knowledge14®

April 27–May 1, 2014 • Moscone Center • San Francisco, CA

Debugging 301: Problem Diagnosis and Prevention Techniques

Chris Maloy
Team Lead Expert Services
ServiceNow

Matt Gaide
Sr. Technical Consultant Expert Services
ServiceNow

Oliver Schmitt
Team Lead Central Europe PS
ServiceNow

servicenow®

© 2014 ServiceNow All Rights Reserved

Agenda

Debugging Concepts

Server Side Debugging & Troubleshooting

Client Side Debugging & Troubleshooting

Coding Patterns

Additional Resources & Q&A

© 2014 ServiceNow All Rights Reserved

Debugging Concepts

Deal with – It does not work, It is slow

© 2014 ServiceNow All Rights Reserved

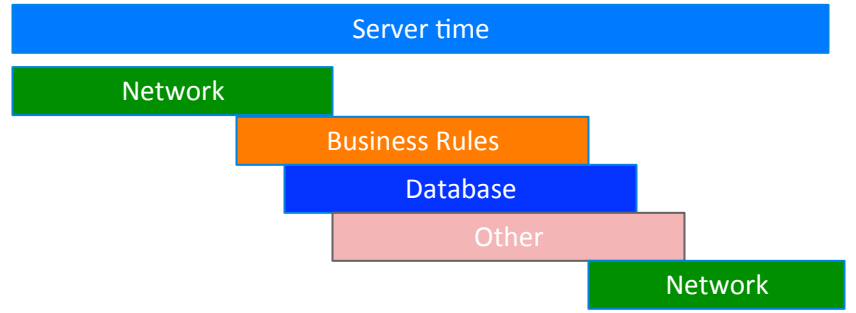
Debugging Concepts

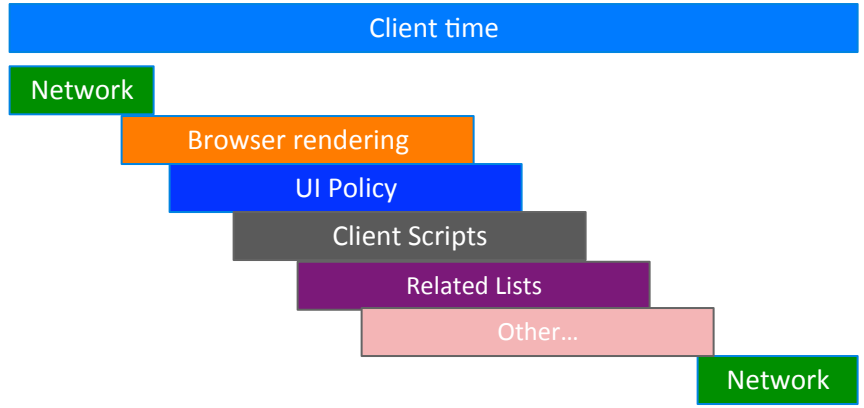
- Avoidance – The best way to troubleshoot and debug is to plan and do it right the first time
  - Strict naming conventions
  - Technical Best Practices
  - Coding Standards
  - Coding Patterns
- Wiki:
  - http://wiki.servicenow.com/index.php?title=Technical_Best_Practices
- Server
  - Detection
  - Pin-pointing and triage
- Client
  - Analyze script runtime
  - Identify bottleneck
  - Server round-trips

© 2014 ServiceNow All Rights Reserved

Performance Supply Chain

- Server:
  - Request will be posted to the Server over the Network
  - Business Rules, Script Includes and Workflows will be executed
  - Potential DB calls run
  - Response will be sent back to the client
- Client
  - Server Response will be received
  - Browser renders the DOM
  - First UI Policies and the all related Client scripts will be executed
  - Related lists will be loaded





© 2014 ServiceNow All Rights Reserved

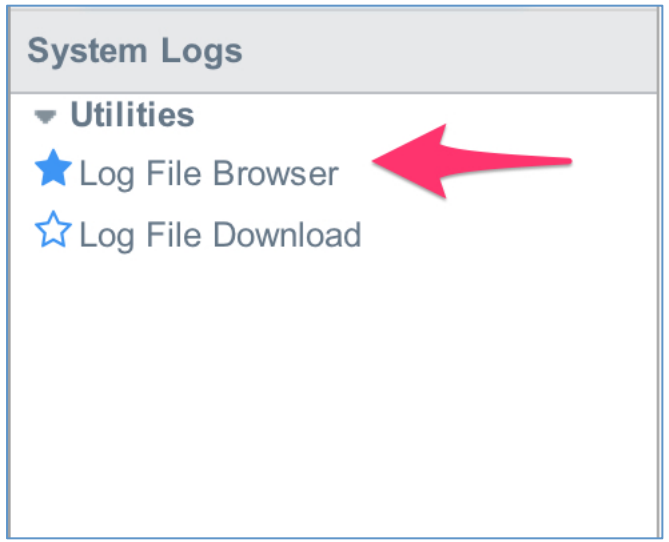
Server Side Debugging & Troubleshooting

Where can I find what is running slow?

© 2014 ServiceNow All Rights Reserved

System Log Files - App log

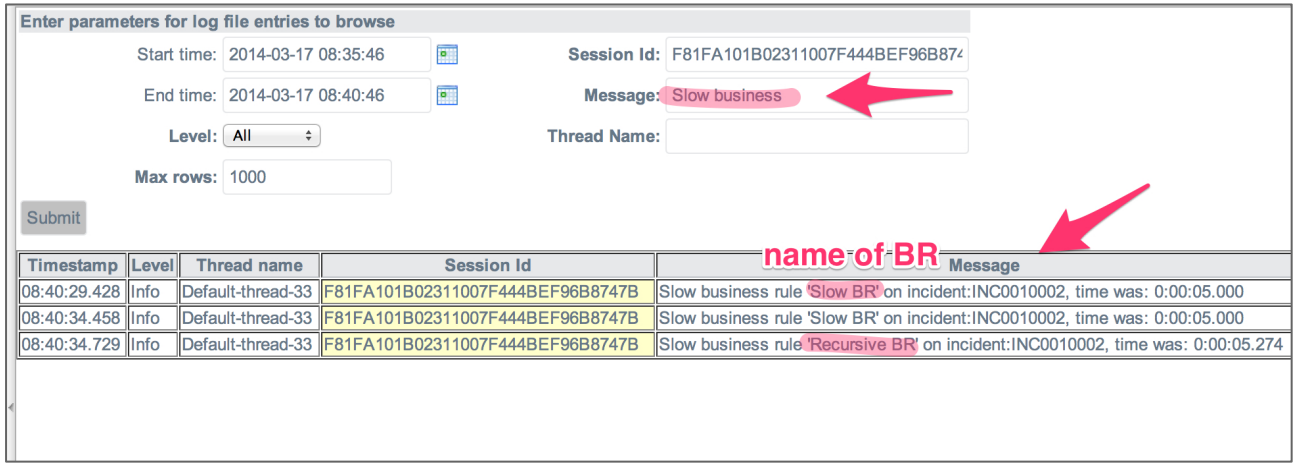
- Check the size of log files regularly
- ServiceNow creates compressed archives of system logs every two days and purges log archives after 21 days.
- Messages to look for
  - Slow evaluate
  - Slow business rule
  - Recursive business rule
  - Compiler exception
  - Warning - large table limit
  - Extremely large result
- Might be a good idea to write your own script to periodically check these files.



© 2014 ServiceNow All Rights Reserved

Log File Browser

- Utility to search the system log files
- This is great if you know a users session Id.
- Necessary parameters:
  - Start time
  - End time
- Example Showing how to get Session Id and querying log file.



© 2014 ServiceNow All Rights Reserved

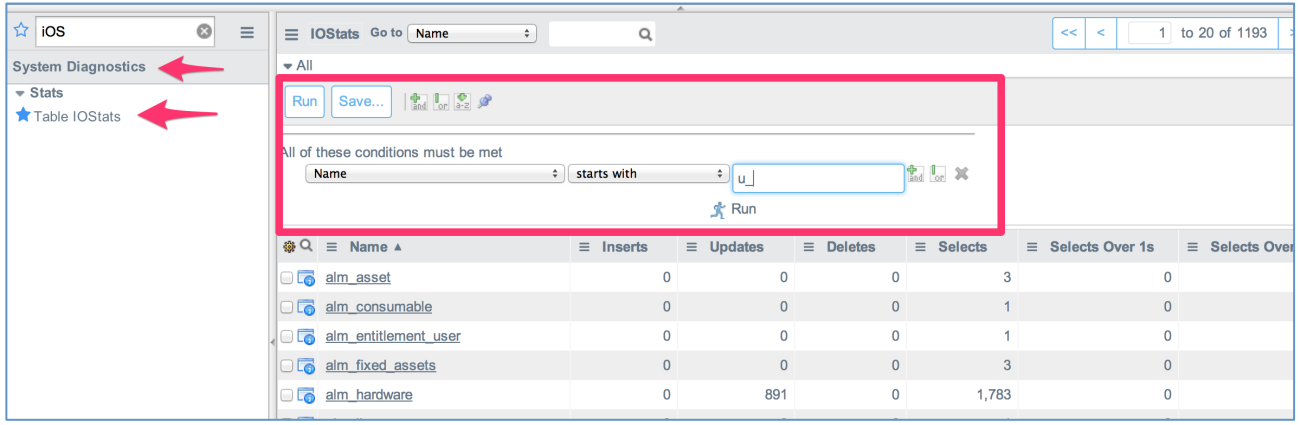
App Log Exercise

- Slow Business Rule and Recursive Business Rule Exercise
- http://wiki.servicenow.com/index.php?title=Logs_and_Queues_Best_Practices#Checking_Log_File_Sizes

© 2014 ServiceNow All Rights Reserved

Table IOStats

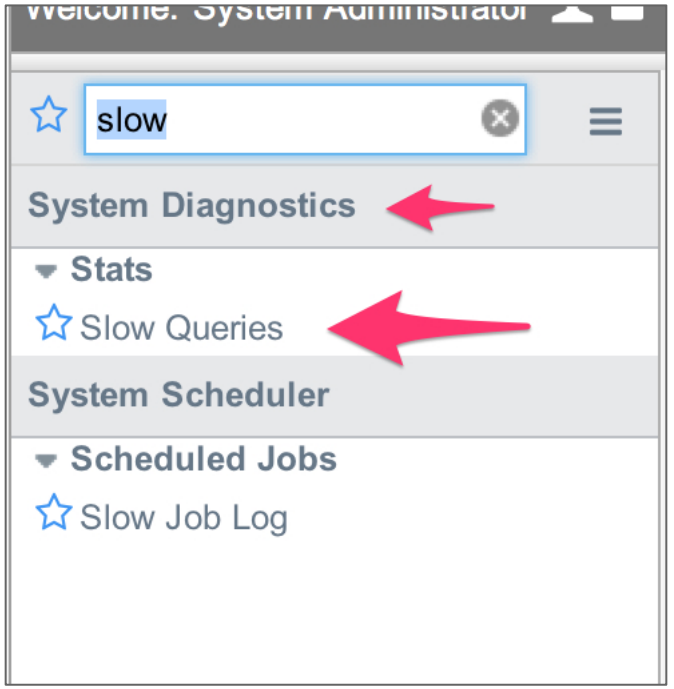
- Useful at a very high level
- Use it with custom tables to identify usage and slowness



© 2014 ServiceNow All Rights Reserved

Slow Queries

- This table stores queries whose total execution time is longer than one second (1000ms).
- Example showing which fields to focus on when using this table.



© 2014 ServiceNow All Rights Reserved

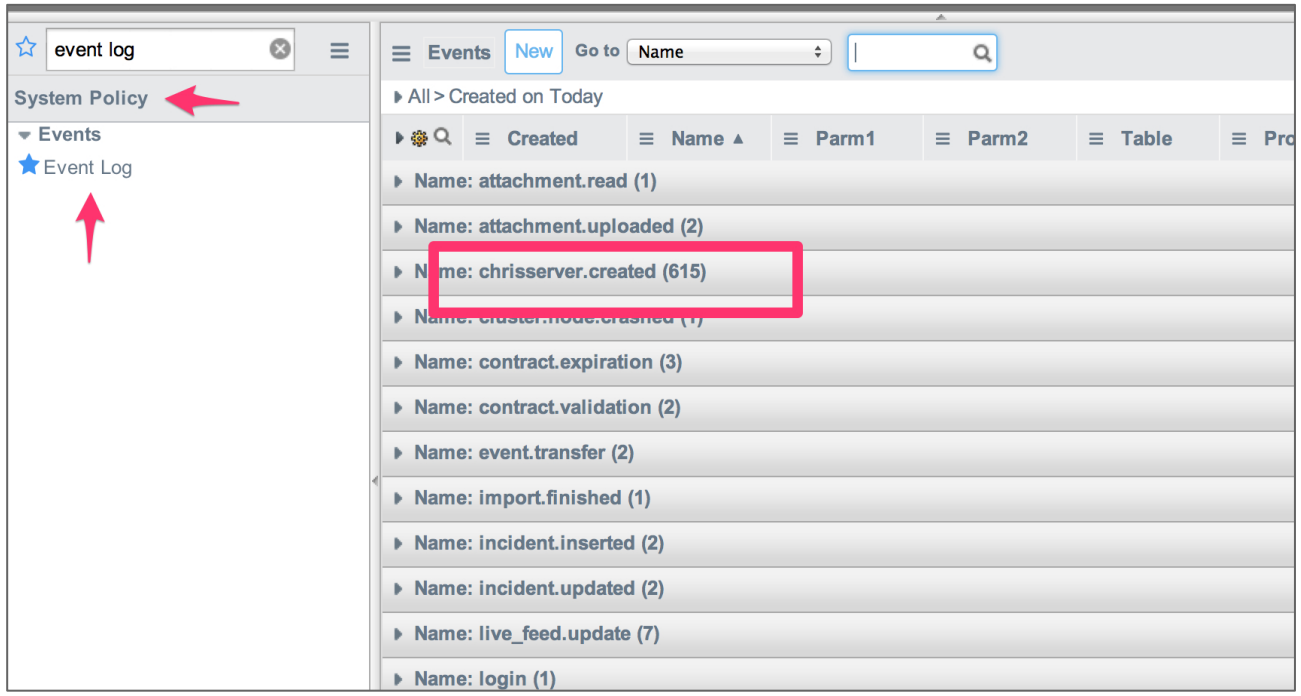
Slow Query Exercise

- IOStats Exercise
- http://wiki.servicenow.com/index.php?title=Logs_and_Queues_Best_Practices#Diagnosing_Problems_With_Slow_SQL_Statements

© 2014 ServiceNow All Rights Reserved

Slow Events – Event log

- Which events are taking the longest to process (the processing duration is listed in milliseconds).
- Use judgment here – email notification events are quick, so if you see one taking a few seconds, something is wrong.
- Check processed column with is empty filter to determine if there is an issue with the backlog.
- Group by name to see which scripts are firing the most and disable if they are not doing anything.



© 2014 ServiceNow All Rights Reserved

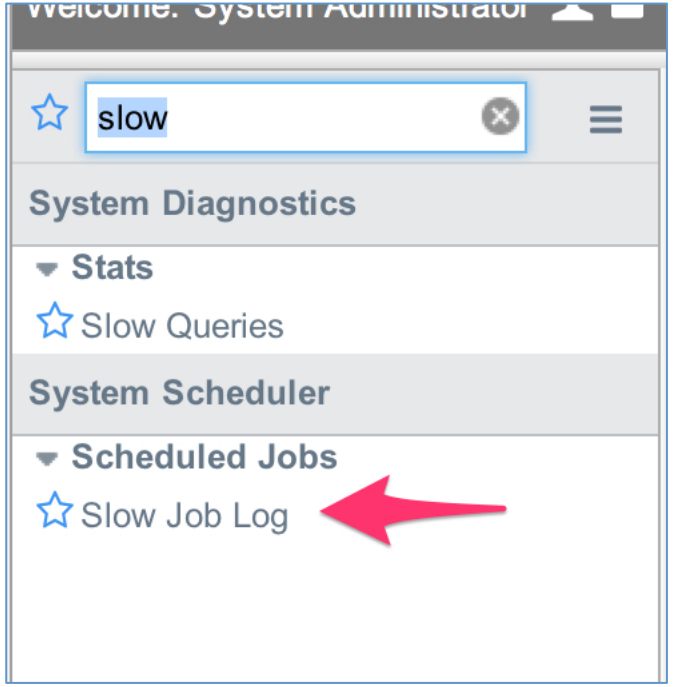
Slow Events Exercise

- Events Exercise
- http://wiki.servicenow.com/index.php?title=Logs_and_Queues_Best_Practices#Reviewing_Event_Logs

© 2014 ServiceNow All Rights Reserved

Slow Jobs– Job log (honorable mention)

- Use this to watch and identify scheduled jobs that are taking a long time.



© 2014 ServiceNow All Rights Reserved

Server Side - Takeaways

1. Use standards and best practices from the beginning.

2. Know how your system functions and be a constant observer of changes to that system using the tools discussed.

3. A wise man to me said, “Common Sense isn’t that common.” Use common sense and ask questions. My life motto is that I am constantly on a quest of discovery and learning. Use that philosophy when administrating your ServiceNow instance. Ask questions.

© 2014 ServiceNow All Rights Reserved

Client Side Debugging & Troubleshooting

How to identify and debug slow scripts?

© 2014 ServiceNow All Rights Reserved

Client Side Performance

- Smaller forms
  - Less data over the network and less to render
  - Less fields / less related lists / less client scripts
  - Use Views to structure
- Reduce round trips
  - Use display business rules
  - Bundle multiple Ajax call into one (JSON)
  - Avoid getReference() and query()
- Less client script, especially show / hide during load
- Avoid custom, not supported libraries

- Blocking Ajax calls
  - getReference() – allows callback, but always retrieves whole object, more network traffic
  - GlideRecord.query() – no callback option, can retrieve up to 500 records, should not be used
  - GlideAjax – allows both, execution of server side code and return of multiple structured values, best option
- Browsers make a big difference
  - IE6/IE7/IE8 are much slower due to old JS engine
- Avoid UI Scripts at any time
- Leverage onDisplay Business rules

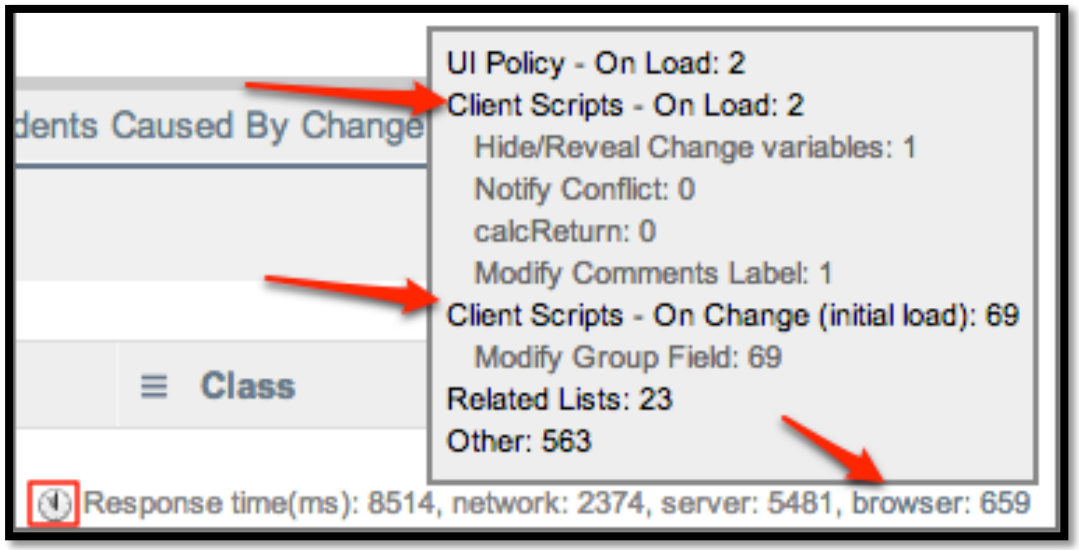
© 2014 ServiceNow All Rights Reserved

Client transaction timings

- Indicator can be found on every list and form in the bottom right
- Provides information where most of the time during a transaction was spent

- Useful to track down:
  - Long running server round trips
  - Optimization of loading times
  - Script who consume long times

- According to best practices, the total browser runtime should not exceed 3 seconds

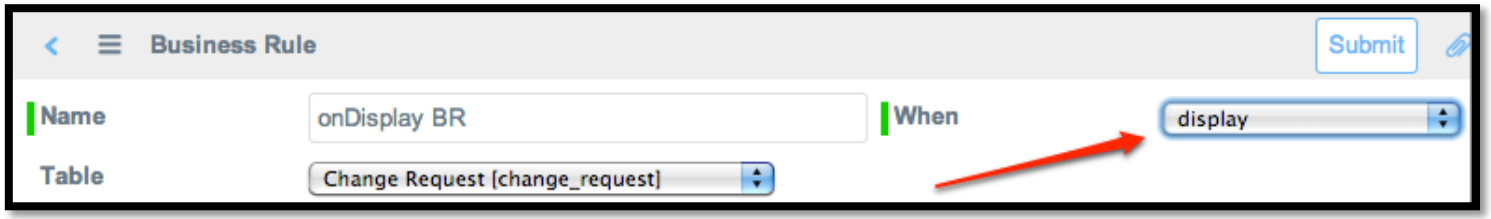


© 2014 ServiceNow All Rights Reserved

onDisplay Business Rules & Scratchpad

- onDisplay Business Rule will be processed when a user requests a form to load
  - Execution:
    - Data is read from the database
    - onDisplay rules are executed
    - Form will be presented to the user
  - Field changes are temporary and not yet submitted to the DB
  - To the client values appear to be the values from the DB
    - No indication that the values were modified
    - Similar concept to calculated fields

- g_scratchpad is sent to the client as part of the form
  - Available for use in DisplayBR to pass additional data
  - Will be passed as String value to the client as an object
  - Can be used in combination with JSON to return complex name/value structures



© 2014 ServiceNow All Rights Reserved

Client Performance Exercise

- What is the code doing? What could be wrong with this code?
- What could we do better? Discuss ideas!
- Go to the Change Management app and investigate

```
function onChange(control, oldValue, newValue, isLoading, isTemplate)
{
   var gr = new GlideRecord('sys_user_grmember');
   gr.addQuery("group", newValue);
   gr.query();

   g_form.clearOptions("u_group_member");

      while(gr.next())
      {
         var gruser = new GlideRecord('sys_user');
         gruser.addQuery("sys_id", gr.user);
         gruser.query();

         if(gruser.next())
         {
            g_form.addOption("u_group_member", gruser.sys_id, gruser.name)
         }
      }
}
```

© 2014 ServiceNow All Rights Reserved

Code Analysis & Suggestions

- What happens in the script?
  - #1: OnLoad or OnChange of the assignment group the script will be triggered
  - #2: It will retrieve the members for a group and for each member queries a different table for their names and adds them into a choice field
- How can we make it better?
  - Add comments for better readability
  - Make use of onDisplay BR & scratchpad
  - Instead of making 1 + Members server round trips, make just one by using GlideAjax
- Observe Client Transaction Timings

- Suggestions:
  - #1: OnLoad:
    - Use onDisplay BR, which retrieves values upfront and puts them to the scratchpad
    - Leverage the values from the scratchpad onLoad to fill the choice field without any round trip
  - #2: OnChange:
    - Refactor into an asynchronous GlideAjax call
    - Only one server round trip would be required and the browser is not blocked
    - Design the GlideAjax script include maintainable and debug-able by using GSLog

© 2014 ServiceNow All Rights Reserved

Browser DevTools & Debugging

- What is it?
  - Built in browser utilities to debug, author, inspect and develop
  - Is used to track down layout issues, set debugging breakpoints and gain insights in code optimisation
  - Most common features:
    - Inspection the DOM and styles
    - Debugging JavaScript
    - Improving/trace network performance
    - Improving render performance
    - Profiling for JavaScript & CSS performance
    - Inline inspection with the console
  - Available for all modern browsers (Chrome, Firefox, IE, Opera)

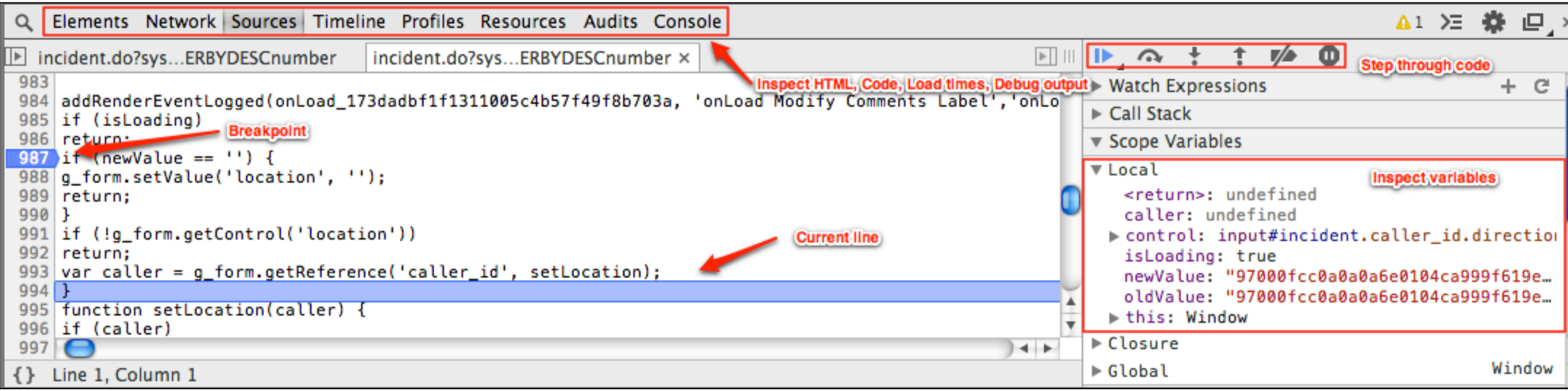
- Further information:
  - Chrome:
    - https://developers.google.com/chrome-developer-tools/
  - Firefox:
    - https://developer.mozilla.org/en-US/docs/Tools
  - Internet Explorer
    - http://msdn.microsoft.com/en-us/library/dd565628(vs.85).aspx
  - Safari
    - https://developer.apple.com/safari/tools/

© 2014 ServiceNow All Rights Reserved

Browser DevTools & Debugging - Exercise

- Fix code by using the build-in browser debugger
  - Search for a specific piece of code
  - Inspect the console output
  - Set breakpoints
  - Watch the variables behavior

- Browser shortcuts:
  - Chrome: Alt + CMD + i
  - Firefox: Alt + CMD + s
  - Internet Explorer: F12
  - Safari: Alt + CMD + i



© 2014 ServiceNow All Rights Reserved

Coding Patterns

Structure, Standards, and Patterns

© 2014 ServiceNow All Rights Reserved

Coding Patterns

- Common Sense Pattern (review)
  - Avoid superfluous code
  - Don’t use sleep in your code
  - Avoid global namespace collisions
- Single variable pattern and the hoisting problem
- Optimize loops
- Memoization pattern
- Throwing errors – you can throw any object
- Credit to Stoyan Stefanov from Yahoo! His book JavaScript Patterns.

© 2014 ServiceNow All Rights Reserved

Single Variable Pattern and Hoisting

- Avoid global namespace collisions by wrapping code in function blocks
- Declare variables with var

- Hoisting
  - JavaScript has function level scope but not block level scope, as you get with if statements for example in Java.
  - Function declarations and variable declarations are always moved (“hoisted”) invisibly to the top of their containing scope by the JavaScript interpreter.
- Single variable pattern – declare all variables at the top of function.

© 2014 ServiceNow All Rights Reserved

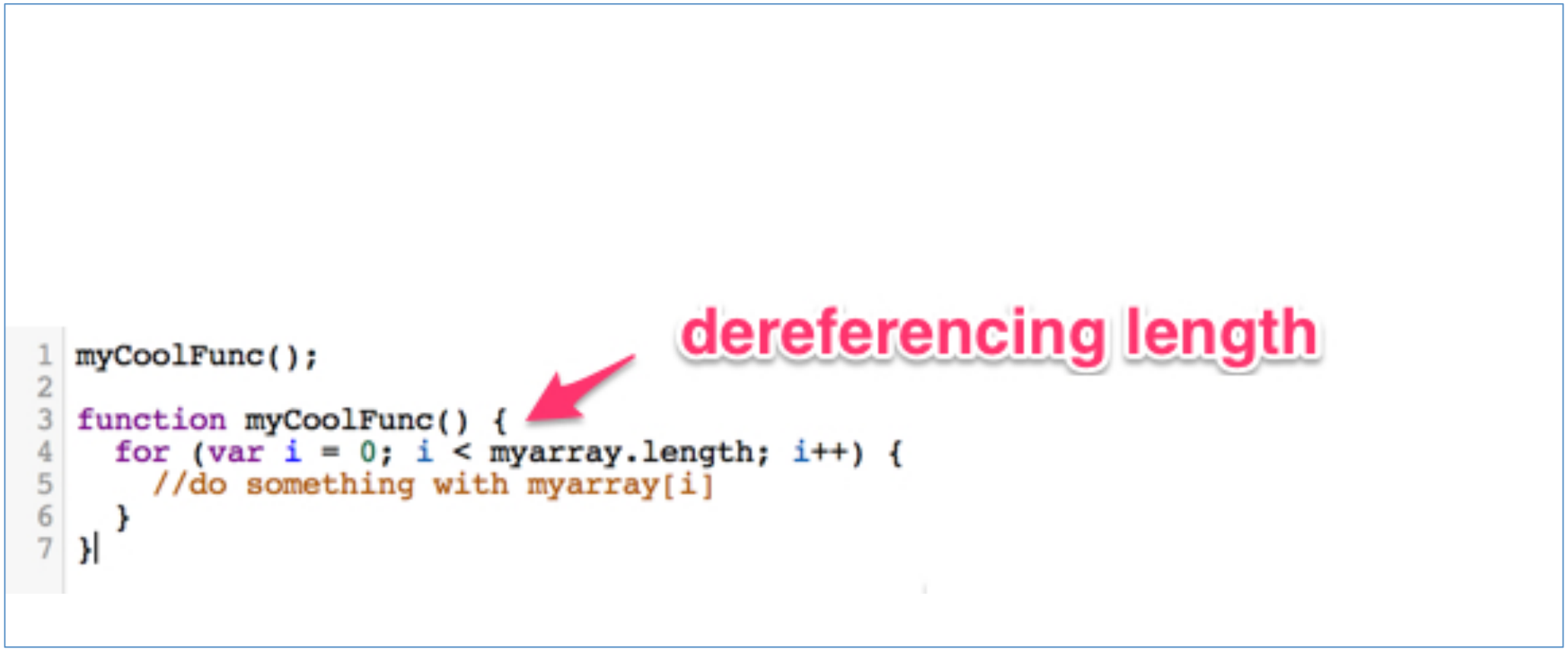
Single variable Pattern and hoisting

```
1  myCoolFunc();
2
3  function myCoolFunc() {
4    var x = 1,
5    y = 2,
6    sum = x + y,
7    obj = {},
8    j;
9
10   //function body
11 }
12
13 function dontDoThis() {
14   var x = y = 0;
15 }
16
```

y is global defined

© 2014 ServiceNow All Rights Reserved

Looping Pattern




```
1 myCoolFunc();
2
3 function myCoolFunc() {
4   for (var i = 0; i < myarray.length; i++) {
5     //do something with myarray[i]
6   }
7 }
```

© 2014 ServiceNow All Rights Reserved

Looping Pattern

```
1  myCoolFunc();
2
3  function myCoolFunc() {
4    var i = 0,
5    myarray = [];
6
7    for(i = myarray.length; i--;) {
8      //do something with myarray[i]
9    }
10 }
11
12 //or
13
14 function myCoolFunc() {
15   var myarray = [],
16   i = myarray.length;
17
18   while (i--) {
19     //do something with myarray[i]
20   }
21 }
```

single var pattern

© 2014 ServiceNow All Rights Reserved

Memoization Pattern

```
1  # Soyan Dtefanov example memoizing
2  var myFunc = function(param) {
3    if (!myFunc.cache[param]) {
4      var result = {};
5      //expenseive operation
6      myFunc.cache[param] = result;
7    }
8    return myFunc.cache[param];
9  }
```



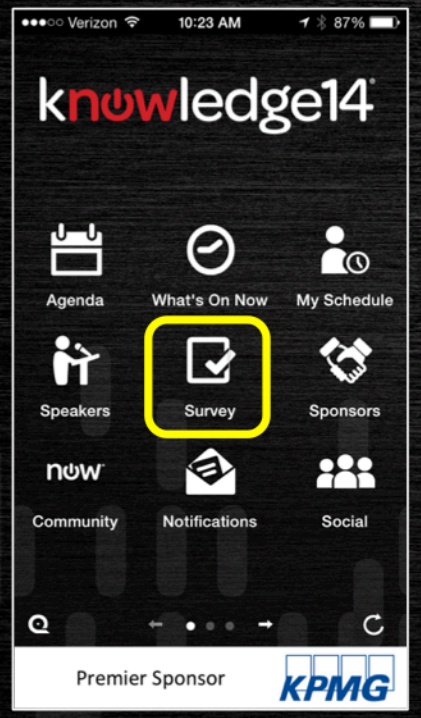
© 2014 ServiceNow All Rights Reserved

Additional Resources

- Client Script Best Practice
  - https://wiki.servicenow.com/index.php?title=Client_Script_Best_Practices
- Coding Best Practice:
  - https://wiki.servicenow.com/index.php?title=Coding_Best_Practices
- Browser DevTools
  - http://devtoolsecrets.com/
- System Performance Best Practice
  - https://wiki.servicenow.com/index.php?title=System_Performance_Best_Practices
- Debugging Tools Best Practice
  - https://wiki.servicenow.com/index.php?title=Debugging_Tools_Best_Practices
- Logs and Queues Best Practice
  - https://wiki.servicenow.com/index.php?title=Logs_and_Queues_Best_Practices

© 2014 ServiceNow All Rights Reserved

Tell us what you think!



Click on the survey link in your Knowledge14 mobile app. Help us make Knowledge 15 even more valuable!

Thank You

Chris Maloy

Team Lead

Expert Services

ServiceNow

chris.maloy@servicenow.com

Matt Gaide

Sr. Technical Consultant

Expert Services

ServiceNow

matt.gaide@servicenow.com

Oliver Schmitt

Team Lead Central Europe PS

ServiceNow

oliver.schmitt@servicenow.com

Want a copy of this deck?

Sign into the BuzzHub to unlock all the slides and lab guides from Knowledge14 sessions

