CREATORCON
May 19 - 20, 2016 • Mandalay Bay • Las Vegas, NV

Lab Guide

Mastering The Force with Xplore: Developer Toolkit

James Neale & Tim Short

Default Login / Password:

admin / Knowledge16

itil / Knowledge16

employee / Knowledge16

This
Page
Intentionally
Left
Blank

Introduction

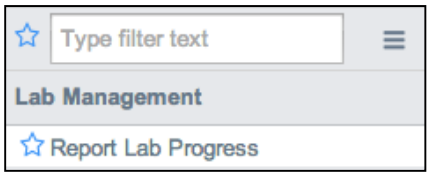
This Lab Guide caters for developers who vary in levels of experience with ServiceNow. Most labs have an additional **Jedi** section with some trickier tasks that will really help get you in the swing of using Xplore and thinking about how you can use it in your day to day activities.

If you haven't already opened Xplore, click the glasses icon in the ServiceNow header or navigate to the Xplore application and open it.

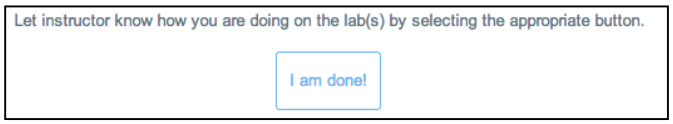


Progress Report

1. At the end of each Lab, please navigate to **Lab Management > Report Lab Progress**.



2. Click **I am done!**



©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 1

In this lab we will look at how Xplore captures and displays standard ServiceNow log statements.

Lab 1
Logging

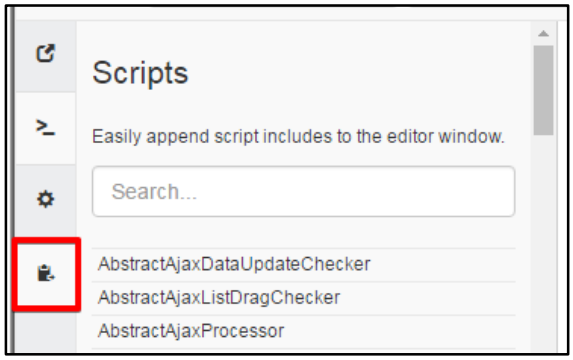
Everyone

1. Write some log statements in the Xplore code editor and see how it responds when you run your code.

```
gs.log('Hello world!');
gs.print('Auf wiedersehn, pet!');
gs.warn('What shall we do today?');
gs.debug('How many messages can these guys come up with?');
gs.error('Something went seriously wrong.');
gs.addInfoMessage('Hello from the dark side.');
gs.addErrorMessage('I love Xplore!');
jslog('Yep, this works as well – but only in the client.');
```

Jedi

1. Open the Script Include importer.



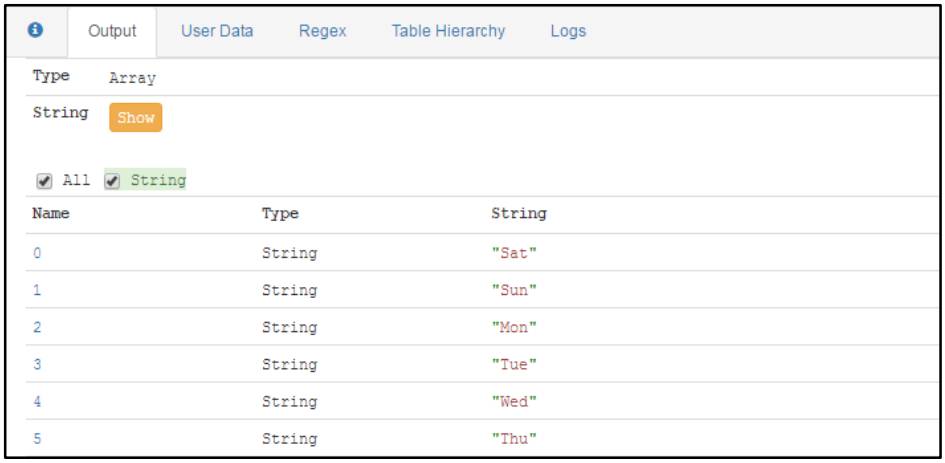
Click the Script Importer button to open the Script Include search pane

©2016 ServiceNow, Inc. All rights reserved.

2. Search for **xplore_lab_1_log_jedi** and click it to add it to your editor (alternatively click replace to start fresh in your editor).

3. Use Xplore to debug the function and make it work. (Hint: there are 2 problems)

4. If you get an array of 365 or 366 day names, then you have succeeded.



Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.

2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 2

Lab 2
GlideRecord

This lab explores the usefulness of writing GlideRecord lookups in Xplore.

- Perform GlideRecord queries and view the record in the Xplore output pane.
- Find out GlideRecord field data types.

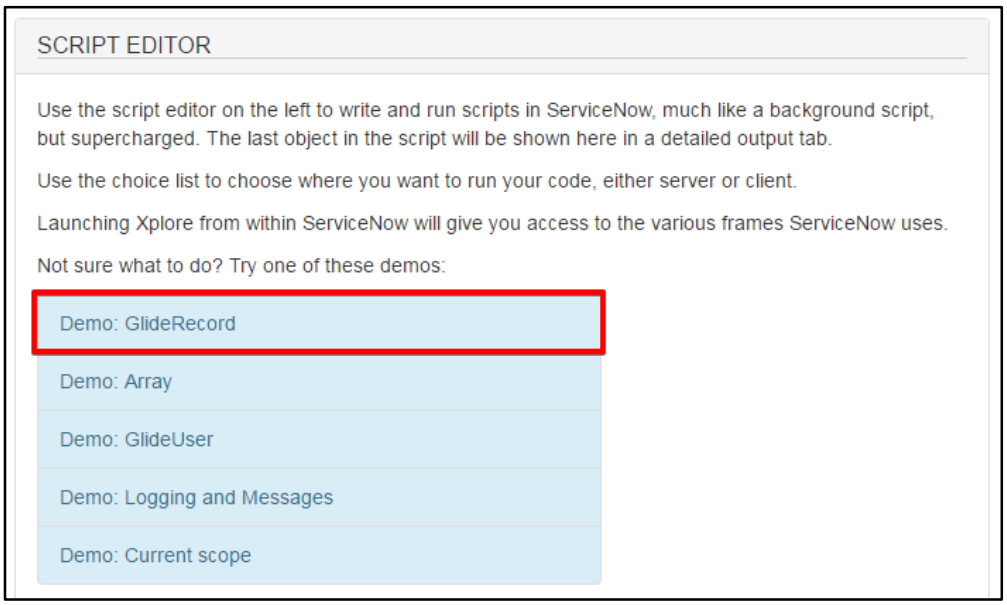
Limitation: You won't see property functions such as getED() and getGlideElement() as Xplore cannot read some Java functions.

Everyone

1. Use the GlideRecord template to lookup an incident and view the record in the Xplore output by hitting the Run button, or Ctrl + Enter.



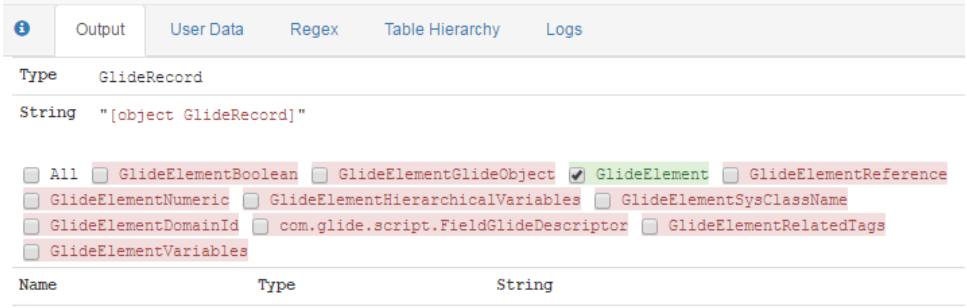
Click the help icon.



Click the Demo: GlideRecord button

©2016 ServiceNow, Inc. All rights reserved.

2. Use the filters find who last updated the record.



Use the filters to search objects quickly.

3. Try modifying the script to get your user record. Hint: the table is *sys_user* and your sys_id is returned using *gs.getUserID()*

Jedi

1. Insert a GlideRecord to the incident table. (Hint: not necessarily as easy as it sounds.)

```
// this should do it, right?
var gr = new GlideRecord('incident');
gr.short_description = 'What happened to Boaty McBoatface?';
gr.insert();
```

Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.

2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 3

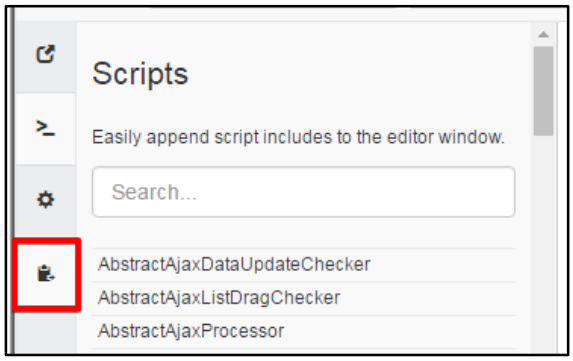
XML and JSON can be tricky to work with in ServiceNow. Thankfully, Xplore makes it really easy.

- Use the User Data functionality to paste in large chunks of text such as XML or JSON, or just really long strings.
- Use the 'user_data' variable in the Xplore script scope.
- Explore the XML object using the output pane and breadcrumb/navigation functionality.

Lab 3
Working with XML/JSON

Everyone

1. Open the Script Include importer.



Click the Script Importer button to open the Script Include search pane

2. Search for **xplore_lab_3_json** and click it to add it to your editor (alternatively click replace to start fresh in your editor).

3. Select the **JSON string** and copy/paste it to the large text area in the **User Data** tab.



Select the User Data tab.

4. Click the Run button (or Ctrl + Enter) and explore the object by clicking on the property names and the breadcrumb in the Output tab.

©2016 ServiceNow, Inc. All rights reserved.

5. Optional questions about the data object:
   - What is each object in the array representing? _____
   - What is the **favourite fruit** of ***Lois Orr***? _____
   - How many **friends** does ***Charles Glenn*** have? _____

Jedi

1. Open the Script Include importer.
2. Search for **xplore_lab_3_xml_jedi** and click replace to add it to your editor.
3. Select the XML string and copy/paste it to the large text area in the User Data tab.
4. Run the script and explore the output. See if you can fix it so the right values are being printed and you have something that looks like this:

```
01:19:14.702: Description: Cardigan Sweater
01:19:14.702: > Gender: Men; Item Number: QWZ5671
01:19:14.702: --> Size: Medium
01:19:14.702: --> Size: Large
01:19:14.703: > Gender: Women; Item Number: RRX9856
01:19:14.703: --> Size: Small
01:19:14.703: --> Size: Medium
01:19:14.703: --> Size: Large
01:19:14.703: --> Size: Extra Large
```

Expected output from XML script.

   Hint: are those XPath's correct?

5. Now you've got it fixed, try adding the Price to the output so it looks like this:

©2016 ServiceNow, Inc. All rights reserved.

```
01:05:04.773: Description: Cardigan Sweater
01:05:04.774: > Gender: Men; Item Number: QWZ5671
01:05:04.774: --> Price: 39.95
01:05:04.774: --> Size: Medium
01:05:04.774: --> Size: Large
01:05:04.774: > Gender: Women; Item Number: RRX9856
01:05:04.774: --> Price: 42.50
01:05:04.775: --> Size: Small
01:05:04.775: --> Size: Medium
01:05:04.775: --> Size: Large
01:05:04.775: --> Size: Extra Large
```

Expected output once the Price has been added.

6. Remove the gs.print calls, instead putting the values into an object that can be rendered in the Xplore output tab.

Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.
2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 4

Xplore was designed in part from the view of wanting to understand how things were built in ServiceNow. The main object of exploratory programming is diving into things you don't understand, simply by working with them.

- Explore objects and find/validate function calls easily.
- Get an overview of a script include and view the code.

Lab 4
Exploratory Programming

Everyone

1. Clear the code editor.
2. Type gs.getSession() and hit Run.
3. Find what function you would call to get the impersonating user's display name when looking at the output of gs.getSession(). Click the parentheses **()** next to the function name to run the function. What happens?
4. Browse some ServiceNow classes and objects that you know and see if things show up you didn't know existed. E.g. gs.getUser() or GlideStringUtil. You'd be surprised!
5. Try running Xplore with just the word 'this' as your script. Look at how much stuff is in the global scope by default!

Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.
2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 5

Lab 5
Performance
& Profiling

One of the great things about Xplore is that you can easily test performance of your code.

- Xplore is great for debugging slow and inefficient code.
- Use functional profiling to highlight slow code with timers.

Everyone

1. Open the Script Include importer.
2. Search for **xplore_timer** and click replace to add it to your editor.
3. Search for **xplore_lab_5_performance** and click the title (not replace) to append it to the timer script that you just opened.
4. Use the timer object to add timers to the script and work out where the latency is.

```
timer.start('test');
// ...
timer.stop('test');
```

Jedi

5. Refactor and memoize the slow GlideRecord query function so it remembers the results of specific value lookups.

   Your timers should show that you saved more than half the time (assuming you haven't removed the artificial latency).

Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.
2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 6

Table hierarchy gives insight into the table structure of ServiceNow.

- Simple but indispensable tool to view the hierarchy of tables.
- Great for CMDB table checking.

Lab 6
Table
Hierarchy

Everyone

1. Use different searches to browse the table hierarchy. Try the wildcard and label search. e.g. incident, *request, *cmdb
2. What is the database name of the Instance Comparison table?
3. A customer wants to add a field to the Direct Current Plant Battery Bank and Power Generator tables. Use Xplore to find the logical place for this field – not cmdb_ci!

Progress Report

4. Navigate to **Lab Management > Report Lab Progress**.
5. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 7

Lab 7 Regular Expressions

Regular expressions work in browsers and third party testers but they haven't always worked in ServiceNow. Use the built in expression tester in Xplore to test directly on the server (or client).

WARNING: we have seen regular expressions kill an instance when the string size is too big: somewhere around 18MB.

Everyone

1. Open the Script Include importer.
2. Search for **xplore_lab_7_regex** and click replace to add it to your editor.
3. Copy the text to the regular expression string field.
4. Try out the included expressions by copying them to the regular expression input box. Some of the expressions don't work straight away, can you work out why? (Hint: Options)
5. Read through the included reference sections and try matching some of your own text. Use your name as the test string and match it using the expression [a-zA-Z]+ which matches 1 or more upper or lowercase letters.

Jedi

1. Write a regular expression to match the Incident number shown in the text.
2. Write a regular expression to match the script tags show in the text.
3. Modify your script tag matcher to return **only** the source file names of the scripts.
4. Fix the HTML tag matcher so it matches the IP Address paragraph.

Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.
2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 8

Scopes limit the amount of stuff you have access to. Xplore allows switching between scopes so you can verify your code easily.

Scoped scripts only have access to classes, methods and properties that have been explicitly approved for use in a scope. This means that you can get errors. The Xplore blacklist allows you to work around these.

Blacklists allow you to ignore specific properties to prevent ServiceNow throwing uncatchable exceptions.

Lab 8
Scopes & Blacklists

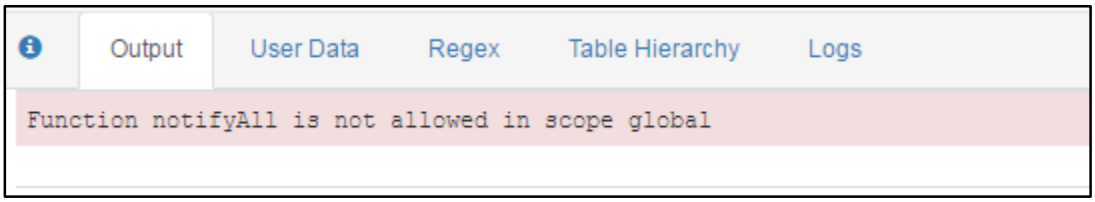
Everyone

1. Run the GlideRecord demo script in global scope then run it in the Social Knowledge [sn_kb_social_qa] scope. What differences do you see? Explore other objects and see what happens.

2. Try out the following scripts:

   - gs.getSession();
   - gs.getUser();

   If you see a message like this, it means you need to blacklist that property/method:



   The current workaround for the access errors is to manually add the offending property name to the Xplore blacklist at the start of your script.

```
// totally ignore these properties/methods
global.snd_xplore.blacklist('notify,notifyAll,equals,wait,unwrap',
        'toString,hashCode,getClass');
```

©2016 ServiceNow, Inc. All rights reserved.

**The black lists will be data driven at some point in the future given enough demand. It's likely that the availability of methods and properties will change, which is why nothing has been hard coded into Xplore to this point.*

Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.
2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Lab Goal 9

Lab 9
Client Side

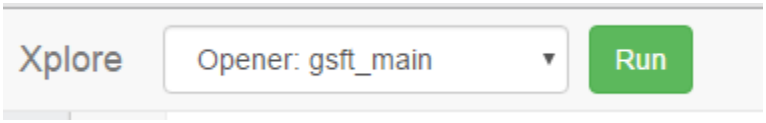
Xplore will also work with the ServiceNow client.

- Run g_form methods and more.
- Explore ServiceNow client-side objects.

TIP: Xplore will have access to the ServiceNow frames only when you open it from ServiceNow (click on the module or the glasses). Typing snd_xplore into a new browser window will not give access back to the ServiceNow client.

Everyone

1. Load an incident in ServiceNow
2. Click the glasses icon to open Xplore (or go back to it if it's already open).
3. Change the target to 'Opener: gsft_main'.



The target selector

4. Type g_form and run it.
5. Find the functions that turn attachments on and off and run them. (Remember you can just click on the parentheses () to run a function without any parameters)
6. Generate an alert:

```
alert('Hello world!'); // fun times!
```

7. Change the target.
   - Select Client to just run JavaScript in the browser.
   - Anything prefixed by Opener is a frame from the window that you launched Xplore from.

©2016 ServiceNow, Inc. All rights reserved.

Progress Report

1. Navigate to **Lab Management > Report Lab Progress**.
2. Click **I am done!**

©2016 ServiceNow, Inc. All rights reserved.

Answers

Only read this section if you are stumped and want the answers to the labs.



©2016 ServiceNow, Inc. All rights reserved.

Lab 1: Logging Answer

```
/**
 summary:
   Generate a list of days for a given year.
 param: year [Integer]
   The year to generate the days for.
 returns: [Array]
   Array of short day names.
**/
function generateDaysInYear(year) {
 var result = [],
     days = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'],

 // step 2: fix date format
 d = new Date(year + '/01/01 00:00:00');

 // step 1: does date exist?
 gs.print(d);

 // step 3: infinite while loop?
 gs.print(d.getYear() + ' ?< ' + year);
 //return; // comment out in step 4

 // step 4: getYear -> getFullYear
 while (d.getFullYear() < year + 1) {
   result.push(days[d.getDay()]);
   d.setTime(d.getTime() + 86400000);
 }

 return result;
}

// Generate a date for a year
generateDaysInYear(2016);
```

©2016 ServiceNow, Inc. All rights reserved.

Lab 3: XML Answer

```
var doc = new XMLDocument(user_data),
   root_path,
   sub_path,
   product,
   items,
   sizes,
   i, j;

product = doc.getNode('//product');
root_path = '//catalog_item';
items = doc.getNodes(root_path);

// get the product description
gs.print('Description: ' + product.getAttribute('description'));

// step 1.2 - add getLength() to each nodes container, as they are objects
if (items.getLength()) {
 for (i = 0; i < items.getLength(); i++){
   sub_path = root_path + '[' + (i + 1) + ']';

   // get the gender the item is intended for, and its number
   gs.print('> Gender: ' + doc.getNodeText(sub_path + '/@gender') +
         '; Item Number: ' + doc.getNodeText(sub_path + '/item_number'));

   // Task 5: Jedi - get the price
   gs.print('--> Price: ' + doc.getNodeText(sub_path + '/price'));

   // get the descriptions of each size available
   sizes = doc.getNodes(sub_path + '/size[@description]');

   // step 1.2
   if (sizes.getLength()) {
    for (j = 0; j < sizes.getLength(); j++) {
      gs.print('--> Size: ' + sizes.item(j).getAttribute('description'));
    }
   }

   // optional step 1.4
   else {
    gs.print('No sizes.');
   }

 }
}
```

©2016 ServiceNow, Inc. All rights reserved.

```
// optional step 1.3
else {
  gs.print('No items.');
}


// step 1.1 - seeing no items are being printed, look at items which are not printing
// and we see zero length
//items;

// step 2 - seeing items are zero length, we verify root_path is correct
// step 3 - seeing number is undefined, we find the real node for number
// View a nicely formatted XML document
//doc.toIndentedString();
```

©2016 ServiceNow, Inc. All rights reserved.

Lab 5: Performance Answer

```
// simple timer object
var timer = (function () {
  var timers = {};
  return {
    start: function (name) {
      timers[name] = (new Date()).getTime();
    },
    stop: function (name) {
      if (timers[name]) {
        time = (new Date()).getTime() - timers[name];
        gs.print('[timer] ' + name + ': ' + time + 'ms');
      }
    }
  };
})();

// memoization of getUserName function
var getUserName = (function () {
  var cache = {};
  return function (user_id) {
    var user;

    user_id += '';
    if (!(user_id in cache)) {
      timer.start('getUserName');

      // Artificial timer to increase function run time
      gs.sleep(100);

      user = new GlideRecord('sys_user');
      user.addQuery('sys_id', '=', user_id);
      user.query();

      timer.stop('getUserName');

      cache[user_id] = user.next() ? user.getValue('user_name') : '';
    } else {
      gs.print('Using cache: ' + cache[user_id]);
    }

    return cache[user_id];
  };
})();
```

©2016 ServiceNow, Inc. All rights reserved.

```
function getUserNames(group) {
  var gr,
      user_name,
      user_names;

  //timer.start('getUserNames');

  gr = new GlideRecord('sys_user_grmember');
  gr.addQuery('group', '=', group);
  gr.query();

  user_names = [];
  while (gr.next()) {
    user_name = getUserName(gr.getValue('user'));
    if (user_name) {
      user_names.push(user_name);
    }
  }

  //timer.stop('getUserNames');

  return user_names;
}

function getAllUsersByGroup() {
  var all_user_names = {},
      gr;

  gr = new GlideRecord('sys_user_group');
  gr.query();
  while (gr.next()) {
    all_user_names[gr.getDisplayValue()] = getUserNames(gr.sys_id);
  }

  return all_user_names;
}

timer.start('getAllUsersByGroup');
getAllUsersByGroup();
timer.stop('getAllUsersByGroup');
```

©2016 ServiceNow, Inc. All rights reserved.

Lab 7: Regex Answers

```
! Match the incident number
INC\d+

! Match the script tags
<script\b.*?>[^<]*</script>

! Match the script tag src only
(?:<script\b.*?src=["'])([^'"]+)

! Modified HTML tag matcher to match carriage returns as well
<([A-Z][A-Z0-9]*)\b[^>]*>([^>]*?)</\1>
```

©2016 ServiceNow, Inc. All rights reserved.