QUICK REFERENCE: K2 LOGGING AND TROUBLESHOOTING

K2 Server and hosted services

Logging/Troubleshooting tool	Notes	Learn More
K2 Host Server Console Mode	Run K2 Host Server in console mode to see real-time activity in K2 Host Server. Useful to troubleshoot performance issues, authentication issues and issues when K2 service fails to start. ***Hint:*** To avoid permission inconsistencies, run the K2 Host Server console under the SAME user account that is used to run the service. Use the ***K2 blackpearl Server (Service Account)*** shortcut under the "Program Files" menu to launch K2 Host server in console mode under the service account context. Remember to stop the K2 blackpearl service first!	**Video:** Using K2 host server console mode http://help.k2.com/en/displaycontent.aspx?id=6236
K2 Host Server Logging	Enable and configure logging on the K2 blackpearl server to output different levels of logging information to different target locations. Level of logging and active log locations are determined by the settings in the logging configuration file: *[Program Files (x86)]\K2 blackpearl\Host Server\Bin\HostServerLogging.config* ***Hint:*** For all environments it is recommended to enable "Error" level debugging to the windows event log so that any runtime errors are easily identifiable.	**KB:** Host Server Logging http://help.k2.com/en/kb000309.aspx **KB:** K2 Auditing and Logging http://help.k2.com/en/kb000298.aspx
Active Directory User Manager error logs	Errors encountered when interacting with Active Directory are logged in separate files. Log files' location and name are something like this: *[Program Files (x86)]\K2 blackpearl\Host Server\Bin\AdumErrornnnnnn_n.log* ***Note:*** the Logging configuration settings are stored in the K2 database – see the KB article for more information	**KB:** Configuration options for Active Directory User Manager message logging http://help.k2.com/en/kb001286.aspx **Links:** How to enable Kerberos event logging
Windows Event Log	The windows event log may contain useful troubleshooting information, especially if some external issue is causing K2 to behave unexpectedly. Remember to check the Application, Security and System logs. In distributed environments, you may need to check the windows event log for more than one server and other servers (such as the SQL server) as well. ***Note:*** K2 logging can be configured to output information to the Windows Event Log.	*[No additional content for this item]*
SQL troubleshooting: ▪ SQL Logging ▪ SQL Server tracing ▪ DTC ping	It may be necessary to troubleshoot the underlying K2 databases in MS-SQL. SQL logging may contain useful troubleshooting information. These logs are available from the Windows Application log or in a directory like *[Program Files\Microsoft SQL Server\Mssql]\Log* In rare cases, you may need to run SQL traces on the K2 database(s) or to identify issues, or troubleshoot issues with the Distributed Transaction Coordinator (MS- DTC). ***Note:*** SQL tracing is a technical procedure and should only be performed by a suitable skilled SQL DBA. Also, note that direct K2 database access is not supported nor recommended by K2. If you have any doubts, please log a K2 support ticket.	**Links:** Create a trace (SQL Server Profiler) How to troubleshoot connectivity issues in MS DTC by using the DTCPing tool
IIS Logging	When troubleshooting K2's hosted services and K2 web services, some useful information may be found in IIS logfiles.	**Links:** IIS 6.0 Logging overview Configuring Logging in IIS 7 Log Files for a Web Site (IIS 7)

SmartObjects

Logging/Troubleshooting tool	Notes	Learn More
SmartObject Runtime Logging	Enable SmartObject runtime logging by changing the <logging> settings in the file [Program Files (x86)]\K2 blackpearl\Host Server\Bin\SourceCode.SmartObjects.Runtime.config Run SmartObject methods to output logging information. Log data is output to the directory specified in the SourceCode.SmartObjects.Runtime.config file. • *brokerpackagein.log*: Logs the SmartObject that called the broker and the method of the SmartObject that uses the Broker. This information is useful to see which SmartObject and method were executed, when. • *brokerpackageout.log*: Number of records that were returned by the broker. This data can be useful when you need to troubleshoot why not the number of expected records was returned to the UI. • *serviceauth.log*: Security credentials used when the service was executed. Check the current thread user to see which user account executed the service broker code. • *servicepackagein.log*: Service Instance configuration, the Service Objects for the Service Instance that were called along with their Methods and Properties, the Mappings that define how the Provider Types are mapped to Service Object Types. When troubleshooting, one of the most useful values is probably the *<serviceauthentication>* section, where you can see the user account context that was used to query the provider. • *servicepackageout.log*: "raw" data returned by the Service Instance. This information can be useful when the data which eventually surfaces to the UI does not match the data you were expecting. Check this log file to see if the expected data was returned to the SmartObject, in which case the problem is likely on the UI side.	**Video:** SmartObject Runtime logging http://help.k2.com/en/displaycontent.aspx?id=6242
SmartObject Service Tester Utility	Located by default in [Program Files (x86)]\K2 blackpearl\bin\SmartObject Service Tester.exe Use this tool to run SmartObject methods directly without UI or workflow layers, construct ADO.NET queries, update Service Instances, generate code for SmartObject and much more.	**Video:** Using the SmartObject Service Tester utility for troubleshooting http://help.k2.com/en/displaycontent.aspx?id=6243
Provider logging	You may need to investigate the logs supplied by the data provider that the SmartObject Broker is using. The available logging will depend on the Provider being used, so check the providing technology's documentation.	**Poster:** K2 SmartObject architecture http://help.k2.com/K2LP-300.BNE.aspx

©SourceCode Technology Holdings, Inc. All Rights Reserved

SmartForms

Logging/Troubleshooting tool	Notes	Learn More
SmartForms runtime logging	Add a query string parameter to the Form/View URL in Run mode to start logging runtime information for a Form or View. Query String Parameter: _debug=*N* where *N* is a number from 1 to 3. 1 = most verbose logging, 3 = least verbose logging.	**Video:** How to investigate SmartForms performance issues http://help.k2.com/en/displaycontent.aspx?id=5994 **Video:** Runtime logging in K2 smartforms http://help.k2.com/en/displaycontent.aspx?id=6241
SmartForms Logging configuration	Change logging settings in the SmartForms web.config file to enable logging output and specify logging target. <add key="Forms.Global.ErrorLogging.Enabled" value="true" /> <add key="Forms.Global.ErrorLogging.FilePath" value="C:\Debug\SmartForms.Web.error" />	**Help:** K2 smartforms Developers Reference http://help.k2.com/helppages/K2smartforms1.02DevRef/web_config_Debugging.html
SmartForms Runtime Errors logfile	Server-side runtime errors in SmartForms will be logged in the following logfile: *[Program Files (x86)]\K2 blackpearl\Host Server\Bin\SourceCode.Forms.Runtime.Error*	*[No additional content for this item]*
IIS Logfiles	The standard IIS logfiles can be useful when troubleshooting SmartForms performance, authentication and missing file issues. These logfiles are normally located at: *C:\inetpub\logs\LogFiles\W3SVC**nnn*** where ***nnn*** is the web site ID of the IIS website that is hosting the SmartForms Designer/Runtime site. Use IIS Manager to determine the site ID of the SmartForms websites.	**Links:** IIS 6.0 Logging overview Configuring Logging in IIS 7 Log Files for a Web Site (IIS 7)
Browser-based debugging tools and Network traffic analyzers	Browser-based debugging tools like FireBug and dedicated network traffic analyzer tools like WireShark or Fiddler can be useful when deep troubleshooting performance, authentication issues or other hard-to-pin-down issues. ***Hint:*** disable minified .js files if you need to debug the standard JavaScript files by editing the web.config file for each SmartForms website and changing **UseBundledFiles** to **false**	**Links:** FireBug (Firefox) Introduction to F12 developer tools (IE) WireShark Fiddler
Windows Event Log	The windows event log may contain useful troubleshooting information. Remember to check the Application, Security and System logs. In distributed environments, you may need to check the windows event log for more than one server and other servers (such as the SQL server) as well.	*[No additional content for this item]*

Workflows

Logging/Troubleshooting tool	Notes	Learn More
K2 Workspace: ▪ Error Profiles and Retry K2 Process Portals: ▪ Error Logs ▪ Instance overview and Retry	Unhandled Workflow runtime errors are reported in the "Error Profiles" section of K2 workspace and "Error Logs" in K2 process portals. You can Retry workflows in error state using K2 Workspace, K2 Process Portal or K2 Process Management in Visual Studio. ***Note:*** Retry will execute the same event/activity rule/line where the failure occurred. If the event performs multiple actions or custom code, check whether it is safe to issue a Retry statement before doing so. Retry will not restart the entire workflow or go back to a previous step of the workflow	**Video:** Error Profiles and Retrying errors http://help.k2.com/en/displaycontent.aspx?id=6238
K2 Process Management in Visual Studio	Use K2 Process Management in Visual Studio to download and repair workflows in Error State. You can optionally apply the repair to all instances of the same workflow that are in currently in Error State, or repair each error separately. This approach is often used to fix configuration or coding errors in a workflow definition. ***Note:*** Error Repair results in a new workflow definition published to the K2 server. This workflow definition will automatically become the Default version. You may need to reset the Default Version for the workflow after repairing an error.	**Video:** Using K2 process management in Visual Studio http://help.k2.com/en/displaycontent.aspx?id=6239
Debug workflows with Visual Studio	You may use Remote Debugging in Visual Studio to attach to a K2 server process and debug workflow code. ***Note:*** Attaching a debugger and hitting a breakpoint with subsequent stepping-through code will suspend all other execution in the K2 host server. Use remote debugging with extreme caution in shared environments.	**Video:** Debugging workflows with Visual Studio http://help.k2.com/en/displaycontent.aspx?id=6237
Workflow Exception handling	When using Visual Studio, developers may write custom exception handling code for events, activities, lines and processes. Any unhandled exceptions will place the workflow into Error state and the error will be reported to the Workflow Error Profiles. ***Hint:*** if you Enable an exception rule but do not enable "Log exception to the server" or "Log exception to the error log", the error will be "swallowed" and the workflow will continue executing. This approach is not recommended unless you want to ignore errors for specific events/lines/activities. ***Best Practice:*** Workflow developers should implement a consistent and global exception handling approach for K2 workflows.	**Links:** Generic Send Mail K2 Exception Handler (K2 UG)
Workflow Reports, View Flow and Process Instances list	The Process Instances report and View Flow report are the best reports to use when troubleshooting workflow issues. The Process Instances view in K2 workspace can also be used to list processes by status and review errors	**Video:** Using reports to troubleshoot workflows http://help.k2.com/en/displaycontent.aspx?id=6240
Workflow Versioning	Use K2 Workspace to manage the Default Version of a workflow, download the definition (source) for any version of a workflow or delete workflow versions. ***Note:*** It is possible to migrate in-flight workflow instances with the workflow management API. See "Live Instance Management" in the K2 product documentation or a community tool like the Process version migration utility	**Video:** Understanding workflow versioning http://help.k2.com/en/displaycontent.aspx?id=6244

©SourceCode Technology Holdings, Inc. All Rights Reserved

SharePoint and InfoPath integration

Logging/Troubleshooting tool	Notes	Learn More
SharePoint Diagnostic Logging	When troubleshooting K2 integration with SharePoint, configure SharePoint diagnostic logging and review the SharePoint log files for useful troubleshooting information.	**Links:** Configure diagnostic logging (SharePoint 2010) Configure diagnostic logging (SharePoint 2013)
InfoPath debugging and diagnostic logging	InfoPath Forms Services forms do not always return useful troubleshooting information in the errors that surface to the user. To troubleshoot these issues, configure InfoPath diagnostic logging or debug the InfoPath form using standard Microsoft tools (details are in the "learn more" section. You can also force InfoPath to open the form in the thick-client application by changing the *DefaultItemOpen* query string parameter to 1. That will open the form in thick-client which may help with troubleshooting.	**Links:** Investigating Issues with InfoPath 2010 Applications Troubleshooting InfoPath Forms Services Errors
Windows Event Log	The windows event log on SharePoint WFE, Central Admin, SQL or other SharePoint servers may contain useful troubleshooting information.	*[No additional content for this item]*
IIS Logfiles	The IIS logfiles on the SharePoint WFE server(s) may contain useful troubleshooting information, especially if you need to troubleshoot authentication issues.	**Links:** IIS 6.0 Logging overview Configuring Logging in IIS 7 Log Files for a Web Site (IIS 7)
Disable custom errors in web.config	To show the "real" SharePoint error in the browser, disable custom errors in the web.config file for the WFE server(s). Remember to change the settings back once you are done troubleshooting.	**Links:** MSDN: Debugging SharePoint Solutions Disable Custom Error Mode In SharePoint 2010
Claims-based authentication troubleshooting	Claims Viewer web parts can be very useful when troubleshooting SharePoint environments where Claims-Based authentication (CBA) has been installed. You may use community-provided web parts or the K2 Claims Viewer Web Part Several Microsoft resources are also available to troubleshoot Claims-Based authentication: see the links on the right.	**Links:** SharePoint 2010 – Claims Viewer Web Part SharePoint 2013: Claims-Based Authentication Claims authentication does not validate user

Custom applications *(e.g. ASP.NET applications or thick-client applications)*

Note: *since K2 does not control or own custom code, you will need to identify and use whatever troubleshooting tools are available for the technologies used in your application. We have listed some common troubleshooting tools below, but this is by no means and exhaustive list of troubleshooting resources for custom applications*

Logging/Troubleshooting tool	Notes	Learn More
IDE debugging tools	Use the available debugging tools for your Integrated Development Environment to troubleshoot custom code.	*[No additional content for this item]*
IIS Logfiles	The standard IIS logfiles can be useful when troubleshooting web app performance, authentication and missing file issues. These logfiles are normally located at: *C:\inetpub\logs\LogFiles\W3SVC**nnn*** where ***nnn*** is the web site ID of the IIS website that is hosting your application.	**Links:** IIS 6.0 Logging overview Configuring Logging in IIS 7 Log Files for a Web Site (IIS 7)
Browser-based debugging tools and Network traffic analyzers	Browser-based debugging tools like FireBug and dedicated network traffic analyzer tools like WireShark or Fiddler can be useful when deep troubleshooting for performance or authentication issues is required.	**Links:** FireBug (Firefox) Introduction to F12 developer tools (IE) WireShark Fiddler
Windows Event Log	The windows event log may contain useful troubleshooting information. ***Best practice:*** Developers should implement and document a consistent error logging mechanism in their custom applications. Many organizations use a standardized logging framework or log messages to the Windows Event Log.	*[No additional content for this item]*
K2 logging	You may need to use the standard K2 logging resources mentioned throughout this document to troubleshoot your custom code's integration with K2.	*[See resources listed in the other sections of this document]*

Custom K2 Extensions

Logging/Troubleshooting tool	Notes	Learn More
Custom extensions hosted by K2 server: ▪ Custom SmartObject Service Broker ▪ Custom Logging Provider ▪ Custom Security Provider ▪ Custom Inline Function ▪ Custom Notification Provider ▪ Custom Logging Provider ▪ Custom Event recorder	Use Visual Studio remote debugging to debug the code for any custom extensions hosted by the K2 Host Server, such as custom Service Brokers, Logging Providers, Security Providers, Event Recorders, Notification Providers and so on. Remember to copy the .pdb file along with the .dll file to the K2 server so that the breakpoints will be hit. ***Best practice:*** *Developers* may import the *SourceCode.Logging* namespaces and write custom logging information which will use the standard K2 logging framework. Alternatively, all custom extensions should use a consistent and documented logging mechanism.	**Links:** Remote Debugging the K2 Host Server
Custom SmartForm Controls	Server-side code can be debugged by attaching a Remote Debugger to the *w3wp.exe* process of the web app pool that is hosting the website running the custom control (SmartForms Designer or SmartForms Runtime site). Remember to copy both the .dll and the .pdb file to the SmartForms server so that the breakpoints can be located. Client-side code (e.g. javascript or css issues) can be debugged using client-side debugging tools as described in the SmartForms debugging section earlier in this document.	*[See SmartForms section]*

©SourceCode Technology Holdings, Inc. All Rights Reserved

K2 connect

Logging/Troubleshooting tool	Notes	Learn More
K2 connect Administration Tools	Use the connect Administration Console (*[Program Files (x86)]\K2 connect\Admin\SourceCode.ServiceObjectModel.UI.Administration.exe*) to test connections to SAP Destinations and enable Protocolling (Logging) to output runtime logging information. To enable protocolling output, open the K2 connect Administration console's **Settings** window. Select the **Enable Protocolling** checkbox and restart the K2 connect service. Once you have restarted the K2 connect service, use the SmartObject Service Tester utility to execute the SAP-backed SmartObjects, and then open the **Protocol** window of the K2 connect administration console. Review the output of the Protocol window and see if any useful troubleshooting information was logged	*[No additional content for this item]*
K2 connect Test Cockpit	Use the K2 connect Test Cockpit in Visual Studio and execute the BAPI. Check the RETURN value to see if any errors were reported. ***Note:*** Test Cockpit uses the ERP Connect framework to connect to SAP directly: the K2 connect server is bypassed when you use Test Cockpit to execute SAP BAPIs	*[No additional content for this item]*
ERP Connect Test Code	If needed, write code using the ERPConnect adapter, which will exclude K2 connect from the equation. See the connect product documentation and ERP documentation for code samples	*[No additional content for this item]*
SAP GUI and ABAP Debugging	If available, use the SAP GUI to execute the BAPI to determine whether the problem is on the BAPI layer. You can also use ABAP debugging to troubleshoot BAPI-level errors	*[No additional content for this item]*
RFC Exception Trace file	ERP Connect will output a trace file with errors. This file can be located at *[Program Files\K2 connect\admin\dev_rfc.trc* or *[Program Files]\K2 blackpearl]\Host Server\Bin\dev_rfc.trc.*	*[No additional content for this item]*
Windows Event Log	The windows event log may contain useful information. Remember to check the Application and Security logs as well	*[No additional content for this item]*

Other Resources

Resource	Notes	Links
K2 customer and partner portal	Log support tickets, download product updates, review product compatibility information, provide product feedback, request K2 licenses	**Links:** http://portal.k2.com K2 blackpearl 4.6 Compatibility Matrix K2 smartforms Compatibility Matrix K2 connect Compatibility Matrix
K2 Help and K2 University	K2 Help includes Knowledge Base (KB) articles, technical whitepapers and the online version of the K2 product documentation. K2 University includes training videos, guides, posters and quick-reference sheets.	**Links:** http://help.k2.com http:/help.k2.com/K2U.aspx
K2 Underground	Forums, Blogs and community-supported components available on K2 Underground.	**Links:** http://www.k2underground.com K2 blackmarket
General troubleshooting tools	Some applications can be handy when troubleshooting, especially for editing text files, checking the validity of XML configuration files, comparing files or monitoring system activity. See the resources listed for some commonly-used tools.	**Links:** Notepad++ Process Monitor from Microsoft

©SourceCode Technology Holdings, Inc. All Rights Reserved