such. The answer will appear on a line with an **A:**, execution time to process the query is indicated by **T:**, and if an error occurs, designate it with an **E:**. Type information, if enabled, is indicated by **I:**.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)
QnA
Evaluate

```
Q: name of operating system
A: Win2008R2
T: 6.137 ms
```

Line 1 Char 28 Total evaluation time: 6.137 ms

When you enter code into the **(single clause)** tab, enter it directly in the **Relevance** text box and without the **Q:** prefix. Results appear in the **Output** text box.



```
name of operating system
```

```
Win2008R2


Evaluation time: 0.041 ms
```

Note: You will perform the labs for this module using the HCL BigFix Fixlet Debugger on the **student** virtual machine. You can also perform these labs on a local Windows system by downloading and installing the most recent version of the Fixlet Debugger from the BigFix page.

Exercise 1 Starting the environment

1. If you are logged off, log in to the server as **Administrator** with a password of **bigfixrocks**.

2. Click **Start > BigFix > BigFix Fixlet Debugger**. The BigFix Fixlet Debugger opens.

Exercise 2 Determining the size of a Windows file

In this exercise, you will develop a Relevance query to determine the size of a Windows file named `calc.exe`. This file is in the Windows system folder. This Relevance query must run correctly on any version of Windows, on any installation drive, and in any unique installation directory. You can query the size of a file in a specified path by using the following syntax:

```
Q: size of file “regedit.exe” of windows folder
```

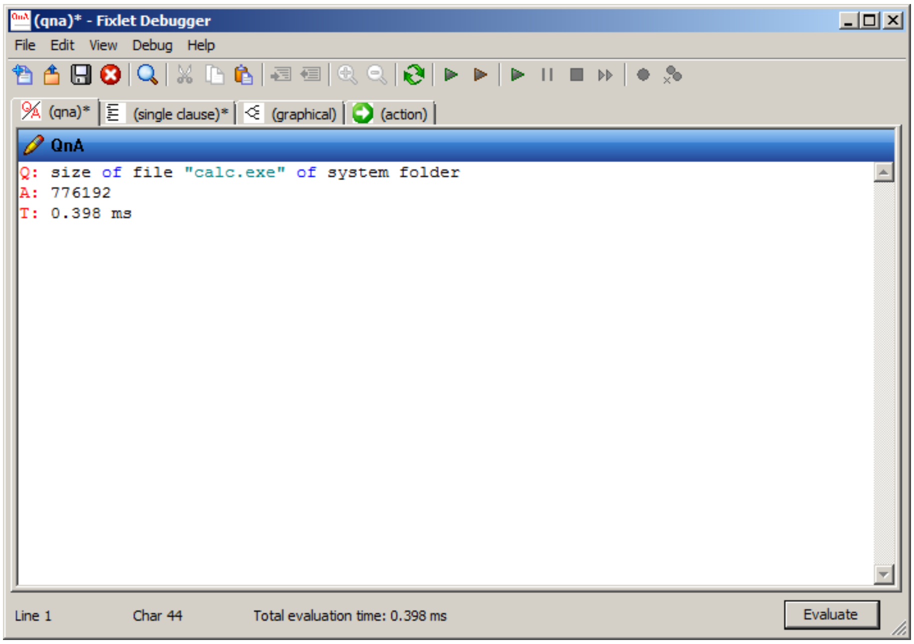
Use these steps to perform the query:

1. In the Fixlet Debugger, select the **(qna)** tab.

2. In the **QnA** text box, enter this query:

   ```
   Q: size of file “calc.exe” of system folder
   ```

3. Click **Evaluate**.



```
Q: size of file "calc.exe" of system folder
A: 776192
T: 0.398 ms
```

Exercise 3 Determining the version of a file

In this exercise, you develop a Relevance query to determine the version of a Windows file named `mshtml.dll`. This file is in the Windows system folder.

You can query the version of a file in a specified path by using the following syntax:

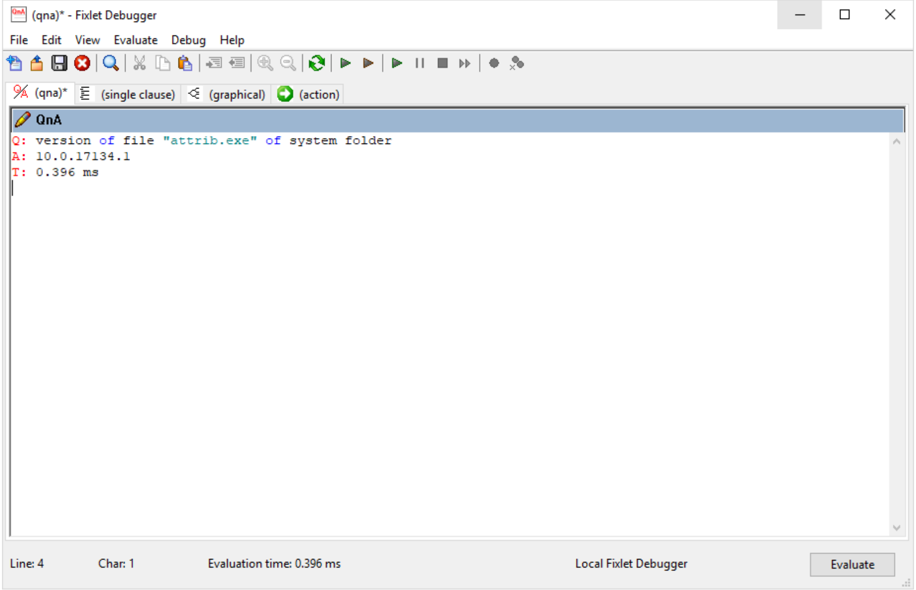
```
Q: version of file “regedit.exe” of windows folder
```

Use these steps to perform the query:

1. In the Fixlet Debugger, select the **(qna)** tab.
2. In the **QnA** text box, enter this query:

   ```
   Q: version of file “attrib.exe” of system folder
   ```

3. Click **Evaluate**.



```
Q: version of file "attrib.exe" of system folder
A: 10.0.17134.1
T: 0.396 ms
```

Exercise 4 Comparing versions

In this exercise, you develop a Relevance query to find the version of the Windows **cmd.exe** program and verify whether it is greater than or equal to version 5.

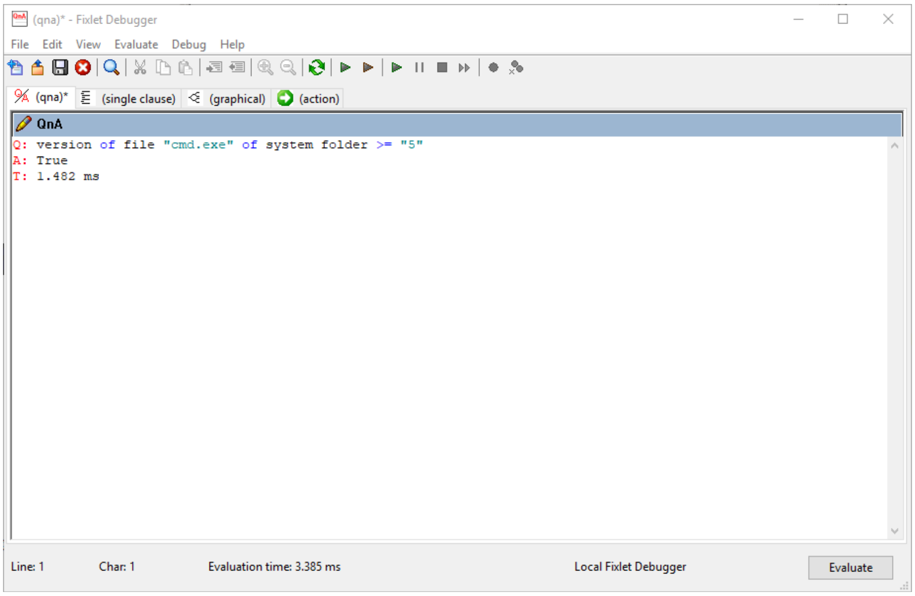
This file is in the system folder. You can query the version of a file and compare the version to a static value by using the following syntax:

```
Q: version of file “attrib.exe” of system folder < “10.0.18000.1”
```

Use these steps to perform the query:

1. In the Fixlet Debugger, select the **(qna)** tab.
2. In the **QnA** text box, enter this query:

```
Q: version of file “cmd.exe” of system folder >= “5”
```



```
Q: version of file "cmd.exe" of system folder >= "5"
A: True
T: 1.482 ms
```

Line: 1 Char: 1 Evaluation time: 3.385 ms Local Fixlet Debugger

Exercise 5 Checking for the existence of a file

In this exercise, you develop a Relevance query to check whether the file named `calc.exe` exists in the Windows system folder.

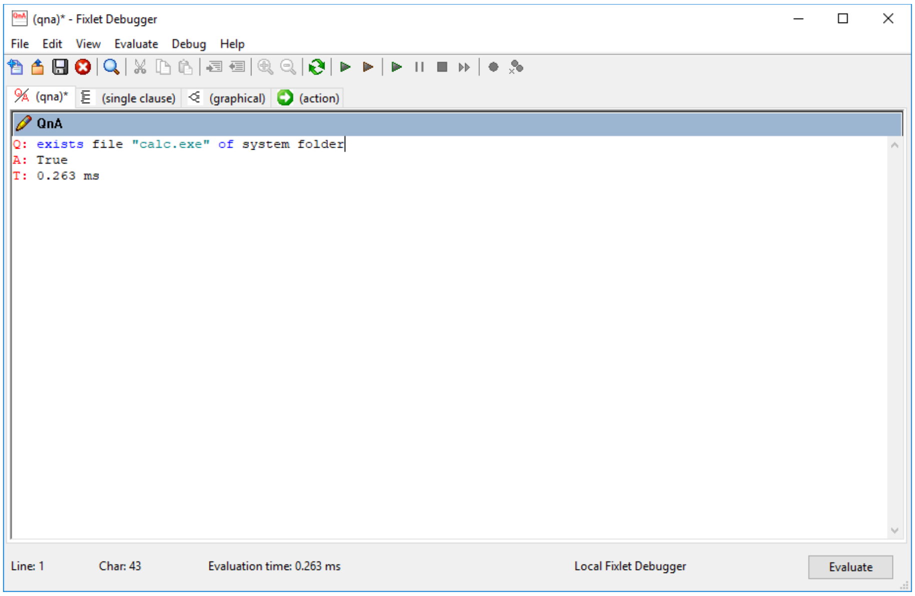
You can query the existence of a file by using the following syntax:

```
Q: exists file “c:\windows\explorer.exe”
```

Use these steps to perform the query:

1. In the Fixlet Debugger, select the **(qna)** tab.
2. In the **QnA** text box, enter this query:

```
Q: exists file “calc.exe” of system folder
```



```
Q: exists file "calc.exe" of system folder
A: True
T: 0.263 ms
```

Exercise 6 Using Boolean operators

In this exercise, you use Boolean operators to develop a Relevance query to return **true** for two specific criteria:

- The file `write.exe` exists in the System folder.
- The version of this file is later than version 9.1.2600.1106.

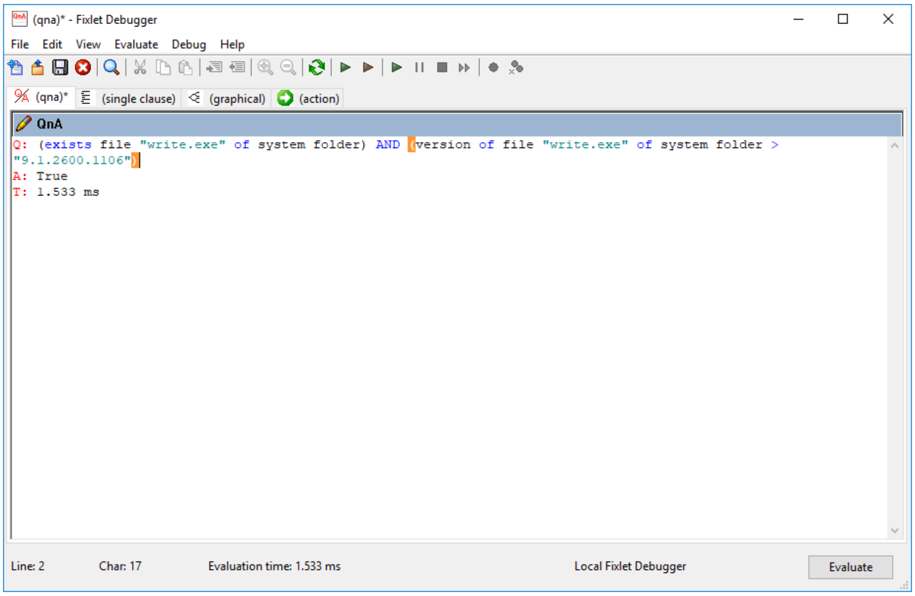
You can query the existence of a file and incorporate a version comparison by using the following syntax:

```
Q: not exists file “url.dll” of system folder OR version of file “url.dll” of
system folder < “6.0.2800.1643”
```

Use these steps to perform the query:

1. In the Fixlet Debugger, select the **(qna)** tab.
2. In the **QnA** text box, enter this query:

```
Q: (exists file “write.exe” of system folder) AND (version of file “write.exe” of
system folder > “9.1.2600.1106”)
```



```
Q: (exists file "write.exe" of system folder) AND (version of file "write.exe" of system folder >
"9.1.2600.1106")
A: True
T: 1.533 ms
```

Line: 2 Char: 17 Evaluation time: 1.533 ms

You can also use this query:

```
Q: (NOT exists file “write.exe” of system folder) AND (version of file
“write.exe” of system folder > “9.1.2600.1106”)
```

Exercise 7 Determining whether an application is running

In this exercise, you determine whether an application is running on the Win10 system and if that application is a minimum acceptable version. You develop a Relevance query to return **true** for two specific criteria:

- The user is running the BigFix console (**Besconsole.exe**).
- The version of the console is later than V9.0.

You can query the existence of a running application and the installation path of it by using the following syntax:

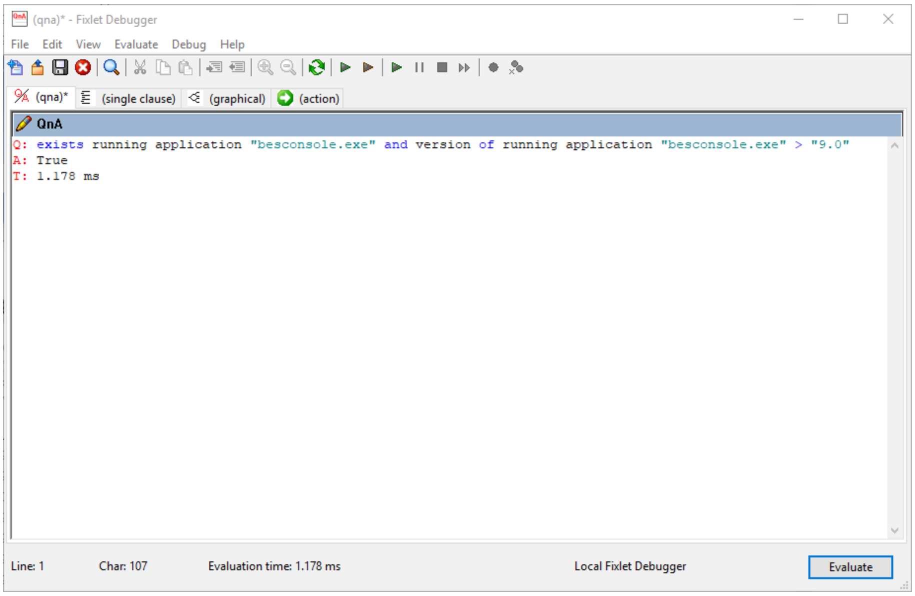
```
Q: Exists running application “besclient.exe” AND pathname of regapp
“besclient.exe” != “C:\Program Files\BigFix Enterprise\BES
Client\BESClient.exe”
```

Use these steps to perform the query:

1. If it is not already running, start the BigFix console. On the Windows desktop, double-click the **BigFix Console** icon.
2. Verify the server host name and username (**adminmo**) is correct.
3. Enter **`B1gfixrocks`** in the **Password** field and click **OK**. The BigFix Console opens.
4. Minimize the console so that it does not interfere with your exercises. You must have the program running, but it does not have to be visible.
5. In the Fixlet Debugger, select the **(qna)** tab.
6. In the **QnA** text box, enter this query:

   ```
   Q: exists running application “BESConsole.exe” AND version of running
   application “BESConsole.exe” > “9.0”
   ```

7. Click **Evaluate**.


(qna)* - Fixlet Debugger
File Edit View Evaluate Debug Help
(qna)* (single clause) (graphical) (action)
QnA
Line: 1
Char: 107
Evaluation time: 1.178 ms
Local Fixlet Debugger
Evaluate

```
Q: exists running application "besconsole.exe" and version of running application "besconsole.exe" > "9.0"
A: True
T: 1.178 ms
```

Exercise 8 Using the if-then-else structure

In this exercise, you use the if-then-else structure within a Relevance query to find a specified log file and return the value of its size. If the specified file does not exist, you indicate this by an error statement. Your objective is to write a retrieved property or analysis that returns the size of the `c:\applog.txt` log file.

Relevance

- If the log does not exist, return **"No Log File"**.
- Create the file to test the Relevance and return a value.

You can query the size of a file by using the following syntax:

```
Q:  Size of <file>
```

You can implement an **`if-then-else`** construct by using the following syntax:

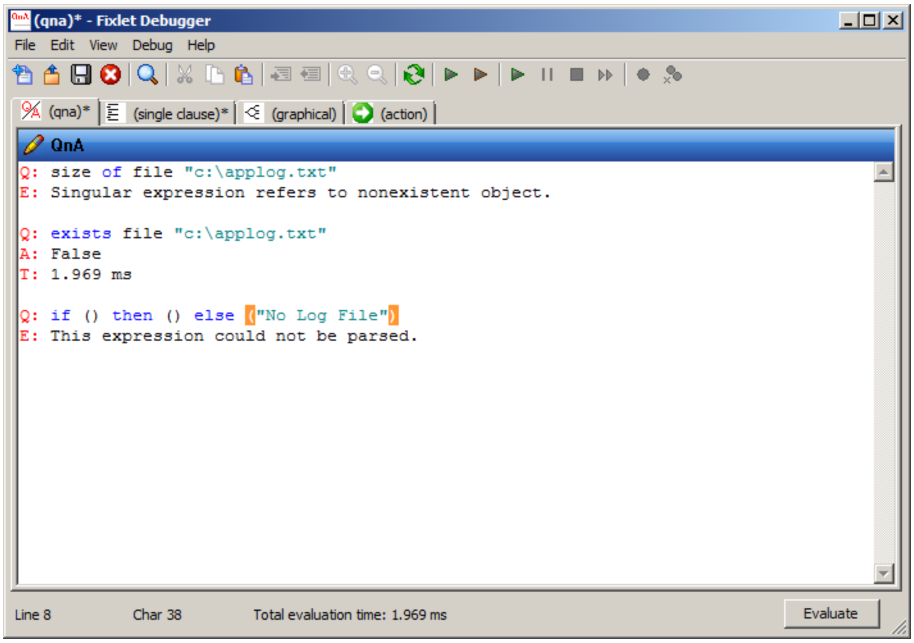
```
if () then () else ()
```

Use these steps to perform the query:

1. In the Fixlet Debugger, select the **(qna)** tab.

2. To test each of the intended queries, write each clause on an individual line and click **Evaluate** to test.

```
Q: size of file “c:\applog.txt”
Q: exists file “c:\applog.txt”
Q: if () then () else (“No Log File”)
```



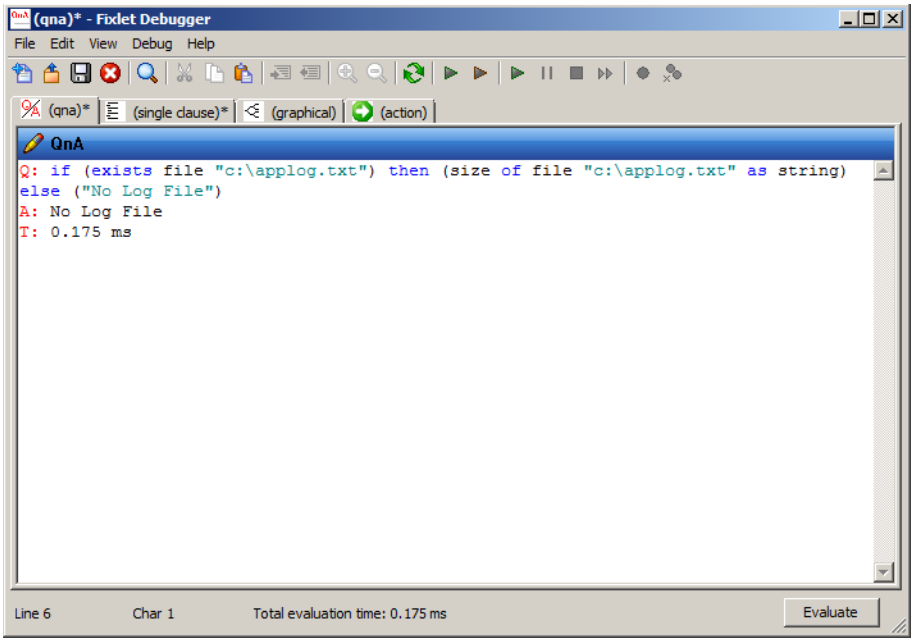
```
Q: size of file "c:\applog.txt"
E: Singular expression refers to nonexistent object.

Q: exists file "c:\applog.txt"
A: False
T: 1.969 ms

Q: if () then () else ("No Log File")
E: This expression could not be parsed.
```

3. Paste each of the Relevance queries into an **`if () then () else ()`** construct.

Hint: The size of the file returns an integer data type. It must be converted to a string to match the string type that is used in the **`else`** clause.

```
Q: if (exists file “c:\applog.txt”) then (size of file “c:\applog.txt” as
string) else (“No Log File”)
```

Because the specified file does not exist, the result of the completed Relevance is **No Log File**.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA
Evaluate

```
Q: if (exists file "c:\applog.txt") then (size of file "c:\applog.txt" as string)
else ("No Log File")
A: No Log File
T: 0.175 ms
```

Line 6 Char 1 Total evaluation time: 0.175 ms

4. You now create the file `C:\applog.txt` and test the Relevance statement again.
   a. Right-click Windows **Start** and select **Open Windows Explorer** from the menu.

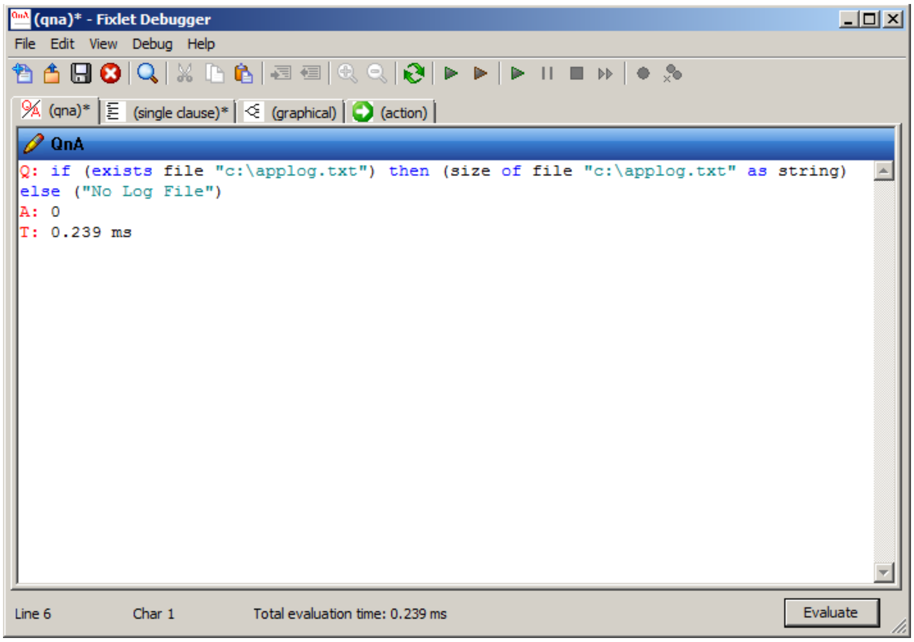
   b. Navigate to the **Local Disk (C:)** folder.

   c. In the right pane, right-click and select **New > Text Document**.

   d. Enter the name `applog.txt`. You can leave the file empty.

5. Return to the Fixlet Debugger and click **Evaluate** to run the same Relevance statement again.

This time the result is shown as 0.



```
Q: if (exists file "c:\applog.txt") then (size of file "c:\applog.txt" as string)
else ("No Log File")
A: 0
T: 0.239 ms
```

6. Edit the `C:\applog.txt` file that you previously created, add some text and save the file.
7. Return to the Fixlet Debugger and click **Evaluate** to run the same Relevance statement again. The result now shows as a positive number and indicates the number of characters in the file.

Exercise 9 Querying for quantities

In this exercise, you develop a Relevance query to return the quantities of each of these criteria:

- The number of running applications
- The number of files in the `C:\` directory
- The number of environment variables
- The number of processors in the system

You can perform the queries by using Relevance statements with syntax similar to the following example:

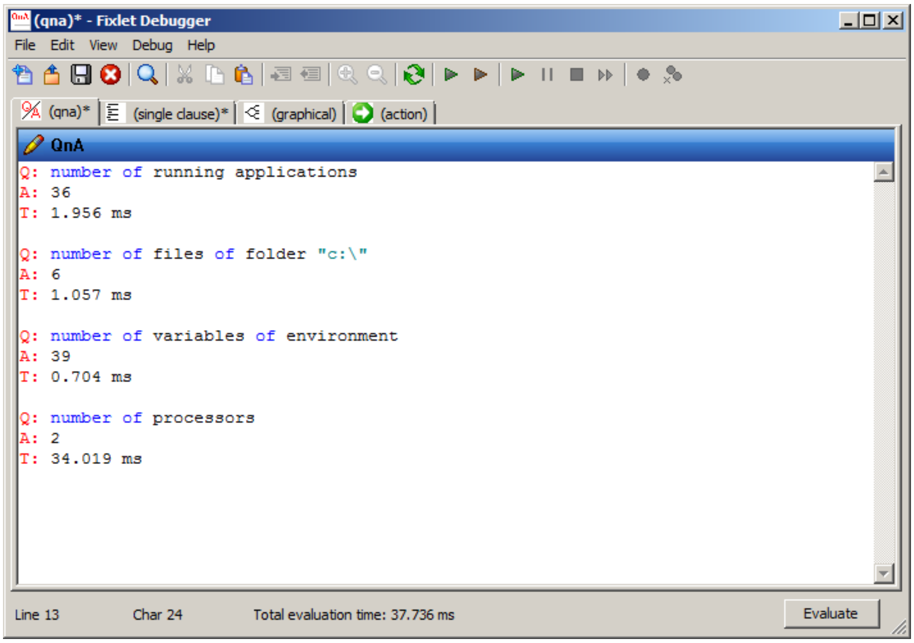
```
Q: number of files of system folder
Q: number of keys of key “HKLM/Software” of registry
Q: number of regapps
```

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. To test each of the intended queries, write each clause on an individual line and click **Evaluate**.

```
Q: number of running applications
Q: number of files of folder “c:\”
Q: number of variables of environment
Q: number of processors
```



```
Q: number of running applications
A: 36
T: 1.956 ms

Q: number of files of folder "c:\"
A: 6
T: 1.057 ms

Q: number of variables of environment
A: 39
T: 0.704 ms

Q: number of processors
A: 2
T: 34.019 ms
```

Exercise 10 Using the whose-it clause for counting executable files

In this exercise, you use the **`whose-it`** clause to develop a Relevance query that counts the number of applications in the system folder. For this exercise, an application is any file with an extension of `.exe`.

You query the files with a **`whose-it`**construct similar to the following example:

```
Q: names of files whose (version of it > “5.0”) of folder “c:\temp”
```

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. To test each of the intended queries, write each clause on an individual line.

   a. Begin with a statement that lists the files of the specified system path and click **Evaluate** to test.

   ```
   Q: names of files of system folder
   ```

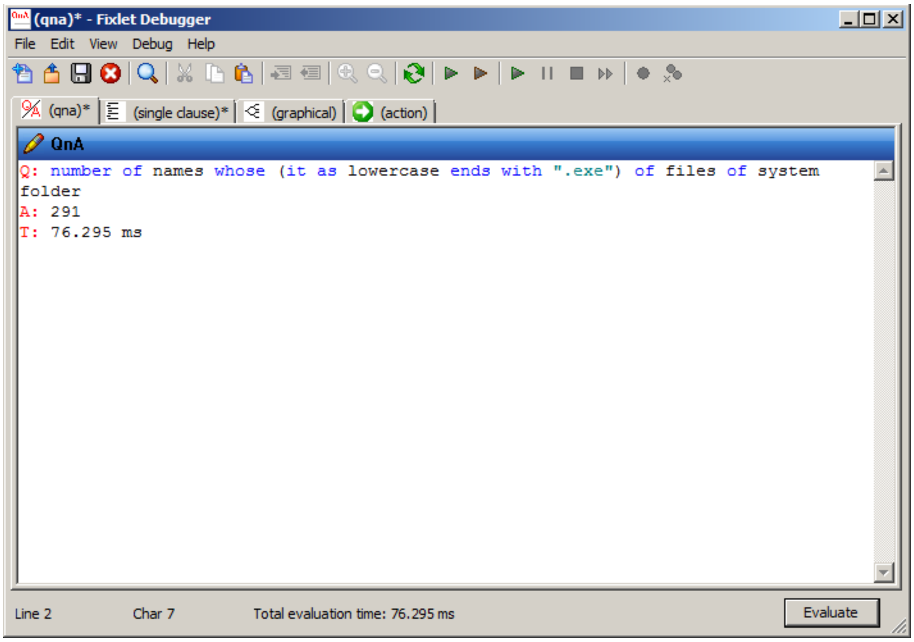
b. Add the count to identify a numeric result and click **Evaluate** to test.

```
Q: number of names of files of system folder
```

c. Apply the filter to the names result by using the `whose-it` clause.

Hint: You must use parentheses around the object that you want to filter against.

```
Q: number of names whose (it as lowercase ends with “.exe”) of files of
system folder
```



```
Q: number of names whose (it as lowercase ends with ".exe") of files of system
folder
A: 291
T: 76.295 ms
```

You can also use this query:

```
Q: number of names of files whose (name of it as lowercase ends with “.exe”) of
system folder
```

Deeper Exploration: There are at least two other methods to solve this solution. Attempt to determine them.

Exercise 11 Using the whose-it clause for finding services

In this exercise, you apply the `whose-it` clause to develop a Relevance query. With this query, you obtain a list of the display names of all the services that use an account other than LocalSystem as their login account. You can examine inspectors for the <Service> object in the Inspector Help file.

2. Create a query that lists all services.

   `Q: services`

3. Add the initial filter placeholder, which does not function.

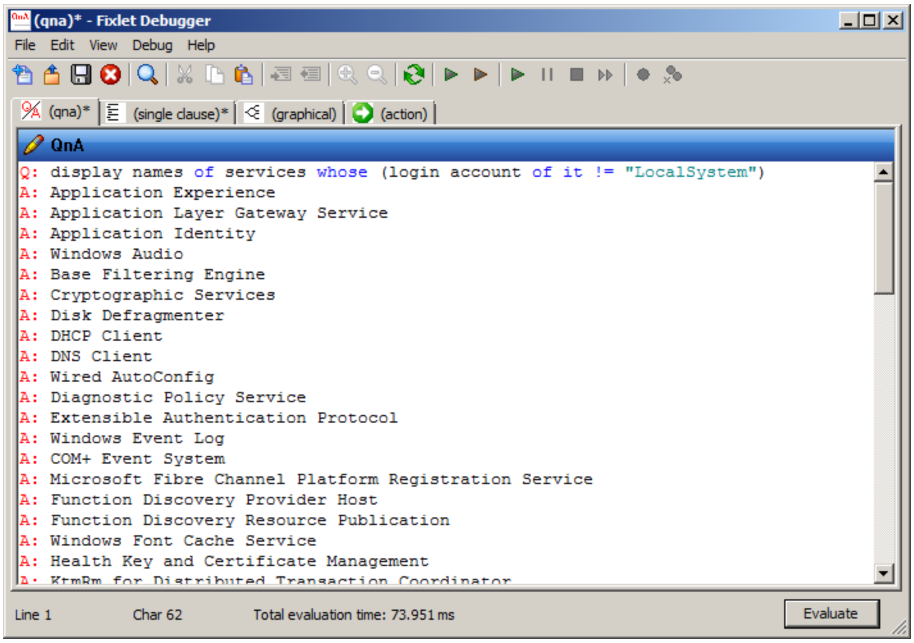
   `Q: services whose (it)`

4. Add the condition for the filter.

   `Q: services whose (login account of it != “LocalSystem”)`

5. Locate the property that you are interested in. Complete the query as follows and click **Evaluate**:

   `Q: display names of services whose (login account of it != “LocalSystem”)`



```
Q: display names of services whose (login account of it != "LocalSystem")
A: Application Experience
A: Application Layer Gateway Service
A: Application Identity
A: Windows Audio
A: Base Filtering Engine
A: Cryptographic Services
A: Disk Defragmenter
A: DHCP Client
A: DNS Client
A: Wired AutoConfig
A: Diagnostic Policy Service
A: Extensible Authentication Protocol
A: Windows Event Log
A: COM+ Event System
A: Microsoft Fibre Channel Platform Registration Service
A: Function Discovery Provider Host
A: Function Discovery Resource Publication
A: Windows Font Cache Service
A: Health Key and Certificate Management
A: KtmRm for Distributed Transaction Coordinator
```

Exercise 12 Using the whose-it clause to avoid errors

In this exercise, you develop a Relevance query that uses **`whose-it`**to return **true** only if the following criteria are met:

- The file `C:\applog.txt` exists
- The size of the file `C:\applog.txt` is greater than **0** bytes

Use a **`whose-it`**clause and verify that your Relevance does not return an error if the file does not exist.

2. Create a query for the file object.

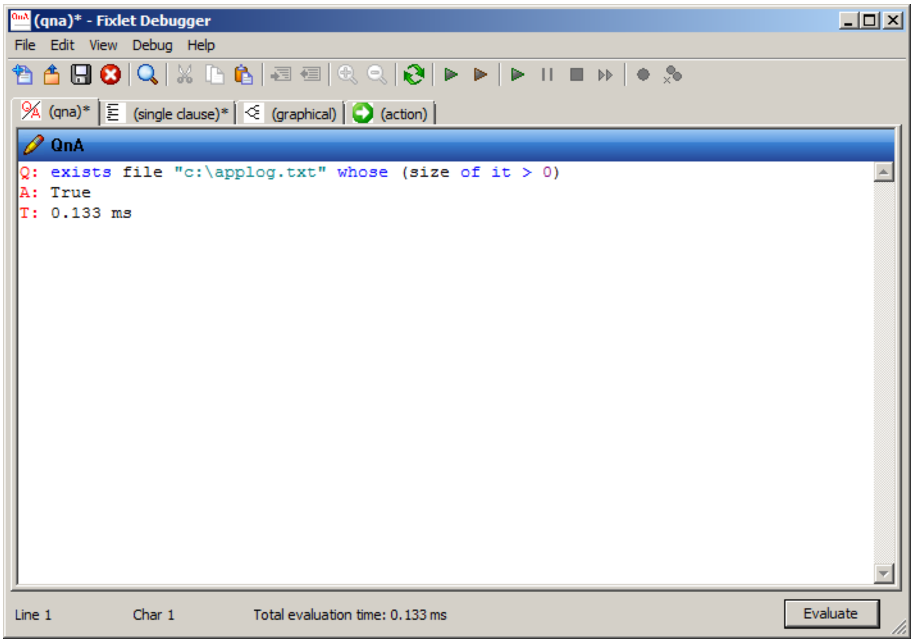
   `Q: file “c:\applog.txt”`

3. Check for the existence of the file.

   `Q: exists file“c:\applog.txt”`

4. Apply the filter by using the **`whose-it`** construct.

   `Q: exists file “c:\applog.txt” whose (size of it > 0)`



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA

```
Q: exists file "c:\applog.txt" whose (size of it > 0)
A: True
T: 0.133 ms
```

Line 1
Char 1
Total evaluation time: 0.133 ms
Evaluate

Hint: You can perform steps 2 and 3 in either order, depending on your preference.

Exercise 13 Using an it-without-a-whose construct

In this exercise, you develop a Relevance query to obtain a list of file names and file sizes in the Windows folder. When you concatenate, ensure that the integer returned by size is converted to a string. Format the output as follows:

```
Explorer.exe - 1004032
```

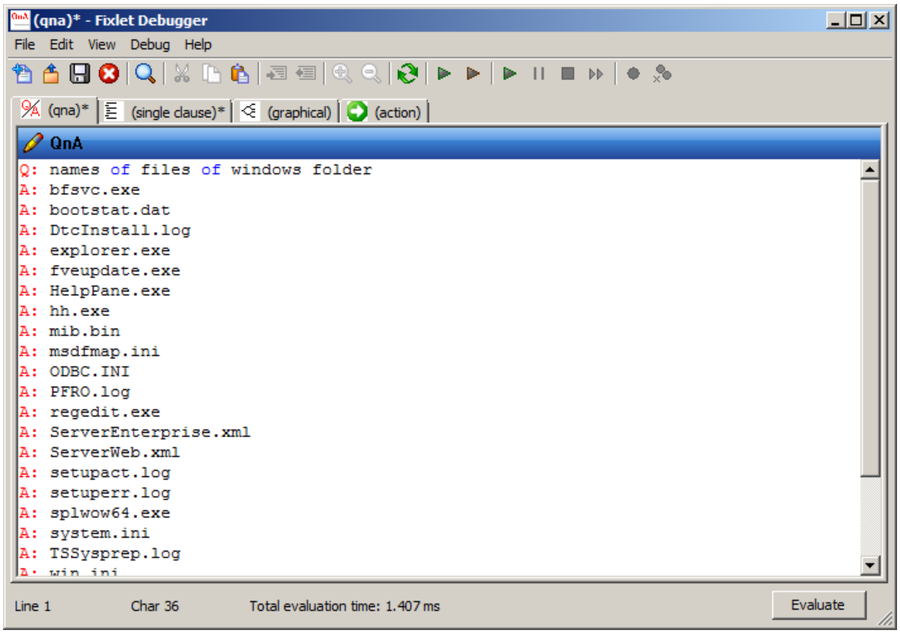
construct, similar to the following one:

```
Q: (size of it as string & “ : “ & sha1 of it) of files of folder “c:\temp”
```

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. Create a basic query to return the names of all of the files in the windows folder and click **Evaluate**.

```
Q: names of files of windows folder
```

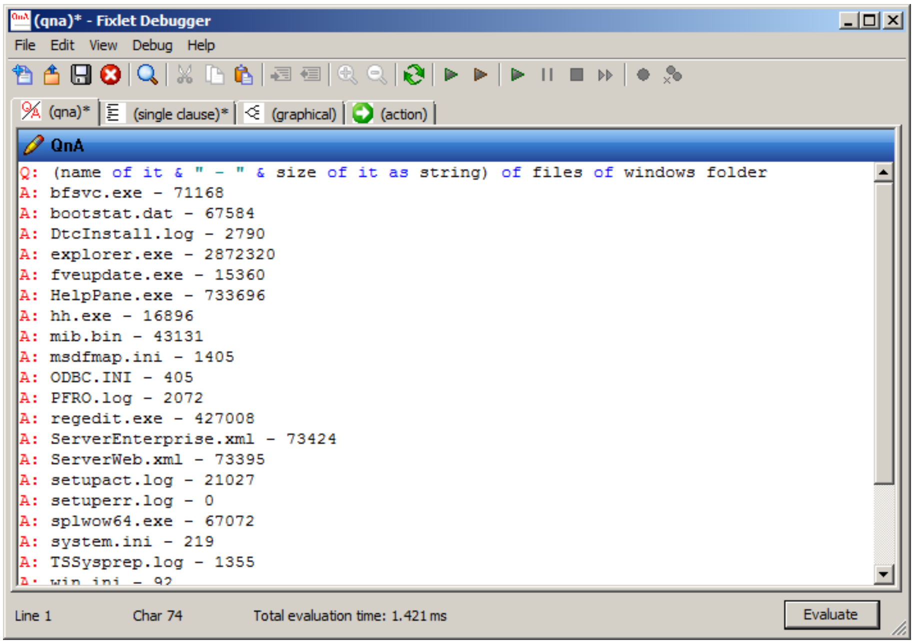


```
Q: names of files of windows folder
A: bfsvc.exe
A: bootstat.dat
A: DtcInstall.log
A: explorer.exe
A: fveupdate.exe
A: HelpPane.exe
A: hh.exe
A: mib.bin
A: msdfmap.ini
A: ODBC.INI
A: PFRO.log
A: regedit.exe
A: ServerEnterprise.xml
A: ServerWeb.xml
A: setupact.log
A: setuperr.log
A: splwow64.exe
A: system.ini
A: TSSysprep.log
A: win.ini
```

Line 1 Char 36 Total evaluation time: 1.407 ms

3. Replace **names** with the concatenation of output and click **Evaluate**.

   ```
   Q: (name of it & “ – “ & size of it as string) of files of windows folder
   ```



```
Q: (name of it & " - " & size of it as string) of files of windows folder
A: bfsvc.exe - 71168
A: bootstat.dat - 67584
A: DtcInstall.log - 2790
A: explorer.exe - 2872320
A: fveupdate.exe - 15360
A: HelpPane.exe - 733696
A: hh.exe - 16896
A: mib.bin - 43131
A: msdfmap.ini - 1405
A: ODBC.INI - 405
A: PFRO.log - 2072
A: regedit.exe - 427008
A: ServerEnterprise.xml - 73424
A: ServerWeb.xml - 73395
A: setupact.log - 21027
A: setuperr.log - 0
A: splwow64.exe - 67072
A: system.ini - 219
A: TSSysprep.log - 1355
A: win.ini - 92
```

Exercise 14 Using an it-without-a-whose clause and including the version

In this exercise, you modify the Relevance query solution from Exercise 13, “Using an it-without-a-whose construct,” on page 2-15 to include the version for each file.

The objectives for this exercise are to complete these tasks:

- Use an **`if-then-else`** construct for validating file existence.
- Use an **`it-without-a-whose`** clause, which requires the **`it`** clause to be in parentheses.
- Build from the previous query.

  ```
  Q: (name of it & “ – “ & size of it as string) of files of windows folder
  ```

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. In the text box, create an **`if-then-else`** construct.

   ```
   if () then () else ()
   ```

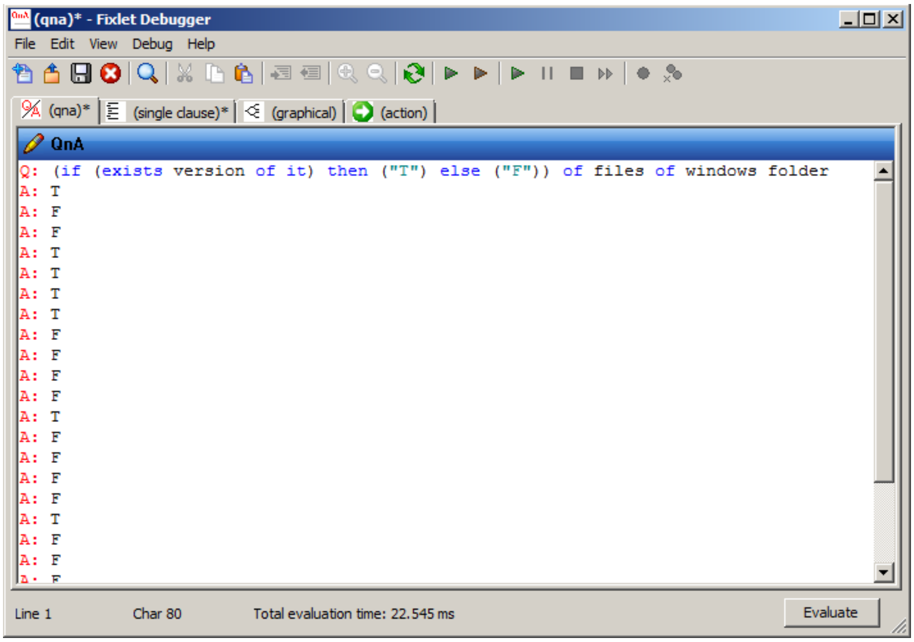
3. Place appropriate parentheses around the construct and append the queried designated objects: **of files of windows folder**.

   ```
   (if () then () else ()) of files of windows folder
   ```

4. Build out each portion of the **`if`** clause and evaluate each step.

   a. Perform a version existence check that outputs a Boolean state.

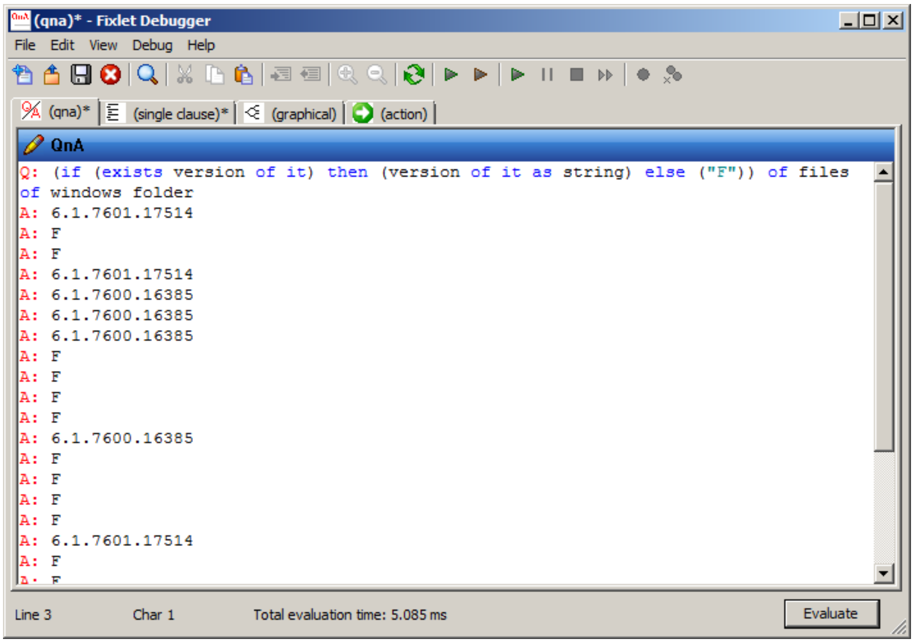
   ```
   Q: (if (exists version of it) then (“T”) else (“F”)) of files of windows folder
   ```



```
Q: (if (exists version of it) then ("T") else ("F")) of files of windows folder
A: T
A: F
A: F
A: T
A: T
A: T
A: T
A: F
A: F
A: F
A: F
A: T
A: F
A: F
A: F
A: F
A: T
A: F
A: F
A: F
```

   b. Replace the **True** Boolean state with the version value.

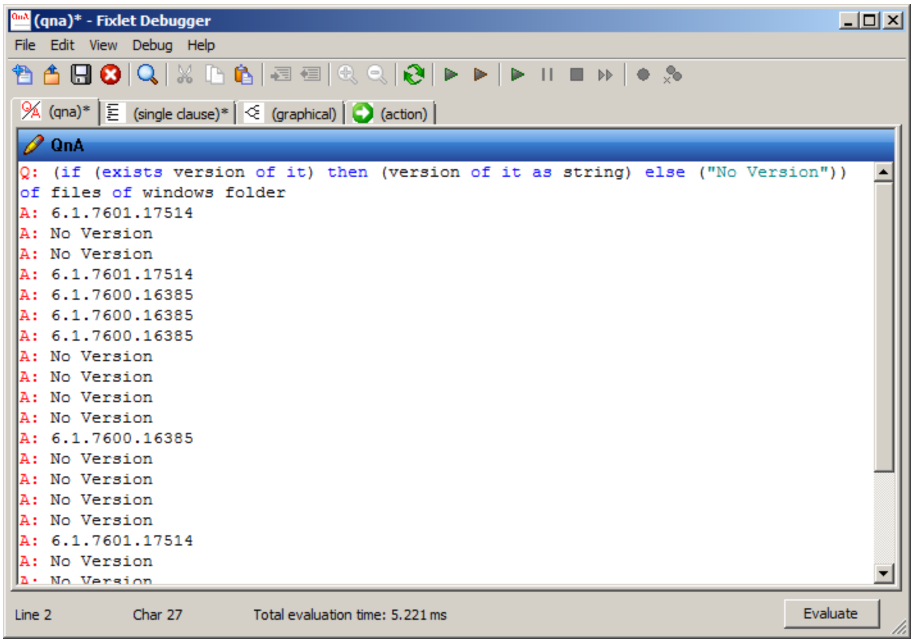
   ```
   Q: (if (exists version of it) then (version of it as string) else (“F”)) of
   files of windows folder
   ```



```
Q: (if (exists version of it) then (version of it as string) else ("F")) of files
of windows folder
A: 6.1.7601.17514
A: F
A: F
A: 6.1.7601.17514
A: 6.1.7600.16385
A: 6.1.7600.16385
A: 6.1.7600.16385
A: F
A: F
A: F
A: F
A: 6.1.7600.16385
A: F
A: F
A: F
A: F
A: 6.1.7601.17514
A: F
A: F
```

c. Replace the **False** Boolean state with a more meaningful string.

```
Q: (if (exists version of it) then (version of it as string) else (“No Version”)) of files of windows folder
```

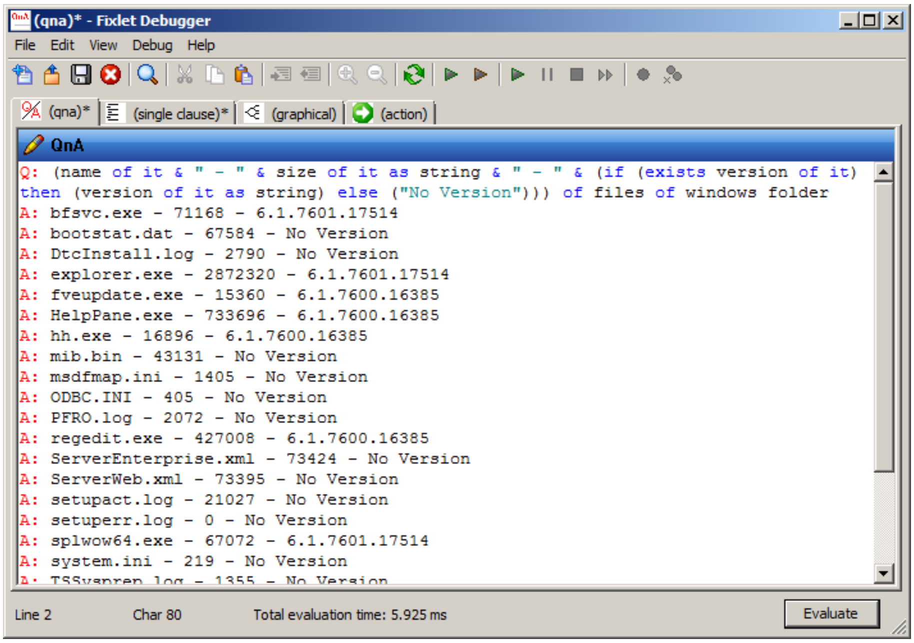


```
Q: (if (exists version of it) then (version of it as string) else ("No Version"))
of files of windows folder
A: 6.1.7601.17514
A: No Version
A: No Version
A: 6.1.7601.17514
A: 6.1.7600.16385
A: 6.1.7600.16385
A: 6.1.7600.16385
A: No Version
A: No Version
A: No Version
A: No Version
A: 6.1.7600.16385
A: No Version
A: No Version
A: No Version
A: No Version
A: 6.1.7601.17514
A: No Version
A: No Version
```

Line 2 Char 27 Total evaluation time: 5.221 ms

5. Finalize the solution by incorporating the previous string concatenation up to the version handler that you built.

```
Q: (name of it & “ – “ & size of it as string & “ – “ & (if (exists version of it) then (version of it as string) else (“No Version”))) of files of windows folder
```



```
Q: (name of it & " - " & size of it as string & " - " & (if (exists version of it)
then (version of it as string) else ("No Version"))) of files of windows folder
A: bfsvc.exe - 71168 - 6.1.7601.17514
A: bootstat.dat - 67584 - No Version
A: DtcInstall.log - 2790 - No Version
A: explorer.exe - 2872320 - 6.1.7601.17514
A: fveupdate.exe - 15360 - 6.1.7600.16385
A: HelpPane.exe - 733696 - 6.1.7600.16385
A: hh.exe - 16896 - 6.1.7600.16385
A: mib.bin - 43131 - No Version
A: msdfmap.ini - 1405 - No Version
A: ODBC.INI - 405 - No Version
A: PFRO.log - 2072 - No Version
A: regedit.exe - 427008 - 6.1.7600.16385
A: ServerEnterprise.xml - 73424 - No Version
A: ServerWeb.xml - 73395 - No Version
A: setupact.log - 21027 - No Version
A: setuperr.log - 0 - No Version
A: splwow64.exe - 67072 - 6.1.7601.17514
A: system.ini - 219 - No Version
A: TSSysprep.log - 1355 - No Version
```

Line 2 Char 80 Total evaluation time: 5.925 ms

Exercise 15 Determining the type of operating system

In this exercise, develop a Relevance query that returns **true** only if a computer has Win10 R2 with Service Pack 1.

Use the following Relevance constructs to complete this exercise:

- Use an **`it-without-a-whose`** clause so that you do not duplicate the phrase **`operating system`** in the query.
- Use a query syntax similar to the following example:

```
Q: (size of it > 35000 AND version of it < “6.0”) of file “mshtml.dll” of system
folder
```

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. In the text box, create a simple operating system query to validate the name.

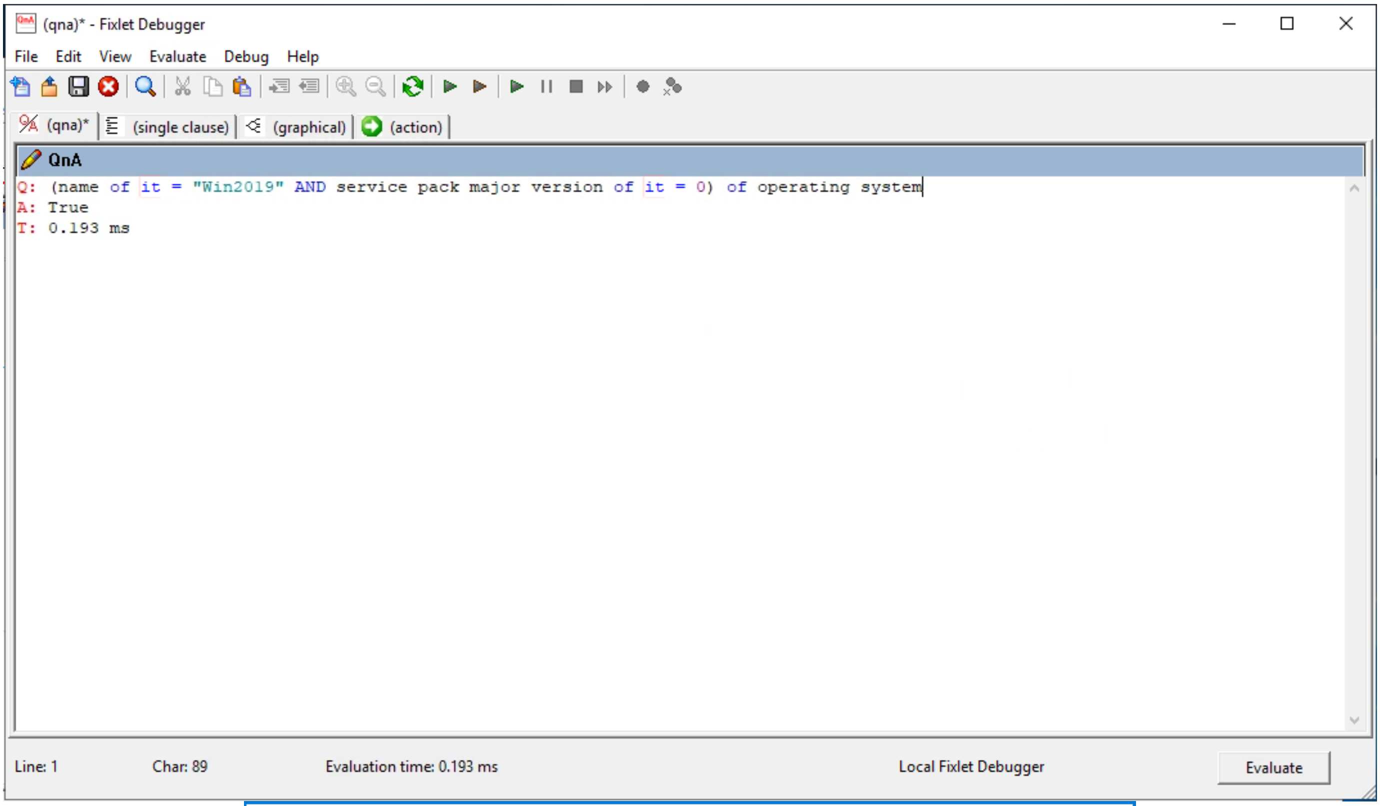
```
Q: name of operating system
```

3. In the text box, create a simple operating system query to validate the service pack.

```
Q: service pack major version of operating system
```

4. Combine the simple queries and compare to the static strings.

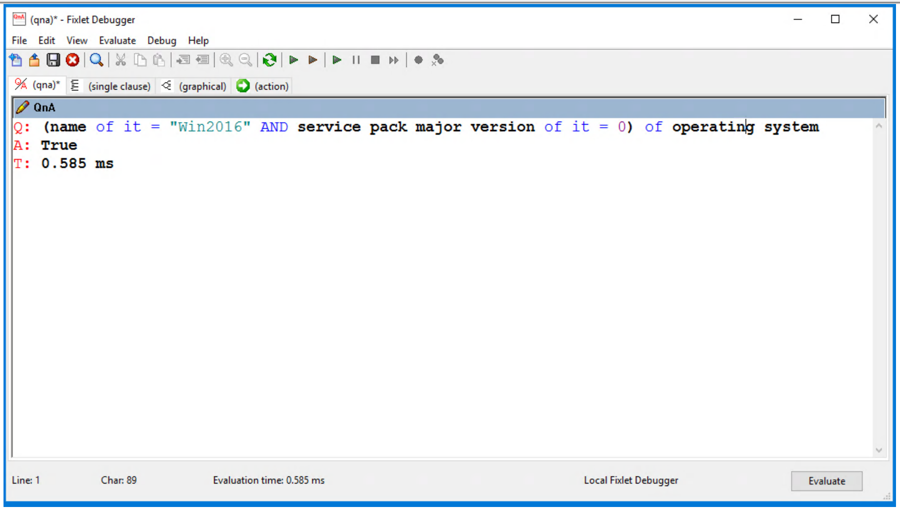
```
Q: (name of it = “Win2019” AND service pack major version of it = 0) of operating
system
```



(qna)* - Fixlet Debugger
File Edit View Evaluate Debug Help
(qna)* (single clause) (graphical) (action)
QnA

```
Q: (name of it = "Win2019" AND service pack major version of it = 0) of operating system
A: True
T: 0.193 ms
```

Line: 1
Char: 89
Evaluation time: 0.193 ms
Local Fixlet Debugger
Evaluate



(qna)* - Fixlet Debugger
File Edit View Evaluate Debug Help
(qna)* (single clause) (graphical) (action)
QnA

```
Q: (name of it = "Win2016" AND service pack major version of it = 0) of operating system
A: True
T: 0.585 ms
```

Line: 1
Char: 89
Evaluation time: 0.585 ms
Local Fixlet Debugger
Evaluate

You can also use this query:

```
Q: (name of it = “Win2008R2” AND csd version of it = “Service Pack 1”) of
operating system
```

Unit 3 Common Objects Exercises

You perform these exercises by using the BigFix Fixlet Debugger. This tool is installed on the **BESFNDWIN10** virtual machine. To start the tool, click **Start > BigFix > BigFix Fixlet Debugger**.

When you enter the code in the Fixlet Debugger on the query and answer **(qna)** tab, you must preface it with a **Q:** to designate it as a query. You designate the answer in the return code with an **A:**, designate the time it took to process the query with a **T:**, and if an error occurs, designate it with an **E:**.

Exercise 16 Starting the environment

Perform the following steps to start the virtual machines in the lab environment if they are not already started.

1. Verify that your student virtual machine is running.
2. Switch to the student virtual machine. If you are logged off, log in to the machine with your assigned username and password.
3. Click **Start > > BigFix > BigFix Fixlet Debugger**. The BigFix Fixlet Debugger opens.

Exercise 17 Determining the percentage of free space

In this exercise, you develop a Relevance query that determines the system drive and evaluates the overall percentage of free space available on that drive.

The formula *`<amount of free space> * 100 / <size of drive>`* provides the percentage of free space available. The result should be shown like the following example:

```
20 percent free on System Drive
```

Use the Fixlet Debugger to perform the following steps:

1. In the Fixlet Debugger, select the **(qna)** tab.

2. Return the drive letter for the drive that maintains the system folder.

   ```
   Q: name of drive of system folder
   ```

3. Calculate the percentage of free space.

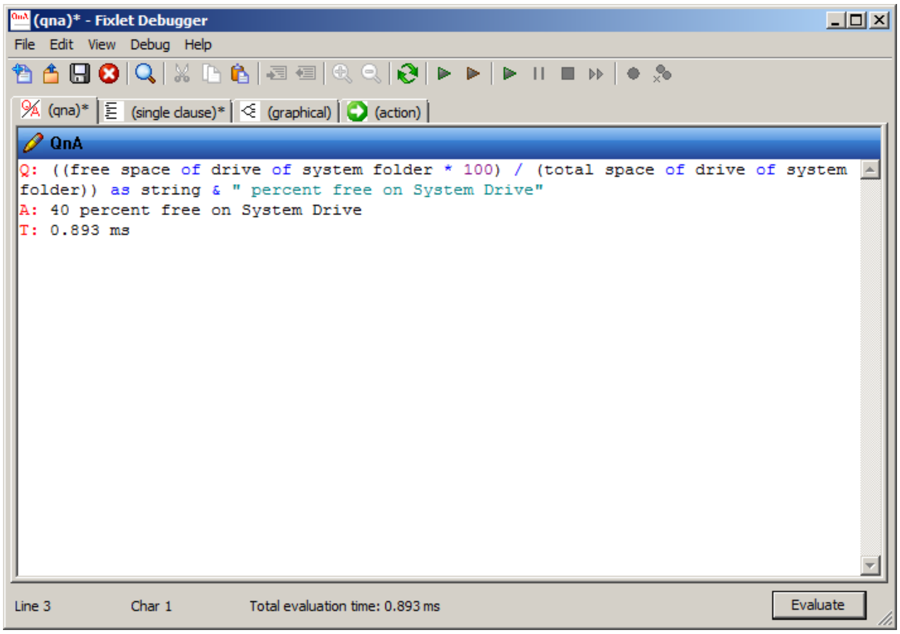
   ```
   Q: free space of drive of system folder * 100 / total space of drive of system
   folder
   ```

4. Convert the result to a string for string concatenation.

   ```
   Q: (free space of drive of system folder * 100 / total space of drive of system
   folder) as string
   ```

5. Use string concatenation to format the output.

   ```
   Q: ((free space of drive of system folder * 100) / (total space of drive of
   system folder)) as string & “ percent free on System Drive”
   ```



```
Q: ((free space of drive of system folder * 100) / (total space of drive of system
folder)) as string & " percent free on System Drive"
A: 40 percent free on System Drive
T: 0.893 ms
```

Exercise 18 Determining the free space on all drives

In this exercise, you develop a Relevance query that returns the free space that is available on all hard disks for the endpoint. Please complete Exercise 2 before attempting this exercise.

Use the Inspector Library to determine the following information:

- Free space of the drive
- Name of the drive
- Type of the drive as hard drive (fixed drive)

Use the formula *`1023/1024/1024`* to determine the size in GB. Calculation of free space is *`<size of drive> / <GB calculation>`*. Use a **`whose-it`** clause for filtering and an **`it-without-a-whose`** clause to provide the output. To concatenate strings, use the ampersand (&) character between strings.

Display the output in the following format:

```
C: - 32GB
D: - 10GB
```

Use the Fixlet Debugger to perform the following steps:

1. In the Fixlet Debugger, select the **(qna)** tab.
2. Create a query to return the drive letters for all drives.

   ```
   Q: names of drives
   ```

3. Filter for the hard disks only.

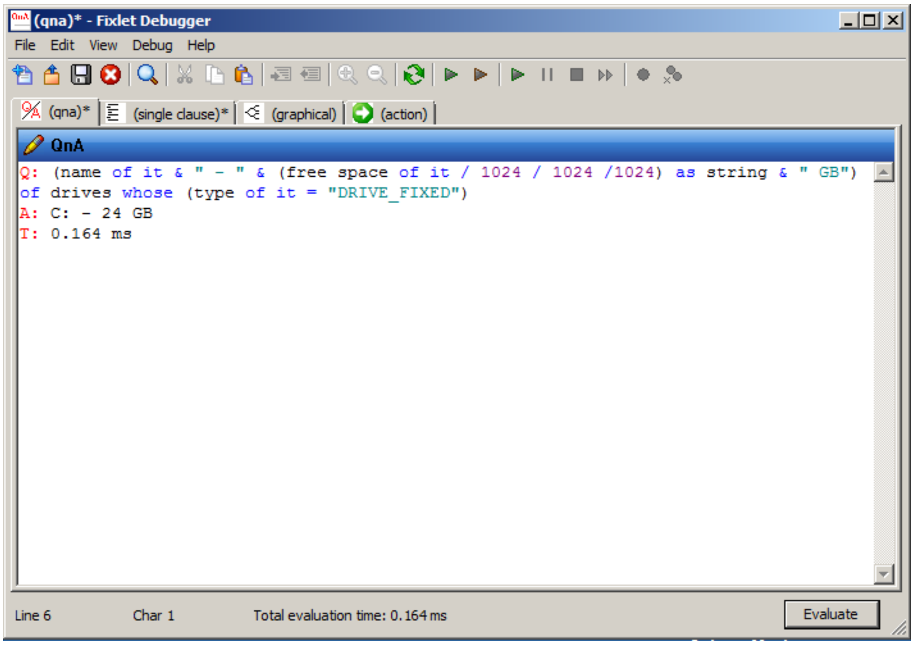
   ```
   Q: names of drives whose (type of it = “DRIVE_FIXED”)
   ```

4. Use an **`it-without-a-whose`** clause to produce the results.

   ```
   Q: (name of it) of drives whose (type of it = “DRIVE_FIXED”)
   ```

5. Using string concatenation, format the output. Ensure that you convert the Relevance query of Step 4 to a string type.

```
Q: (name of it & “ - “ & (free space of it / 1024 / 1024 /1024) as string & “ GB”)
of drives whose (type of it = “DRIVE_FIXED”)
```



```
Q: (name of it & " - " & (free space of it / 1024 / 1024 /1024) as string & " GB")
of drives whose (type of it = "DRIVE_FIXED")
A: C: - 24 GB
T: 0.164 ms
```

Exercise 19 Determining the system architecture

In this exercise, you develop a Relevance query to determine the architecture of a system.

Your Relevance query must perform the following operations.

- It must run on any Windows or non-Windows platform.
- It must use order of processing to avoid errors on Windows systems.
- You must display the output as **x64 Architecture** or **x86 Architecture**.

In order to accomplish this task, use nested **`if-then-else`** clauses.

Beginning with BigFix version 8.2, the **`architecture of <operating system>`** inspector works on all operating systems. In prior versions, Windows clients had to be evaluated using the **`x64 of <operating system>`** inspector.

Windows and Apple operating systems report the following information when using **`architecture of <operating system>`** inspector:

i386 for a 32-bit kernel

x86_64 for a 64-bit kernel

Linux operating systems report the following information when using the **`architecture of <operating system>`**inspector:

i686 for a 32-bit kernel

x86_64 for a 64-bit kernel

In this lab, you must check the version of the BigFix client with an **`if-then-else`**clause. If the operating system is non-Windows, you can automatically use the **`architecture of <operating system>`** inspector. If the operating system is Windows, you must first determine whether the BigFix client is version 8.2 or later so that you know which inspector to use to retrieve the architecture.

The structure of this lab is as follows:

```
if(Operating System is Non-Windows OR (Operating System is Windows AND the
version of the client is >= 8.2))
then
if (Operating System is 64-bit)
then (“x64 Architecture”)
else(“x86 Architecture”)
else
if(Operating System is 64-bit)
then (“x64 Architecture”)
else (“x86 Architecture”)
```

Use the Fixlet Debugger to perform the following steps:

1. In the Fixlet Debugger, select the **(qna)**tab.

2. Begin by creating the required **`if-then-else`**clause. Always use parentheses around each component of the **`if-then-else`** clause to ensure that the Relevance engine performs the correct evaluation.

   ```
   Q: if () then () else ()
   ```

3. In the **if** portion of the clause, you determine whether the operating system is non-Windows. If the operating system is Windows then you determine whether the version of the BigFix client is greater than or equal to version 8.2.

   a. On a new line in the **QnA** tab, create the clause to test for a non-Windows operating system.

      ```
      Q: name of operating system does not start with “Win”
      ```

   b. On a new line in the **QnA** tab, create the clause to test if the operating system is Windows.

      ```
      Q: name of operating system starts with “Win”
      ```

   c. On a new line in the **QnA** tab, create the clause to test that the version of the client is >= 8.2.

      ```
      Q: version of client >= “8.2”
      ```

d. Combine the query of Step 3b and Step 3c together by using AND. This query tests whether the name of the operating system is Windows and that the version of the client is >= 8.2.

```
Q: name of operating system starts with “Win” AND version of client >= “8.2”
```

e. Copy the clauses from Step 3a and Step 3d into the **if** clause that you created in Step 2. Remember to put parentheses around the clause you created in Step 3d to force the clause to test both conditions as one.

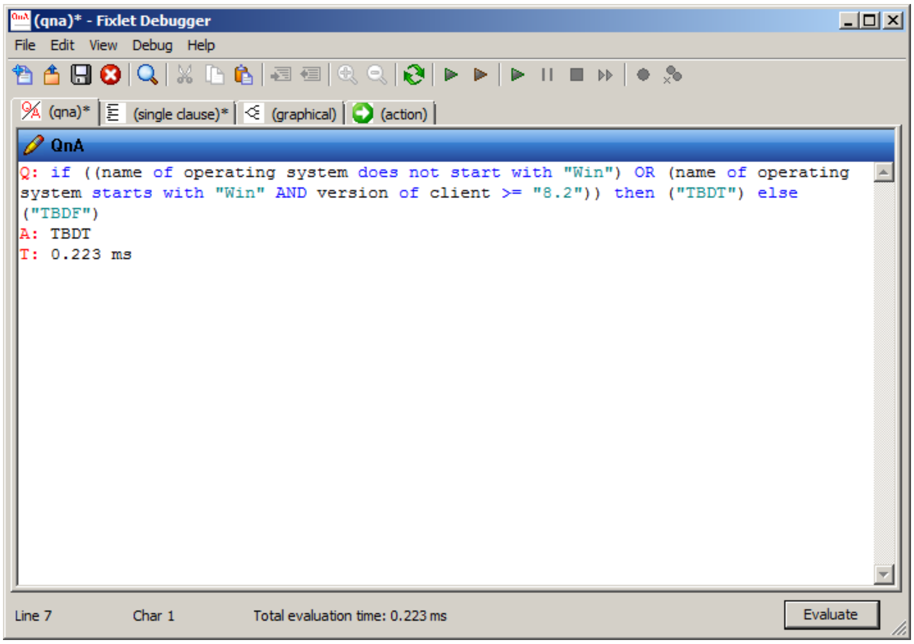
```
Q: if ((name of operating system does not start with “Win”) OR (name of
operating system starts with “Win” AND version of client >= “8.2”)) then ()
else ()
```

f. For the **then** clause, use a test string such as **TBDT**.

g. For the **else** clause, use a test string such as **TBDF**.

```
Q: if ((name of operating system does not start with “Win”) OR (name of
operating system starts with “Win” AND version of client >= “8.2”)) then
(“TBDT”) else (“TBDF”)
```

h. Evaluate the query to ensure that it works as expected.



```
Q: if ((name of operating system does not start with "Win") OR (name of operating
system starts with "Win" AND version of client >= "8.2")) then ("TBDT") else
("TBDF")
A: TBDT
T: 0.223 ms
```

4. Open a new **QnA** tab by selecting **File > New Tab > New QnA Tab**.

5. Build the nested **if-then-else** clause to report the operating system type for the **then** clause that you created in Step 3. You replace the then test statement that you created in Step 3 with this nested **if-then-else** clause.

a. Begin by creating the required **if-then-else** clause. Always use parentheses around each component of the **if-then-else** clause to ensure that the Relevance engine performs the correct evaluation.

```
Q: if () then () else ()
```

b. The **if** clause uses the **architecture of <operating system>** inspector to determine the architecture of the operating system. The return values of this inspector are shown at the beginning of this exercise. Using this inspector, you test to see whether the architecture of the operating system returns 64.

```
Q: if (architecture of operating system contains “64”) then () else ()
```

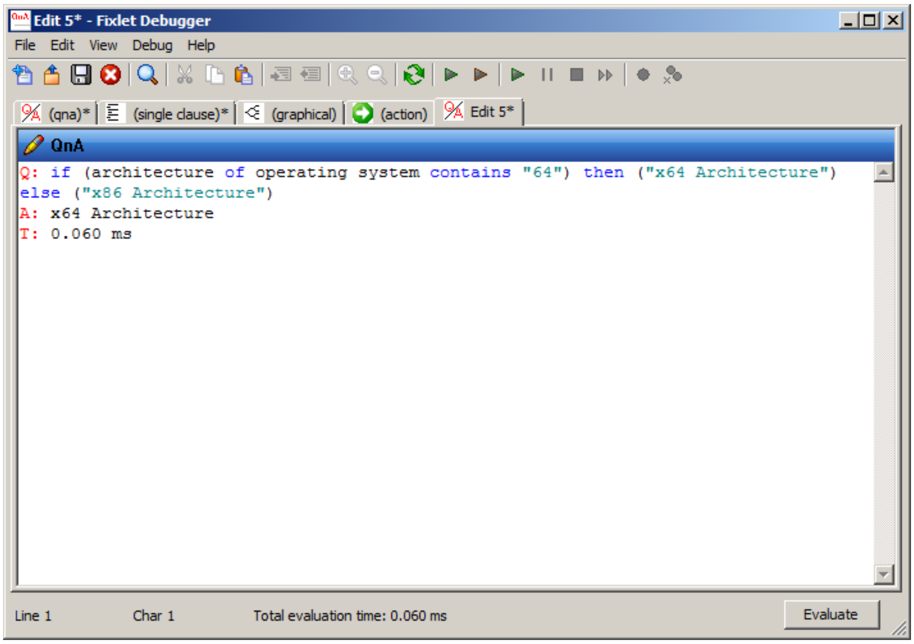
c. Modify the **then**clause to format the output string as **x64 Architecture**.

```
Q: if (architecture of operating system contains “64”) then (“x64
Architecture”) else ()
```

d. Modify the **else** clause toformat the output string as **x86 Architecture**.

```
Q: if (architecture of operating system contains “64”) then (“x64
Architecture”) else (“x86 Architecture”)
```

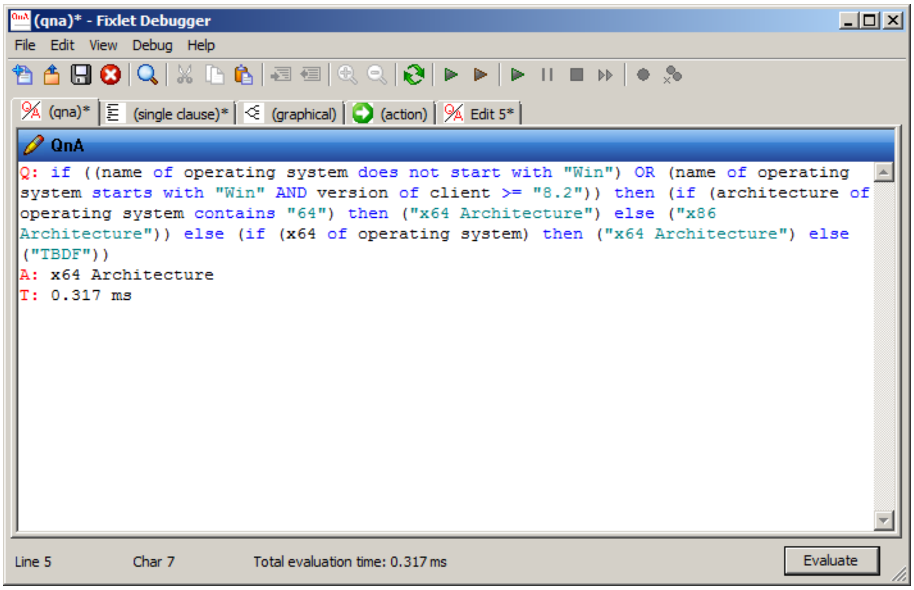
e. Evaluate the query. You should see the string that corresponds to the architecture of your system displayed on a client that is greater than or equal to version 8.2.



```
Q: if (architecture of operating system contains "64") then ("x64 Architecture")
else ("x86 Architecture")
A: x64 Architecture
T: 0.060 ms
```

6. You now have the **then**clause for the **if-then-else** statement that you created in Step 3. Copy and paste this clause over the test string of the **then**clause that you created in Step 3.

7. Evaluate the statement. If you are running this query on a system with the **client >= 8.2**, you should see the string corresponding to your architecture. If you are running on a system with a **client < 8.2**, you see the **`else`** statement.



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action) Edit 5*
QnA

```
Q: if ((name of operating system does not start with "Win") OR (name of operating
system starts with "Win" AND version of client >= "8.2")) then (if (architecture of
operating system contains "64") then ("x64 Architecture") else ("x86
Architecture")) else (if (x64 of operating system) then ("x64 Architecture") else
("TBDF"))
A: x64 Architecture
T: 0.317 ms
```

Line 5
Char 7
Total evaluation time: 0.317 ms
Evaluate

8. Build the nested **`if-then-else`** clause for a Windows system that is running a client where the version is earlier than 8.2. This action replaces the `else` test string of the statement that you created in Step 7.

   a. Select the **QnA** tab that you used in Step 5.

   b. Begin by creating the required **`if-then-else`** clause. Always use parentheses around each component of the **`if-then-else`** clause to ensure that the Relevance engine performs the correct evaluation. You can overwrite the previous query or start on a new line.

   ```
   Q: if () then () else ()
   ```

   c. The **`if`** clause uses the **`architecture of <operating system>`** inspector to determine the architecture of the operating system. Look up this inspector in the Inspector Library and note that it returns a Boolean type.

   ```
   Q: if (x64 of operating system) then () else ()
   ```

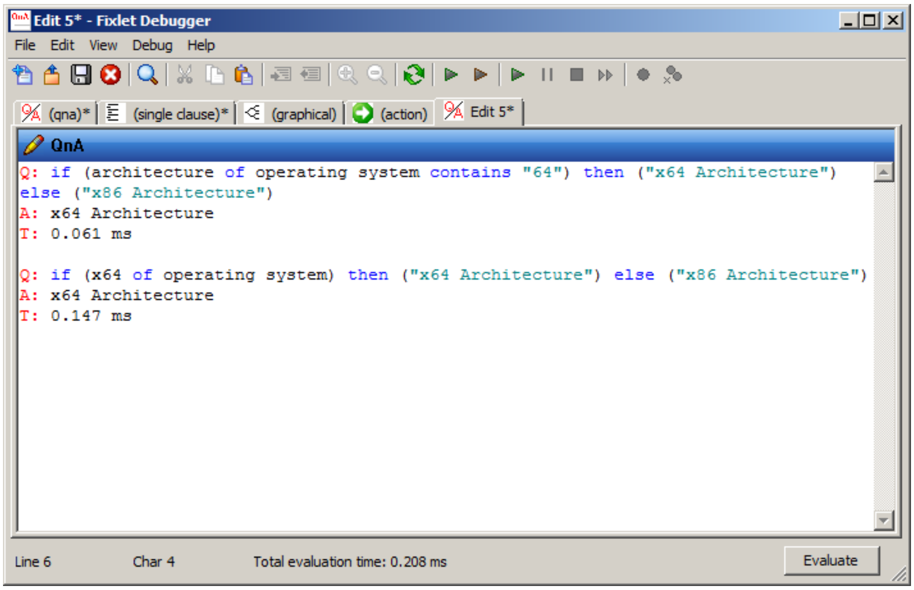
   d. Format the output of the **`then`** clause to return the string **x64 Architecture**.

   ```
   Q: if (x64 of operating system) then (“x64 Architecture”) else ()
   ```

   e. Format the output of the **`else`** clause to return the string **x86 Architecture**.

   ```
   Q: if (x64 of operating system) then (“x64 Architecture”) else (“x86
   Architecture”)
   ```

f. Evaluate the query. You should see the string that corresponds to the architecture of your system displayed on a client that is greater than or equal to version 8.2.

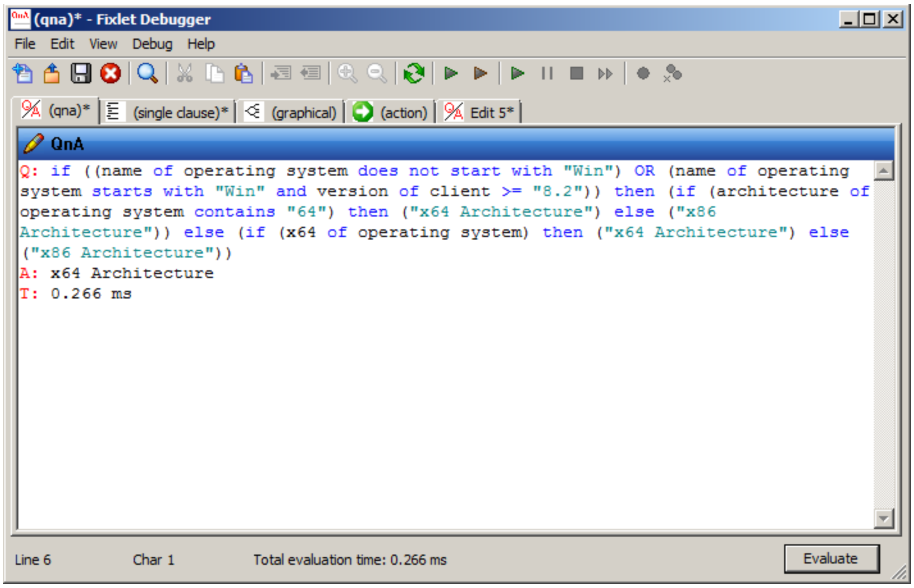


```
Q: if (architecture of operating system contains "64") then ("x64 Architecture")
else ("x86 Architecture")
A: x64 Architecture
T: 0.061 ms

Q: if (x64 of operating system) then ("x64 Architecture") else ("x86 Architecture")
A: x64 Architecture
T: 0.147 ms
```

9. You now have the **else** clause for the **if-then-else** statement you created in Step 7. Copy and paste this clause over the test string of the **else** clause that you created in Step 7.

```
Q: if ((name of operating system does not start with “Win”) OR (name of
operating system starts with “Win” and version of client >= “8.2”)) then (if
(architecture of operating system contains “64”) then (“x64 Architecture”)
else (“x86 Architecture”)) else (if (x64 of operating system) then (“x64
Architecture”) else (“x86 Architecture”))
```

10. Evaluate the statement. This expression now works on all operating systems in your environment.



```
Q: if ((name of operating system does not start with "Win") OR (name of operating
system starts with "Win" and version of client >= "8.2")) then (if (architecture of
operating system contains "64") then ("x64 Architecture") else ("x86
Architecture")) else (if (x64 of operating system) then ("x64 Architecture") else
("x86 Architecture"))
A: x64 Architecture
T: 0.266 ms
```

Exercise 20 Locating a file

In this exercise, you develop a Relevance query to find the version of a Mozilla Firefox file named `nss3.dll`. You use the folder location of a known file to provide the folder location for the target file. The file `nss3.dll` is located in the same folder as the registered application for Firefox: **firefox.exe**.

Use the **`parent folder of <regapp>`** inspector to create the <folder> object corresponding to the folder that contains both `firefox.exe` and `nss3.dll`.

Use a query syntax similar to the following one:

```
Q: exists file “AdInspectHelper.dll” of parent folder of running application
“qna.exe”
```

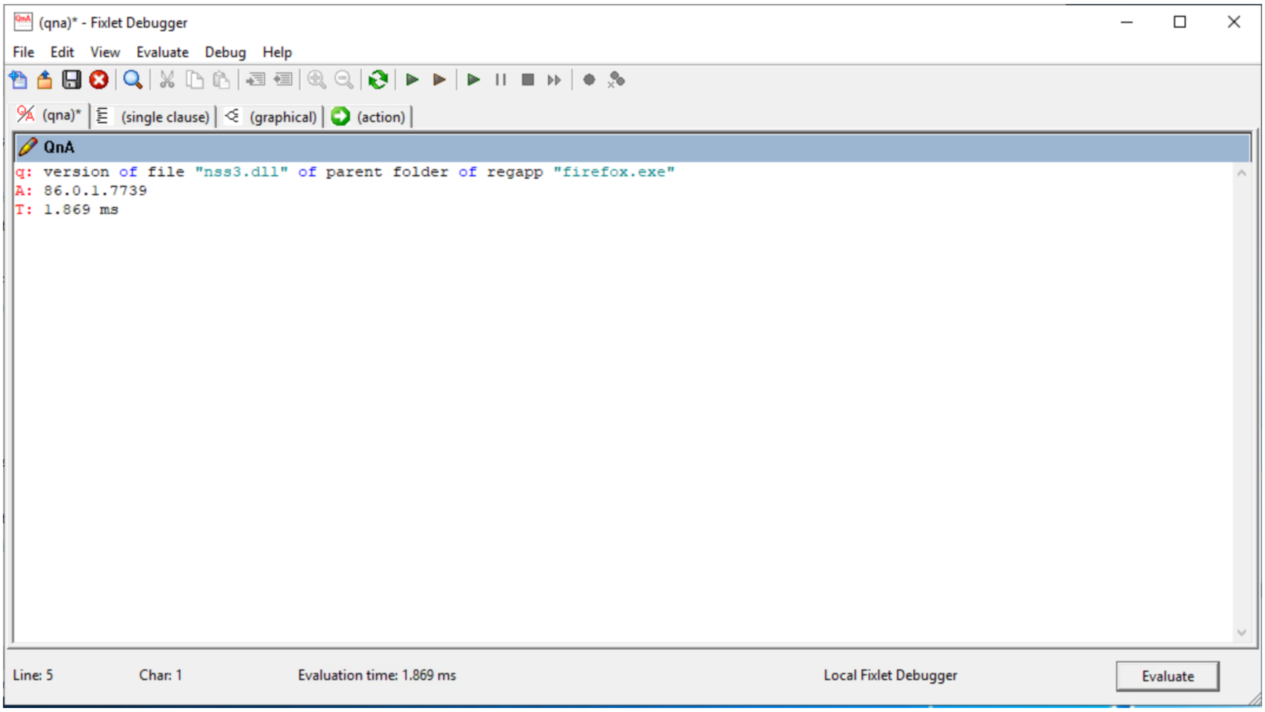
Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. In the text box, create the query by checking the version of the file. An error is returned if you attempt to evaluate the query because the file location is not specified.

   ```
   Q: version of file“nss3.dll”
   ```

3. Complete the query by specifying the file location in relation to the known registered program **firefox.exe** and click **Evaluate**.

   ```
   Q: version of file “nss3.dll” of parent folder of regapp “firefox.exe”
   ```



(qna)* - Fixlet Debugger
File Edit View Evaluate Debug Help
(qna)* (single clause) (graphical) (action)
QnA
Line: 5 Char: 1 Evaluation time: 1.869 ms Local Fixlet Debugger
Evaluate

```
q: version of file "nss3.dll" of parent folder of regapp "firefox.exe"
A: 86.0.1.7739
T: 1.869 ms
```

You can also use this query:

```
Q: version of file (preceding text of last "\" of (pathname of regapp
"firefox.exe") & "\nss3.dll")
```

Note:

- This statement creates the whole path name of the file by parsing the path name of the regapp application. Then, it uses file <string> to create the file object.
- The parentheses around the string are required; they delineate the file object.
- The file object becomes **"C:\Program Files\Mozilla Firefox\nss3.dll"** when fully expanded.

Exercise 21 Using address of network inspector

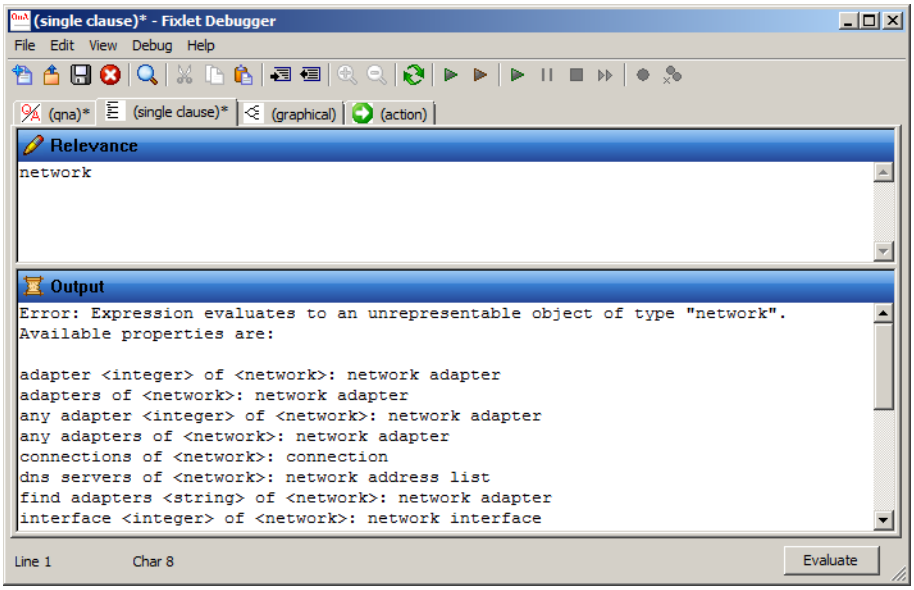
In this exercise, you use a Windows-only inspector, **`address of <network address list>`**, to determine the IP address of the system.

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(single clause)** tab.

2. Enter `network` in the Relevance window and evaluate the query.

   This query results in an error. However, in the **(single clause)** tab, the Fixlet Debugger displays potential properties that are available for this object. Review the various properties. Note the **`adapters of <network>: network adapter`** property.

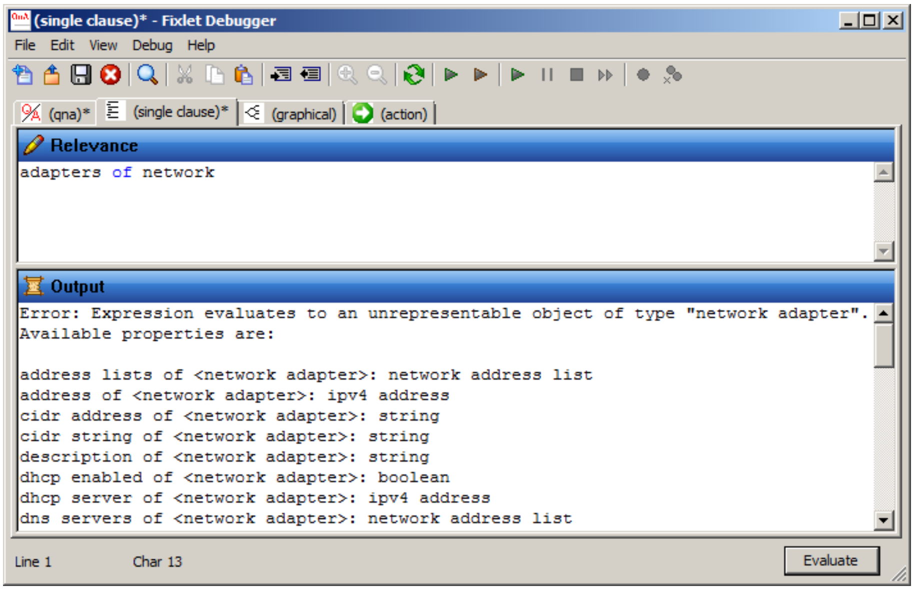


```
network
```

```
Error: Expression evaluates to an unrepresentable object of type "network".
Available properties are:

adapter <integer> of <network>: network adapter
adapters of <network>: network adapter
any adapter <integer> of <network>: network adapter
any adapters of <network>: network adapter
connections of <network>: connection
dns servers of <network>: network address list
find adapters <string> of <network>: network adapter
interface <integer> of <network>: network interface
```

3. In the Relevance window, replace `network` with `adapters of network`. Click **Evaluate**.

The Fixlet Debugger displays an error along with the potential properties for this inspector. Review the various properties. Note the **`address of <network adapter>: ipv4 address`** property.

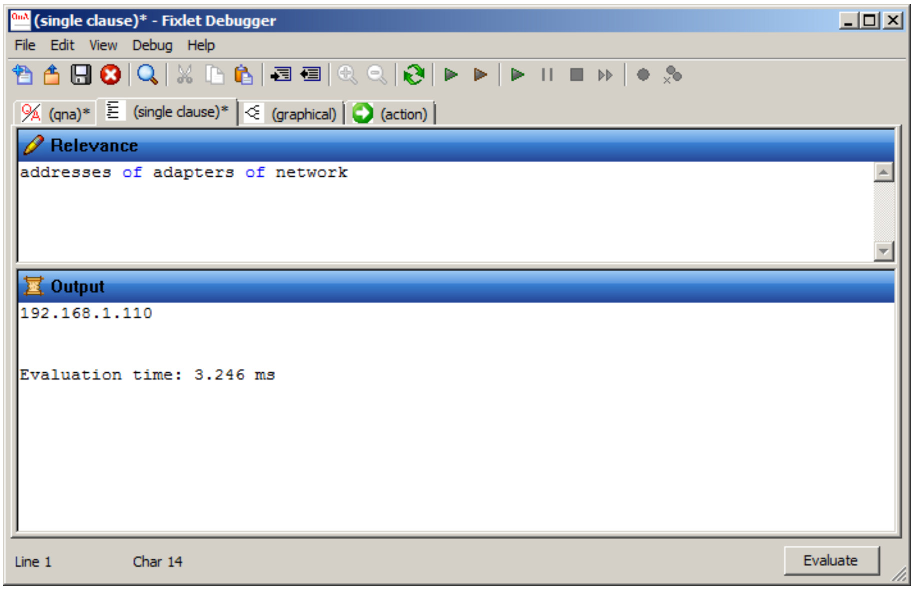


```
adapters of network
```

```
Error: Expression evaluates to an unrepresentable object of type "network adapter".
Available properties are:

address lists of <network adapter>: network address list
address of <network adapter>: ipv4 address
cidr address of <network adapter>: string
cidr string of <network adapter>: string
description of <network adapter>: string
dhcp enabled of <network adapter>: boolean
dhcp server of <network adapter>: ipv4 address
dns servers of <network adapter>: network address list
```

4. In the Relevance pane, enter `addresses of adapters of network`. Click **Evaluate**.

   The Fixlet Debugger now displays the addresses for all the networks within your system.



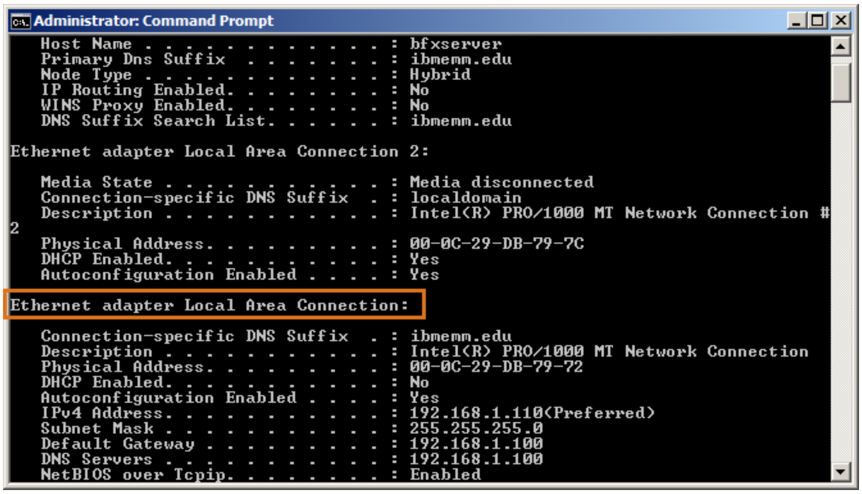
```
addresses of adapters of network
```

```
192.168.1.110


Evaluation time: 3.246 ms
```

5. To understand this query, open a Command Prompt by double-clicking the **Command Prompt** icon on the desktop.

6. In the Command Prompt window, enter `ipconfig /all` and press Enter.

If required, scroll up and down through the Command Prompt window to locate the occurrences of **Ethernet Adapter**. Each connected network adapter corresponds to the **`adapters of <network>`** inspector.



```
   Host Name . . . . . . . . . . . . : bfxserver
   Primary Dns Suffix  . . . . . . . : ibmemm.edu
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : ibmemm.edu

Ethernet adapter Local Area Connection 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : localdomain
   Description . . . . . . . . . . . : Intel(R) PRO/1000 MT Network Connection #
2
   Physical Address. . . . . . . . . : 00-0C-29-DB-79-7C
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . : ibmemm.edu
   Description . . . . . . . . . . . : Intel(R) PRO/1000 MT Network Connection
   Physical Address. . . . . . . . . : 00-0C-29-DB-79-72
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 192.168.1.110(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.100
   DNS Servers . . . . . . . . . . . : 192.168.1.100
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

7. In the Inspector Library, locate the **Network Adapter** inspector. Review each property and note that these properties correspond to the values displayed by the `ipconfig /all` command.

8. Close the Command Prompt window.

Exercise 22 Interrogating network adapters

In this exercise, you interrogate the network adapters by developing a Relevance query that evaluates to Boolean True if the last octet of the operating system IP address is an even number.

The objectives for this exercise are to perform the following tasks:

- Use the `following text of last "." of <network address>` syntax to obtain the last octet of the network address.
- Use `mod` to determine whether the address is even or odd.
  - 4 mod 2 = 0 (even)
  - 5 mod 2 = 1 (odd)

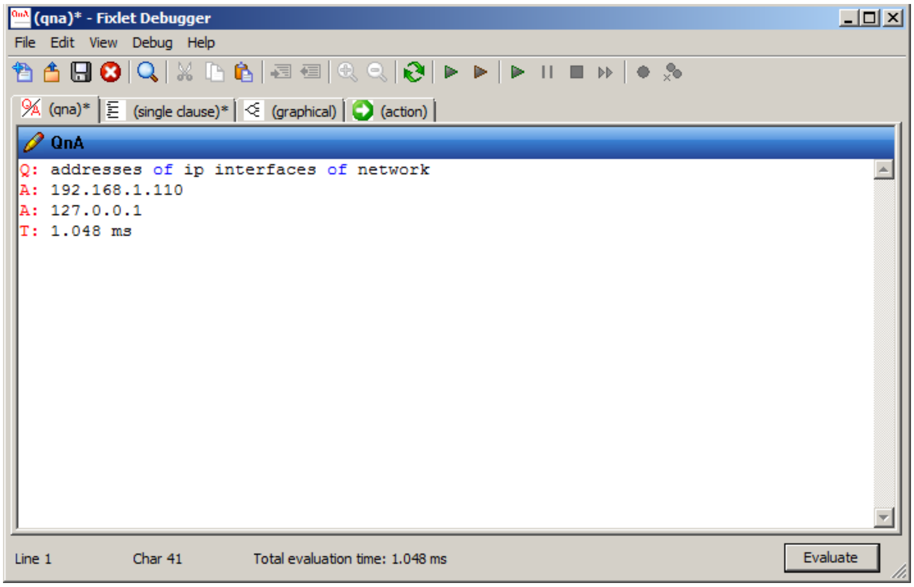
You use the **`ip interface of <network>`** inspector for this exercise. This object returns the IP address for any operating system supported by the BigFix client.

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. In the text box, create the query by obtaining the addresses of network adapters in the system.

   `Q: addresses of ip interfaces of network`



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA

```
Q: addresses of ip interfaces of network
A: 192.168.1.110
A: 127.0.0.1
T: 1.048 ms
```

Line 1
Char 41
Total evaluation time: 1.048 ms
Evaluate

   At least two IP addresses are listed. The first address is the IP of the adapter while the second is the loopback adapter. If you have multiple networks, additional IP addresses are listed.

3. Modify the query to convert the results to a string data type to perform search and string manipulation.

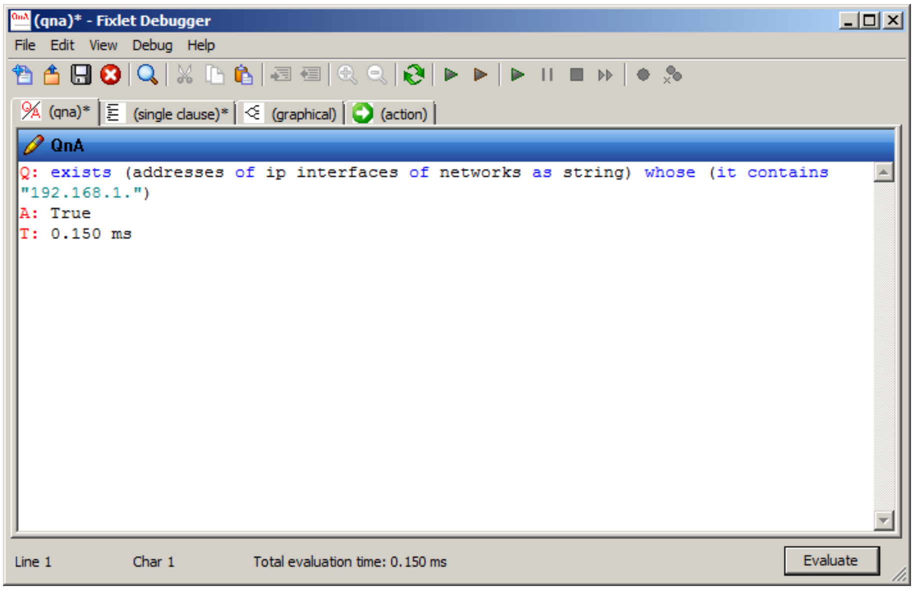
   `Q: addresses of ip interfaces of network as string`

4. Use a **`whose-it`** clause to retrieve the required IP address.

   `Q:(addresses of ip interfaces of networks as string) whose (it contains “10.0.0.”)`

5. Ensure that an address on the subnet **10.0.0.*** exists by checking existence in a **`whose-it`** construct.

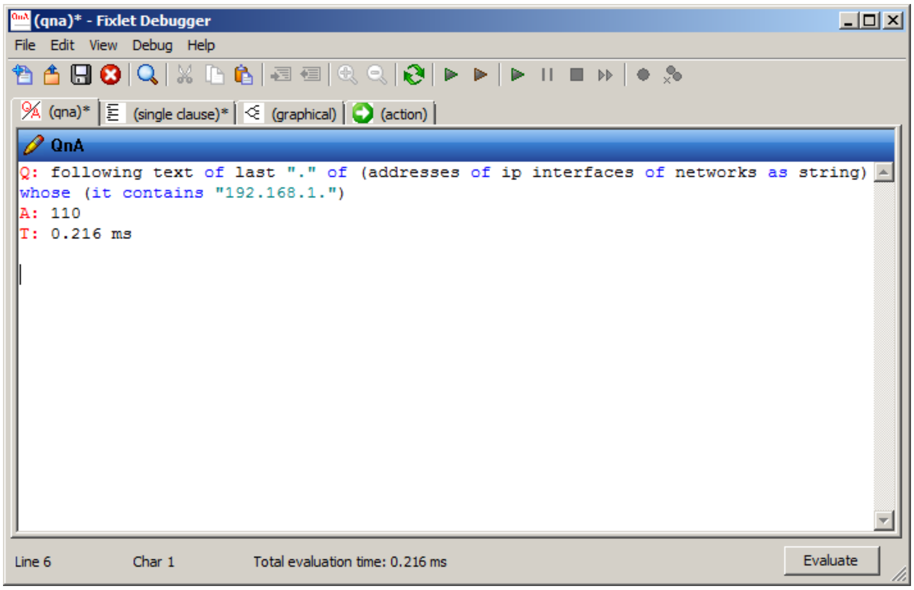
   Q: exists (addresses of ip interfaces of networks as string) whose (it contains "10.0.0.")


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA
Line 1 Char 1 Total evaluation time: 0.150 ms
Evaluate

```
Q: exists (addresses of ip interfaces of networks as string) whose (it contains
"192.168.1.")
A: True
T: 0.150 ms
```

6. Query the last octet from the identified subnet address.

   Q: following text of last "." of (addresses of ip interfaces of networks as string) whose (it contains"10.0.0.")


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA
Line 6 Char 1 Total evaluation time: 0.216 ms
Evaluate

```
Q: following text of last "." of (addresses of ip interfaces of networks as string)
whose (it contains "192.168.1.")
A: 110
T: 0.216 ms
```

7. Convert the result to an integer to use the **`mod`** operation. Use parentheses carefully.

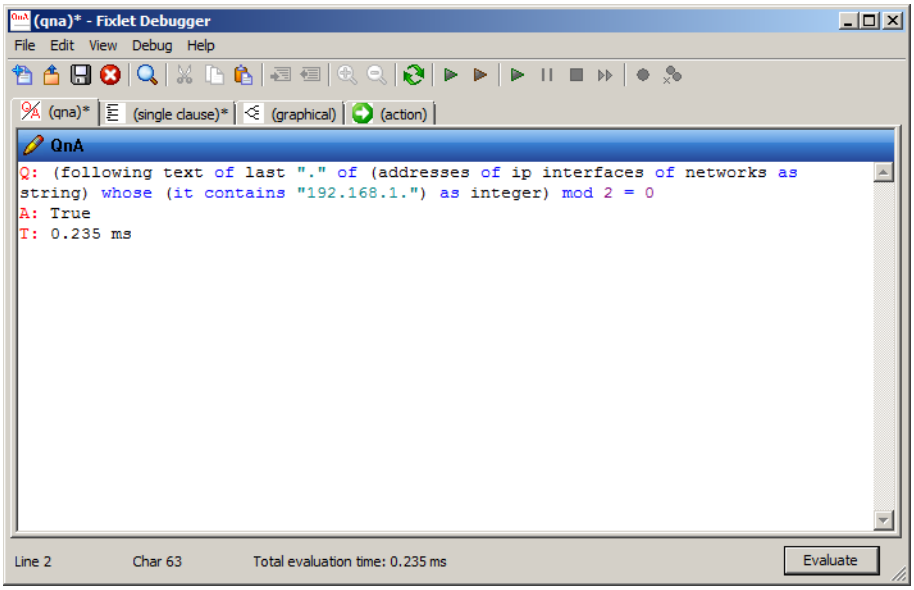
   Q: (following text of last "." of (addresses of ip interfaces of networks as string) whose (it contains "10.0.0.") as integer)

8. Apply the **mod** operation to the overall numeric result. The **mod** operator returns only the value of 0 for even results or 1 for odd results.

```
Q: (following text of last “.” of (addresses of ip interfaces of networks as string) whose (it contains “10.0.0.”) as integer) mod 2
```

9. Convert to Boolean by performing a value comparison to determine whether the address is even or odd.

```
Q: (following text of last “.” of (addresses of ip interfaces of networks as string) whose (it contains “10.0.0.”) as integer) mod 2 = 0
```



```
Q: (following text of last "." of (addresses of ip interfaces of networks as
string) whose (it contains "192.168.1.") as integer) mod 2 = 0
A: True
T: 0.235 ms
```

Exercise 23 Inspecting the environment

In this exercise, you develop a Relevance query to determine the modification time of the `actionsite.afxm` file by using the **ProgramFiles(x86)** environment variable.

NOTE: There are no spaces in this variable

The objective for this exercise is to use an environment variable object to find the path of the Program Files directory. The file is in the following folder:

```
C:\Program Files (x86)\BigFix Enterprise\BES Client\__BESData\actionsite
```

For this exercise, you use the **variables of environment** inspector to determine the **ProgramFiles(x86)** environment variable:

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. Begin by locating the BigFix client with the value of the **ProgramFiles(x86)** environment variable.

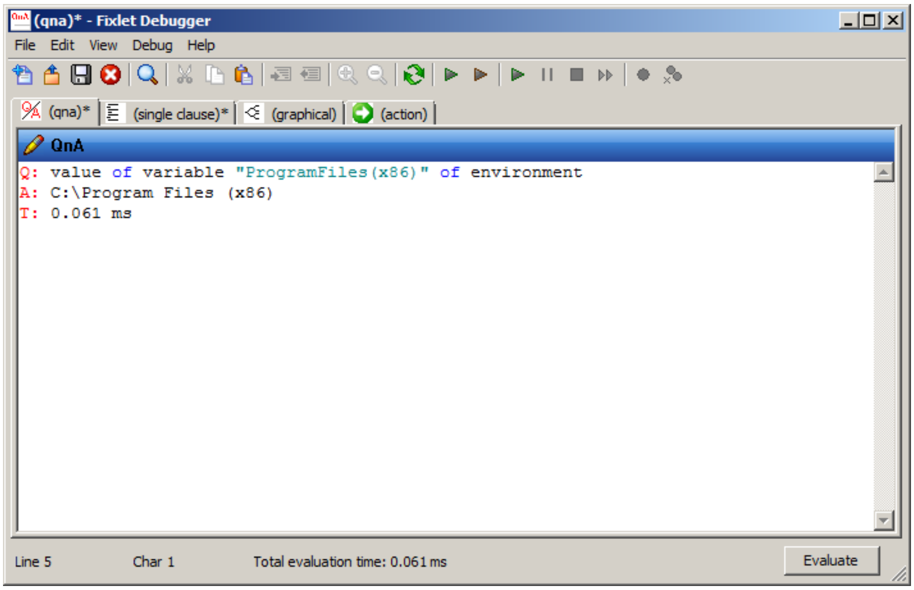
Q: variables of environment

3. Locate the desired variable and copy into a modified command.

   Q: variables “ProgramFiles(x86)” of environment

4. To obtain only the value of the query, use the **value of** inspector.

   Q: value of variables “ProgramFiles(x86)” of environment

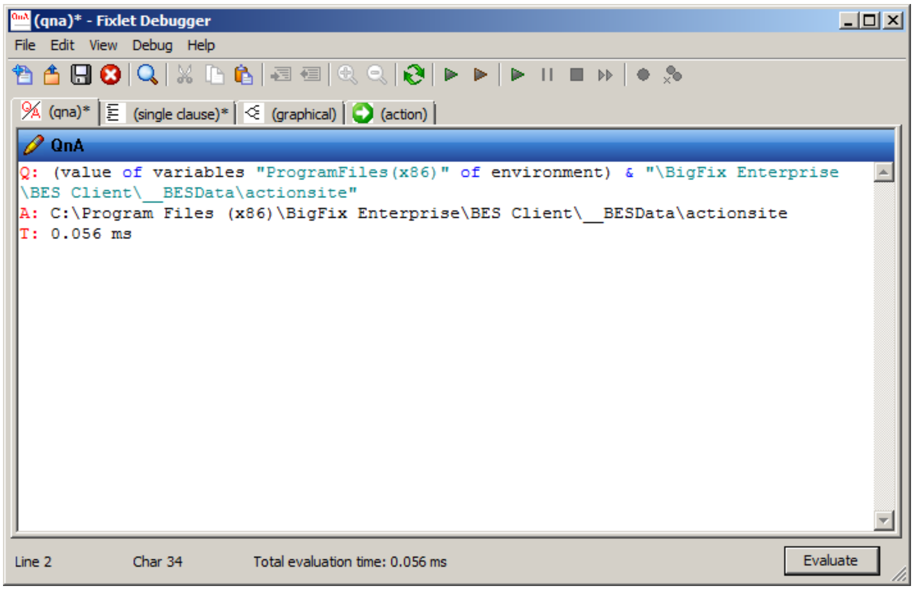

(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA
Evaluate

```
Q: value of variable "ProgramFiles(x86)" of environment
A: C:\Program Files (x86)
T: 0.061 ms
```

Line 5 Char 1 Total evaluation time: 0.061 ms

5. Now you use string concatenation to build the full path for the folder that contains the actionsite.afxm file. Use parentheses to clearly delineate the query portion of the query from the hardcoded string portion.

   Q: (value of variables “ProgramFiles(x86)” of environment) & “\BigFix Enterprise\BES Client__BESData\actionsite”


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA
Evaluate

```
Q: (value of variables "ProgramFiles(x86)" of environment) & "\BigFix Enterprise
\BES Client\__BESData\actionsite"
A: C:\Program Files (x86)\BigFix Enterprise\BES Client\__BESData\actionsite
T: 0.056 ms
```

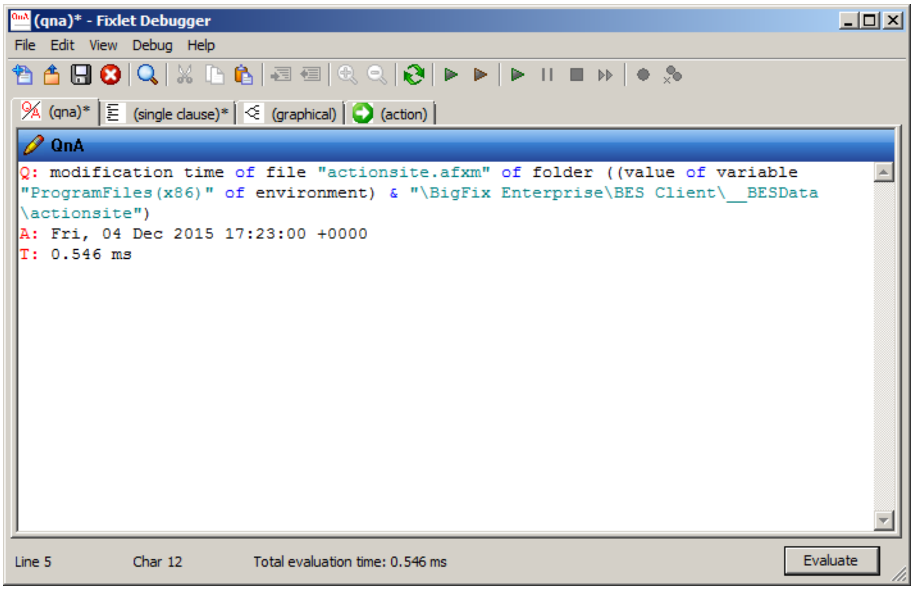
Line 2 Char 34 Total evaluation time: 0.056 ms

6. You now use this string value to define the folder where the `actionsite.afxm`file is located. Remember to put parentheses around the entire query for the folder object.

   ```
   Q: file “actionsite.afxm” of folder ((value of variable “ProgramFiles(x86)” of
   environment) & “\BigFix Enterprise\BES Client\__BESData\actionsite”)
   ```

7. Finish by getting the modification time of the file.

   ```
   Q: modification time of file “actionsite.afxm” of folder ((value of variable
   “ProgramFiles(x86)” of environment) &“\BigFix Enterprise\BES
   Client\__BESData\actionsite”)
   ```



```
Q: modification time of file "actionsite.afxm" of folder ((value of variable
"ProgramFiles(x86)" of environment) & "\BigFix Enterprise\BES Client\__BESData
\actionsite")
A: Fri, 04 Dec 2015 17:23:00 +0000
T: 0.546 ms
```

Line 5 Char 12 Total evaluation time: 0.546 ms

Exercise 24 Inspecting the registry

In this exercise, you develop a Relevance query to inspect the Windows registry of the local host to determine the version of DirectX that is installed.

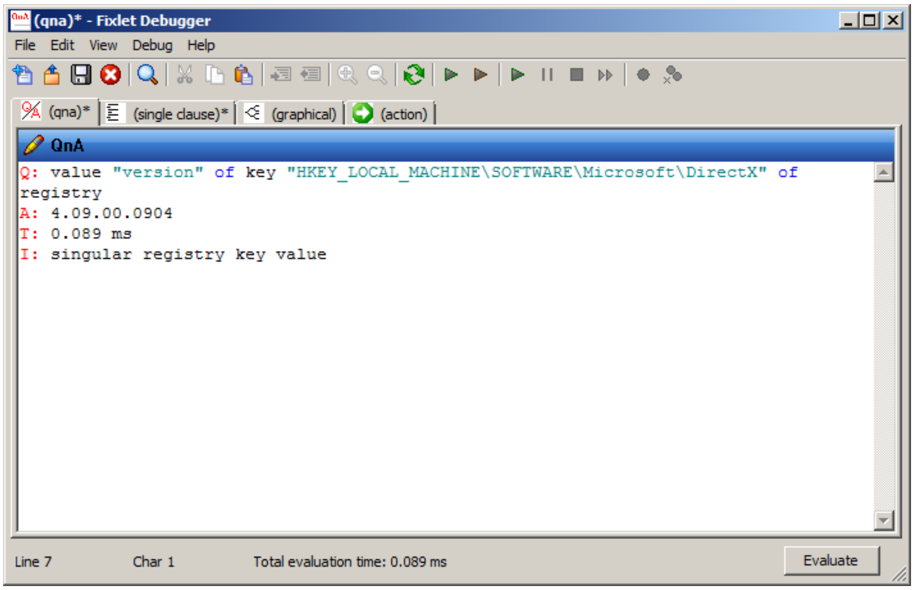
To create this query, you must examine the following registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DirectX
```

Retrieve this version information by examining the value of the **Version** subkey. This value is not a version object in Relevance, but represents the value that is stored in this key.

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. In the text box, create the query by obtaining the value of the target registry key. Click **Evaluate**.

   ```
   Q: value “version” of key “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DirectX” of
   registry
   ```



```
Q: value "version" of key "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DirectX" of
registry
A: 4.09.00.0904
T: 0.089 ms
I: singular registry key value
```

Note: This query returns a **singular registry key value** data type.

Exercise 25 Using a whose-it clause to avoid errors

In this exercise, you develop a Relevance query to return True if the specified registry key exists and the key has a string value that is named **value** set to **BigFix**.

Hint: This registry key is created when any component of BigFix is installed. If you are not performing these exercises on the lab images and do not have BigFix installed, you might need to create this registry key.

```
HKEY_LOCAL_MACHINE\SOFTWARE\BigFix\Brand
```

- Use a **`whose-it`** clause to avoid any errors
- Ensure that no error is generated if the following conditions exist:
  - The specified key does not exist.
  - The specified value does not exist.

Use the Fixlet Debugger to perform the following steps:

1. Begin with the registry key.

   ```
   Q: key “HKEY_LOCAL_MACHINE\SOFTWARE\BigFix\Brand” of registry
   ```

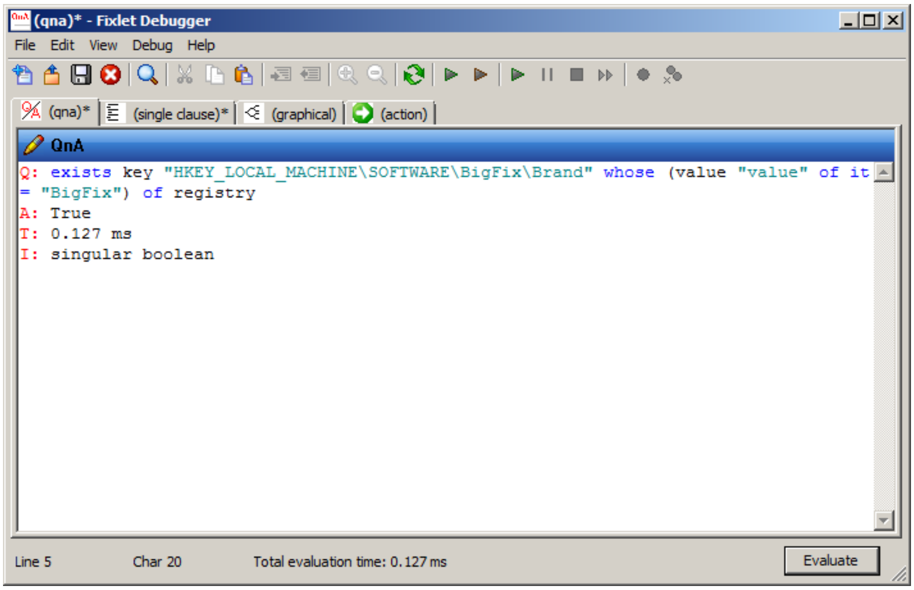
Hint: This statement will generate an error if the specified key does not already exist.

2. Test to confirm that the key exists.

   ```
   Q: exists key “HKEY_LOCAL_MACHINE\SOFTWARE\BigFix\Brand” of registry
   ```

3. Use a **`whose-it`** clause to test for the existence of the **value** of the key and determine whether it has the required data.

   ```
   Q: exists key “HKEY_LOCAL_MACHINE\SOFTWARE\BigFix\Brand” whose (value “value”
   of it = “BigFix”) of registry
   ```


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)
QnA
Line 5 Char 20 Total evaluation time: 0.127 ms
Evaluate

```
Q: exists key "HKEY_LOCAL_MACHINE\SOFTWARE\BigFix\Brand" whose (value "value" of it
= "BigFix") of registry
A: True
T: 0.127 ms
I: singular boolean
```

Exercise 26 Extracting a path name

In this exercise, you develop a Relevance query to find the version of a Windows address book file, `wab32.dll`. You use Relevance to examine the following key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WAB\DLLPath
```

Refer to the inspector library to see how to retrieve the default value of a registry key.

You can develop your query by using a syntax similar to the following example:

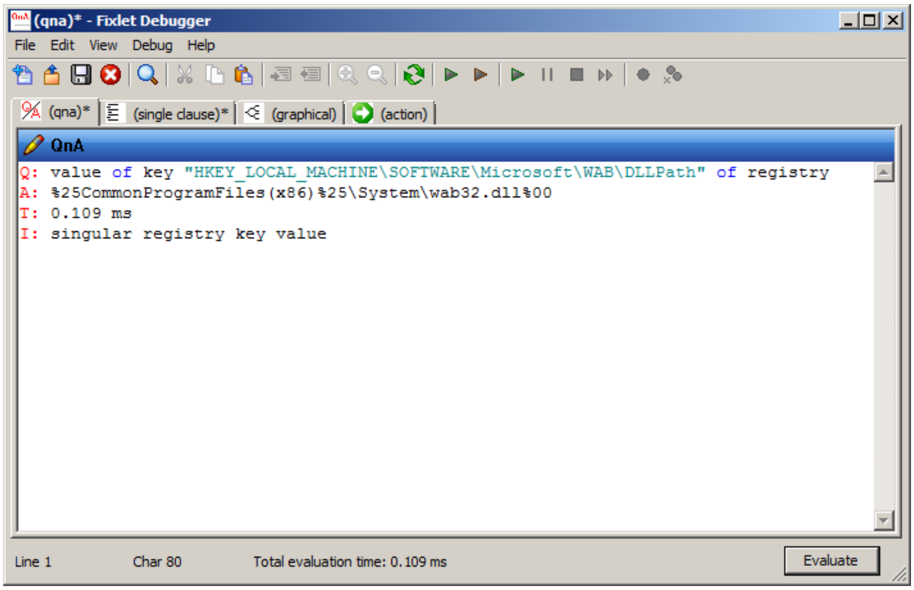
```
Q: version of file “H3lib.dll” of folder (value “Path” of key
“HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App
Paths\BESConsole.exe” of registry as string)
```

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. Examine the String objects of the BigFix Inspector Library. In particular, review the inspector named **`expand environment string of <string>`**.

3. Obtain the value of the registry key.

   ```
   Q: value of key “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WAB\DLLPath” of registry
   ```



```
Q: value of key "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WAB\DLLPath" of registry
A: %25CommonProgramFiles(x86)%25\System\wab32.dll%00
T: 0.109 ms
I: singular registry key value
```

Note: An environment string is used for the property in later versions of Windows instead of a static value.

4. Use the **expand environment string of <string>** inspector to extract the environment variable of the registry key.

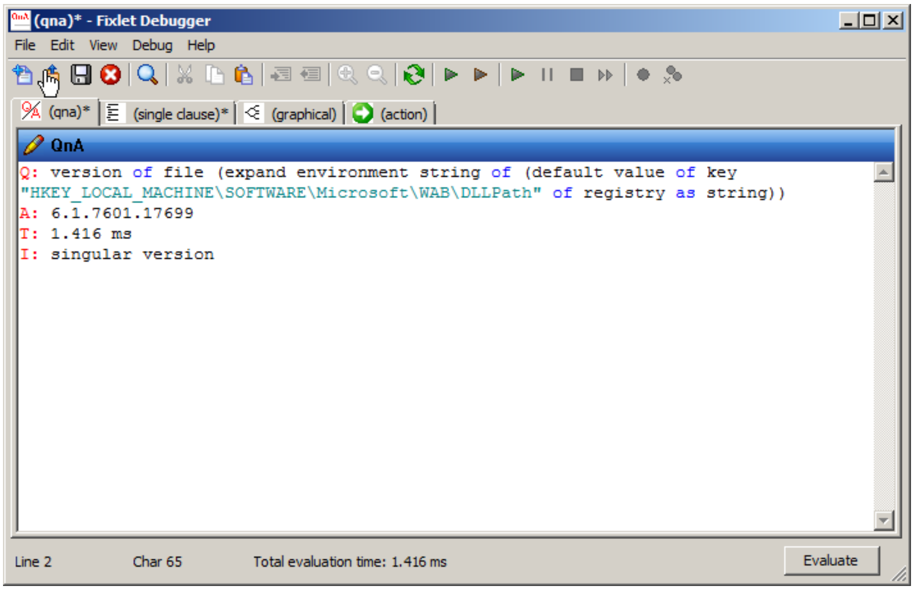
   ```
   Q: expand environment string of (default value of key
   “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WAB\DLLPath” of registry as string)
   ```

5. Test to confirm that the file exists.

   ```
   Q: file (expand environment string of (default value of key
   “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WAB\DLLPath” of registry as string))
   ```

6. Extract the version of the file.

   ```
   Q: version of file (expand environment string of (default value of key
   “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WAB\DLLPath” of registry as string))
   ```



```
Q: version of file (expand environment string of (default value of key
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WAB\DLLPath" of registry as string))
A: 6.1.7601.17699
T: 1.416 ms
I: singular version
```

Line 2   Char 65   Total evaluation time: 1.416 ms

Exercise 27 Determining the amount of time that a host is operational

In this exercise, you develop a Relevance query to return a Boolean value that indicates whether the host was operational for more than one day.

The objectives for this exercise are to perform the following tasks:

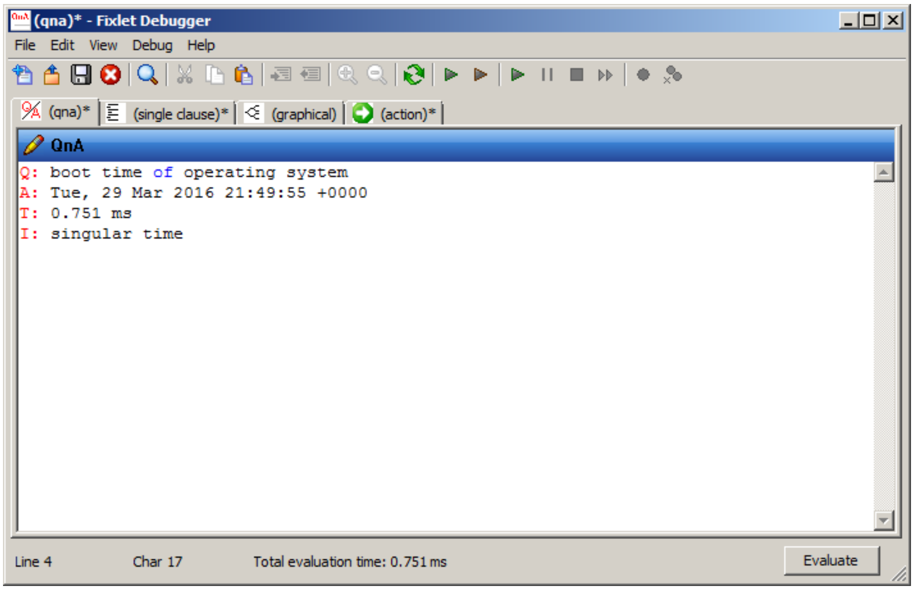
- Determine that the time of the last reboot corresponds to the boot time of operating system.
- Return a Boolean result.

Use these steps to create the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. In the text box, create a basic query to obtain the boot time and click **Evaluate**.

   `Q: boot time of operating system`



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA
Line 4
Char 17
Total evaluation time: 0.751 ms
Evaluate

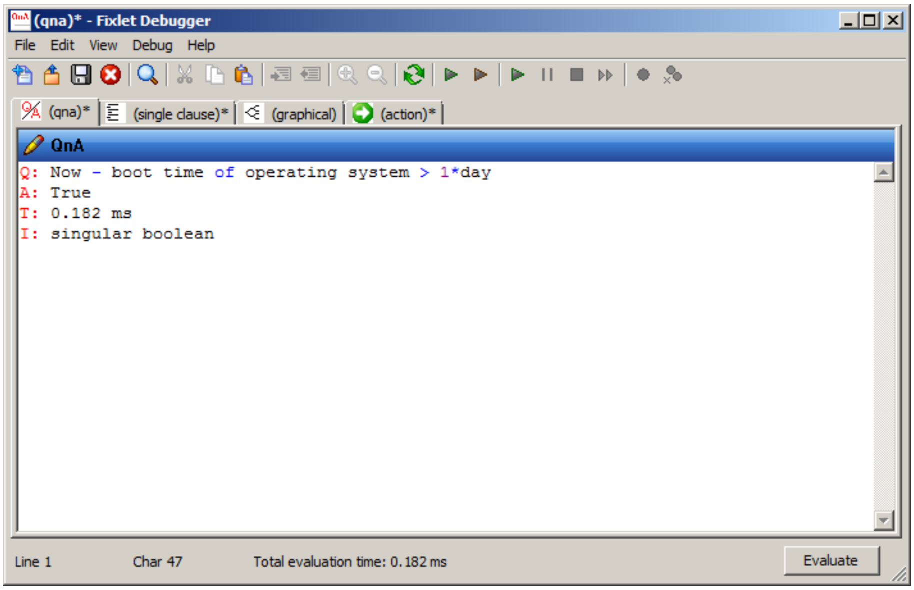
```
Q: boot time of operating system
A: Tue, 29 Mar 2016 21:49:55 +0000
T: 0.751 ms
I: singular time
```

3. Subtract the boot time from the current time **(Now)** and click **Evaluate** to see the elapsed time.

   `Q: Now - boot time of operating system`

4. Compare the elapsed time to the length of one day and convert the result to a Boolean value. Click **Evaluate**.

   `Q: Now - boot time of operating system > 1*day`



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA
Line 1
Char 47
Total evaluation time: 0.182 ms
Evaluate

```
Q: Now - boot time of operating system > 1*day
A: True
T: 0.182 ms
I: singular boolean
```

Exercise 28 Date constructs

In this exercise, you develop a Relevance query to display the current date and time in the following format:

YYYY/MM/DD HH:MM:SS

During this exercise, you use the various date and time inspectors to display the date and time in the desired format. To complete this exercise, you must use the local time zone of the system.

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. Begin constructing the query by using the `now` inspector to obtain the current date and time.

   ```
   Q: now
   ```

Complete the following steps to display the current date in the **YYYY/MM/DD** format.

3. Use the date object `year`to obtain the year for the current date.

   ```
   Q: year of date (local time zone) of now
   ```

4. Because you must use string concatenation to display the date in the desired format, you must convert the result to a string.

   ```
   Q: year of date (local time zone) of now as string
   ```

5. Use the date object `month`to obtain the month for the current date.

   ```
   Q: month of date (local time zone) of now
   ```

6. This query displays the name of the month but you must show the month as a numeric value. For example, 02 must be displayed instead of February. Refer to the Inspector Library under Casting Operators and locate the `<month> as two digits` casting object. This operator converts the month into 2 digits. Append the `as two digits`operator to the previous query.

   ```
   Q: month of date (local time zone) of now as two digits
   ```

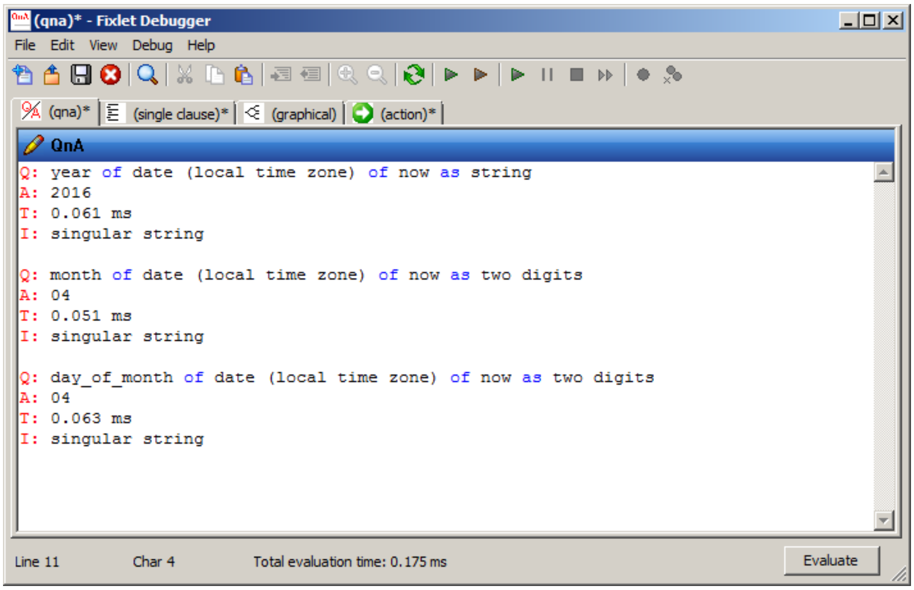
7. Use the date object `day_of_month`to obtain the day of the month for the current date.

   ```
   Q: day_of_month of date (local time zone) of now
   ```

Hint: This query returns the day of the month. However, if you run this query between the first and the ninth of the month, only a single digit is returned to represent the day. You must use the `as two digits`casting operator.

8. Append the **`as two digits`** casting operator to the end of the previous query to always return the day of the month as a two-digit string.

   `Q: day_of_month of date (local time zone) of now as two digits`

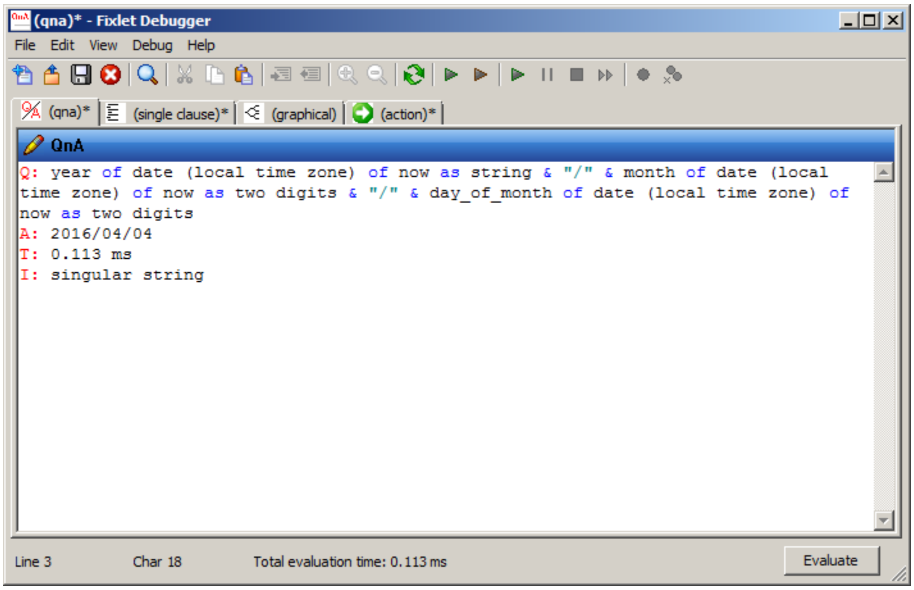

(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA

```
Q: year of date (local time zone) of now as string
A: 2016
T: 0.061 ms
I: singular string

Q: month of date (local time zone) of now as two digits
A: 04
T: 0.051 ms
I: singular string

Q: day_of_month of date (local time zone) of now as two digits
A: 04
T: 0.063 ms
I: singular string
```

Line 11 Char 4 Total evaluation time: 0.175 ms
Evaluate

9. Now that all of the components of the date are formatted properly and converted to strings, you use string concatenation to build and display the date in the **YYYY/MM/DD** format.

   `Q: year of date (local time zone) of now as string & “/” & month of date (local time zone) of now as two digits & “/” & day_of_month of date (local time zone) of now as two digits`


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA

```
Q: year of date (local time zone) of now as string & "/" & month of date (local
time zone) of now as two digits & "/" & day_of_month of date (local time zone) of
now as two digits
A: 2016/04/04
T: 0.113 ms
I: singular string
```

Line 3 Char 18 Total evaluation time: 0.113 ms
Evaluate

Hint: Notice that the phrase *date (local time zone) of now* is repeated three times in this query. You should use an **`it-without-a-whose`** construct to refer to this object once.

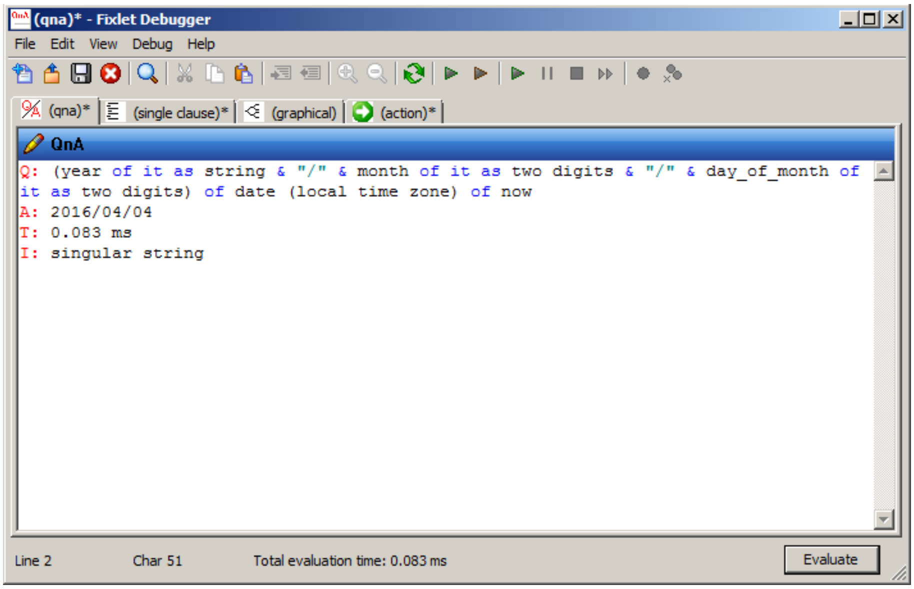
10. Begin by surrounding the entire query with parentheses.

11. Copy the duplicated phrase to the right of the parentheses. Be certain to create a proper construct by adding `of`.

```
Q: (year of date (local time zone) of now as string & "/" & month of date (local
time zone) of now as two digits & "/" & day_of_month of date (local time zone)
of now as two digits) of date (local time zone) of now
```

12. Replace the duplicated phrase *of date (local time zone) of now* within the parentheses with `it`.

```
Q: (year of it as string & "/" & month of it as two digits & "/" & day_of_month of
it as two digits) of date (local time zone) of now
```


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA
Line 2
Char 51
Total evaluation time: 0.083 ms
Evaluate

```
Q: (year of it as string & "/" & month of it as two digits & "/" & day_of_month of
it as two digits) of date (local time zone) of now
A: 2016/04/04
T: 0.083 ms
I: singular string
```

Now that the date is reported in the desired format, you must display the time in the **hh:mm:ss** format.

To develop this query, you must use the following three new string properties:

- two digit hour of <time of day>
- two digit minute of <time of day>
- two digit second of <time of day>

Each of these properties return string values that allow you to format the output as required.

13. Open a new **QnA** tab by selecting **File > New Tab > New QnA Tab**. Select the new tab if it is not already selected.

Hint: By opening a new **QnA** tab, you can retain the date query while you work on the new query to extract and format the time.

14. Create the query for the hour of the day with the **`two digit hour of <time of day>`** construct.

    `Q: two digit hour of time (local time zone) of now`

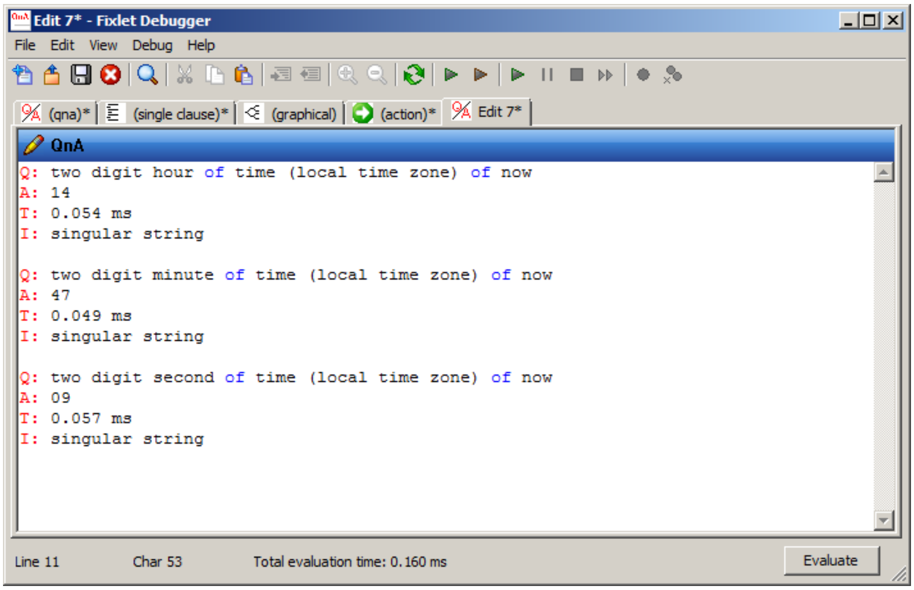
Hint: When extracting the date properties, you use the **`date (local time zone) of now`** construct, but for the time properties you use the **`time (local time zone) of now`** construct.

15. Create the query for the minute of the day using the **`two digit minute of <time of day>`** construct.

    `Q: two digit minute of time (local time zone) of now`

16. Create the query for the seconds of the day using the **`two digit second of <time of day>`** construct.

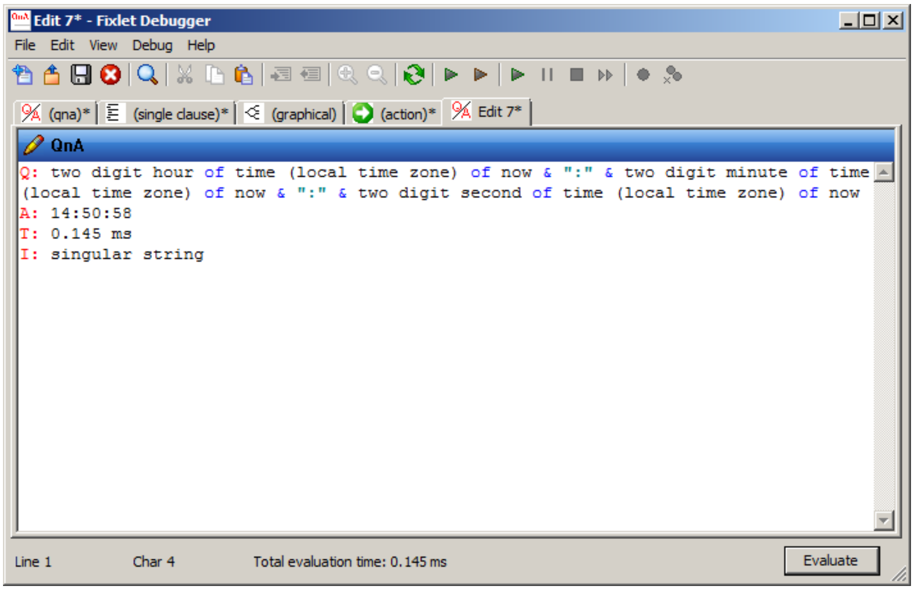
    `Q: two digit second of time (local time zone) of now`


Edit 7* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)* Edit 7*
QnA
Line 11 Char 53 Total evaluation time: 0.160 ms
Evaluate

```
Q: two digit hour of time (local time zone) of now
A: 14
T: 0.054 ms
I: singular string

Q: two digit minute of time (local time zone) of now
A: 47
T: 0.049 ms
I: singular string

Q: two digit second of time (local time zone) of now
A: 09
T: 0.057 ms
I: singular string
```

17. Now that you have all of the components of the time, use string concatenation to place the various components into the **hh:mm:ss** format.

```
Q: two digit hour of time (local time zone) of now & ":" & two digit minute of time (local time zone) of now & ":" & two digit second of time (local time zone) of now
```



Edit 7* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)* Edit 7*
QnA
Line 1 Char 4 Total evaluation time: 0.145 ms
Evaluate

```
Q: two digit hour of time (local time zone) of now & ":" & two digit minute of time
(local time zone) of now & ":" & two digit second of time (local time zone) of now
A: 14:50:58
T: 0.145 ms
I: singular string
```

Hint: Notice that the phrase *time (local time zone) of now* is repeated three times in this query. You should use an **`it-without-a-whose`** construct to refer to this object only once.

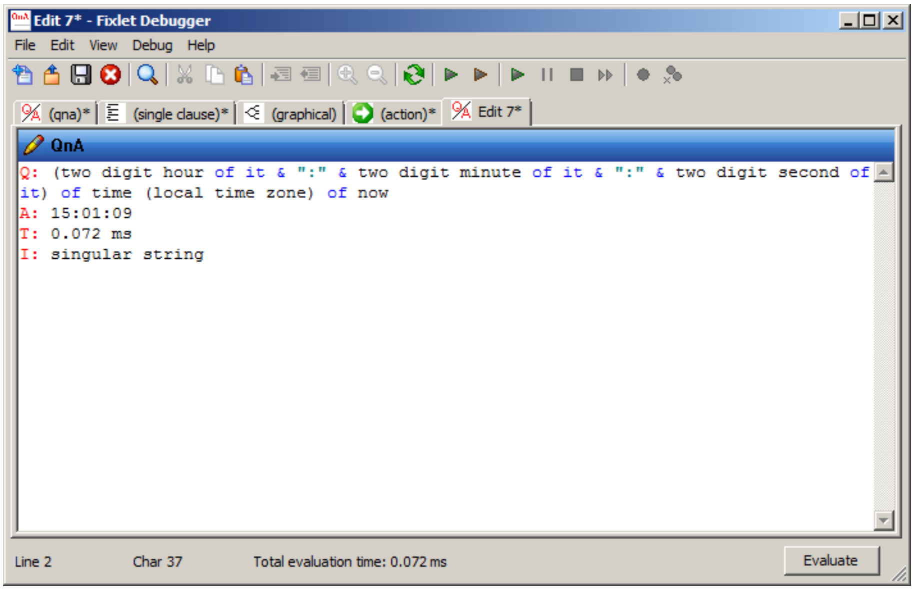
18. Begin by surrounding the entire query with parentheses.

19. Copy the duplicated phrase *time (local time zone) of now* to the right of the parentheses and precede it with the **`of`** construct.

```
Q: (two digit hour of time (local time zone) of now & ":" & two digit minute of time (local time zone) of now & ":" & two digit second of time (local time zone) of now) of time (local time zone) of now
```

20. Replace the duplicated phrase *time (local time zone) of now* within the parentheses with the **`it`** construct.

```
Q: (two digit hour of it & “:” & two digit minute of it & “:” & two digit second
of it) of time (local time zone) of now
```



```
Q: (two digit hour of it & ":" & two digit minute of it & ":" & two digit second of
it) of time (local time zone) of now
A: 15:01:09
T: 0.072 ms
I: singular string
```

Line 2   Char 37   Total evaluation time: 0.072 ms

You now have two queries on two separate **QnA** tabs:

- The date is formatted as YYYY/MM/DD.
- The time is formatted as hh:mm:ss.

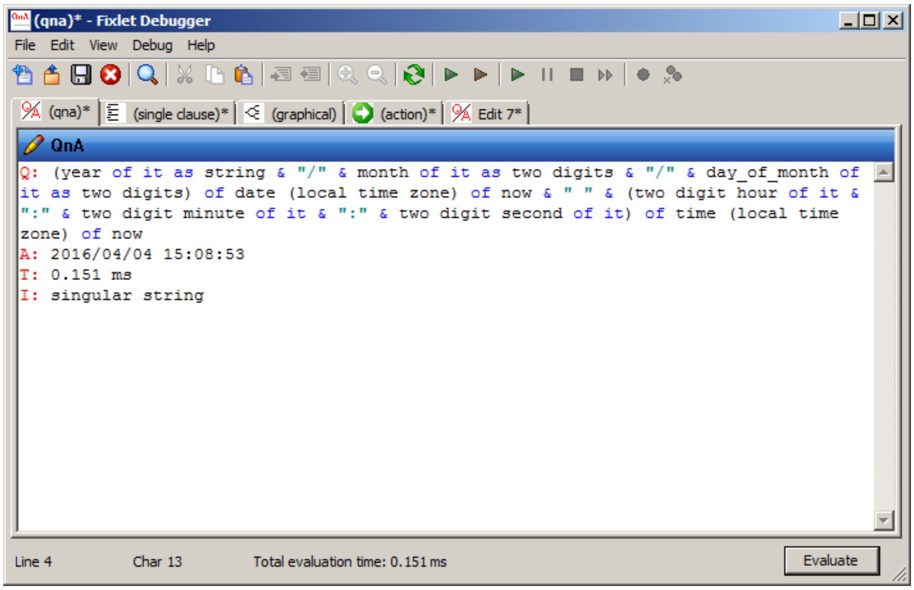
You must now combine these queries to create the final output.

21. Copy the query for the time excluding the **Q:** at the beginning of the query.

22. Select the **QnA** tab that has the query for the date.

23. Add the concatenation syntax **`& " " &`** at the end of the date query so that a space is placed between the two elements, then paste the time query after the concatenation construct.

```
Q: (year of it as string & “/” & month of it as two digits & “/” & day_of_month of
it as two digits) of date (local time zone) of now & “ “ & (two digit hour of
it & “:” & two digit minute of it & “:” & two digit second of it) of time (local
time zone) of now
```



(qna)* - Fixlet Debugger

File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)* Edit 7*
QnA

```
Q: (year of it as string & "/" & month of it as two digits & "/" & day_of_month of
it as two digits) of date (local time zone) of now & " " & (two digit hour of it &
":" & two digit minute of it & ":" & two digit second of it) of time (local time
zone) of now
A: 2016/04/04 15:08:53
T: 0.151 ms
I: singular string
```

Line 4 Char 13 Total evaluation time: 0.151 ms

Exercise 29 Finding the first of the month

In this exercise, you develop a Relevance query that uses date calculations to determine what day that the first day of the current month fell on. There are many ways to achieve this goal, but the method used in this exercise was chosen to help you understand the use of date calculations and the **`current date`**expression.

Use these steps to create the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

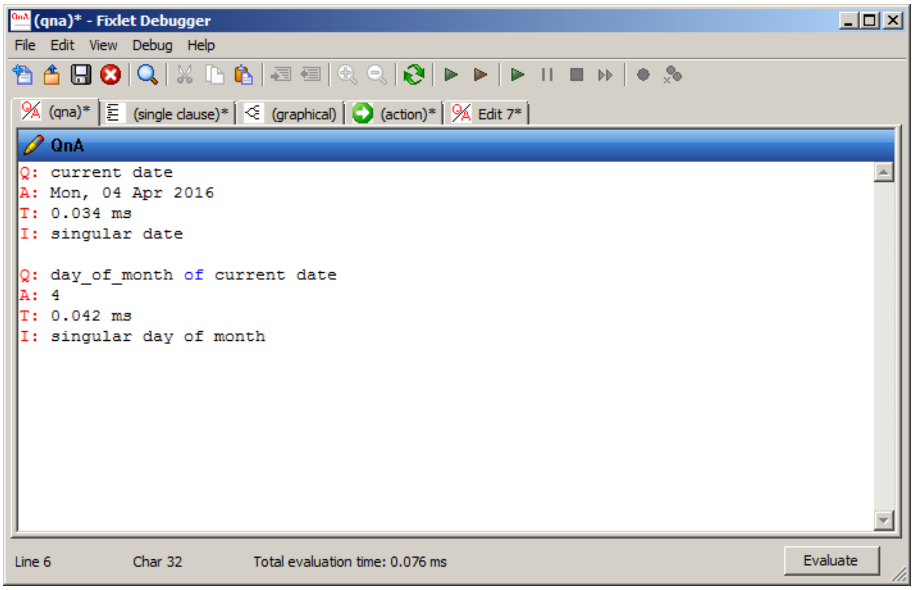
2. Use the **`current date`** date object to return the current date.

   ```
   Q: current date
   ```

Note: This query returns a **singular date** data type.

3. To obtain the day of the month for the calculation, use the date object **day_of_month<string>**.

   `Q: day_of_month of current date`



```
Q: current date
A: Mon, 04 Apr 2016
T: 0.034 ms
I: singular date

Q: day_of_month of current date
A: 4
T: 0.042 ms
I: singular day of month
```

   From the previous exercise, you know how to retrieve the day of the month. To obtain the first day of the month, you must use date calculations to subtract the current day of the month minus one day from the current day of the month.

4. To perform the data calculations, start with the current date and subtract 1 day.

   `Q: current date - 1*day`

5. You now need to change the **date** type into a **day of the month** type so that you can perform the date calculations.

   `Q: day_of_month of (current date - 1*day)`

Note: This query returns a **singular day-of-month** data type.

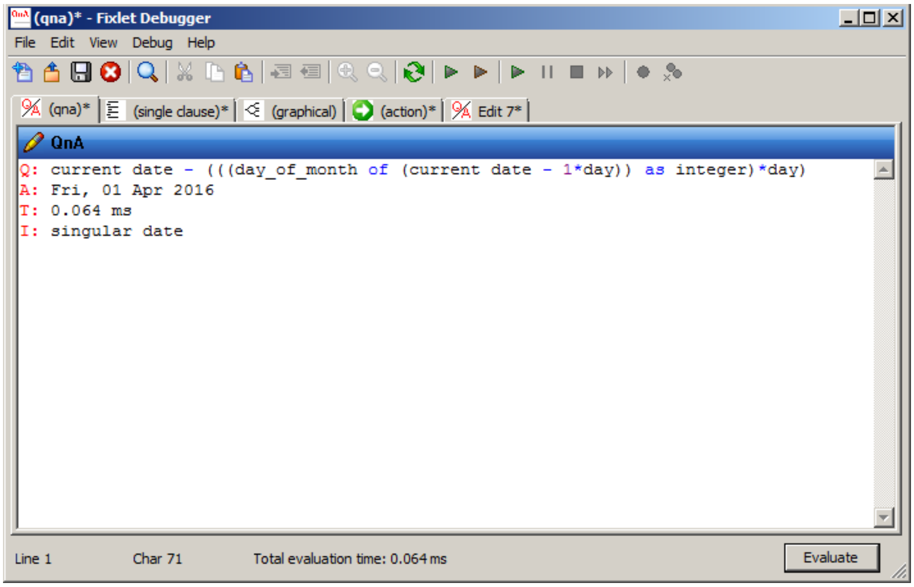
6. Now that you have calculated the day of the month minus one day, you can use this value to perform the date calculation. Because you have a **day of month** type, you must first convert the result to an **integer** to perform the calculation. Remember to include parentheses to correctly perform the conversion.

   `Q: (day_of_month of (current date - 1*day)) as integer`

7. You can now formulate the final query. Remember to include parentheses as required to delineate the various properties. Note – interesting find the “old” version does not work if the day of the month is the 1st.

```
Old Q: current date - (((day_of_month of (current date - 1*day)) as
integer)*day)
```

```
New Q: current date - (((day_of_month of (current date)) as integer -1)*day)
```



```
Q: current date - (((day_of_month of (current date - 1*day)) as integer)*day)
A: Fri, 01 Apr 2016
T: 0.064 ms
I: singular date
```

Exercise 30 Displaying the last line of a log file

In this exercise, you determine the number of lines in the BigFix client log file and then display the last line of the file.

Note: If you are not performing the labs on the supplied lab environment, you must ensure that the system that you are using has the BigFix client installed and has generated at least two days of log files. The client log file for the current day is locked.

The default location of the BigFix client log file varies by platform, as shown below: Windows

x86

```
C:\Program Files\BigFix Enterprise\BES Client\__BESData\__Global\Logs
```

Windows x64

```
C:\Program Files (x86)\BigFix Enterprise\BES Client\__BESData\__Global\Logs
```

UNIX / Linux systems

```
/var/opt/BESClient/  BESData/__Global/Logs
```

Mac systems

```
/Library/Application Support/Bigfix/BES Agent/  BESData/__Global/Logs
```

The name of each file is based on the creation date in the `YYYYMMDD.log`format. For this exercise, you can use hardcoded paths and file names. For the final query, you use an **`it-without-a-whose`** clause so that you do not need to specify the file name multiple times.

You use a syntax similar to the following one:

```
Q: line 54 of file “c:\temp\test.txt”
```

Note: The instructions for this exercise assume that you are performing the lab on the **BESFNDWIN10** virtual machine, which is a Windows x64 platform. If you perform the lab on another system type, you must substitute the required path using the platform examples shown above.

Use these steps to perform the query:

1. Return to the Fixlet Debugger on the **BESFNDWIN10** virtual machine and select the **(qna)** tab.

Note: If you are not using the lab image set, manually copy a log file to the required folder as indicated above in order to create the environment that you can use in this exercise.

2. Using the Windows Explorer, browse to the following path:

   ```
   C:\Program Files (x86)\BigFix Enterprise\BES Client\__BESData\__Global\Logs
   ```

3. Select a log file for this exercise with the following naming convention:

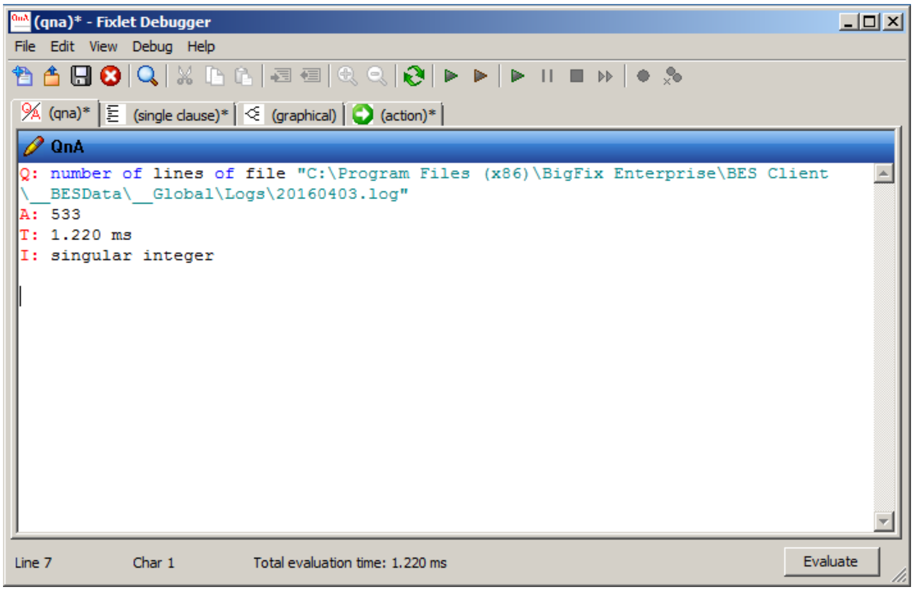
   ```
   YYYYMMDD.log
   ```

Hint: You must select a file other than the one from today. The current log file is regularly locked by the BigFix client. You might receive an error if you try to access it.

4. In the text box, create a basic query to check the number of lines of the file by using the hardcoded path:

   ```
   Q: number of lines of file “C:\Program Files (x86)\BigFix Enterprise\BES
   Client\__BESData\__Global\Logs\<Logfile_date>.log”
   ```

Hint: Be sure and substitute the actual filename for `<Logfile_date>.log` in the query.



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA

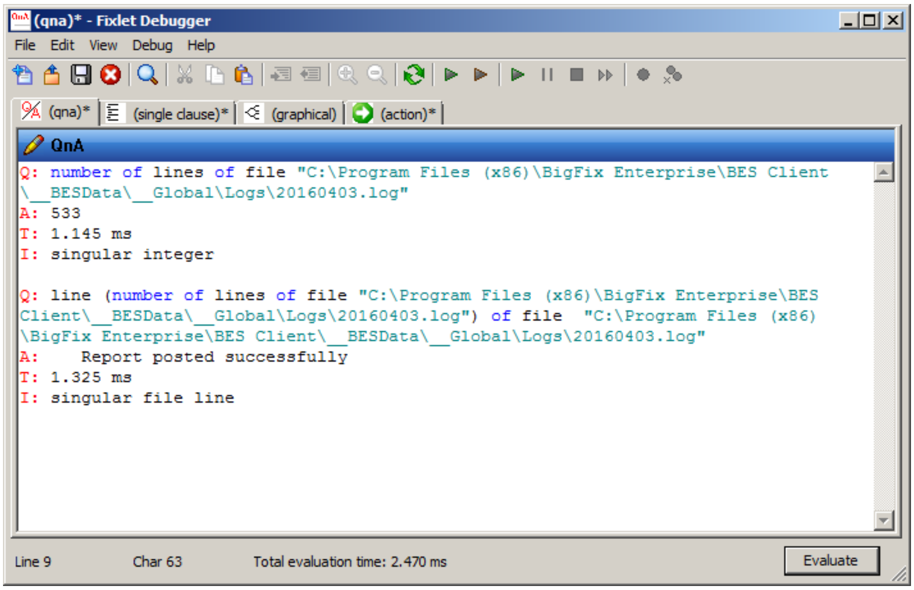
```
Q: number of lines of file "C:\Program Files (x86)\BigFix Enterprise\BES Client
\__BESData\__Global\Logs\20160403.log"
A: 533
T: 1.220 ms
I: singular integer
```

Line 7
Char 1
Total evaluation time: 1.220 ms
Evaluate

5. Extend the query to locate and display the last line of the file.

   Q: line (number of lines of file “C:\Program Files (x86)\BigFix Enterprise\BES Client__BESData__Global\Logs\<Logfile_date>.log”) of file “C:\Program Files (x86)\BigFix Enterprise\BES Client__BESData__Global\Logs\<Logfile_date>.log”

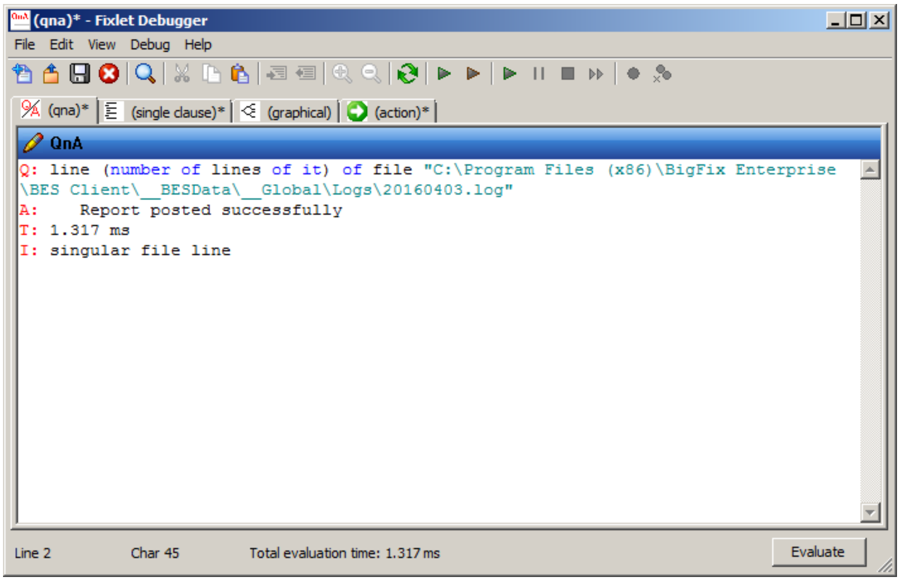
Hint: Remember to substitute the actual filename for `<Logfile_date>.log` in the query.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA

```
Q: number of lines of file "C:\Program Files (x86)\BigFix Enterprise\BES Client
\__BESData\__Global\Logs\20160403.log"
A: 533
T: 1.145 ms
I: singular integer

Q: line (number of lines of file "C:\Program Files (x86)\BigFix Enterprise\BES
Client\__BESData\__Global\Logs\20160403.log") of file  "C:\Program Files (x86)
\BigFix Enterprise\BES Client\__BESData\__Global\Logs\20160403.log"
A:    Report posted successfully
T: 1.325 ms
I: singular file line
```

6. Optimize the query by using the **`it-without-a-whose`** construct to replace the initial file reference.

   ```
   Q: line (number of lines of it) of file “C:\Program Files (x86)\BigFix
   Enterprise\BES Client\__BESData\__Global\Logs\<Logfile_date>.log”
   ```


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA

```
Q: line (number of lines of it) of file "C:\Program Files (x86)\BigFix Enterprise
\BES Client\__BESData\__Global\Logs\20160403.log"
A:    Report posted successfully
T: 1.317 ms
I: singular file line
```

Line 2
Char 45
Total evaluation time: 1.317 ms
Evaluate

Exercise 31 Determining the number of lines in a log file without hardcoded paths

In this exercise, enhance the query that you developed in Exercise 31, “Displaying the last line of a log file,” except that you use dynamically generated paths and file names instead of hardcoding them.

To successfully complete this exercise, you must use the following methods:

- You use regapp to locate the BigFix Client installation folder.
- You use time and date inspectors to determine the file name for the log file.

Use these steps to perform the query:

1. Open an instance of File Explorer and browse to this location:

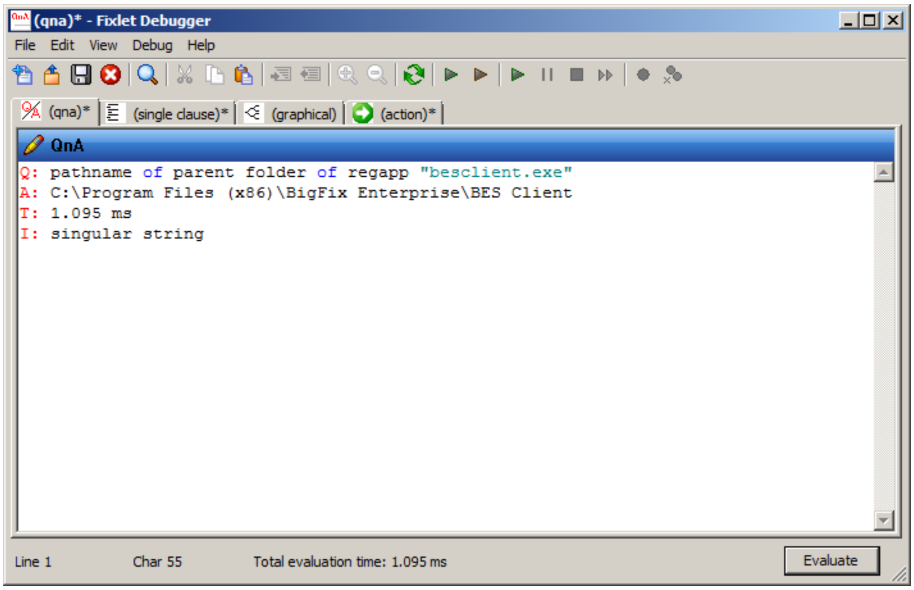
   `C:\Program Files (x86)\BigFix Enterprise\BES Client\__BESData\__Global\Logs`

2. If a log file for the previous day does not exist, copy today’s log file and rename it to yesterday’s date by using the correct file-naming convention, *YYYMMDD.log*.

3. Return to the Fixlet Debugger and select the **(qna)** tab.

4. In the text box, create a basic query to determine the path to the BigFix Client installation folder.

   `Q: pathname of parent folder of regapp “besclient.exe”`



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA
Line 1 Char 55 Total evaluation time: 1.095 ms
Evaluate

```
Q: pathname of parent folder of regapp "besclient.exe"
A: C:\Program Files (x86)\BigFix Enterprise\BES Client
T: 1.095 ms
I: singular string
```

5. Concatenate the relative subpath of the BigFix Client log files to the BigFix Client installation folder to build a string that represents the full path to the log files.

   ```
   Q: pathname of parent folder of regapp “besclient.exe” &
   “\__BESData\__Global\Logs”
   ```

6. Use the following steps to build a query that represents the name for the client log file that was created yesterday:

   a. Obtain the current year.

   ```
   Q: year of current date
   ```

   b. Convert the results to a string for concatenation purposes.

   ```
   Q: year of current date as string
   ```

   c. Obtain the name of the month.

   ```
   Q: month of current date
   ```

   d. Convert the month to two digits.

   ```
   Q: month of current date as two digits
   ```

   e. Obtain the day of the month.

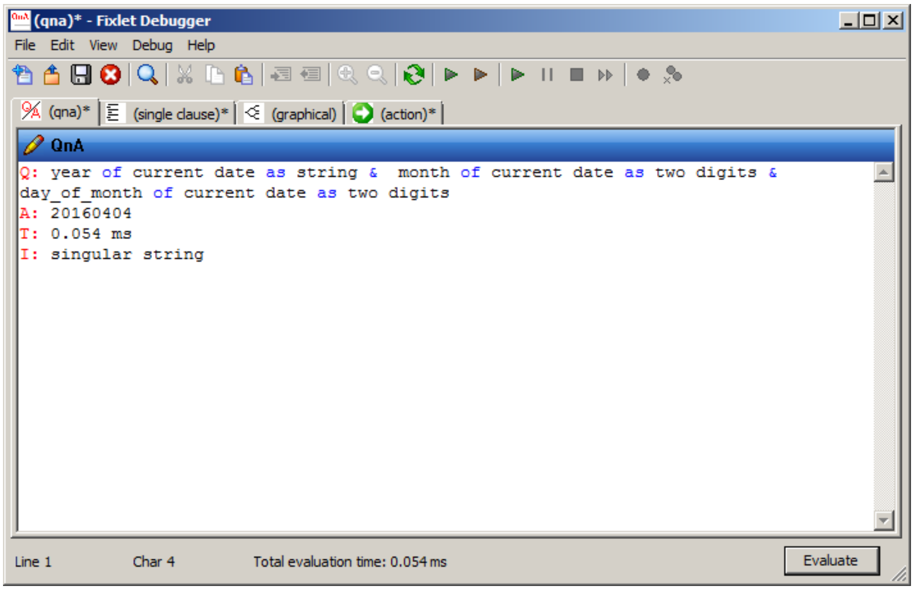
   ```
   Q: day_of_month of current date
   ```

   f. Convert the day of the month to two digits to account for the 1st - 9th day of the month.

   ```
   Q: day_of_month of current date as two digits
   ```

   g. Build the file name by concatenating the results of the previous queries.

   ```
   Q: year of current date as string &  month of current date as two digits &
   day_of_month of current date as two digits
   ```



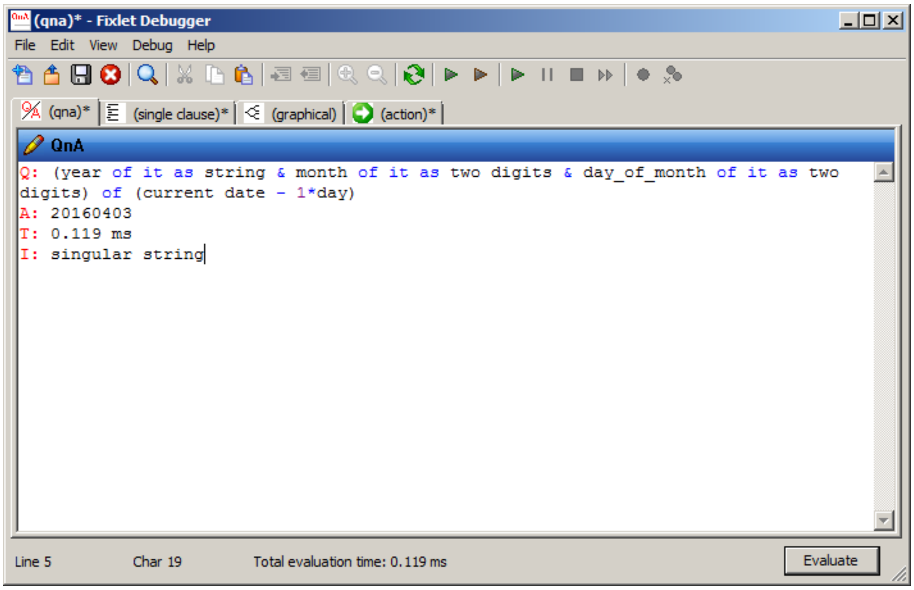
```
Q: year of current date as string &  month of current date as two digits &
day_of_month of current date as two digits
A: 20160404
T: 0.054 ms
I: singular string
```

   h. You now use the **it-without-a-whose** construct to simplify the query to remove the duplicated **current date** reference.

   ```
   Q: (year of it as string & month of it as two digits & day_of_month of it as
   two digits) of current date
   ```

i. You must now subtract 1 day from the current date to determine yesterday's date.

```
Q: (year of it as string & month of it as two digits & day_of_month of it as two digits) of (current date - 1*day)
```

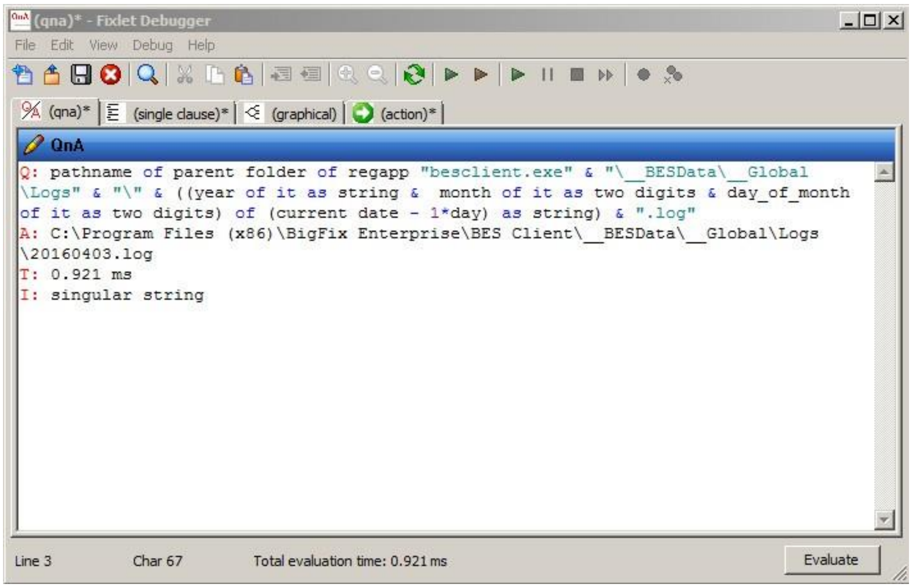


```
Q: (year of it as string & month of it as two digits & day_of_month of it as two
digits) of (current date - 1*day)
A: 20160403
T: 0.119 ms
I: singular string
```

Line 5 Char 19 Total evaluation time: 0.119 ms

7. You now include the location to the log file to build the complete file name and append the file extension `.log` using concatenation.

```
Q: pathname of parent folder of regapp “besclient.exe” &
“\__BESData\__Global\Logs” & “\” & ((year of it as string &  month of it as two
digits & day_of_month of it as two digits) of (current date - 1*day) as string)
& “.log”
```

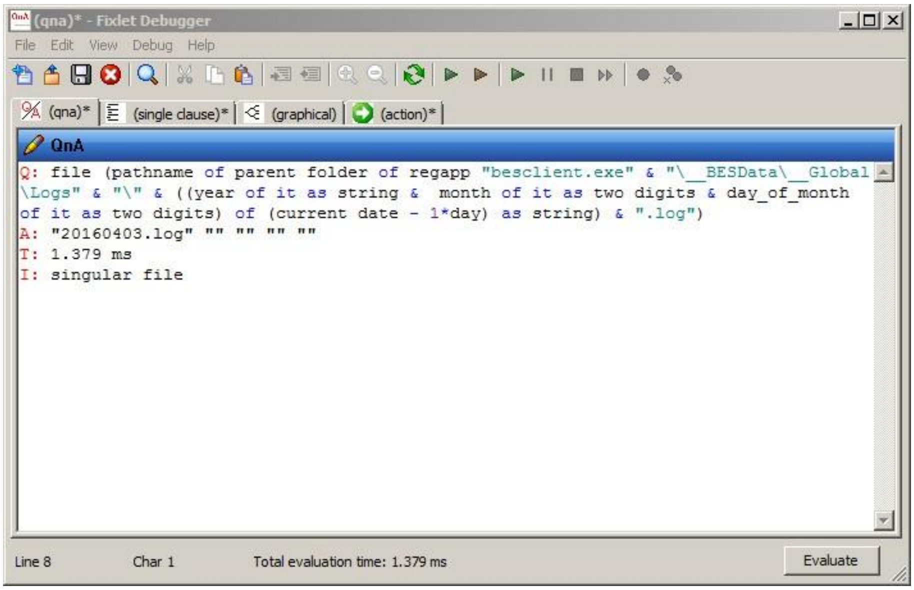


```
Q: pathname of parent folder of regapp "besclient.exe" & "\__BESData\__Global
\Logs" & "\" & ((year of it as string &  month of it as two digits & day_of_month
of it as two digits) of (current date - 1*day) as string) & ".log"
A: C:\Program Files (x86)\BigFix Enterprise\BES Client\__BESData\__Global\Logs
\20160403.log
T: 0.921 ms
I: singular string
```

Line 3 Char 67 Total evaluation time: 0.921 ms

8. Convert the results to a file object.

   Q: file (pathname of parent folder of regapp “besclient.exe” & “__BESData__Global\Logs” & “\” & (year of it as string & month of it as two digits & day_of_month of it as two digits) of (current date - 1*day) & “.log”)



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA

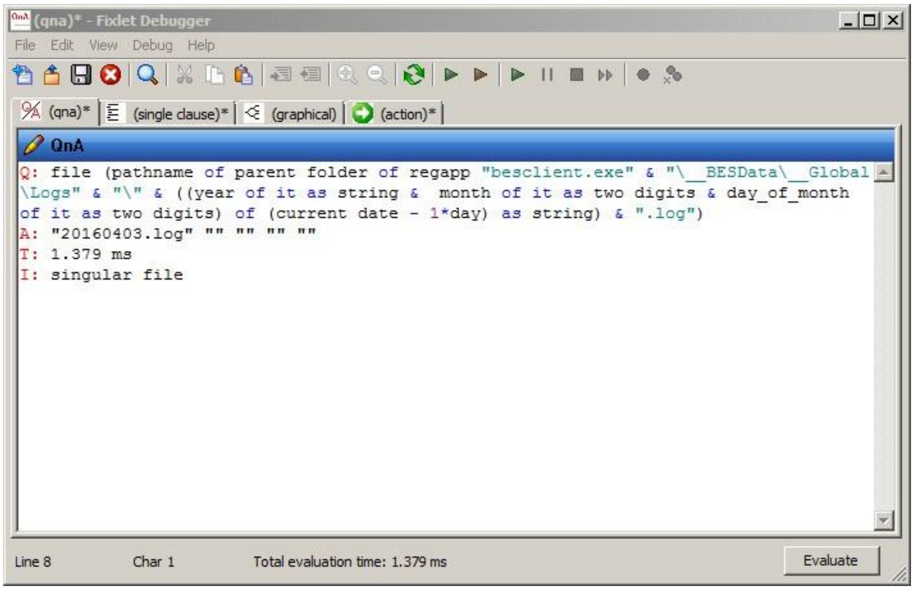
```
Q: file (pathname of parent folder of regapp "besclient.exe" & "\__BESData\__Global
\Logs" & "\" & ((year of it as string &  month of it as two digits & day_of_month
of it as two digits) of (current date - 1*day) as string) & ".log")
A: "20160403.log" "" "" "" ""
T: 1.379 ms
I: singular file
```

Line 8 Char 1 Total evaluation time: 1.379 ms

9. Complete the query by including the line reference.

   Q: line (number of lines of it) of file (pathname of parent folder of regapp “besclient.exe” & “__BESData__Global\Logs” & “\” & (year of it as string & month of it as two digits & day_of_month of it as two digits) of (current date - 1*day) & “.log”)

Note: Your results might differ, depending on the actual contents of the log file that is used for this exercise.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA
Line 8
Char 1
Total evaluation time: 1.379 ms
Evaluate

```
Q: file (pathname of parent folder of regapp "besclient.exe" & "\__BESData\__Global
\Logs" & "\" & ((year of it as string &  month of it as two digits & day_of_month
of it as two digits) of (current date - 1*day) as string) & ".log")
A: "20160403.log" "" "" "" ""
T: 1.379 ms
I: singular file
```

Note: To evaluate against the current file that is in use/locked, the verbiage “locked” can be entered in front of “line(s)”

Q: **locked** line (number of **locked** lines of it) of file (pathname of parent folder of regapp “besclient.exe” & “__BESData__Global\Logs” & “\” & (year of it as string & month of it as two digits & day_of_month of it as two digits) of (current date~~ - **1*day**~~) & “.log”)

Exercise 32 Determining the number of lines in a log file and using error checking

In this exercise, you extend the query that you created in Exercise 32, “Determining the number of lines in a log file without hardcoded paths” to add error checking. The purpose of the error checking is to ensure that the BigFix Client log file exists before you try to parse it.

You use an **`if-then-else`** construct to setup the error checking. The **then** portion uses the Relevance from Exercise 17. The **else** portion displays a message that indicates that the file does not exist.

Use these steps to perform the query:

1. Return to the Fixlet Debugger.

2. Open a new **QnA** tab by selecting **File > New Tab > New QnA Tab**.

   The new **QnA** tab opens and is automatically selected.

3. Perform the following steps to create the **`if-then-else`** construct.

   a. Create the initial if-then-else template.

   ```
   Q: if () then () else ()
   ```

   b. Copy the final Relevance statement from Exercise 17 inside the parenthesis that represent the **Then** portion of the **`if-then-else`** construct.

   ```
   Q: if () then (line (number of lines of it) of file (pathname of parent
   folder of regapp “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year
   of it as string & month of it as two digits & day_of_month of it as two
   digits) of (current date - 1*day) & “.log”)) else ()
   ```

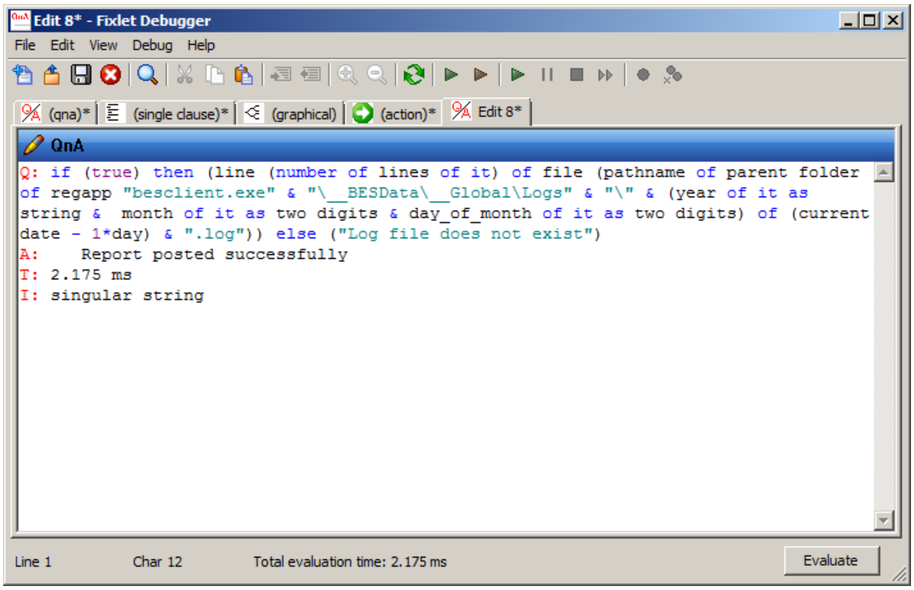
   c. Add a meaningful error message in the **Else** portion of the **`if-then-else`** construct.

   ```
   Q: if () then (line (number of lines of it) of file (pathname of parent
   folder of regapp “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year of
   it as string & month of it as two digits & day_of_month of it as two
   digits) of (current date - 1*day) & “.log”)) else (“Log file does not exist”)
   ```

4. Perform the following steps to test the **`if-then-else`** construct by using Boolean operators in the **If** portion of the statement.

   a. Insert the Boolean **True** operator in the **If** portion of the statement. This indicates that the file is found and runs the **Then** portion of the statement to return the last line of the file.

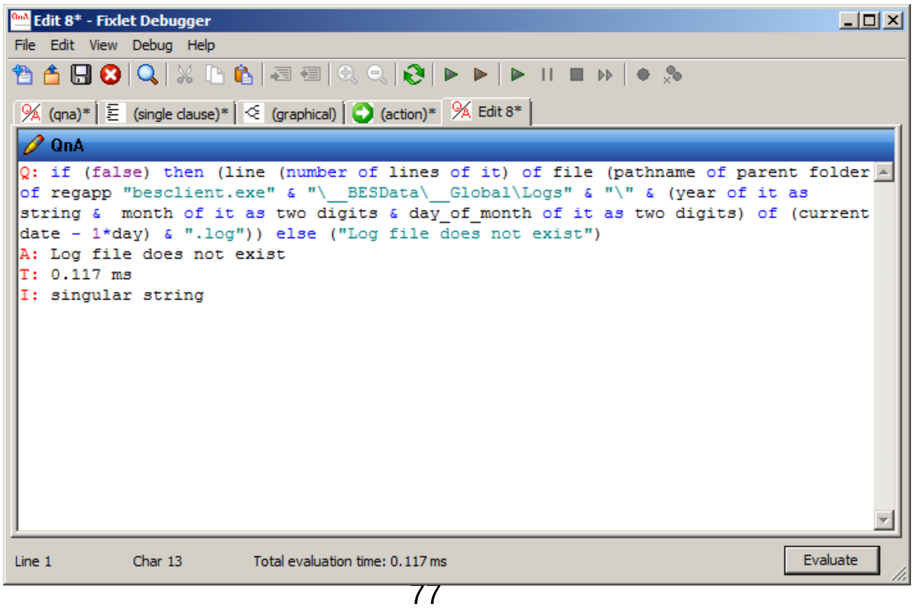
   ```
   Q: if (true) then (line (number of lines of it) of file (pathname of parent
   folder of regapp “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year of
   it as string & month of it as two digits & day_of_month of it as two
   digits) of (current date - 1*day) & “.log”)) else (“Log file does not exist”)
   ```



```
Q: if (true) then (line (number of lines of it) of file (pathname of parent folder
of regapp "besclient.exe" & "\__BESData\__Global\Logs" & "\" & (year of it as
string &  month of it as two digits & day_of_month of it as two digits) of (current
date - 1*day) & ".log")) else ("Log file does not exist")
A:    Report posted successfully
T: 2.175 ms
I: singular string
```

   b. Insert the **Boolean false** operator in the **If** portion of the statement. This indicates that the file is not found and displays the error message from the **Else** portion of the statement.

   ```
   Q: if (false) then (line (number of lines of it) of file (pathname of parent
   folder of regapp “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year
   of it as string & month of it as two digits & day_of_month of it as two
   digits) of (current date - 1*day) & “.log”)) else (“Log file does not exist”)
   ```

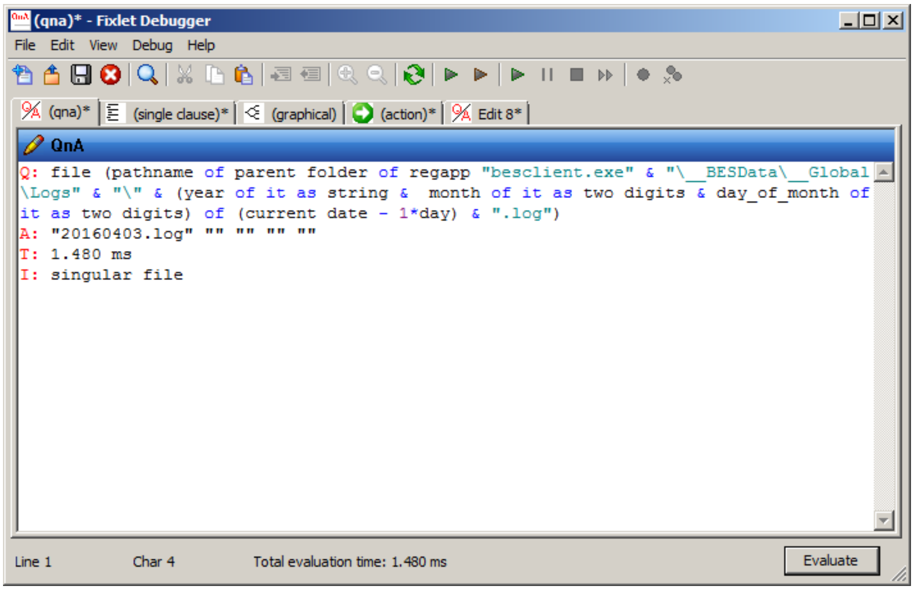


```
Q: if (false) then (line (number of lines of it) of file (pathname of parent folder
of regapp "besclient.exe" & "\__BESData\__Global\Logs" & "\" & (year of it as
string &  month of it as two digits & day_of_month of it as two digits) of (current
date - 1*day) & ".log")) else ("Log file does not exist")
A: Log file does not exist
T: 0.117 ms
I: singular string
```

5. Perform the following steps to create the test condition for the **If** portion of the **`if-then-else`** statement.

   a. Switch to the **(qna)** tab in the Fixlet debugger. Modify the query that you created in Exercise 17, to include only the file portion of the query. Verify that the statement is valid by clicking **Evaluate**.

```
Q: file (pathname of parent folder of regapp “besclient.exe” &
“\__BESData\__Global\Logs” & “\” & (year of it as string & month of it as two
digits & day_of_month of it as two digits) of (current date - 1*day) &
“.log”)
```

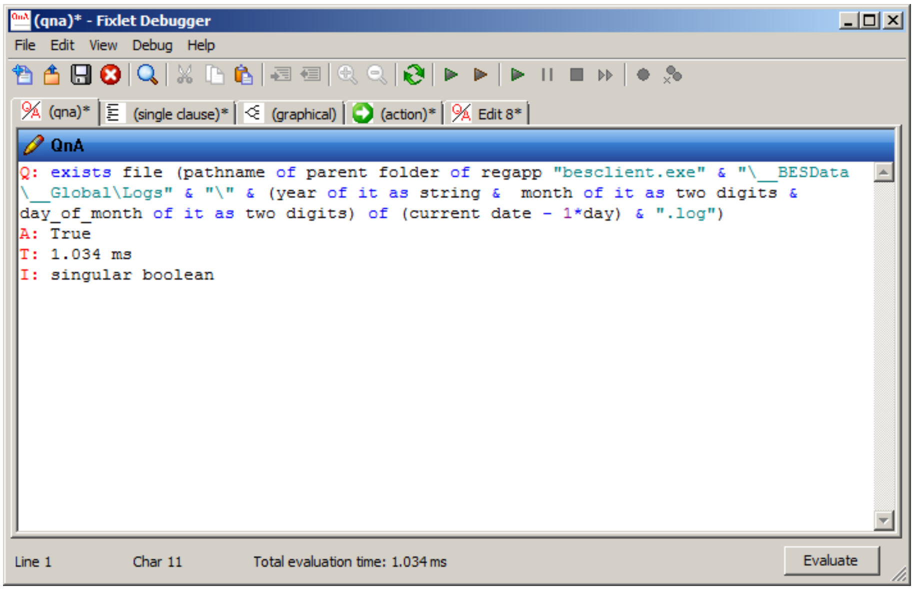


```
Q: file (pathname of parent folder of regapp "besclient.exe" & "\__BESData\__Global
\Logs" & "\" & (year of it as string &  month of it as two digits & day_of_month of
it as two digits) of (current date - 1*day) & ".log")
A: "20160403.log" "" "" "" ""
T: 1.480 ms
I: singular file
```

Line 1 Char 4 Total evaluation time: 1.480 ms

b. Modify the query to test that the file exists and return a Boolean True or False.

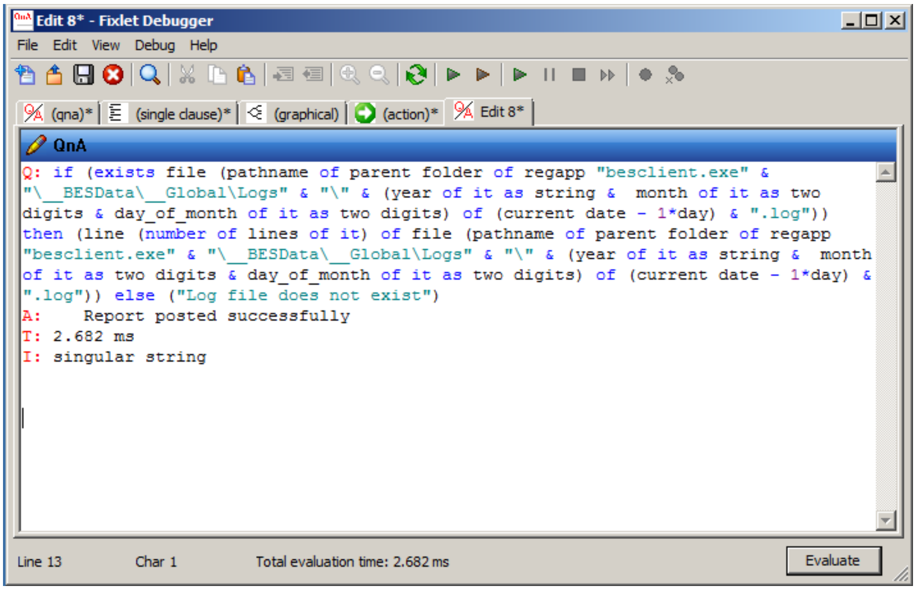
```
Q: exists file (pathname of parent folder of regapp “besclient.exe” &
“\__BESData\__Global\Logs” & “\” & (year of it as string & month of it as two
digits & day_of_month of it as two digits) of (current date - 1*day) &
“.log”)
```



```
Q: exists file (pathname of parent folder of regapp "besclient.exe" & "\__BESData
\__Global\Logs" & "\" & (year of it as string &  month of it as two digits &
day_of_month of it as two digits) of (current date - 1*day) & ".log")
A: True
T: 1.034 ms
I: singular boolean
```

c. Copy this query into the **If** portion of the `if-then-else`statement on the other **QnA** tab.

```
Q: if (exists file (pathname of parent folder of regapp “besclient.exe” &
“\__BESData\__Global\Logs” & “\” & (year of it as string & month of it as two
digits & day_of_month of it as two digits) of (current date - 1*day) &
“.log”)) then (line (number of lines of it) of file (pathname of parent
folder of regapp “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year
of it as string & month of it as two digits & day_of_month of it as two
digits) of (current date - 1*day) & “.log”)) else (“Log file does not exist”)
```



```
Q: if (exists file (pathname of parent folder of regapp "besclient.exe" &
"\__BESData\__Global\Logs" & "\" & (year of it as string &  month of it as two
digits & day_of_month of it as two digits) of (current date - 1*day) & ".log"))
then (line (number of lines of it) of file (pathname of parent folder of regapp
"besclient.exe" & "\__BESData\__Global\Logs" & "\" & (year of it as string &  month
of it as two digits & day_of_month of it as two digits) of (current date - 1*day) &
".log")) else ("Log file does not exist")
A:     Report posted successfully
T: 2.682 ms
I: singular string
```

You now optimize the query by using an **`it-without-a-whose`** clause to eliminate specifying the file multiple times.

6. Place parentheses around the entire **`if-then-else`** statement. Add the keyword **of** after the closing parenthesis and copy the file name portion and then paste it after the **of**.

   At this stage, the query appears as follows:

   ```
   Q: (if (exists file (pathname of parent folder of regapp “besclient.exe” &
   “\__BESData\__Global\Logs” & “\” & (year of it as string & month of it as two
   digits & day_of_month of it as two digits) of (current date - 1*day) & “.log”))
   then (line (number of lines of it) of file (pathname of parent folder of regapp
   “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year of it as string &
   month of it as two digits & day_of_month of it as two digits) of (current date -
   1*day) & “.log”)) else (“Log file does not exist”)) of (pathname of parent
   folder of regapp “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year of
   it as string & month of it as two digits & day_of_month of it as two digits) of
   (current date - 1*day) & “.log”)
   ```

7. Replace each occurrence of the query that represents the file name with the **`it`** construct.

   ```
   Q: (if (exists file it) then (line (number of lines of it) of file it) else
   (“Log file does not exist”)) of (pathname of parent folder of regapp
   “besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year of it as string &
   month of it as two digits & day_of_month of it as two digits) of (current date -
   1*day) & “.log”)
   ```

Alternatively, you can use this query:

```
Q: (if (exists it) then (line (number of lines of it) of it) else (“Log file
does not exist”)) of ( file (pathname of parent folder of regapp
“besclient.exe” & “\__BESData\__Global\Logs” & “\” & (year of it as string &
month of it as two digits & day_of_month of it as two digits) of (current
date - 1*day) & “.log”))
```

Exercise 33 Finding a specific line in a file

In this exercise, you develop a Relevance query that searches the `C:\WINDOWS\system32\drivers\etc\hosts` file for the word *Microsoft*.

The objective for this exercise is to examine the file line objects to determine how to extract this line. No error checking is required.

Hint: Beginning with BigFix version 9 and later, an inspector named **Native System Folder** allows the use of simplified Relevance to handle both 32-bit and 64-bit Windows operating systems.

Example Syntax:

```
Q: file “hosts” of folder “drivers/etc” of native system folder
```

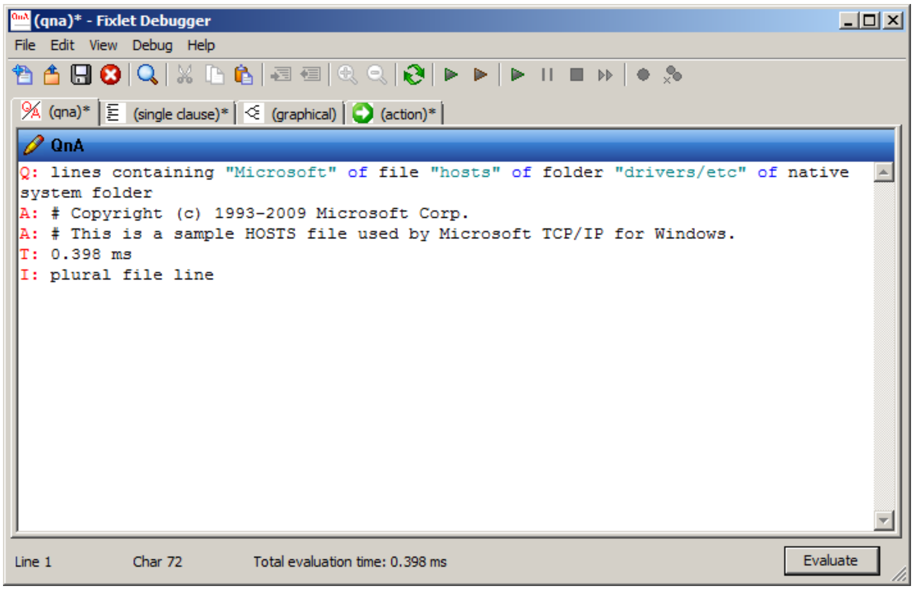
Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. In the text box, enter a query to create a file object for the specific file. Click **Evaluate**.

   ```
   Q: file “hosts” of folder “drivers/etc” of native system folder
   ```

3. Complete the query by finding specific lines in the file that contain the string **Microsoft**.

   ```
   Q: lines containing “Microsoft” of file “hosts” of folder “drivers/etc” of
   native system folder
   ```

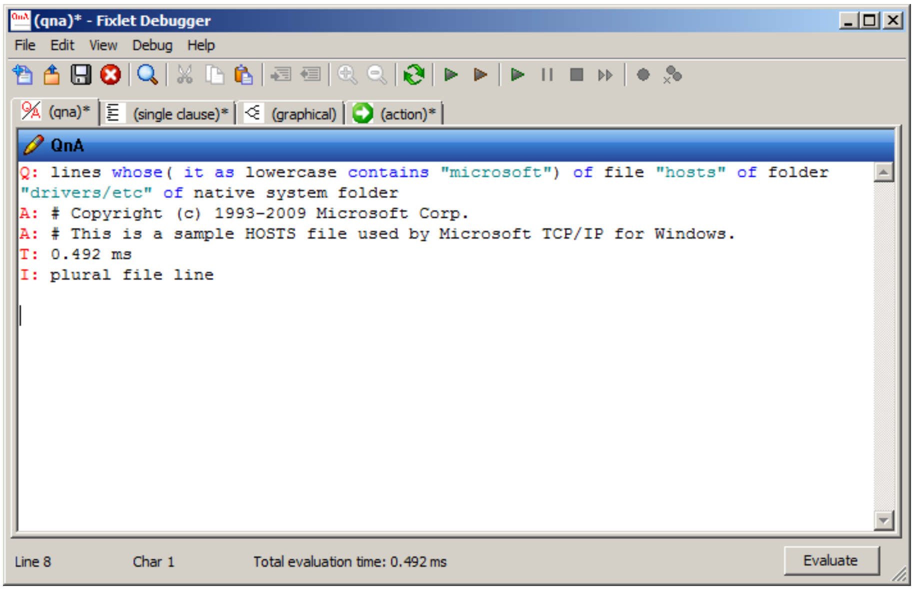


```
Q: lines containing "Microsoft" of file "hosts" of folder "drivers/etc" of native
system folder
A: # Copyright (c) 1993-2009 Microsoft Corp.
A: # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
T: 0.398 ms
I: plural file line
```

Line 1   Char 72   Total evaluation time: 0.398 ms

4. To refer to the file regardless of case, you must convert the file as lowercase, by using an **`it-without-a-whose`** construct.

   `Q: lines whose( it as lowercase contains “microsoft”) of file “hosts” of folder “drivers/etc” of native system folder`



```
Q: lines whose( it as lowercase contains "microsoft") of file "hosts" of folder
"drivers/etc" of native system folder
A: # Copyright (c) 1993-2009 Microsoft Corp.
A: # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
T: 0.492 ms
I: plural file line
```

Line 8 Char 1 Total evaluation time: 0.492 ms

Exercise 34 Performing a WMI query

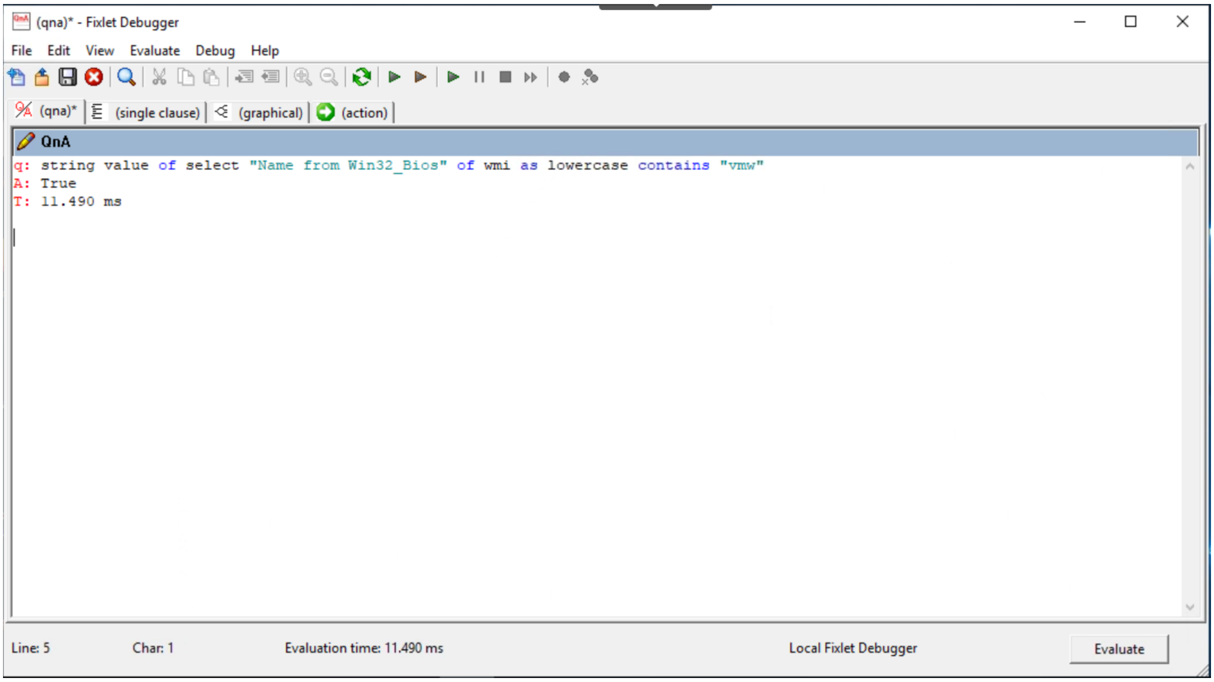
In this exercise, you develop a Relevance query to perform a direct WMI query of the BIOS type by name and return True if you have a VMware BIOS in your workstation. You accomplish this by inspecting the **Name** property of the **Win32_Bios** class in the WMI.

Use these steps to perform the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. In the text box, create a query that uses the WMI inspector to query the system BIOS name.

   ```
   Q: string value of select “Name from Win32_Bios” of wmi as lowercase contains “vmw”
   ```



```
q: string value of select "Name from Win32_Bios" of wmi as lowercase contains "vmw"
A: True
T: 11.490 ms
```

Line: 5 Char: 1 Evaluation time: 11.490 ms Local Fixlet Debugger

Exercise 35 Inspecting client settings

In this exercise, you develop a Relevance query to determine whether the **_BESClient_LastShutdown_Reason** BigFix client setting exists. If the setting exists, you return the value of the setting and if it does not exist, return a **<not set>**.

To complete this exercise, your query must perform the following tasks:

- Use the **client setting** object to determine the value of the setting.
- Use error checking to confirm that the setting exists.
- Test this property by creating or deleting settings in the registry.

You can develop your solution by using syntax similar to the following one:

- Settings of client
- Value of setting **“_BESClient_LastShutdown_Reason”** of client

Perform the following steps to create the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.

2. In the text box, create a query to obtain the value of the specific BigFix client setting.

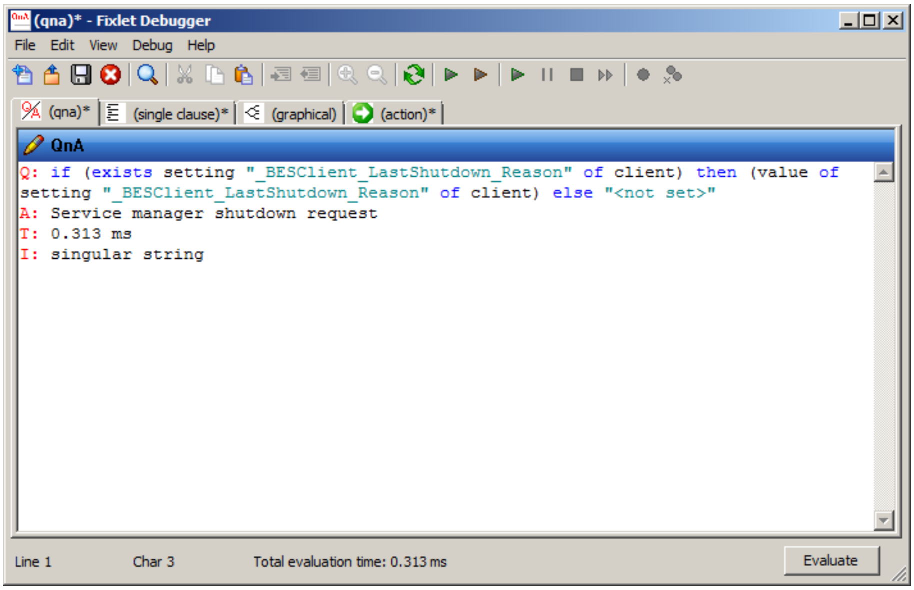
   ```
   Q: value of setting “_BESClient_LastShutdown_Reason” of client
   ```

3. You should not assume that the client setting exists. You must include a test for the existence of the client setting.

   ```
   Q: exists setting “_BESClient_LastShutdown_Reason” of client
   ```

4. Complete the query by using an `if-then-else` construct. The test query from Step 3 becomes the **If** portion of the `if-then-else` construct. The query from Step 2 becomes the **Then** portion of the `if-then-else` construct, and the string **“<not set>“** becomes the **Else** portion.

   ```
   Q: if (exists setting “_BESClient_LastShutdown_Reason” of client) then (value of
   setting “_BESClient_LastShutdown_Reason” of client) else “<not set>“
   ```



```
Q: if (exists setting "_BESClient_LastShutdown_Reason" of client) then (value of
setting "_BESClient_LastShutdown_Reason" of client) else "<not set>"
A: Service manager shutdown request
T: 0.313 ms
I: singular string
```

Exercise 36 Using a tuple construct

In this exercise, you develop a Relevance query that shows all files and their versions in the specified Windows folder. You then modify the query to show only those files with a version.

Complete this exercise by using the following methods:

- You must use a tuple construct.
- You must use error checking to avoid errors if an element of the tuple does not exist.

Perform these steps to create the query:

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. In the text box, create a query to list all the files in the Windows folder.

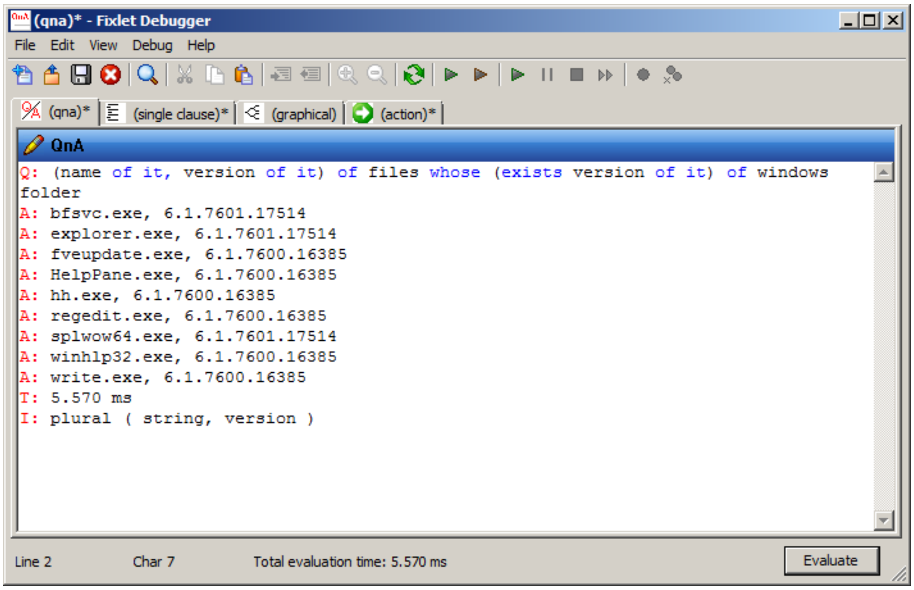
   ```
   Q: names of files of windows folder
   ```

3. Extend the query to include the versions of the files in the Windows folder only if a version exists. You use an **`if-then-else`** construct to perform this test. If a file does not have a version, you must return the string **“No version”**.

   ```
   Q: (name of it, (if (exists version of it) then (version of it as string) else
   (“No version”))) of files of windows folder
   ```

4. Use a **`whose-it`** construct to filter the output and show only the files that have a version.

   ```
   Q: (name of it, version of it) of files whose (exists version of it) of windows
   folder
   ```



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause)* (graphical) (action)*
QnA
Line 2
Char 7
Total evaluation time: 5.570 ms
Evaluate

```
Q: (name of it, version of it) of files whose (exists version of it) of windows
folder
A: bfsvc.exe, 6.1.7601.17514
A: explorer.exe, 6.1.7601.17514
A: fveupdate.exe, 6.1.7600.16385
A: HelpPane.exe, 6.1.7600.16385
A: hh.exe, 6.1.7600.16385
A: regedit.exe, 6.1.7600.16385
A: splwow64.exe, 6.1.7601.17514
A: winhlp32.exe, 6.1.7600.16385
A: write.exe, 6.1.7600.16385
T: 5.570 ms
I: plural ( string, version )
```

Exercise 37 (Optional) Using regular expressions to remove control characters in string expressions

In this exercise, you create a regular expression (RegEx) to remove control characters from a string.

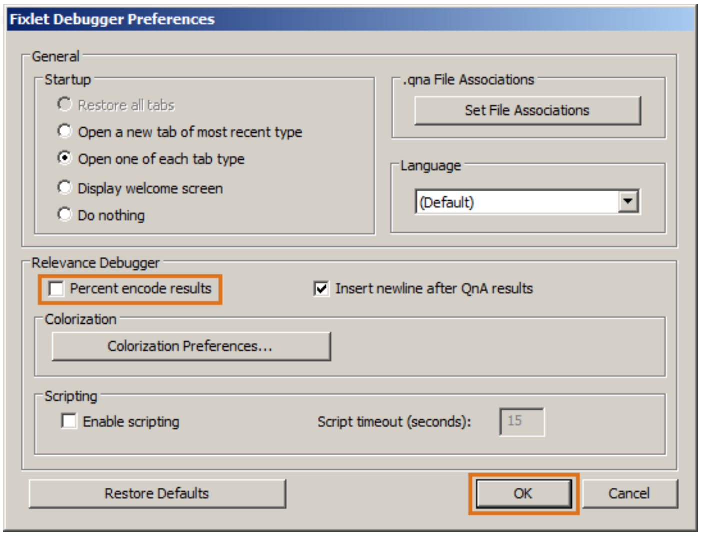
Use the following string to test the query:

```
This is a test string%09 with some control characters%07.
```

Perform the following steps to create this query:

1. Return to the Fixlet Debugger, and select the **(qna)** tab.
2. Perform the following steps to configure the Fixlet Debugger so that it does not show the control characters as **Percent Encode Results**:
   a. Select **File > Preferences**.

   b. Clear the **Percent Encode Results** option and click **OK**.



Fixlet Debugger Preferences

General

Startup

○ Restore all tabs
○ Open a new tab of most recent type
● Open one of each tab type
○ Display welcome screen
○ Do nothing

.qna File Associations

Language: (Default)

Relevance Debugger

☐ Percent encode results
☑ Insert newline after QnA results

Colorization

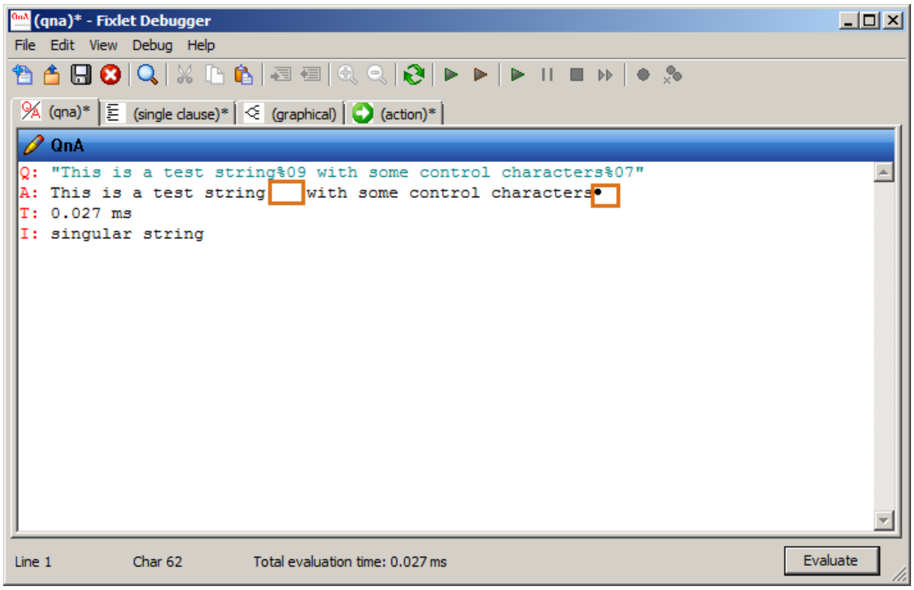
Scripting

☐ Enable scripting
Script timeout (seconds): 15

3. Enter the query string that contains the control characters and click **Evaluate**.

```
Q: “This is a test string%09 with some control characters%07”
```

Note: Notice that an extended space following the word *string* results from the **TAB** character. A small box is shown at the end of the string that identifies a non-displayable character.



```
Q: "This is a test string%09 with some control characters%07"
A: This is a test string    with some control characters•
T: 0.027 ms
I: singular string
```

Line 1 Char 62 Total evaluation time: 0.027 ms

You now build the RegEx filter using the **`match <regular expression> of <string>`** inspector.

4. In **QnA**, start a new line. Enter the required RegEx filter to remove control characters. You must ignore the control characters using the **^** symbol and exclude all control characters *%01-%1F*. Remember that the **^** character must be within the string component.

   ```
   Q: regex “[^%01-%1F]”
   ```

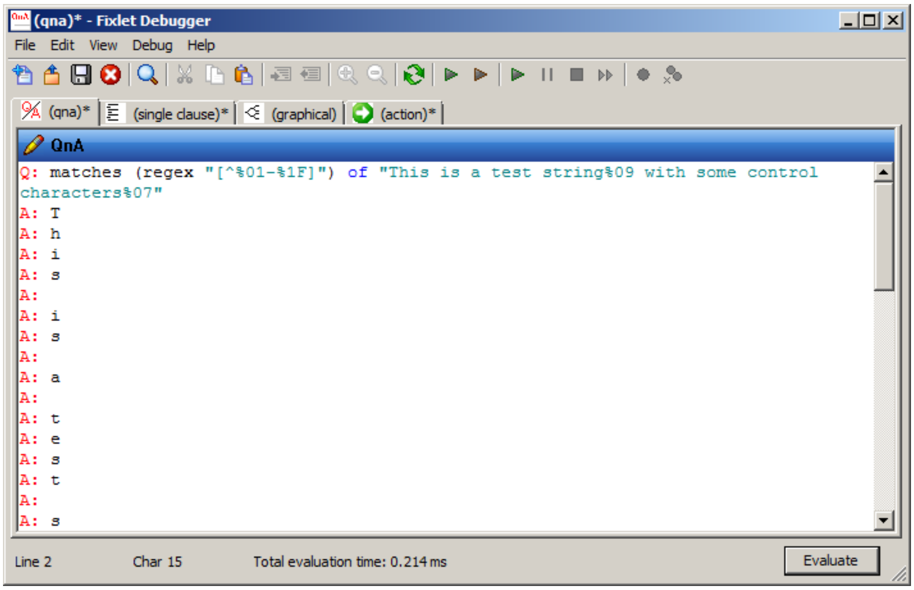
5. Place the entire RegEx expression inside parentheses and add the string to the end of the expression.

   ```
   Q: (regex “[^%01-%1F]”) of “This is a test string%09 with some control
   characters%07”
   ```

6. Complete the query by adding the **match** keyword to the beginning of the expression.

Hint: The match keyword must be the plural form **matches** to account for multiple instances.

```
Q: matches (regex “[^%01-%1F]”) of “This is a test string%09 with some control
characters%07”
```



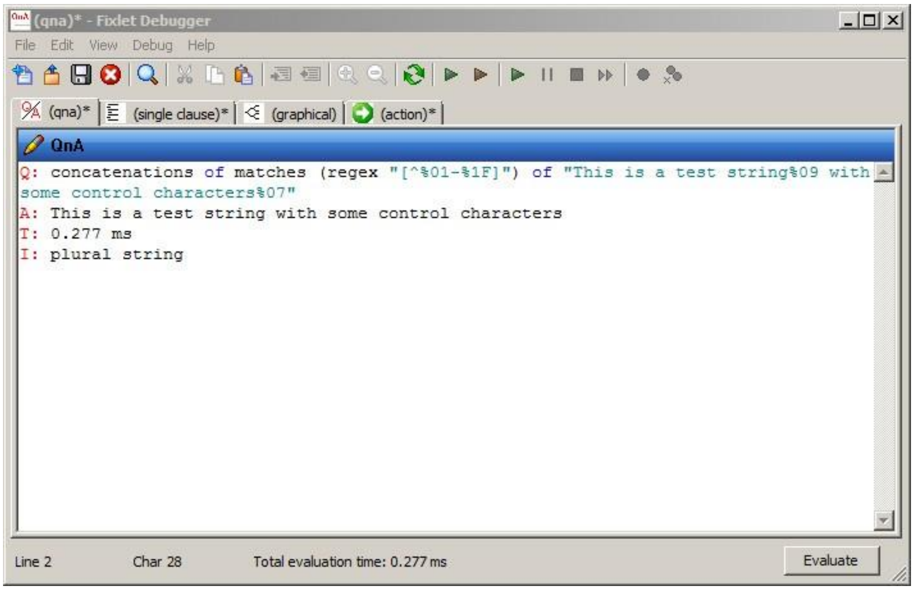
```
Q: matches (regex "[^%01-%1F]") of "This is a test string%09 with some control
characters%07"
A: T
A: h
A: i
A: s
A: 
A: i
A: s
A: 
A: a
A: 
A: t
A: e
A: s
A: t
A: 
A: s
```

Line 2   Char 15   Total evaluation time: 0.214 ms

Multiple responses that are composed of each character of the string are shown. Using the **`match <regular expression> of <string>`** inspector causes each match to be shown as an answer.

7. Use the string construct **`concatenation of <string>`** to concatenate each response into a single string.

   ```
   Q: concatenations of matches (regex “[^%01-%1F]”) of “This is a test string%09
   with some control characters%07”
   ```



```
Q: concatenation of matches (regex "[^%01-%1F]") of "This is a test string%09 with
some control characters%07"
A: This is a test string with some control characters
T: 0.277 ms
I: plural string
```

   The string is shown without the control characters.

Exercise 38 (Optional) Using regular expressions to extract portions of a string

In this exercise, you create a regular expression to extract portions of a string. It is often easier to use RegEx than multiple **`preceding texts of preceding text`** constructs when parsing a string, such as in the following example:

```
asdf:adsf:aedf:aadf
```

For this exercise, you create a Relevance query to parse an existing file named C:\Windows\Regex.txt that contains the following lines:

```
u686689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
u870235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
```

You then display the following output:

```
u686689, Donald Duck
u870235, Mickey Mouse
```

To create this query, you must use the **`parenthesized part <integer> of <regular expression match>`** inspector. This inspector returns the nth parenthetical specified by **<integer>** in the regular expression match.

Perform the following steps to create the query:

1. On the **BESFNDWIN10** virtual machine, locate the file Regex.txt in C:\Windows.

2. Return to the Fixlet Debugger and select the **(qna)** tab.

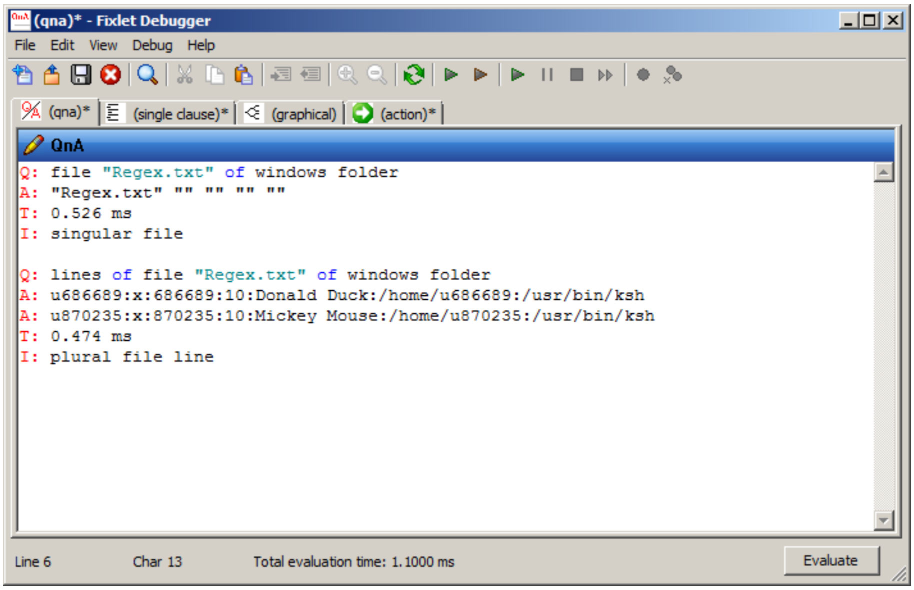
3. Begin by writing a query to locate the `Regex.txt`file in the Windows folder. Click **Evaluate**.

   ```
   Q: file “Regex.txt” of windows folder
   ```

   The query is successfully evaluated and it returns no errors.

4. Below the query from Step 3, create another query to list all of the lines of the `Regex.txt`file.

   ```
   Q: lines of file “Regex.txt” of windows folder
   ```



```
Q: file "Regex.txt" of windows folder
A: "Regex.txt" "" "" "" ""
T: 0.526 ms
I: singular file

Q: lines of file "Regex.txt" of windows folder
A: u686689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: u870235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
T: 0.474 ms
I: plural file line
```

Line 6 Char 13 Total evaluation time: 1.1000 ms

5. Perform the following steps to build the RegEx expression to match the seven components of each line delimited by the colon (:).

   a. Begin by creating the filter based on the colon delimiter. For each line of the file, you look for any character with the **:** as the separator with the **(.+)** filter.

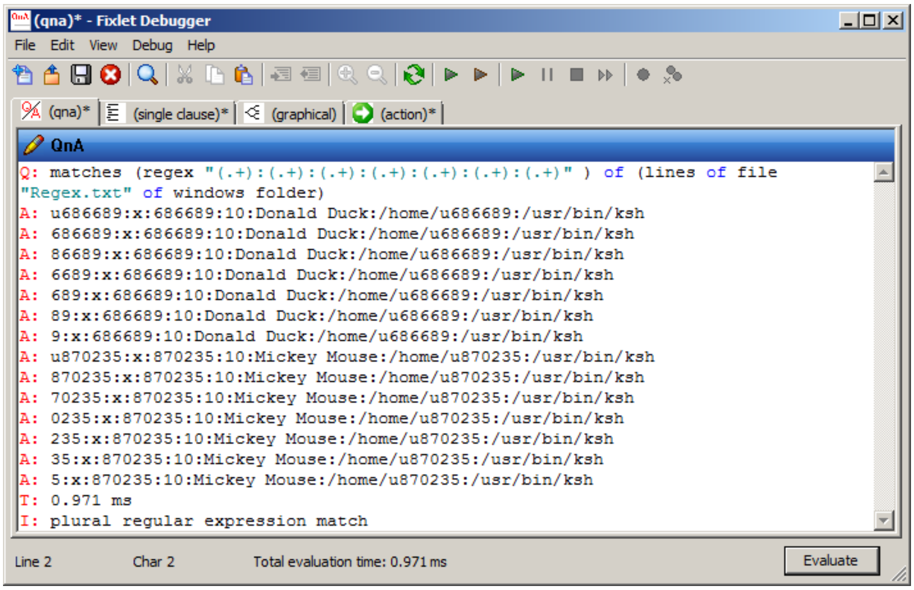
   ```
   regex (“(.+):(.+):(.+):(.+):(.+):(.+):(.+)” )
   ```

   b. Add the RegEx expression to the query to return the matches found in the `Regex.txt`file.

   ```
   Q: matches (regex “(.+):(.+):(.+):(.+):(.+):(.+):(.+)” ) of (lines of file
   “Regex.txt” of windows folder)
   ```

Hint: When you evaluate this query, **QnA** displays multiple matches for each line in the file. This is caused by the following circumstances:

- If multiple matches exist, the BigFix client returns all matches.
- By default, the **(.+)** token with RegEx performs a **greedy** search. Refer to the **Limiting Repetition** at http://www.regular-expressions.info/repeat.html for additional information.



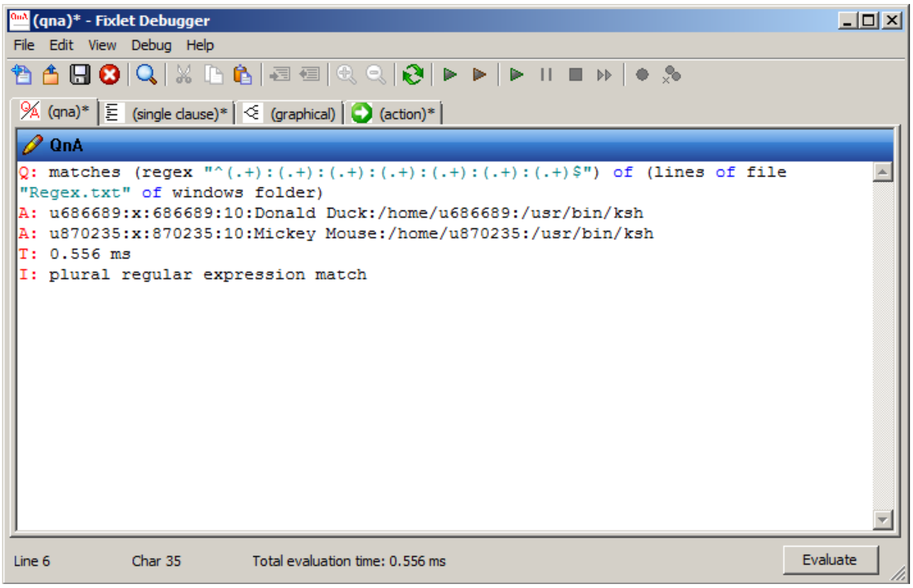
```
Q: matches (regex "(.+):(.+):(.+):(.+):(.+):(.+):(.+)" ) of (lines of file
"Regex.txt" of windows folder)
A: u686689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: 686689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: 86689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: 6689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: 689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: 89:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: 9:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: u870235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
A: 870235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
A: 70235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
A: 0235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
A: 235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
A: 35:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
A: 5:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
T: 0.971 ms
I: plural regular expression match
```

Line 2   Char 2   Total evaluation time: 0.971 ms

6. To limit the matches, you must force RegEx to filter the content from the beginning to the end of the line. To accomplish this, you use the carat (**^**) symbol to indicate the start of the line

followed by the expression. Finally, you use the dollar sign (**$**) after the expression to indicate the end of the line.

```
Q: matches (regex “^(.+):(.+):(.+):(.+):(.+):(.+):(.+)$”) of (lines of file
“Regex.txt” of windows folder)
```

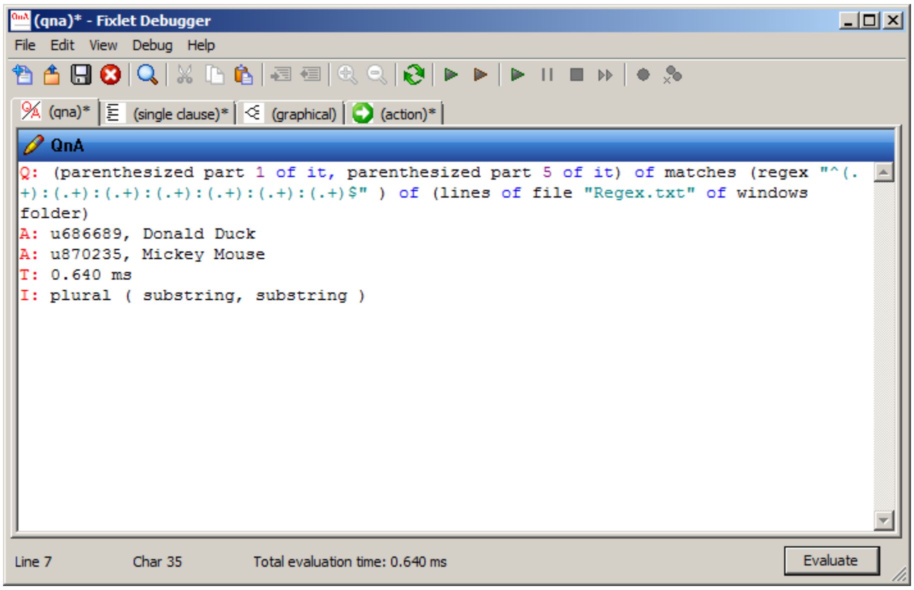


```
Q: matches (regex "^(.+):(.+):(.+):(.+):(.+):(.+):(.+)$") of (lines of file
"Regex.txt" of windows folder)
A: u686689:x:686689:10:Donald Duck:/home/u686689:/usr/bin/ksh
A: u870235:x:870235:10:Mickey Mouse:/home/u870235:/usr/bin/ksh
T: 0.556 ms
I: plural regular expression match
```

When you evaluate this expression, you see the same output that was returned from the query in Step 4 above. The difference is that you can now extract portions of these lines by using the **`parenthesized part <integer> of <regular expression match>`** inspector.

7. Modify the query to use a **`tuple`** construct to extract only the first and fifth element of each line of the `Regex.txt` file.

```
Q: (parenthesized part 1 of it, parenthesized part 5 of it) of matches (regex
“^(.+):(.+):(.+):(.+):(.+):(.+):(.+)$” ) of (lines of file “Regex.txt” of windows
folder)
```



```
Q: (parenthesized part 1 of it, parenthesized part 5 of it) of matches (regex "^(.
+):(.+):(.+):(.+):(.+):(.+):(.+)$" ) of (lines of file "Regex.txt" of windows
folder)
A: u686689, Donald Duck
A: u870235, Mickey Mouse
T: 0.640 ms
I: plural ( substring, substring )
```

Exercise 39 Using user objects

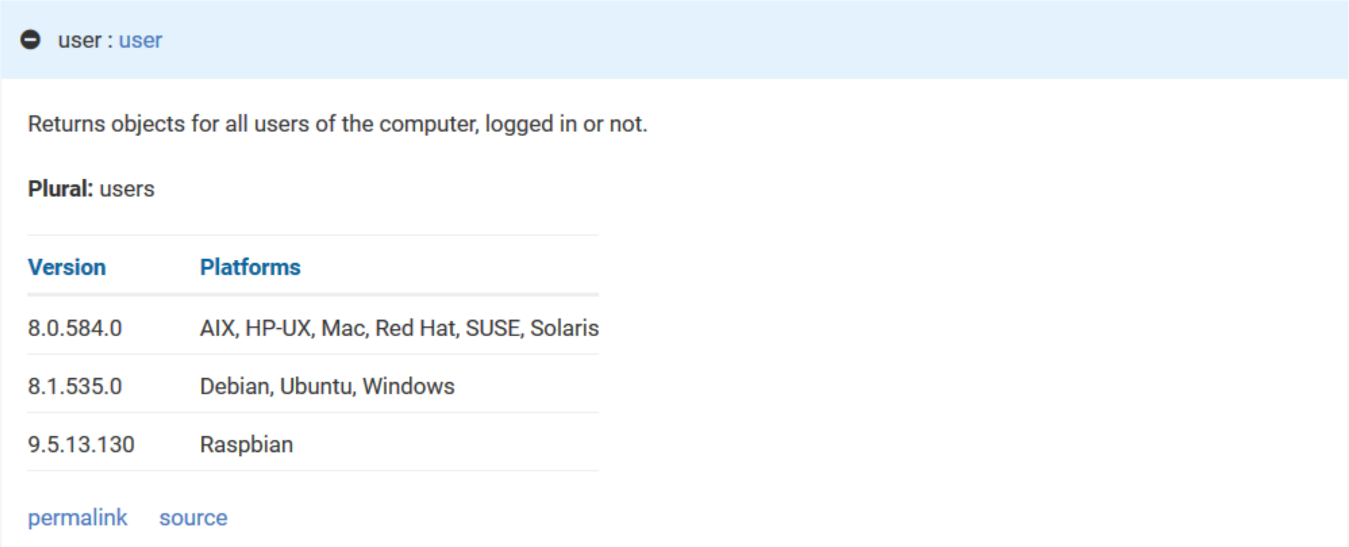
In this exercise, you develop various Relevance queries that illustrate the use of the user objects within Relevance.

Hint: Refer to the **Reference Documentation** section of the BigFix Developer web site for additional information:

https://developer.bigfix.com/relevance/reference/user.html

Perform the following steps to create this query:

1. Switch to the **BESFNDWIN10** virtual machine. Log in as `tecuser` with a password credentials provided at the beginning of this document.
2. From the Windows Start menu, select **All Programs > BigFix > BigFix Fixlet Debugger**.
3. Select the **(qna)** tab.
4. Review the descriptions of the **Creation Methods** for the User Objects inspectors on the developer.bigfix.com site.



user : user

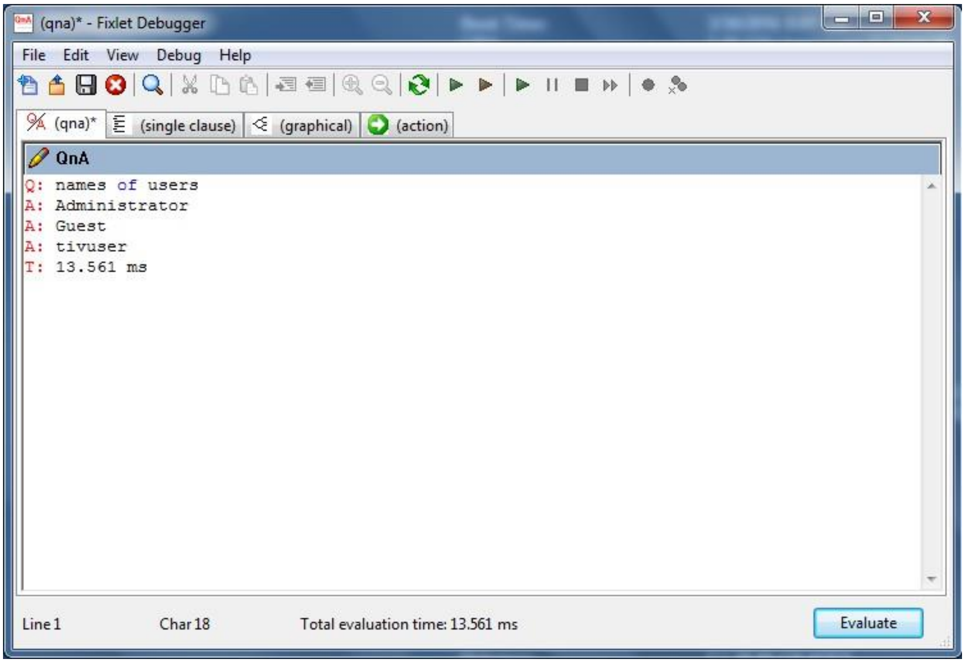
Returns objects for all users of the computer, logged in or not.

Plural: users

Version	Platforms
8.0.584.0	AIX, HP-UX, Mac, Red Hat, SUSE, Solaris
8.1.535.0	Debian, Ubuntu, Windows
9.5.13.130	Raspbian

5. Return to the Fixlet Debugger on the **BESFNDWIN10** virtual machine. Create a query to display the names for all the users of this system, regardless of whether they are logged in.

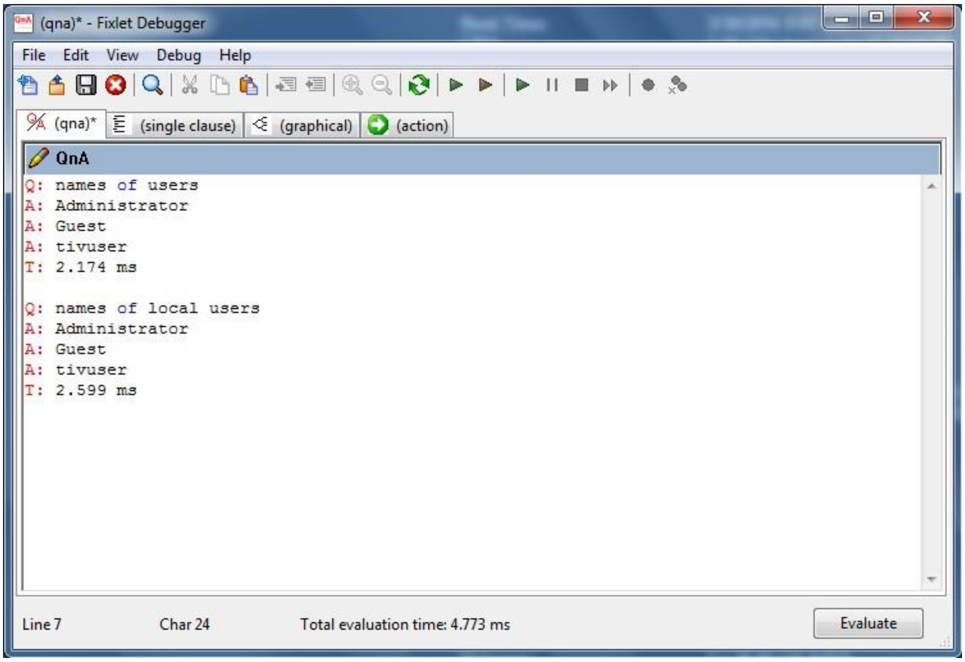
```
Q: names of users
```



```
Q: names of users
A: Administrator
A: Guest
A: tivuser
T: 13.561 ms
```

6. Create a new query below the existing query on the **(qna)** tab to show only the local accounts of a domain system. You use the **local users** object.

   ```
   Q: names of local users
   ```

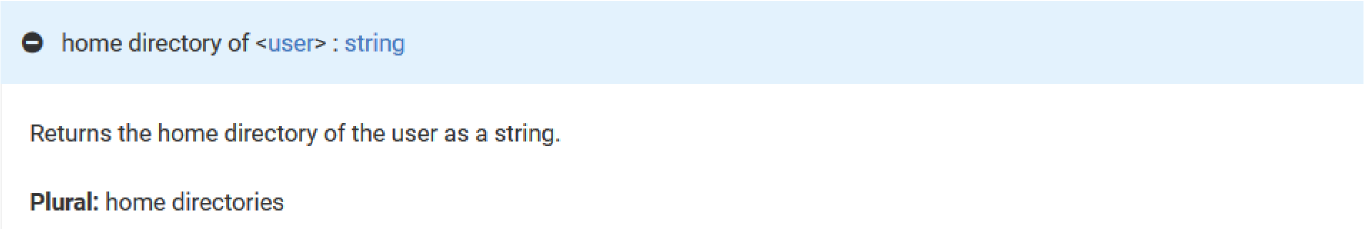


```
Q: names of users
A: Administrator
A: Guest
A: tivuser
T: 2.174 ms

Q: names of local users
A: Administrator
A: Guest
A: tivuser
T: 2.599 ms
```

   Notice that the same list of users is displayed.

You now use the **home directory of <user>** property to examine the home directories of the users.

7. Review the documentation for the **home directory of <user>** property.



home directory of <user> : string

Returns the home directory of the user as a string.

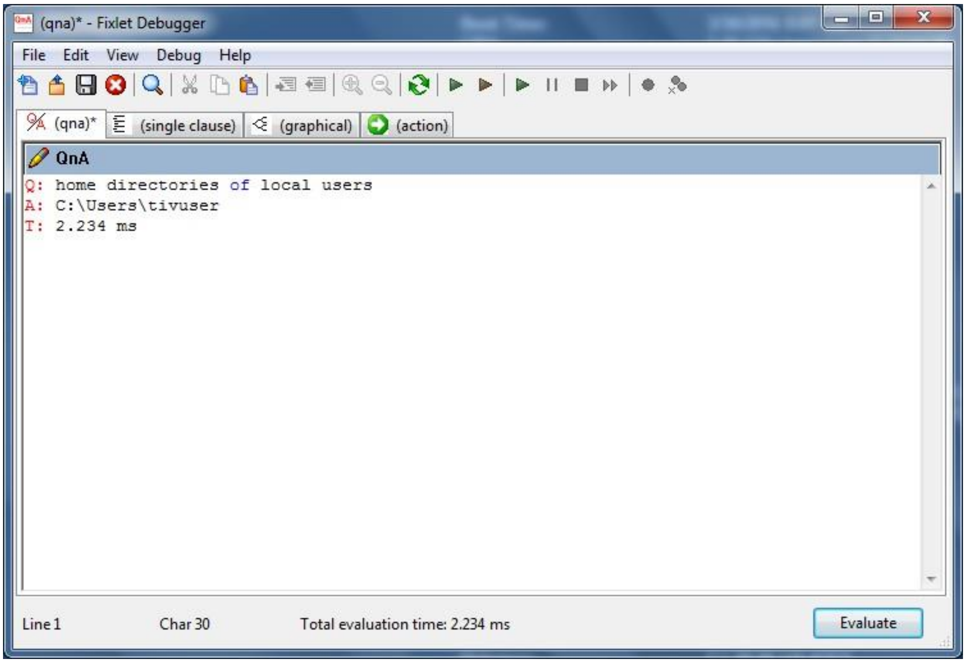
Plural: home directories

8. Return to the Fixlet Debugger on the virtual machine.

9. Clear the previous queries on the **(qna)** tab.

10. Create a query to examine the home directories for the local users.

    ```
    Q: home directories of local users
    ```

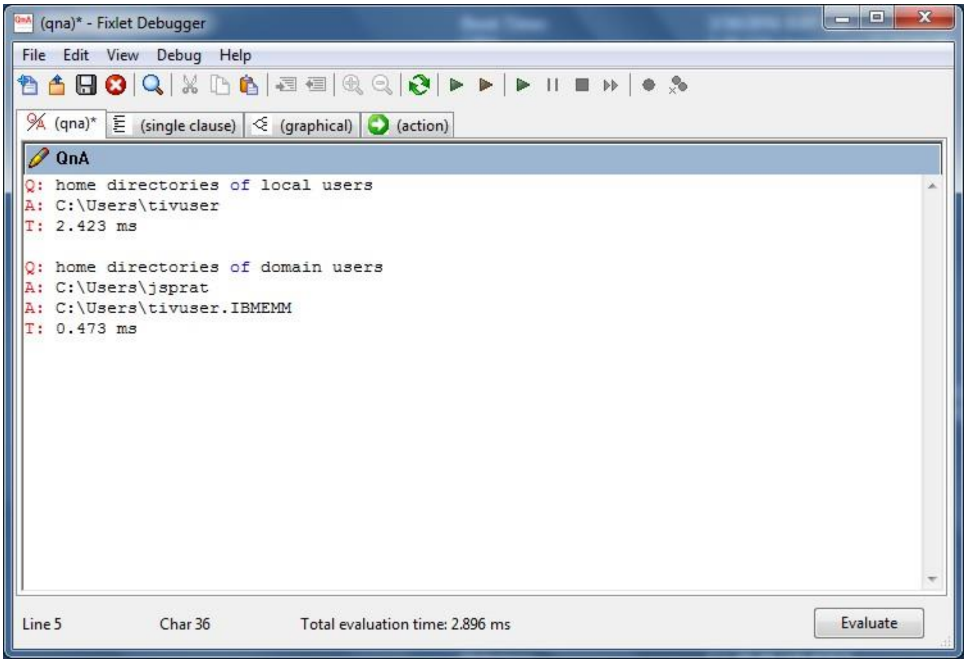


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)
QnA
Line 1
Char 30
Total evaluation time: 2.234 ms
Evaluate

```
Q: home directories of local users
A: C:\Users\tivuser
T: 2.234 ms
```

The query returns the home directory path for local users that have previously logged in to the system.

```
Q: home directories of domain users
```



```
Q: home directories of local users
A: C:\Users\tivuser
T: 2.423 ms

Q: home directories of domain users
A: C:\Users\jsprat
A: C:\Users\tivuser.IBMEMM
T: 0.473 ms
```

The query returns the home directory path for each domain user that has logged in to the system.

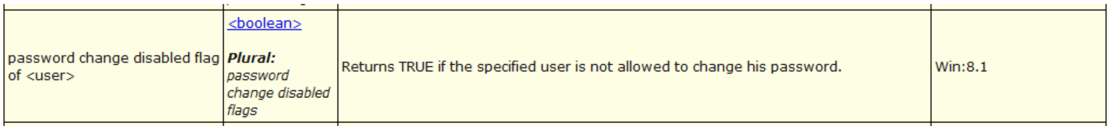
Hint: Only the home directory paths of users where a current profile that exists on the system are returned. When users log in to a system, Windows creates a profile for them and then creates their home directory. Therefore, local or domain users who have not previously logged in to the system have no current profile.

You can also query a specific user by following the object with the desired user name. For example, the **`home directory of domain user "tecuser"`** Relevance clause returns the home directory of the specified user.

You now use other properties associated with the **`User`** object to build a **`tuple`** construct that evaluates the name of the local users and whether the **password change disabled flag** for each local user is set.

13. Return to the Fixlet Debugger and clear the previous queries on the **(qna)** tab.

14. Review the **password change disabled flag of <user>** property of the User object.



password change disabled flag of <user>	<boolean> ***Plural:*** *password change disabled flags*	Returns TRUE if the specified user is not allowed to change his password.	Win:8.1

15. Return to the Fixlet Debugger in the **BESFNDWIN10** image and create the initial query to retrieve the name of each local user.

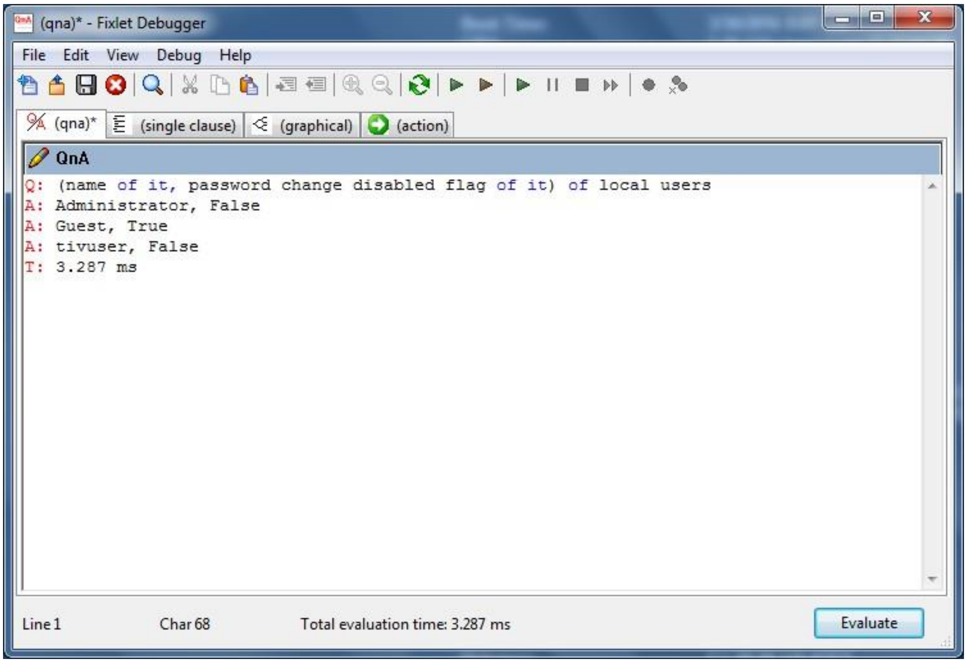
```
Q: names of local users
```

16. Modify this query to use a **`whose-without-an-it`** construct.

```
Q: (name of it) of local users
```

17. Modify the query to create the **`tuple`** construct to retrieve the **user name** and the **password change disabled flag** for each local user.

```
Q: (name of it, password change disabled flag of it) of local users
```

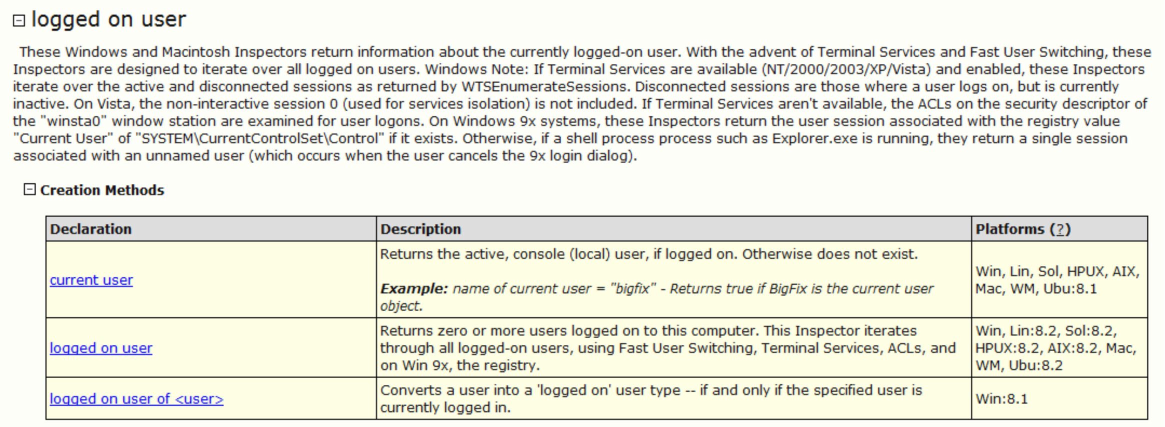


```
Q: (name of it, password change disabled flag of it) of local users
A: Administrator, False
A: Guest, True
A: tivuser, False
T: 3.287 ms
```

Line 1   Char 68   Total evaluation time: 3.287 ms

You now explore the use of the **logged on user** object.

18. Review the **Creation Methods** for the **logged on user** object.



⊟ logged on user

These Windows and Macintosh Inspectors return information about the currently logged-on user. With the advent of Terminal Services and Fast User Switching, these Inspectors are designed to iterate over all logged on users. Windows Note: If Terminal Services are available (NT/2000/2003/XP/Vista) and enabled, these Inspectors iterate over the active and disconnected sessions as returned by WTSEnumerateSessions. Disconnected sessions are those where a user logs on, but is currently inactive. On Vista, the non-interactive session 0 (used for services isolation) is not included. If Terminal Services aren't available, the ACLs on the security descriptor of the "winsta0" window station are examined for user logons. On Windows 9x systems, these Inspectors return the user session associated with the registry value "Current User" of "SYSTEM\CurrentControlSet\Control" if it exists. Otherwise, if a shell process process such as Explorer.exe is running, they return a single session associated with an unnamed user (which occurs when the user cancels the 9x login dialog).

⊟ **Creation Methods**

Declaration	**Description**	**Platforms (?)**
current user	Returns the active, console (local) user, if logged on. Otherwise does not exist. ***Example:*** *name of current user = "bigfix" - Returns true if BigFix is the current user object.*	Win, Lin, Sol, HPUX, AIX, Mac, WM, Ubu:8.1
logged on user	Returns zero or more users logged on to this computer. This Inspector iterates through all logged-on users, using Fast User Switching, Terminal Services, ACLs, and on Win 9x, the registry.	Win, Lin:8.2, Sol:8.2, HPUX:8.2, AIX:8.2, Mac, WM, Ubu:8.2
logged on user of <user>	Converts a user into a 'logged on' user type -- if and only if the specified user is currently logged in.	Win:8.1

Note: As with the **user** object, the **logged on user** property provides various properties that are access using Relevance.

In this section, you create a query to return the name of the currently logged on user.

19. Return to the Fixlet Debugger on the **BESFNDWIN10** virtual machine.

20. Clear the existing queries on the **(qna)** tab of the Fixlet Debugger.

21. Create a query to list the names of all users that are currently logged in to this system with the **logged on user** object.

Hint: The **logged on user** object displays every logged on user, including users that are logged in to the system through the terminal server.

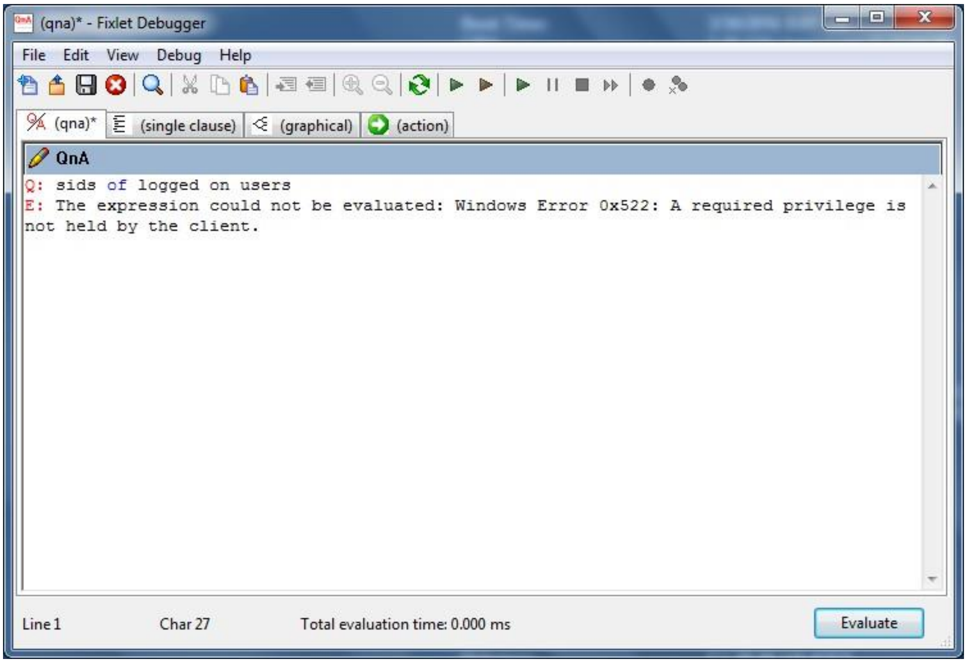
```
Q: names of logged on users
```

22. Determine the Security Identifier (SID) of the user named **tecuser**.

Note: An SID is a unique, immutable identifier of a user, user group, or other security principal. A security principal has a single SID for life, and all properties of the principal, including its name, are associated with the SID. This design allows a principal to be renamed. For example, you can rename a user from John to Jane without affecting the security attributes of objects that refer to the principal.

```
Q: sids of logged on users
```

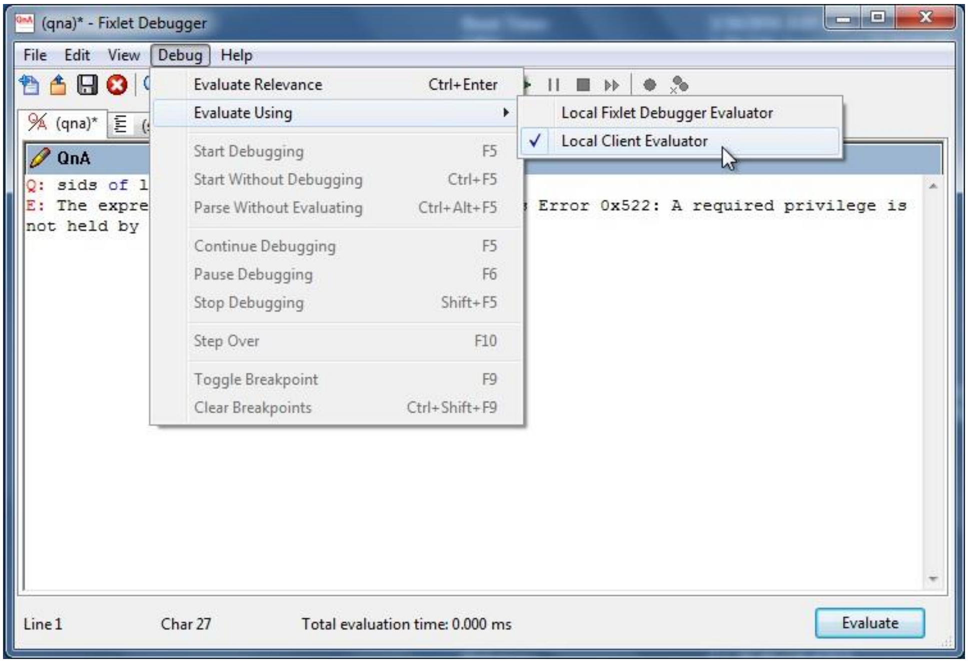
The Fixlet Debugger returns an error. This error occurs because the Fixlet Debugger does not have the appropriate rights to enumerate the SIDs.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)
QnA
Evaluate

```
Q: sids of logged on users
E: The expression could not be evaluated: Windows Error 0x522: A required privilege is
not held by the client.
```

Line 1 Char 27 Total evaluation time: 0.000 ms

23. Reconfigure the Fixlet Debugger to use the BigFix client to evaluate the results by selecting **Debug > Evaluate using > Local Client Evaluator**.

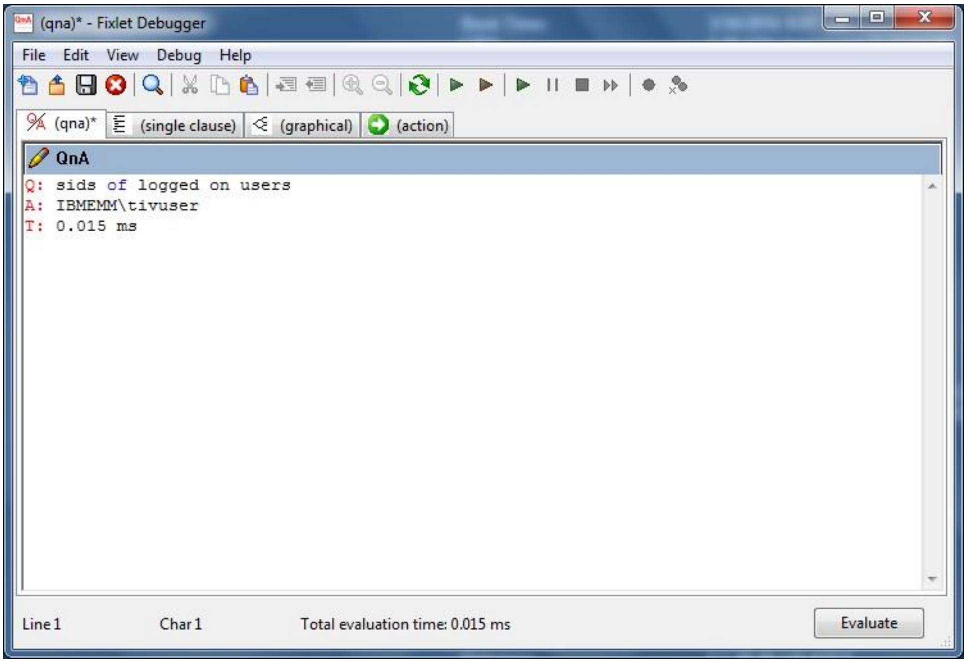


```
Q: sids of l
E: The expre                                   Error 0x522: A required privilege is
not held by
```

Line 1 Char 27 Total evaluation time: 0.000 ms

24. Click **Evaluate** to perform the query using the BigFix client.

Note: The response takes much longer when you use the BigFix client to evaluate the Relevance queries. The query must be passed to the BigFix client for evaluation and then returned to the Fixlet Debugger.

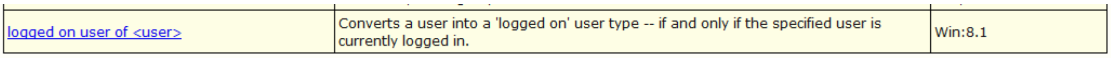


```
Q: sids of logged on users
A: IBMEMM\tivuser
T: 0.015 ms
```

Hint: Occasionally, the Fixlet Debugger might respond with the following error: **An error occurred while communicating with the client**. If this happens, stop and then start the **BESClient** service

You now use Relevance to determine whether a specific user is logged in to the system. This might be useful if you want to create an action that is applicable only when a specific user logs in to a system.

25. Review the creation method for the **logged on user** object named **logged on user of <user>**.



logged on user of <user>	Converts a user into a 'logged on' user type -- if and only if the specified user is currently logged in.	Win:8.1

A user must be specified when calling this method.

26. Return to the Fixlet Debugger on the **BESFNDWIN10** virtual machine.

27. Clear the existing queries on the **(qna)** tab of the Fixlet Debugger.

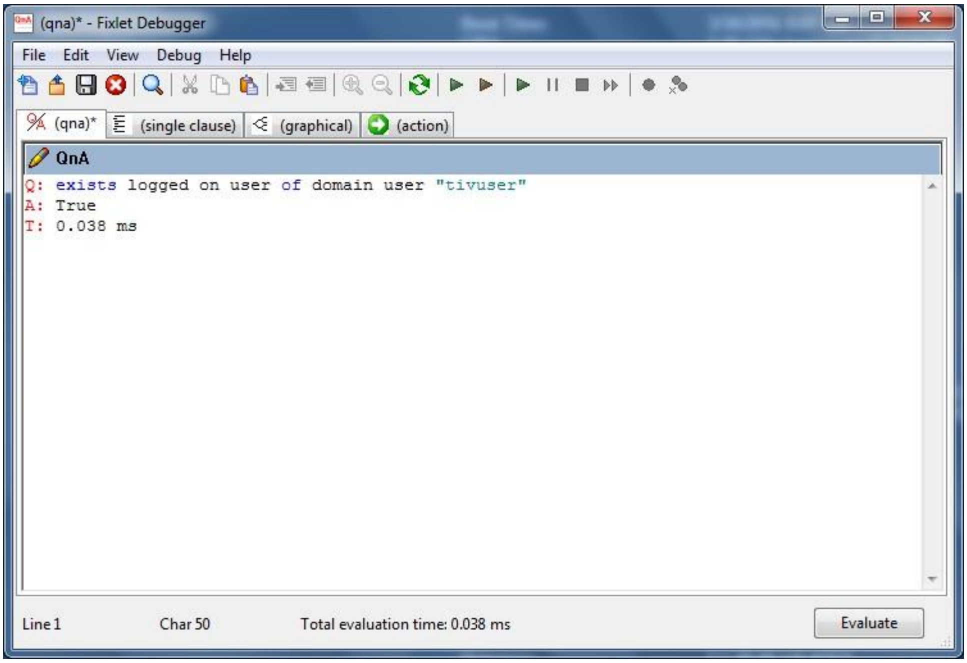
28. Create a query that checks to see whether any logged-on user exists.

```
Q: exists logged on user
```

29. Expand the query to return a Boolean True or false if the domain user named **“tecuser”** is logged on.

```
Q: exists logged on user of domain user “tecuser”
```

30. Click **Evaluate** to view the results of the query.

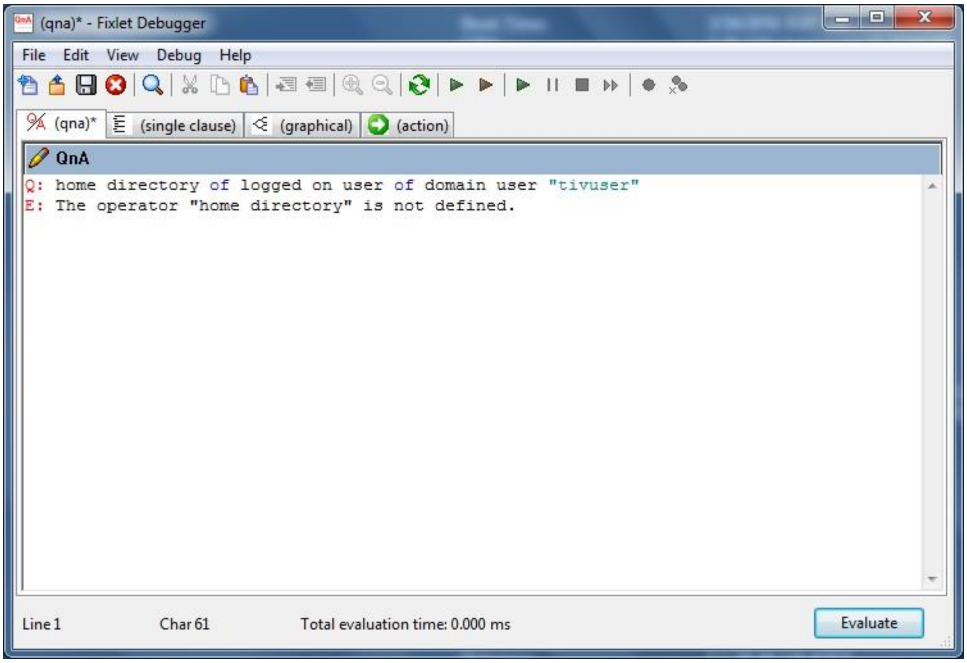


```
Q: exists logged on user of domain user "tivuser"
A: True
T: 0.038 ms
```

Note: To properly perform this query, the Fixlet Debugger must evaluate the Relevance statement using the BigFix Client evaluator as it was configured in Step 23 on page 3-70.

31. Modify the query to attempt to return the **home directory** of the logged on user instead of checking whether the user is logged on.

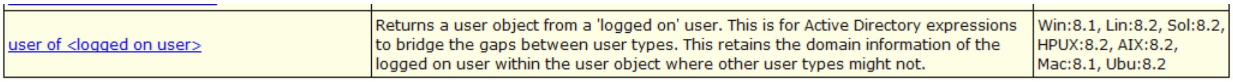
    `Q: home directory of logged on user of domain user “tecuser”`



```
Q: home directory of logged on user of domain user "tivuser"
E: The operator "home directory" is not defined.
```

This query generates an error.

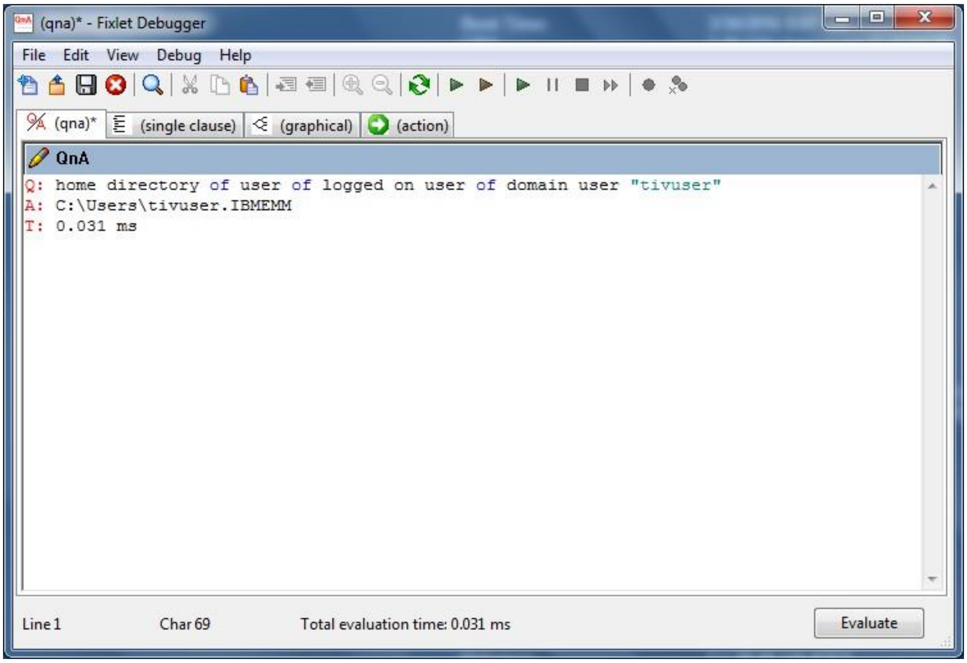
Hint: The reason that this query generates an error is because the **home directory of <user>** property is not available with the **logged on user** object. It is associated with the **user of <logged on user>** creation method for the **user** object.



user of <logged on user>	Returns a user object from a 'logged on' user. This is for Active Directory expressions to bridge the gaps between user types. This retains the domain information of the logged on user within the user object where other user types might not.	Win:8.1, Lin:8.2, Sol:8.2, HPUX:8.2, AIX:8.2, Mac:8.1, Ubu:8.2

32. Modify the query to include the **user of domain user <user>** method.

```
Q: home directory of user of logged on user of domain user “tecuser”
```



```
Q: home directory of user of logged on user of domain user "tivuser"
A: C:\Users\tivuser.IBMEMM
T: 0.031 ms
```

Unit 4 Action Script Exercises

You perform these exercises by using the BigFix Fixlet Debugger to develop Relevance queries and action scripts. This tool is installed on the **student** virtual machine. To start the tool, click **Start > BigFix > BigFix Fixlet Debugger**.

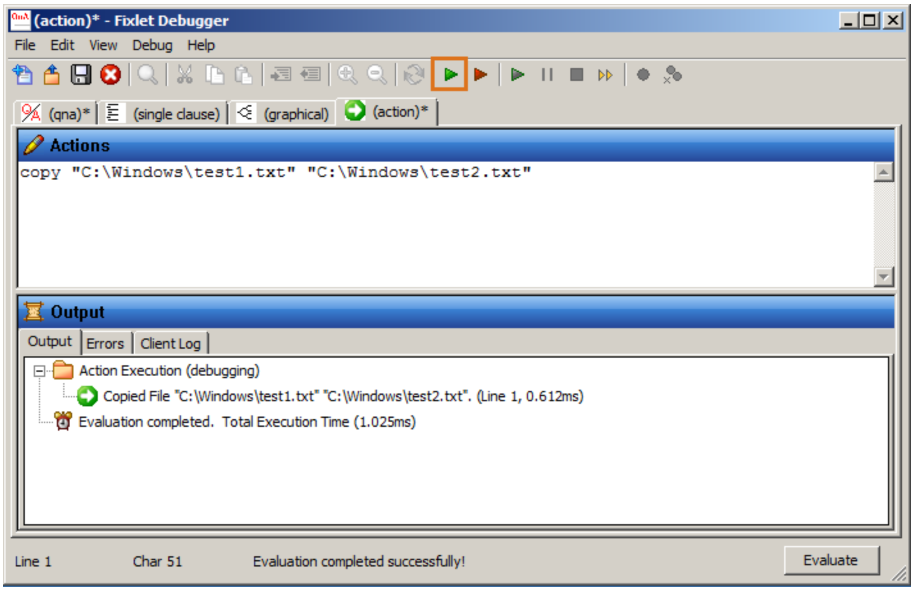
When you enter the code in the Fixlet Debugger on the query and answer **(qna)** tab, you must preface it with a **Q:** to designate it as a query. You designate the answer in the return code with an **A:**, designate the time it took to process the query with a **T:**, and if an error occurs, designate it with an **E:**.

When you enter code into the **Relevance** text box on the **(single clause)** tab, you enter the code without the **Q:** designation. The results of the query show in the **Output** text box.

When you develop action scripts using the Fixlet Debugger, you enter the code on the **(action)** tab. You enter the code in the identical format that the commands must execute. Do not precede the commands with any characters. The following rules assist you with working in this tab:

- You can create comments in the action script by preceding them with a double forward slash (**//**). The comment must start in the first column and all of the characters that follow the slashes are ignored. Comments can be used to document the action script code and they can also be used to temporarily comment out code that you want to prevent from running as you develop the script.
- When you are ready to test the action script, you click the green arrow at the top of the Fixlet Debugger.

- You can use the **Output**, **Errors**, and **Client Log** tabs in the Output frame at the bottom of the Fixlet Debugger to review the results of your action script test.



```
copy "C:\Windows\test1.txt" "C:\Windows\test2.txt"
```

```
Action Execution (debugging)
    Copied File "C:\Windows\test1.txt" "C:\Windows\test2.txt". (Line 1, 0.612ms)
Evaluation completed.  Total Execution Time (1.025ms)
```

Line 1 Char 51 Evaluation completed successfully!

Exercise 40 Starting the environment

Perform the following steps to start the virtual machines in the lab environment if they are not already started.

1. Verify that your student virtual machine is running.

2. Switch to the **student** virtual machine. If you are logged off, log in to the machine with your assigned username and password.

3. Click **Start > BigFix > BigFix Fixlet Debugger**. The BigFix Fixlet Debugger opens.

Exercise 41 Using Relevance to create an automatic group

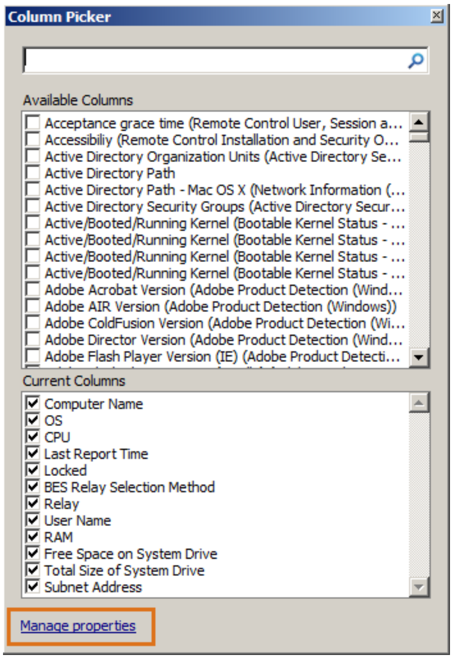
In this exercise, you develop the Relevance that is required to create two automatic computer groups. The members of the first automatic group are HCL BigFix infrastructure systems and must contain any computer where the HCL BigFix Server or Relay component is installed. You then modify the Relevance and create another automatic group that contains non-infrastructure systems. Members of this group are only HCL BigFix clients.

To assist in the development of the Relevance that is required to create the automatic groups, you use the BigFix Console to locate and copy an existing Relevance statement for the **BES Relay Installed Status** property.

1. Switch to the **student** virtual machine.
2. Double-click the **BigFix Console** icon on the Windows desktop. The BigFix Console login window opens.
3. Use your assigned credentials to log into the console.
4. Click **Login**.
   The BigFix Console opens.
5. Select the **All Content** domain in the lower-left portion of the console.
6. Click the **Computers** node in the All Content navigation tree in the left portion of the console.
7. Right-click the retrieved properties column headings to display the Column Picker window.

Hint: The retrieved properties headings are located in the Computers pane. You can place the cursor over any of these headings before performing the right-click to open the Column Picker window.

8. Click the **Manage Properties** link in the lower-left portion of the Column Pickerwindow.



Column Picker

Available Columns

- □ Acceptance grace time (Remote Control User, Session a…
- □ Accessibiliy (Remote Control Installation and Security O…
- □ Active Directory Organization Units (Active Directory Se…
- □ Active Directory Path
- □ Active Directory Path - Mac OS X (Network Information (…
- □ Active Directory Security Groups (Active Directory Secur…
- □ Active/Booted/Running Kernel (Bootable Kernel Status - …
- □ Active/Booted/Running Kernel (Bootable Kernel Status - …
- □ Active/Booted/Running Kernel (Bootable Kernel Status - …
- □ Active/Booted/Running Kernel (Bootable Kernel Status - …
- □ Adobe Acrobat Version (Adobe Product Detection (Wind…
- □ Adobe AIR Version (Adobe Product Detection (Windows))
- □ Adobe ColdFusion Version (Adobe Product Detection (Wi…
- □ Adobe Director Version (Adobe Product Detection (Wind…
- □ Adobe Flash Player Version (IE) (Adobe Product Detecti…

Current Columns

- ☑ Computer Name
- ☑ OS
- ☑ CPU
- ☑ Last Report Time
- ☑ Locked
- ☑ BES Relay Selection Method
- ☑ Relay
- ☑ User Name
- ☑ RAM
- ☑ Free Space on System Drive
- ☑ Total Size of System Drive
- ☑ Subnet Address

Manage properties

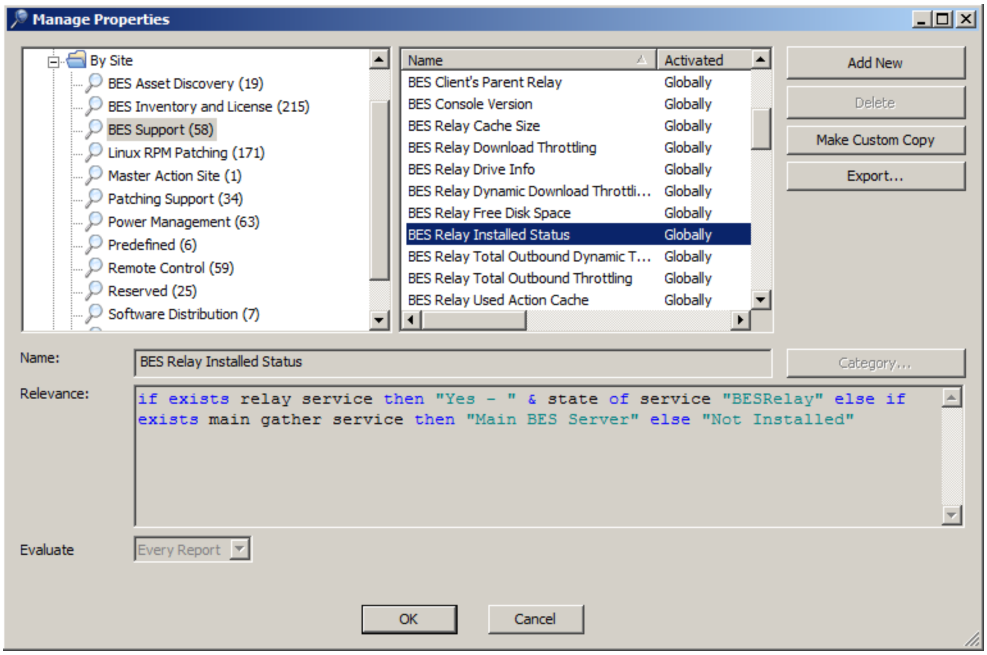
The Manage Properties window opens.

9. In the upper-left frame of the Manage Properties window, expand the **By Site** node.

10. Click the **BES Support** site.

    The list of properties in the Manage Properties window updates to show the properties that are associated with the BES Support site.

11. In the properties list that is located in the frame to the right of the site list, scroll down and click the **BES Relay Installed Status** property.



Name	Activated
BES Client's Parent Relay	Globally
BES Console Version	Globally
BES Relay Cache Size	Globally
BES Relay Download Throttling	Globally
BES Relay Drive Info	Globally
BES Relay Dynamic Download Throttli...	Globally
BES Relay Free Disk Space	Globally
BES Relay Installed Status	Globally
BES Relay Total Outbound Dynamic T...	Globally
BES Relay Total Outbound Throttling	Globally
BES Relay Used Action Cache	Globally

Name: BES Relay Installed Status

Relevance:

```
if exists relay service then "Yes - " & state of service "BESRelay" else if
exists main gather service then "Main BES Server" else "Not Installed"
```

Evaluate: Every Report

The Relevance for the selected property shows in the Relevance box at the bottom of the Manage Properties window.

12. Click inside the Relevance box and press Ctrl+A to select the entire Relevance statement. Press Ctrl+C to copy the Relevance statement to the Windows clipboard.

13. Click **Cancel** to close the Manage Properties window.

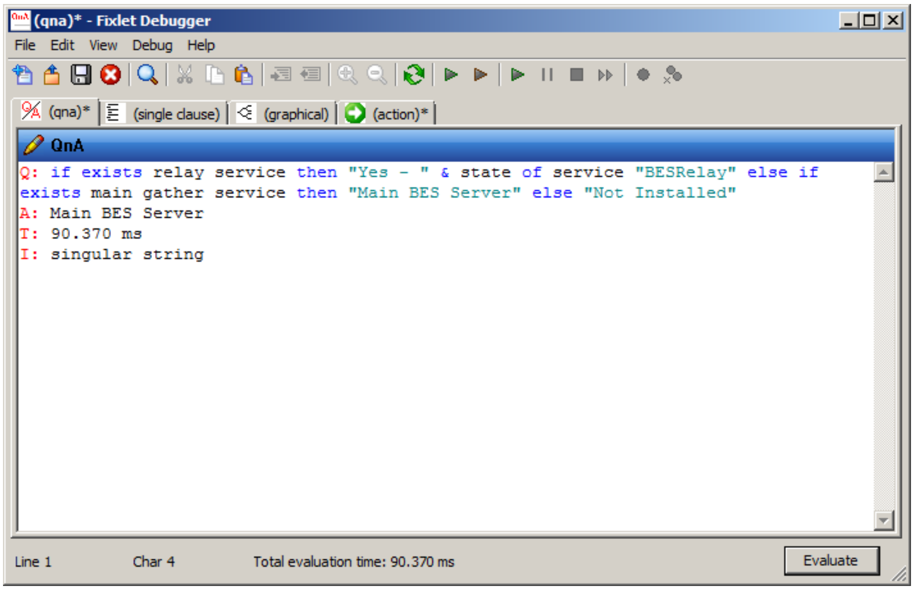
14. Return to the Fixlet Debugger and select the **(qna)** tab if it is not already selected.

15. Paste the Relevance statement that was copied in Step 12 into the **(qna)** tab of the Fixlet Debugger. Ensure that you add the query character **Q:** to the start of the statement.

```
Q: if exists relay service then “Yes – “ & state of service “BESRelay” else if
exists main gather service then “Main BES Server” else “Not Installed”
```

16. Click **Evaluate**.

The query returns the **Main BES Server** answer.



```
Q: if exists relay service then "Yes - " & state of service "BESRelay" else if
exists main gather service then "Main BES Server" else "Not Installed"
A: Main BES Server
T: 90.370 ms
I: singular string
```

Line 1 Char 4 Total evaluation time: 90.370 ms

You must now modify the Relevance expression to return a Boolean true if either the **relay service** or **main gather service** is installed on the system.

17. Copy the **if** portion of the current expression to a new line in the **(qna)** tab.

Hint: It is recommended that you define Boolean expressions by surrounding them with parentheses.

```
Q: (exists relay service)
```

This statement checks to see whether the BigFix relay service is installed on the endpoint. You are not checking the status of the service with this statement, only whether the service is installed.

18. At the end of the statement that you created in Step 17, add **OR ()**. This forces the BigFix client to evaluate the second statement if the first statement returns **False**.

```
Q: (exists relay service) OR ()
```

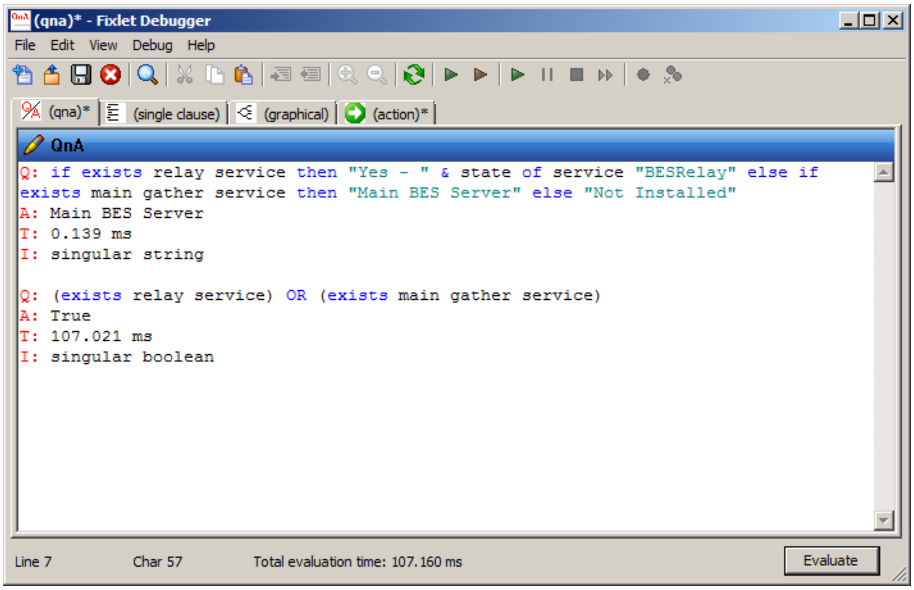
19. You now modify the query to check whether the main gather service is installed on the system. Inside the blank parentheses after the **OR** construct, type `exists main gather service`.

```
Q: (exists relay service) OR (exists main gather service)
```

This **main gather service** is installed on the BigFix server. The Relevance statement is now checking to see whether the endpoint is either a relay or a server.

20. Click **Evaluate**.

The query returns **True**.



(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* | (single clause) | (graphical) | (action)*
QnA
Evaluate

```
Q: if exists relay service then "Yes - " & state of service "BESRelay" else if
exists main gather service then "Main BES Server" else "Not Installed"
A: Main BES Server
T: 0.139 ms
I: singular string

Q: (exists relay service) OR (exists main gather service)
A: True
T: 107.021 ms
I: singular boolean
```

Line 7   Char 57   Total evaluation time: 107.160 ms

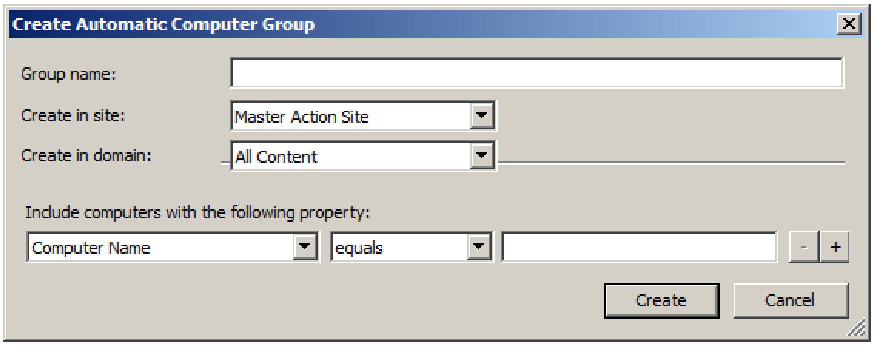
21. Copy the final Relevance statement from the **(qna)** tab with the exception of the query (`Q:`) construct.

You now use this Relevance query to define the membership criteria for the BigFix Infrastructure Systems automatic group.

22. Return to the **BigFix Console**.

23. From the menu, select **Tools > Create New Automatic Computer Group**.

    The Create New Automatic Computer Group window is displayed.



Group name:

Create in site: Master Action Site

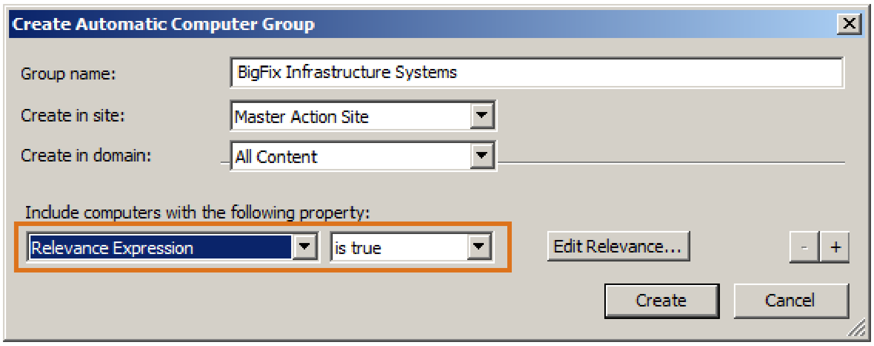
Create in domain: All Content

Include computers with the following property:

Computer Name | equals |

24. In the **Group name** field, enter `BigFix Infrastructure Systems`.

25. Accept the default values for the **Create in site** and **Create in Domain** fields. Select **Relevance Expression** from the drop-down box in the lower-left portion of the Create Automatic Computer Group window.



Group name: BigFix Infrastructure Systems

Create in site: Master Action Site

Create in domain: All Content

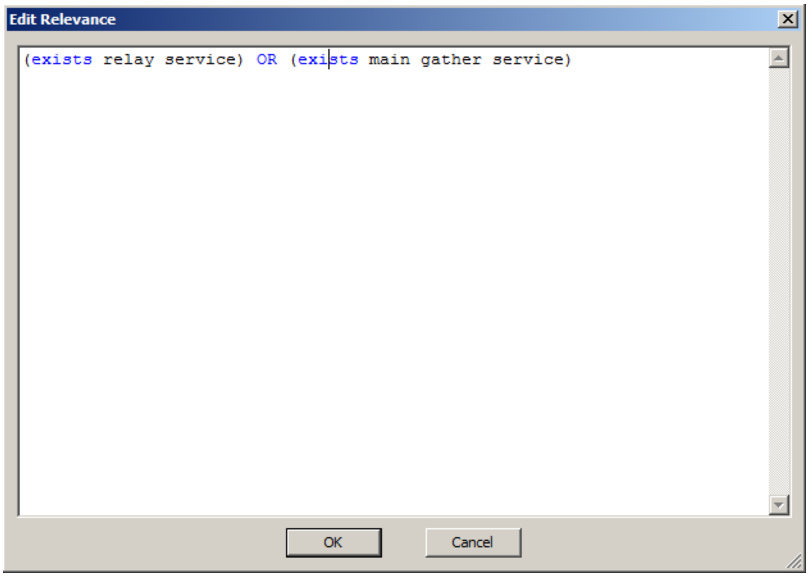
Include computers with the following property:

Relevance Expression | is true

26. Click **Edit Relevance**.

    The Edit Relevance window is displayed.

27. In the **Edit Relevance** window, remove any existing Relevance statements and then paste the new Relevance expression that you created in Step 19 into this window.



```
(exists relay service) OR (exists main gather service)
```

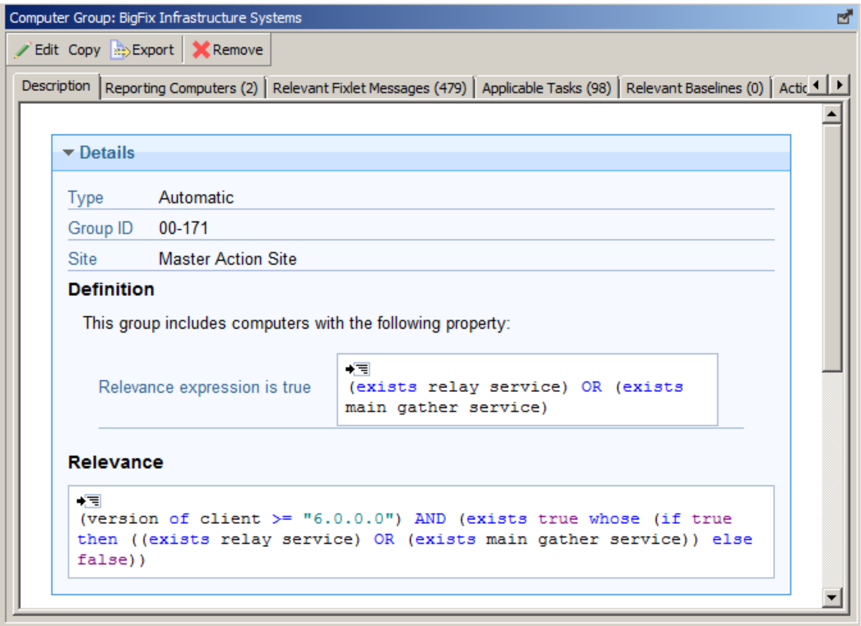
28. Click **OK**.

    The Relevance is saved and the Edit Relevance window closes.

29. Click **Create** to finish creating the automatic group.

    The Computer Group: BigFix Infrastructure Systems pane opens and you see the details for the new group. The Definition section shows the Relevance expression that you created to define the membership criteria for the automatic group.

Note: Because automatic groups are only supported in version 6.0 or later of the BigFix client, additional Relevance is automatically added to the Relevance that you created that tests for the version of the client.



Type Automatic

Group ID 00-171

Site Master Action Site

Definition

This group includes computers with the following property:

Relevance expression is true

```
(exists relay service) OR (exists
main gather service)
```

Relevance

```
(version of client >= "6.0.0.0") AND (exists true whose (if true
then ((exists relay service) OR (exists main gather service)) else
false))
```

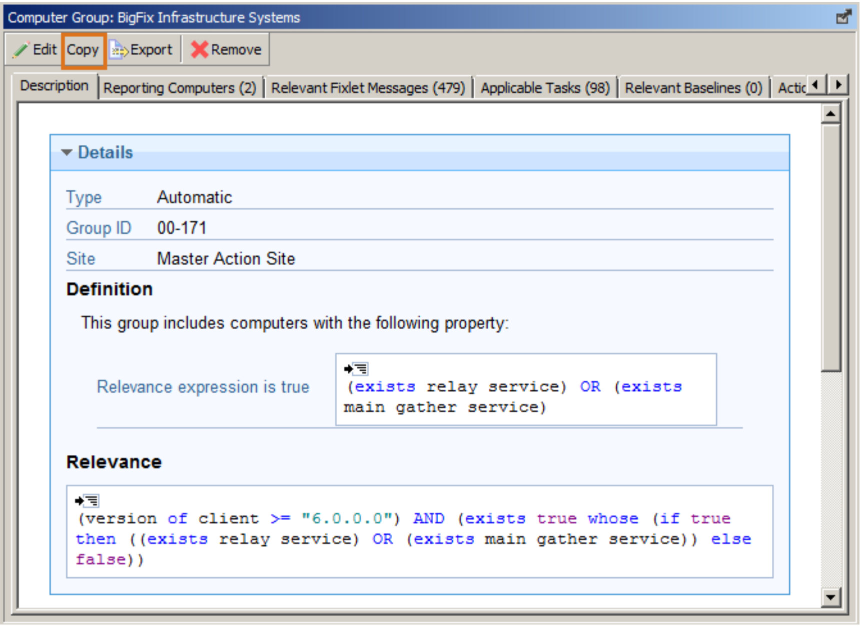
30. Verify that the appropriate computers are added to the **BigFix Infrastructure Systems** automatic group by clicking the **Reporting Computers** tab.

One computer is listed as a member of the automatic group:

– BESFNDWINROOT

You now create a new automatic group named BigFix Non-Infrastructure Systems by copying the group that you created in Step 23 to Step 29 and modifying the Relevance as required.

31. Create a copy of the **BigFix Infrastructure Systems** automatic group by clicking **Copy** in the Computer Group pane.



Type Automatic

Group ID 00-171

Site Master Action Site

Definition

This group includes computers with the following property:

Relevance expression is true

```
(exists relay service) OR (exists
main gather service)
```

Relevance

```
(version of client >= "6.0.0.0") AND (exists true whose (if true
then ((exists relay service) OR (exists main gather service)) else
false))
```

The Create Automatic Group window is displayed.

32. In the **Group name** field, enter `BigFix Non-Infrastructure Systems`.
Change the Boolean condition of the Relevance statement from **is true** to **is false**. Click **Create.**

Hint: There are several other ways that you can specify the membership criteria for the BigFix Non-Infrastructure Systems automatic group. These include the following ways:

- You could leave the Boolean condition set to **is true** and then modify the Relevance query to check that the relay service and the main gather service do not exist on the system.

  ```
  Q: (NOT exists relay service) AND (NOT exists main gather service)
  ```

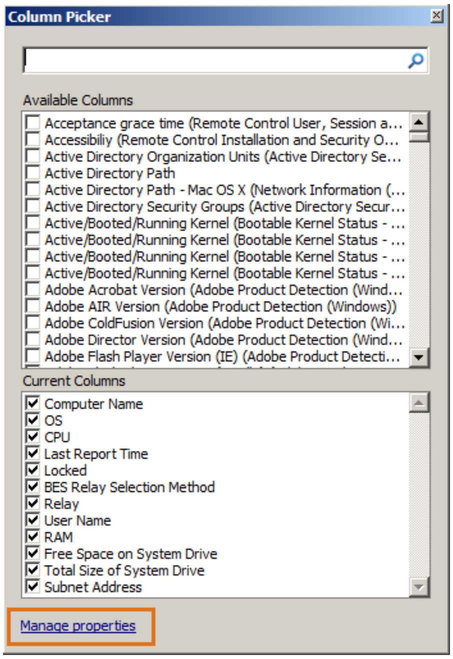
- You could also evaluate Group Membership instead of Relevance. In this case, you would specify that if an endpoint is not a member of the BigFix Infrastructure Systems automatic group, it must be a member of the BigFix Non-Infrastructure Systems automatic group.

Exercise 42 Editing Relevance to create a property

In this exercise, you modify the existing Relevance for multiple retrieved properties that display information about the DNS servers to create a single property. This single property displays the DNS servers for Windows, Linux, and Mac systems.

To perform the steps in this exercise, you use the BigFix console on the **BESFNDWIN10** virtual machine along with the Fixlet Debugger to test the Relevance.

1. If it is not already open, launch the BigFix Console as instructed in Step 2 to Step 4 of Exercise 2, “Using Relevance to create an automatic group,” on page 4-3
2. Select the **All Content** domain in the lower-left portion of the console.
3. Click the **Computers** node in the All Content navigation tree in the left portion of the console.
4. Right-click the retrieved properties column headings to display the Column Picker window.
5. Click **Manage Properties** at the bottom of the Column Picker window to display the Manage Properties Window.



Column Picker

Available Columns

□ Acceptance grace time (Remote Control User, Session a…
□ Accessibiliy (Remote Control Installation and Security O…
□ Active Directory Organization Units (Active Directory Se…
□ Active Directory Path
□ Active Directory Path - Mac OS X (Network Information (…
□ Active Directory Security Groups (Active Directory Secur…
□ Active/Booted/Running Kernel (Bootable Kernel Status - …
□ Active/Booted/Running Kernel (Bootable Kernel Status - …
□ Active/Booted/Running Kernel (Bootable Kernel Status - …
□ Active/Booted/Running Kernel (Bootable Kernel Status - …
□ Adobe Acrobat Version (Adobe Product Detection (Wind…
□ Adobe AIR Version (Adobe Product Detection (Windows))
□ Adobe ColdFusion Version (Adobe Product Detection (Wi…
□ Adobe Director Version (Adobe Product Detection (Wind…
□ Adobe Flash Player Version (IE) (Adobe Product Detecti…

Current Columns

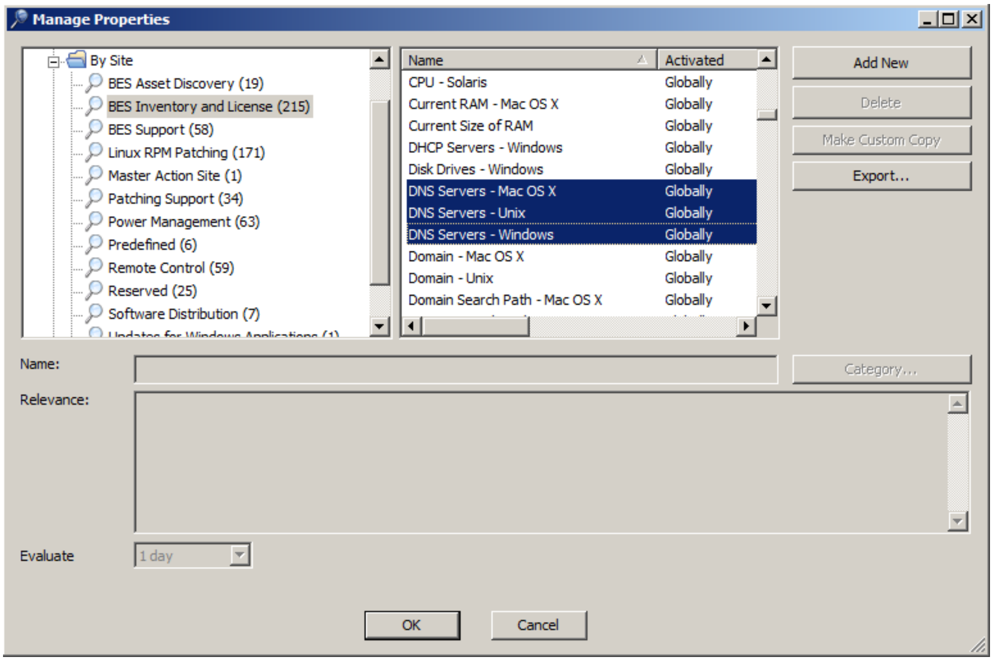
☑ Computer Name
☑ OS
☑ CPU
☑ Last Report Time
☑ Locked
☑ BES Relay Selection Method
☑ Relay
☑ User Name
☑ RAM
☑ Free Space on System Drive
☑ Total Size of System Drive
☑ Subnet Address

Manage properties

6. In the upper-left frame of the Manage Properties window, expand the **By Site** node.
7. Click the **BES Inventory and License** site.

   The list of properties in the Manage Properties window updates to show the properties that are associated the BES Inventory and License site.

8. In the right side of the Manage Properties window, scroll down until you locate the following DNS properties:
   - DNS Servers - Mac OS X
   - DNS Servers - Unix
   - DNS Servers - Windows



- By Site
  - BES Asset Discovery (19)
  - BES Inventory and License (215)
  - BES Support (58)
  - Linux RPM Patching (171)
  - Master Action Site (1)
  - Patching Support (34)
  - Power Management (63)
  - Predefined (6)
  - Remote Control (59)
  - Reserved (25)
  - Software Distribution (7)
  - Updates for Windows Applications (1)

Name	Activated
CPU - Solaris	Globally
Current RAM - Mac OS X	Globally
Current Size of RAM	Globally
DHCP Servers - Windows	Globally
Disk Drives - Windows	Globally
DNS Servers - Mac OS X	Globally
DNS Servers - Unix	Globally
DNS Servers - Windows	Globally
Domain - Mac OS X	Globally
Domain - Unix	Globally
Domain Search Path - Mac OS X	Globally

9. In the Manage Properties window, select the **DNS Servers - Windows** property.

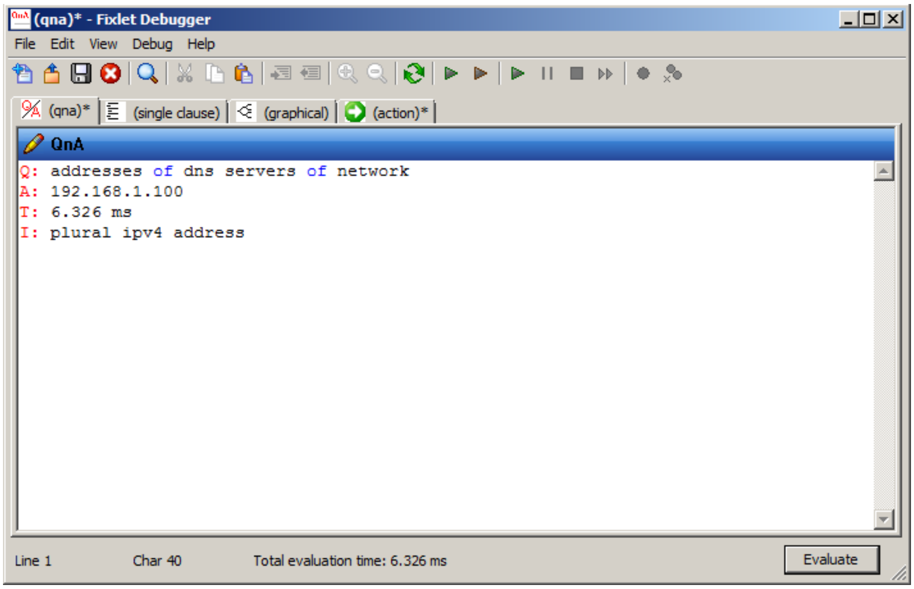
   The Relevance statement that is associated with the selected property displays in the Relevance pane at the bottom of the Manage Properties window.

10. In the Relevance pane, click anywhere in the Relevance statement and press Ctrl+A to select the entire statement. Press Ctrl+C to copy the statement to the Windows clipboard.

11. Switch to the Fixlet Debugger and select the (qna) tab. Clear any existing statement and paste the Relevance statement from the Windows clipboard into the blank **(qna)** tab. Add the query keyword **Q**: to the beginning of the statement to make it a valid query.

12. Click **Evaluate**.

The query is evaluated and returns the IP address of the configured DNS servers.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)*
QnA
Line 1
Char 40
Total evaluation time: 6.326 ms
Evaluate

```
Q: addresses of dns servers of network
A: 192.168.1.100
T: 6.326 ms
I: plural ipv4 address
```

13. You now modify the query to use an **`if-then-else`** expression to test for the Windows operating system. Perform the following steps to update the query.

    `Q: if () then () else ()`

    a. Create the basic if-then-else template on a new line below the existing query in the **(qna)** tab of the Fixlet Debugger.

    `Q: if () then () else ()`

    b. Add a statement for the **if** expression to test for the Windows operating system.

    `Q: if (`**`name of operating system starts with "Win"`**`) then () else ()`

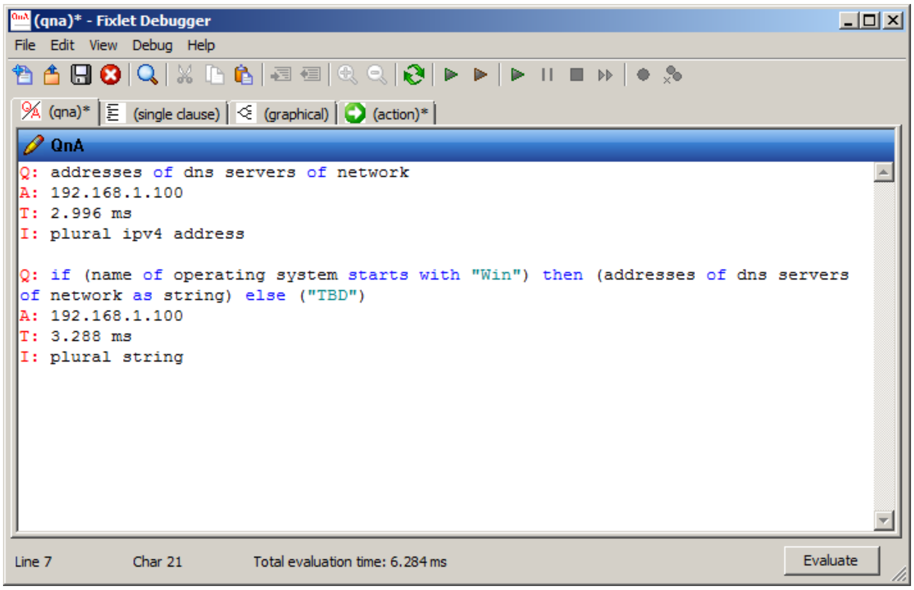
    c. Paste the expression that was copied in Step 10 inside the parentheses for the **then** portion of the expression. You must modify the query to convert the result to a string to ensure that the data types are compatible.

    `Q: if (name of operating system starts with "Win") then (`**`addresses of dns servers of network  as string`**`) else ()`

    d. Enter a string inside the parentheses for the **else** expression. This string is returned if the query is evaluated on a non-Windows system. For this exercise, you enter **"TBD"**.

    `Q: if (name of operating system starts with "Win") then (addresses of dns servers of network  as string) else (`**`"TBD"`**`)`

e. Click **Evaluate**.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)*
QnA

```
Q: addresses of dns servers of network
A: 192.168.1.100
T: 2.996 ms
I: plural ipv4 address

Q: if (name of operating system starts with "Win") then (addresses of dns servers
of network as string) else ("TBD")
A: 192.168.1.100
T: 3.288 ms
I: plural string
```

Line 7
Char 21
Total evaluation time: 6.284 ms
Evaluate

You now modify the query to return the DNS Server information for the UNIX and Mac systems. You must replace the string in the `else` clause with the Relevance statement that you copy from the Manage Properties window in the console.

Hint: The Relevance statement for the two **DNS Servers - Mac OS X** and **DNS Servers - UNIX** properties are identical. You can copy the Relevance expression from either of these properties directly into the `else` clause of your current query.

14. Return to the Manage Properties window of the BigFix Console.

15. Select either the **DNS Servers - Mac OS X** or the **DNS Servers - UNIX** property.

    The Relevance statement for the selected property is displayed in the Relevance pane of the Manage Properties window.

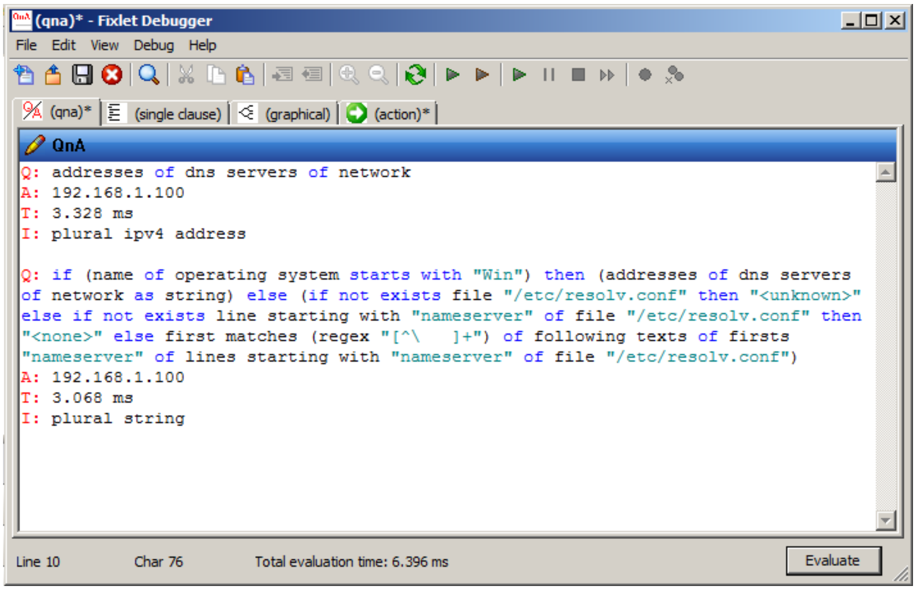
16. In the Relevance pane, click anywhere in the Relevance statement and press Ctrl+A to select the entire statement. Press Ctrl+C to copy the statement to the Windows clipboard.

17. Return to the **(qna)** tab in the Fixlet Debugger and replace the **"TBD"** string in the **else** portion of the statement with the query that you copied from the Manage Properties window.

Hint: This property already returns a string data type, so you do not have to convert it to a string.

18. Click **Evaluate**.

Because the query is evaluated on a Windows system, you see the same DNS server that was displayed in Step 13.



```
Q: addresses of dns servers of network
A: 192.168.1.100
T: 3.328 ms
I: plural ipv4 address

Q: if (name of operating system starts with "Win") then (addresses of dns servers
of network as string) else (if not exists file "/etc/resolv.conf" then "<unknown>"
else if not exists line starting with "nameserver" of file "/etc/resolv.conf" then
"<none>" else first matches (regex "[^\   ]+") of following texts of firsts
"nameserver" of lines starting with "nameserver" of file "/etc/resolv.conf")
A: 192.168.1.100
T: 3.068 ms
I: plural string
```

You now create a new property in the BigFix Console that uses the Relevance query that you just created.

19. In the **(qna)** tab of the Fixlet Debugger, copy the entire expression that you evaluated in Step 18 without the query (**Q:**) keyword.

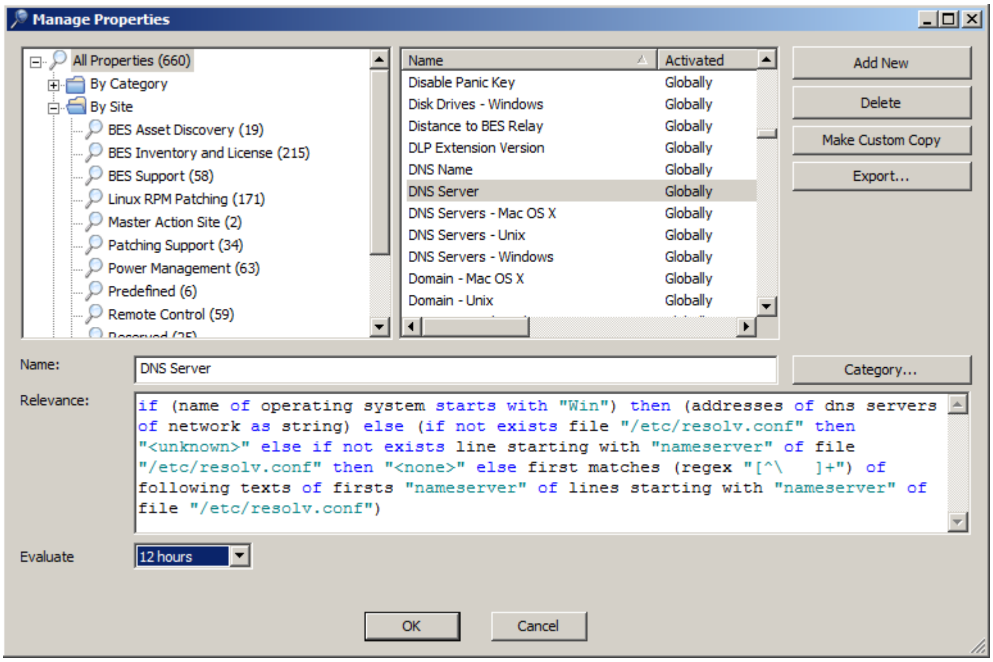
20. .Return to the Manage Properties window in the BigFix Console.

21. Click **Add New**.

22. In the **Name** field, enter `DNS Servers`.

23. In the **Relevance** field, replace the default text by pasting the expression from that you copied from the **(qna)** tab in Step 19.

24. From the **Evaluate** drop-down menu, select **Every 12 hours**.



- All Properties (660)
  - By Category
  - By Site
    - BES Asset Discovery (19)
    - BES Inventory and License (215)
    - BES Support (58)
    - Linux RPM Patching (171)
    - Master Action Site (2)
    - Patching Support (34)
    - Power Management (63)
    - Predefined (6)
    - Remote Control (59)
    - Reserved (25)

Name	Activated
Disable Panic Key	Globally
Disk Drives - Windows	Globally
Distance to BES Relay	Globally
DLP Extension Version	Globally
DNS Name	Globally
DNS Server	Globally
DNS Servers - Mac OS X	Globally
DNS Servers - Unix	Globally
DNS Servers - Windows	Globally
Domain - Mac OS X	Globally
Domain - Unix	Globally

Name: DNS Server

Relevance:

```
if (name of operating system starts with "Win") then (addresses of dns servers
of network as string) else (if not exists file "/etc/resolv.conf" then
"<unknown>" else if not exists line starting with "nameserver" of file
"/etc/resolv.conf" then "<none>" else first matches (regex "[^\   ]+") of
following texts of firsts "nameserver" of lines starting with "nameserver" of
file "/etc/resolv.conf")
```

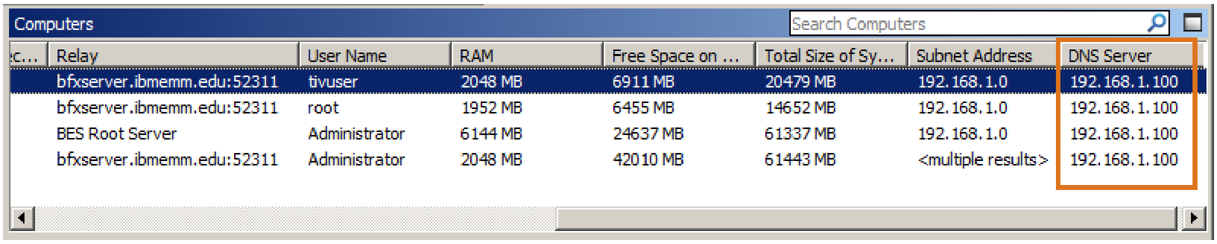
Evaluate: 12 hours

25. Click **OK**.

    The new property is saved and the Manage Properties window closes.

26. In the Computers pane of the BigFix Console, scroll to the right to locate the new **DNS Server** property.

Hint: Initially, the value of the property is **<not reported>**. It might take several minutes for the new property to be evaluated by the clients and return the results. You can also resize and move the columns as desired.



c...	Relay	User Name	RAM	Free Space on ...	Total Size of Sy...	Subnet Address	DNS Server
	bfxserver.ibmemm.edu:52311	tivuser	2048 MB	6911 MB	20479 MB	192.168.1.0	192.168.1.100
	bfxserver.ibmemm.edu:52311	root	1952 MB	6455 MB	14652 MB	192.168.1.0	192.168.1.100
	BES Root Server	Administrator	6144 MB	24637 MB	61337 MB	192.168.1.0	192.168.1.100
	bfxserver.ibmemm.edu:52311	Administrator	2048 MB	42010 MB	61443 MB	<multiple results>	192.168.1.100

Exercise 43 Renaming a file

In this exercise, you develop an action script to rename a file you create in c:\windows, let's call it `test3.txt`to a new file named `test4.txt`. Next, you create a backup copy of the `test4.txt` file in the same directory named `test4bak.txt`.

Use the following criteria when you develop action script:

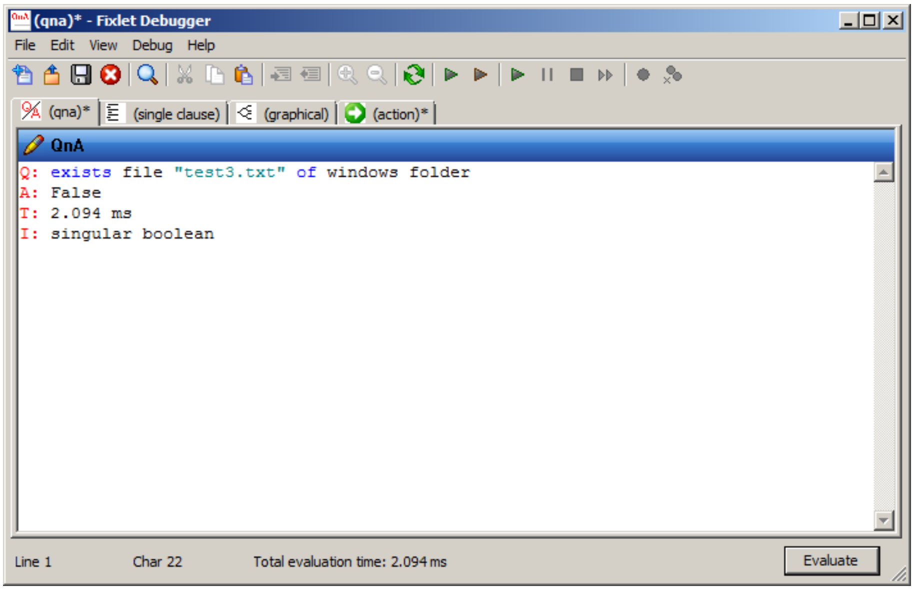
- The action must only evaluate as relevant if the `test3.txt` file exists in the specified directory.
- The Relevance query must not use any hardcoded paths for the Relevance check.
- You must use the **`delete`** command in the action script before you move or copy the files.
- You can use hardcoded paths for the file location in the action script.

Perform these steps to complete the exercise:

1. Open the Fixlet Debugger and select the **(qna)** tab.
2. Perform the following steps to create the Relevance query to check for the existence of the `test3.txt` file in the Windows folder.
   a. Enter the following expression:

   ```
   Q: exists file “test3.txt” of windows folder
   ```

   b. Click **Evaluate** and note that the result is **False**.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)*
QnA

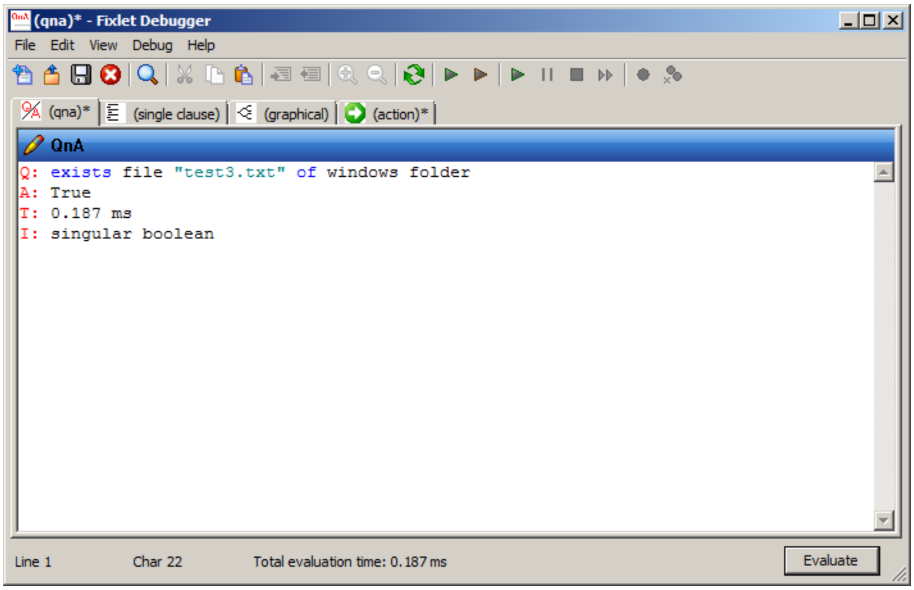
```
Q: exists file "test3.txt" of windows folder
A: False
T: 2.094 ms
I: singular boolean
```

Line 1 Char 22 Total evaluation time: 2.094 ms
Evaluate

3. Create the `test3.txt` file in the `C:\Windows` folder.
   a. Open Windows Explorer and browse to the `C:\Windows` folder.
   b. Right-click in the right pane of Windows Explorer. From the menu, select **New > Text Document**.
   c. Name the file `test3.txt`.
   d. Double-click the `test3.txt` file to edit it and place some text in the file.

e. Close the file and click **Yes** when you are prompted to save the changes.

f. Return to the Debugger tool and click **Evaluate**. Now the result is **True**.



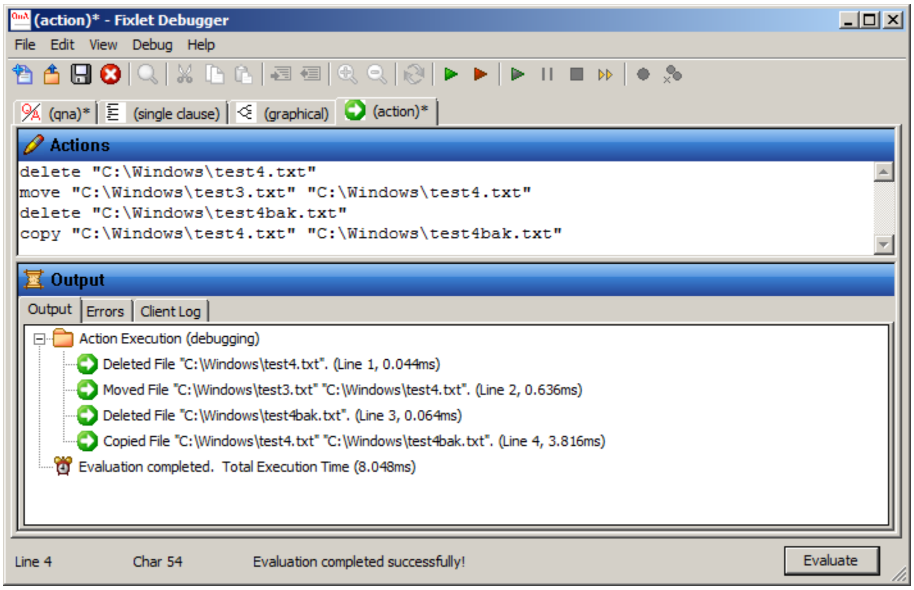
```
Q: exists file "test3.txt" of windows folder
A: True
T: 0.187 ms
I: singular boolean
```

Line 1 Char 22 Total evaluation time: 0.187 ms

4. Switch to the **(action)** tab of the Fixlet Debugger. Perform the following steps create an action script that creates two new text files by copying the original file. You must ensure that existing files that have the destination file names do not exist by attempting to deleting them first.

   a. Enter the following information into the **Actions** field:

```
delete “C:\Windows\test4.txt”
move “C:\Windows\test3.txt” “C:\Windows\test4.txt”
delete “C:\Windows\test4bak.txt”
copy “C:\Windows\test4.txt” “C:\Windows\test4bak.txt”
```



```
delete "C:\Windows\test4.txt"
move "C:\Windows\test3.txt" "C:\Windows\test4.txt"
delete "C:\Windows\test4bak.txt"
copy "C:\Windows\test4.txt" "C:\Windows\test4bak.txt"
```

```
Action Execution (debugging)
    Deleted File "C:\Windows\test4.txt". (Line 1, 0.044ms)
    Moved File "C:\Windows\test3.txt" "C:\Windows\test4.txt". (Line 2, 0.636ms)
    Deleted File "C:\Windows\test4bak.txt". (Line 3, 0.064ms)
    Copied File "C:\Windows\test4.txt" "C:\Windows\test4bak.txt". (Line 4, 3.816ms)
Evaluation completed.  Total Execution Time (8.048ms)
```

Line 4 Char 54 Evaluation completed successfully!

c. Verify the results. In the Explorer instance, select **View > Refresh** and scroll down to the `test4.txt` file.

d. Open the `test4.txt` file and verify that it is identical to the original file.

Exercise 44 Using Relevance substitution

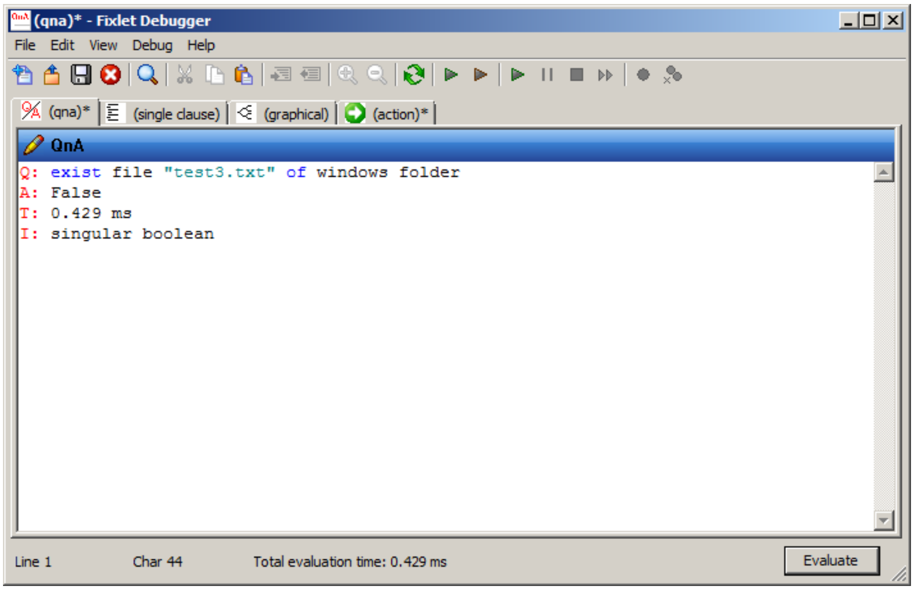
In this exercise, you modify the action script that you created in Exercise 4, "Renaming a file," on page 4-17 to replace the hardcoded paths by using Relevance substitution.

Perform the following steps to complete this exercise:

1. Return to the Fixlet Debugger and select the **(qna)** tab.
2. Verify that the Relevance expression to check existence of the `test3.txt` file that you created in Exercise 4 is still available on the **(qna)** tab. If it is not, you can enter the following text again:

   ```
   Q: exists file “test3.txt” of windows folder
   ```

3. Click **Evaluate**.


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)*
QnA
Line 1 Char 44 Total evaluation time: 0.429 ms
Evaluate

```
Q: exist file "test3.txt" of windows folder
A: False
T: 0.429 ms
I: singular boolean
```

4. Copy the `C:\Windows\test4.txt` file to `C:\Windows\test3.txt`.

5. Return to the **(qna)** tab in the Fixlet Debugger the single-clause tab. Click **Evaluate**.

   The Relevance query returns a result of **True**.

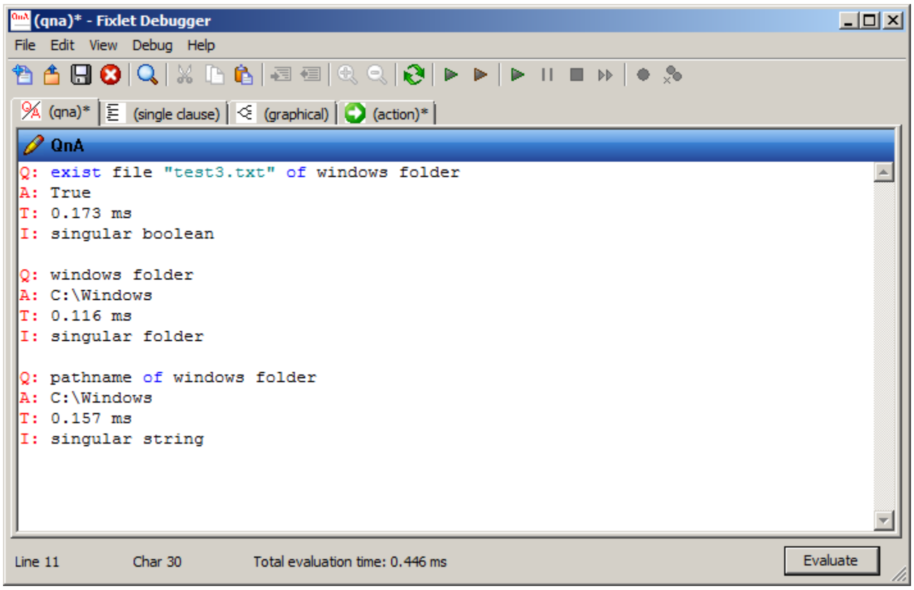
6. Below the existing Relevance query in the **(qna)** tab, perform the following steps to create a new query that is used for the Relevance substitution in the action script.

   a. Return the Windows folder when the query is evaluated.

   ```
   Q: windows folder
   ```

   b. Verify that only the folder is returned.

   ```
   Q: pathname of windows folder
   ```


(qna)* - Fixlet Debugger
File Edit View Debug Help
(qna)* (single clause) (graphical) (action)*
QnA
Line 11 Char 30 Total evaluation time: 0.446 ms
Evaluate

```
Q: exist file "test3.txt" of windows folder
A: True
T: 0.173 ms
I: singular boolean

Q: windows folder
A: C:\Windows
T: 0.116 ms
I: singular folder

Q: pathname of windows folder
A: C:\Windows
T: 0.157 ms
I: singular string
```

c. Copy the Relevance query from Step b to use as Relevance substitution in the **(action)** tab.

7. Perform the following steps to use the Find and Replace capabilities of the Fixlet Debugger to modify the action script.

   a. Select the **(action)** tab.

   b. Select **Edit > Find and Replace**

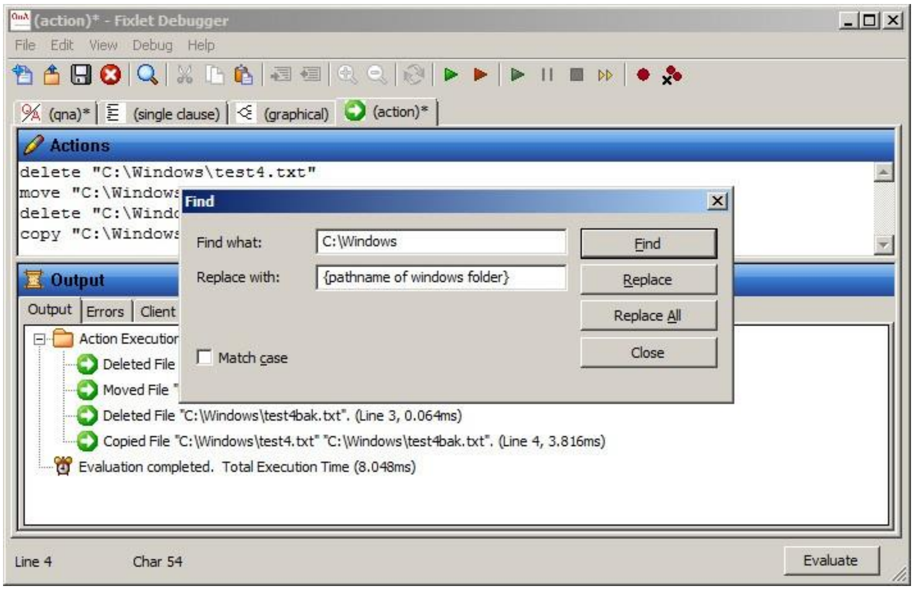
   The Find window opens.

Hint: You can also use the keyboard command Ctrl+F to open the **Find** window.

   c. In the **Find what** field, enter `C:\windows`.

   d. In the **Replace with** field, paste the previously copied Relevance query between curly braces.

   ```
   {pathname of windows folder}
   ```



```
delete "C:\Windows\test4.txt"
move "C:\Windows
delete "C:\Windc
copy "C:\Windows
```

Find what: C:\Windows

Replace with: {pathname of windows folder}

☐ Match case

Action Executior

Deleted File

Moved File "

Deleted File "C:\Windows\test4bak.txt". (Line 3, 0.064ms)

Copied File "C:\Windows\test4.txt" "C:\Windows\test4bak.txt". (Line 4, 3.816ms)

Evaluation completed. Total Execution Time (8.048ms)

Line 4 Char 54

   e. Click **Replace All**.

   A message that indicates that 6 instances were replaced is displayed.

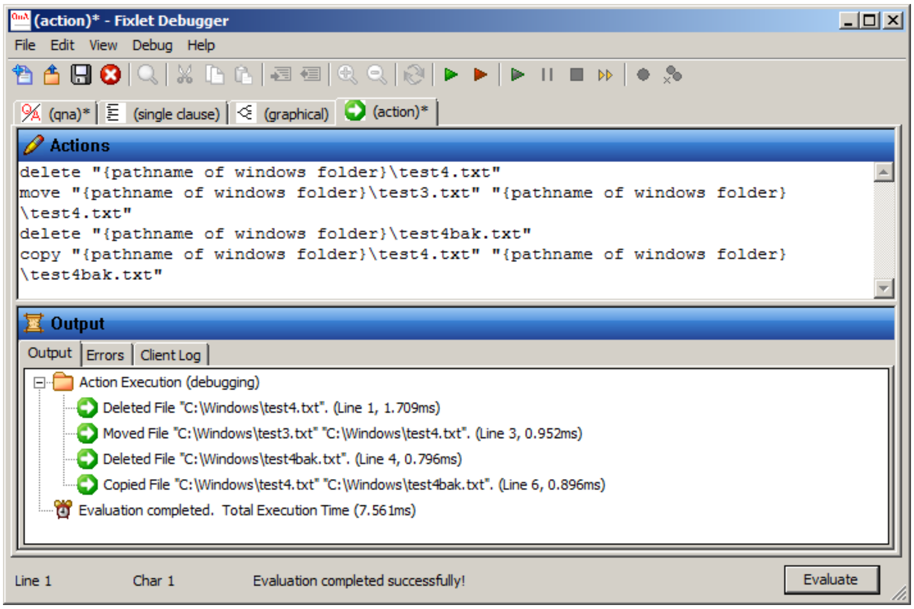
   f. Click **OK** and then click **Close**.

   The Find window closes.

   g. Review the action script to confirm that it has been updated with all of the required changes.

   h. Click **Evaluate**.

The action script is run and a message that the evaluation completed successfully is displayed.



```
delete "{pathname of windows folder}\test4.txt"
move "{pathname of windows folder}\test3.txt" "{pathname of windows folder}
\test4.txt"
delete "{pathname of windows folder}\test4bak.txt"
copy "{pathname of windows folder}\test4.txt" "{pathname of windows folder}
\test4bak.txt"
```

```
Action Execution (debugging)
    Deleted File "C:\Windows\test4.txt". (Line 1, 1.709ms)
    Moved File "C:\Windows\test3.txt" "C:\Windows\test4.txt". (Line 3, 0.952ms)
    Deleted File "C:\Windows\test4bak.txt". (Line 4, 0.796ms)
    Copied File "C:\Windows\test4.txt" "C:\Windows\test4bak.txt". (Line 6, 0.896ms)
Evaluation completed.  Total Execution Time (7.561ms)
```

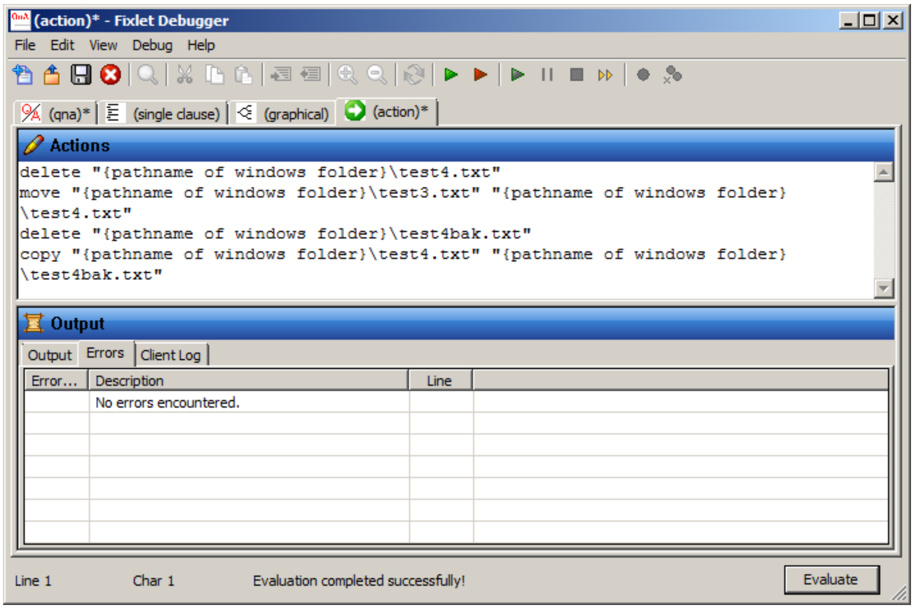
Line 1 Char 1 Evaluation completed successfully!

i. Switch to Windows Explorer and verify the results.

Hint: You might have to select **View > Refresh** from the menu to update the Windows Explorer window with the new results.

8. Review the remaining tabs in the Output pane of the **(action)** tab.

   a. Click the **Errors** tab and verify that the **No errors encountered** message is shown.



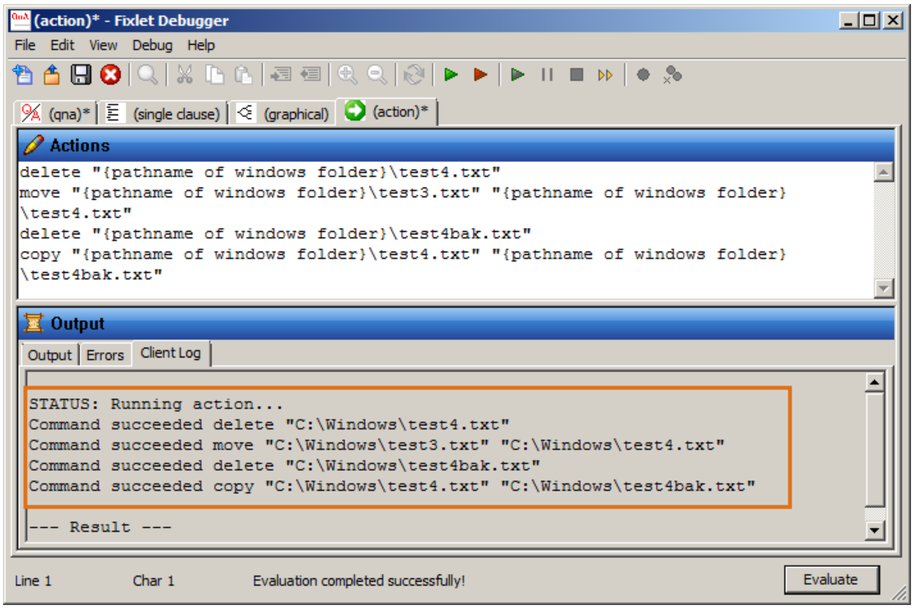
```
delete "{pathname of windows folder}\test4.txt"
move "{pathname of windows folder}\test3.txt" "{pathname of windows folder}
\test4.txt"
delete "{pathname of windows folder}\test4bak.txt"
copy "{pathname of windows folder}\test4.txt" "{pathname of windows folder}
\test4bak.txt"
```

Error...	Description	Line
	No errors encountered.	

Line 1 Char 1 Evaluation completed successfully!

   b. Click the **Client Log** tab and review the output.

Note: The output on the **Client Log** tab replicates what is reported in the BigFix Client Log when the action is run on the client. Note that the Relevance substitution was performed and that the output shows the actual path to the Windows folder.



```
delete "{pathname of windows folder}\test4.txt"
move "{pathname of windows folder}\test3.txt" "{pathname of windows folder}
\test4.txt"
delete "{pathname of windows folder}\test4bak.txt"
copy "{pathname of windows folder}\test4.txt" "{pathname of windows folder}
\test4bak.txt"
```

```
STATUS: Running action...
Command succeeded delete "C:\Windows\test4.txt"
Command succeeded move "C:\Windows\test3.txt" "C:\Windows\test4.txt"
Command succeeded delete "C:\Windows\test4bak.txt"
Command succeeded copy "C:\Windows\test4.txt" "C:\Windows\test4bak.txt"

--- Result ---
```

Exercise 45 Using Override in ActionScript

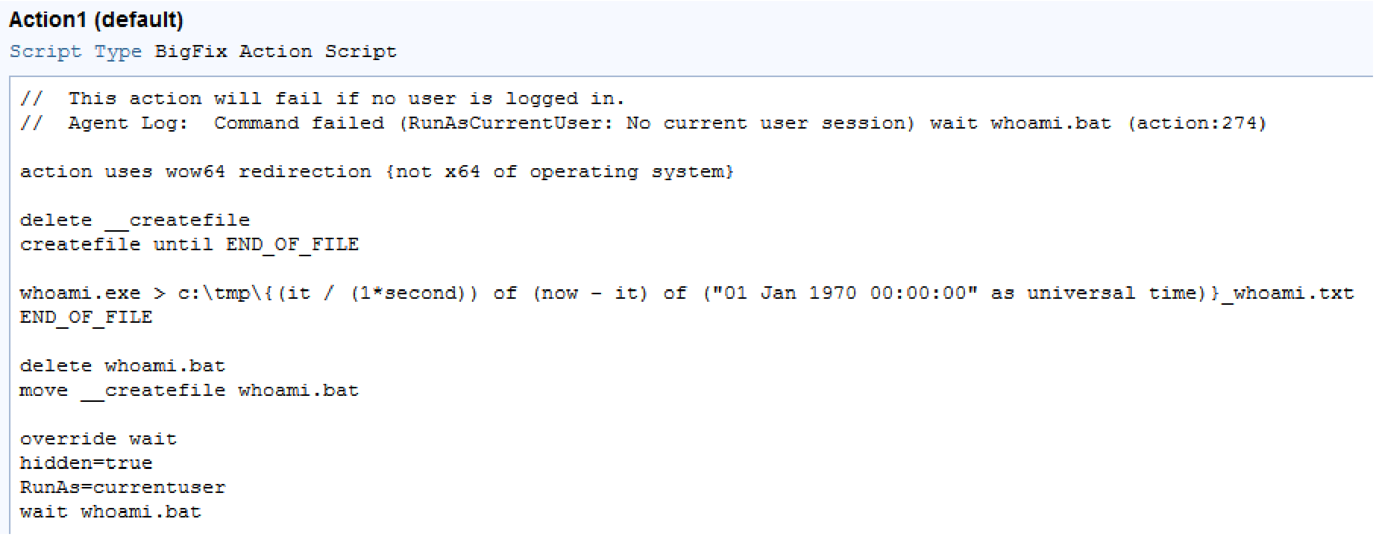
In this exercise, you create a custom task that demonstrates the runas capabilities of the Override command block. This exercise will also demonstrate the creation and use of a multi-action task.

The result will be a task which can run as a currently logged in user, a specific local user, and local system. By using the whoami.exe command, we will discover the identity of the current running process. These actions may be used as a template for many other tasks which require specific user access or permissions.

Part 1 – Running as CurrentUser

1. Open the BigFix console and select Tools>Create New Task from the menu.
2. Change the name of the Task to Override Test.
3. Select the Actions tab.
4. The action script will use a number of commands we have covered including createfile, delete, relevance substitution, move, and override.
5. Enter the following actionscript for Action1:

```
action uses wow64 redirection {not x64 of operating system}

delete __createfile

createfile until END_OF_FILE

whoami.exe > c:\tmp\{(it / (1*second)) of (now - it) of (“01 Jan 1970 00:00:00” as universal time)}_whoami.txt

END_OF_FILE


delete whoami.bat

move __createfile whoami.bat

override wait

hidden=true

RunAs=currentuser

wait whoami.bat
```



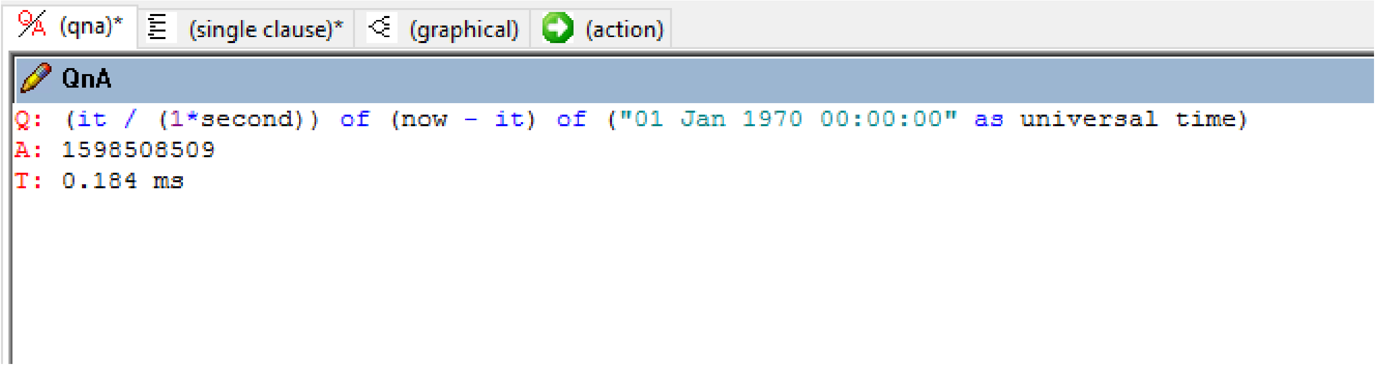
Action1 (default)

Script Type BigFix Action Script

```
//  This action will fail if no user is logged in.
//  Agent Log:  Command failed (RunAsCurrentUser: No current user session) wait whoami.bat (action:274)

action uses wow64 redirection {not x64 of operating system}

delete __createfile
createfile until END_OF_FILE

whoami.exe > c:\tmp\{(it / (1*second)) of (now - it) of ("01 Jan 1970 00:00:00" as universal time)}_whoami.txt
END_OF_FILE

delete whoami.bat
move __createfile whoami.bat

override wait
hidden=true
RunAs=currentuser
wait whoami.bat
```

6. The script creates a Windows batch file which will execute the whoami.exe command and output to a temporary file with a numeric filename equivalent to the number of seconds since January 1st, 1970 (The beginning of Time.)

7. Test the relevance substitution (the text between the curly braces) in the Fixlet Debugger to see what the substitution will look like. This technique allows the task to run multiple times without overwriting the previous results.



```
Q: (it / (1*second)) of (now - it) of ("01 Jan 1970 00:00:00" as universal time)
A: 1598508509
T: 0.184 ms
```

8. In this case the output file of the script would be named **`c:\tmp\1598508509_whoami.txt`**

9. The next section of the script moves the temporary file __createfile to whoami.bat

10. It is important to note that this script will fail if there is currently no user logged in.

11. You can save the Task at this stage and test it using the BigFix Server as a target. This will avoid an error if no user is logged should you run this action against the Win 10 system in your environment.

12. Execute the Task and view the contents of the file created in c:\tmp

13. You should see the name of the administrative user.

Part 2 – Running as a localuser

1. Open the Task again and click the Edit button.
2. Select the Actions tab and click the Add button to add Action2.
3. Enter the following actionscript for Action2.

```
action uses wow64 redirection {not x64 of operating system}
delete __createfile
createfile until END_OF_FILE
whoami.exe > c:\tmp\{(it / (1*second)) of (now - it) of (“01 Jan 1970 00:00:00” as
universal time)}_whoami.txt
END_OF_FILE
delete whoami.bat
move __createfile whoami.bat

override wait
hidden=true
RunAs=localuser
user=tecuser
password=required
wait whoami.bat
```

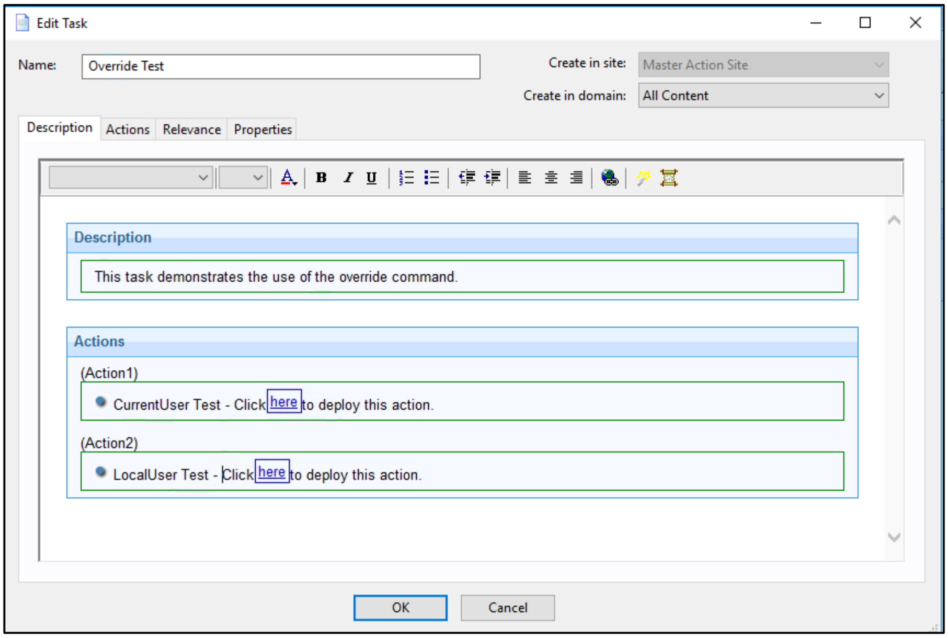
4. Since we now have two actions available we should change the action descriptions on the Description Tab before saving. This will make it easier to select the appropriate action when we click Take Action.

5. Save the actionscript.

6. For this part of the exercise we will execute the script on BESFNDWIN10 where we know the tecuser exists. The user does not need to be logged in for this test to be successful.

Note: Make sure that the C:\tmp directory exists on **BESFNDWIN10**. Create directory if it’s not present.

7. Click on the Take Action button and select the second action.



Name: Override Test

Create in site: Master Action Site

Create in domain: All Content

Description

This task demonstrates the use of the override command.

Actions

(Action1)

- CurrentUser Test - Click here to deploy this action.

(Action2)

- LocalUser Test - Click here to deploy this action.

8. You will be prompted for the password for the tecuser account. Enter it in the field and click OK.

9. Select BESFNDWIN10 as the target for the action and click OK.

10. When the action is complete, log into the BESFNDWIN10 system and view the contents of the c:\tmp\#####_whoami.txt file.

Exercise 46 Using Parameters

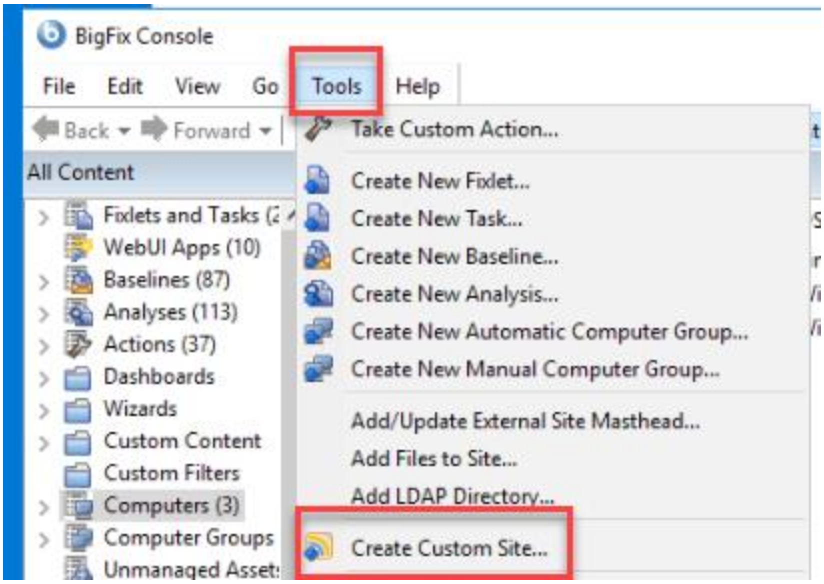
In this exercise, you develop a task in the BigFix Console that prompts the operator for the value of a property and sets the property to the specified value on the targeted computers. You also create a computer setting that is displayed in the BigFix console if it exists and has a value assigned.

This task uses the **Action Parameter Query** syntax in the action script to prompt the operator for the value of the specified setting. It also updates the setting on the target endpoint with the specified value along with the action issue date.

In this section, you create a property on the **BESFNDWIN10** virtual machine.

1. Return to the BigFix Console on the **BESFNDWIN10** virtual machine.

2. Create your custom site, Tools->Create Custom Site:
   name: **Testing**



3. Once you create the site, you need to assign Computers and operators:

   a. Computer Subscriptions tab: **Select All Computers**

   b. Operator Permissions tab: **Select adminmo and assign "Writer"**

4. Click: **SAVE Changes**

5. Select the **Computers** node in the navigation pane.
   A list of managed endpoints is shown in the list panel.

6. Right-click the **BESFNDWIN10** and select **Edit Computer Settings**. The Edit Settings window opens.

7. Click **Add**.
   The Add Custom Setting window opens.

8. In the **Setting Name** field, enter `GoldImage` and in the **Setting Value** field, enter `YES`.

9. Click **OK**.
   The Add Custom Setting window closes.

10. Click **OK**.
    The Edit Settings window closes and the Action window opens.

11. Switch to the **Fixlet Debugger** on the **BESFNDWIN10** virtual machine.

12. Select the **(qna)** tab and clear any existing queries.

13. Perform the following steps to create the Relevance expression to return the value of the specified property if it exists.

    a. Enter the following to create a query that returns the value of the property named **GoldImage**. Click **Evaluate**.

    ```
    Q: value of setting “GoldImage” of client
    ```

    The query returns a string as **“YES”**.

b. Enter the following on a new line to enhance the query using a **`whose-it`** construct as a filter to confirm that the specified setting has a value. Click **Evaluate**.

`Q: value of setting “GoldImage” whose (exists value of it) of client`

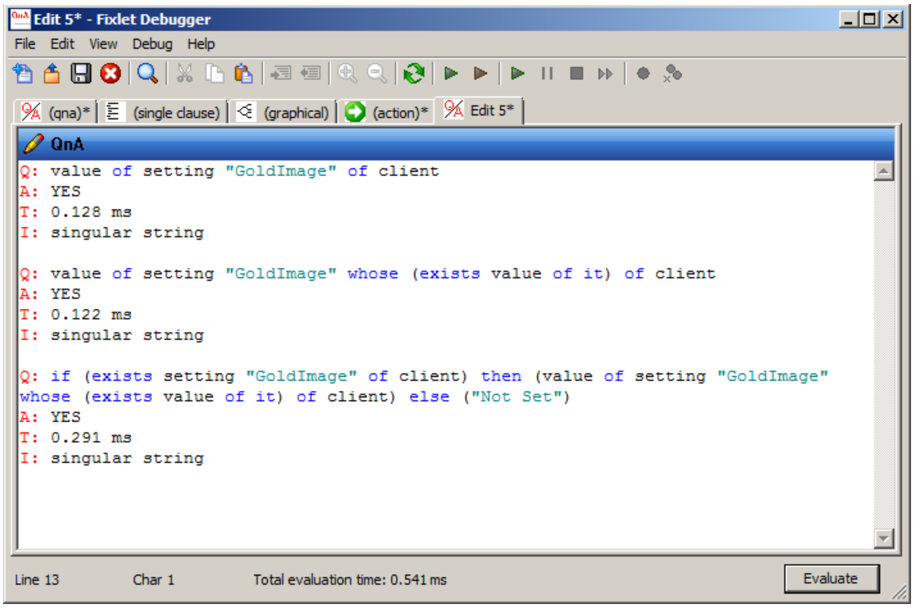
The query returns a string as **“YES”**.

c. On a new line, enter the following text to create the template for the **`if-then-else`** clause. When this query is complete, it returns the value of the specified property if it is set and if it is not set, it returns a message.

`Q: if () then () else ()`

d. Complete the query by copying the previous queries into their proper locations within the **`if-then-else`** construct. The **`if`** clause checks that the setting exists. The **`then`** clause is the expression that you created in Step b. The **`else`** clause returns an error message.

```
Q: if (exists setting “GoldImage” of client) then (value of setting “GoldImage”
whose (exists value of it) of client) else (“Not Set”)
```



```
Q: value of setting "GoldImage" of client
A: YES
T: 0.128 ms
I: singular string

Q: value of setting "GoldImage" whose (exists value of it) of client
A: YES
T: 0.122 ms
I: singular string

Q: if (exists setting "GoldImage" of client) then (value of setting "GoldImage"
whose (exists value of it) of client) else ("Not Set")
A: YES
T: 0.291 ms
I: singular string
```

Line 13   Char 1   Total evaluation time: 0.541 ms

14. Return to the BigFix console. Select **Tools > Manage Properties**. The Manage Properties window opens.

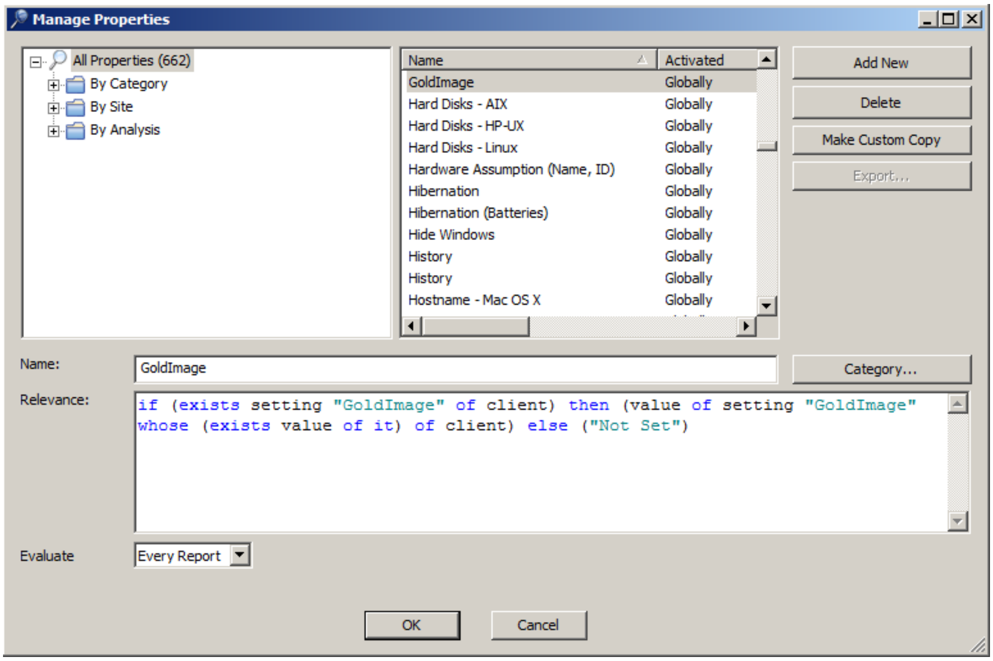
15. Click **Add New**.

The fields at the bottom of the Manage Properties window become active so that you can create a new property.

16. In the **Name** field, enter `Gold Image`.

17. In the **Relevance** field, paste the Relevance expression from the Fixlet Debugger that you created in Step d.

```
if (exists setting “GoldImage” of client) then (value of setting “GoldImage” whose
(exists value of it) of client) else (“Not Set”)
```



Name	Activated
GoldImage	Globally
Hard Disks - AIX	Globally
Hard Disks - HP-UX	Globally
Hard Disks - Linux	Globally
Hardware Assumption (Name, ID)	Globally
Hibernation	Globally
Hibernation (Batteries)	Globally
Hide Windows	Globally
History	Globally
History	Globally
Hostname - Mac OS X	Globally

Name: GoldImage

Relevance:

```
if (exists setting "GoldImage" of client) then (value of setting "GoldImage"
whose (exists value of it) of client) else ("Not Set")
```

Evaluate: Every Report

18. Click **OK**.

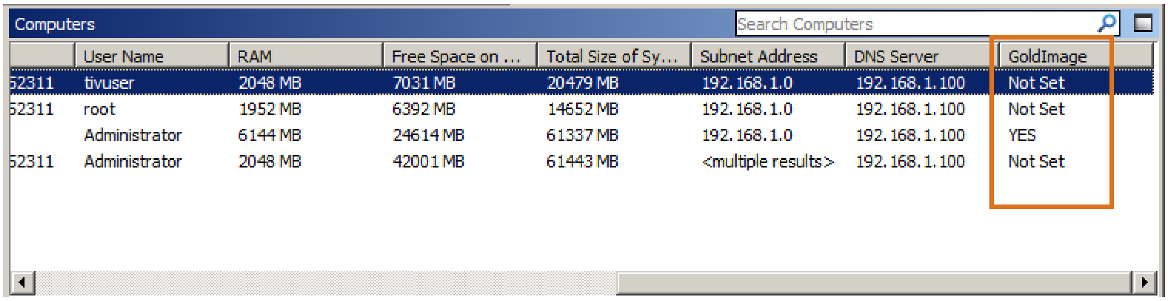
    The property is saved and the Manage Properties window closes.

19. Click the **Computers** node in the navigation pane.

    A list of managed endpoints is shown in the list pane.

20. In the Computers list pane, scroll to the far right and locate the newly created **GoldImage** property.

    The value or the error message shows for each managed endpoint in the list.



	User Name	RAM	Free Space on ...	Total Size of Sy...	Subnet Address	DNS Server	GoldImage
52311	tivuser	2048 MB	7031 MB	20479 MB	192.168.1.0	192.168.1.100	Not Set
52311	root	1952 MB	6392 MB	14652 MB	192.168.1.0	192.168.1.100	Not Set
	Administrator	6144 MB	24614 MB	61337 MB	192.168.1.0	192.168.1.100	YES
52311	Administrator	2048 MB	42001 MB	61443 MB	<multiple results>	192.168.1.100	Not Set

You now create a task to set the GoldImage property.

21. From the BigFix console menu, select **Tools > Create New Task**.

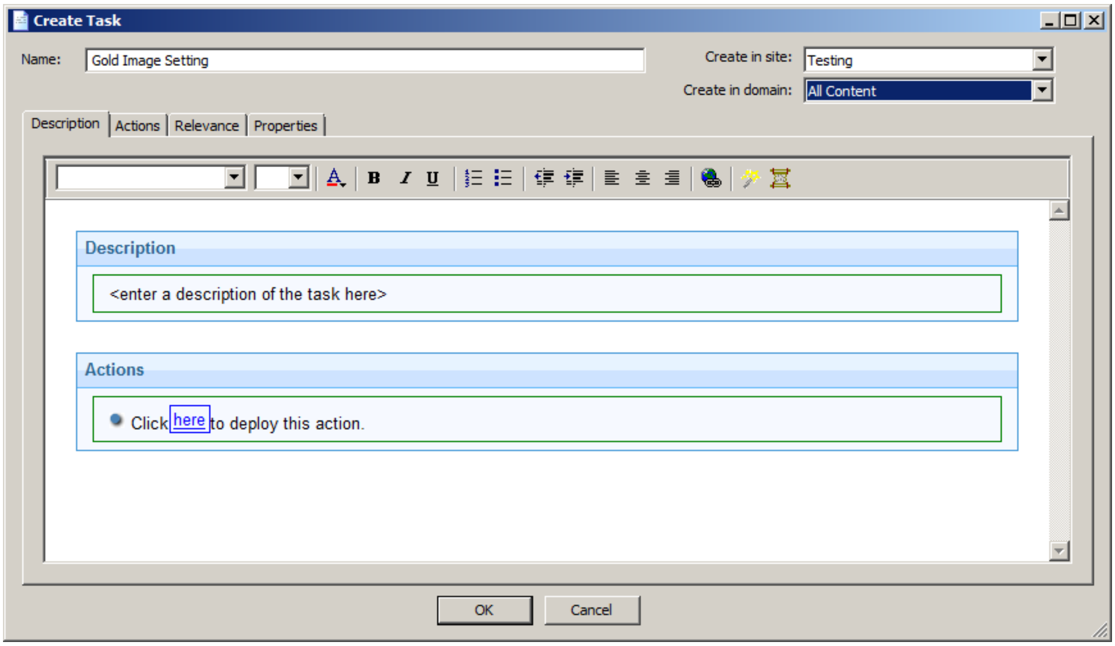
    The Create Task window opens.

22. Set the basic parameters for the task as follows:

   – **Name** field: `Gold Image Setting`

   – **Create in site**: Select **Testing** from the drop-down menu.

   – **Create in domain**: Accept the default selection of **All Content**.



Create Task

Name: Gold Image Setting

Create in site: Testing

Create in domain: All Content

Description

<enter a description of the task here>

Actions

- Click here to deploy this action.

23. Select the **Relevance** tab.

24. Select the **Computers which match all of the relevance clauses below** option.

    The default Relevance expression of **true** is displayed. Do not change the default Relevance statement. This ensures that the task is always applicable.

25. Select the **Actions** tab.

    The Action Script pane opens.

You now create the action script that is used to set the GoldImage property.

26. Perform the following steps to create the **Action Parameter Query**:

Note: The Action Parameter Query presents a window to the operator when the task that prompts for the input a value for the requested parameter is executed. This value is applicable to all targeted endpoints.

   a. Enter the following basic structure for the **`Action Parameter Query`** command in the **Action Script** pane.

```
action parameter query "" with description "" with default value ""
```

b. Modify the action script to add the parameter named **Gold_Image** within the first set of double quotation marks following keyword **query**.

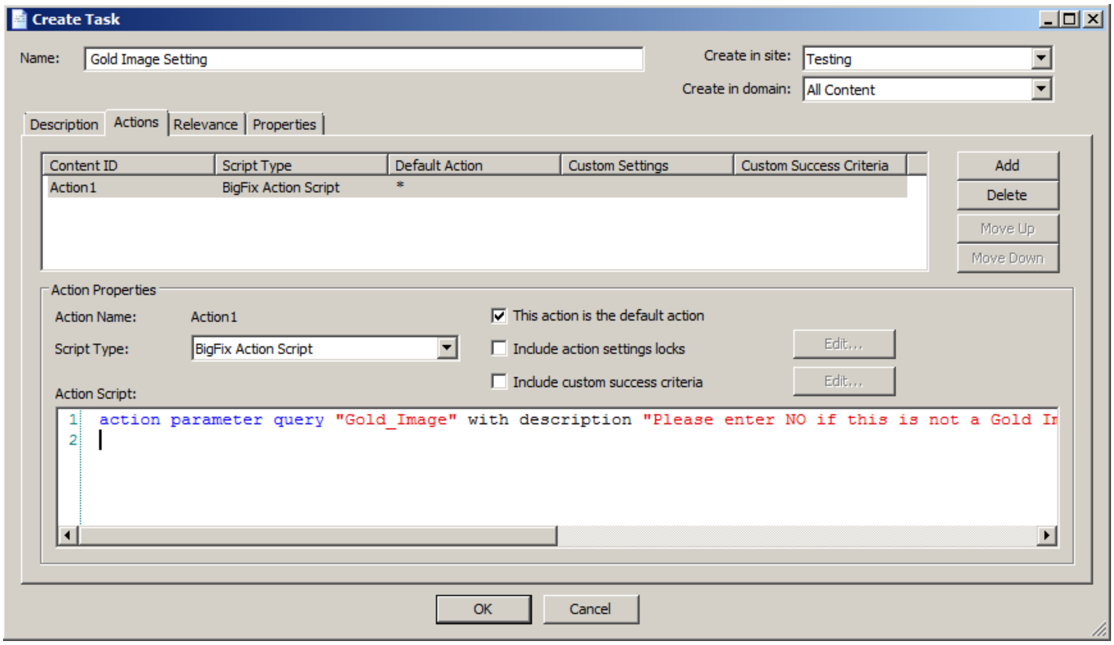
```
action parameter query "Gold_Image" with description "" with default value ""
```

c. You now modify the action script to provide a meaningful description for the parameter that appears when the operator takes action on the task. Insert the description between the double quotation marks following the keyword **description** as follows:

```
action parameter query "Gold_Image" with description "Please enter NO if this is not a Gold Image or YES if it is." with default value ""
```

d. Next, you modify the action script to provide a default value for the parameter. This value is displayed when an operator takes action on the task. The default value that you specify is normally the most common value for the parameter. For this exercise, use **NO** as the default value. Modify the action script to provide the default value between the last set of double quotation marks as follows:

```
action parameter query "Gold_Image" with description "Please enter NO if this is not a Gold Image or YES if it is." with default value "NO"
```



Name: Gold Image Setting

Create in site: Testing

Create in domain: All Content

Content ID	Script Type	Default Action	Custom Settings	Custom Success Criteria
Action1	BigFix Action Script	*		

Action Properties

Action Name: Action1

Script Type: BigFix Action Script

☑ This action is the default action

☐ Include action settings locks

☐ Include custom success criteria

Action Script:

```
action parameter query "Gold_Image" with description "Please enter NO if this is not a Gold Im

```

27. Perform the following steps to create the action script to take the value received by the **`Action Parameter Query`** command, and set the client setting on the targeted endpoints:

a. You first create the basic syntax that is used for setting a client setting on a new line in the Action Script pane as follows:

```
setting ""="{parameter "" of action}" on "" for client
```

b. Enter `GoldImage` as the name of the actual client setting between the first set of double quotation marks as follows:

```
setting "GoldImage"="" on "" for client
```

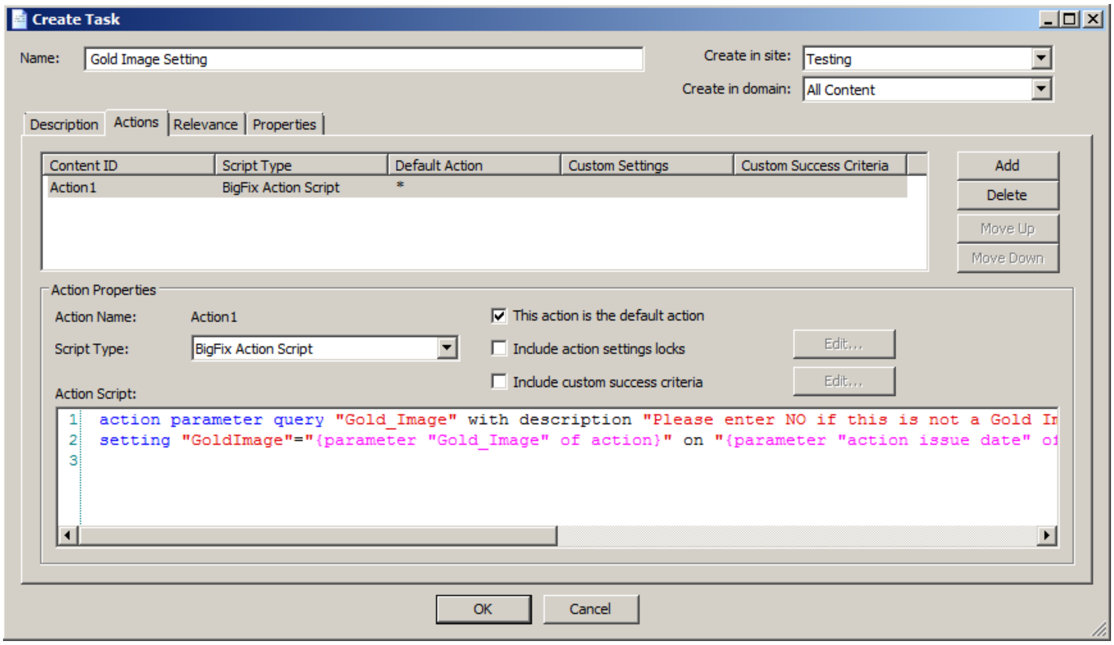
c. Modify the action script to provide the value for the setting that was collected by the **Action Parameter Query**. This value is stored in the **`Gold_Image`** parameter from the Action Parameter Query in the action script and is placed between the double quotation marks after the **parameter** keyword as follows:

```
setting “GoldImage”=“{parameter “Gold_Image” of action}” on “” for client
```

Hint: Because you are using Relevance in an action script, you must use Relevance substitution. The entire parameter expression must be enclosed within curly braces.

d. You now add the parameter to set the effective date for this setting. You must use the **parameter “action issue date” of action** expression as Relevance substitution that is placed between curly braces. You place this expression between the double quotation marks following the keyword **on** as follows:

```
setting “GoldImage”=“{parameter “Gold_Image” of action}” on “{parameter
“action issue date” of action}” for client
```


Create Task

Name: Gold Image Setting

Create in site: Testing

Create in domain: All Content

Content ID	Script Type	Default Action	Custom Settings	Custom Success Criteria
Action1	BigFix Action Script	*		

Action Properties

Action Name: Action1

Script Type: BigFix Action Script

☑ This action is the default action

□ Include action settings locks

□ Include custom success criteria

Action Script:

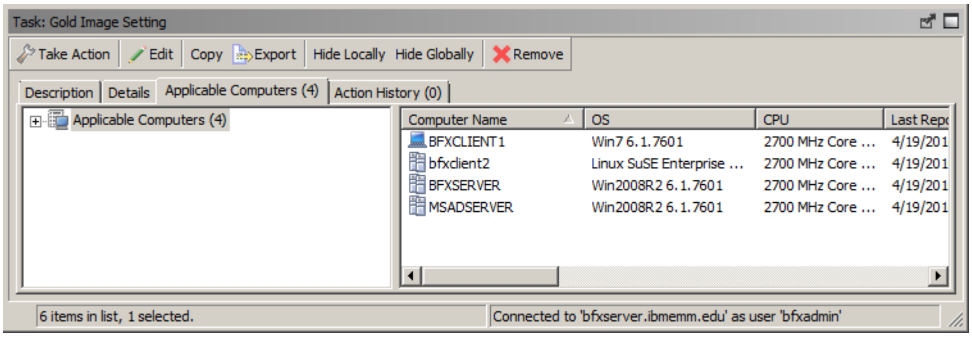
```
action parameter query "Gold_Image" with description "Please enter NO if this is not a Gold Im
setting "GoldImage"="{parameter "Gold_Image" of action}" on "{parameter "action issue date" of
```

28. Click **OK**.

The Create Task window closes and the details for the task are displayed.

You now test the custom task that you just created.

29. Click the **Applicable Computers** tab. Wait until all four managed endpoints are shown in the list of applicable computers. It might take several minutes for the clients to evaluate the Relevance and report to the server.



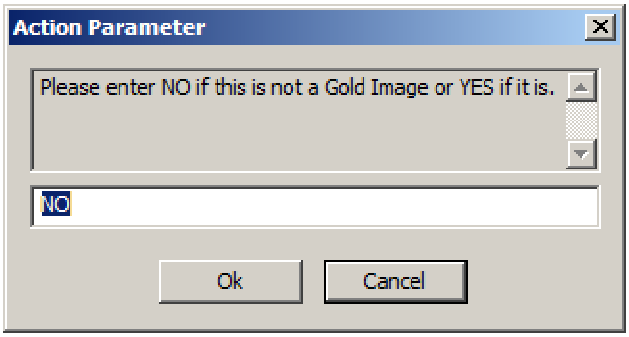
Computer Name	OS	CPU	Last Repo
BFXCLIENT1	Win7 6.1.7601	2700 MHz Core ...	4/19/201
bfxclient2	Linux SuSE Enterprise ...	2700 MHz Core ...	4/19/201
BFXSERVER	Win2008R2 6.1.7601	2700 MHz Core ...	4/19/201
MSADSERVER	Win2008R2 6.1.7601	2700 MHz Core ...	4/19/201

6 items in list, 1 selected.

Connected to 'bfxserver.ibmemm.edu' as user 'bfxadmin'

30. Click **Take Action**.

The Action Parameter window opens and is populated with the default value that you specified in the action script.



Please enter NO if this is not a Gold Image or YES if it is.

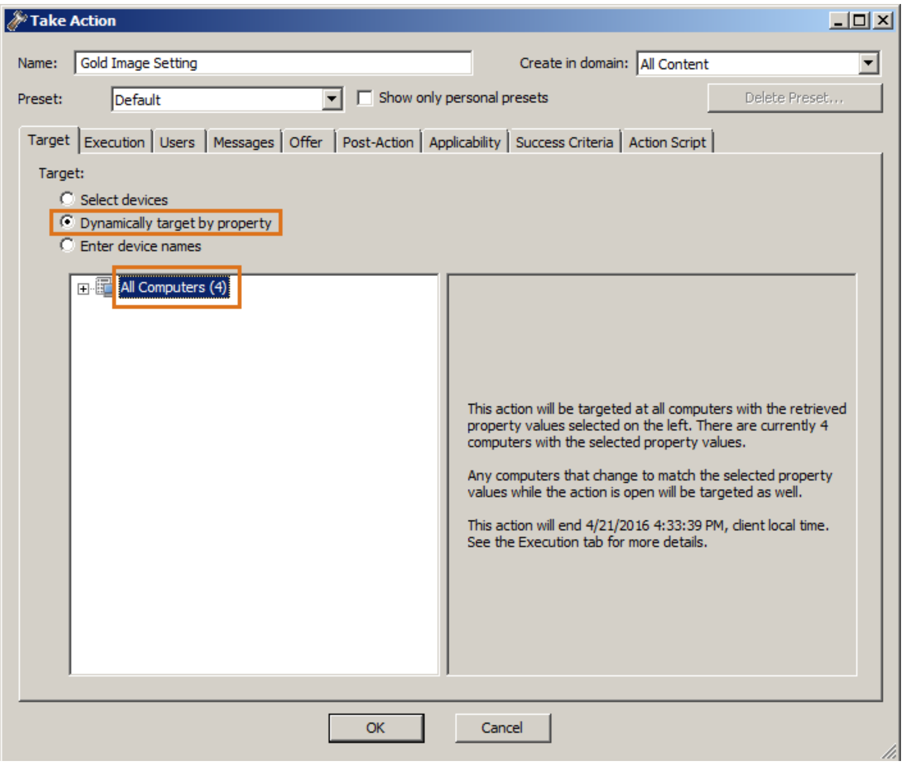
NO

31. Accept this default value of **NO**. Click **OK**.

The Take Action window opens.

32. Click the **Target** tab if it is not already selected.

33. Select the **Dynamically target by property** option and verify that **All Computers (4)** is selected.



Name: Gold Image Setting

Create in domain: All Content

Preset: Default

□ Show only personal presets

Target:

○ Select devices

● Dynamically target by property

○ Enter device names

⊞ All Computers (4)

This action will be targeted at all computers with the retrieved property values selected on the left. There are currently 4 computers with the selected property values.

Any computers that change to match the selected property values while the action is open will be targeted as well.

This action will end 4/21/2016 4:33:39 PM, client local time. See the Execution tab for more details.

34. Click **OK**.

    The Take Action window closes and the Action pane opens in the console.

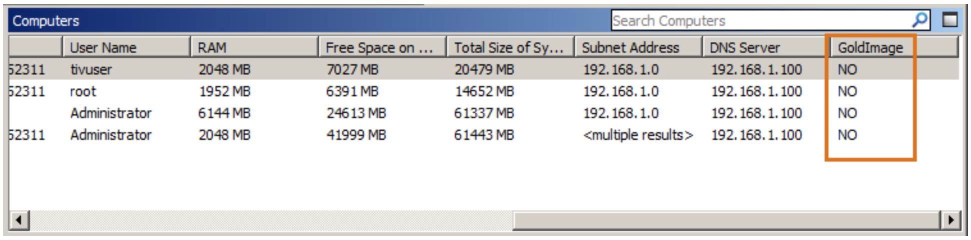
35. Monitor the status of the action and wait until it changes to **Completed** for all four managed endpoints.

36. Click the **Computers** node.

    The list of managed computers is shown in the Computers pane in the list area of the console.

37. In the Computers pane, scroll to the right and located the **Gold Image** property.

    After several minutes, the Gold Image for all systems is shown as **NO**.



	User Name	RAM	Free Space on ...	Total Size of Sy...	Subnet Address	DNS Server	GoldImage
52311	tivuser	2048 MB	7027 MB	20479 MB	192.168.1.0	192.168.1.100	NO
52311	root	1952 MB	6391 MB	14652 MB	192.168.1.0	192.168.1.100	NO
	Administrator	6144 MB	24613 MB	61337 MB	192.168.1.0	192.168.1.100	NO
52311	Administrator	2048 MB	41999 MB	61443 MB	<multiple results>	192.168.1.100	NO

Exercise 47 Creating a UNIX script

In this exercise, you create a task that generates a UNIX script that is used to start QnA or xQnA and places it on the desktop of the CENTOS7 Linux system **BESFNDCENTOS**.

More information on the QnA tool is available here:

https://developer.bigfix.com/tools/qna.html

UNIX systems support the following two versions of QnA:

- /opt/BESClient/bin/qna - a command-line version of QnA
- /opt/BESClient/bin/xqna - an X Windows version of QnA

In order for the QnA tool to work properly on a UNIX system, you set environment variables that point to the configuration location of the BigFix client and library path. The task that you create in this exercise generates the following script named `runqna.sh`and places it on the desktop of the root user:

```
#!/bin/sh
BESClientConfigPath=/var/opt/BESClient/besclient.config
export BESClientConfigPath
export LD_LIBRARY_PATH=/opt/BESClient/bin
/opt/BESClient/bin/qna
```

In this section, you create the required applicability Relevance for the task.

1. Return to the Fixlet Debugger on the **BESFNDWIN10** virtual machine and select the **(qna)** tab. Clear any existing Relevance queries from the **(qna)** tab.

2. Create a query to determine whether the operating system is Linux.

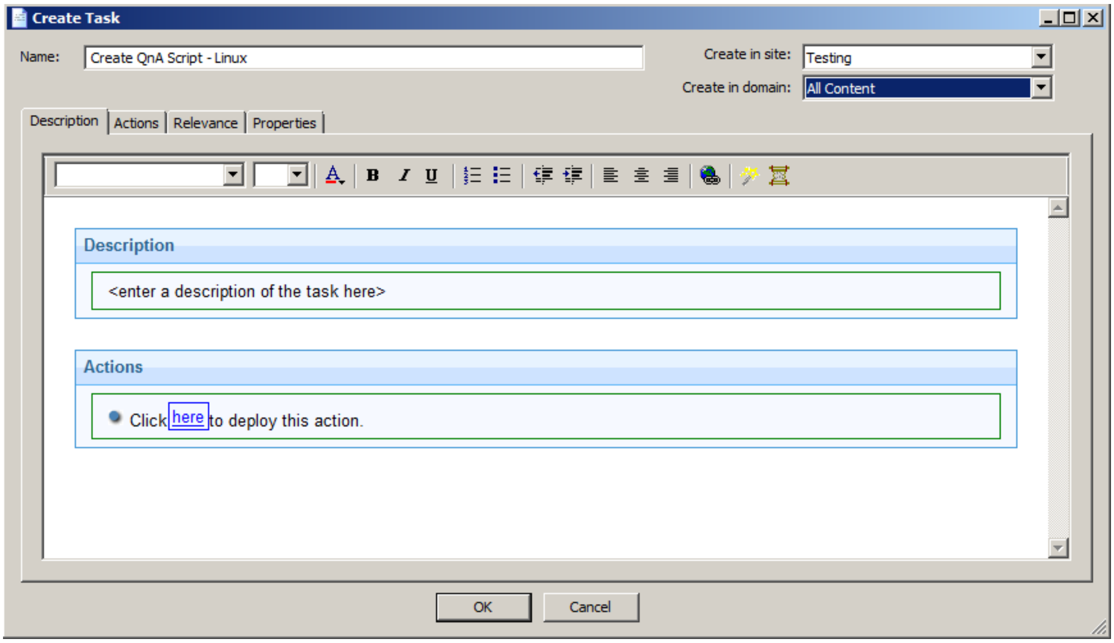
Hint: All Linux operating systems contain the string **Linux** in the operating system name so you can check that the operating system name contains *Linux*.

```
Q: name of operating system contains “Linux”
```

3. Return to the BigFix Console on the **BESFNDWIN10** virtual machine. Select **Tools > Create New Task**.

   The Create Task window opens.

4. Set the basic parameters for the task as follows:
   – **Name** field: `Create QnA Script - Linux`
   – **Create in site**: Select **Testing** from the drop-down menu.
   – **Create in domain**: Accept the default selection of **All Content**.



Create Task

Name: Create QnA Script - Linux

Create in site: Testing

Create in domain: All Content

Description

<enter a description of the task here>

Actions

- Click here to deploy this action.

5. Select the **Relevance** tab.

6. Select the **Computers which match all of the relevance clauses below** option.

   The default Relevance expression of **true** is displayed.

7. Copy the Relevance expression from the **(qna)** tab of the Fixlet Debugger that you created in Step 2, paste it into the **Relevance** text box in the console. Make sure you overwrite the existing **true** statement.

You now create the action script for the custom task.

8. Select the **Actions** tab.

9. Perform the following steps to create the action script using the **`appendfile`** action script command:

   a. In the Action Script window, enter the following line to delete any previous version of the __appendfile

   ```
   delete __appendfile
   ```

   b. On the next line, enter the following command to delete the existing file named `runqna.sh` from the desktop of the **root** user. For this task, you delete any existing `runqna.sh` that can exist on root’s desktop.

   ```
   delete /root/Desktop/runqna.sh
   ```

Note: This statement automatically deletes the file and is used in the lab environment in case you make an error in the action script while testing. Using this procedure, you do not have to manually delete the file before running the task again. Normally, you would check for the existence of the file on the desktop and proceed as required.

c. Create the `runqna.sh` script using the **`appendfile`** command before each line as follows:

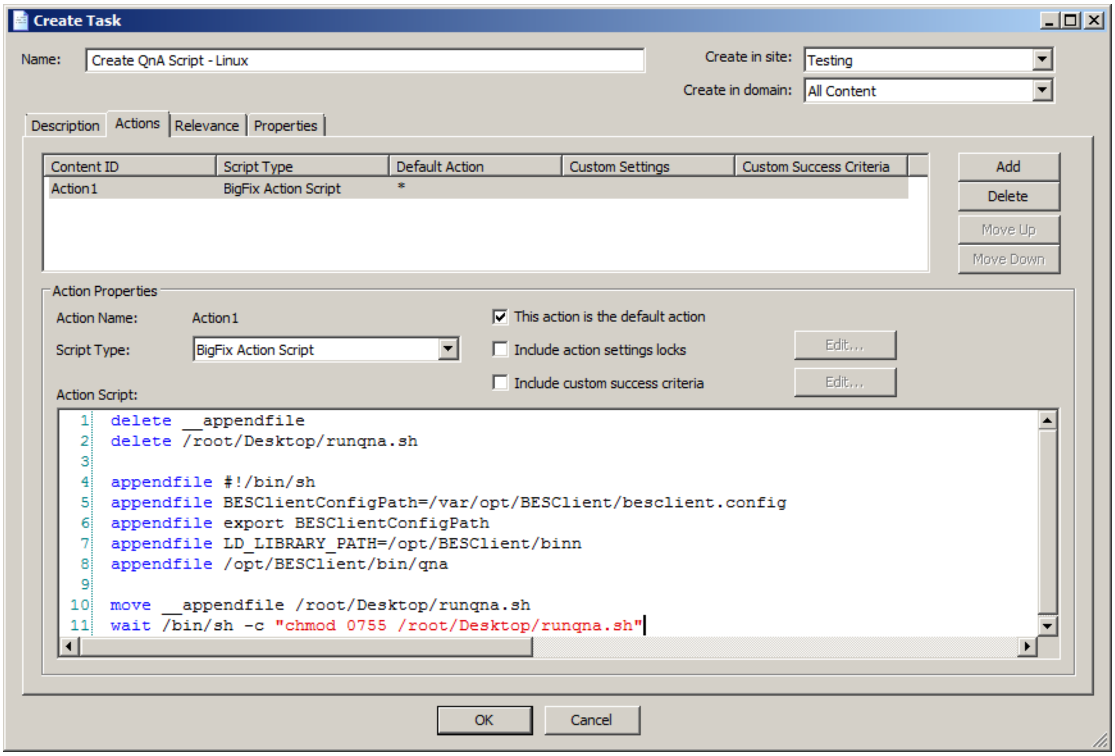
```
appendfile #!/bin/sh
appendfile BESClientConfigPath=/var/opt/BESClient/besclient.config
appendfile export BESClientConfigPath
appendfile LD_LIBRARY_PATH=/opt/BESClient/bin
appendfile /opt/BESClient/bin/qna
```

d. Now move the temporary file (`__appendfile`) to the `/root/Desktop` folder.

```
move   __appendfile /root/Desktop/runqna.sh
```

e. Add the following line in the action script to make the UNIX shell script executable using the **`chmod`** command:

```
wait /bin/sh -c “chmod 0755 /root/Desktop/runqna.sh”
```



Name: Create QnA Script - Linux

Create in site: Testing

Create in domain: All Content

Content ID	Script Type	Default Action	Custom Settings	Custom Success Criteria
Action1	BigFix Action Script	*		

Action Properties

Action Name: Action1

Script Type: BigFix Action Script

☑ This action is the default action

☐ Include action settings locks

☐ Include custom success criteria

Action Script:

```
delete __appendfile
delete /root/Desktop/runqna.sh

appendfile #!/bin/sh
appendfile BESClientConfigPath=/var/opt/BESClient/besclient.config
appendfile export BESClientConfigPath
appendfile LD_LIBRARY_PATH=/opt/BESClient/binn
appendfile /opt/BESClient/bin/qna

move __appendfile /root/Desktop/runqna.sh
wait /bin/sh -c "chmod 0755 /root/Desktop/runqna.sh"
```

Hint: You can also use the **`createfile`** command in the action script instead of **`appendfile`**.

For example, you might use the following action script if using the **`createfile`** command.

```
delete   createfile
delete /root/Desktop/runqna.sh

createfile until ZZZZZZ
#!/bin/sh
BESClientConfigPath=/var/opt/BESClient/besclient.config
export BESClientConfigPath
LD_LIBRARY_PATH=/opt/BESClient/bin
/opt/BESClient/bin/qna
ZZZZZZ

move   createfile /root/Desktop/runqna.sh
wait /bin/sh -c “chmod 0755 /root/Desktop/runqna.sh”
```

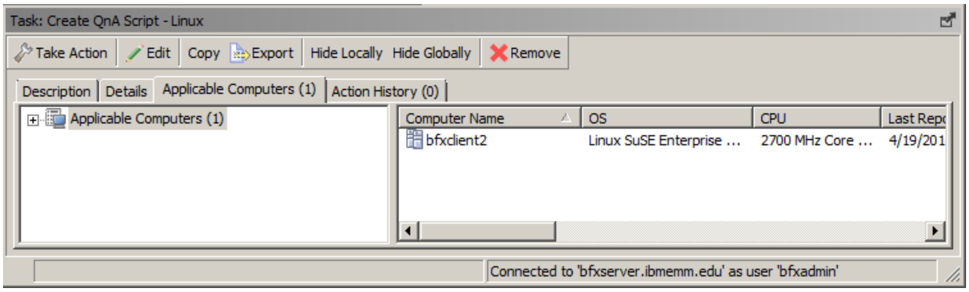
10. Review each tab in the Create Task window to verify that you have correctly configured the Relevance and action scripts.

11. Click **OK**.

    The task is saved and the Create Task window closes.

In this section, you take action on the custom task to test it.

12. Click the **Applicable Computers** tab. Wait until **BESFNDCENTOS** is shown in the list of applicable computers before continuing.



Task: Create QnA Script - Linux

Applicable Computers (1)

Computer Name	OS	CPU	Last Repo
bfxclient2	Linux SuSE Enterprise ...	2700 MHz Core ...	4/19/201

Connected to 'bfxserver.ibmemm.edu' as user 'bfxadmin'

13. Click **Take Action**.

    The Take Action window opens.

14. Click the **Target** tab if it is not already selected and **BESFNDCENTOS** from the list of available targets. Click **OK**.

    The Take Action window closes and the Action window opens and displays the status of the running action.

15. Monitor the status of the action and wait until it changes to **Completed** before continuing. This might take several minutes.

16. Switch to the **BESFNDCENTOS** virtual machine. Log on as `root` with a password of **`bigfixrocks`** if you are not already logged on.

17. Click on **Applications** in top-left and choose **Terminal**.

18. Enter the following then hit Enter

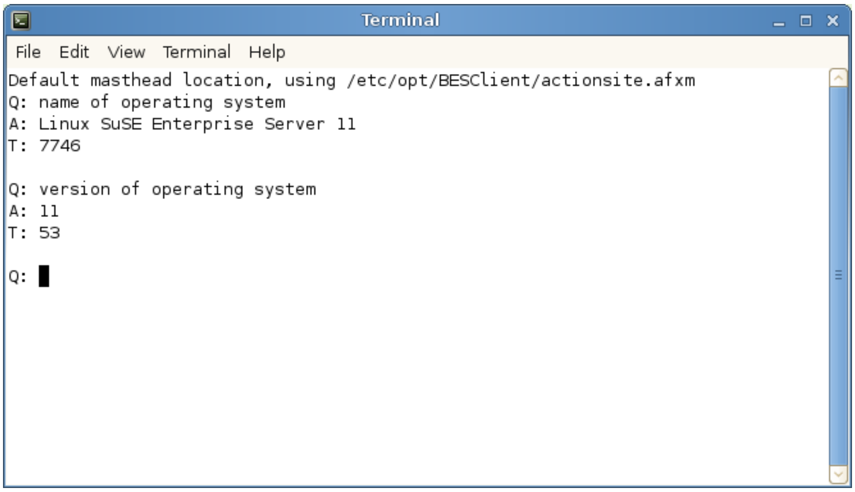
```
/root/Desktop/runqna.sh
```

The QnA application opens in a terminal window.

19. Enter the following basic Relevance commands and press Enter after each command:

```
name of operating system
version of operating system
```

Hint: Notice that the query command (**Q**:) is automatically added to each new line in the Terminal window. When you press Enter, the query is automatically evaluated.



```
Default masthead location, using /etc/opt/BESClient/actionsite.afxm
Q: name of operating system
A: Linux SuSE Enterprise Server 11
T: 7746

Q: version of operating system
A: 11
T: 53

Q: 
```

20. Press Ctrl+C.

The QnA terminal window closes.

Exercise 48 Archiving the client log files

In this exercise, you develop a task in the BigFix Console that archives the BigFix client log files. Because all clients produce log files, your action script must be capable of running on any operating system.

In this exercise, you use the following client settings to set up the archive:

- _BESClient_ArchiveManager_SendAll
- _BESClient_ArchiveManager_OperatingMode
- _BESClient_ArchiveManager_MaxArchiveSize
- _BESClient_ArchiveManager_FileSet-

Because this task must be applicable for any supported BigFix client, you do not need to create an applicability Relevance expression.

The log files that you are archiving are in the following locations:

Windows

```
C:\<parent folder of client>\__BESData\__Global\Logs
```

Linux

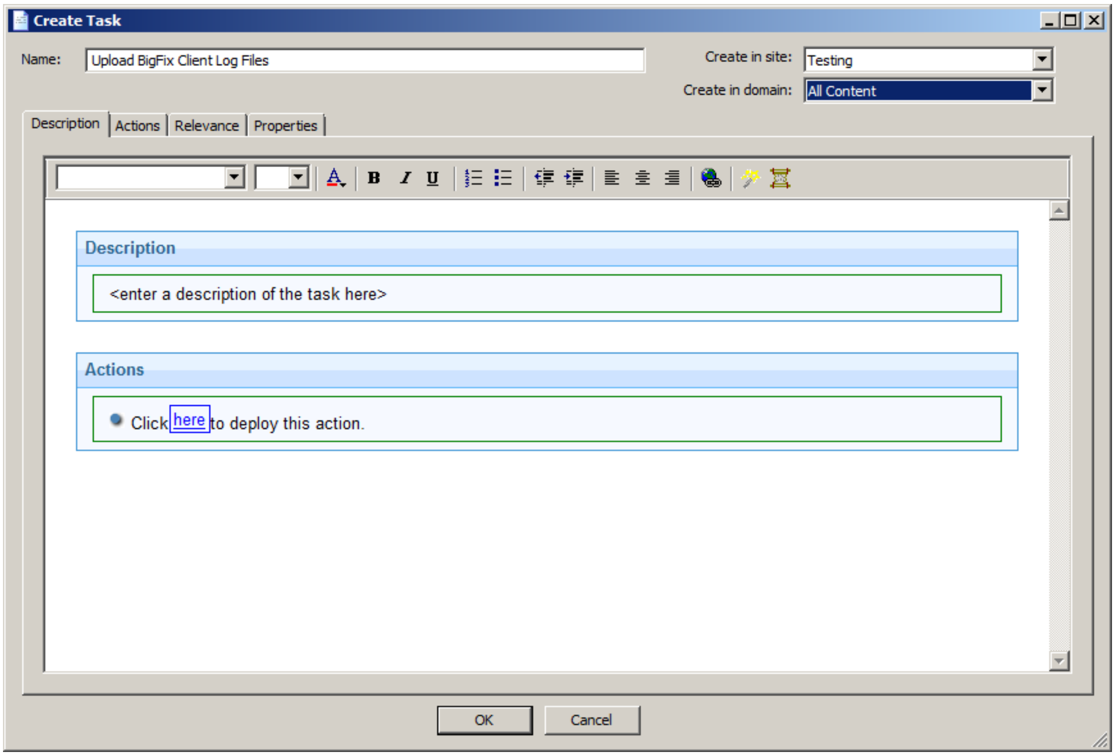
```
/var/opt/BESClient/  BESData/__Global/Logs
```

Because the log files exist in different directory locations, you must use an **`if-else-endif`** construct in the action script to establish the **_BESClient_ArchiveManager_FileSet-** setting.

1. Return to the BigFix Console on the **BESFNDWIN10** virtual machine.
2. From the console menu, select **Tools > Create New Task**.

   The Create Task window opens.

3. Set the basic parameters for the task as follows:
   - **Name** field: `Upload BigFix Client Log Files`
   - **Create in site**: Select **Testing** from the drop-down menu.
   - **Create in domain**: Accept the default selection of **All Content**.



Create Task

Name: Upload BigFix Client Log Files

Create in site: Testing

Create in domain: All Content

Description

<enter a description of the task here>

Actions

- Click here to deploy this action.

4. Select the **Relevance** tab.

5. Select the **Computers which match all of the relevance clauses below** option.

   The default Relevance expression of **true** is displayed.

Hint: Leave the default Relevance expression unchanged to ensure that the task is applicable on any system.

6. Select the **Actions** tab.

7. Perform the following steps to set the **_BESClient_ArchiveManager_SendAll** client setting:

Note: You set the value of this client setting to **1** to force all of the client log files to be archived. For this exercise, you must collect all the log files, not just the ones that have changed since the last archive.

   a. In the Action Script pane, begin by adding the basic action script template that is used to set a client setting as follows:

   ```
   setting ““ = ““ for client
   ```

   b. Insert the client setting named **_BESClient_ArchiveManager_SendAll** between the first set of double quotation marks to the left of the equal (**=**) sign.

   ```
   setting “_BESClient_ArchiveManager_SendAll” = ““ for client
   ```

   c. Insert the value of **1** between the quotation marks to the right of the equal sign.

   ```
   setting “_BESClient_ArchiveManager_SendAll” =“1” for client
   ```

   d. You now set the effective date for this setting by adding the **“{parameter “action issue date” of action}”** expression to the action script statement as follows:

   ```
   setting “_BESClient_ArchiveManager_SendAll”=“1” on “{parameter “action issue
   date” of action}” for client
   ```

8. Perform the following steps to set the **_BESClient_ArchiveManager_OperatingMode** client setting. The value of this client setting must be set to **2** to configure the archive mode to manual.

   a. Copy the final action script line that you created in Step 7.

   b. Paste the copied line as a new line in the Action Script pane.

   c. Replace the original setting **_BESClient_ArchiveManager_SendAll** with the new setting named **_BESClient_ArchiveManager_OperatingMode** as follows:

   ```
   setting “_BESClient_ArchiveManager_OperatingMode”=“1” on “{parameter “action
   issue date” of action}” for client
   ```

   d. Replace the original setting value of **1** with the new value of **2**.

   ```
   setting “_BESClient_ArchiveManager_OperatingMode”=“2” on “{parameter “action
   issue date” of action}” for client
   ```

9. Perform the following steps to set the **_BESClient_ArchiveManager_MaxArchiveSize** client setting. The value of this client setting must be set to **4194304** to represent a 4 MB file size.

   a. Copy the action script line that you created in Step 8.

   b. Paste the copied line as a new line in the Action Script pane.

c. Replace the original setting **_BESClient_ArchiveManager_SendAll** with the new setting named **_BESClient_ArchiveManager_MaxArchiveSize** as follows:

```
setting "_BESClient_ArchiveManager_MaxArchiveSize"="2" on "{parameter "action
issue date" of action}" for client
```

d. Replace the original setting value of **2** with the new value of **4194304** as follows:

```
setting "_BESClient_ArchiveManager_MaxArchiveSize"="4194304" on "{parameter
"action issue date" of action}" for client
```

10. Perform the following steps to set the **_BESClient_ArchiveManager_FileSet-** client setting.

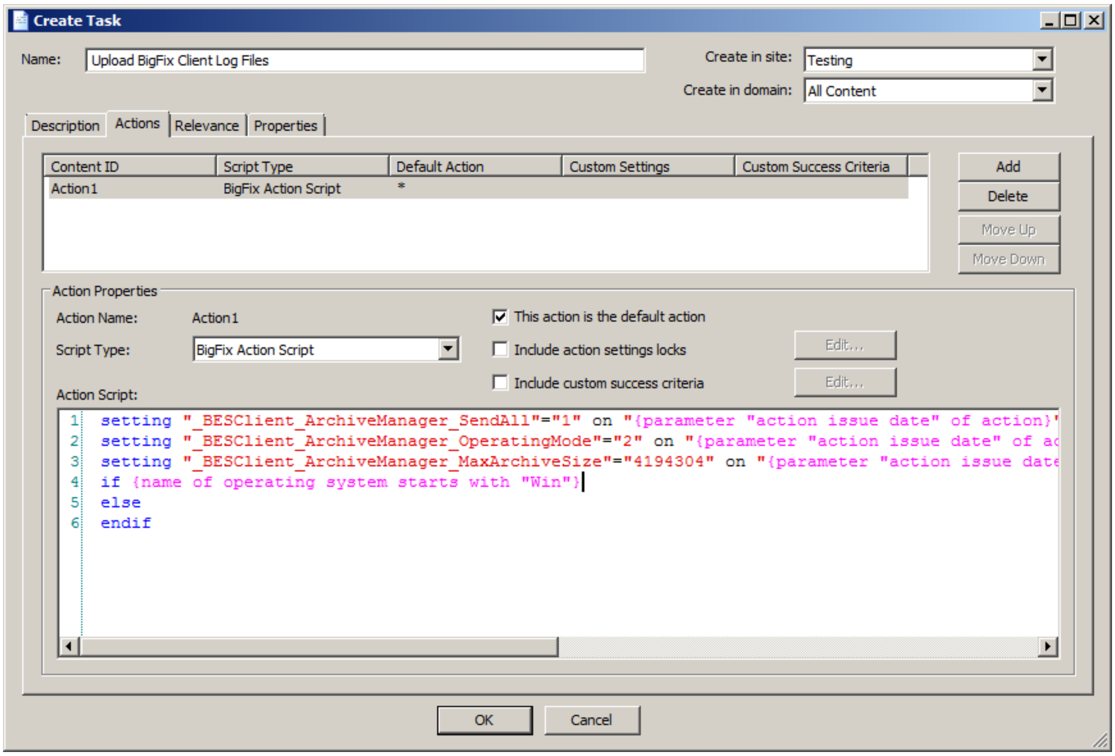
    Because there are multiple operating system types, you must use an **`if-else-endif`** construct in the action script that is used to check for Windows and non-Windows systems.

11. Create a new line in the Action Script pane and enter the **`if-else-endif`** action script structure as follows:

```
if {}
else
endif
```

12. Create the following Relevance substitution expression for the **`if`** statement to check whether the operating system starts with **Win**.

```
if {name of operating system starts with "Win"}
```



Name: Upload BigFix Client Log Files

Create in site: Testing

Create in domain: All Content

Content ID	Script Type	Default Action	Custom Settings	Custom Success Criteria
Action1	BigFix Action Script	*		

Action Name: Action1

Script Type: BigFix Action Script

☑ This action is the default action

☐ Include action settings locks

☐ Include custom success criteria

Action Script:

```
setting "_BESClient_ArchiveManager_SendAll"="1" on "{parameter "action issue date" of action}"
setting "_BESClient_ArchiveManager_OperatingMode"="2" on "{parameter "action issue date" of ac
setting "_BESClient_ArchiveManager_MaxArchiveSize"="4194304" on "{parameter "action issue date
if {name of operating system starts with "Win"}
else
endif
```

13. You now create the action script statement that is executed when the `if` statement is **True**. In this case, you must set the value of the client setting **_BESClient_ArchiveManager_FileSet-** to the folder where the client log files for Windows endpoints are located.

    a. Insert a blank line after the `if` statement you created in Step 12.

    b. Copy the first line of the action script that is setting the value for the **_BESClient_ArchiveManager_SendAll** client setting and paste it into the blank line.

    c. Replace the setting named **_BESClient_ArchiveManager_SendAll** with the new setting named **_BESClient_ArchiveManager_FileSet-** as follows:

Important: Make sure that you do not omit the dash character (**-**) at the end of the setting name.

```
setting “_BESClient_ArchiveManager_FileSet-”=“1” on “{parameter “action issue
date” of action}” for client
```

    d. You must provide a tag following the setting name to indicate the files that you want to archive. Append **(log)** to the setting name as follows to indicate that you want to archive the `.log` files:

```
setting “_BESClient_ArchiveManager_FileSet-(log)”=“1” on “{parameter “action
issue date” of action}” for client
```

    e. You now must specify the installation folder of the BigFix client the Relevance expression **pathname of parent folder of client**. The relative path to the client log files below the installation folder is always `BESData\__Global\Logs`. Modify the expression as follows to specify the full path to the BigFix client log files on Windows systems:

```
setting “_BESClient_ArchiveManager_FileSet-(log)”=“{pathname of parent
folder of client}\__BESData\__Global\Logs” on “{parameter “action issue
date” of action}” for client
```

14. Perform the following steps to create the action script statement that is executed when the `if` statement evaluates as **False**.

Note: Because the lab environment has only two operating system types, you use the simplified if-else-endif condition branching structure.

    a. Insert a blank line after the `else` statement that you created in Step 11 on page 153.

    b. Copy the action script line that you created in Step e and paste it into the blank line.

c. Replace the **`{pathname of parent folder of client}`** Relevance substitution clause with the Linux path `/var/opt/BESClient` as follows:

Important: The UNIX operating system is case sensitive, so the path names must be entered exactly as shown.

```
setting
"_BESClient_ArchiveManager_FileSet-(log)"="/var/opt/BESClioent\__BESData\__G
lobal\Logs" on "{parameter "action issue date" of action}" for client
```

d. The remaining relative path to the client log files is the same, but you must change the backslashes (\) in the path to forward slashes (**/**) as follows:

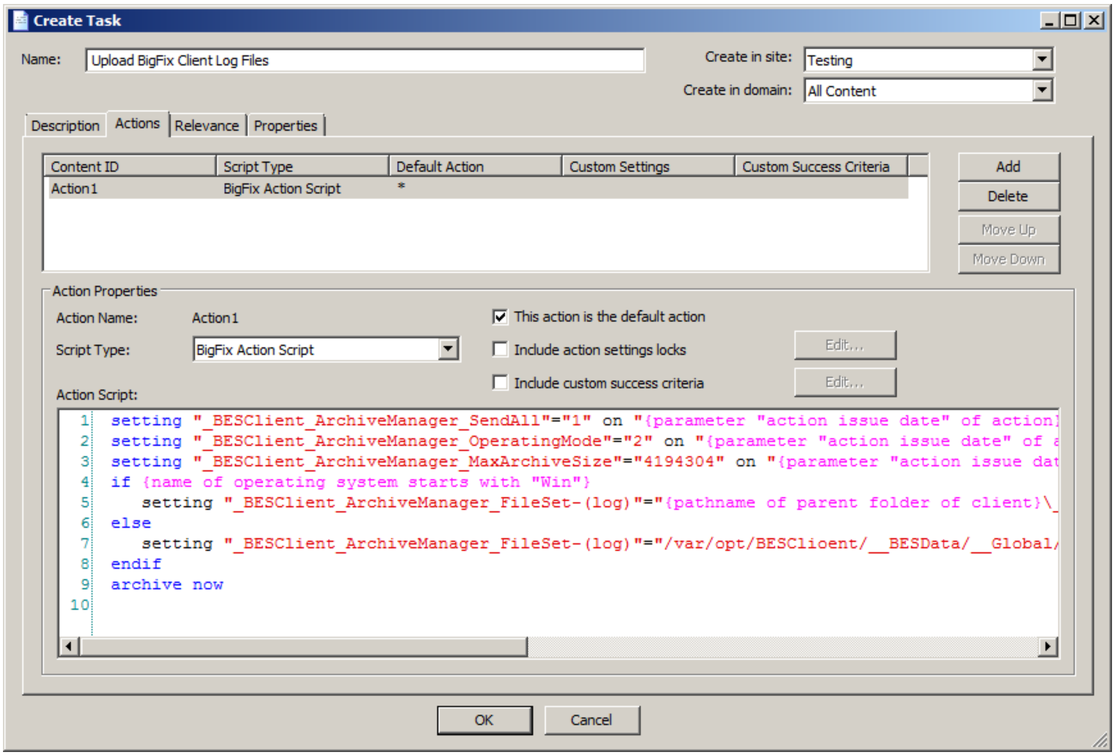
```
setting
"_BESClient_ArchiveManager_FileSet-(log)"="/var/opt/BESClioent/__BESData/__G
lobal/Logs" on "{parameter "action issue date" of action}" for client
```

15. You now add the following line to the action script to perform the archive of the client log files:

```
archive now
```

The complete action script is shown below:

```
setting "_BESClient_ArchiveManager_SendAll"="1" on "{parameter "action issue
date" of action}" for client
setting "_BESClient_ArchiveManager_OperatingMode"="2" on "{parameter "action
issue date" of action}" for client
setting "_BESClient_ArchiveManager_MaxArchiveSize"="4194304" on "{parameter
"action issue date" of action}" for client
if {name of operating system starts with "Win"}
   setting "_BESClient_ArchiveManager_FileSet-(log)"="{pathname of parent
folder of client}\__BESData\__Global\Logs" on "{parameter "action issue
date" of action}" for client
else
   setting "_BESClient_ArchiveManager_FileSet-
   (log)"="/var/opt/BESClient/__BESData/__Global/Logs" on "{parameter "action
   issue date" of action}" for client
endif
archive now
```



Name: Upload BigFix Client Log Files

Create in site: Testing

Create in domain: All Content

Content ID	Script Type	Default Action	Custom Settings	Custom Success Criteria
Action1	BigFix Action Script	*		

Action Properties

Action Name: Action1

Script Type: BigFix Action Script

☑ This action is the default action

☐ Include action settings locks

☐ Include custom success criteria

Action Script:

```
setting "_BESClient_ArchiveManager_SendAll"="1" on "{parameter "action issue date" of action}
setting "_BESClient_ArchiveManager_OperatingMode"="2" on "{parameter "action issue date" of a
setting "_BESClient_ArchiveManager_MaxArchiveSize"="4194304" on "{parameter "action issue dat
if {name of operating system starts with "Win"}
   setting "_BESClient_ArchiveManager_FileSet-(log)"="{pathname of parent folder of client}\_
else
   setting "_BESClient_ArchiveManager_FileSet-(log)"="/var/opt/BESClioent/__BESData/__Global/
endif
archive now
```

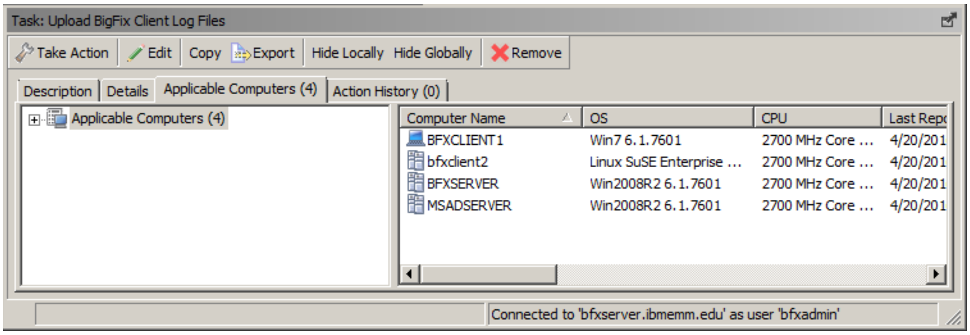
16. Review each tab of the custom task to ensure that the action script and Relevance are entered correctly.

17. Click **OK**.

    The custom task is created and the Create Task window closes. The details for the new custom task are shown in the BigFix Console.

In this section, you take action on the custom task and verify that task archived the client log files as expected.

18. Click the **Applicable Computers** tab. Wait until all four managed endpoints show in the list of applicable computers before continuing.



Computer Name	OS	CPU	Last Repo
BFXCLIENT1	Win7 6.1.7601	2700 MHz Core ...	4/20/201
bfxclient2	Linux SuSE Enterprise ...	2700 MHz Core ...	4/20/201
BFXSERVER	Win2008R2 6.1.7601	2700 MHz Core ...	4/20/201
MSADSERVER	Win2008R2 6.1.7601	2700 MHz Core ...	4/20/201

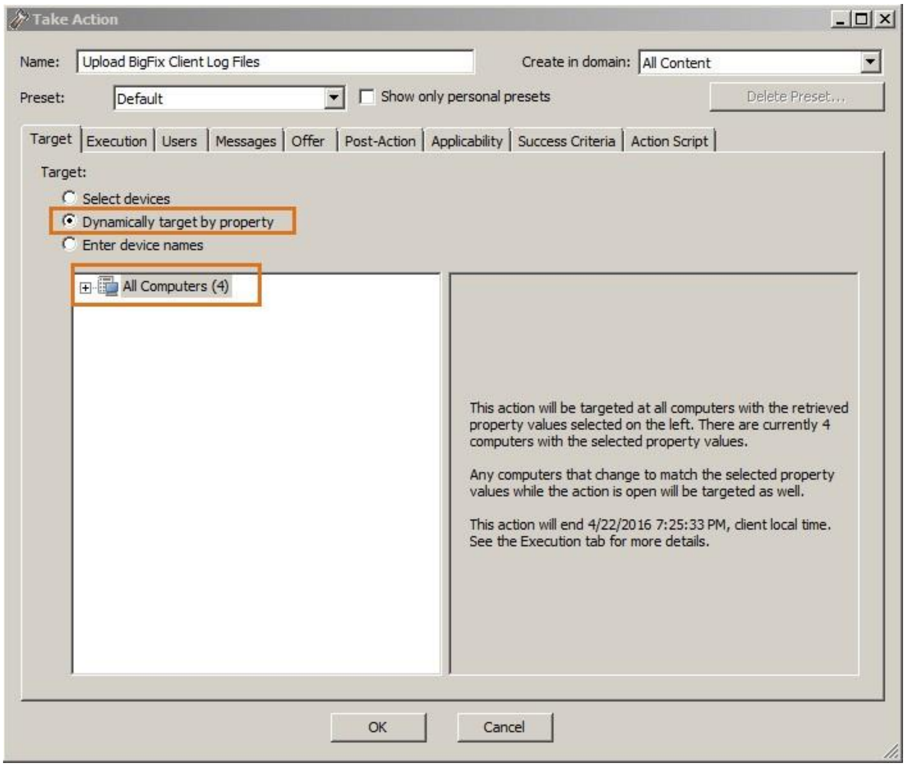
Connected to 'bfxserver.ibmemm.edu' as user 'bfxadmin'

19. Click **Take Action**.

    The Take Action window opens.

20. Click the **Target** tab if it is not already selected.

21. Select the **Dynamically target by property** option and verify that **All Computers (4)** is selected.



Name: Upload BigFix Client Log Files

Create in domain: All Content

Preset: Default

□ Show only personal presets

Target:

○ Select devices

● Dynamically target by property

○ Enter device names

All Computers (4)

This action will be targeted at all computers with the retrieved property values selected on the left. There are currently 4 computers with the selected property values.

Any computers that change to match the selected property values while the action is open will be targeted as well.

This action will end 4/22/2016 7:25:33 PM, client local time. See the Execution tab for more details.

22. Click **OK**.

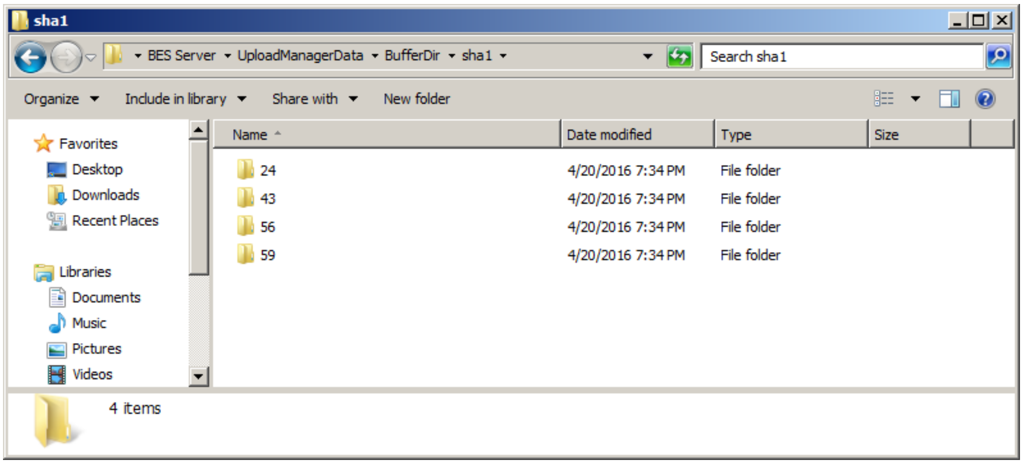
    The Take Action window closes and the Action pane opens in the console.

23. Monitor the status of the action and wait until it changes to **Completed** for all four managed endpoints before continuing.

24. Open Windows Explorer on the **BESFNDWINROOT** virtual machine if it is not already open.

25. Navigate to the `C:\Program Files (x86)\BigFix Enterprise\BES Server\UploadManagerData\BufferDir\sha1` folder.

Hint: The folders that are listed within this directory correspond to the last two digits of the ID for the managed systems. You can show the ID in the Computers pane of the console by selecting the ID property from the Column Picker.



Name	Date modified	Type	Size
24	4/20/2016 7:34 PM	File folder	
43	4/20/2016 7:34 PM	File folder	
56	4/20/2016 7:34 PM	File folder	
59	4/20/2016 7:34 PM	File folder	

26. In Windows Explorer, navigate to one of the folders that are listed in the **sha1** directory.

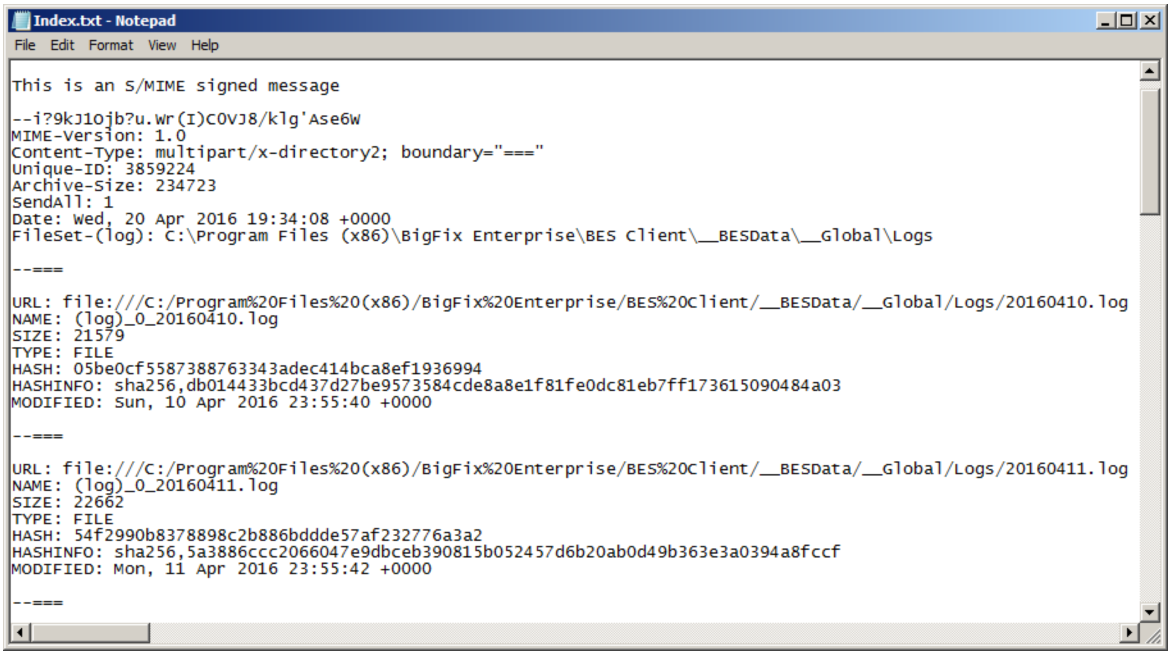
    Additional folders are listed with names that correspond to the full Computer ID of that managed endpoint.

27. In Windows Explorer, navigate to the folder that corresponds to the full **Computer ID** ofthe managed system that it represents.

    A list of files that begin with **(log)_0_** before the actual client log are displayed. These files represent the client log files that have been archived from the managed endpoint. A file named `Index.txt` also contains the details for the archive.

28. Double-click the `Index.txt` file and review its contents.

Hint: The value for **Fileset** corresponds to the path to the BigFix client log files that you set in the client settings through the action script for the task. Note the values for the URLs that were used to upload the client log files.



```
This is an S/MIME signed message

--i?9kJ1Ojb?u.Wr(I)C0VJ8/klg'Ase6W
MIME-Version: 1.0
Content-Type: multipart/x-directory2; boundary="==="
Unique-ID: 3859224
Archive-Size: 234723
SendAll: 1
Date: Wed, 20 Apr 2016 19:34:08 +0000
FileSet-(log): C:\Program Files (x86)\BigFix Enterprise\BES Client\__BESData\__Global\Logs

--===

URL: file:///C:/Program%20Files%20(x86)/BigFix%20Enterprise/BES%20Client/__BESData/__Global/Logs/20160410.log
NAME: (log)_0_20160410.log
SIZE: 21579
TYPE: FILE
HASH: 05be0cf5587388763343adec414bca8ef1936994
HASHINFO: sha256,db014433bcd437d27be9573584cde8a8e1f81fe0dc81eb7ff173615090484a03
MODIFIED: Sun, 10 Apr 2016 23:55:40 +0000

--===

URL: file:///C:/Program%20Files%20(x86)/BigFix%20Enterprise/BES%20Client/__BESData/__Global/Logs/20160411.log
NAME: (log)_0_20160411.log
SIZE: 22662
TYPE: FILE
HASH: 54f2990b8378898c2b886bddde57af232776a3a2
HASHINFO: sha256,5a3886ccc2066047e9dbceb390815b052457d6b20ab0d49b363e3a0394a8fccf
MODIFIED: Mon, 11 Apr 2016 23:55:42 +0000

--===
```

29. Close the `Index.txt` file.

Exercise 49 Downloading and installing an application

In this exercise, you develop a task to download and install 7-Zip. This program is freely available at http://www.7-zip.org/.

For this exercise, you create a task that meets the following criteria:

- You create the applicability Relevance so that the task is only relevant for all of the variants of Win10 64-bit operating systems.
- You also query the following Windows registry key to determine whether 7-Zip is currently installed. The GUID will vary depending on software version.

  `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{23170F69-40C1-2702-0920-000001000000}`

Note: You can automatically create the task that is created during this lab using the Software Distribution wizards in BigFix. However, building this task manually is intended to help you understand how to create actions to download files, which is a common requirement when you are developing custom content.

When 7-Zip is installed on a system, the following key to the registry key is created. The action uses this key to determine whether the task is applicable.

The installation file for 7-Zip is on the **BESFNDWIN10** virtual machine in the following directory:

```
C:\7-Zip\7z1900-x64.msi
```

In this section, you create the required applicability Relevance for the custom task.

1. Switch to the **BESFNDWIN10** virtual machine and return to the **(qna)** tab In the Fixlet Debugger.

2. Begin by developing the Relevance expression to test for the Win10 operating system.

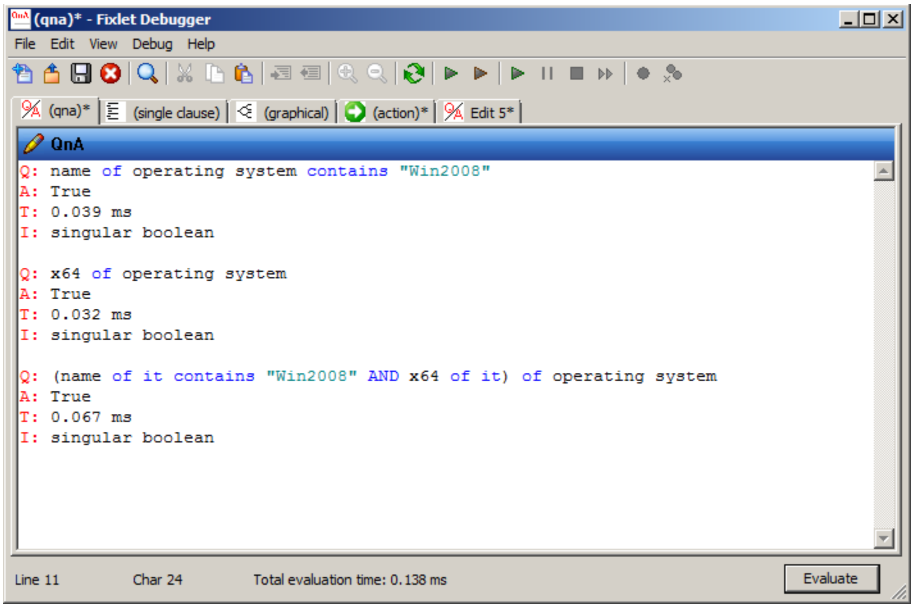
   ```
   Q: name of operating system contains “Win10”
   ```

3. On a new line in the **(qna)** tab, create a query to test whether the operating system is 64-bit using the property **x64 of <operating system>**.

   ```
   Q: x64 of operating system
   ```

4. On a new line in the **(qna)** tab, create a query that uses an **`it-without-a-whose`** construct to reference the operating system object only once as follows:

   ```
   Q: (name of it contains “Win10” AND x64 of it) of operating system
   ```



```
Q: name of operating system contains "Win2008"
A: True
T: 0.039 ms
I: singular boolean

Q: x64 of operating system
A: True
T: 0.032 ms
I: singular boolean

Q: (name of it contains "Win2008" AND x64 of it) of operating system
A: True
T: 0.067 ms
I: singular boolean
```

5. Perform the following steps to create the Relevance expression that is used to test the registry to determine whether 7-Zip can be installed on the endpoint.

   a. Open the Registry Editor by entering `regedit` in the search box of the Windows Start menu and click **regedit.exe** in the filtered list of programs.

      The Registry Editor window opens.

   b. Locate the following key in the Registry Editor:

      ```
      HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall
      ```

   c. Right-click the registry key name and select **Copy Key Name** from the menu.

d. Return to the Fixlet Debugger, start a new line in the **(qna)** tab.

e. Begin creating the query by entering the following template code for the registry expression:

```
Q: key "" of x64 registry
```

Note: The 7-Zip application that you are installing is a 64-bit application and Windows does not redirect a 64-bit application. The **x64 registry** object is used to force the BigFix agent to use the 64-bit registry.

f. Paste the registry key value between the double quotation marks in the query from Step e.

```
Q: key
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall" of
x64 registry
```

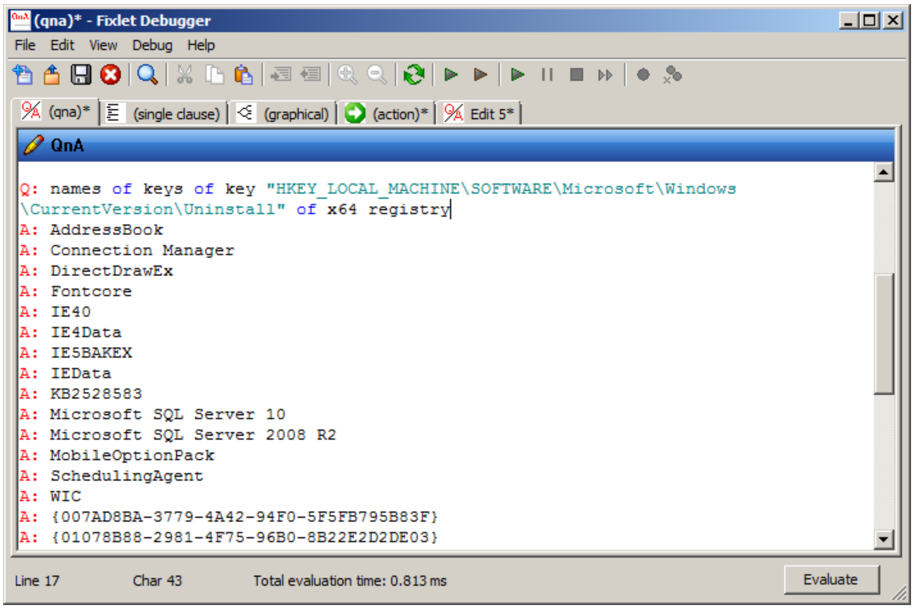
g. Modify the Relevance expression to add the string **names of keys of** before the key in the current expression.

Hint: This expression extracts the names of all of the subkeys of the specified key. Make sure to use the plural form because the expression returns multiple values.

```
Q: names of keys of key
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall" of
x64 registry
```

h. Click **Evaluate**.

The query returns a list of all of the subkeys for the specified registry key.



```
Q: names of keys of key "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
\CurrentVersion\Uninstall" of x64 registry
A: AddressBook
A: Connection Manager
A: DirectDrawEx
A: Fontcore
A: IE40
A: IE4Data
A: IE5BAKEX
A: IEData
A: KB2528583
A: Microsoft SQL Server 10
A: Microsoft SQL Server 2008 R2
A: MobileOptionPack
A: SchedulingAgent
A: WIC
A: {007AD8BA-3779-4A42-94F0-5F5FB795B83F}
A: {01078B88-2981-4F75-96B0-8B22E2D2DE03}
```

Line 17 Char 43 Total evaluation time: 0.813 ms

i. In the Fixlet Debugger, press Ctrl+R.

The responses for all of the Evaluated queries are removed and only the queries are shown.

j. Modify the query to use a **`whose-it`**construct to extract the required key.

Hint: The **whose-it** filter is applied to the subkeys of the specified key, so insert the construct after the **keys** keyword.

Q: names of keys **whose (it)** of key “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall” of x64 registry

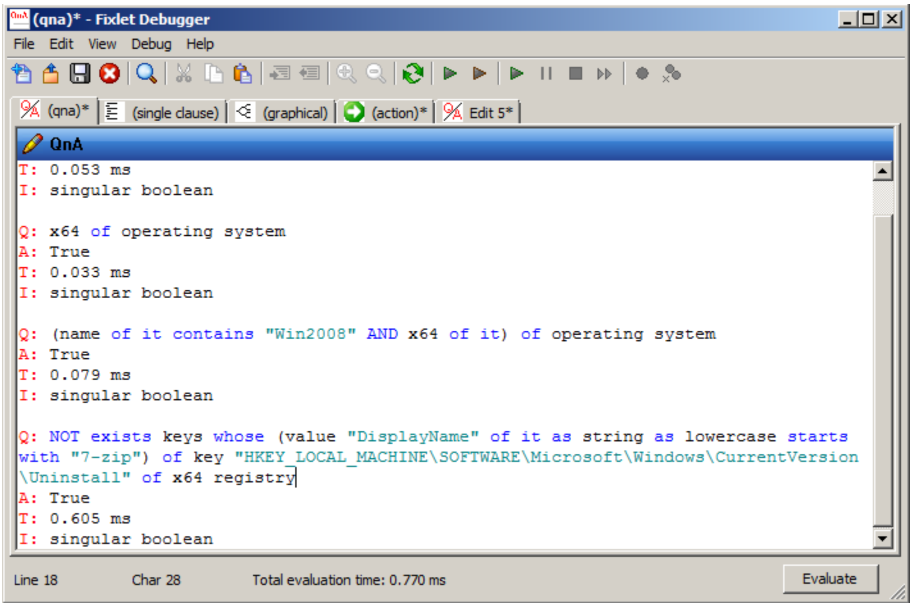
k. Complete the filter by replacing **(it)** with the key that has a **DisplayName** value that contains **7-zip**. Remember to convert the filter to a string before you convert it to lowercase.

Q: names of keys whose (**value “DisplayName” of it as string as lowercase starts with “7-zip”**) of key “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall” of x64 registry

l. Modify the Relevance query so that when it is used for the applicability Relevance, it returns a Boolean True if the specified registry key does not exist on the endpoint. This result indicates that 7-Zip is not currently installed. To make this change, you must convert this expression to return a Boolean value by checking that the specified key does not exist.

Q: NOT exists keys whose (value “DisplayName” of it as string as lowercase starts with “7-zip”) of key

```
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall" of
x64 registry
```



```
T: 0.053 ms
I: singular boolean

Q: x64 of operating system
A: True
T: 0.033 ms
I: singular boolean

Q: (name of it contains "Win2008" AND x64 of it) of operating system
A: True
T: 0.079 ms
I: singular boolean

Q: NOT exists keys whose (value "DisplayName" of it as string as lowercase starts
with "7-zip") of key "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion
\Uninstall" of x64 registry
A: True
T: 0.605 ms
I: singular boolean
```

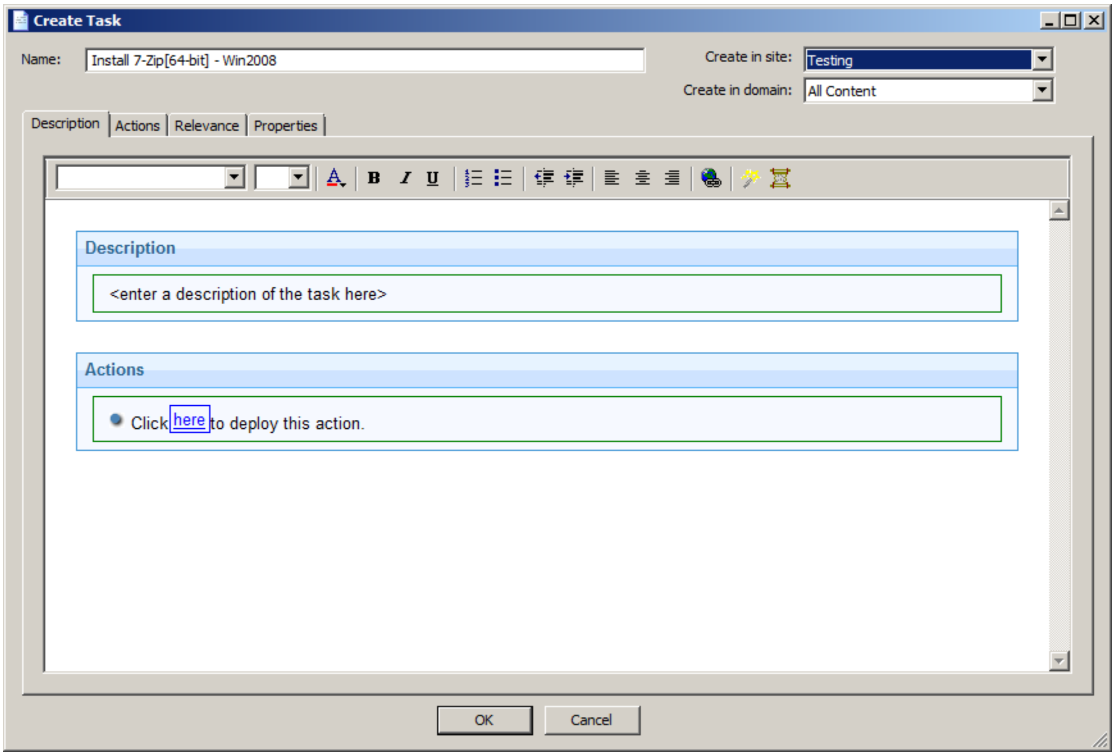
In this section, you create the custom task using the Relevance queries that you developed in the Fixlet Debugger.

6. Return to the BigFix Console on the **BESFNDWIN10** virtual machine.

7. From the console menu, select **Tools > Create New Task**.

   The Create Task window opens.

8. Set the basic parameters for the task as follows:
   - **Name** field: `Install 7-Zip[64-bit]` – `Win10`
   - **Create in site**: Select **Testing** from the drop-down menu.
   - **Create in domain**: Accept the default selection of **All Content**.



Create Task

Name: Install 7-Zip[64-bit] - Win2008

Create in site: Testing

Create in domain: All Content

Description

<enter a description of the task here>

Actions

- Click here to deploy this action.

9. Select the **Relevance** tab.

10. Select the **Computers which match all of the relevance clauses below** option.

    The default Relevance expression of **true** is displayed.

11. Copy the Relevance expression from the **(qna)** tab of the Fixlet Debugger that you created in Step 4 on page 160 and paste it into the **Relevance** text box, replacing any current content.

    ```
    (name of it contains “Win10” AND x64 of it) of operating system
    ```

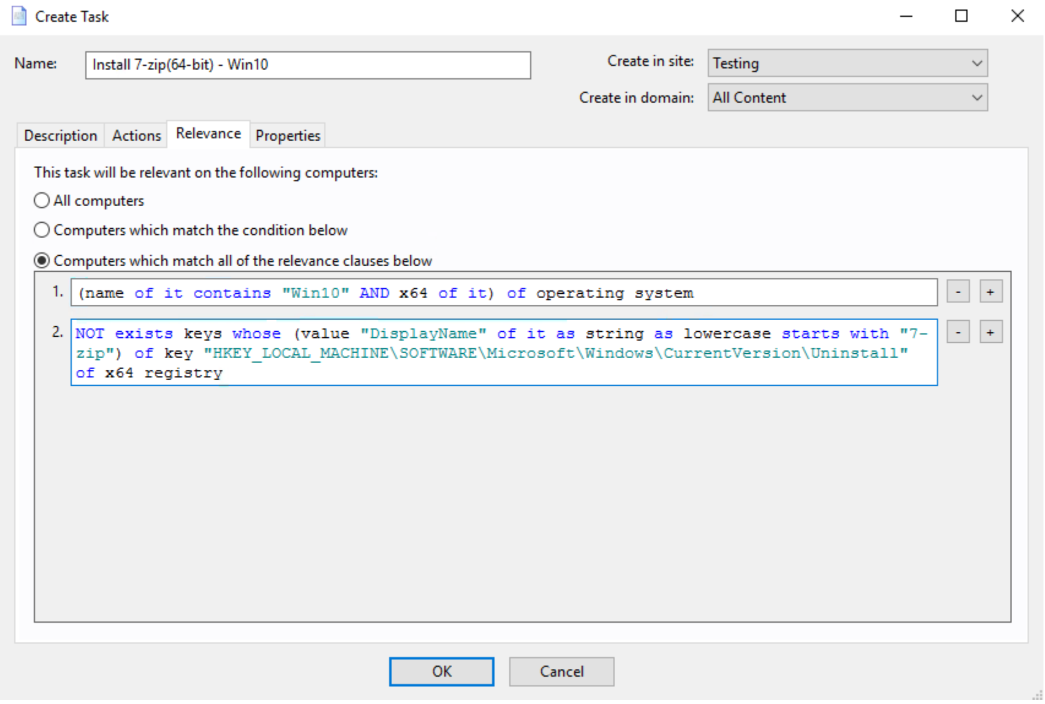
12. Click the plus (**+**) symbol to the right of the Relevance expression.

    A second Relevance expression is added with a default value of **true**.

13. Copy the Relevance expression from the **(qna)** tab of the Fixlet Debugger that you created in Step j on page 162 and paste it into the second **Relevance** text box, replacing the current **true** expression.

    ```
    NOT exists keys whose (value “DisplayName” of it as string as lowercase starts
    with “7-zip”) of key
    ```

"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall" of x64 registry



Name: Install 7-zip(64-bit) - Win10

Create in site: Testing

Create in domain: All Content

This task will be relevant on the following computers:

○ All computers

○ Computers which match the condition below

● Computers which match all of the relevance clauses below

```
1. (name of it contains "Win10" AND x64 of it) of operating system
2. NOT exists keys whose (value "DisplayName" of it as string as lowercase starts with "7-zip") of key "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall" of x64 registry
```

14. Click **OK**.

    The task is saved and the Create Task window closes.

In this section, you create the action script for the custom 7-Zip task.

15. Copy the file `from: C:\7-Zip\7z1900-x64.msi` to the `C:\Program Files (x86)\BigFix Enterprise\BES Server\wwwrootbes\Uploads` folder and to c:\tmp.

16. Return to the Fixlet Debugger on the **BESFNDWIN10** virtual machine and select the **(qna)** tab.

17. Clear any existing queries that might already exist on the **(qna)** tab.

18. To perform the download, you must use a **`prefetch`** command that includes the SHA1 value and size of the file that is used to install 7-Zip. If you have configured enhanced security, you

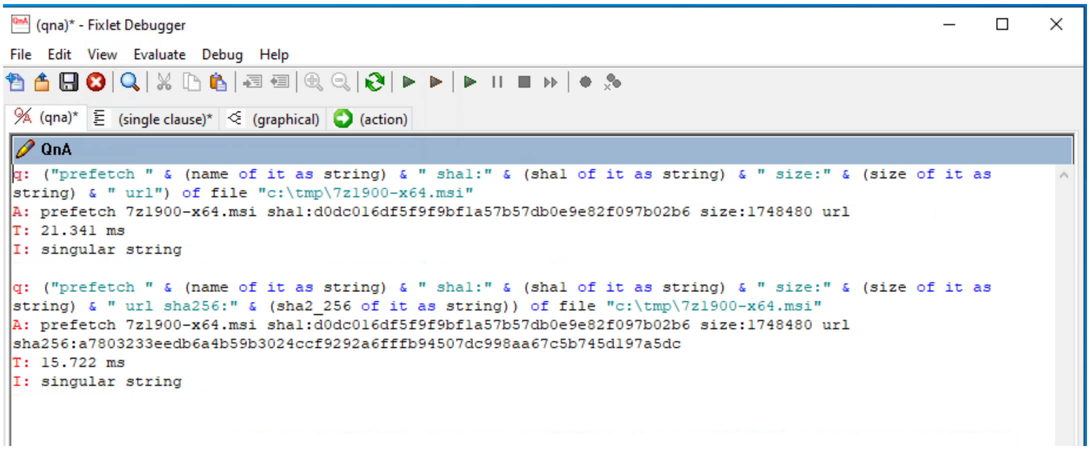
must also provide the SHA256 value of the file. In the **(qna)** tab of the Fixlet Debugger, enter the following queries and click **Evaluate**:

– The following query generates the **`prefetch`** command without the **`<SHA256:value>`** parameter:

```
Q: (“prefetch “ & (name of it as string) & “ sha1:” & (sha1 of it as string) &
“ size:” & (size of it as string) & “ url”) of file “c:\tmp\7z1900-x64.msi”
```

– The following query generates the **`prefetch`** command with the **`<SHA256:value>`** parameter:

```
Q: (“prefetch “ & (name of it as string) & “ sha1:” & (sha1 of it as string) &
“ size:” & (size of it as string) & “ url sha256:” & (sha2_256 of it as
string)) of file“c:\tmp\7z1900-x64.msi”
```



```
q: ("prefetch " & (name of it as string) & " sha1:" & (sha1 of it as string) & " size:" & (size of it as
string) & " url") of file "c:\tmp\7z1900-x64.msi"
A: prefetch 7z1900-x64.msi sha1:d0dc016df5f9f9bf1a57b57db0e9e82f097b02b6 size:1748480 url
T: 21.341 ms
I: singular string

q: ("prefetch " & (name of it as string) & " sha1:" & (sha1 of it as string) & " size:" & (size of it as
string) & " url sha256:" & (sha2_256 of it as string)) of file "c:\tmp\7z1900-x64.msi"
A: prefetch 7z1900-x64.msi sha1:d0dc016df5f9f9bf1a57b57db0e9e82f097b02b6 size:1748480 url
sha256:a7803233eedb6a4b59b3024ccf9292a6fffb94507dc998aa67c5b745d197a5dc
T: 15.722 ms
I: singular string
```

The **`prefetch`** command for the `C:\tmp\7z1900-x64.msi` file that you can use in the action script is returned by the queries.

19. Copy the results for one of the queries to the Windows clipboard. Make sure that you do not include the preceding **A:** tag.

Note: The BigFix deployment for the lab environment does not use the enhanced security configuration, so you can use either result in the action script. However, if you want to create a task that works properly in any configuration, you should use the result from the query that includes both the SHA1 and SHA256 values.

20. Return to the BigFix Console on the **BESFNDWIN10** virtual machine.

21. In the navigation pane, expand the **Custom Content** node and select the **All Custom Content** node.

    The All Custom Content pane is displayed in the upper-right portion of the console and shows a list all of the custom Fixlets and tasks.

22. In the All Custom Content pane, select the **Install 7-Zip [64-bit] – Win10** task.

    The details for the selected task are displayed in the lower-right portion of the console.

23. Click **Edit**.

    The Edit Task window opens.

24. Select the **Actions** tab.

25. Perform the following steps to create the action script code to download the installation file using the **`prefetch`** command that you created in Step 18 on page 165:

    a. In the Action Script pane, paste the expression that you copied in Step 19 on page 166, making sure that you overwrite the existing comment.

    b. Locate the url keyword in the **`prefetch`** command and replace it with the location of the `7z1900-x64.msi` file on the BigFix server as follows:

    ```
    http://BESFNDWINROOT:52311/Uploads/7z1900-x64.msi
    ```

Hint: Depending on which **`prefetch`** command you copied, the result is one of the following action script codes:

- Action script prefetch code excluding the **`<SHA256:value>`** parameter:

  ```
  prefetch 7z1900-x64.msi sha1:d0dc016df5f9f9bf1a57b57db0e9e82f097b02b6
  size:1748480 http://BESFNDWINROOT:52311/Uploads/7z1900-x64.msi
  ```

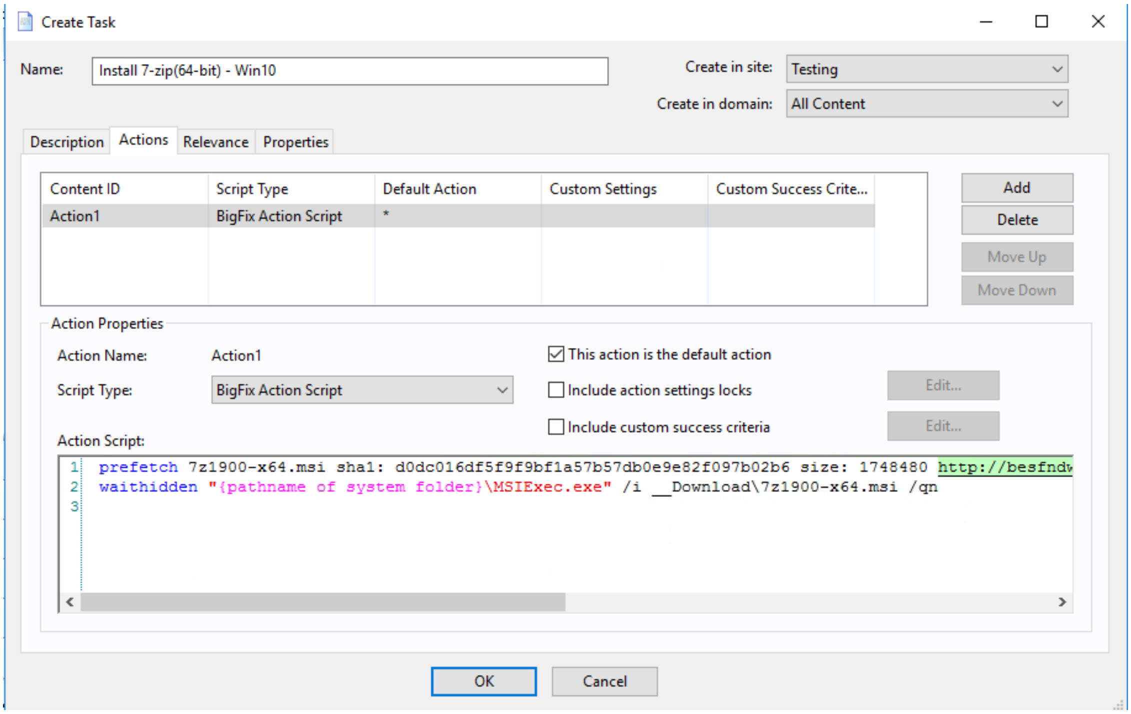
- Action script prefetch code including the **`<SHA256:value>`** parameter:

  ```
  prefetch 7z1900-x64.msi sha1:d0dc016df5f9f9bf1a57b57db0e9e82f097b02b6 size:
  1748480 http://BESFNDWINROOT:52311/Uploads/7z1900-x64.msi
  sha256:a7803233eedb6a4b59b3024ccf9292a6fffb94507dc998aa67c5b745d197a5dc
  ```

26. On a new line in the Action Script pane, you now create the command to install the 7-Zip installation file using **MSIExec.exe**. You must use the **`WaitHidden`** command to call **MSIExec.exe**, which is located in the Windows System folder. After the `7z920-x64.msi` file is downloaded from the BigFix server using the **`prefetch`** command, it is placed in the `__Download` folder on the client. You must also include the appropriate flags to inform

MSIExec.exe that the installation is performed in quite mode with no user intervention. Enter the following statement in the Action Script pane following the prefetch statement:

```
waithidden “{pathname of system folder}\MSIExec.exe”/i
__Download\7z1900-x64.msi /qn
```



Name: Install 7-zip(64-bit) - Win10

Create in site: Testing

Create in domain: All Content

Content ID	Script Type	Default Action	Custom Settings	Custom Success Crite...
Action1	BigFix Action Script	*		

Action Name: Action1

Script Type: BigFix Action Script

☑ This action is the default action

☐ Include action settings locks

☐ Include custom success criteria

Action Script:

```
prefetch 7z1900-x64.msi sha1: d0dc016df5f9f9bf1a57b57db0e9e82f097b02b6 size: 1748480 http://besfndw
waithidden "{pathname of system folder}\MSIExec.exe" /i __Download\7z1900-x64.msi /qn
```

27. Perform the following steps to add custom success criteria for that task. The custom success criteria is based on the applicability Relevance that you created to check that the registry key did not exist. You now set the custom success criteria so that the task is not Relevant if the applicability Relevance evaluates as False. This indicates that the specified registry key does exist.

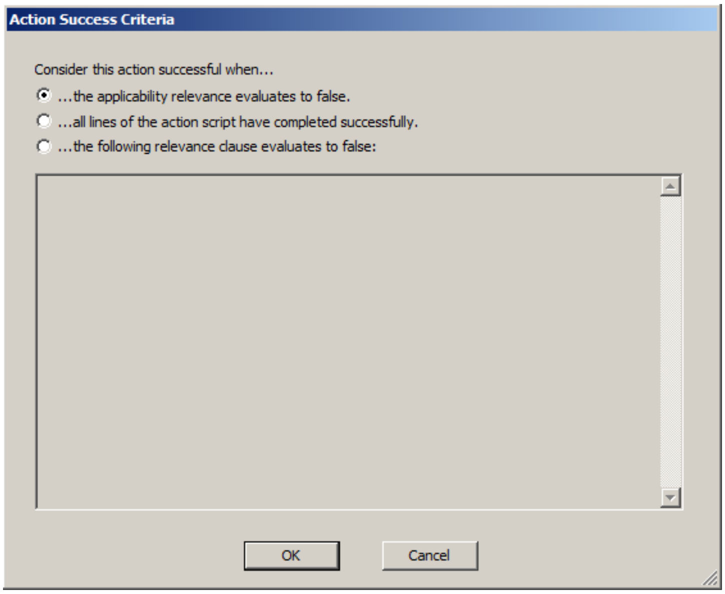
    a. In the Edit Task pane, select the **Include custom success criteria** check box.

       The **Edit** button beside the **Include custom success** criteria option becomes active.

    b. Click **Edit**.

       The Action Success Criteria window opens.

c. Select the option named **the applicability relevance evaluates to false**.



Action Success Criteria

Consider this action successful when…

● …the applicability relevance evaluates to false.

○ …all lines of the action script have completed successfully.

○ …the following relevance clause evaluates to false:

d. Click **OK**.

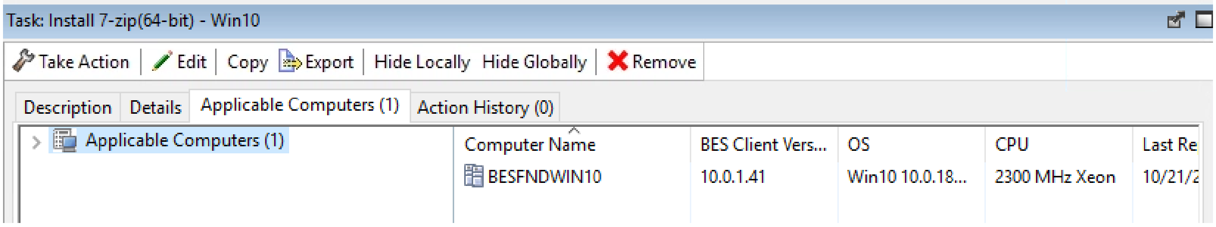
The custom success criteria is saved and the Action Success Criteria window closes.

28. On the **Edit Task** window, click **OK**.

The task is saved and the Edit Task window closes. The details for the custom task are shown in the console.

In this section, you take action on the custom task to test it.

29. Click the **Applicable Computers** tab. Wait until **BESFNDWIN10** is shown in the list of applicable computers before continuing.



Task: Install 7-zip(64-bit) - Win10

Computer Name	BES Client Vers...	OS	CPU	Last Re
BESFNDWIN10	10.0.1.41	Win10 10.0.18...	2300 MHz Xeon	10/21/2

30. Click **Take Action**.

The Take Action window opens.

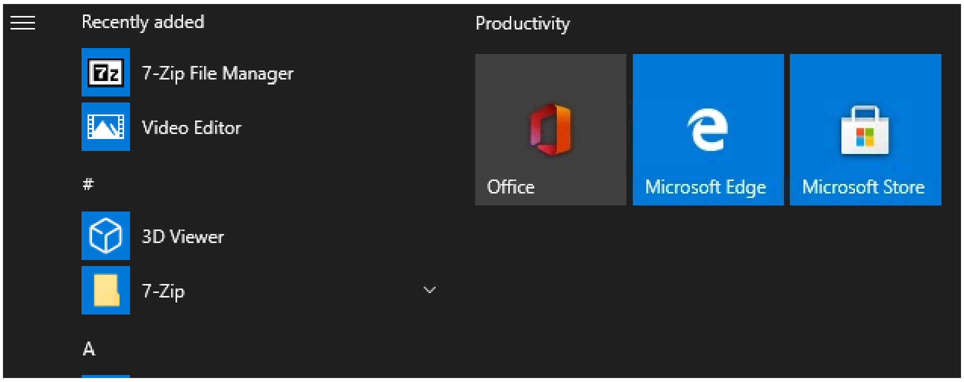
31. Click the **Target** tab if it is not already selected. Select **BESFNDWIN10** from the list of available targets. Click **OK**.

    The Take Action window closes and the Action window opens and displays the status of the running action

32. Monitor the status of the action and wait until it changes to **Completed** before continuing. This might take several minutes.

> **Note:** Observe that the Downloads section of the Action window shows that the `7z920-x64.msi` file is **Cached on Server**.

33. Click the Windows Start menu and observe that **7-Zip File Manager** is now listed.



/

Exercise 50 Uninstalling an application

In this exercise, you copy the task that you created in Exercise 12, “Downloading and installing an application,” on page 4-83 and modify it to uninstall the 7-Zip File Manager application.

1. Return to the BigFix Console on the **BESFNDWIN10** virtual machine.

2. In the navigation pane, expand the **Custom Content** node and select the **All Custom Content** node.

    The All Custom Content pane is displayed in the upper-right portion of the console and lists all of the custom Fixlets and tasks.

3. In the All Custom Content pane, select the **Install 7-Zip [64-bit] – Win10** task.

   The details for the selected task are displayed in the lower-right portion of the console.

4. .Click **Copy**.

   The Create Task window opens.

5. In the **Name** field, replace the word `Install` with `Uninstall` and keep the rest of the name the same.

6. Select the **Relevance** tab.

7. Modify the second Relevance clause so that it now evaluates as True if the specified registry key does not exist. To accomplish this, you must remove the keyword **NOT** at the beginning of the expression as follows:

   ```
   exists keys whose (value “DisplayName” of it as string as lowercase starts with
   “7-zip”) of key
   “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall” of x64
   registry
   ```

8. Copy the second Relevance expression to the Windows clipboard.

9. Select the **Actions** tab.

10. Delete the entire **`prefetch`**action command. This is no longer needed because this task is performing an uninstall of the application.

11. You now need to use the registry key for the 7-Zip application to locate the **`uninstall`** command.

12. Open the Windows **Regedit** utility if it is not already opened.

Hint: You use the **Regedit** utility to locate the value for the Uninstall key. This value is passed to the **MSIExec.exe** program in the action script to uninstall the application.

13. In the Registry Editor, locate the following registry key: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{23170F69-40C1-2702-0920-000001000000}.

Hint: To uninstall 7-Zip, you must run **MSIExec.exe** with the Uninstall key value using the following command:

```
MSIExec.exe” /x {23170F69-40C1-2702-0920-000001000000} /qn
```

The **/x** is the uninstall flag and the **/qn** flag indicates that the application is uninstalled quietly with no user intervention.

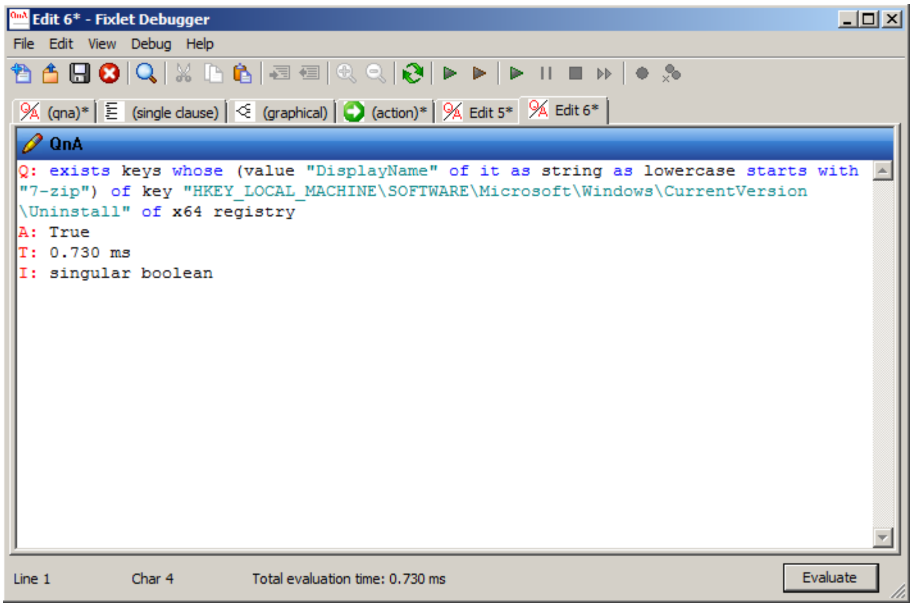
14. Switch to the Fixlet Debugger and select **File > New Tab > New QnA Tab**.

    The new tab opens and is automatically selected.

15. Paste the Relevance expression that you copied in Step 8 on page 4-95 and paste it into the new tab in the Fixlet Debugger. Be sure to include the query character (**Q:**) before the expression.

16. Click **Evaluate**.

    Because 7-Zip is installed on **BESFNDWIN10**, the expression evaluates as **True**.



```
Q: exists keys whose (value "DisplayName" of it as string as lowercase starts with
"7-zip") of key "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion
\Uninstall" of x64 registry
A: True
T: 0.730 ms
I: singular boolean
```

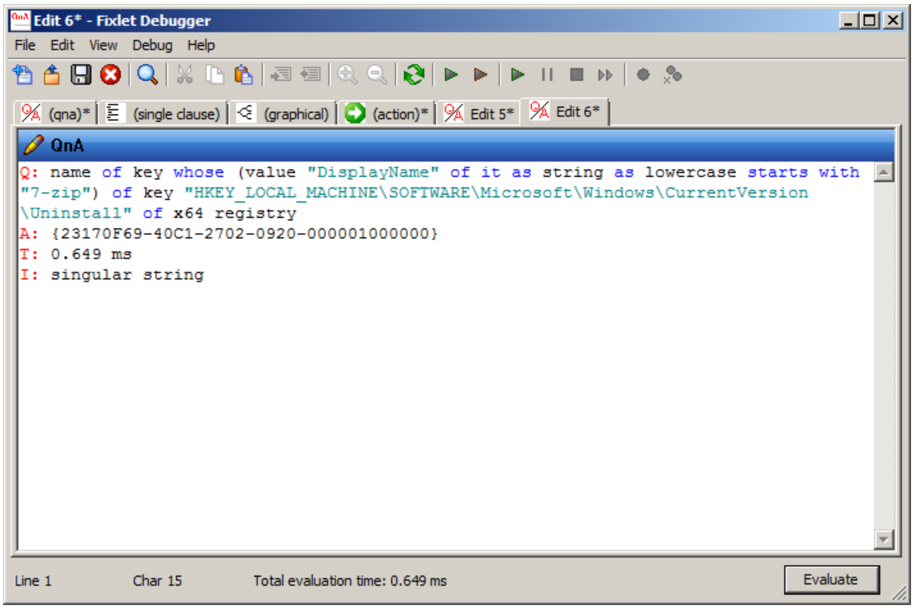
Line 1 Char 4 Total evaluation time: 0.730 ms

17. You now must modify the query to extract a key name so that you can use it in the `uninstall` command in the action script. Replace the **exists keys** phrase at the beginning of the expression with the phrase **name of key** as follows:

    Q: **name of key** whose (value “DisplayName” of it as string as lowercase starts with “7-zip”) of key “HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall” of x64 registry

18. Click **Evaluate**.

The name of the 7-Zip uninstall key is returned.



```
Q: name of key whose (value "DisplayName" of it as string as lowercase starts with
"7-zip") of key "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion
\Uninstall" of x64 registry
A: {23170F69-40C1-2702-0920-000001000000}
T: 0.649 ms
I: singular string
```

Line 1   Char 15   Total evaluation time: 0.649 ms

19. Copy the final Relevance expression to the Windows clipboard excluding the query keyword **Q:**.

20. Return to the **Actions** tab in the Create Task window of the console.

21. Delete the **i __Download\7z1900-x64.msi** portion of the remaining action script line in the Action Script pane, leaving the following incomplete command:

    ```
    waithidden "{pathname of system folder}\MSIExec.exe" / /qn
    ```

22. After the first forward slash (/), enter **x {}**. At this stage, the remaining action script line appears as follows:

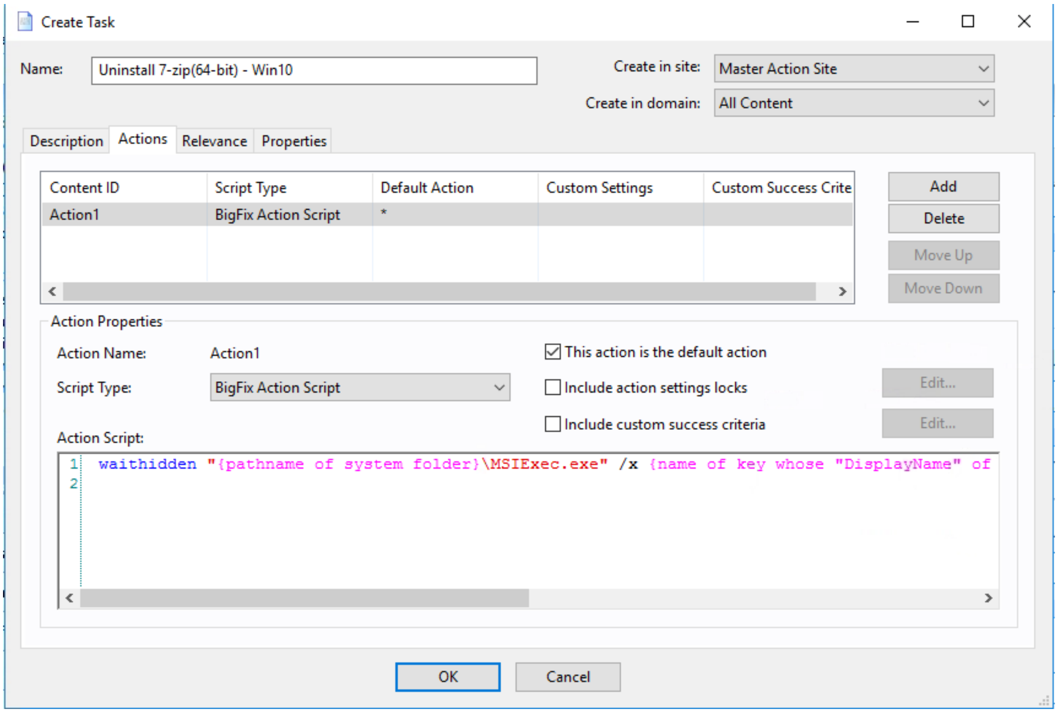
    ```
    waithidden "{pathname of system folder}\MSIExec.exe" /x {} /qn
    ```

23. Paste the Relevance expression that was copied in Step 19 on page 4-97 between the two curly braces in the remaining action script line.

Important: If the closing curly brace moves to a new line in the Action Script pane, press the backspace key to remove the line feed and display the command as a single line.

```
waithidden "{pathname of system folder}\MSIExec.exe" /x {name of key whose
(value "DisplayName" of it as string as lowercase starts with "7-zip") of key
```

```
“HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall” of x64
registry} /qn
```



Create Task

Name: Uninstall 7-zip(64-bit) - Win10

Create in site: Master Action Site

Create in domain: All Content

Content ID	Script Type	Default Action	Custom Settings	Custom Success Crite
Action1	BigFix Action Script	*		

Action Properties

Action Name: Action1

Script Type: BigFix Action Script

☑ This action is the default action

☐ Include action settings locks

☐ Include custom success criteria

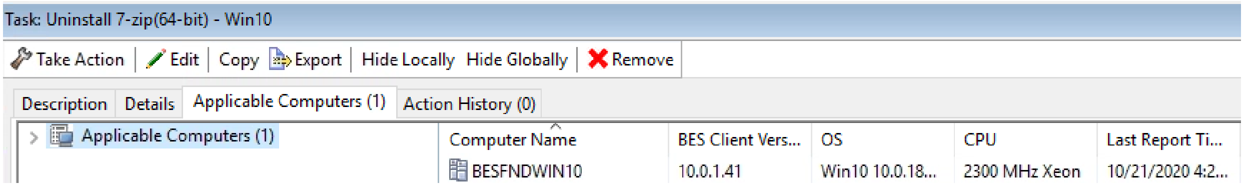
Action Script:

```
waithidden "{pathname of system folder}\MSIExec.exe" /x {name of key whose "DisplayName" of
```

The task is saved and the Create Task window closes. The details for the custom task are shown in the console.

In this section, you take action on the custom task to test it.

24. Click the **Applicable Computers** tab. Wait until **BESFNDWIN10** is shown in the list of applicable computers before continuing.



Task: Uninstall 7-zip(64-bit) - Win10

Computer Name	BES Client Vers...	OS	CPU	Last Report Ti...
BESFNDWIN10	10.0.1.41	Win10 10.0.18...	2300 MHz Xeon	10/21/2020 4:2...

25. Click **Take Action**.

    The Take Action window opens.

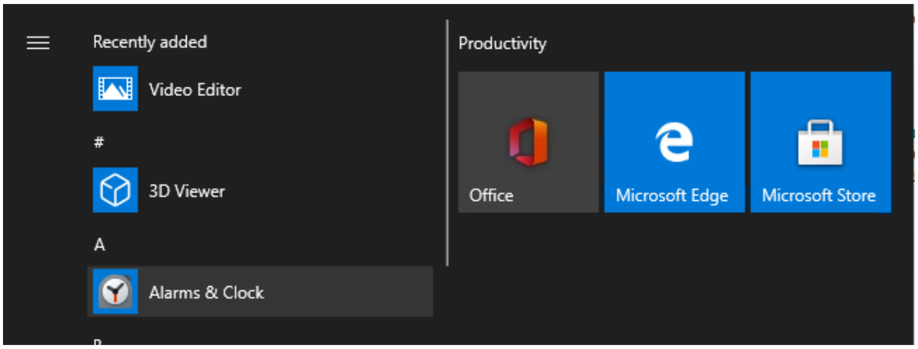
26. Click the **Target** tab if it is not already selected. Select **BESFNDWIN10** from the list of available targets.

27. Click **OK**.

The Take Action window closes and the Action window opens and displays the status of the running action

28. Monitor the status of the action and wait until it changes to **Completed** before continuing. This might take several minutes.

29. Click the Windows Start menu and observe that **7-Zip File Manager** is no longer listed.

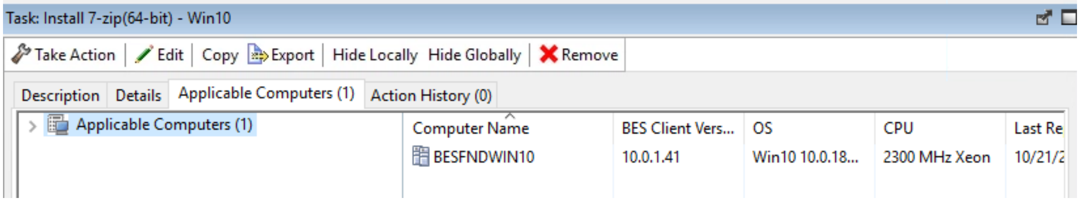


30. In the navigation pane in the BigFix Console, expand the **Custom Content** node and select the **All Custom Content** node.

The All Custom Content pane is displayed in the upper-right portion of the console and lists all of the custom Fixlets and tasks.

31. In the All Custom Content pane, select the **Install 7-Zip [64-bit] – Win10** task. The details for the selected task are displayed in the lower-right portion of the console.

32. Click the **Applicable Computers** tab and verify that this task is once again applicable to the **BESFNDWIN10** managed endpoint. It might take several minutes for the client to evaluate Relevance and report the results to the server.



Computer Name	BES Client Vers...	OS	CPU	Last Re
BESFNDWIN10	10.0.1.41	Win10 10.0.18...	2300 MHz Xeon	10/21/2

This ends the course: BESFND401R