HP Business Service Management

Software Version: 9.25

BPM Monitoring Solutions - Best Practices

Document Release Date: April 2017
Software Release Date: January 2015

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: https://softwaresupport.hp.com/group/softwaresupport/search-result?keyword=.

This site requires an HP Passport account. If you do not have one, click the **Create an account** button on the HP Passport Sign in page.

Support

Visit the HP Software Support web site at: **https://softwaresupport.hp.com**

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software Support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to **https://softwaresupport.hp.com** and click **Register**.

To find more information about access levels, go to: **https://softwaresupport.hp.com/web/softwaresupport/access-levels**

HP Software Integrations, Solutions and Best Practices

Visit the Integrations and Solutions Catalog at https://softwaresupport.hp.com/group/softwaresupport/search-result/-/facetsearch/document/KM01702710 to explore how the products in the HP Software catalog work together, exchange information, and solve business needs.

Visit the Cross Portfolio Best Practices Library at **https://hpln.hp.com/group/best-practices-hpsw** to access a wide variety of best practice documents and materials.

Contents

Chapter 1: Overview

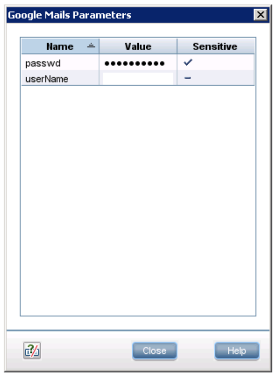
The goal of these case studies is to show how to establish end user monitoring on a cloud application using TruClient for Internet Explorer protocol. These case studies provide the basic instructions required to facilitate the process of building end user monitoring solutions using WYSIWYG protocol.

Where applicable, the case studies are built around monitoring web applications. You can use TruClient for IE or Mobile protocol, depending on your business needs.

Chapter 2: How to Use Script Parameters and Functions

Script Parameters

Generally, parameters are used when values change frequently or you want to easily change them. Instead of changing the content of the script, you can just change the parameters. Script parameters defined in VuGen are seen in End User Management Administration in BSM.

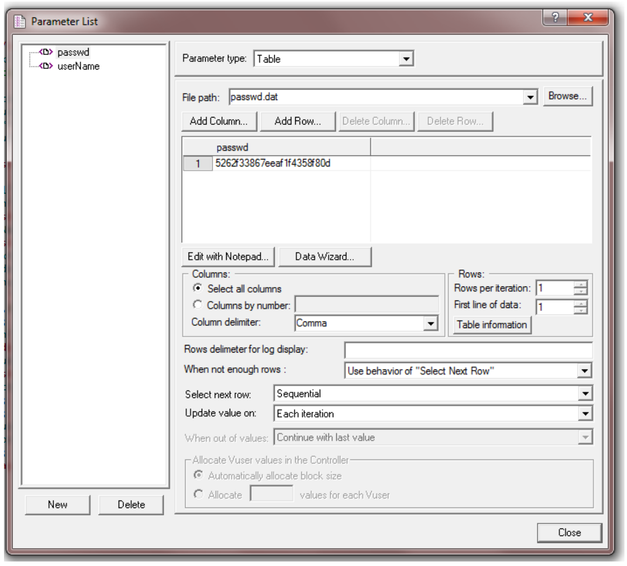


Name	Value	Sensitive
passwd	••••••••••	✓
userName		–

To achieve this, create the parameters in VuGen. Notice that the value for the passwd parameter is marked as sensitive. Let us show you how we did this.

All parameters except passwd are plain text:

- **passwd** – Hidden and encrypted password.
- **userName** – Username in the form of an email address (user@domain.com).
- **userNameSurname** – This parameter is created because in some steps we need to click the full name of the user.



Parameter type: Table

File path: passwd.dat

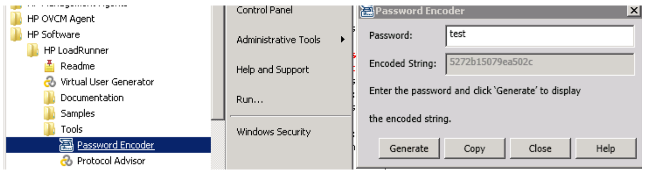
	passwd
1	5262f33867eeaf1f4358f80d

Columns: Select all columns; Columns by number:
Column delimiter: Comma

Rows: Rows per iteration: 1; First line of data: 1

Rows delimeter for log display:
When not enough rows : Use behavior of "Select Next Row"
Select next row: Sequential
Update value on: Each iteration
When out of values: Continue with last value
Allocate Vuser values in the Controller: Automatically allocate block size; Allocate values for each Vuser

To encrypt the password, we first need to use the Password Encoder:



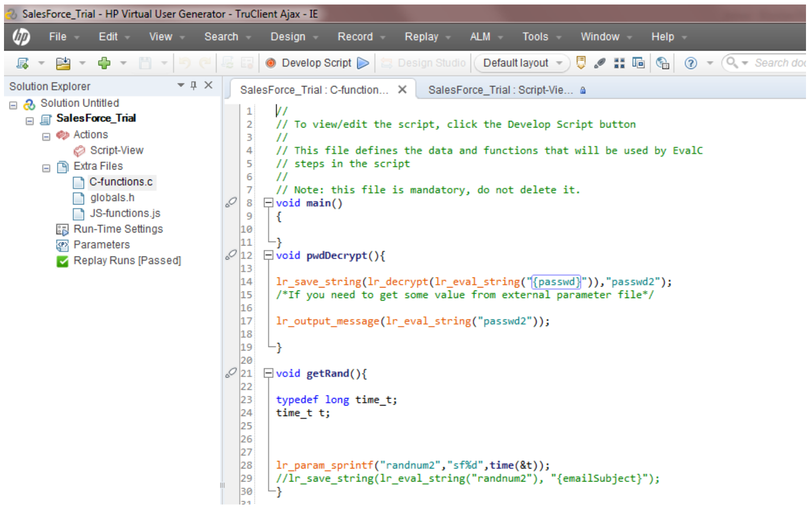
Password: test
Encoded String: 5272b15079ea502c
Enter the password and click `Generate' to display the encoded string.

The encoded string is then copied to the passwd value.

Script Functions

Functions are used to enhance a script. We use functions to achieve the following goals:

- Decrypt the parameter value and put it in a new parameter that is visible only within the script instance running on VuGen or the BPM instance.
- Generate a random number that is unique. In this example, we use Time as the random number.



```
//
// To view/edit the script, click the Develop Script button
//
// This file defines the data and functions that will be used by EvalC
// steps in the script
//
// Note: this file is mandatory, do not delete it.
void main()
{

}
void pwdDecrypt(){

lr_save_string(lr_decrypt(lr_eval_string("{passwd}")),"passwd2");
/*If you need to get some value from external parameter file*/

lr_output_message(lr_eval_string("passwd2"));

}

void getRand(){

typedef long time_t;
time_t t;



lr_param_sprintf("randnum2","sf%d",time(&t));
//lr_save_string(lr_eval_string("randnum2"), "{emailSubject}");
}
```

To decrypt this code, we use a C function within the script. Functions are stored in the C function.c file.

```
void pwdDecrypt(){

lr_save_string(lr_decrypt(lr_eval_string("{passwd}")),"passwd2");

}
```

The function `lr_eval_string` converts the passwd parameter to a string. Then `lr_decrypt` decrypts the string and `lr_save_string` copies the string as plain text to passwd2. This plain text password is contained within the boundaries of the script instance that is executed on the BPM instance.

Another function used is getRand(). It generates a unique Subject text that we can locate later and delete:

```
void getRand(){

typedef long time_t;

time_t t;

lr_param_sprintf("randnum2","bpm%d",time(&t));

}
```

These functions are used within the script.

Chapter 3: Cloud Email Provider – Office 365 Transaction Flow

This case study uses Microsoft Office 365 cloud service. The case study uses one flow/script with the following transactions:

1. **Login** – Open a login page, enter your credentials, and confirm that the first page loads.

2. **Create an email** – Create a new email and send it with a unique subject to the logged in user.

3. **Delete the email** – Check for this unique email, select it, and delete it.

4. **Calendar** – Click the Calendar link and verify that the calendar loads.

5. **Logoff** – Log off from the web application.

The main purpose of running this script is to confirm that the application's processes run successfully. If successful, the transactions complete without any errors.

You can view a sample Office365 script by downloading the following file https://softwaresupport.hpe.com/group/softwaresupport/search-result/-/facetsearch/document/KM00658290.

General Flow

Microsoft Office 365 is a subscription-based online office and software plus services suite which offers access to various services and software built around the Microsoft Office platform. In this flow, we test Microsoft Office 365's email and calendar functionality.

A sequence of transactions creates the business process flow. A transaction is a unit that is measurable by availability and performance. Therefore, we group activities that perform a specific transaction.

To describe and validate these transactions, logically group the transactions into the following steps:

1. Login.

   a. Navigate to https://portal.microsoftonline.com.

   b. Enter your credentials and confirm that the first page (Inbox) loads.

2. Create a new email.

   a. Click **New Mail**.

   b. Enter the email address used to login in step 1.

   c. Enter a unique subject.

   d. Send the email.

3. Delete the email.

   a. Refresh and check if the email arrived.

   b. Select the email.

   c. Delete the email.

4. Calendar

   a. Click the **Calendar** button and confirm that the Calendar page loads.

5. Logoff

   a. Log off from the web application.

The following sections describe how to create these transactions.

A good approach, before using this tool, is to know what we want to achieve. Create a blueprint as follows:

1. Run the Office 365 web application in Internet Explorer 9. Perform the steps of the flow and create annotations.

2. Determine the boundaries that form the transactions.

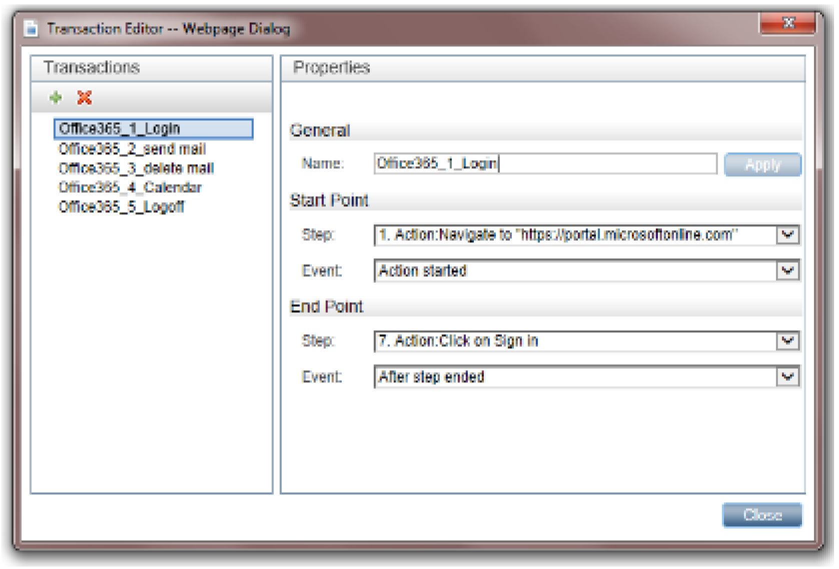
3. Select which values to assign to the parameters.

4. Select what values to protect and encrypt.

Record all the information you collect since it will come in handy later.

We recommend you start using the script. Refer to sections in this document for assistance.

Transaction — Office365_1_Login



Office365_1_Login
Office365_2_send mail
Office365_3_delete mail
Office365_4_Calendar
Office365_5_Logoff

General
Name: Office365_1_Login

Start Point
Step: 1. Action:Navigate to "https://portal.microsoftonline.com"
Event: Action started

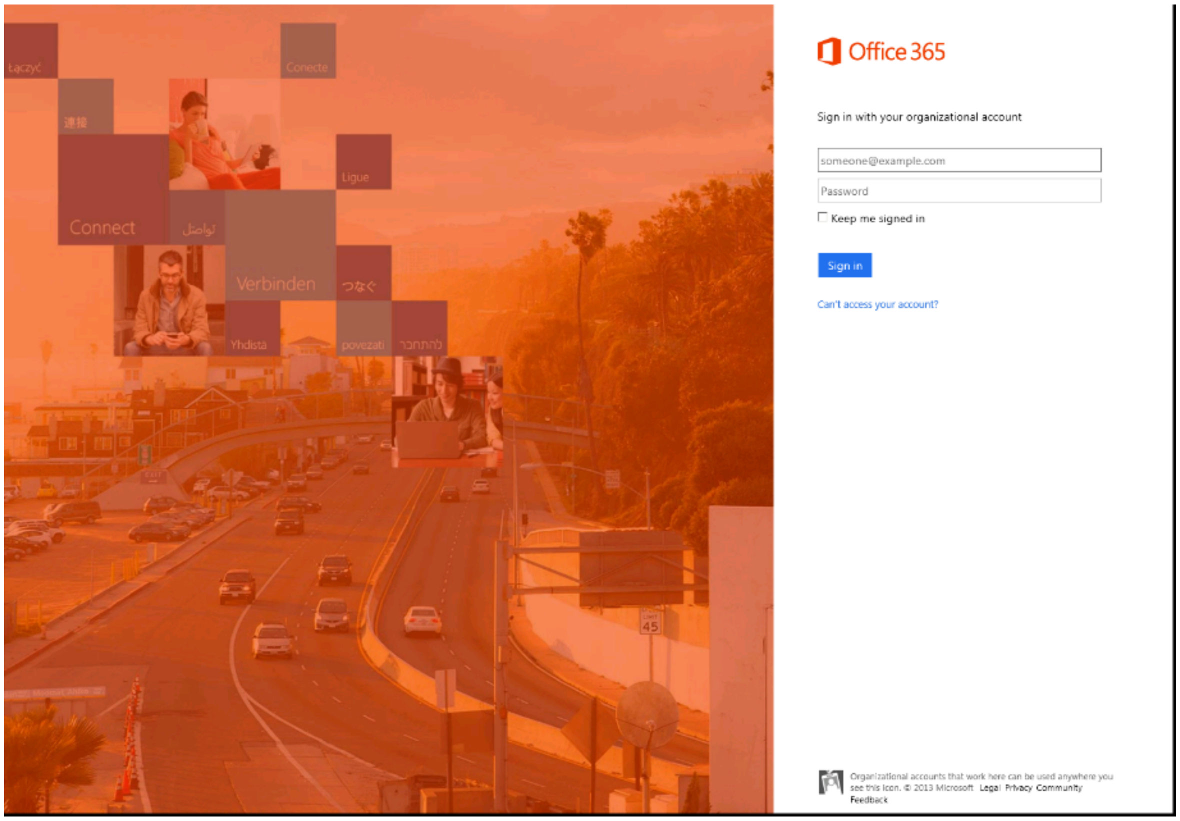
End Point
Step: 7. Action:Click on Sign in
Event: After step ended

Define the name of the transaction. In this example, we call this transaction “Office365_1_Login”. Naming conventions should provide a clear view of the application, the transaction order, and create an overview of the business processes covered by this transaction.

Define the steps and events for the starting and ending points of the transaction. The most appropriate event for the start step is Action started. The most appropriate event for the end step is After step ended.

This transaction contains all the steps required for logging into the Office 365 application.

Navigate to https://portal.microsoftonline.com



Office 365

Sign in with your organizational account

someone@example.com

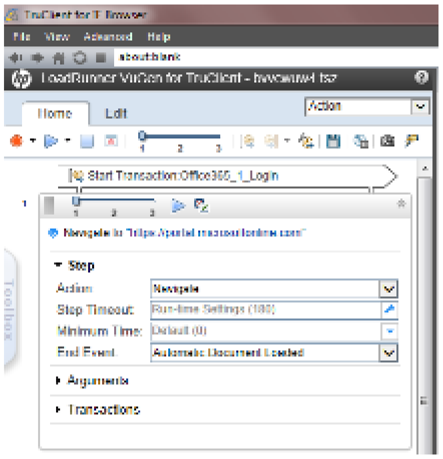
Password

Keep me signed in

Sign in

Can't access your account?

Organizational accounts that work here can be used anywhere you see this icon. © 2013 Microsoft Legal Privacy Community Feedback



Start Transaction:Office365_1_Login

1 Navigate to "https://portal.microsoftonline.com"

Step

Action	Navigate
Step Timeout:	Run-time Settings (180)
Minimum Time:	Default (0)
End Event:	Automatic Document Loaded

Arguments

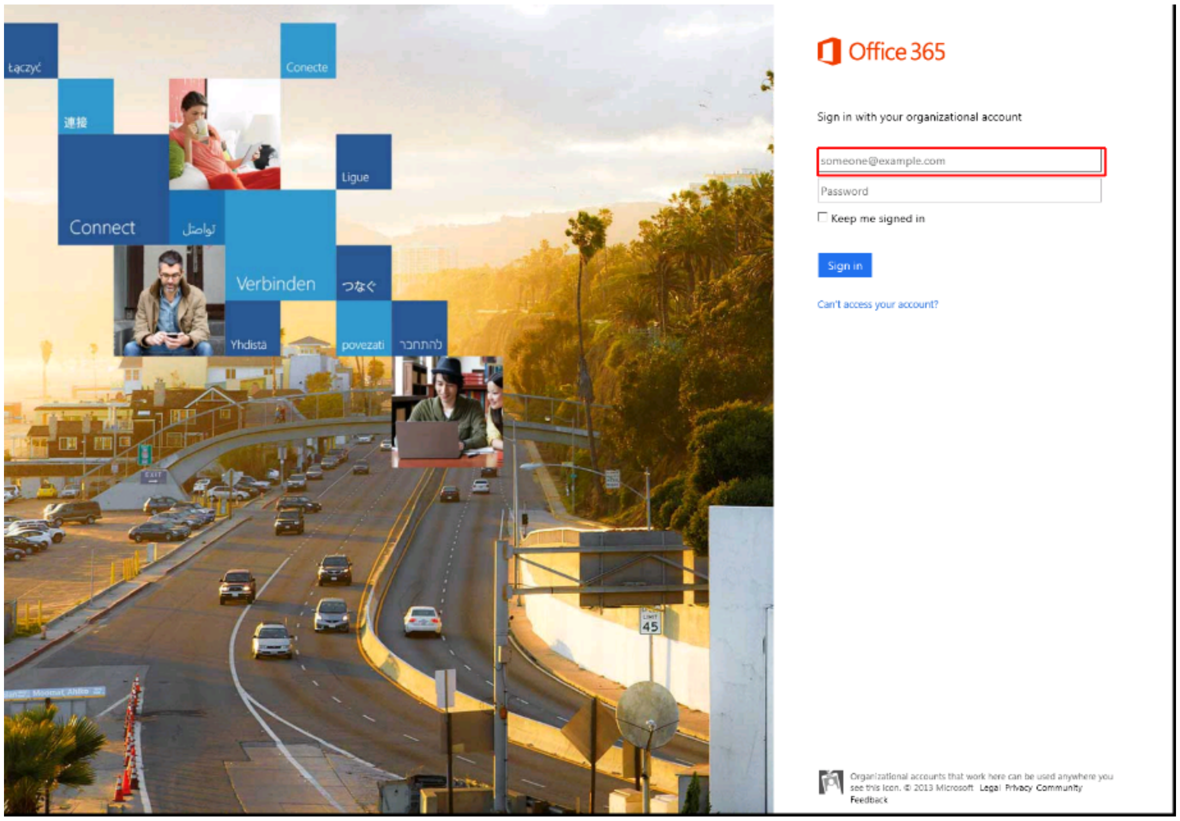
Transactions

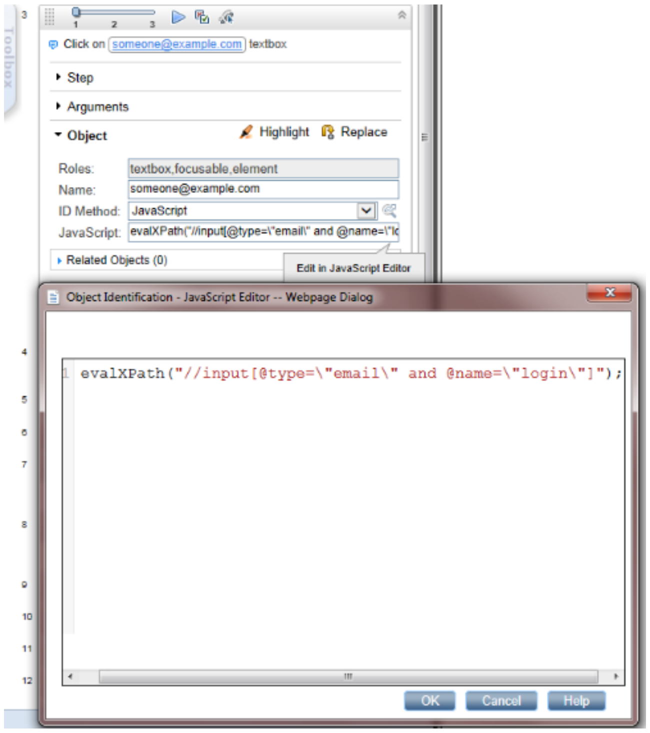
In the Navigation step, specify the URL.

While recording, VuGen automatically selects the appropriate End Event, in this case Document downloaded. The step ends when the process of loading a document completes. In most cases, you can freely select the default values.

If the pages load slowly, you can use a Wait step to make sure the page loads. For information on Wait steps, see Meaning of Wait Step on page 94.

Click in the someone@example.com Textbox


łączyć
Conecte
連接
Ligue
Connect
لتواصل
Verbinden
つなぐ
Yhdistä
povezati
להתחבר
Office 365
Sign in with your organizational account
someone@example.com
Password
Keep me signed in
Sign in
Can't access your account?
Organizational accounts that work here can be used anywhere you see this icon. © 2013 Microsoft Legal Privacy Community Feedback


Toolbox
Click on someone@example.com textbox
Step
Arguments
Object
Highlight
Replace
Related Objects (0)
Edit in JavaScript Editor
Object Identification - JavaScript Editor -- Webpage Dialog
OK
Cancel
Help

Roles: textbox,focusable,element
Name: someone@example.com
ID Method: JavaScript
JavaScript: evalXPath("//input[@type=\"email\" and @name=\"lo

```
evalXPath("//input[@type=\"email\" and @name=\"login\"]");
```

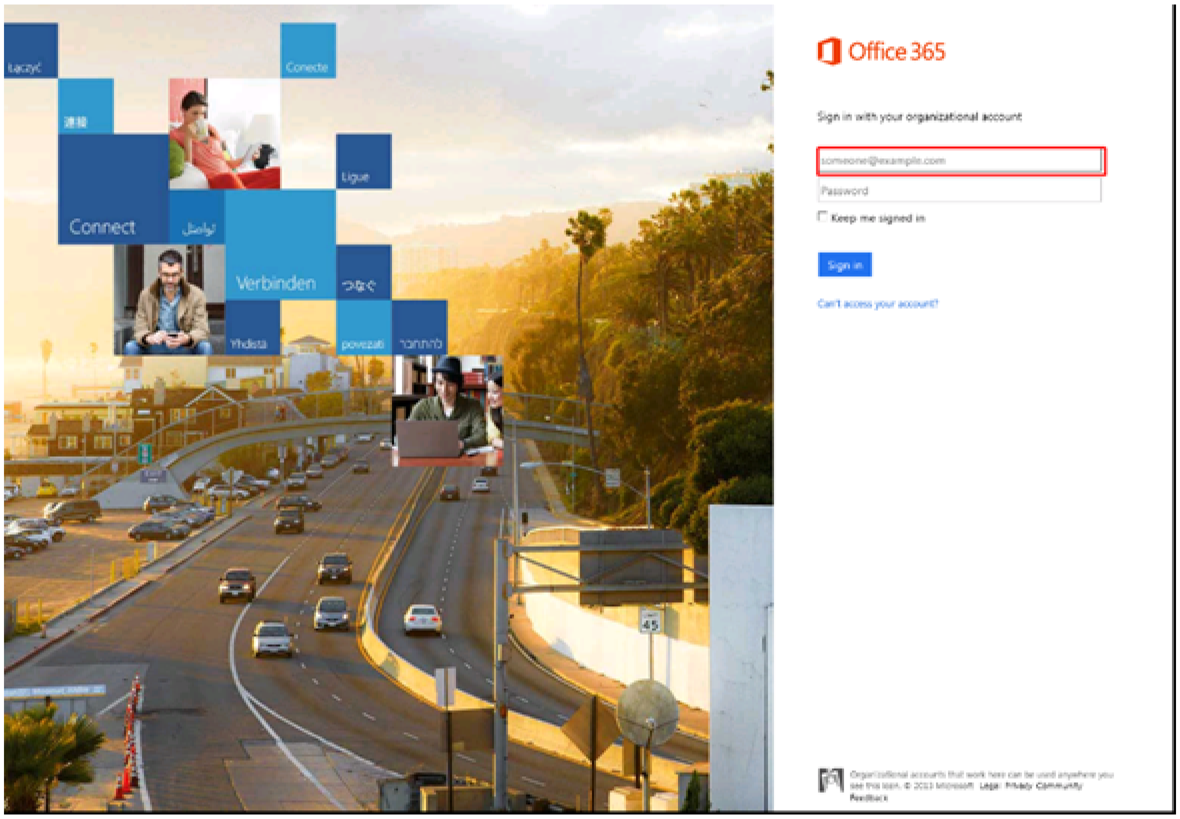
You can either select the **Automatic** or **JavaScript** ID method.

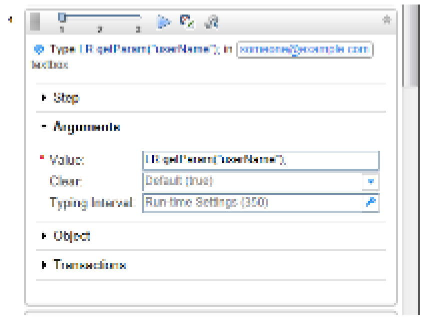
For the JavaScript ID Method, enter the following in the JavaScript field:

```
evalXPath("//input[@type=\"email\" and @name=\"login\"]");
```

There are cases when you will change the ID Method, as we will see later.

Type LR.getParam('userName') in someone@example.com Textbox





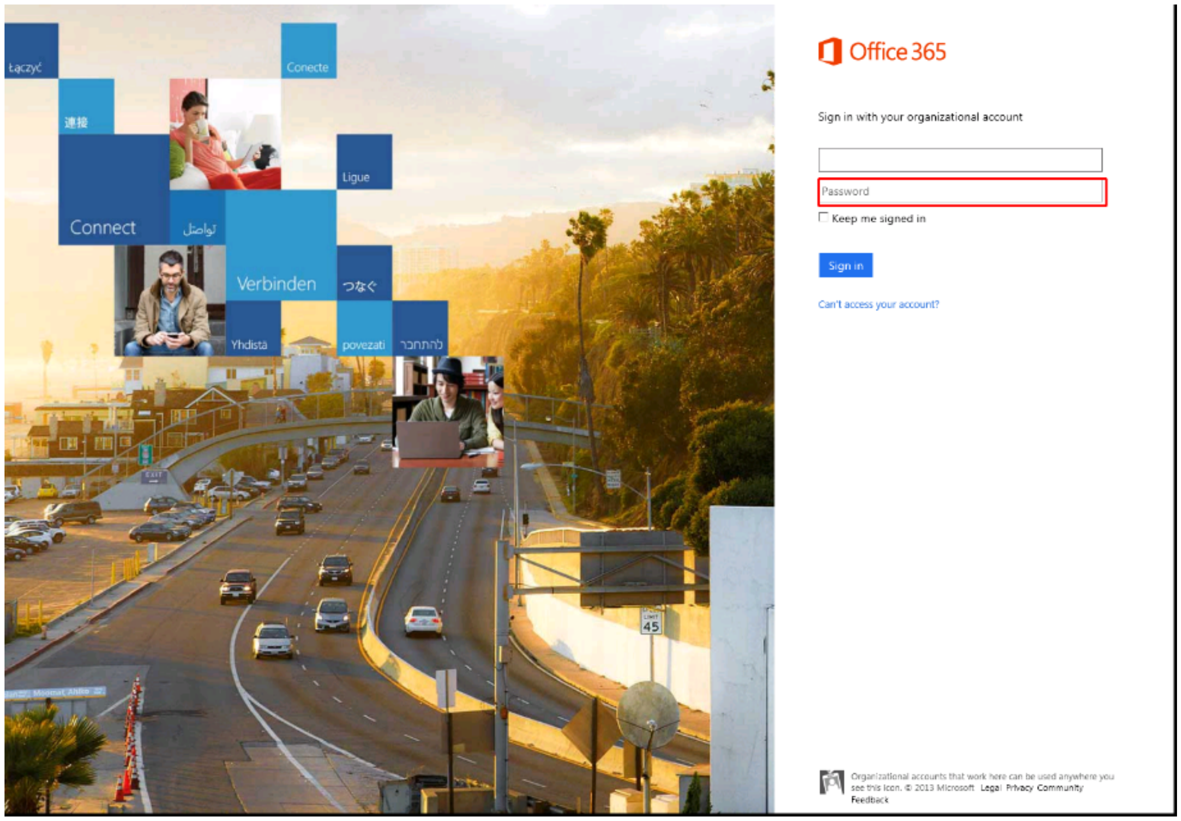
Type LR.getParam('userName'); in someone@example.com textbox

- Step
- Arguments
  - Value: LR.getParam('userName');
  - Clear: Default (true)
  - Typing Interval: Run-time Settings (350)
- Object
- Transactions

Type the username in the **someone@example.com** textbox.

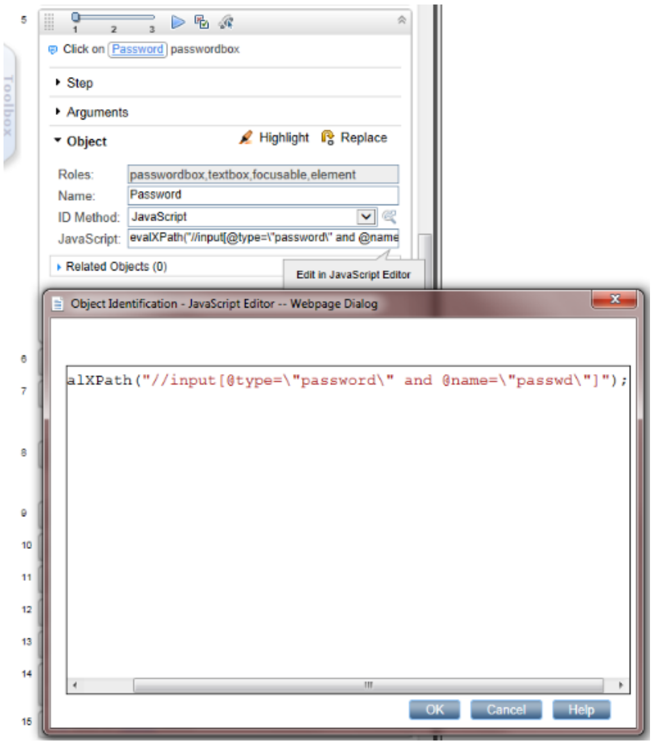
Since we are using parameters, you can enter the *LR.getParam('userName');* command in the **Value** field and the result is the defined username.

Click in the Password Textbox



Sign in with your organizational account

Organizational accounts that work here can be used anywhere you see this icon. © 2013 Microsoft Legal Privacy Community Feedback


Toolbox
Click on Password passwordbox
Step
Arguments
Object
Highlight
Replace
Related Objects (0)
Edit in JavaScript Editor
Object Identification - JavaScript Editor -- Webpage Dialog
OK
Cancel
Help

Roles: passwordbox,textbox,focusable,element
Name: Password
ID Method: JavaScript
JavaScript: evalXPath("//input[@type=\"password\" and @name

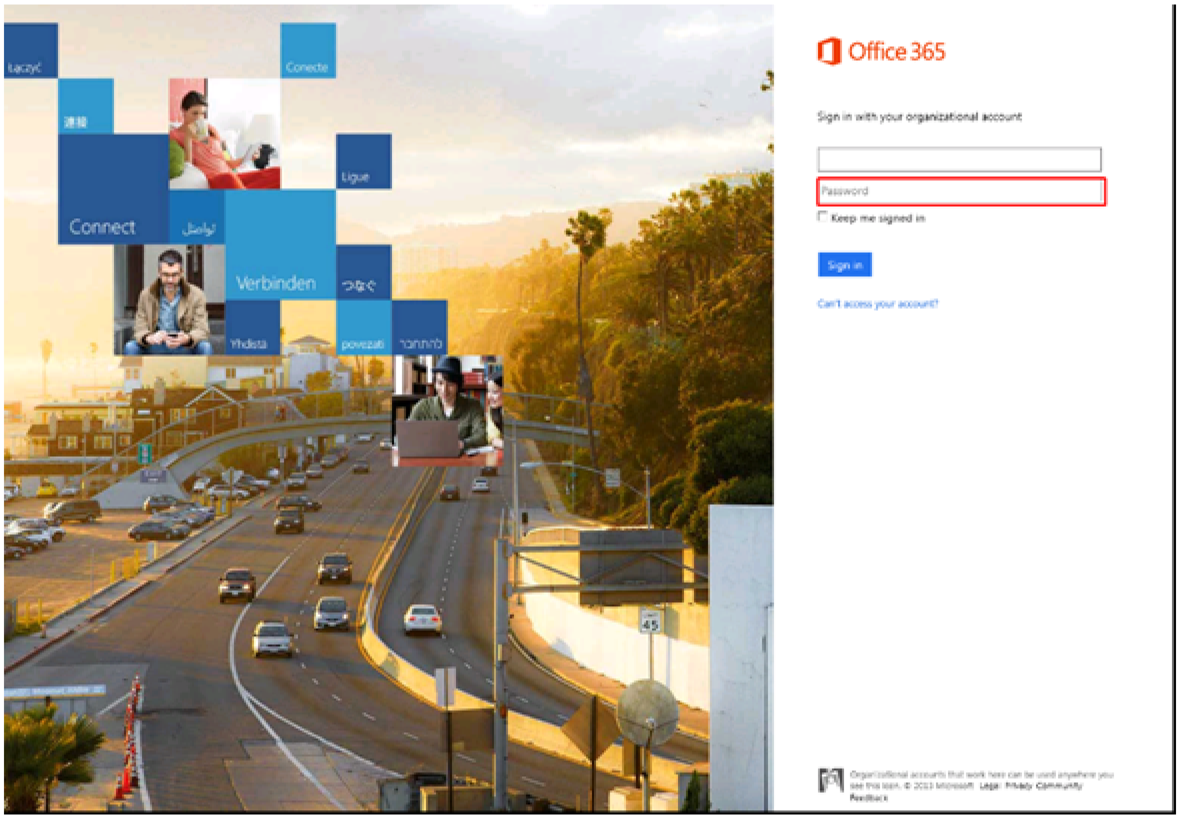
```
alXPath("//input[@type=\"password\" and @name=\"passwd\"]");
```

Similar to the *Click in the someone@example.com Textbox* step, you can select the **Automatic** or **JavaScript** ID method.

For the JavaScript ID Method, enter the following in the JavaScript field:

```
evalXPath("//input[@type=\"password\" and @name=\"passwd\"]");
```

Type **************** in the Password Textbox



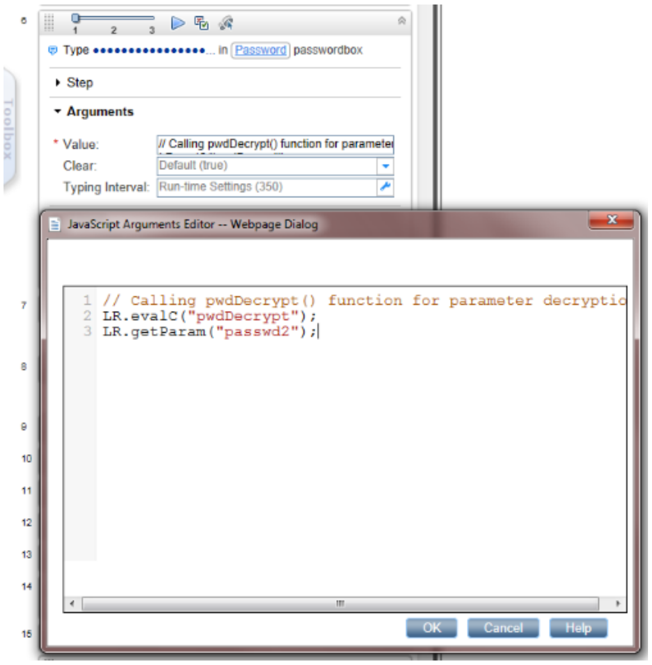
Office 365

Sign in with your organizational account

Password

Keep me signed in

Organizational accounts that work here can be used anywhere you see this icon. © 2013 Microsoft Legal Privacy Community Feedback



Type ••••••••••••••••... in Password passwordbox

* Value: // Calling pwdDecrypt() function for parameter

Clear: Default (true)

Typing Interval: Run-time Settings (350)

JavaScript Arguments Editor -- Webpage Dialog

```
// Calling pwdDecrypt() function for parameter decryptio
LR.evalC("pwdDecrypt");
LR.getParam("passwd2");
```

Type a password in the **Password** textbox. To display an encrypted password, type the following:

```
// Calling pwdDecrypt() function for parameter decryption
LR.evalC("pwdDecrypt");
// This is a parameter with a decrypted value
LR.getParam("passwd2");
```

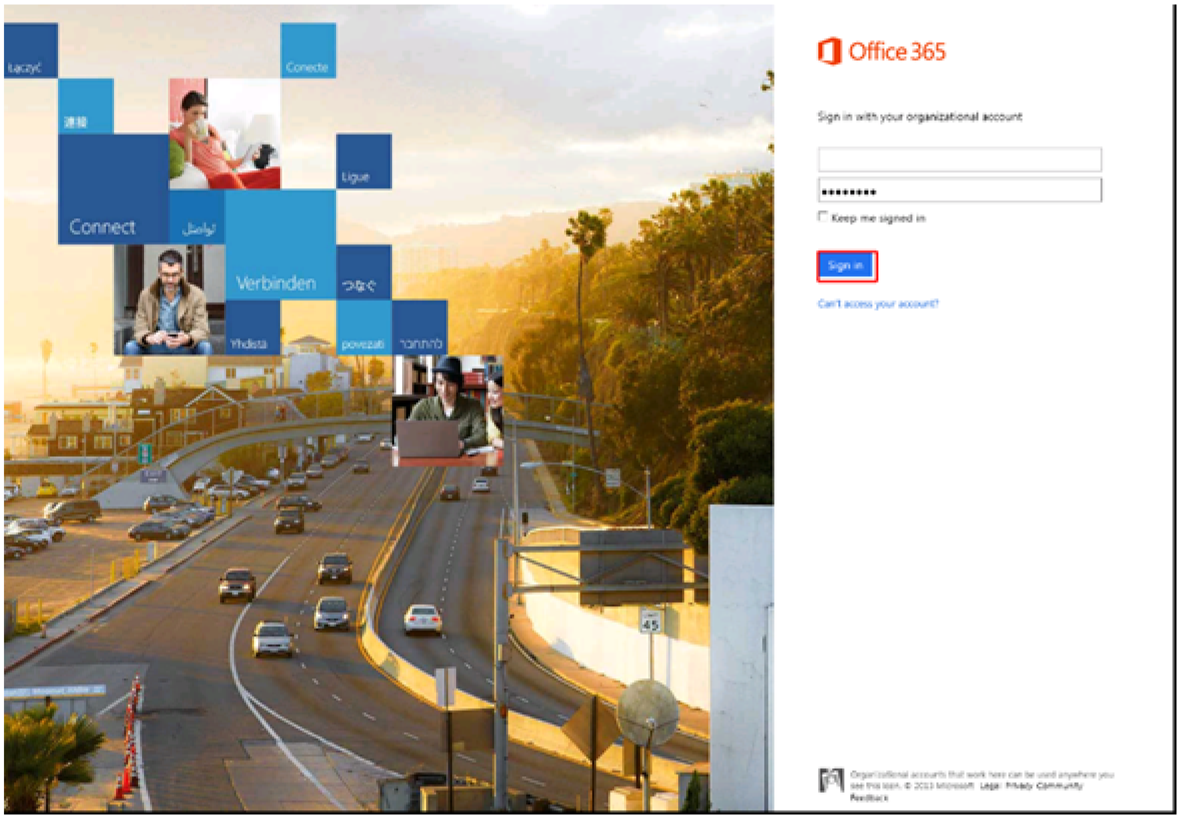
The `pwdDecrypt()` function is defined in C-function.c file:

```
void pwdDecrypt(){
lr_save_string(lr_decrypt(lr_eval_string("{passwd}")),"passwd2");
/*If you need to get some value from external parameter file*/
lr_output_message(lr_eval_string("passwd2"));
}
```

`passwd` is the parameter defined for external use (BSM – EUM) and is encrypted.

`passwd2` is the parameter seen only within this script instance.

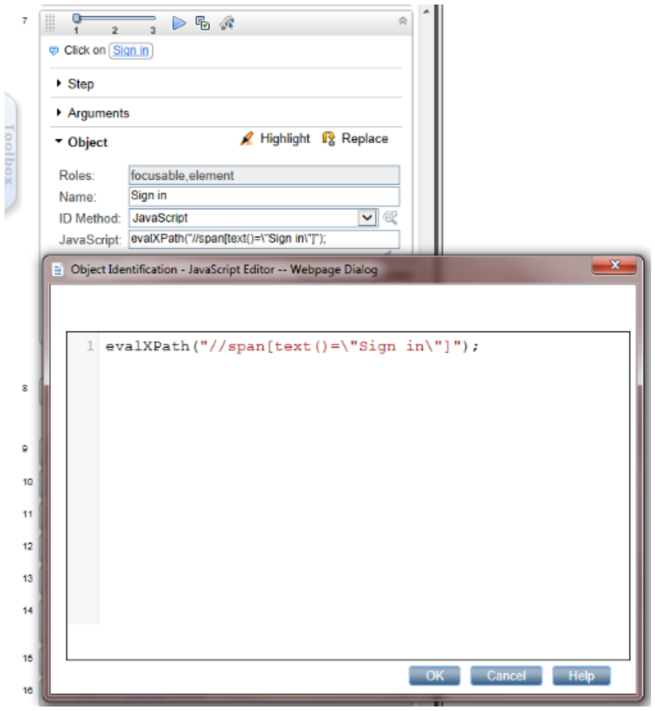
Click Sign In



Sign in with your organizational account

••••••••

Organizational accounts that work here can be used anywhere you see this icon. © 2013 Microsoft Legal Privacy Community Feedback



Roles: focusable,element
Name: Sign in
ID Method: JavaScript
JavaScript: evalXPath("//span[text()=\"Sign in\"]");

```
evalXPath("//span[text()=\"Sign in\"]");
```

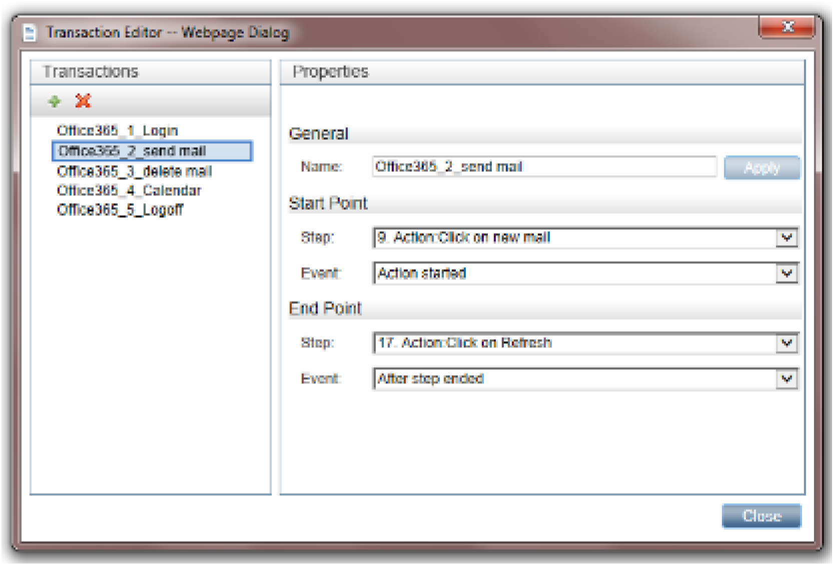
This is defined as the End step for Transaction Office365_1_Login.

The JavaScript ID Method used is:

```
evalXPath("//span[text()=\"Sign in\"]");
```

The **End Event Step** is set to automatic. So in this case, the **End Event Step** is **Step Synchronous network completed**. The step ends when all HTTP requests have completed excluding requests initiated by XMLHttpRequest.

Transaction — Office365_2_send mail



Office365_1_Login
Office365_2_send mail
Office365_3_delete mail
Office365_4_Calendar
Office365_5_Logoff

General

Name: Office365_2_send mail

Start Point

Step: 9. Action:Click on new mail

Event: Action started

End Point

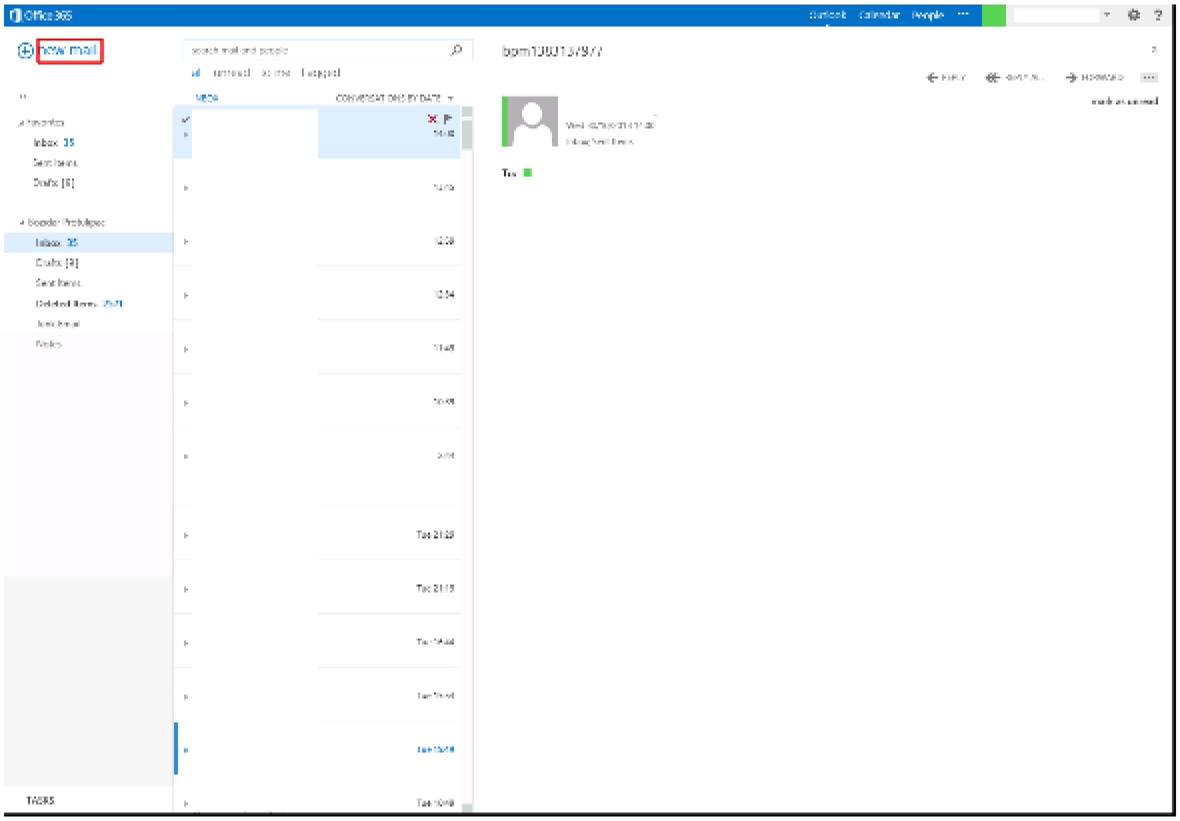
Step: 17. Action:Click on Refresh

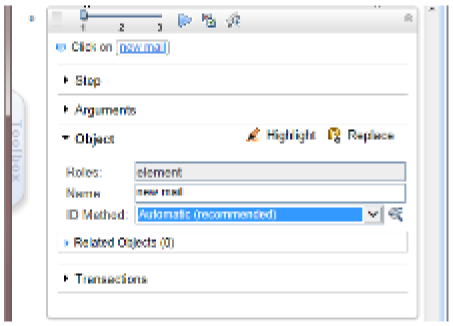
Event: After step ended

The name of this transaction is “Office365_2_send mail”.

This transaction contains all the steps necessary for sending an email to the logged in user with a unique subject field.

Click New Mail





Click on [new mail]

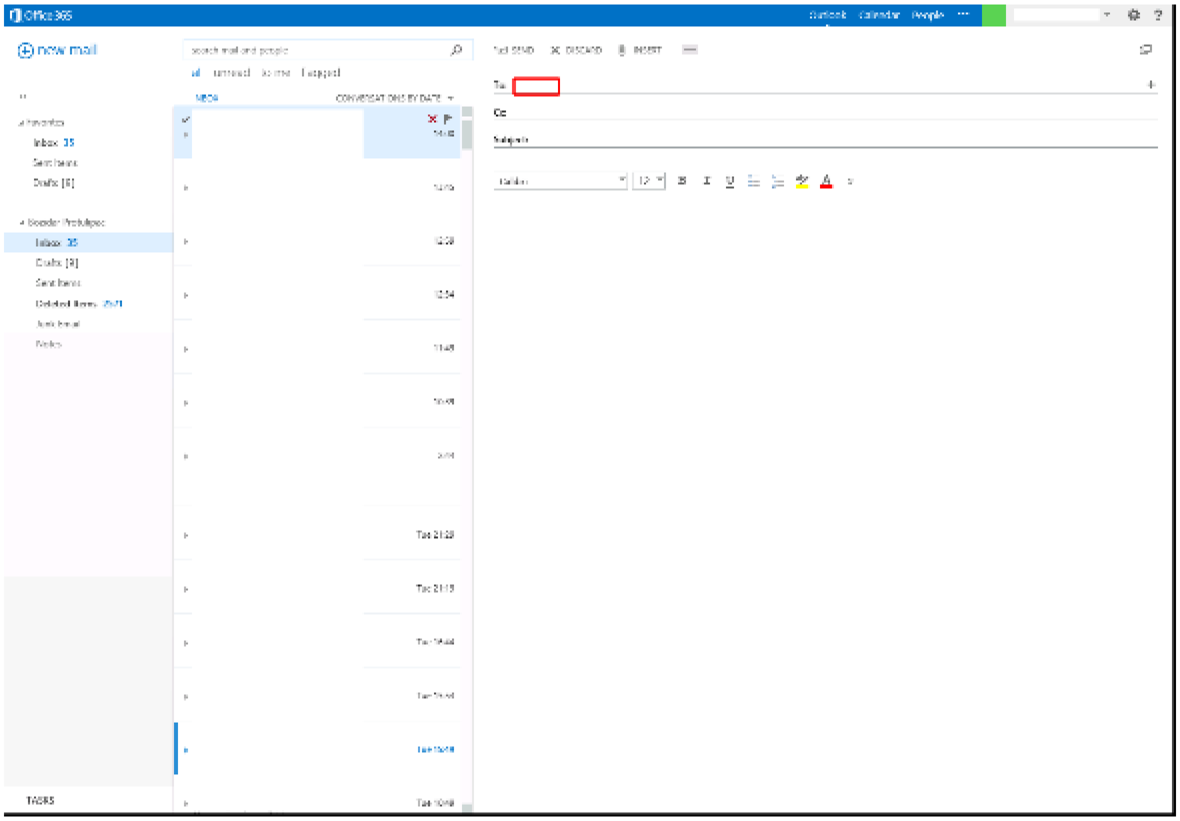
- Step
- Arguments
- Object — Highlight — Replace

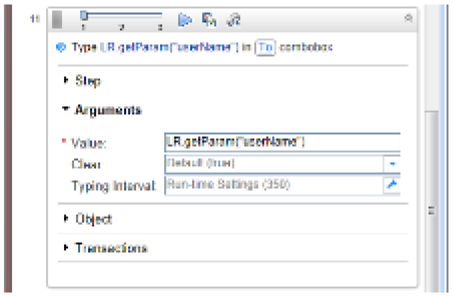
Roles:	element
Name:	new mail
ID Method:	Automatic (recommended)

- Related Objects (0)
- Transactions

When you click **New Mail**, a form appears for sending a new email. By running the script multiple times, we can confirm that this element is recognized using the Automatic ID method.

Type LR.getParam('userName') in To Field





Type LR.getParam("userName") in To combobox

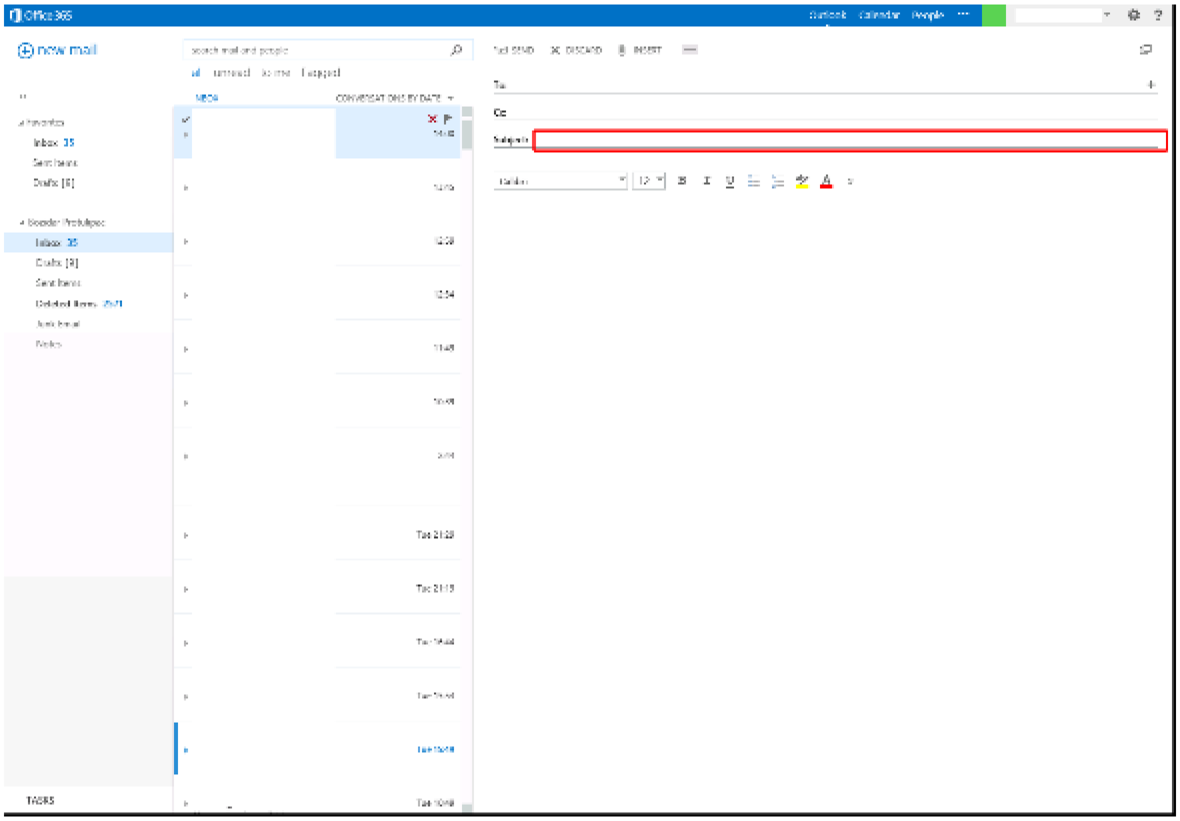
- Step
- Arguments

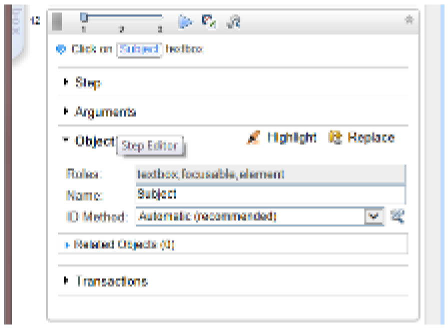
* Value:	LR.getParam("userName")
Clear:	Default (true)
Typing Interval:	Run-time Settings (350)

- Object
- Transactions

To make this script robust, we use parameters. If the username or password is changed at a later time, you do not need to change the script, only the parameter's value. The username is in the form of an email address.

Click in Subject Field




Click on Subject textbox

Step

Arguments

Object Step Editor Highlight Replace

Rules: textbox,focusable,element

Name: Subject

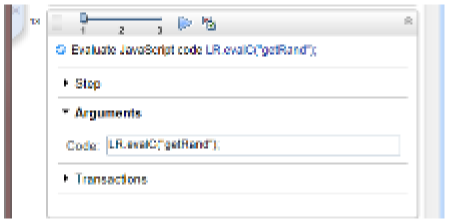
ID Method: Automatic (recommended)

Related Objects (0)

Transactions

Click in the **Subject** field to select it.

Evaluate JavaScript Code LR.evalC('getRand');


Evaluate JavaScript code LR.evalC("getRand");

Step

Arguments

Code: LR.evalC("getRand");

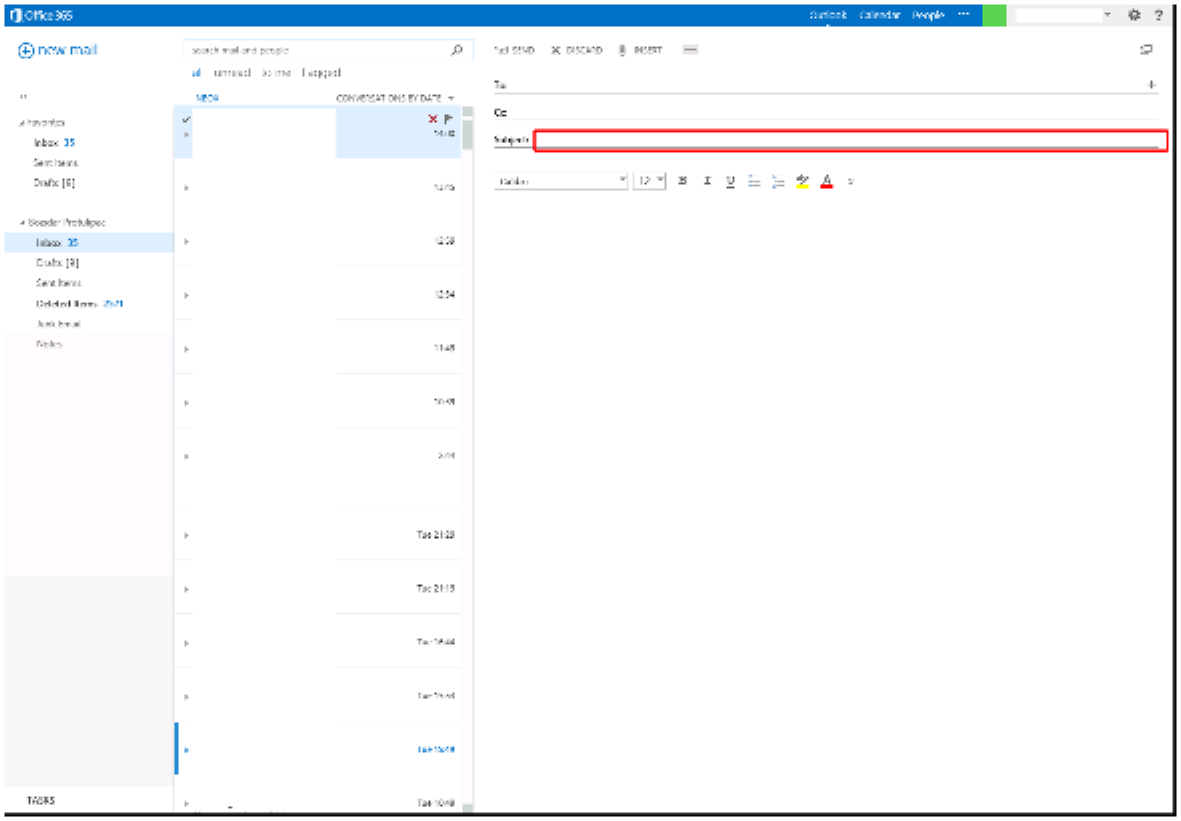
Transactions

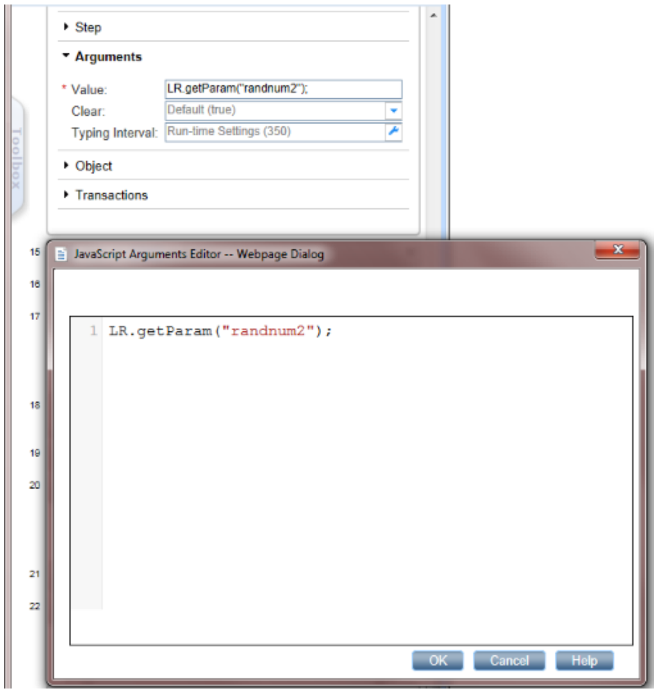
This is a very important step. By calling this function, we create a Subject value.

The `getRand()` function is defined in the C-function.c file:

```
void getRand(){

typedef long time_t;

time_t t;

lr_param_sprintf("randnum2","bpm%d",time(&t));

}
```

Type //LR.evalC('getRand'); L...am('randnum2'); in Subject Field





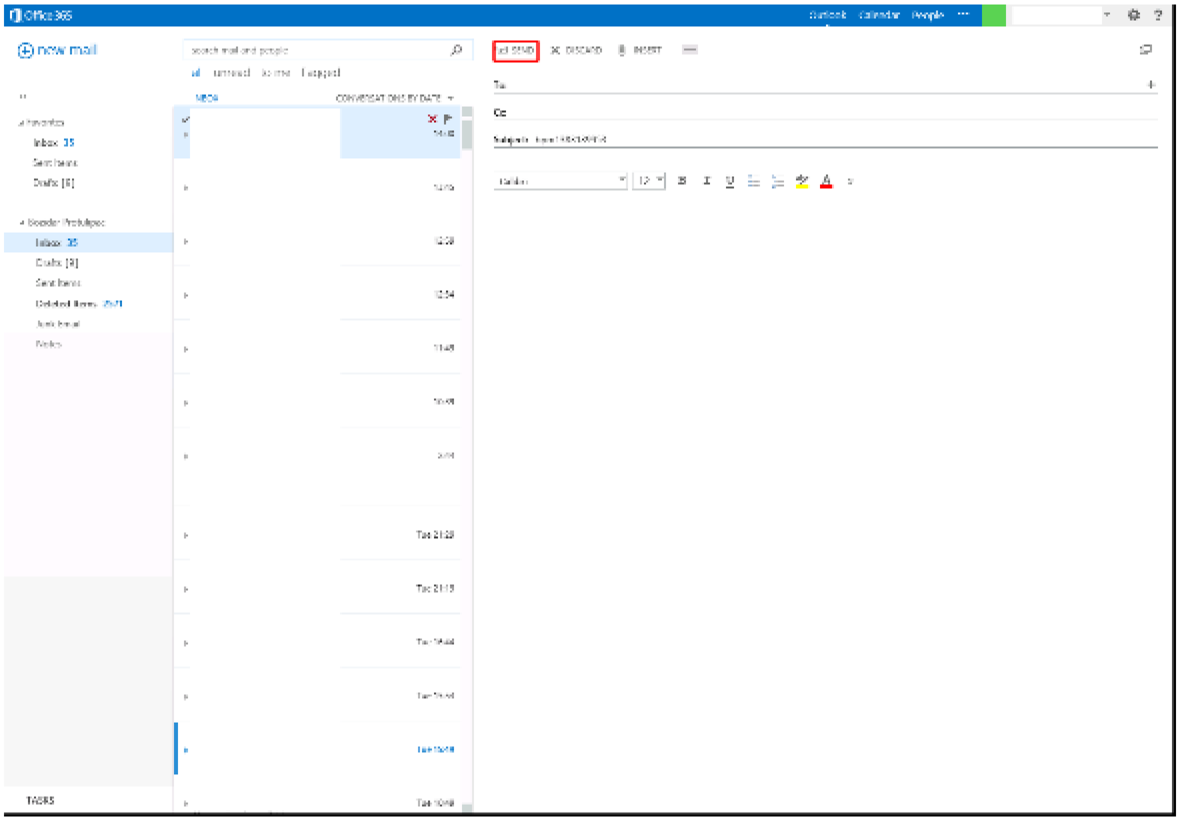
Value: LR.getParam("randnum2");
Clear: Default (true)
Typing Interval: Run-time Settings (350)

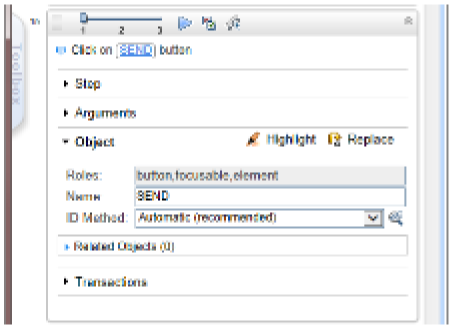
```
LR.getParam("randnum2");
```

In this step, we acquire the value from the parameter:

```
LR.getParam("randnum2");
```

Click Send





Click on SEND button

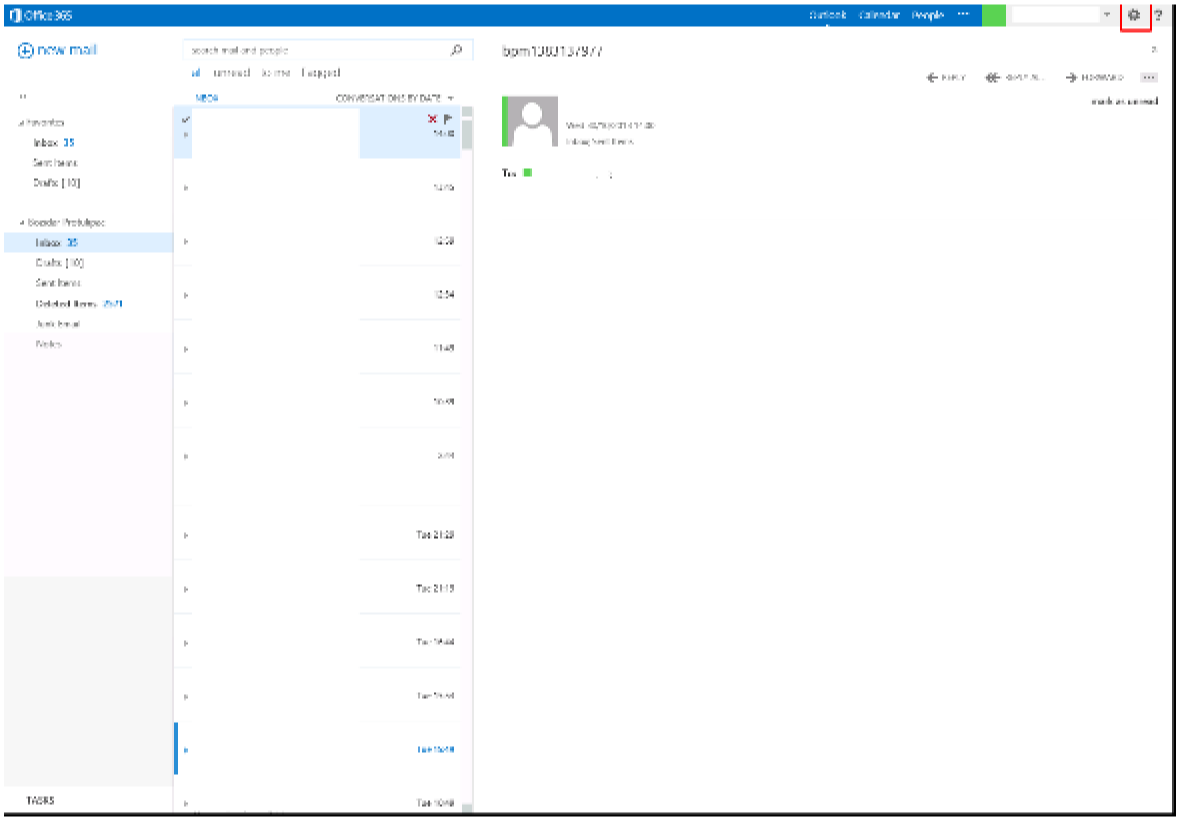
- Step
- Arguments
- Object — Highlight — Replace

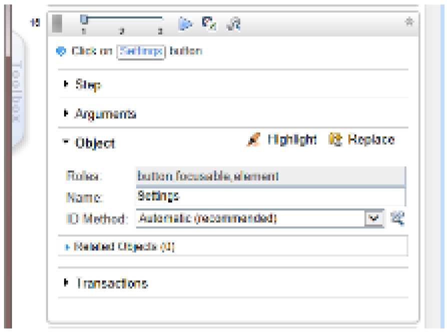
Roles:	button,focusable,element
Name:	SEND
ID Method:	Automatic (recommended)

- Related Objects (0)
- Transactions

Click **Send**. The Send button is easy to recognize so there was no need for any additional corrections.

Click Settings





Click on Settings button

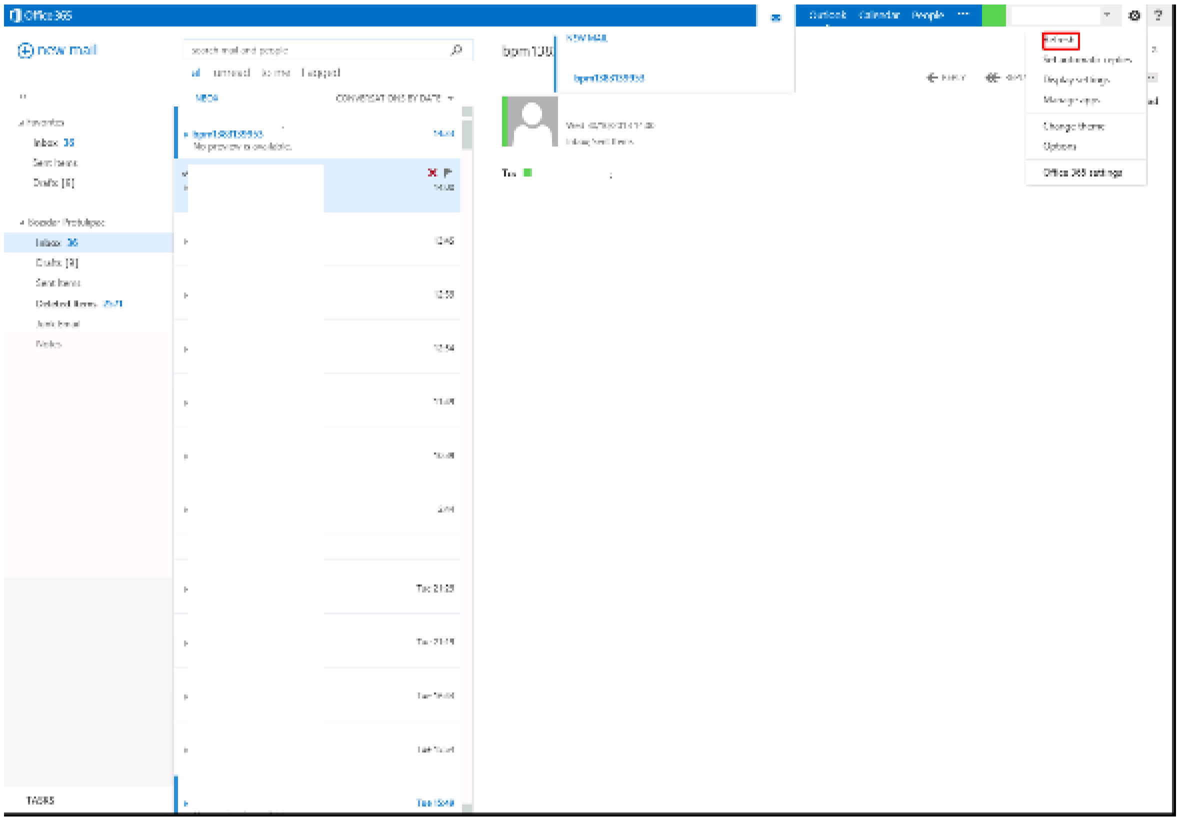
- Step
- Arguments
- Object — Highlight — Replace

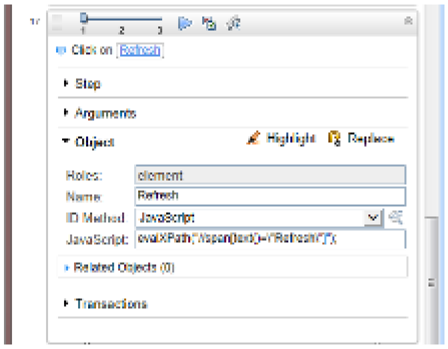
Roles: button,focusable,element
Name: Settings
ID Method: Automatic (recommended)

- Related Objects (0)
- Transactions

Click **Settings** to refresh the Inbox and check whether the email arrived.

Click Refresh





Click on Refresh

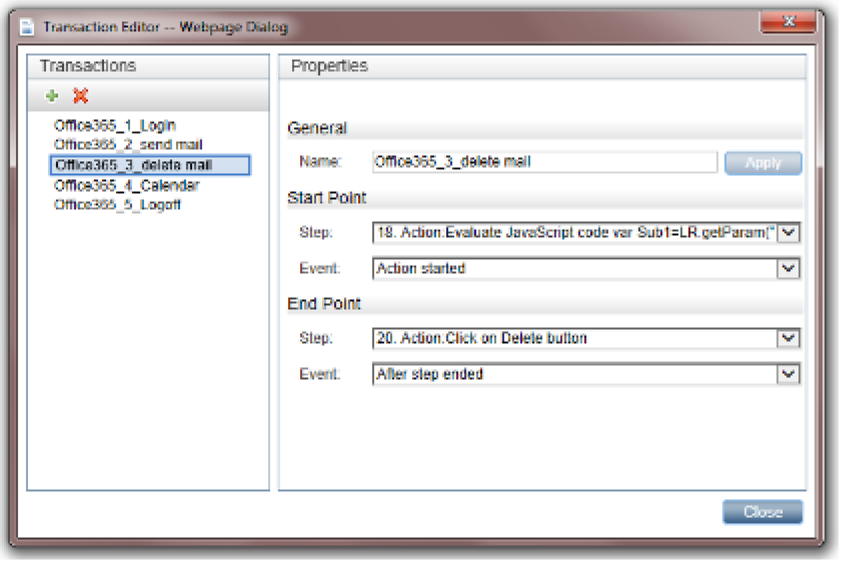
Roles: element
Name: Refresh
ID Method: JavaScript
JavaScript: evalXPath("//span[text()=\"Refresh\"]");

After clicking **Refresh**, the transaction completes and we confirm that the email arrived. If the email still has not arrived, repeat the previous step (Click Settings) and this step (Click Refresh).

We use the JavaScript ID Method:

```
evalXPath("//span[text()=\"Refresh\"]");
```

Transaction — Office365_3_delete mail



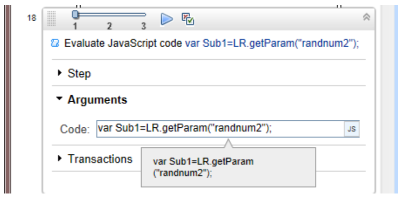
Office365_1_Login
Office365_2_send mail
Office365_3_delete mail
Office365_4_Calendar
Office365_5_Logoff

General
Name: Office365_3_delete mail
Start Point
Step: 18. Action.Evaluate JavaScript code var Sub1=LR.getParam("
Event: Action started
End Point
Step: 20. Action.Click on Delete button
Event: After step ended

The name of this transaction is "Office365_3_delete mail".

This transaction contains all the steps necessary for checking and deleting an email that was sent in the current interaction.

Evaluate JavaScript code var Sub1=LR.getParam ('randnum2');



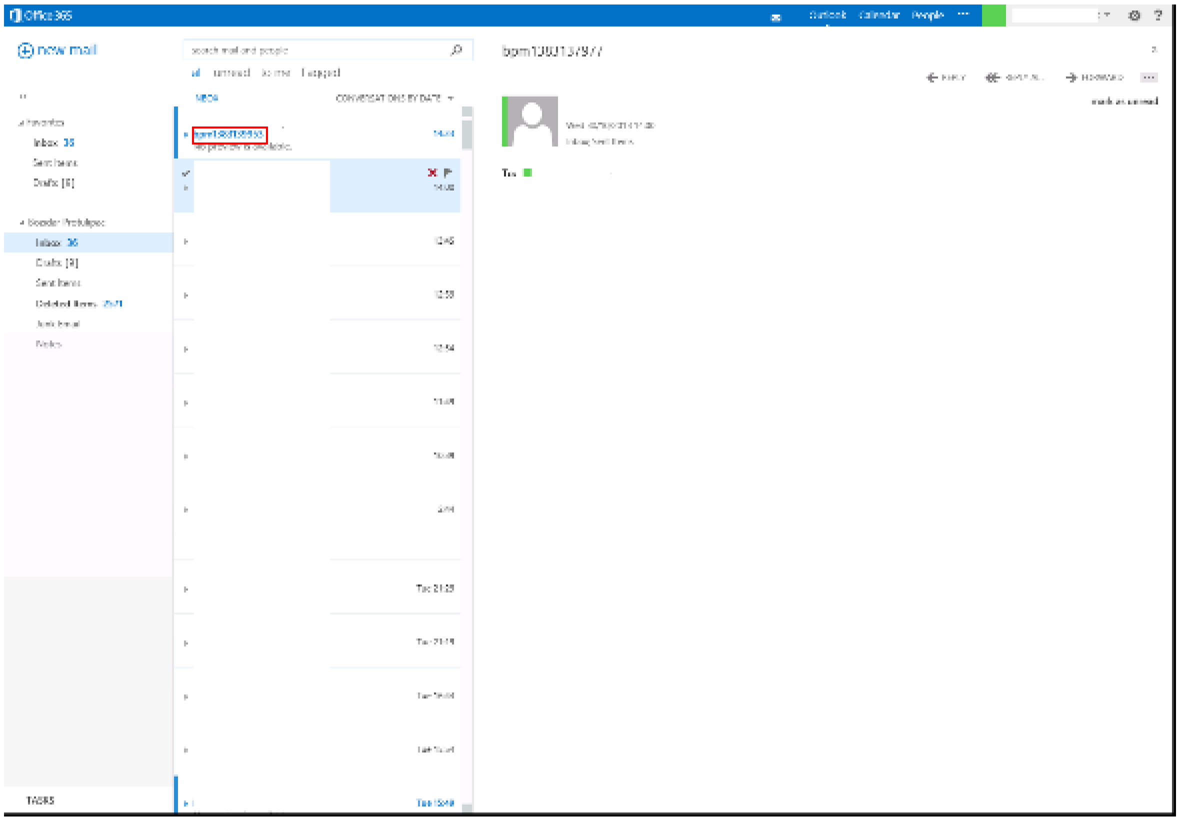
Evaluate JavaScript code var Sub1=LR.getParam("randnum2");
Code: var Sub1=LR.getParam("randnum2");

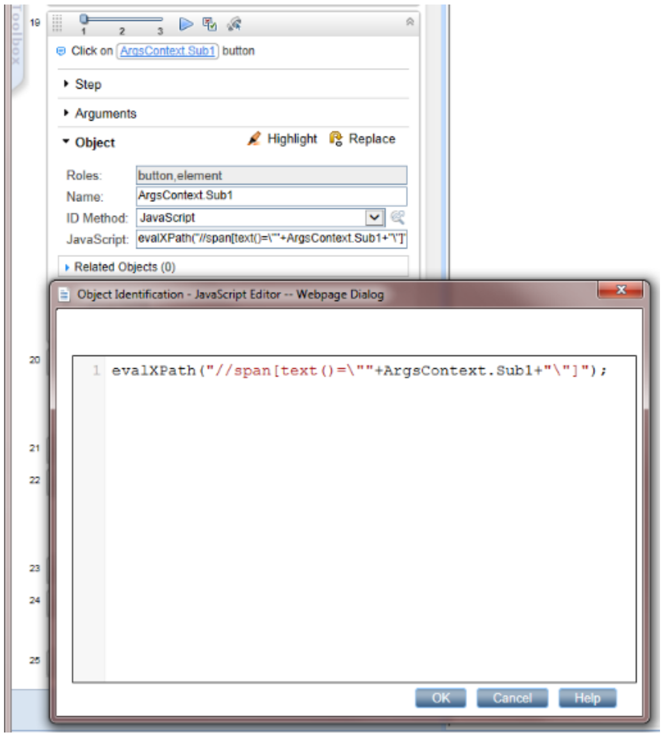
Create the Sub1 variable and get its value from the parameter *randum2*:

```
var Sub1=LR.getParam("randnum2");
```

This variable is used to check (identify) and delete a specific email sent within this interaction. The life span of this variable is within the page. When you change the page, you must re evaluate this JavaScript if you need a value from the *randum2* parameter.

Click ArgsContext.Sub1




Click on ArgsContext.Sub1 button
Step
Arguments
Object
Highlight
Replace
Related Objects (0)
Object Identification - JavaScript Editor -- Webpage Dialog
OK
Cancel
Help

Roles: button,element
Name: ArgsContext.Sub1
ID Method: JavaScript
JavaScript: evalXPath("//span[text()=\""+ArgsContext.Sub1+"\"]

```
evalXPath("//span[text()=\""+ArgsContext.Sub1+"\"]");
```

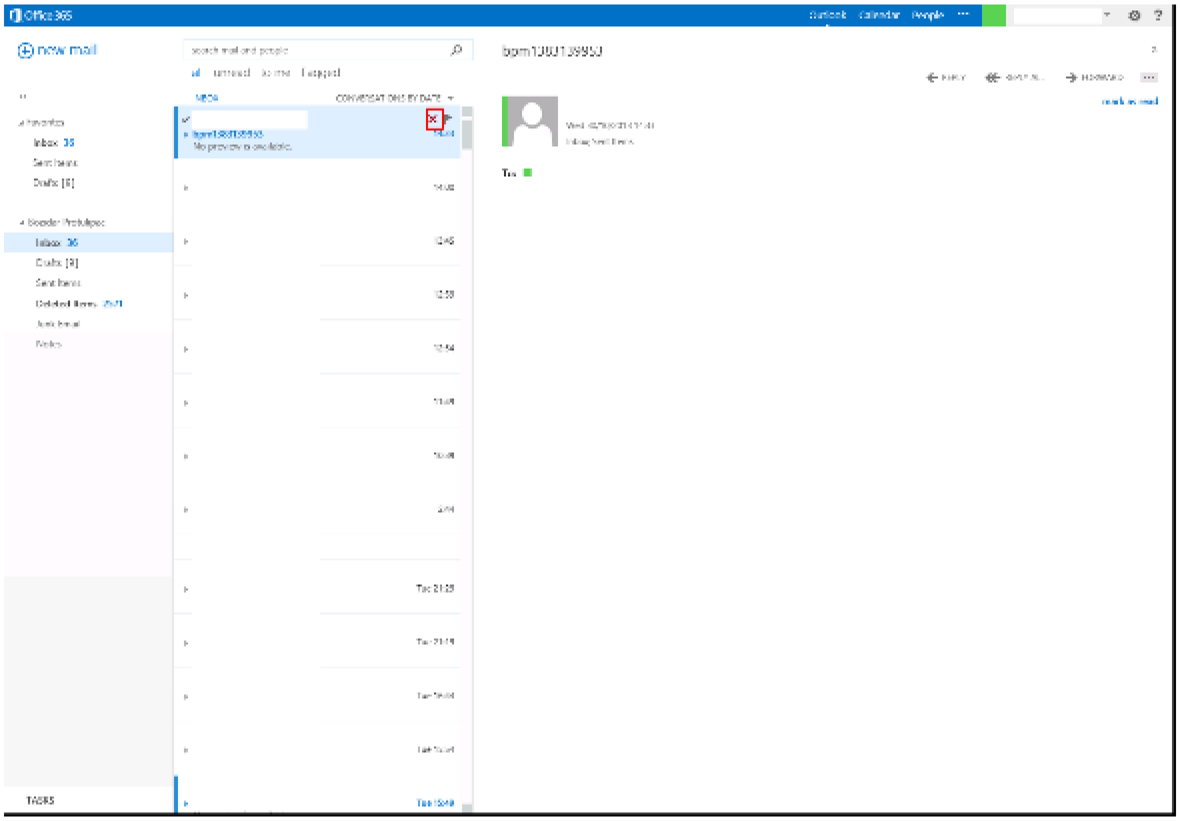
To identify an object, the name must be set as ArgsContext.Sub1

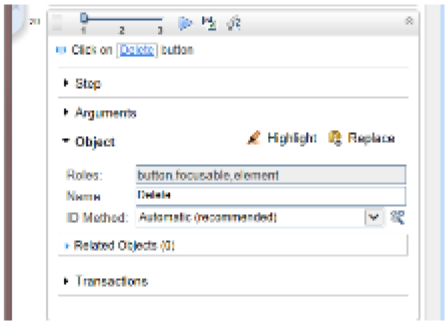
The ID Method is JavaScript:

```
evalXPath("//span[text()=\""+ArgsContext.Sub1+"\"]");
```

The specific email is found.

Click Delete




Click on Delete button

- Step
- Arguments
- Object

Highlight Replace

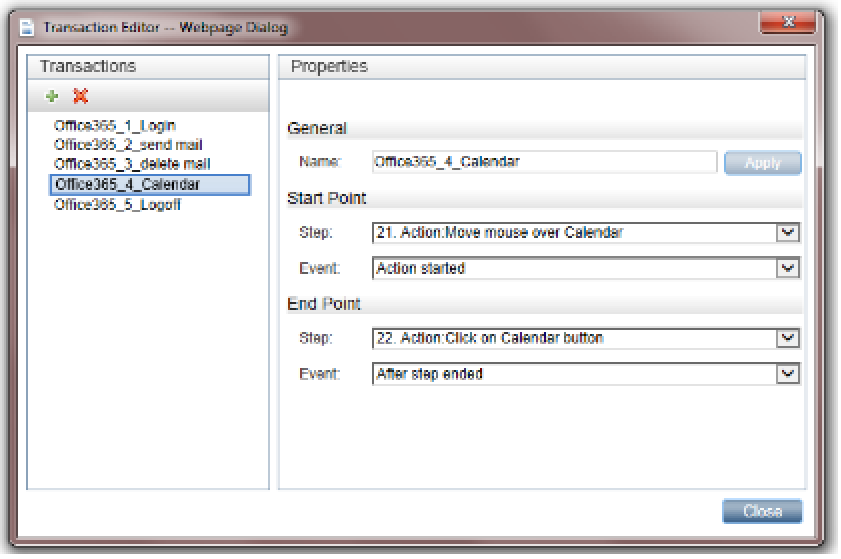
Roles: button,focusable,element
Name: Delete
ID Method: Automatic (recommended)

- Related Objects (0)
- Transactions

The **Delete** button is easy to recognize.

This step completes Office365_3_delete mail.

Transaction — Office365_4_Calendar



Office365_1_Login
Office365_2_send mail
Office365_3_delete mail
Office365_4_Calendar
Office365_5_Logoff

General

Name: Office365_4_Calendar

Start Point

Step: 21. Action:Move mouse over Calendar

Event: Action started

End Point

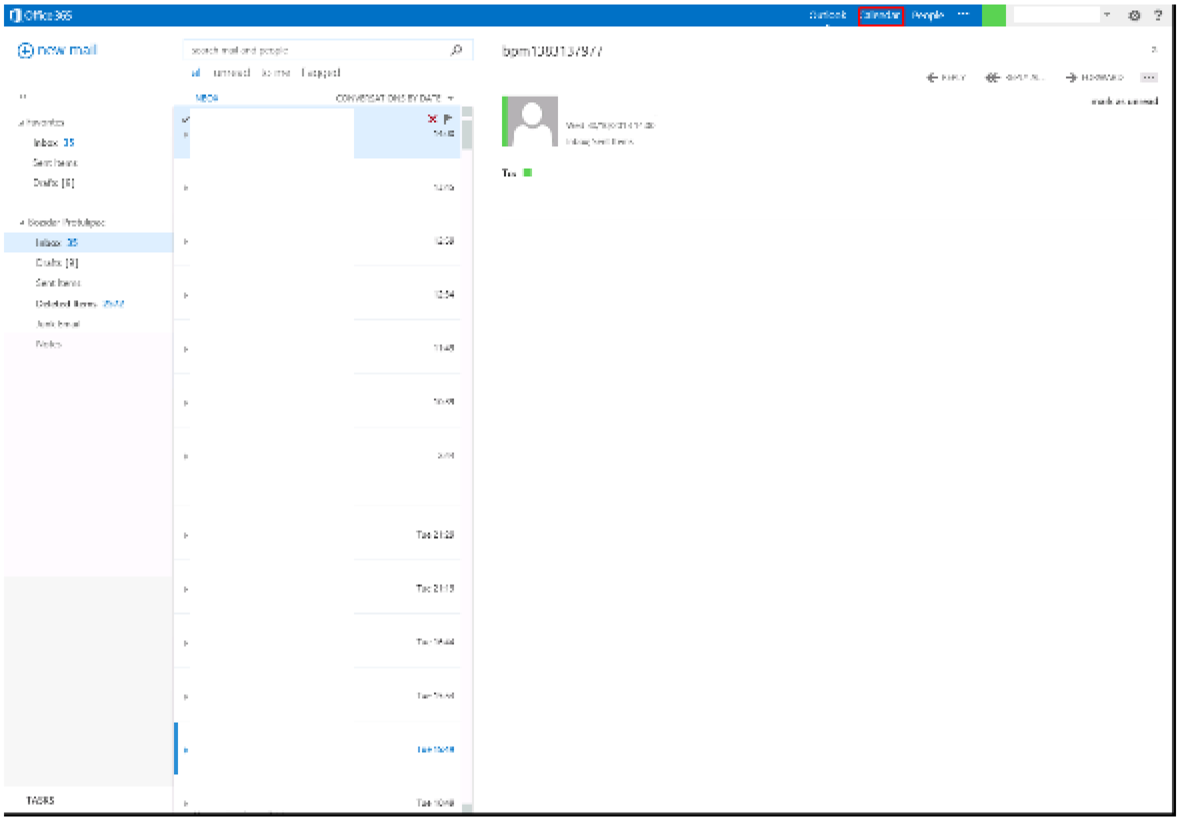
Step: 22. Action:Click on Calendar button

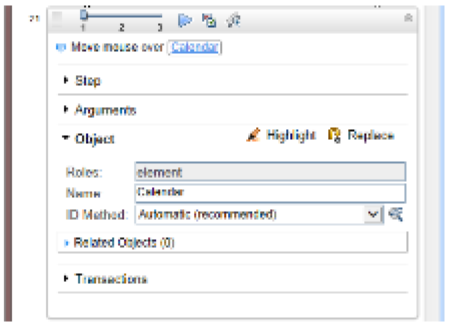
Event: After step ended

The name of this transaction is “Office365_4_Calendar”.

This transaction contains the steps necessary for confirming that the calendar loads.

Move Mouse over Calendar





Move mouse over Calendar

Step

Arguments

Object

Highlight Replace

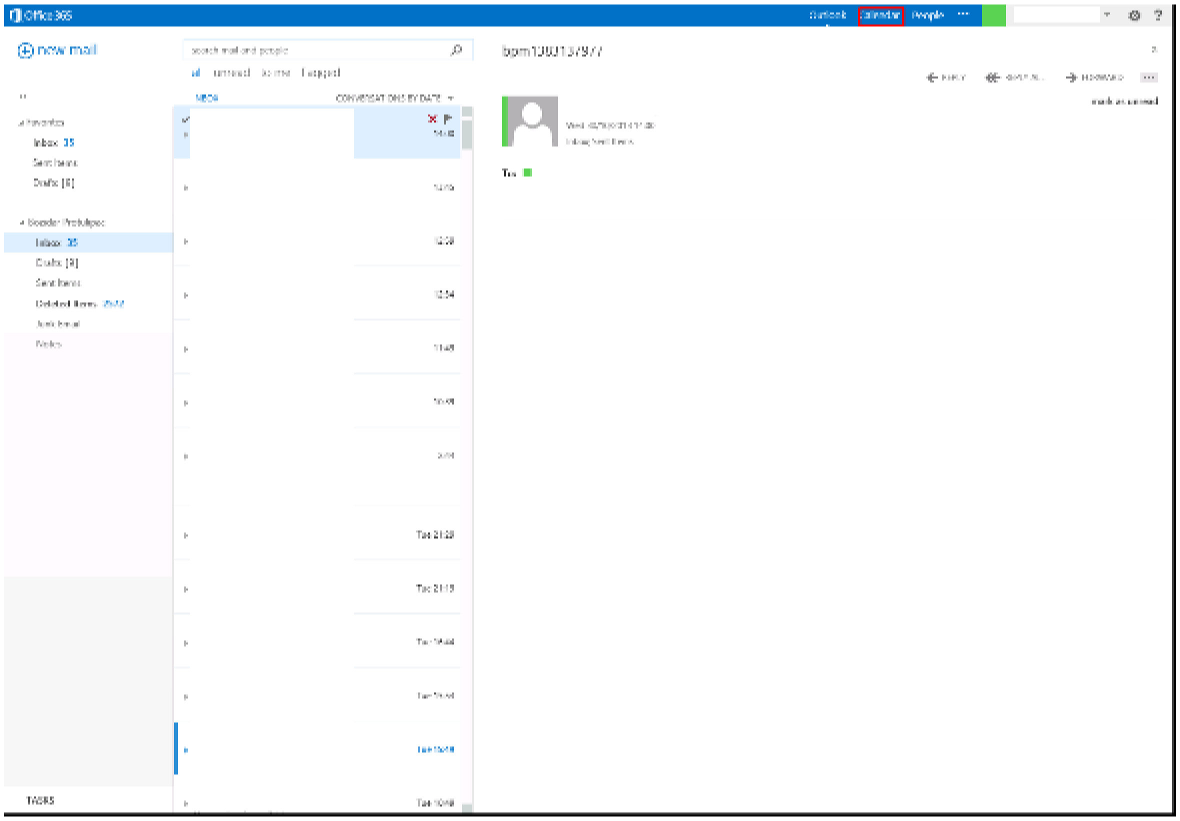
Roles: element
Name: Calendar
ID Method: Automatic (recommended)

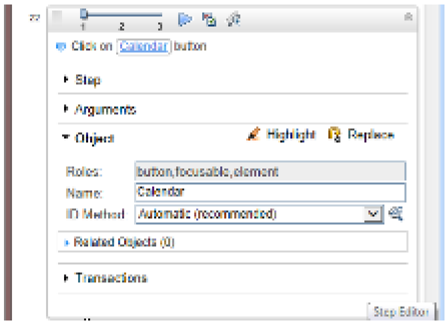
Related Objects (0)

Transactions

This section displays how to create an extra step (mouse over) before clicking an element.

Click Calendar




Click on Calendar button

- Step
- Arguments
- Object

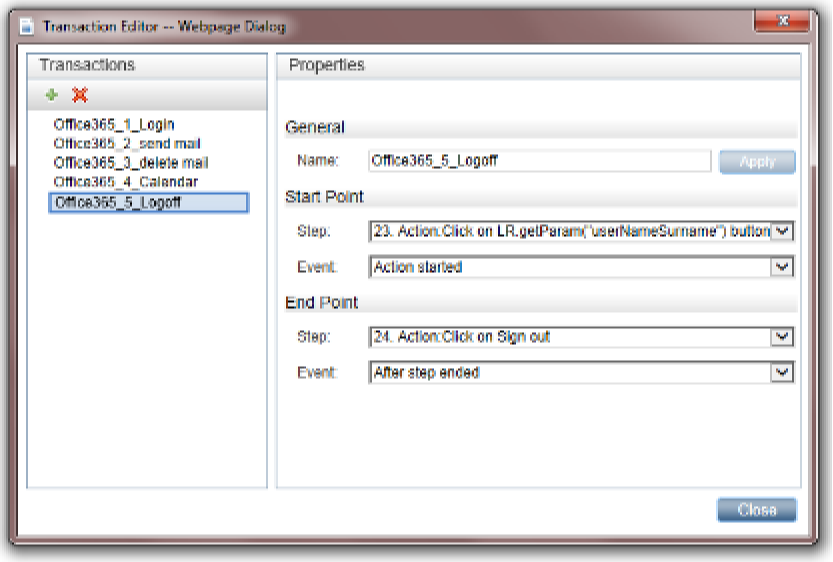
Roles: button,focusable,element
Name: Calendar
ID Method: Automatic (recommended)

Related Objects (0)

- Transactions

This is the end of transaction “Office365_4_Calendar.

Transaction — Office365_5_Logoff



Office365_1_Login
Office365_2_send mail
Office365_3_delete mail
Office365_4_Calendar
Office365_5_Logoff

General

Name: Office365_5_Logoff

Start Point

Step: 23. Action:Click on LR.getParam("userNameSurname") button

Event: Action started

End Point

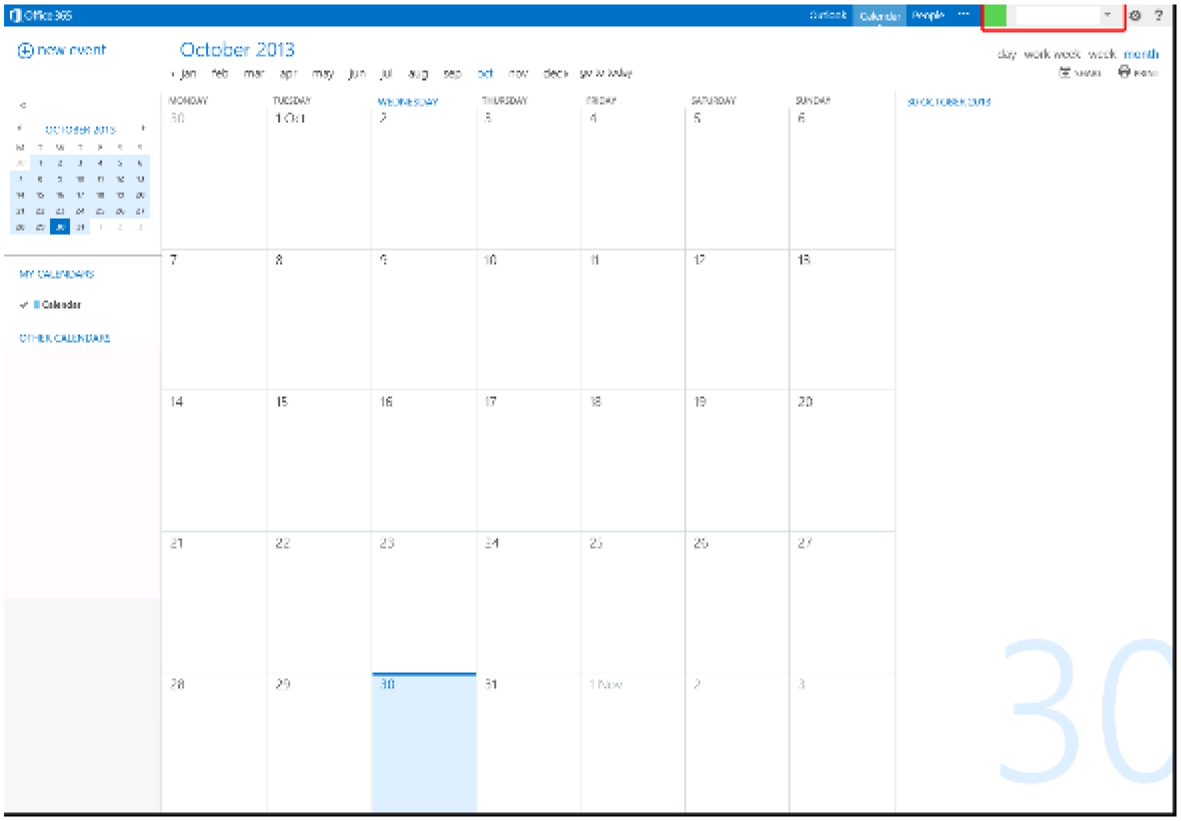
Step: 24. Action:Click on Sign out

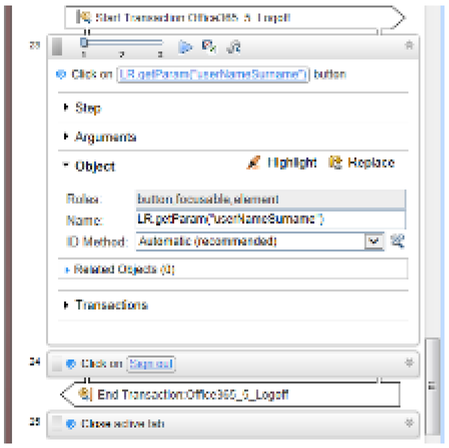
Event: After step ended

The name of this transaction is “Office365_5_Logoff”.

This transaction contains all the steps necessary for logging off.

Click LR.getParam('userNameSurname')





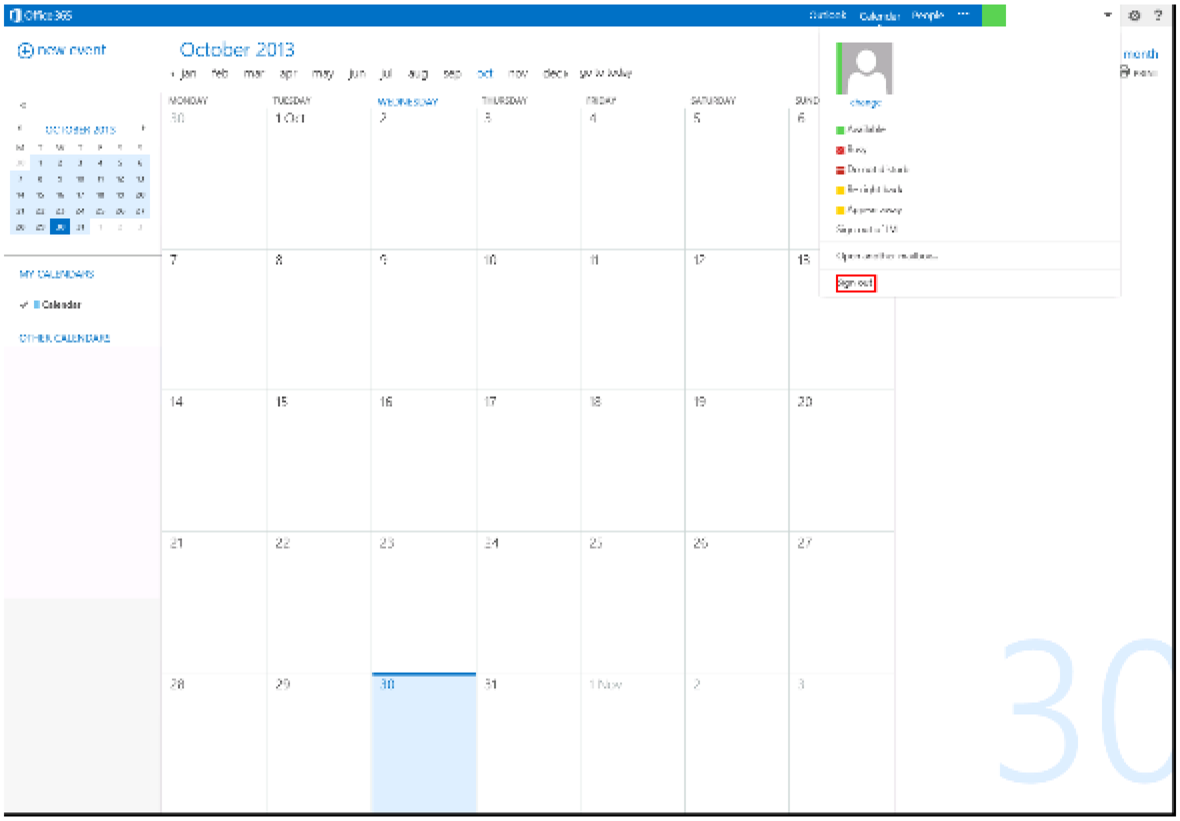
Click on LR.getParam("userNameSurname") button

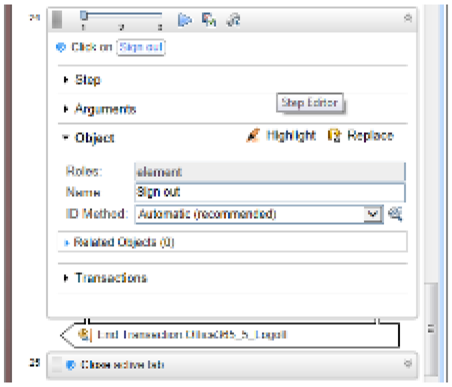
Roles: button,focusable,element
Name: LR.getParam("userNameSurname")
ID Method: Automatic (recommended)

We used another parameter because for logging out of this application you must first click the full user name stored in the parameter *userNameSurname*.

```
LR.getParam("userNameSurname")
```

Click Sign Out





Click on Sign out

Step

Arguments

Object

Roles:	element
Name:	Sign out
ID Method:	Automatic (recommended)

Related Objects (0)

Transactions

End Transaction Office365_5_Logoff

Close active tab

By clicking **Sign out**, we end this transaction and script.

All we need to do now is close the active tab.

Chapter 4: Cloud Email Provider – Gmail Transaction Flow

This case study uses Gmail cloud service. The case study uses one flow/script with the following transactions:

1. **Login** – Open a login page, enter your credentials, and confirm that the first page loads.
2. **Create an email** – Create a new email and send it with a unique subject to the logged in user.
3. **Delete the email** – Check for this unique email, select it, and delete it.
4. **Logoff** – Log off from the web application.

The main purpose of running this script is to confirm that the application's processes run successfully. If successful, the transactions complete without any errors.

You can view a sample Gmail script by downloading the following file https://softwaresupport.hpe.com/group/softwaresupport/search-result/-/facetsearch/document/KM00658288.

General Flow

Google Mail is another name for Google's email tool, Gmail. Gmail is a free, full-featured email service. Anyone can register for an account. Gmail is also available as part of Google Apps. In this flow, we test the email functionality of Gmail.

A sequence of transactions creates the business process flow. A transaction is a unit that is measurable by availability and performance. Therefore, we group activities that perform a specific transaction.

To describe and validate these transactions, logically group the transactions into the following steps:

1. Login.
    a. Navigate to https://gmail.com.
    b. Enter your credentials and confirm that the first page (Inbox) loads.
2. Create a new email.
    a. Click **Compose**.
    b. Enter the email address used to login in step 1.

   c. Enter a unique subject.

   d. Send the email.

3. Delete the email.

   a. Refresh and check if the email arrived.

   b. Select the email.

   c. Delete the email.

4. Logoff

   a. Log off from the web application.

The following sections describe how to create these transactions.

A good approach, before using this tool, is to know what we want to achieve. Create a blueprint as follows:

1. Run the Gmail web application in Internet Explorer 9. Perform the steps of the flow and create annotations.

2. Determine the boundaries that form the transactions.

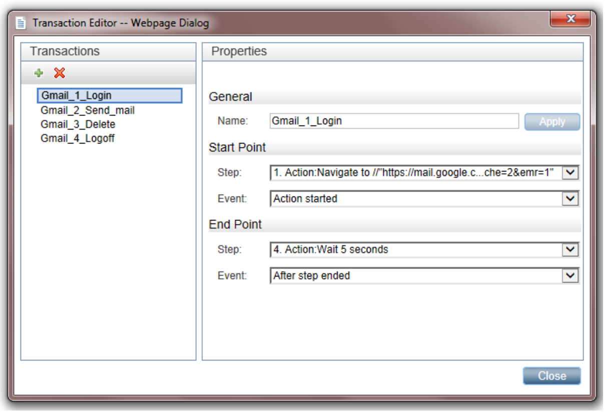
3. Select which values to assign to the parameters.

4. Select what values to protect and encrypt.

Record all the information you collect since it will come in handy later.

We recommend you start using the script. Refer to sections in this document for assistance.

Transaction — Gmail_1_Login



Gmail_1_Login
Gmail_2_Send_mail
Gmail_3_Delete
Gmail_4_Logoff

General

Name: Gmail_1_Login

Start Point

Step: 1. Action:Navigate to //"https://mail.google.c...che=2&emr=1"

Event: Action started

End Point

Step: 4. Action:Wait 5 seconds

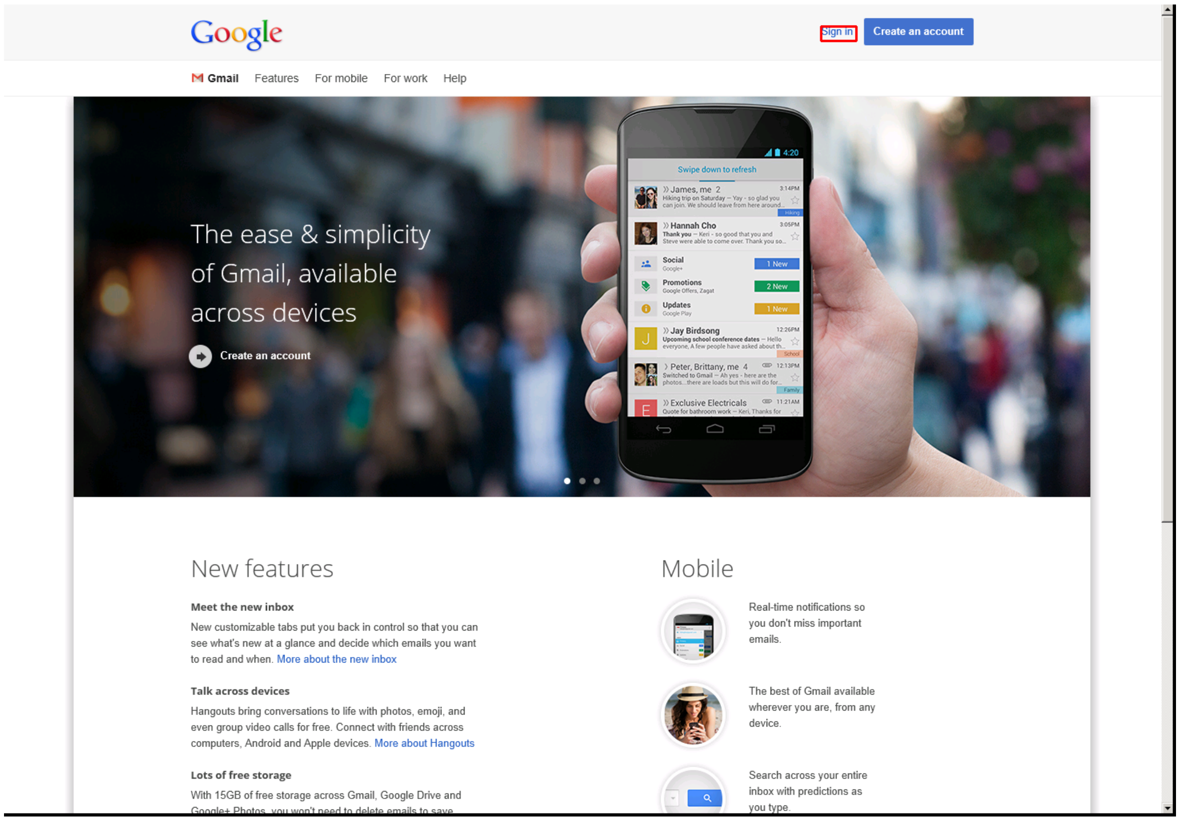
Event: After step ended

Define the name of the transaction. In this example, we call this transaction “Gmail_1_Login”. Naming conventions should provide a clear view of the application, the transaction order, and create an overview of the business processes covered by this transaction.

Define the steps and events for the starting and ending points of the transaction. The most appropriate event for the start step is **Action started**. The most appropriate event for the end step is **After step ended**.

This transaction contains all the steps required for logging into the Gmail application.

Navigate to https://mail.google.c...che=2&emr=1



The ease & simplicity of Gmail, available across devices

New features

Meet the new inbox

New customizable tabs put you back in control so that you can see what's new at a glance and decide which emails you want to read and when. More about the new inbox

Talk across devices

Hangouts bring conversations to life with photos, emoji, and even group video calls for free. Connect with friends across computers, Android and Apple devices. More about Hangouts

Lots of free storage

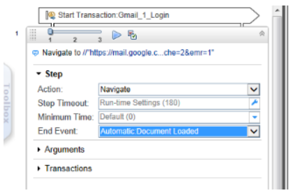
With 15GB of free storage across Gmail, Google Drive and Google+ Photos, you won't need to delete emails to save

Mobile

Real-time notifications so you don't miss important emails.

The best of Gmail available wherever you are, from any device.

Search across your entire inbox with predictions as you type.



Start Transaction:Gmail_1_Login

Navigate to //"https://mail.google.c...che=2&emr=1"

Step

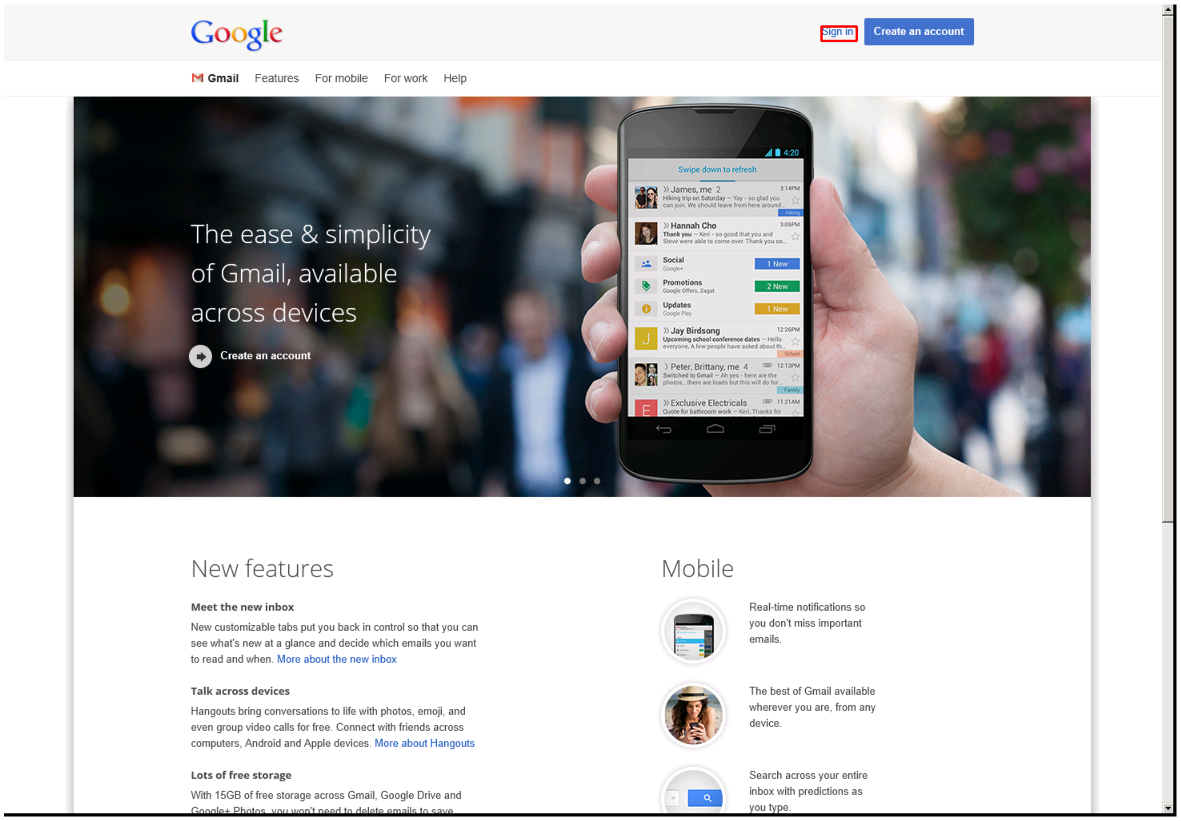
Action:	Navigate
Step Timeout:	Run-time Settings (180)
Minimum Time:	Default (0)
End Event:	Automatic:Document Loaded

In the Navigation step, specify the URL.

While recording, VuGen automatically selects the appropriate End Event, in this case, **Document downloaded**. The step ends when the process of loading a document completes. In most cases, you can freely select the default values.

If the pages load slowly, you can use a Wait step to make sure the page loads. For information on Wait steps, see *Meaning of Wait Step* on page 94.

Click Sign In



Meet the new inbox

New customizable tabs put you back in control so that you can see what's new at a glance and decide which emails you want to read and when. More about the new inbox

Talk across devices

Hangouts bring conversations to life with photos, emoji, and even group video calls for free. Connect with friends across computers, Android and Apple devices. More about Hangouts

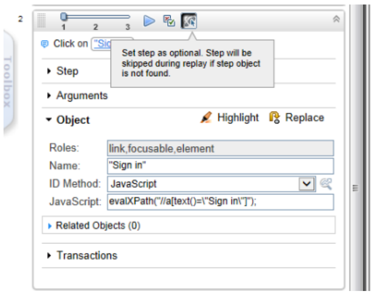
Lots of free storage

With 15GB of free storage across Gmail, Google Drive and Google+ Photos, you won't need to delete emails to save

Real-time notifications so you don't miss important emails.

The best of Gmail available wherever you are, from any device.

Search across your entire inbox with predictions as you type.



Click on "Sig

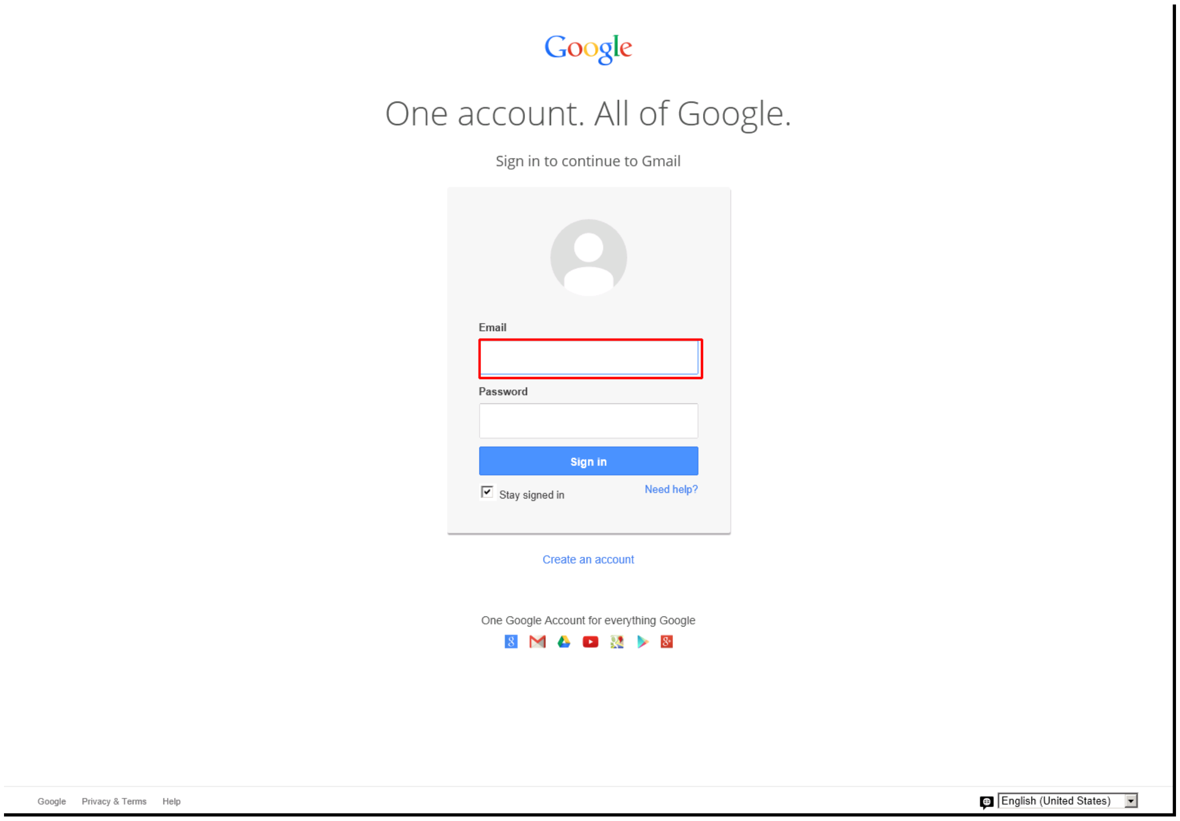
Set step as optional. Step will be skipped during replay if step object is not found.

Roles:	link,focusable,element
Name:	"Sign in"
ID Method:	JavaScript
JavaScript:	evalXPath("//a[text()=\"Sign in\"]");

Sometimes the application does not go straight to the login page. Therefore, insert this optional step, just in case. If the Sign in link does not appear, this step is ignored.

Note: Since providers make changes in their services, you may need to make changes to your scripts to maintain the same outcome.

Click in Username Textbox



Google

One account. All of Google.

Sign in to continue to Gmail

Email

Password

Sign in

Stay signed in

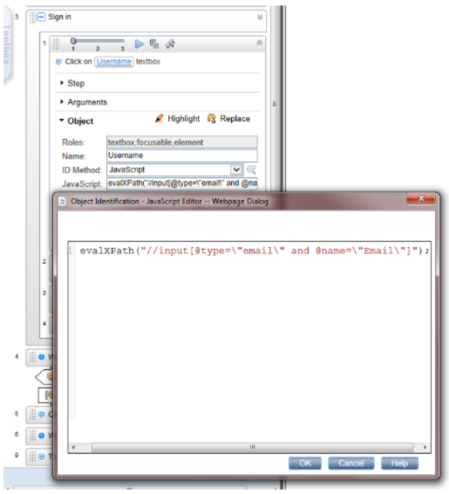
Need help?

Create an account

One Google Account for everything Google

Google Privacy & Terms Help

English (United States)



Sign in

Click on Username textbox

Step

Arguments

Object Highlight Replace

Roles:	textbox,focusable,element
Name:	Username
ID Method:	JavaScript
JavaScript:	evalXPath("//input[@type=\"email\" and @na

Object Identification - JavaScript Editor -- Webpage Dialog

```
evalXPath("//input[@type=\"email\" and @name=\"Email\"]");
```

OK Cancel Help

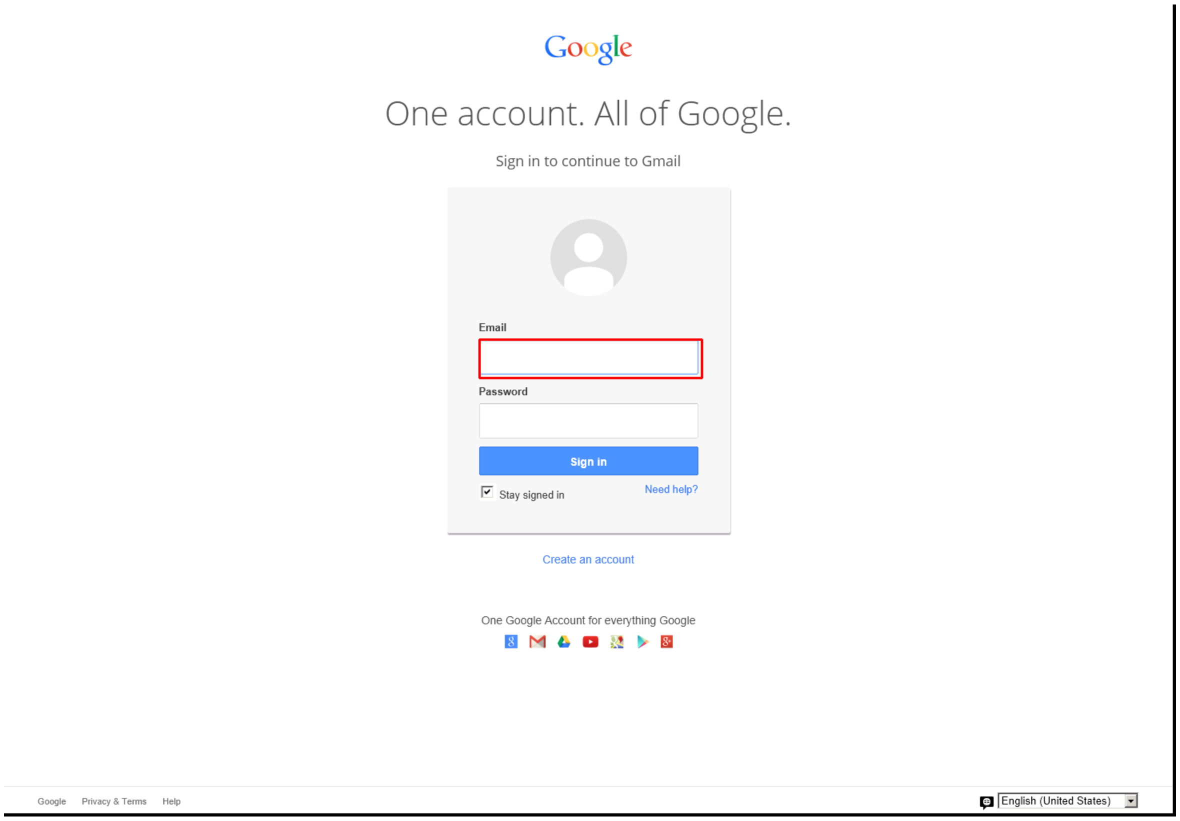
You can either select the **Automatic** or **JavaScript** ID method.

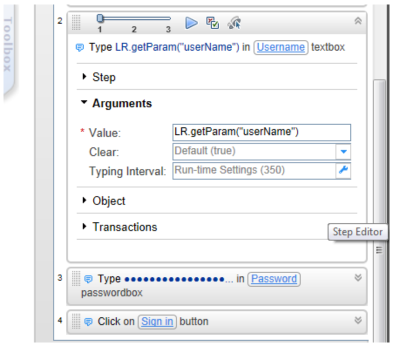
For the JavaScript ID Method, enter the following in the JavaScript field:

```
evalXPath("//input[@type=\"email\" and @name=\"Email\"]");
```

There are cases when you will change the ID Method, as we will see later.

Type LR.getParam('userName') in Username Textbox





2 Type LR.getParam("userName") in Username textbox

Step

Arguments

* Value:	LR.getParam("userName")
Clear:	Default (true)
Typing Interval:	Run-time Settings (350)

Object

Transactions

3 Type ●●●●●●●●●●●●●●●●... in Password passwordbox

4 Click on Sign in button

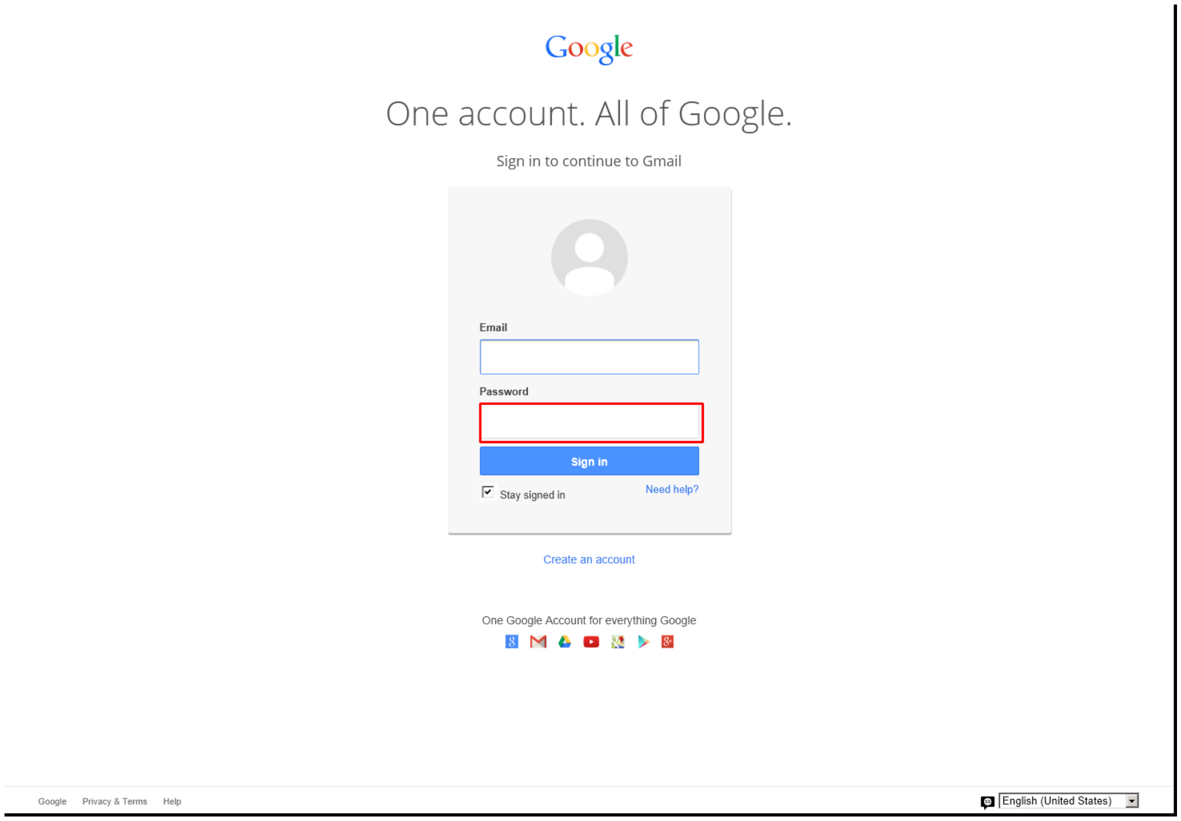
Type the username to the **Username** textbox.

Since we are using parameters, you can enter the *LR.getParam("userName");* command in the Type the username to the Username textbox.

Since we are using parameters, you can enter the LR.getParam("userName"); command in the Value field and the result is the defined username.

field and the result is the defined username.

Type **************** in Password Textbox


Google

One account. All of Google.

Sign in to continue to Gmail

Email

Password

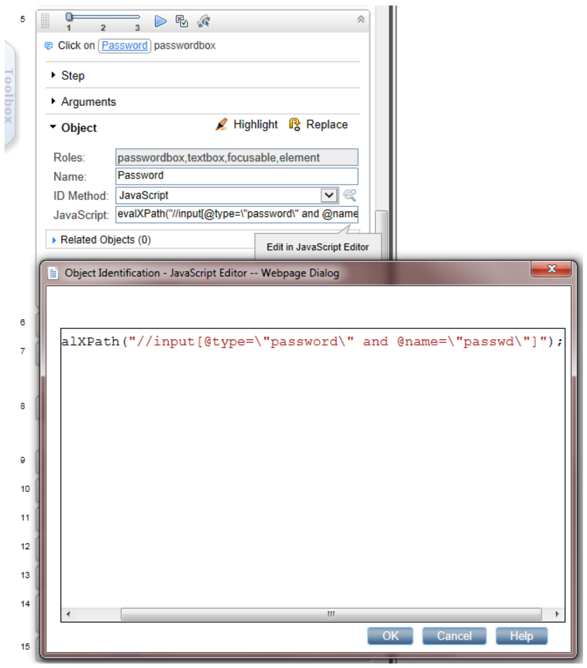
Sign in

☑ Stay signed in   Need help?

Create an account

One Google Account for everything Google

Google   Privacy & Terms   Help   English (United States)


Click on Password passwordbox

- Step
- Arguments
- Object   Highlight   Replace

Roles:	passwordbox,textbox,focusable,element
Name:	Password
ID Method:	JavaScript
JavaScript:	evalXPath("//input[@type=\"password\" and @name

- Related Objects (0)   Edit in JavaScript Editor

Object Identification - JavaScript Editor -- Webpage Dialog

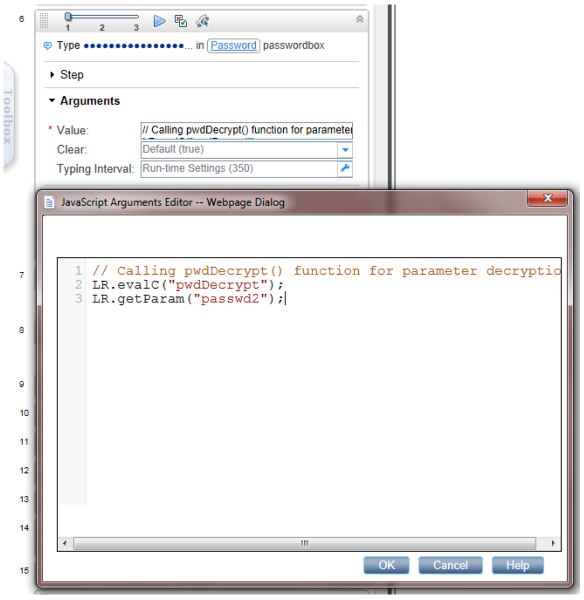
```
alXPath("//input[@type=\"password\" and @name=\"passwd\"]");
```

OK   Cancel   Help

Similar to the *Click in Username Textbox* step, you can select the **Automatic** or **JavaScript** ID method.

For the JavaScript ID Method, enter the following in the JavaScript field:

```
evalXPath("//input[@type=\"password\" and @name=\"passwd\"]");
```



Type ●●●●●●●●●●●●●●●●... in Password passwordbox

* Value:	// Calling pwdDecrypt() function for parameter
Clear:	Default (true)
Typing Interval:	Run-time Settings (350)

JavaScript Arguments Editor -- Webpage Dialog

```
// Calling pwdDecrypt() function for parameter decryptio
LR.evalC("pwdDecrypt");
LR.getParam("passwd2");
```

Type a password in the **Password** textbox. To display an encrypted password (so no one can see it), type the following:

```
// Calling pwdDecrypt() function for parameter decryption

LR.evalC("pwdDecrypt");

// This is a parameter with a decrypted value

LR.getParam("passwd2");
```

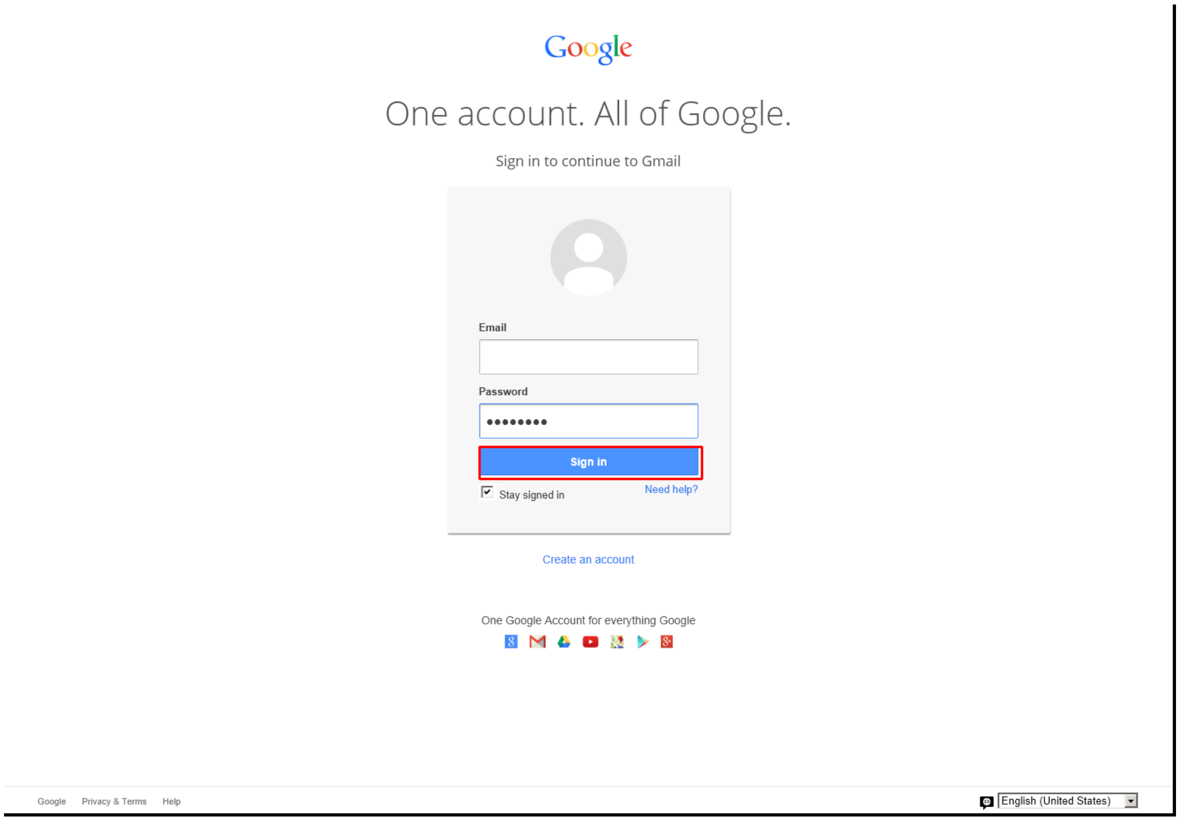
The pwdDecrypt() function is defined in C-function.c file:

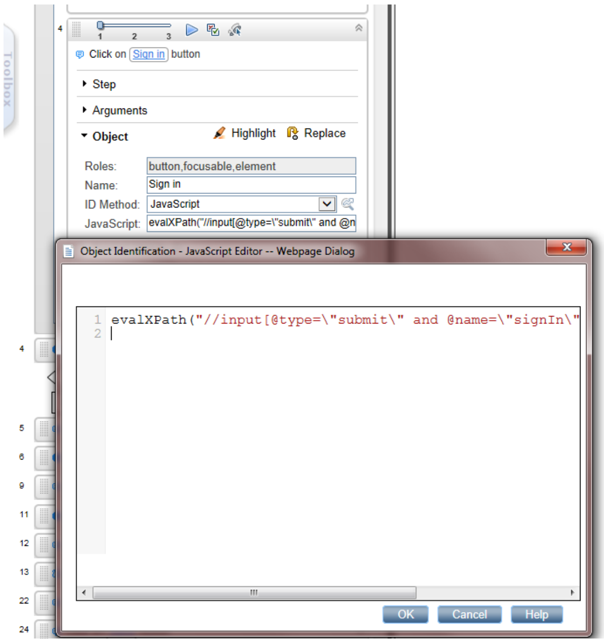
```
void pwdDecrypt(){

lr_save_string(lr_decrypt(lr_eval_string("{passwd}")),"passwd2");

/*If you need to get some value from external parameter file*/

lr_output_message(lr_eval_string("passwd2"));

}
```

`passwd` is the parameter defined for external use (BSM – EUM) and is encrypted.

`passwd2` is the parameter seen only within this script instance.

Click Sign In





Click on Sign in button

Object

Roles: button,focusable,element
Name: Sign in
ID Method: JavaScript
JavaScript: evalXPath("//input[@type=\"submit\" and @n

Object Identification - JavaScript Editor -- Webpage Dialog

```
evalXPath("//input[@type=\"submit\" and @name=\"signIn\"
```

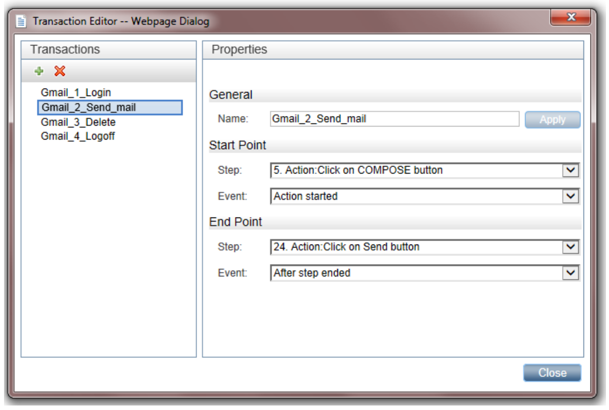
This is defined as the End step for Transaction “Gmail_1_Login”.

The JavaScript ID Method used is:

```
evalXPath("//input[@type=\"submit\" and @name=\"signIn\" and @value=\"Sign in\"]");
```

The End Event Step is set to automatic. So in this case, the End Event Step is **Step Synchronous network completed**. The step ends when all HTTP requests have completed excluding requests initiated by XMLHttpRequest.

Transaction — Gmail_2_Send_mail



Gmail_1_Login
Gmail_2_Send_mail
Gmail_3_Delete
Gmail_4_Logoff

General
Name: Gmail_2_Send_mail

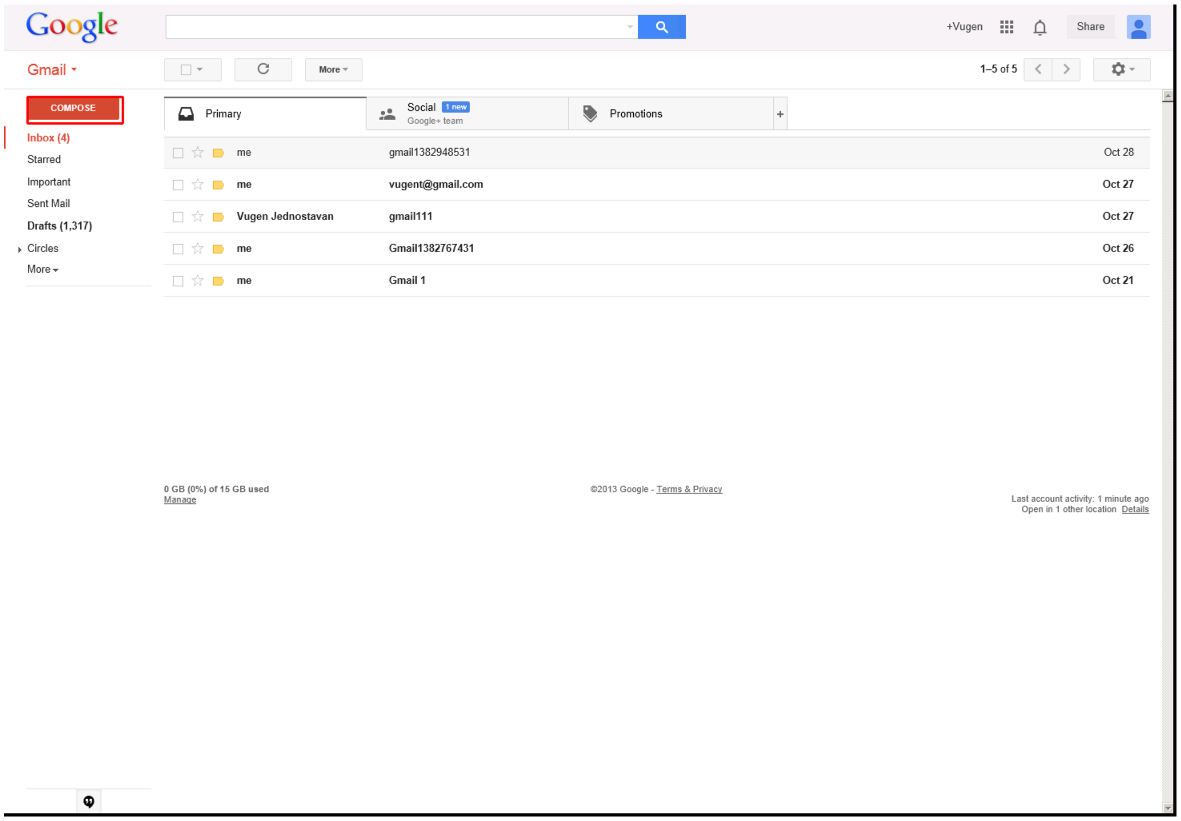
Start Point
Step: 5. Action:Click on COMPOSE button
Event: Action started

End Point
Step: 24. Action:Click on Send button
Event: After step ended

The name of this transaction is “Gmail_2_Send_mail”.

This transaction contains all the steps necessary for sending an email to the logged in user with a unique subject field.

Click Compose

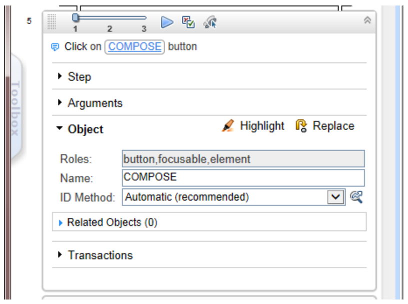


me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21

0 GB (0%) of 15 GB used
Manage

©2013 Google - Terms & Privacy

Last account activity: 1 minute ago
Open in 1 other location Details



Click on COMPOSE button

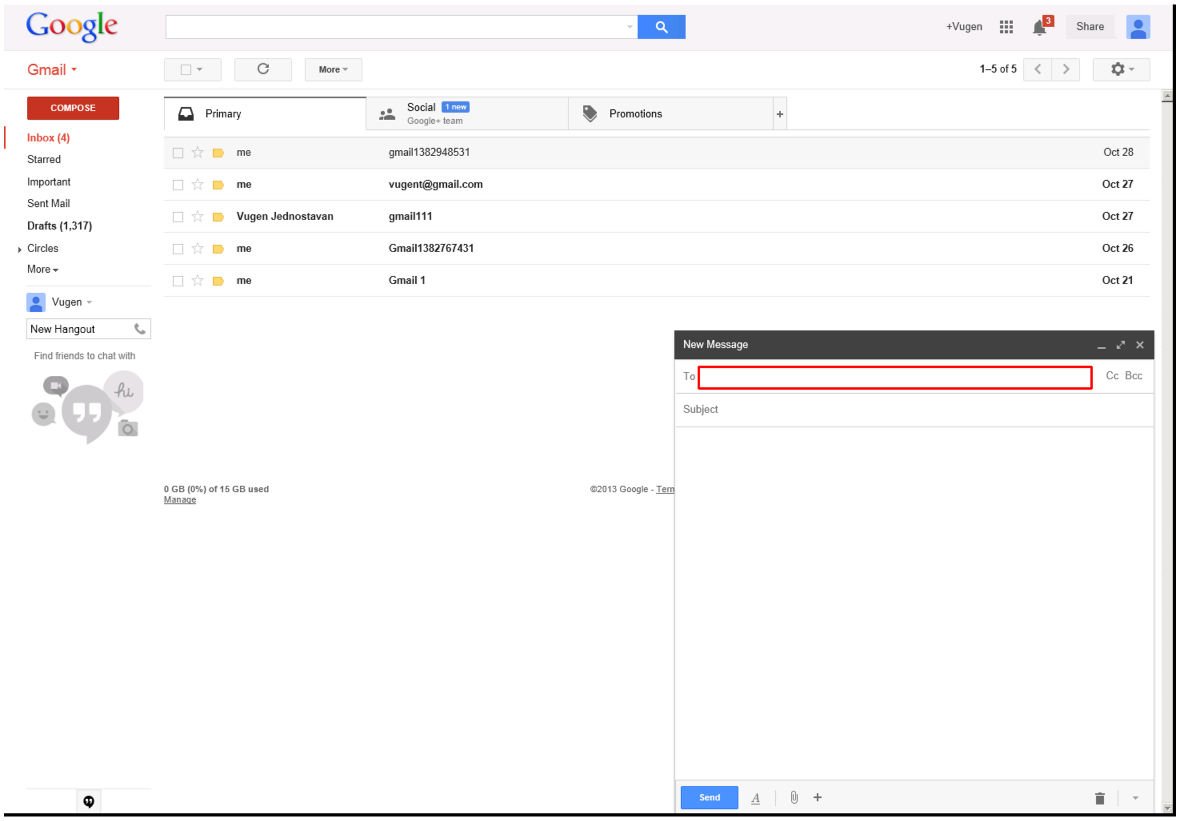
Step
Arguments
Object

Roles: button,focusable,element
Name: COMPOSE
ID Method: Automatic (recommended)

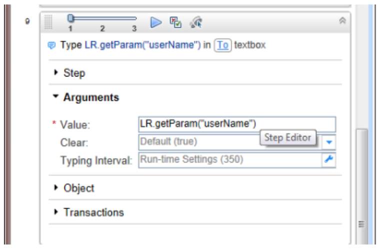
Related Objects (0)
Transactions

When you click **Compose**, a form appears for sending a new email. By running a script multiple times, we can confirm that this element is recognized using the Automatic ID method.

Type LR.getParam('userName') in To Field



me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21


Type LR.getParam("userName") in To textbox

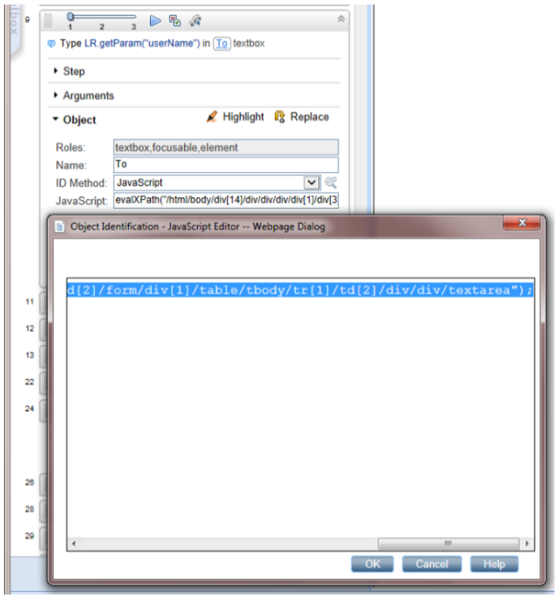
- Step
- Arguments
  - * Value: LR.getParam("userName")
  - Clear: Default (true)
  - Typing Interval: Run-time Settings (350)
- Object
- Transactions

To make this script robust, we use parameters. Therefore, if the username or password is changed at a later time, you do not need to change the script, only the parameter's value. The username is in the form of an email address.

The ID method used in this example is JavaScript:

```
evalXPath("/html/body/div[14]/div/div/div[1]/div[3]/div[1]/div
[1]/div/div/div/div[3]
/div/div/div[4]/table/tbody/tr/td[2]/form/div[1]/table/tbody/tr[1]/td
[2]/div/div/textarea");
```

But this is not a good example, so it is better to use Firebug.



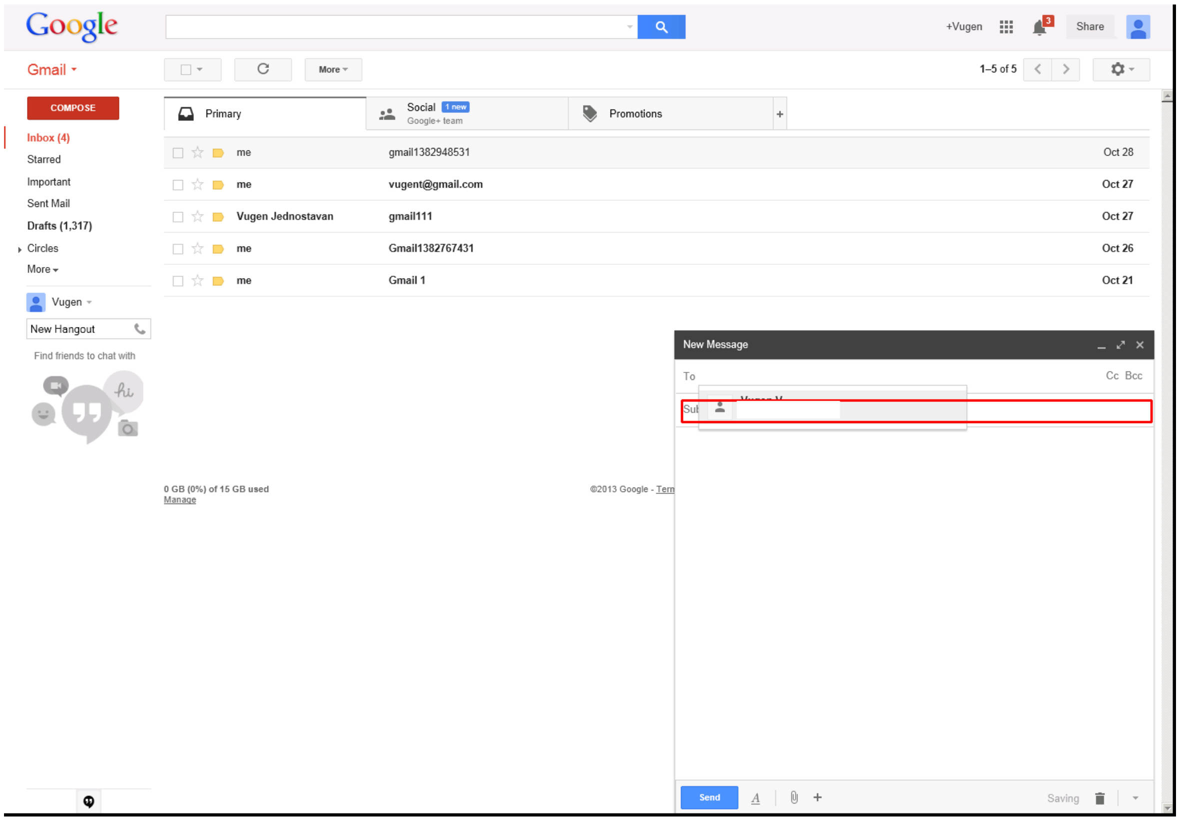
Type LR.getParam("userName") in To textbox

Roles: textbox,focusable,element
Name: To
ID Method: JavaScript
JavaScript: evalXPath("/html/body/div[14]/div/div/div/div[1]/div[3

Object Identification - JavaScript Editor -- Webpage Dialog

```
d[2]/form/div[1]/table/tbody/tr[1]/td[2]/div/div/textarea");
```

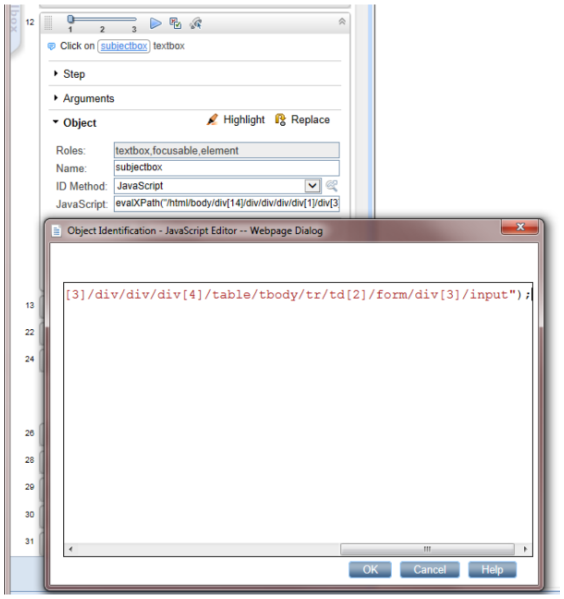
Click Subject Field



me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21

0 GB (0%) of 15 GB used
Manage

©2013 Google - Term


Click on subjectbox textbox

Step

Arguments

Object

Roles: textbox,focusable,element
Name: subjectbox
ID Method: JavaScript
JavaScript: evalXPath("/html/body/div[14]/div/div/div/div[1]/div[3

Object Identification - JavaScript Editor -- Webpage Dialog

```
[3]/div/div/div[4]/table/tbody/tr/td[2]/form/div[3]/input");
```

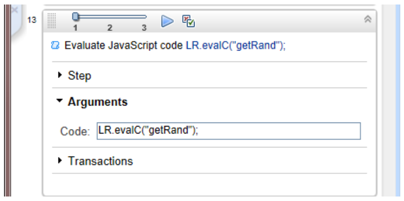
Click in the **Subject** field to select it. The ID method used is JavaScript:

```
evalXPath("/html/body/div[14]/div/div/div/div[1]/div[3]/div[1]/div
[1]/div/div/div/div[3]
/div/div/div[4]/table/tbody/tr/td[2]/form/div[3]/input");
```

But this is not a good example, therefore, it is better to use Firebug usage:

```
evalXPath("//input[@placeholder=\"Subject\" and @name=\"subjectbox\"]");
```

Evaluate JavaScript code LR.evalC('getRand');


Evaluate JavaScript code LR.evalC("getRand");

Step

Arguments

Code: LR.evalC("getRand");

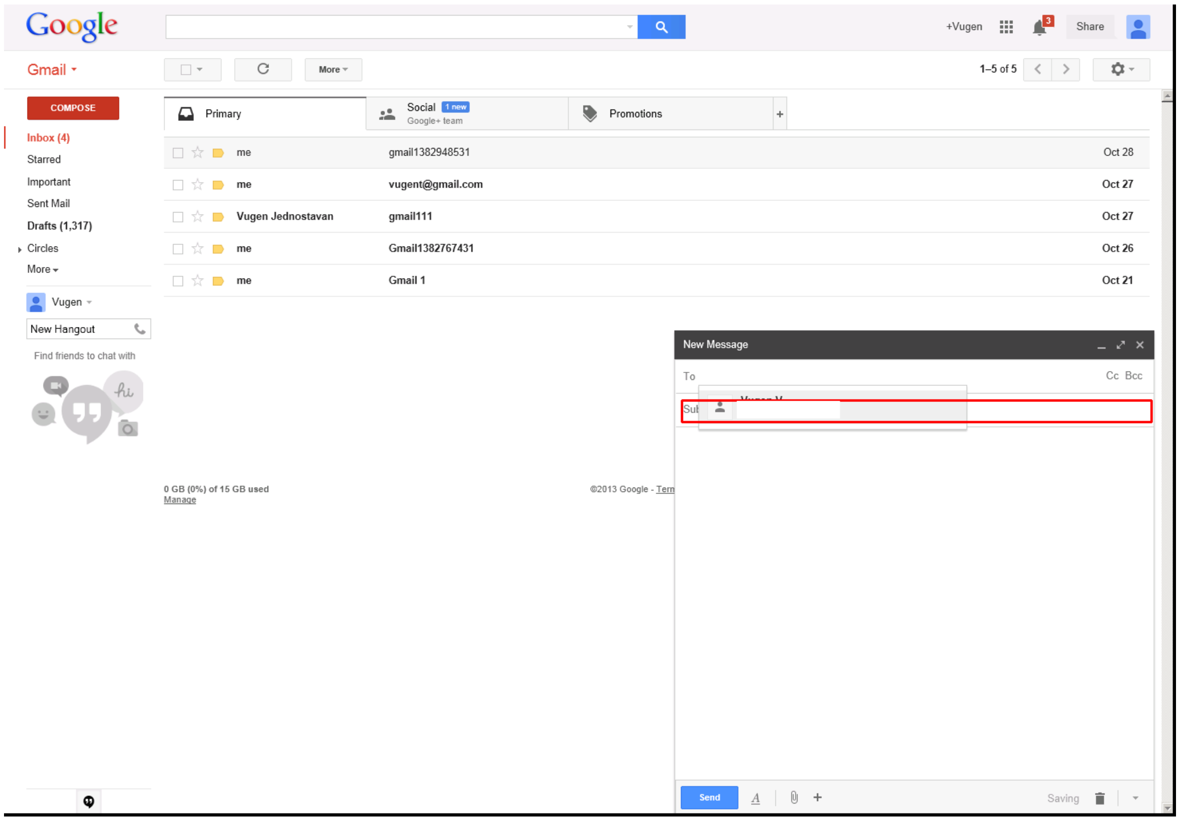
Transactions

This is a very important step. By calling this function, we create a Subject value.

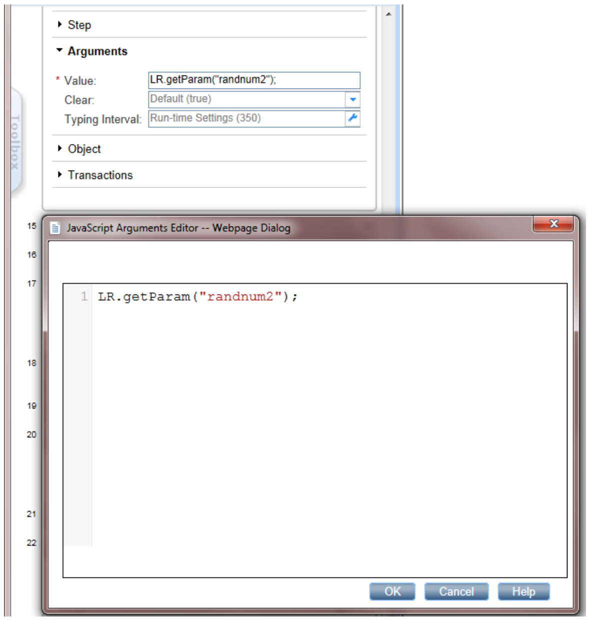
The *getRand()* function is defined in C-function.c file:

```
void getRand(){

typedef long time_t;

time_t t;

lr_param_sprintf("randnum2","bpm%d",time(&t));

}
```

Type LR.getParam('randnum2'); in Subject Field



me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21



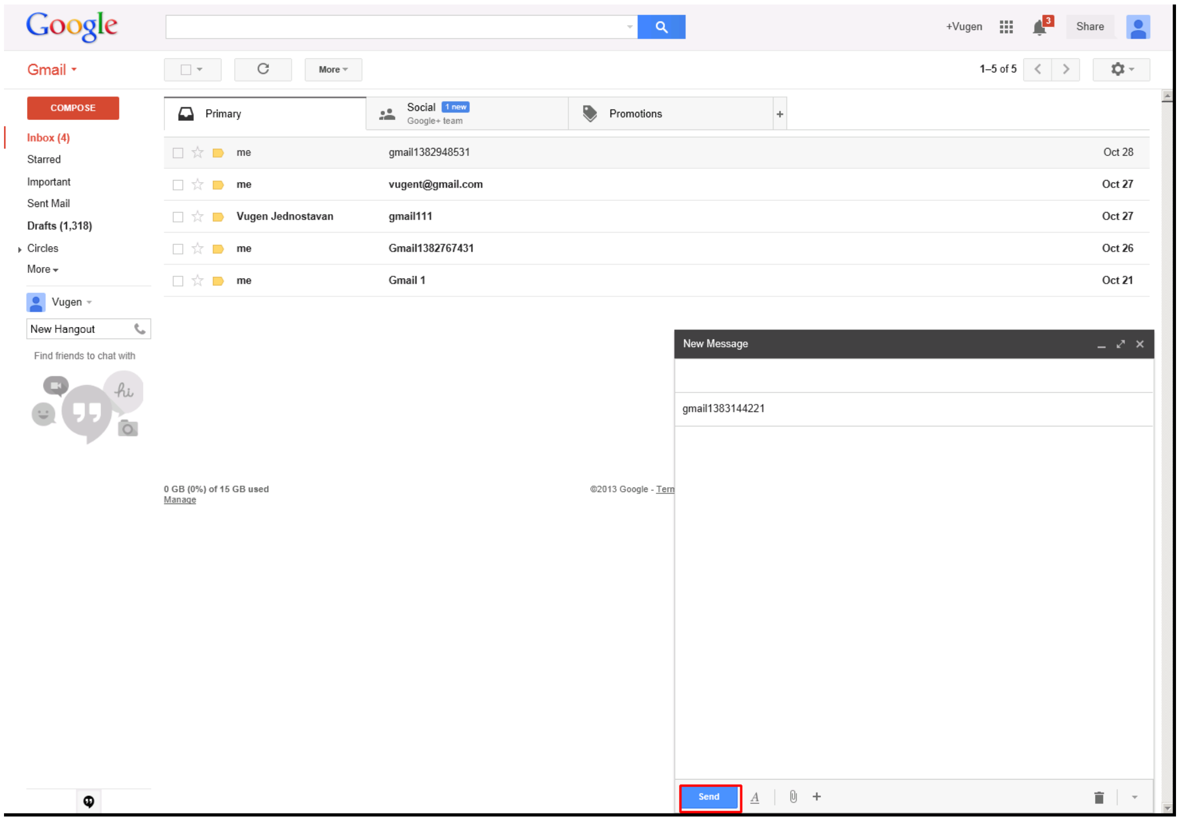
Value: LR.getParam("randnum2");
Clear: Default (true)
Typing Interval: Run-time Settings (350)

```
LR.getParam("randnum2");
```

In this step, we acquire the value from the parameter:

```
LR.getParam("randnum2");
```

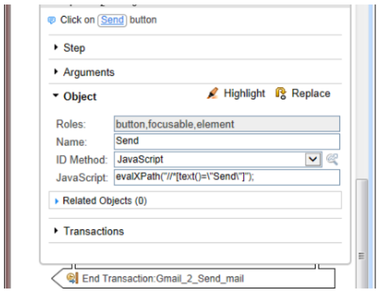
Click Send



me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21

New Message

gmail1383144221



Click on Send button

Object

Roles:	button,focusable,element
Name:	Send
ID Method:	JavaScript
JavaScript:	evalXPath("//*[text()=\"Send\"]");

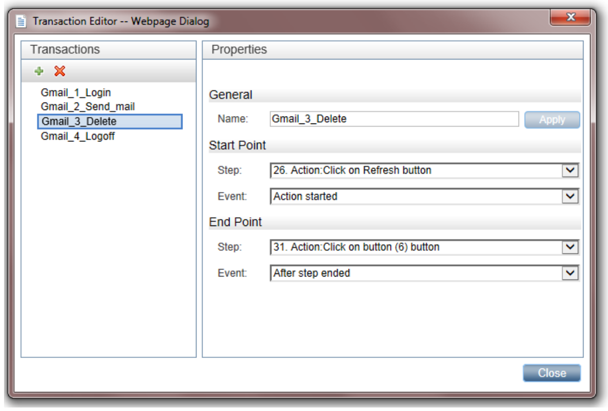
End Transaction:Gmail_2_Send_mail

Click **Send**. The **Send** button is easy to recognize so there was no need for any additional corrections.

The ID Method is JavaScript:

```
evalXPath("//*[text()=\"Send\"]");
```

Transaction — Gmail_3_Delete



Gmail_1_Login
Gmail_2_Send_mail
Gmail_3_Delete
Gmail_4_Logoff

General
Name: Gmail_3_Delete

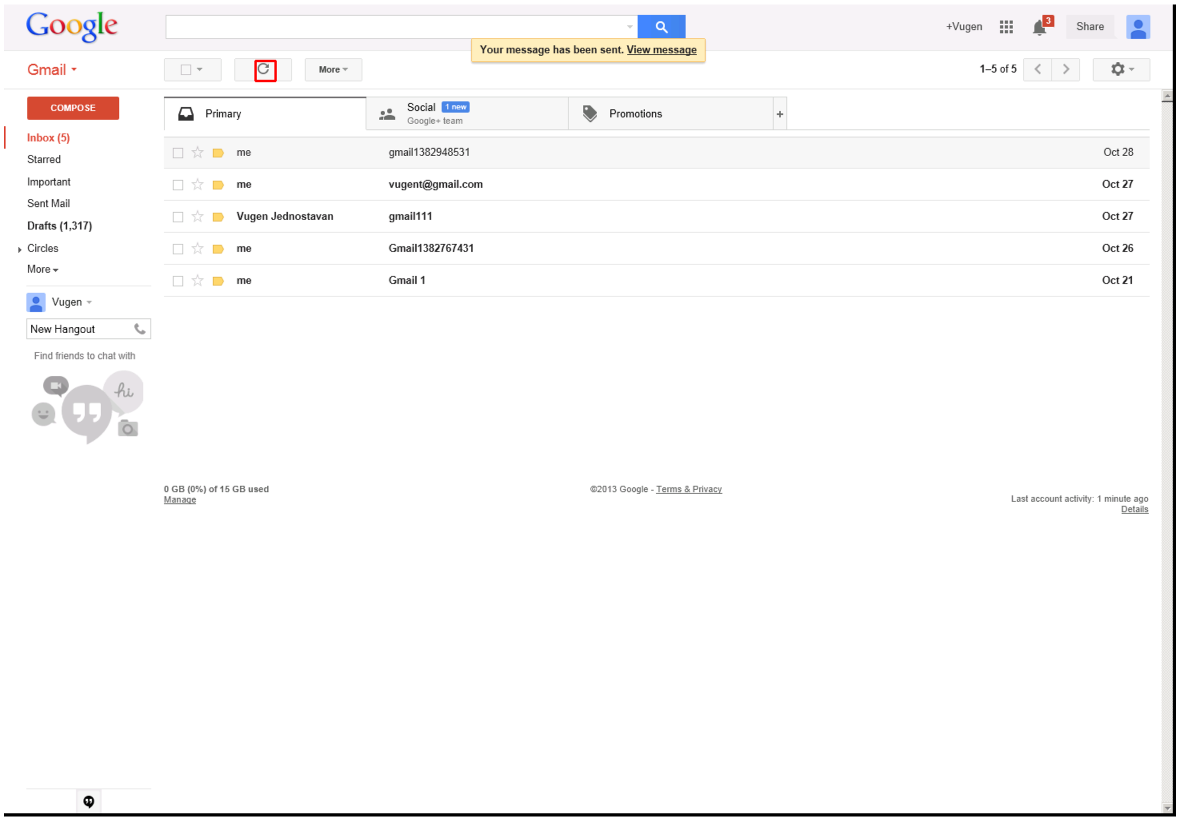
Start Point
Step: 26. Action:Click on Refresh button
Event: Action started

End Point
Step: 31. Action:Click on button (6) button
Event: After step ended

The name of this transaction is “Gmail_3_Delete”.

This transaction contains all the steps necessary for checking and deleting an email that was sent in the current interaction.

Click Refresh



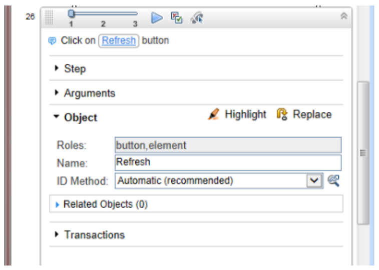
Your message has been sent. View message

me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21

0 GB (0%) of 15 GB used
Manage

©2013 Google - Terms & Privacy

Last account activity: 1 minute ago
Details


26

Click on Refresh button

Step

Arguments

Object

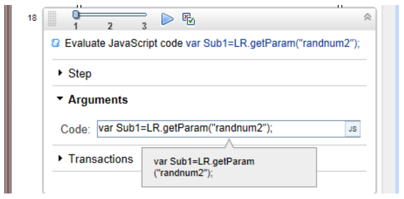
Roles: button,element
Name: Refresh
ID Method: Automatic (recommended)

Related Objects (0)

Transactions

If the email still has not arrived, repeat the previous step (**Click Settings**) and this step (**Click Refresh**).

Evaluate JavaScript code var Sub=LR.getParam ('randnum2')


18

Evaluate JavaScript code var Sub1=LR.getParam("randnum2");

Step

Arguments

Code: var Sub1=LR.getParam("randnum2");

Transactions

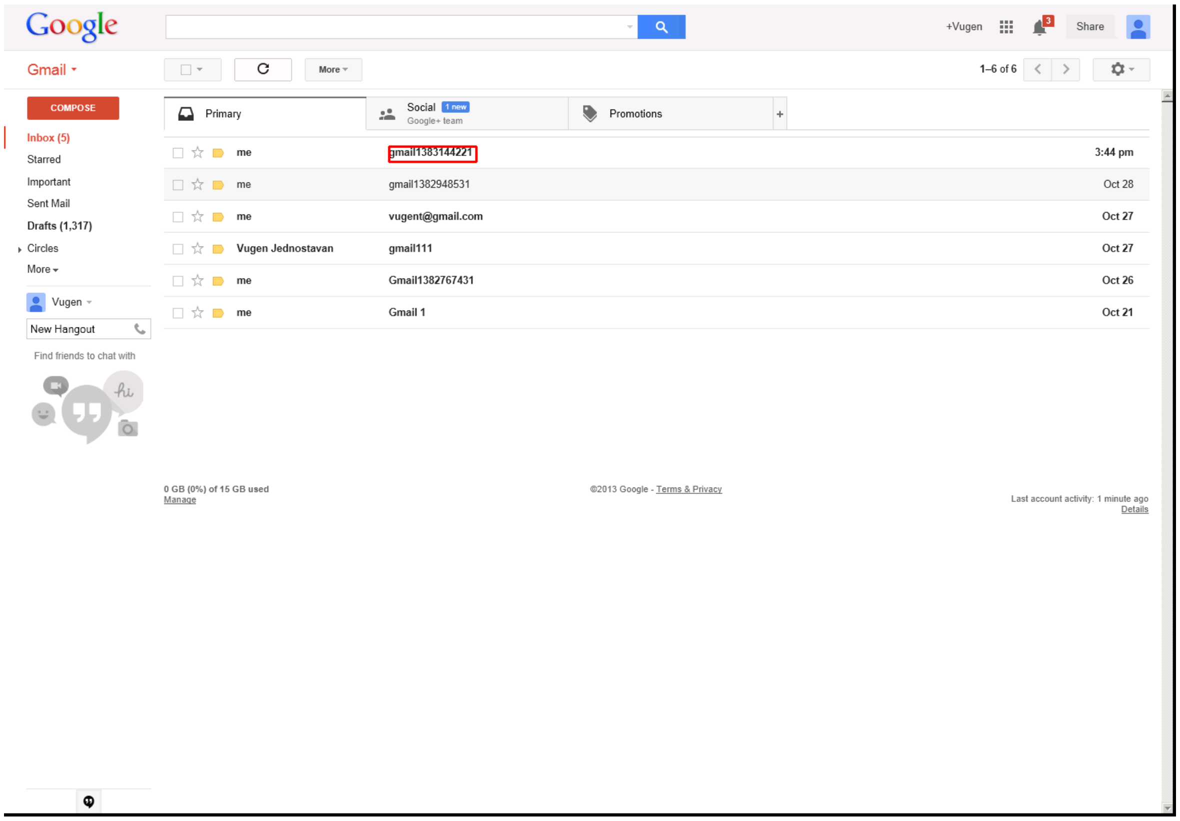
var Sub1=LR.getParam ("randnum2");

Create the Sub1 variable and get its value from the parameter *randum2*:

```
var Sub=LR.getParam("randnum2");
```

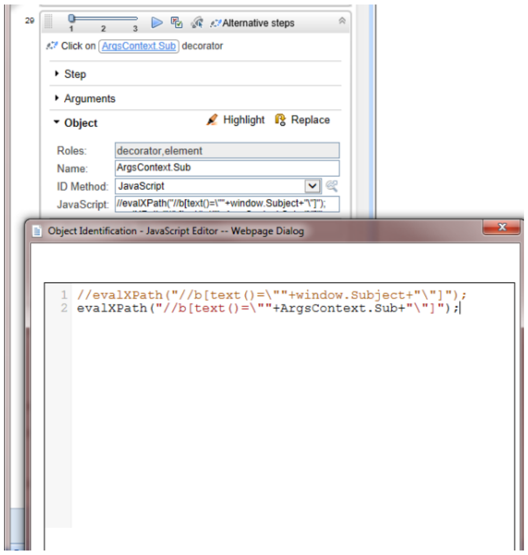
This variable is used to check (identify) and delete a specific account. The life span of this variable is within the page. When you change the page, you must re evaluate this JavaScript if you need a value from the *randum2* parameter.

Click ArgsContext.Sub Decorator



me	gmail1383144221	3:44 pm
me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21

0 GB (0%) of 15 GB used
Manage
©2013 Google - Terms & Privacy
Last account activity: 1 minute ago
Details



Click on ArgsContext.Sub decorator

Roles: decorator,element
Name: ArgsContext.Sub
ID Method: JavaScript
JavaScript: //evalXPath("//b[text()=\""+window.Subject+"\"]");

```
//evalXPath("//b[text()=\""+window.Subject+"\"]");
evalXPath("//b[text()=\""+ArgsContext.Sub+"\"]");
```

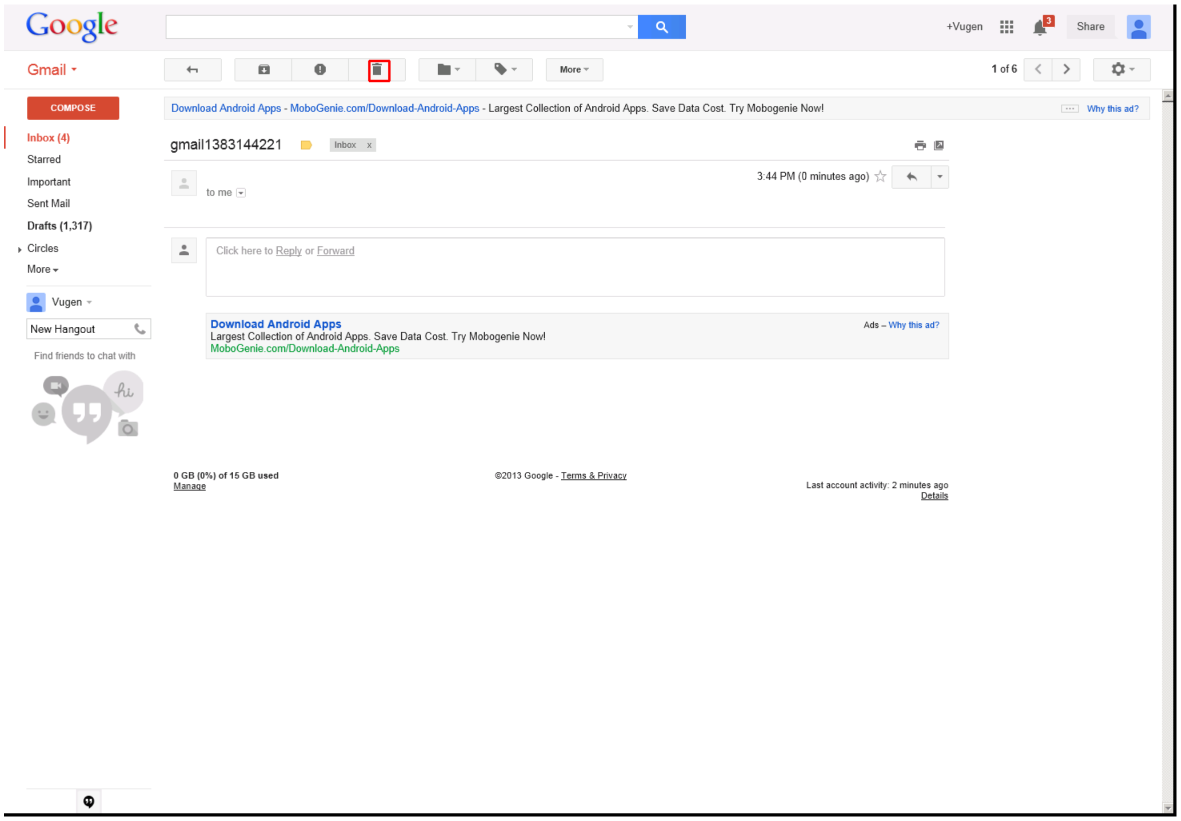
To identify an object, the name must be set as ArgsContext.Sub.

The ID Method is JavaScript:

```
evalXPath("//span[text()=\""+ArgsContext.Sub+"\"]");
```

The specific email is found.

Click Button (6)



Download Android Apps - MoboGenie.com/Download-Android-Apps - Largest Collection of Android Apps. Save Data Cost. Try Mobogenie Now!

gmail1383144221

3:44 PM (0 minutes ago)

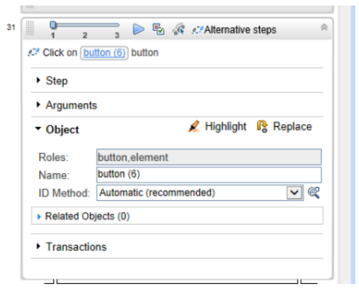
Click here to Reply or Forward

Download Android Apps
Largest Collection of Android Apps. Save Data Cost. Try Mobogenie Now!
MoboGenie.com/Download-Android-Apps

0 GB (0%) of 15 GB used
Manage

©2013 Google - Terms & Privacy

Last account activity: 2 minutes ago
Details



Click on button (6) button

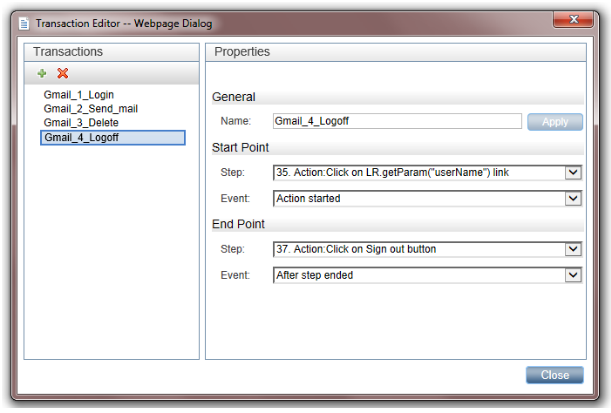
Object

Roles: button,element
Name: button (6)
ID Method: Automatic (recommended)

The **Delete** button (button(6)) is easy to recognize.

This step completes Gmail_3_Delete mail.

Transaction — Gmail_4_Logoff



Gmail_1_Login
Gmail_2_Send_mail
Gmail_3_Delete
Gmail_4_Logoff

General
Name: Gmail_4_Logoff

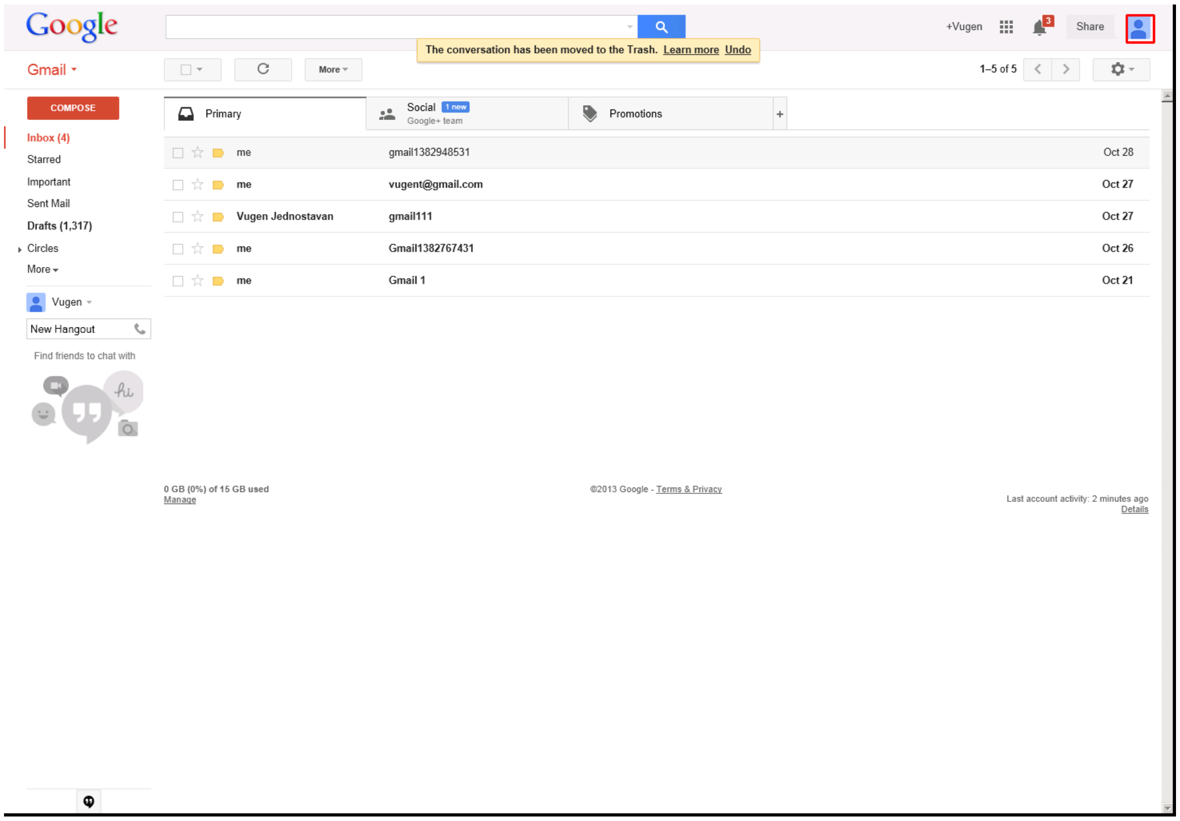
Start Point
Step: 35. Action:Click on LR.getParam("userName") link
Event: Action started

End Point
Step: 37. Action:Click on Sign out button
Event: After step ended

The name of this transaction is “Gmail_4_Logoff”.

This transaction contains all the steps necessary for logging off.

Click LR.getParam('userName')



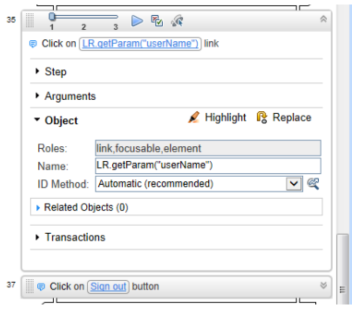
The conversation has been moved to the Trash. Learn more Undo

me	gmail1382948531	Oct 28
me	vugent@gmail.com	Oct 27
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21

0 GB (0%) of 15 GB used
Manage

©2013 Google - Terms & Privacy

Last account activity: 2 minutes ago
Details


35 Click on LR.getParam("userName") link

- Step
- Arguments
- Object

Roles: link,focusable,element
Name: LR.getParam("userName")
ID Method: Automatic (recommended)

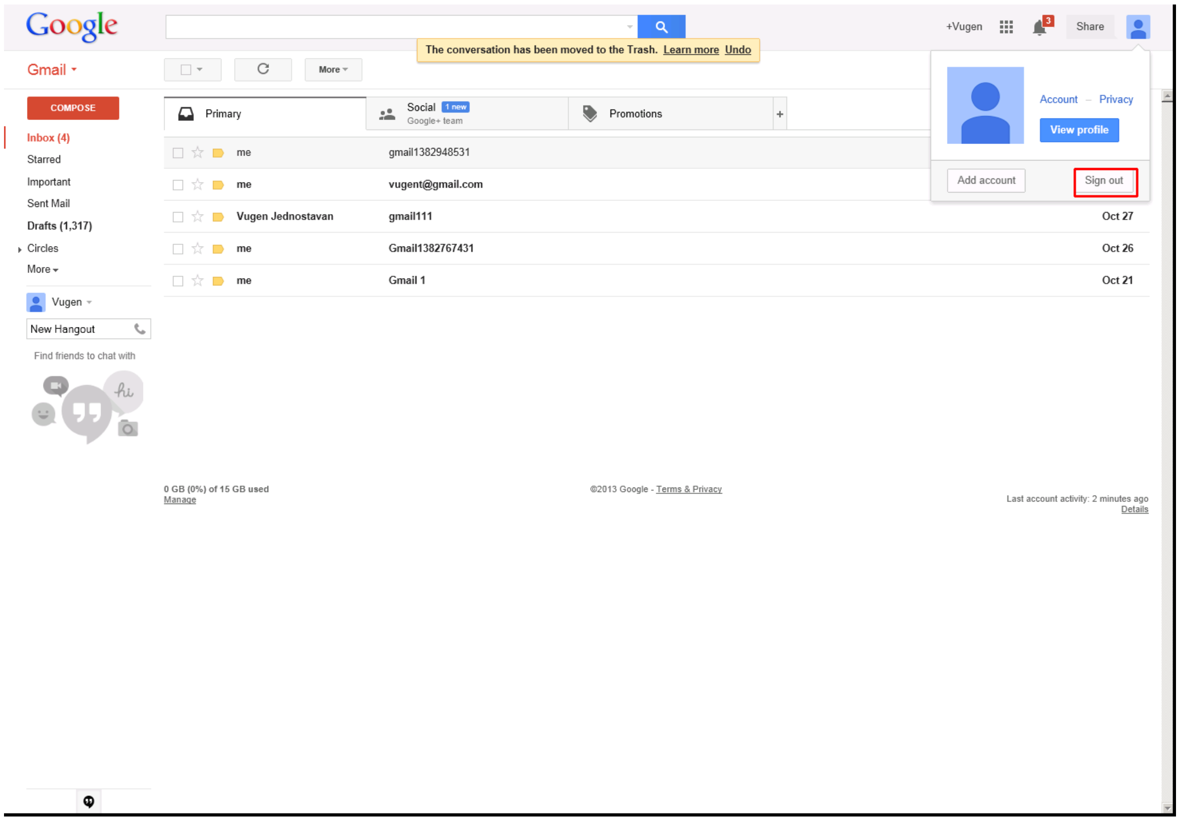
- Related Objects (0)
- Transactions

37 Click on Sign out button

We used the same parameter as we did for the Login:

```
LR.getParam("userName")
```

Click Sign Out

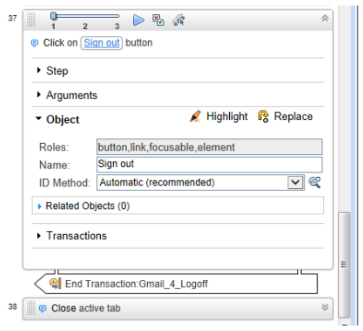

The conversation has been moved to the Trash. Learn more Undo

me	gmail1382948531	
me	vugent@gmail.com	
Vugen Jednostavan	gmail111	Oct 27
me	Gmail1382767431	Oct 26
me	Gmail 1	Oct 21

0 GB (0%) of 15 GB used
Manage

©2013 Google - Terms & Privacy

Last account activity: 2 minutes ago
Details


37
Click on Sign out button
Step
Arguments
Object

Roles: button,link,focusable,element
Name: Sign out
ID Method: Automatic (recommended)
Related Objects (0)
Transactions
End Transaction:Gmail_4_Logoff
38
Close active tab

By clicking **Sign out**, we end this transaction and script.

All we need to do now is close the active tab.

Chapter 5: Cloud CRM Provider – Salesforce Transaction Flow

This case study uses Salesforce cloud service. The case study uses one flow/script with the following transactions:

1. **Login** – Open a login page, enter your credentials, and confirm that the first page loads.
2. **Create item (account)** – Open a new account with a unique account name.
3. **Delete item (account)** – Check for this account, select it and delete it.
4. **Logoff** – Log off from the web application.

The main purpose of running this script is to confirm that the application's processes run successfully. If successful, the transactions complete without any errors.

You can view a sample Salesforce script by downloading the following file https://softwaresupport.hpe.com/group/softwaresupport/search-result/-/facetsearch/document/KM00658289.

General Flow

Salesforce is a cloud CRM (Customer Relationship Management).

A sequence of transactions creates the business process flow. A transaction is a unit that is measurable by availability and performance. Therefore, we group activities that perform a specific transaction.

To describe and validate these transactions, logically group the transactions into the following steps:

1. Login.
   a. Navigate to https://login.salesforce.com.
   b. Enter your credentials and confirm that the first page (Inbox) loads.
2. Create a new account.
   a. Click **New Account**.
   b. Enter a unique account name.
   c. Save the account.

3. Delete the account.
    a. Check that the account was created.
    b. Select the account.
    c. Delete the account.
4. Logoff
    a. Log off from the web application.

The following sections describe how to create these transactions.

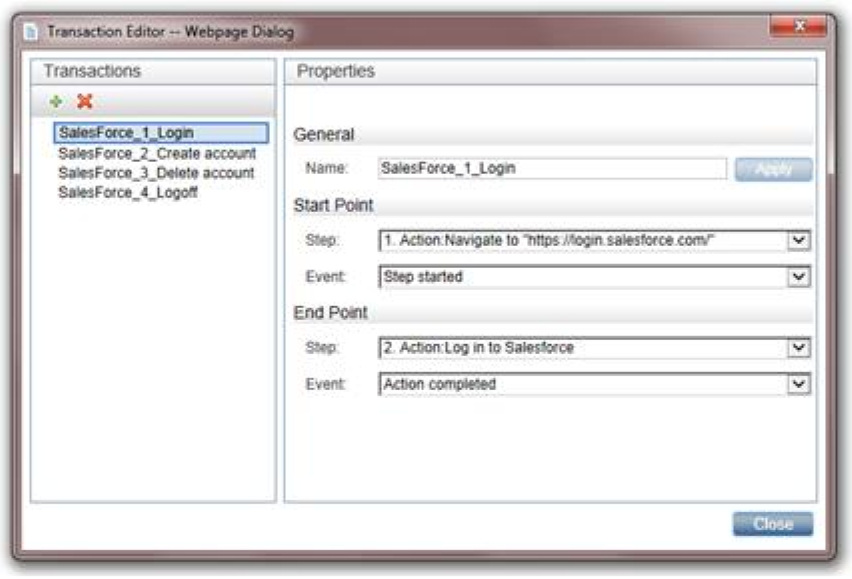
A good approach, before using this tool, is to know what we want to achieve. Create a blueprint as follows:

1. Run the Salesforce web application in Internet Explorer 9. Perform the steps of the flow and create annotations.
2. Determine the boundaries that form the transactions.
3. Select which values to assign to the parameters.
4. Select what values to protect and encrypt.

Record all the information you collect since it will come in handy later.

We recommend you start using the script. Refer to sections in this document for assistance.

Transaction — SalesForce_1_Login



SalesForce_1_Login
SalesForce_2_Create account
SalesForce_3_Delete account
SalesForce_4_Logoff

General
Name: SalesForce_1_Login

Start Point
Step: 1. Action:Navigate to "https://login.salesforce.com/"
Event: Step started

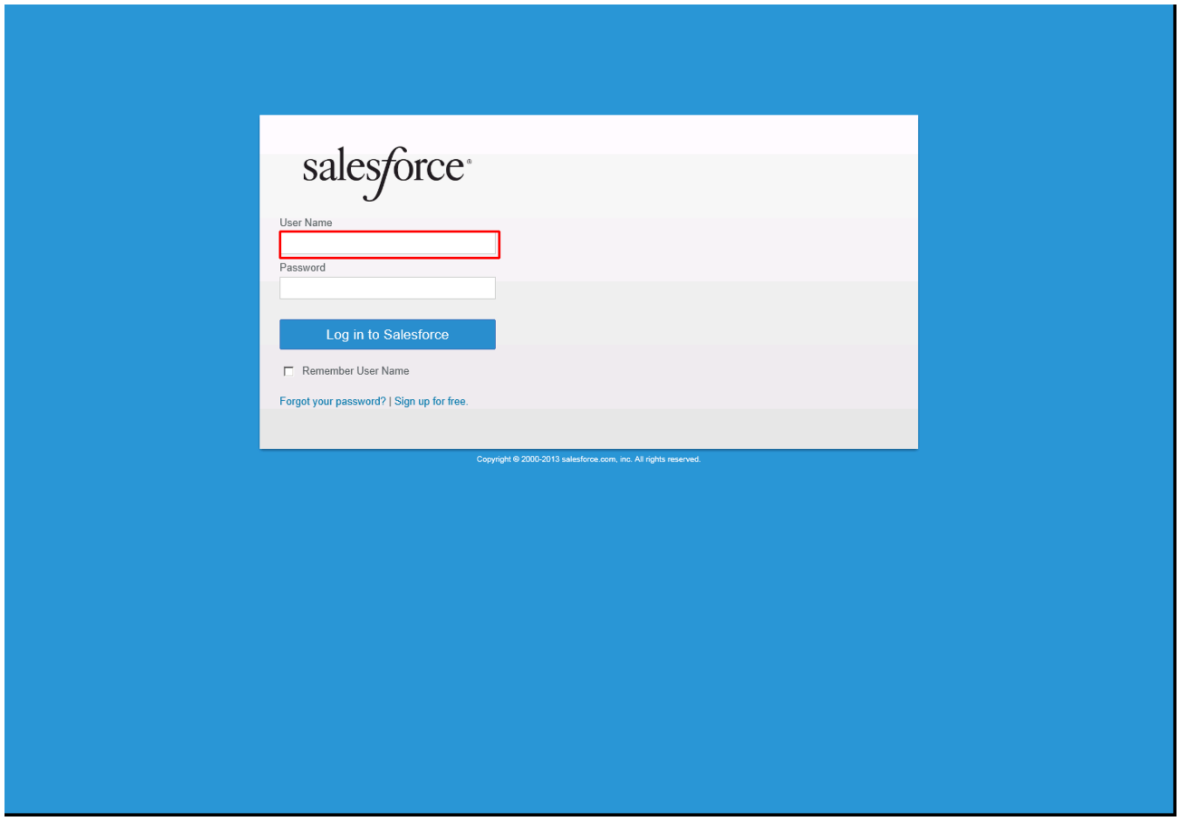
End Point
Step: 2. Action:Log in to Salesforce
Event: Action completed

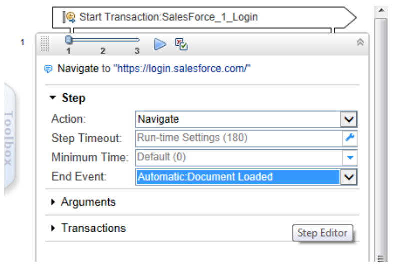
Define the name of the transaction. In this example, we call the transaction “SalesForce_1_Login”. Naming conventions should provide a clear view of the application, the transaction order, and create an overview of the business processes covered by this transaction.

Define the steps and events for the starting and ending points of the transaction. The most convenient event for the start step is **Action started**. The most convenient event for the end step is **After step ended**.

This transaction contains all the steps necessary for logging into the Salesforce application.

Navigate to 'https://login.salesforce.com/'


salesforce
User Name
Password
Log in to Salesforce
Remember User Name
Forgot your password? | Sign up for free.
Copyright © 2000-2013 salesforce.com, inc. All rights reserved.



Start Transaction:SalesForce_1_Login

Navigate to "https://login.salesforce.com/"

Step

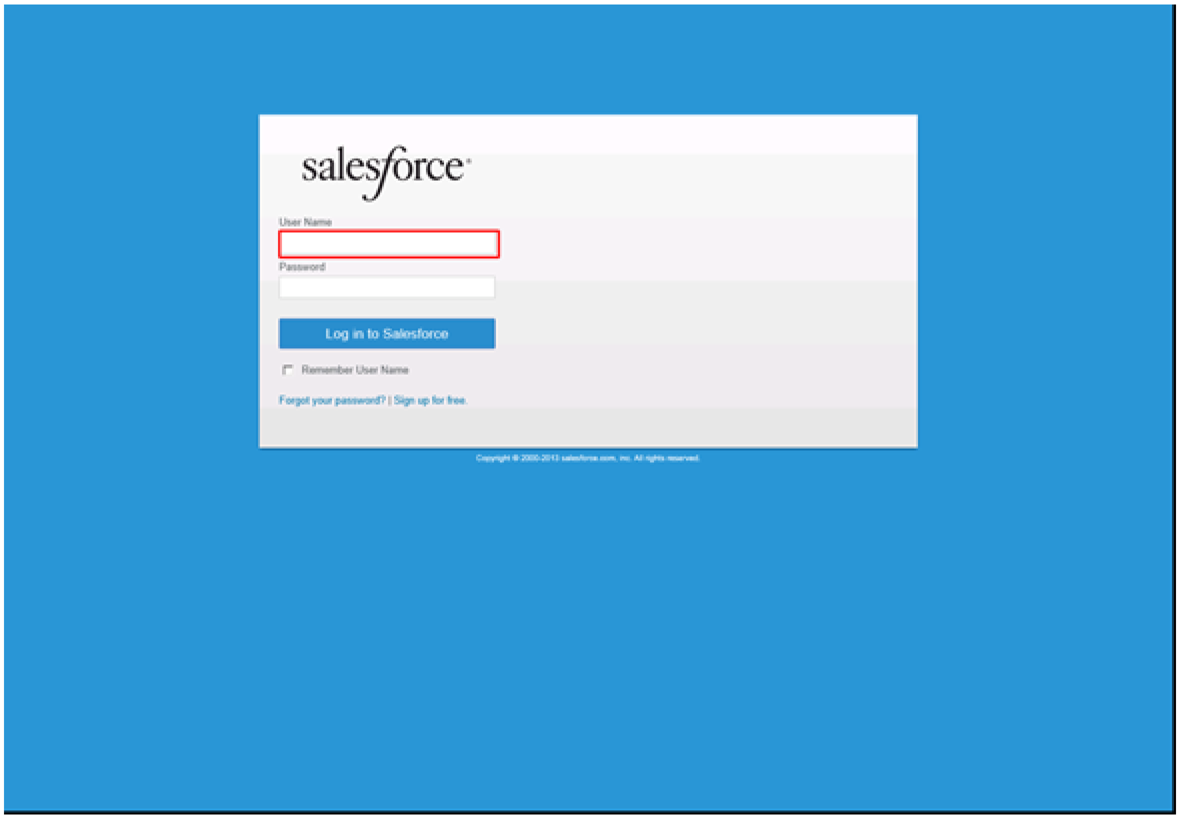
Action:	Navigate
Step Timeout:	Run-time Settings (180)
Minimum Time:	Default (0)
End Event:	Automatic:Document Loaded

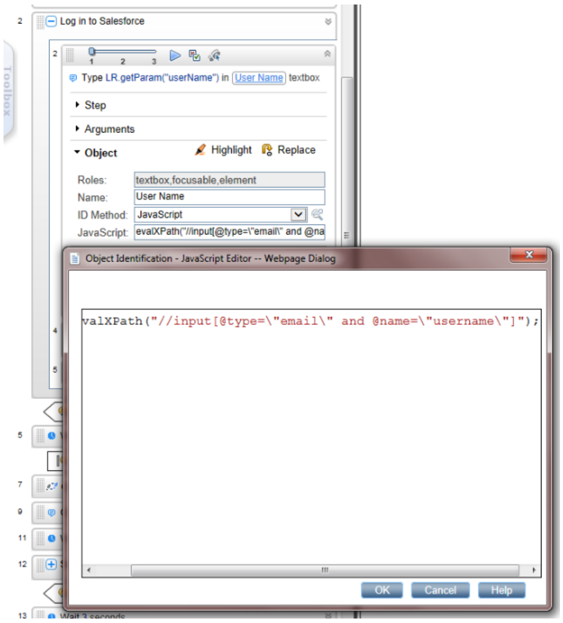
In the Navigation step, specify the URL.

While recording, VuGen automatically selects the appropriate End Event, in this case **Document downloaded**. The step ends when the process of loading a document completes. In most cases, you can freely select the default values.

If the pages load slowly, you can use a Wait step to make sure the page loads. For information on Wait steps, see *Meaning of Wait Step* on page 94.

Type LR.getParam('userName') in User Name Textbox


salesforce
User Name
Password
Log in to Salesforce
Remember User Name
Forgot your password? | Sign up for free.
Copyright © 2000-2013 salesforce.com, inc. All rights reserved.



Type LR.getParam("userName") in User Name textbox

Roles: textbox,focusable,element
Name: User Name
ID Method: JavaScript
JavaScript: evalXPath("//input[@type=\"email\" and @na

```
valXPath("//input[@type=\"email\" and @name=\"username\"]");
```

This step is actually a sub-step of Log in to SalesForce.

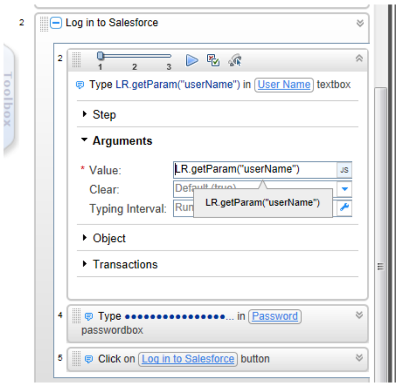
You can either select the **Automatic** or **JavaScript** ID method.

```
evalXPath("//input[@type=\"email\" and @name=\"username\"]");
```

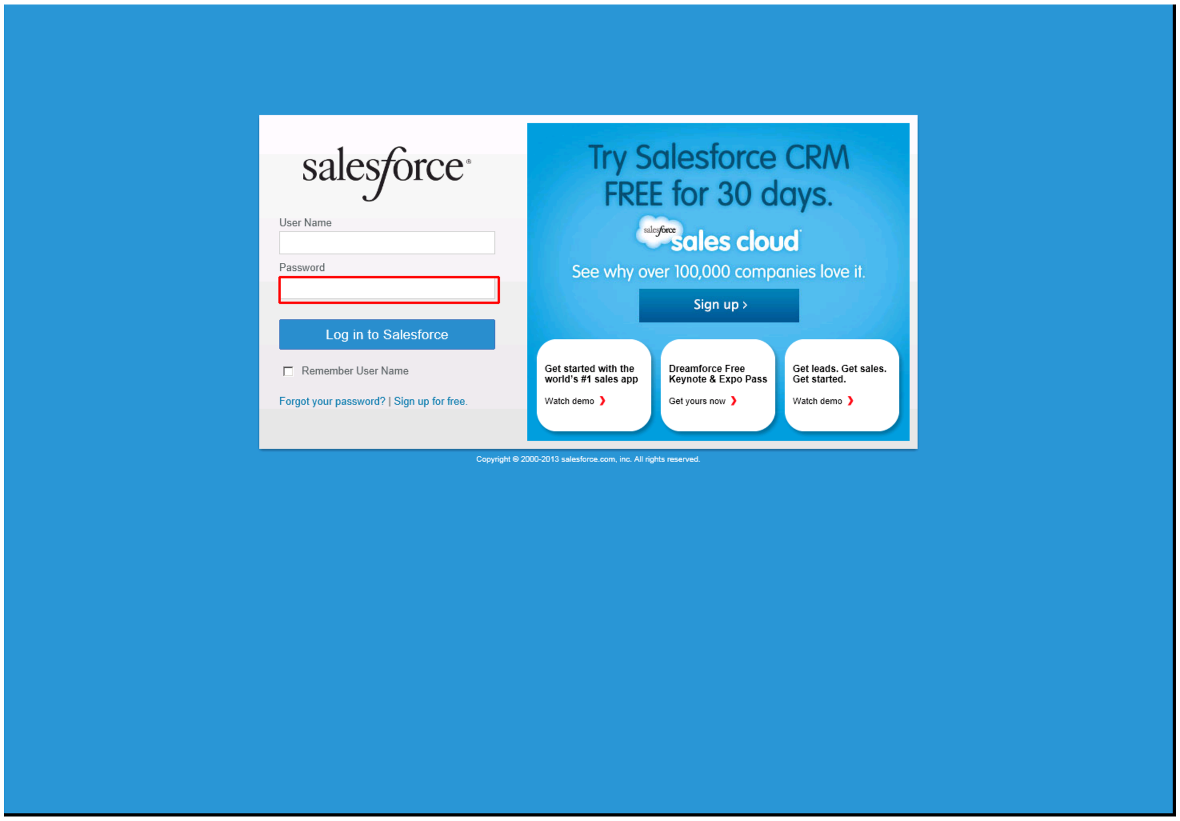
There are cases when you will change the ID Method, as we will see later.

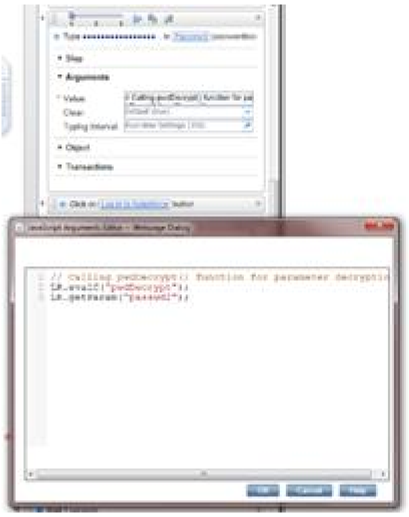
Type your username in the **User Name** textbox.

Since we are using parameters, you can enter the *LR.getParam("userName");* command in the **Value** field and the result is the defined username.


Log in to Salesforce
Type LR.getParam("userName") in User Name textbox
Step
Arguments
Value: LR.getParam("userName")
Clear: Default (true)
Typing Interval: Run
LR.getParam("userName")
Object
Transactions
Type ●●●●●●●●●●●●●●●●... in Password passwordbox
Click on Log in to Salesforce button

Type *************** in Password Textbox


salesforce
User Name
Password
Log in to Salesforce
Remember User Name
Forgot your password? | Sign up for free.
Try Salesforce CRM FREE for 30 days.
sales cloud
See why over 100,000 companies love it.
Sign up ›
Get started with the world's #1 sales app
Watch demo
Dreamforce Free Keynote & Expo Pass
Get yours now
Get leads. Get sales. Get started.
Watch demo
Copyright © 2000-2013 salesforce.com, inc. All rights reserved.



```
// Calling pwdDecrypt() function for parameter decryptio
LR.evalC("pwdDecrypt");
LR.getParam("passwd2");
```

Type a password in the **Password** textbox. To display an encrypted password (so no one can see it), type the following:

```
// Calling pwdDecrypt() function for parameter decryption

LR.evalC("pwdDecrypt");

// This is a parameter with a decrypted value

LR.getParam("passwd2");
```

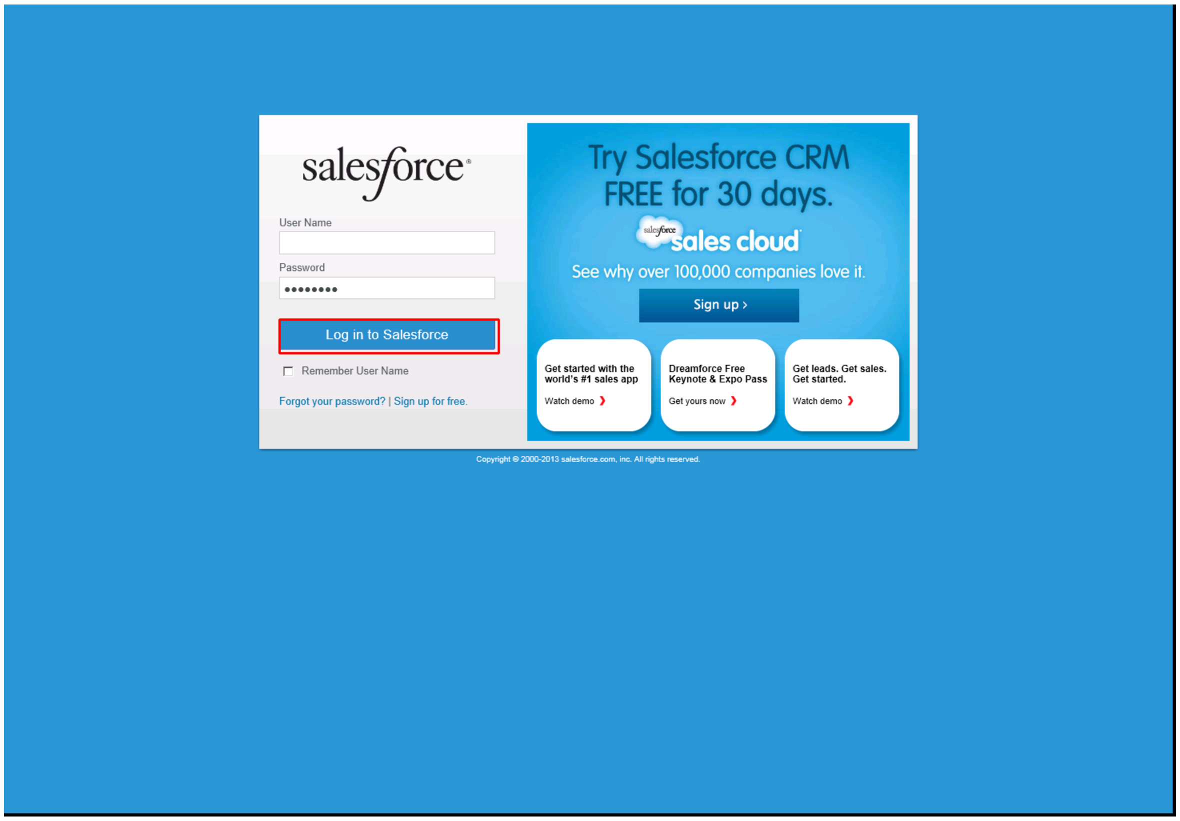
The *pwdDecrypt()* function is defined in C-function.c file:

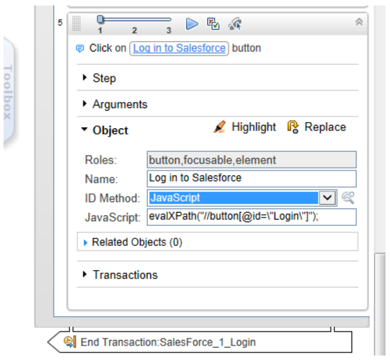
```
void pwdDecrypt(){

lr_save_string(lr_decrypt(lr_eval_string("{passwd}")),"passwd2");

/*If you need to get some value from external parameter file*/

lr_output_message(lr_eval_string("passwd2"));

}
```

`passwd` is the parameter defined for external use (BSM – EUM) and is encrypted.

`passwd2` is the parameter seen only within this script instance.

Click Log in to Salesforce


salesforce
User Name
Password
••••••••
Log in to Salesforce
Remember User Name
Forgot your password? | Sign up for free.
Try Salesforce CRM FREE for 30 days.
sales cloud
See why over 100,000 companies love it.
Sign up ›
Get started with the world's #1 sales app
Watch demo
Dreamforce Free Keynote & Expo Pass
Get yours now
Get leads. Get sales. Get started.
Watch demo
Copyright © 2000-2013 salesforce.com, inc. All rights reserved.



Roles: button,focusable,element
Name: Log in to Salesforce
ID Method: JavaScript
JavaScript: evalXPath("//button[@id=\"Login\"]");

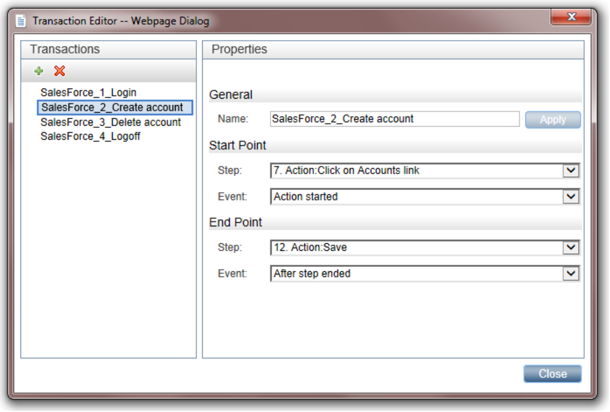
This is defined as the End step for transaction “SalesForce_1_Login”.

The JavaScript ID Method used is:

```
evalXPath("//button[@id=\"Login\"]");
```

The End Event Step is set to automatic, and in this case is **Step Synchronous network completed**. The step ends when all HTTP requests have completed excluding requests initiated by XMLHttpRequest.

Transaction — SalesForce_2_Create account


Transaction Editor -- Webpage Dialog

Transactions
SalesForce_1_Login
SalesForce_2_Create account
SalesForce_3_Delete account
SalesForce_4_Logoff

Properties

General
Name: SalesForce_2_Create account

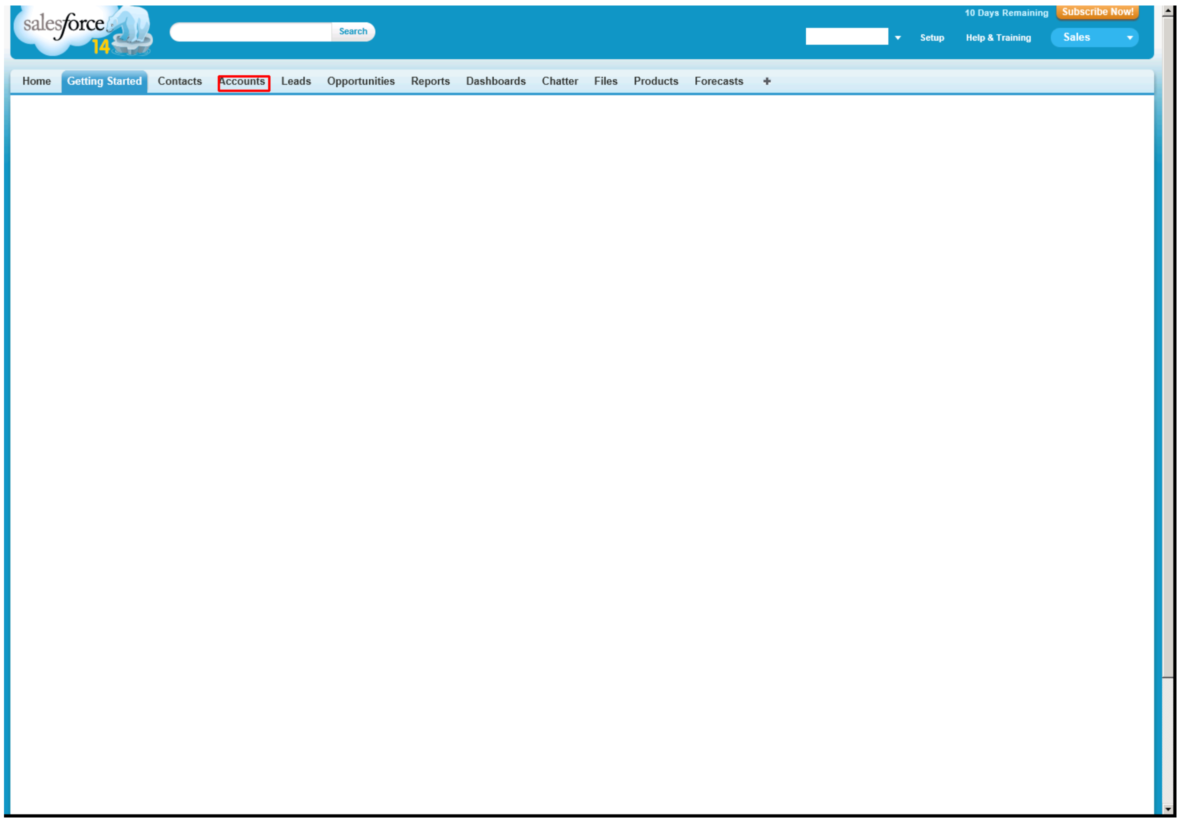
Start Point
Step: 7. Action:Click on Accounts link
Event: Action started

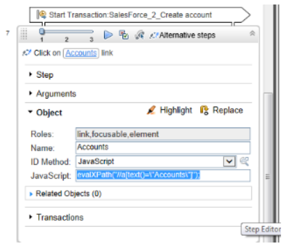
End Point
Step: 12. Action:Save
Event: After step ended

The name of this transaction is “SalesFoerce_2_Create account”.

This transaction contains all the steps necessary for creating an account.

Click Accounts




Start Transaction:SalesForce_2_Create account

Click on Accounts link

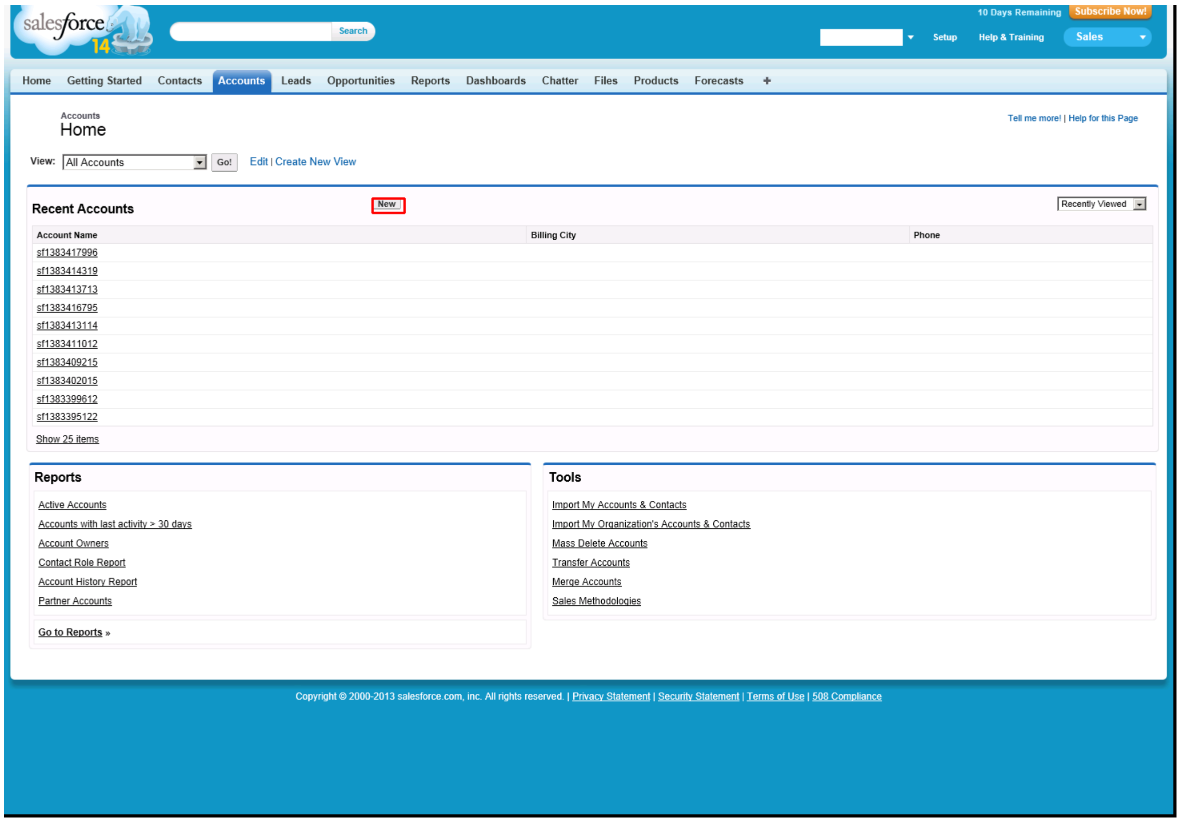
Roles:	link,focusable,element
Name:	Accounts
ID Method:	JavaScript
JavaScript:	evalXPath("//a[text()=\"Accounts\"]");

When you click **Account**, a form appears for listing accounts. By running the script multiple times, we can confirm that this element is recognized with the Automatic ID method.

You can also use the JavaScript ID method:

```
evalXPath("//a[text()=\"Accounts\"]");
```

Click New



Accounts
Home

View: All Accounts

Recent Accounts

Account Name	Billing City	Phone
sf1383417996		
sf1383414319		
sf1383413713		
sf1383416795		
sf1383413114		
sf1383411012		
sf1383409215		
sf1383402015		
sf1383399612		
sf1383395122		

Show 25 items

Reports

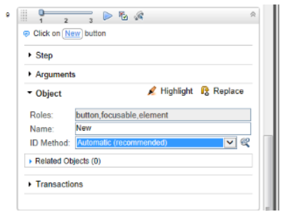
- Active Accounts
- Accounts with last activity > 30 days
- Account Owners
- Contact Role Report
- Account History Report
- Partner Accounts

Go to Reports »

Tools

- Import My Accounts & Contacts
- Import My Organization's Accounts & Contacts
- Mass Delete Accounts
- Transfer Accounts
- Merge Accounts
- Sales Methodologies

Copyright © 2000-2013 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance


Click on New button

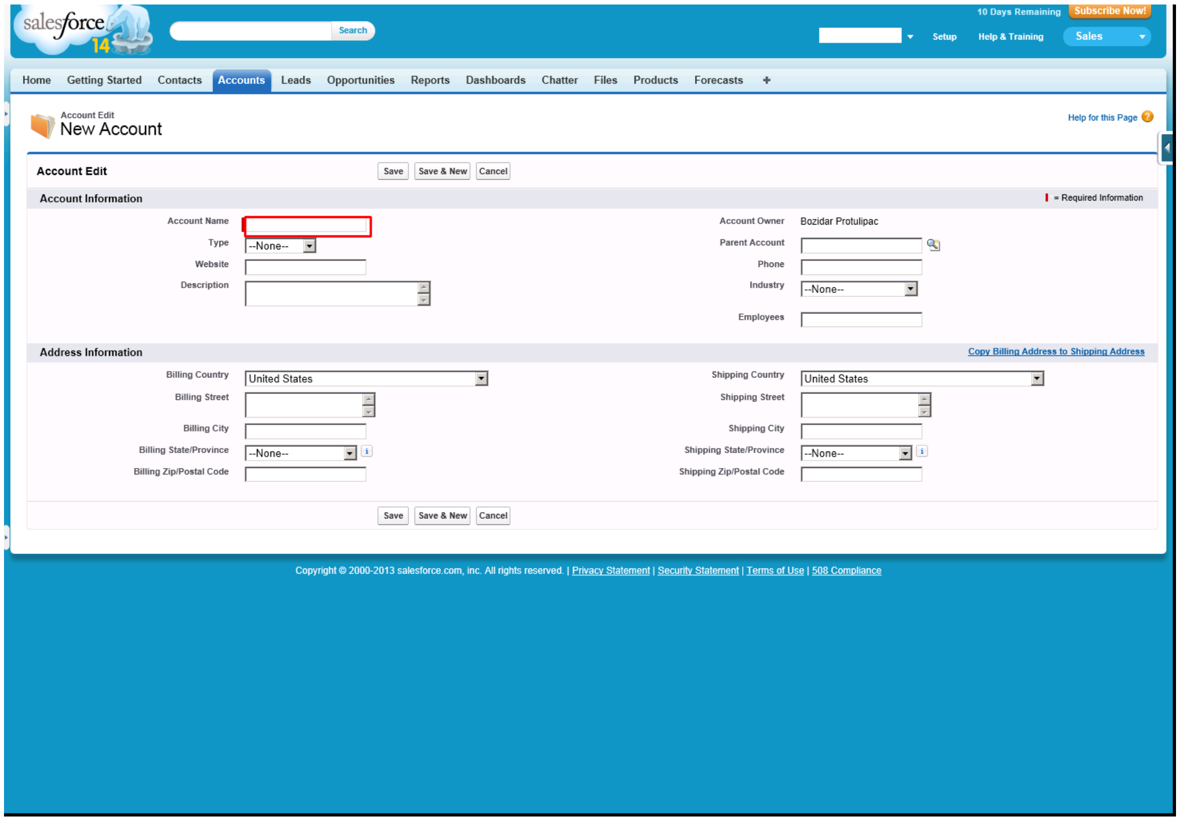
Roles:	button,focusable,element
Name:	New
ID Method:	Automatic (recommended)

When you click **New**, a form appears for creating a new account. By running the script multiple times we can confirm that this element is recognized with the Automatic ID method.

You can also use the JavaScript ID method:

```
evalXPath("//input[@type=\"button\" and @name=\"new\" and @value=\" New \"]");
```

Click Account Name Textbox



Account Edit
New Account

Account Edit

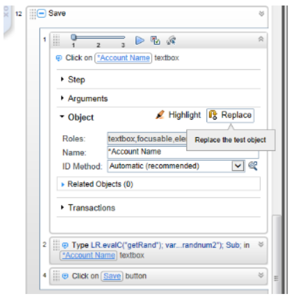
Account Information = Required Information

Account Name		Account Owner	Bozidar Protulipac
Type	--None--	Parent Account	
Website		Phone	
Description		Industry	--None--
		Employees	

Address Information — Copy Billing Address to Shipping Address

Billing Country	United States	Shipping Country	United States
Billing Street		Shipping Street	
Billing City		Shipping City	
Billing State/Province	--None--	Shipping State/Province	--None--
Billing Zip/Postal Code		Shipping Zip/Postal Code	

Copyright © 2000-2013 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance

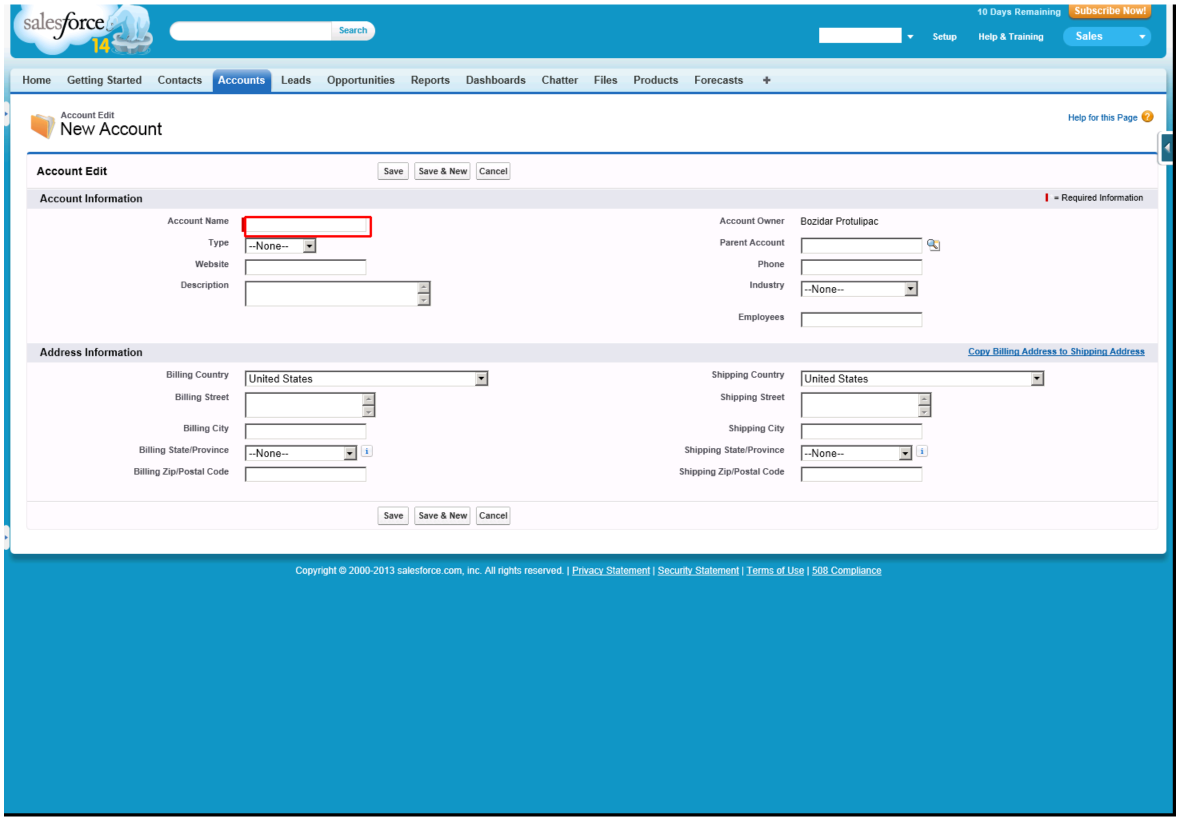


12 Save
1 Click on *Account Name textbox
Roles: textbox,focusable,ele
Name: *Account Name
ID Method: Automatic (recommended)
2 Type LR.evalC("getRand"); var...randnum2"); Sub; in *Account Name textbox
4 Click on Save button

Click **Account Name** textbox.

To make this script robust, we use parameters. Therefore, if the username or password is changed at a later time, you do not need to change the script, only the parameter's value. The username is in the form of an email address.

Type LR.evalC('getRand'); var...randnum2'); Sub; in Account Name Textbox



Account Edit
New Account

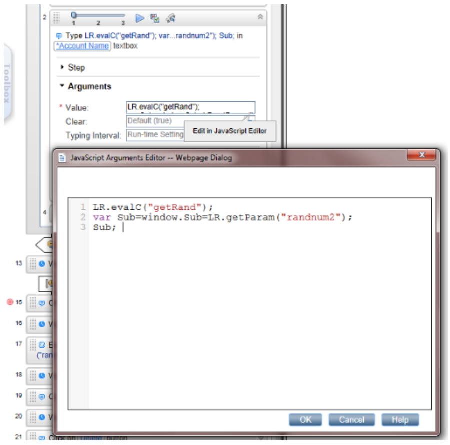
Account Edit

Account Information | = Required Information

Account Name		Account Owner	Bozidar Protulipac
Type	--None--	Parent Account	
Website		Phone	
Description		Industry	--None--
		Employees	

Address Information

Billing Country	United States	Shipping Country	United States
Billing Street		Shipping Street	
Billing City		Shipping City	
Billing State/Province	--None--	Shipping State/Province	--None--
Billing Zip/Postal Code		Shipping Zip/Postal Code	


Toolbox
Step
Arguments
Edit in JavaScript Editor
JavaScript Arguments Editor -- Webpage Dialog
OK
Cancel
Help

Type LR.evalC("getRand"); var...randnum2"); Sub; in *Account Name textbox

Value: LR.evalC("getRand");
Clear: Default (true)
Typing Interval: Run-time Setting

```
LR.evalC("getRand");
var Sub=window.Sub=LR.getParam("randnum2");
Sub;
```

By calling this function, we create a Subject value.

The getRand() function is defined in C-function.c file:

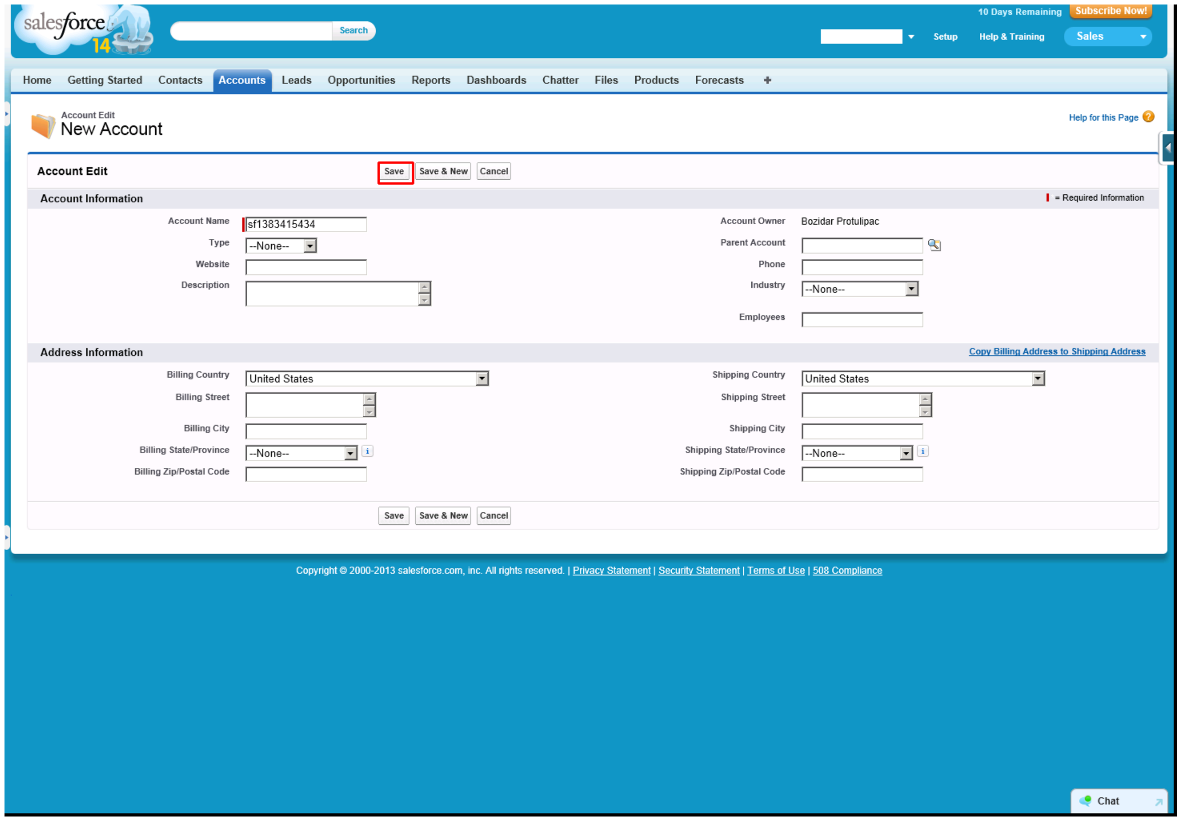
```
void getRand(){
```

```
typedef long time_t;

time_t t;

lr_param_sprintf("randnum2","sf%d",time(&t));
```

In this step, we acquire the value from the parameter:

```
LR.getParam("randnum2");
```

Click Save



Account Edit
New Account

Account Edit

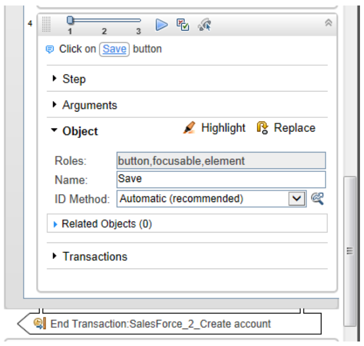
Account Information | = Required Information

Account Name	sf1383415434	Account Owner	Bozidar Protulipac
Type	--None--	Parent Account	
Website		Phone	
Description		Industry	--None--
		Employees	

Address Information

Billing Country	United States	Shipping Country	United States
Billing Street		Shipping Street	
Billing City		Shipping City	
Billing State/Province	--None--	Shipping State/Province	--None--
Billing Zip/Postal Code		Shipping Zip/Postal Code	

Copyright © 2000-2013 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance



Click on Save button

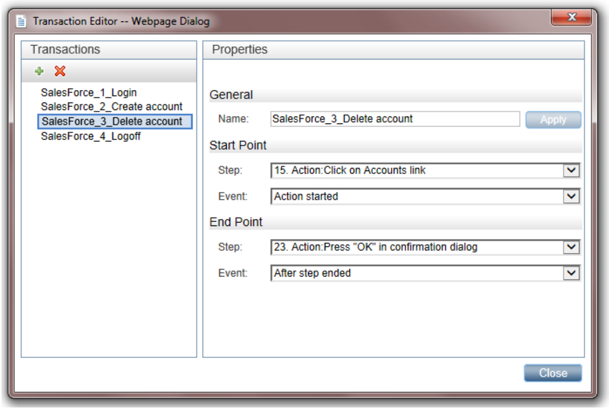
Object

Roles:	button,focusable,element
Name:	Save
ID Method:	Automatic (recommended)

End Transaction:SalesForce_2_Create account

Click **Save**. The **Save** button is easily recognized so there is no need for corrections.

Transaction — SalesForce_3_Delete account



Transaction Editor -- Webpage Dialog

Transactions
- SalesForce_1_Login
- SalesForce_2_Create account
- SalesForce_3_Delete account
- SalesForce_4_Logoff

Properties

General

Name: SalesForce_3_Delete account

Start Point

Step: 15. Action:Click on Accounts link

Event: Action started

End Point

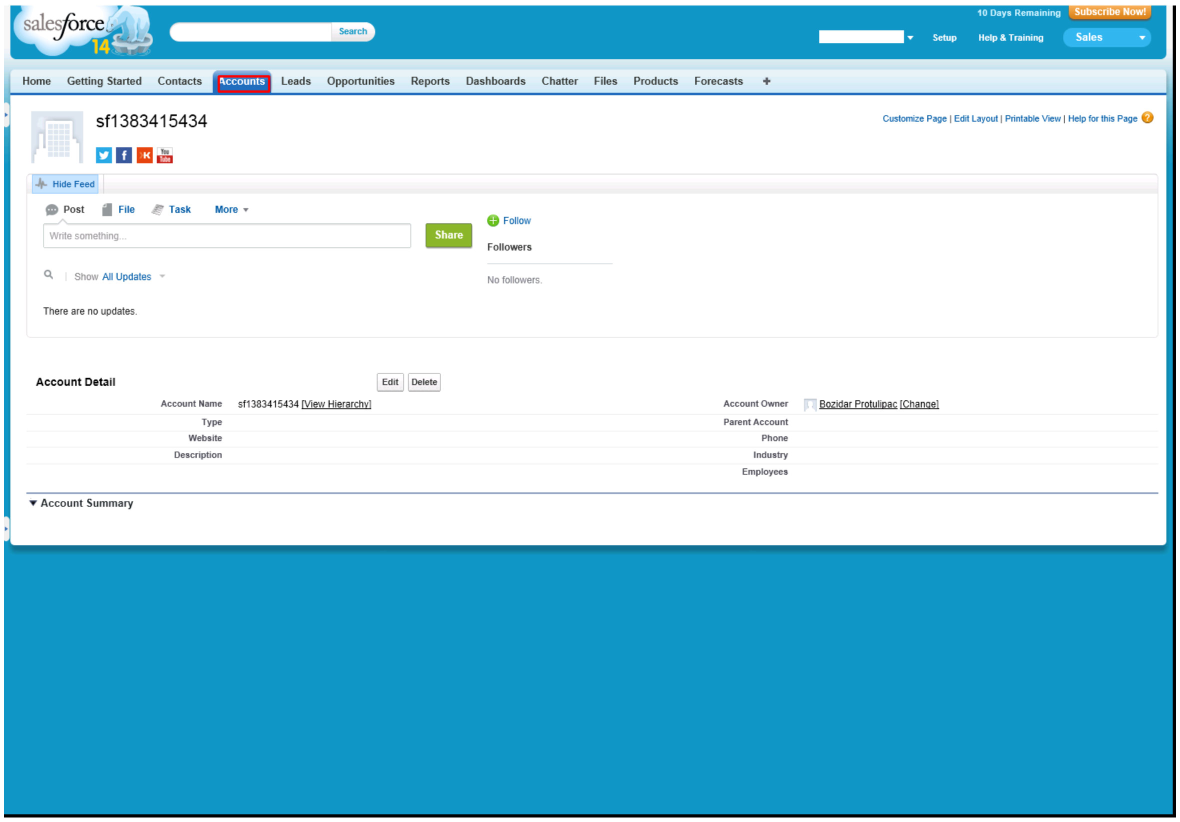
Step: 23. Action:Press "OK" in confirmation dialog

Event: After step ended

The name of this transaction is “SalesForce_3_Delete account”.

This transaction contains all the steps necessary for checking and deleting an account that was created in the current interaction.

Click Accounts



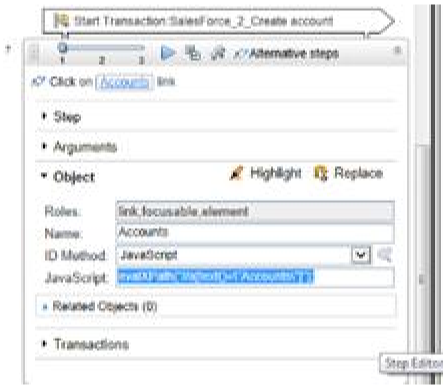
sf1383415434

No followers.

There are no updates.

Account Detail

Account Name	sf1383415434 [View Hierarchy]	Account Owner	Bozidar Protulipac [Change]
Type		Parent Account	
Website		Phone	
Description		Industry	
		Employees	


Start Transaction:SalesForce_2_Create account

Click on Accounts link

Roles: link,focusable,element
Name: Accounts
ID Method: JavaScript
JavaScript: evalXPath("//a[text()=\"Accounts\"]");

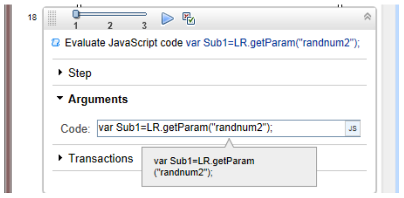
Click **Account** to open a form that lists the accounts.

By running the script multiple times, we can confirm that this element is recognized with the Automatic ID method.

You can also use the JavaScript ID Method:

```
evalXPath("//a[text()=\"Accounts\"]");
```

Evaluate JavaScript code var Sub1=LR.getParam ('randnum2')


Evaluate JavaScript code var Sub1=LR.getParam("randnum2");

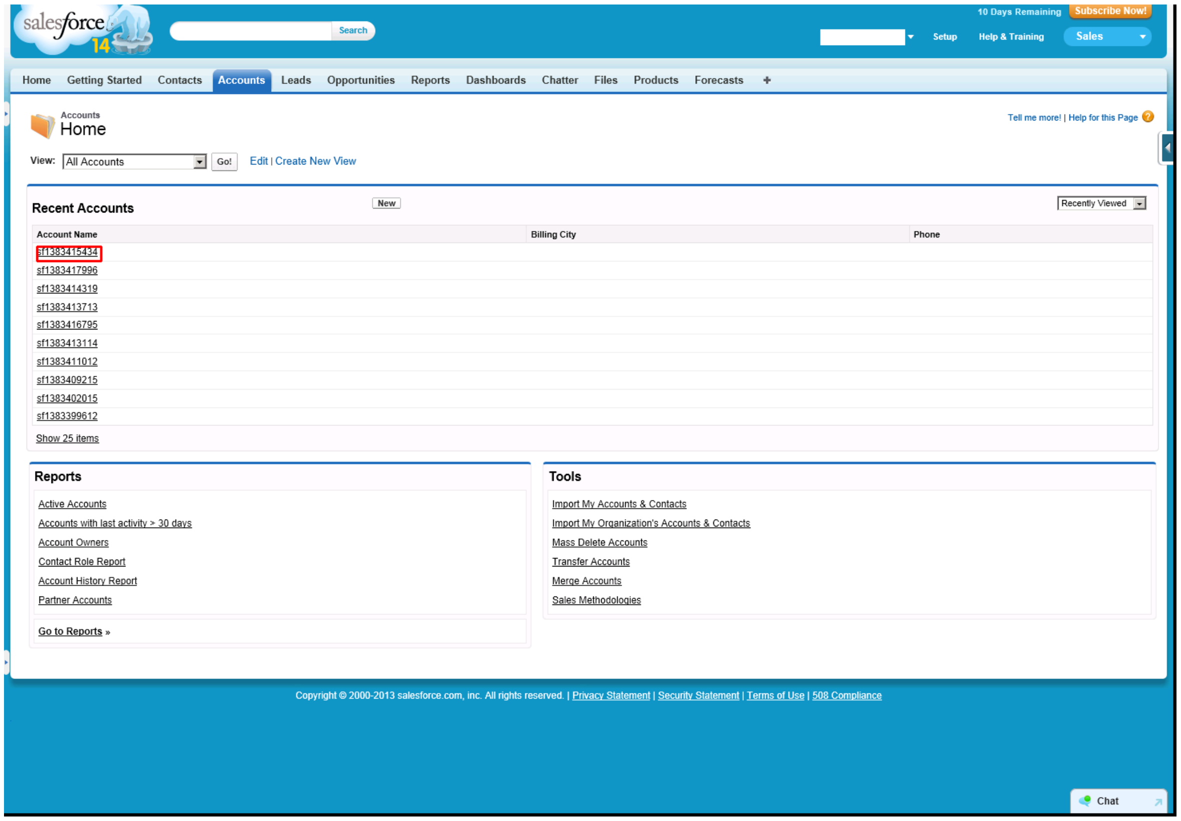
Code: var Sub1=LR.getParam("randnum2");

Create the *Sub1* variable and get its value from the parameter *randum2*:

```
var Sub1=LR.getParam("randnum2");
```

This variable is used to check (identify) and delete specific email sent within this interaction. The life span of this variable is within the page. When you change the page you must evaluate this Java script again if you need a value from the *randum2* parameter.

Click ArgsContext.Sub1



Accounts
Home

View: All Accounts

Recent Accounts

Account Name	Billing City	Phone
sf1383415434		
sf1383417996		
sf1383414319		
sf1383413713		
sf1383416795		
sf1383413114		
sf1383411012		
sf1383409215		
sf1383402015		
sf1383399612		

Show 25 items

Reports

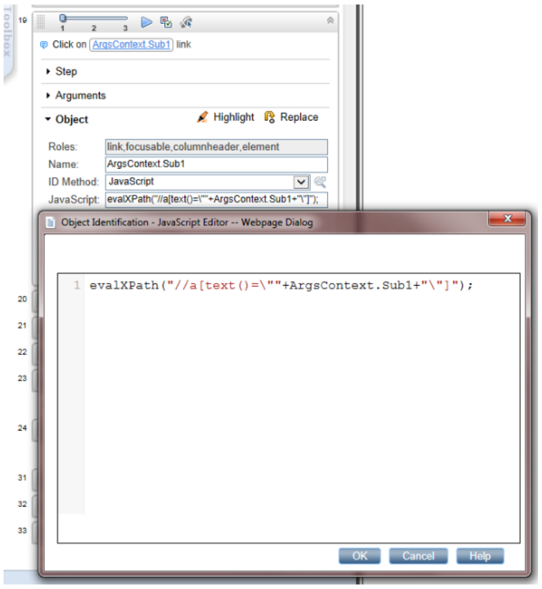
- Active Accounts
- Accounts with last activity > 30 days
- Account Owners
- Contact Role Report
- Account History Report
- Partner Accounts

Go to Reports »

Tools

- Import My Accounts & Contacts
- Import My Organization's Accounts & Contacts
- Mass Delete Accounts
- Transfer Accounts
- Merge Accounts
- Sales Methodologies

Copyright © 2000-2013 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance



Click on ArgsContext.Sub1 link

Object

Roles: link,focusable,columnheader,element
Name: ArgsContext.Sub1
ID Method: JavaScript
JavaScript: evalXPath("//a[text()=\""+ArgsContext.Sub1+"\"]");

Object Identification - JavaScript Editor -- Webpage Dialog

```
evalXPath("//a[text()=\""+ArgsContext.Sub1+"\"]");
```

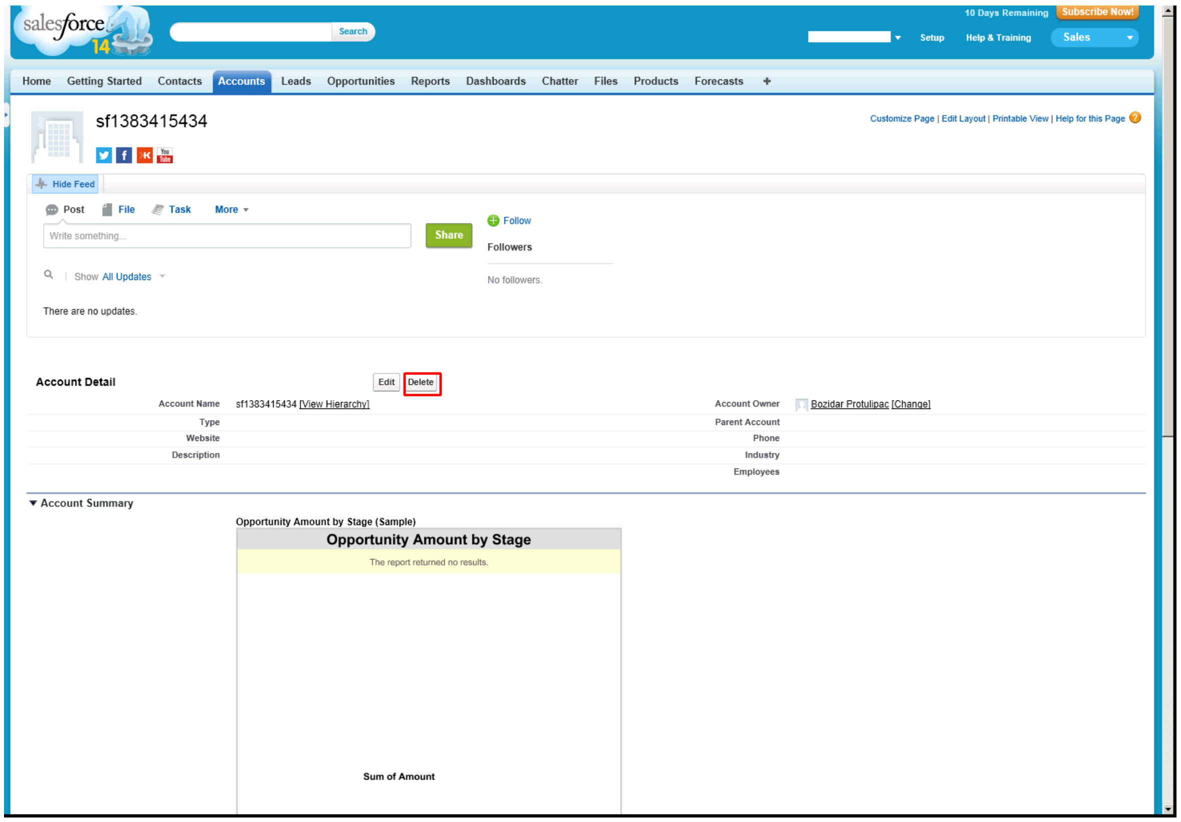
To identify an object, the name must be configured as *ArgsContext.Sub1*.

The ID Method is JavaScript:

```
evalXPath("//a[text()=\""+ArgsContext.Sub1+"\"]");
```

The specific account is found.

Click Delete



sf1383415434

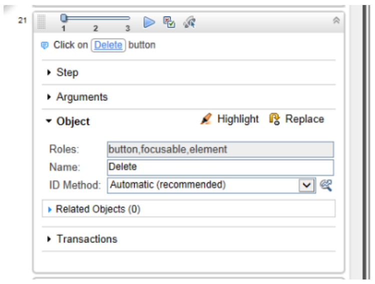
Account Detail

Account Name	sf1383415434 [View Hierarchy]	Account Owner	Bozidar Protulipac [Change]
Type		Parent Account	
Website		Phone	
Description		Industry	
		Employees	

Opportunity Amount by Stage (Sample)

Opportunity Amount by Stage

The report returned no results.



Click on Delete button

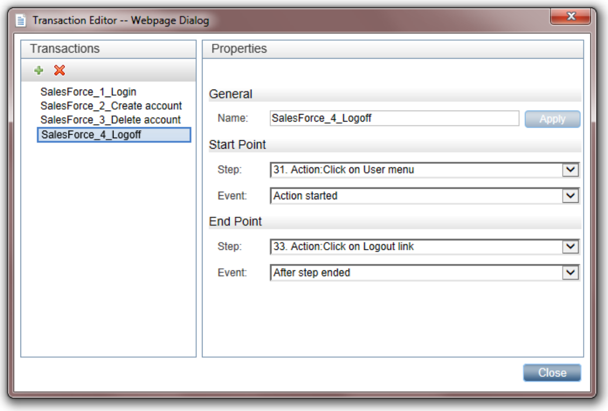
Object

Roles:	button,focusable,element
Name:	Delete
ID Method:	Automatic (recommended)

The **Delete** button is easy to recognize.

This step completes SalesForce_3_delete account.

Transaction — SalesForce_3_Delete account


Transaction Editor -- Webpage Dialog

Transactions

SalesForce_1_Login
SalesForce_2_Create account
SalesForce_3_Delete account
SalesForce_4_Logoff

Properties

General

Name: SalesForce_4_Logoff

Start Point

Step: 31. Action:Click on User menu

Event: Action started

End Point

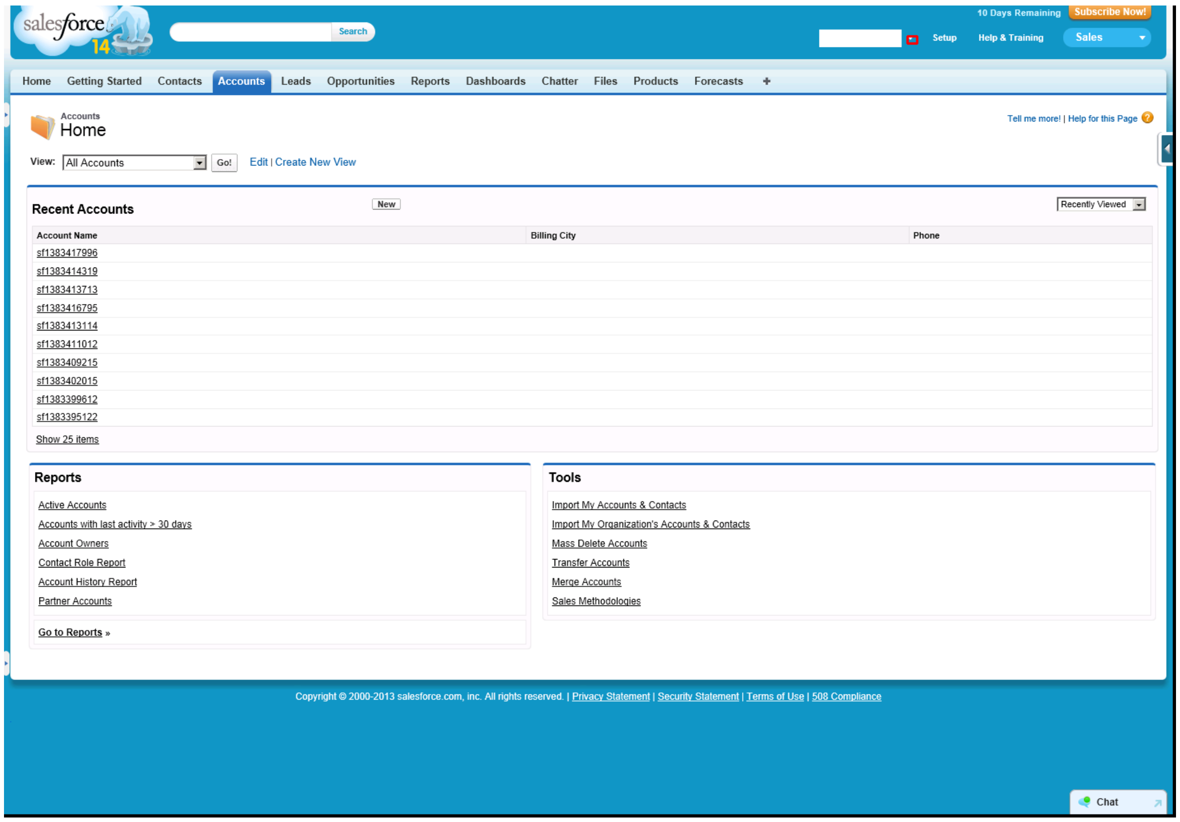
Step: 33. Action:Click on Logout link

Event: After step ended

The name of this transaction is “SalesForce_4_Logoff”.

This transaction contains all the steps necessary for logging off.

Click User Menu



Accounts
Home

View: All Accounts

Recent Accounts

Account Name	Billing City	Phone
sf1383417996		
sf1383414319		
sf1383413713		
sf1383416795		
sf1383413114		
sf1383411012		
sf1383409215		
sf1383402015		
sf1383399612		
sf1383395122		

Show 25 items

Reports

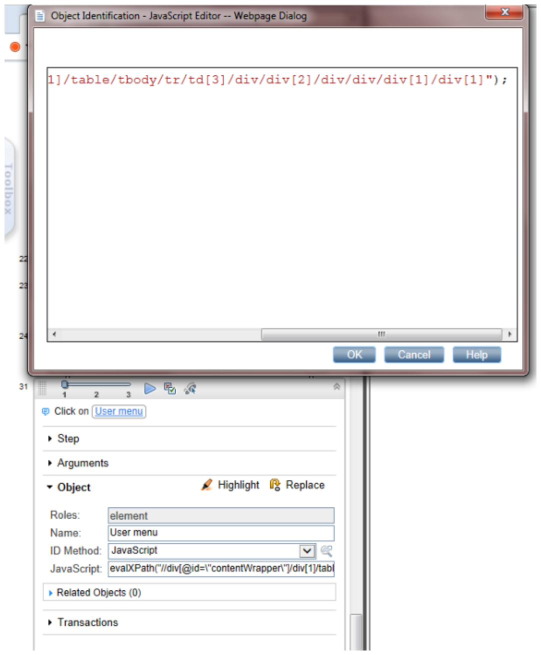
Active Accounts
Accounts with last activity > 30 days
Account Owners
Contact Role Report
Account History Report
Partner Accounts

Go to Reports »

Tools

Import My Accounts & Contacts
Import My Organization's Accounts & Contacts
Mass Delete Accounts
Transfer Accounts
Merge Accounts
Sales Methodologies

Copyright © 2000-2013 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance



```
1]/table/tbody/tr/td[3]/div/div[2]/div/div/div[1]/div[1]");
```

Roles: element
Name: User menu
ID Method: JavaScript
JavaScript: evalXPath("//div[@id=\"contentWrapper\"]/div[1]/tabl

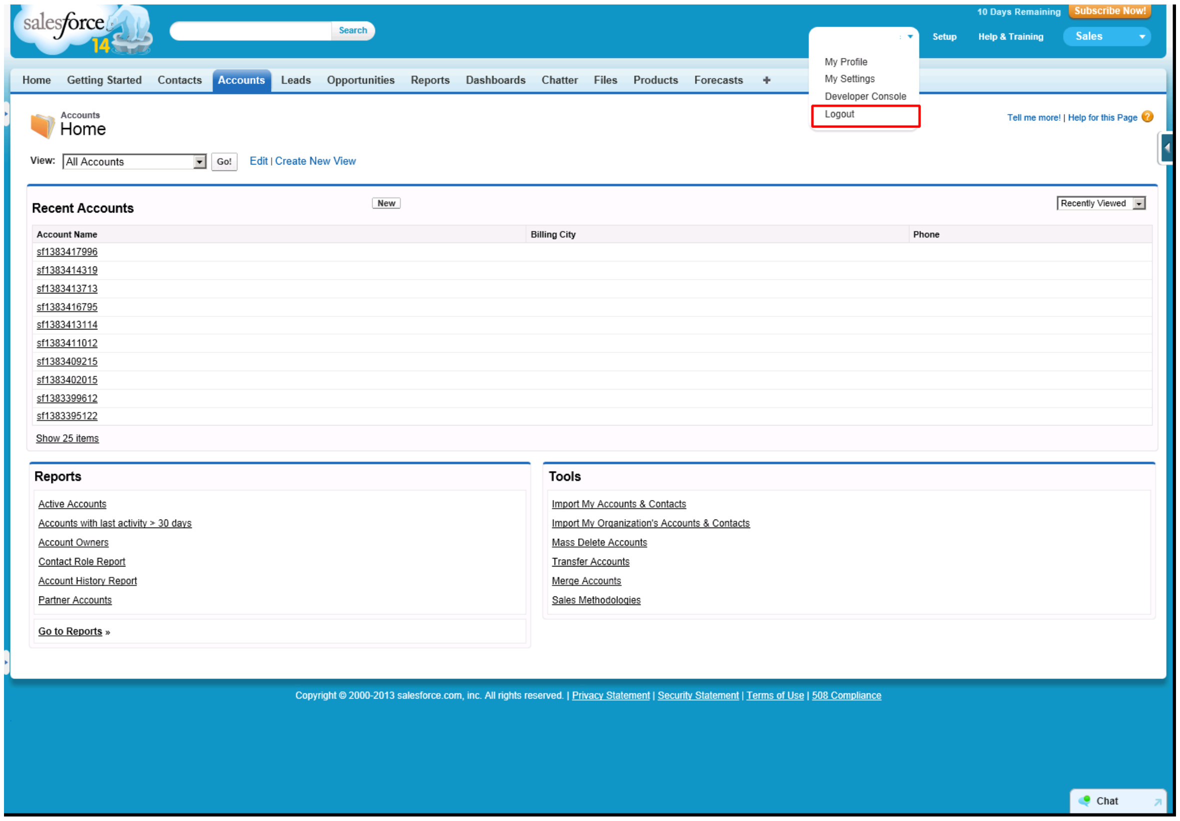
When you click the arrow next to the user name, a dropdown menu appears.

The ID Method is JavaScript:

```
evalXPath("//div[@id=\"contentWrapper\"]/div[1]/table/tbody/tr/td[3]/div/div
[2]/div/div/div[1]/div[1]");
```

You might want to use Firebug here.

Click Logout



Accounts
Home

View: All Accounts Go! Edit | Create New View

Recent Accounts New Recently Viewed

Account Name	Billing City	Phone
sf1383417996		
sf1383414319		
sf1383413713		
sf1383416795		
sf1383413114		
sf1383411012		
sf1383409215		
sf1383402015		
sf1383399612		
sf1383395122		

Show 25 items

Reports

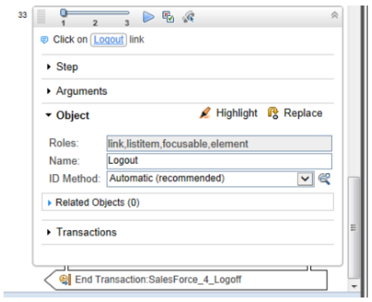
- Active Accounts
- Accounts with last activity > 30 days
- Account Owners
- Contact Role Report
- Account History Report
- Partner Accounts

Go to Reports »

Tools

- Import My Accounts & Contacts
- Import My Organization's Accounts & Contacts
- Mass Delete Accounts
- Transfer Accounts
- Merge Accounts
- Sales Methodologies

Copyright © 2000-2013 salesforce.com, inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | 508 Compliance



Click on Logout link

Object

Roles:	link,listitem,focusable,element
Name:	Logout
ID Method:	Automatic (recommended)

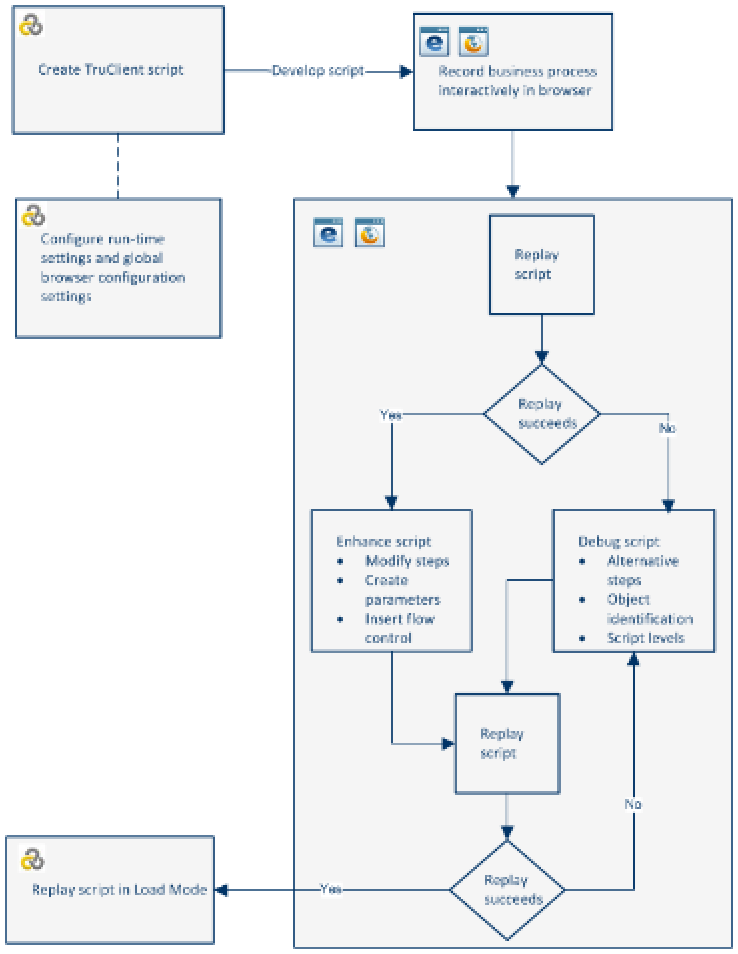
End Transaction:SalesForce_4_Logoff

Click **Logout** to end this transaction and script.

The optional JavaScript ID Method is: `evalXPath("//a[text()=\"Logout\"]");`

Appendix A: About TruClient for IE Protocol

The Ajax TruClient protocol interactively records scripts as you navigate through your business process. This enables TruClient to easily record and replay dynamic, complex web based applications and create user friendly scripts. Scripts are created in real-time and steps can be seen in the TruClient sidebar as they are performed.

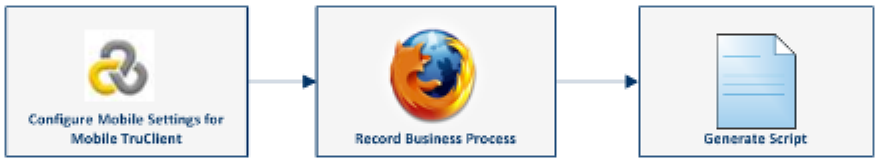


The Ajax TruClient for IE protocol is designed to work with applications running in IE9 in standard mode only. When you need a deeper insight, use Firebug Lite (see Use Firebug Lite on page 90).

Based on LoadRunner's innovative Ajax TruClient technology, Mobile TruClient enables you to test web applications designed for mobile devices. With this protocol you can:

• Simulate various mobile browsers.

• Develop scripts that are recorded on the user level making them clear and easily maintained.


Configure Mobile Settings for Mobile TruClient
Record Business Process
Generate Script

Appendix B: Time to Value

There are a few concepts that can speed things up when creating Office 365 scripts.

Use Firebug Lite

Firebug Lite is a third-party utility that provides many valuable development tools. You can edit, debug, and monitor CSS, HTML, and JavaScript live in any web page. You can access this utility by selecting **Advanced > FireBug Lite** or by pressing **F12** while in TruClient's IE9 browser.

If other default ID methods fail during the replay, you can use Inspect mode in Firebug Lite to select the evalXPath() that will choose the correct object based on the HTML. This is useful when it comes to tricky objects.

Standard Steps

When you look at the script, you will notice that some steps repeat often. We can group these steps as follows:

- "Navigate" (navigate to a URL) below
- "Click Object (Link, Textbox, Button, or Decorator)" on the next page
- "Type in Text Object" on page 91
- "Evaluate JavaScript" on page 91

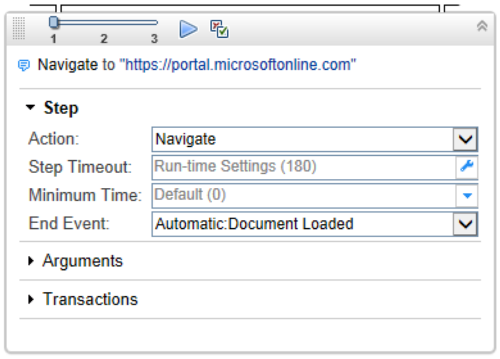
Navigate

Keep in mind when navigating to a defined URL, the End Event value defines when this step finishes. An End Event can be one of the following:

- **Action Completed** – The step ends when its action is completed. An example of an action is clicking a button.
- **DOM load** – The step ends when the process of loading a document completes.
- **DOM content loaded** – The step ends when the HTML parsing of the document completes.
- **Step network completed** – The step ends when all HTTP requests have completed including requests initiated by XMLHttpRequest.

- **Step synchronous network completed** – The step ends when all HTTP requests have completed, excluding requests initiated by XMLHttpRequest.
- **Dialog opened** – The step ends when a dialog box is opened.

We recommend that you set the End Event to **Automatic**, as seen in this script.


Navigate to "https://portal.microsoftonline.com"

Step

Action:	Navigate
Step Timeout:	Run-time Settings (180)
Minimum Time:	Default (0)
End Event:	Automatic:Document Loaded

Click Object (Link, Textbox, Button, or Decorator)

Two things are important for the Click on Object step:

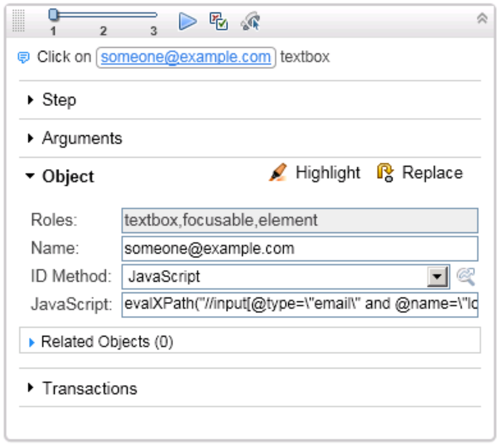
- Arguments
- Object settings

The X and Y coordinates are defined relative to the object in the Arguments settings. If the coordinates are left empty, the click is executed in the middle of the object.

If the object is not recognized with the ID Method set to automatic, you will need to use the JavaScript ID Method. It relies on finding the required element or string in the HTML code. When you select the JavaScript ID Method, you can get:

- Straight forward syntax:
  ```
  evalXPath("//input[@type=\"email\" and @name=\"login\"]");
  ```
- Suspicious syntax:
  ```
  evalXPath("/html/body/div[5]/div/div[2]/div[3]/div/div[1]/div[6]/div/div/div
  [3]/div[2]/div[1]/div[7]/div/div/input");
  ```

When confronted with suspicious syntax, use Firebug Lite as described in Use Firebug Lite on page 90.



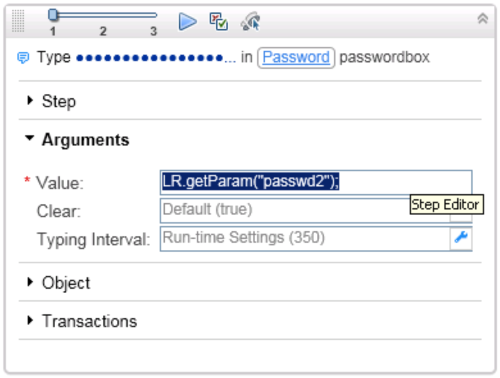
Click on someone@example.com textbox

Roles: textbox,focusable,element
Name: someone@example.com
ID Method: JavaScript
JavaScript: evalXPath("//input[@type=\"email\" and @name=\"lc

Type in Text Object

Argument values can be plain text or a parameter. Examples are:

- Plain text – “this is plain text typed in a text box”
- Getting a value from a parameter – LR.getParam("passwd2");



Type ●●●●●●●●●●●●●●●●... in Password passwordbox

* Value: LR.getParam("passwd2");
Clear: Default (true)
Typing Interval: Run-time Settings (350)

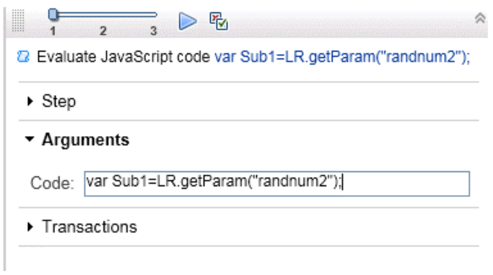
Evaluate JavaScript

By evaluating JavaScript, you can call a function, for example:

```
LR.evalC("getRand");
```

Or define a variable that will be used later:

```
var Sub1=LR.getParam("randnum2");
```



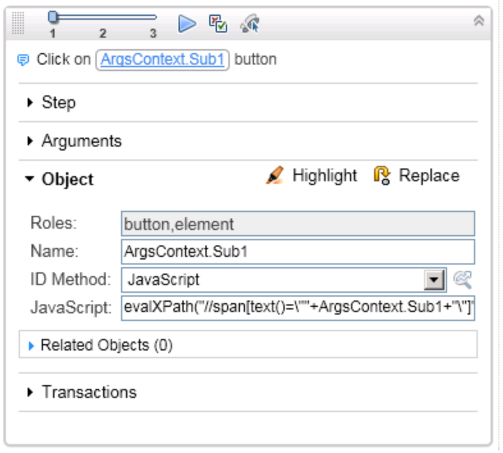
Evaluate JavaScript code var Sub1=LR.getParam("randnum2");

Code: var Sub1=LR.getParam("randnum2");

Variables and ID Methods – How to Combine

To use a defined variable to find a specific and unique object:

1. Record the step by clicking that object.
2. To make a script more robust, make those changes:
   - Change Object name to: ArgsContext.Sub1
   - Change JavaScript ID Method to: evalXPath("//span[text()=\""+ArgsContext.Sub1+"\"]");



Click on ArgsContext.Sub1 button

Roles: button,element
Name: ArgsContext.Sub1
ID Method: JavaScript
JavaScript: evalXPath("//span[text()=\""+ArgsContext.Sub1+"\"]

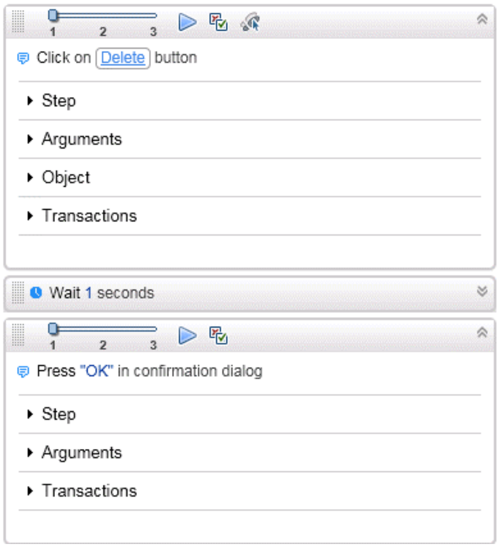
Meaning of Wait Step

A Wait step is useful when you want to wait for a very heavy page to load and you do not want to timeout while waiting. Just put a Wait step after this load step and enter the number of seconds to wait.

Another example of when to use a Wait step is when a dialog box is in the script. This dialog box can be any of the following:

- Alert
- Authentication
- Confirmation
- Prompt
- Prompt Password

Put a Wait step between the step that caused the dialog box to open and the dialog box itself. In this way, you give the dialog box step time to complete.



Click on Delete button

- Step
- Arguments
- Object
- Transactions

Wait 1 seconds

Press "OK" in confirmation dialog

- Step
- Arguments
- Transactions

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on BPM Monitoring Solutions - Best Practices (Business Service Management 9.25)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to Sw-doc@hp.com.

We appreciate your feedback!