.NET Conf China 2025

改变世界 改变自己

2025 年 11 月 30 日 | 中国 上海

.NET Conf China 2025
改变世界 改变自己

应用程序异常管理

Juster Zhu 朱震
Microsoft MVP
GeneralUpdate Owner

- 异常的定义
- 异常的收集
- 异常分析

```
try
{
      //…
}
catch (Exception e)
{
    _logger.LogError(e.Message);
}
```

未将对象设置到对象引用的实例

超出索引范围

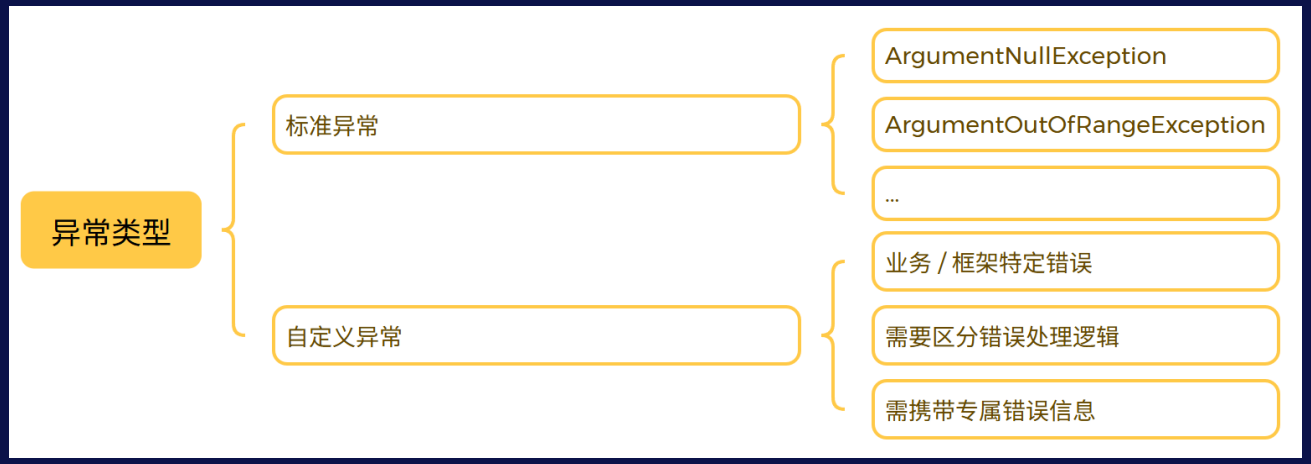
对象类型转换失败

…

如果产品初期在项目架构、设计时对异常不够重视，那么这样的产品一旦具有规模部署在客户那里，无异于亲手培养了一只吞噬成本的怪物。除了主要的业务实现，异常管理也非常重要。

例如：假设现在你的公司是一家做线下零售收银终端的企业，当产品部署在各个地区、各大超市等现场，处理异常的效率将显得极为重要，效率过低除了无法及时响应，还会导致不断堆积新的问题。现场的实施人员、技术支持反馈的信息组织起来非常低效（模糊的图文、残缺单一的日志、记忆混乱的复现步骤）。

异常的定义



异常抛出

- 选对异常类型，根据错误场景匹配最精准的异常（如空参数用ArgumentNullException，而非ArgumentException）；
- 携带有效信息：消息需明确 “什么错、为什么错、怎么改”（如throw new ArgumentOutOfRangeException("age", "年龄必须在1-120之间，当前值：-5")）；
- 保留原始异常：捕获异常后重新抛出时，必须将原始异常作为 “内部异常” 传入（如catch (SqlException ex) { throw new DataAccessException("数据库查询失败", ex); }），避免丢失错误根因。

设计原则

- 异常必须可序列化、结构化
- 保留堆栈信息（异常重抛）
- 使用 Inner Exception 包装低层异常（异常链）
- 异常等级，不同等级触发不同级别的报警和处理
- 明确语义
- 不包含敏感信息（密码、令牌、隐私数据）
- 不可变，异常属性（消息、元数据）创建后禁止修改（避免多线程状态混乱）

使用原则

- 不用于流程控制，异常只用于真正的异常情况。
- 面向开发者的“可诊断性”优先
- 使用 .NET 标准异常优先，减少自定义异常
- 每个异常代表一个稳定、可预期的失败语义
- 不吞噬异常，除非你能恢复，否则不要 catch。
- 就近有效处理：只在 “能解决 / 缓解错误” 的层级捕获（底层不处理无意义异常）
- 不捕获Exception（顶层异常），只捕获具体的、可预期的异常类型（防止掩盖未知错误）

异常传播

- 不覆盖异常：禁止在finally块中抛出新异常（会覆盖try/catch中已有的异常，导致原始错误丢失）；
- 文档化异常，注释说明、文档说明。
- 高层统一处理：框架级可在顶层（如 API 网关、应用入口）捕获未处理异常。
- 异常无法“直接跨线程传播”(线程内抛出的异常默认仅在当前线程生效)，需了解传播机制避免线程内吞噬。

异常捕获

- 内部
  - 局部异常捕获
    - try
  - 全局异常捕获
    - TaskScheduler.UnobservedTaskException
    - AppDomain.CurrentDomain.UnhandledException
- 外部
  - 健康检查框架
  - 工具（进程监控）

分类	字段名称	作用说明	常见用途	备注
基础信息	`timestamp`	异常发生时间（毫秒精度）	时间线还原、排序、关联调用链	推荐使用 UTC
	`level`	日志级别（Error/Fatal）	过滤异常日志	
	`event`	事件类型，通常为 "Exception"	可区分异常与普通日志	
异常主字段（Exception）	`exception.type`	异常类型（如 NullReferenceException）	错误聚合、Top N 异常统计	**最重要字段之一**
	`exception.message`	异常消息	排查原因	保留原文，不要截断
	`exception.stacktrace`	完整堆栈	定位代码位置	必须完整存储
	`exception.source`	抛出异常的程序集	判断组件/模块的问题来源	
	`exception.method`	抛出异常的方法名	快速锁定方法	运行时可自动解析
	`exception.hresult`	系统错误码	Windows/COM 异常分析	可选
inner exception（链路）	`exception.inner.type`	内层异常类型	分析根因	递归结构
	`exception.inner.message`	内层消息	重要细节来源	
	`exception.inner.stacktrace`	内层堆栈	底层错误定位	

业务上下文（Context）	`context.traceId`	分布式追踪 ID	全链路排查	**必须**，可与 APIM/网关关联
	`context.spanId`	调用链子跨度 ID	调用链片段定位	
	`context.requestId`	当前请求唯一 ID	HTTP 请求级诊断	
	`context.userId`	用户标识	识别影响范围	注意脱敏
	`context.tenantId`	多租户场景租户 ID	SaaS 环境排查	
	`context.request.url`	触发错误的 URL	分析接口错误率	
	`context.request.method`	GET/POST 等	过滤异常来源	
	`context.request.body`	输入参数	重现 / 排查输入问题	**注意脱敏敏感信息**
性能相关	`context.elapsedMs`	异常前操作耗时	诊断超时/性能问题	
	`context.cpu`	异常发生时 CPU 使用率	性能瓶颈定位	可选
	`context.memory.used`	进程内存	排查内存泄漏/溢出	
应用上下文（App）	`app.name`	应用名	多服务日志区分	
	`app.version`	构建版本/CI 编号	定位哪次部署引发问题	强烈推荐
系统环境（System）	`system.machine`	服务器机器名	多节点排查	
	`system.ip`	节点 IP	多机房排查	
	`system.os`	操作系统	容器化/跨平台排查	
	`system.processId`	进程 ID	找出具体实例	
	`system.threadId`	异常线程 ID	死锁、线程池不足分析	

异常的收集

异常的收集

Serilog

Serilog 是 .NET 生态中一款流行的**结构化日志库**，核心优势是将日志以键值对的结构化形式记录（而非传统纯文本），让日志更易检索、分析，广泛用于 .NET Framework、.NET Core、.NET 5+ 等项目。

Trace

Trace（跟踪）是System.Diagnostics命名空间下的核心工具类，主要用于**监控和记录应用程序的执行过程**，尤其适合在发布（Release）环境中追踪程序行为、诊断问题，而无需中断程序运行。

异常的收集

Using ProcDump

Capture Usage:

Windows Command Prompt Copy

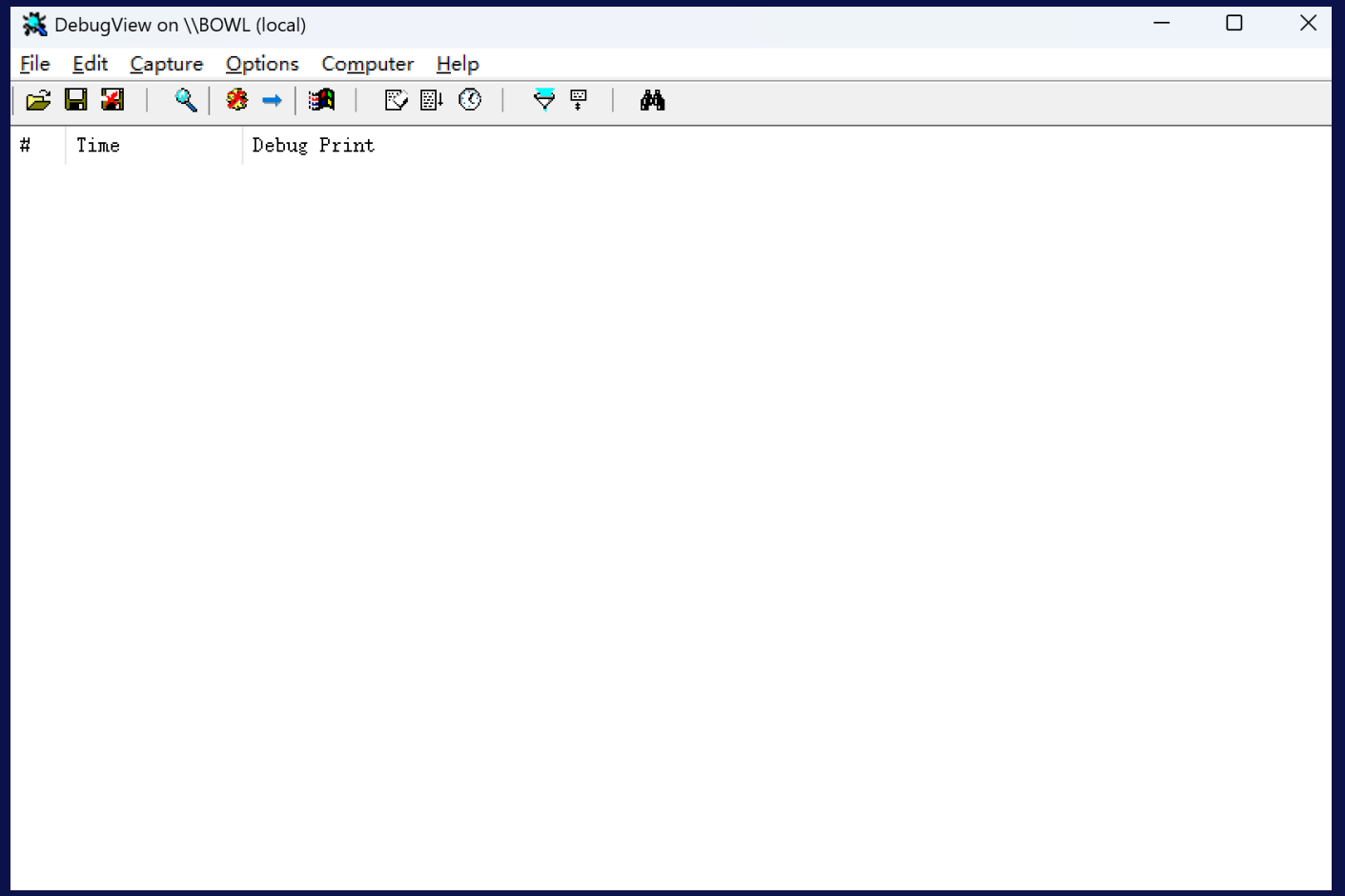
```
procdump.exe [-mm] [-ma] [-mt] [-mp] [-mc <Mask>] [-md <Callback_DLL>] [-mk]
             [-n <Count>]
             [-s <Seconds>]
             [-c|-cl <CPU_Usage> [-u]]
             [-m|-ml <Commit_Usage>]
             [-p|-pl <Counter> <Threshold>]
             [-h]
             [-e [1] [-g] [-b] [-ld] [-ud] [-ct] [-et]]
             [-l]
             [-t]
             [-f  <Include_Filter>, ...]
             [-fx <Exclude_Filter>, ...]
             [-dc <Comment>]
             [-o]
             [-r [1..5] [-a]]
             [-at <Timeout>]
             [-wer]
             [-64]
             {
                 {{[-w] <Process_Name> | <Service_Name> | <PID>} [<Dump_File> | <Dump_Folder>]}
             |
                 {-x <Dump_Folder> <Image_File> [Argument, ...]}
             }
```

辅助信息的收集

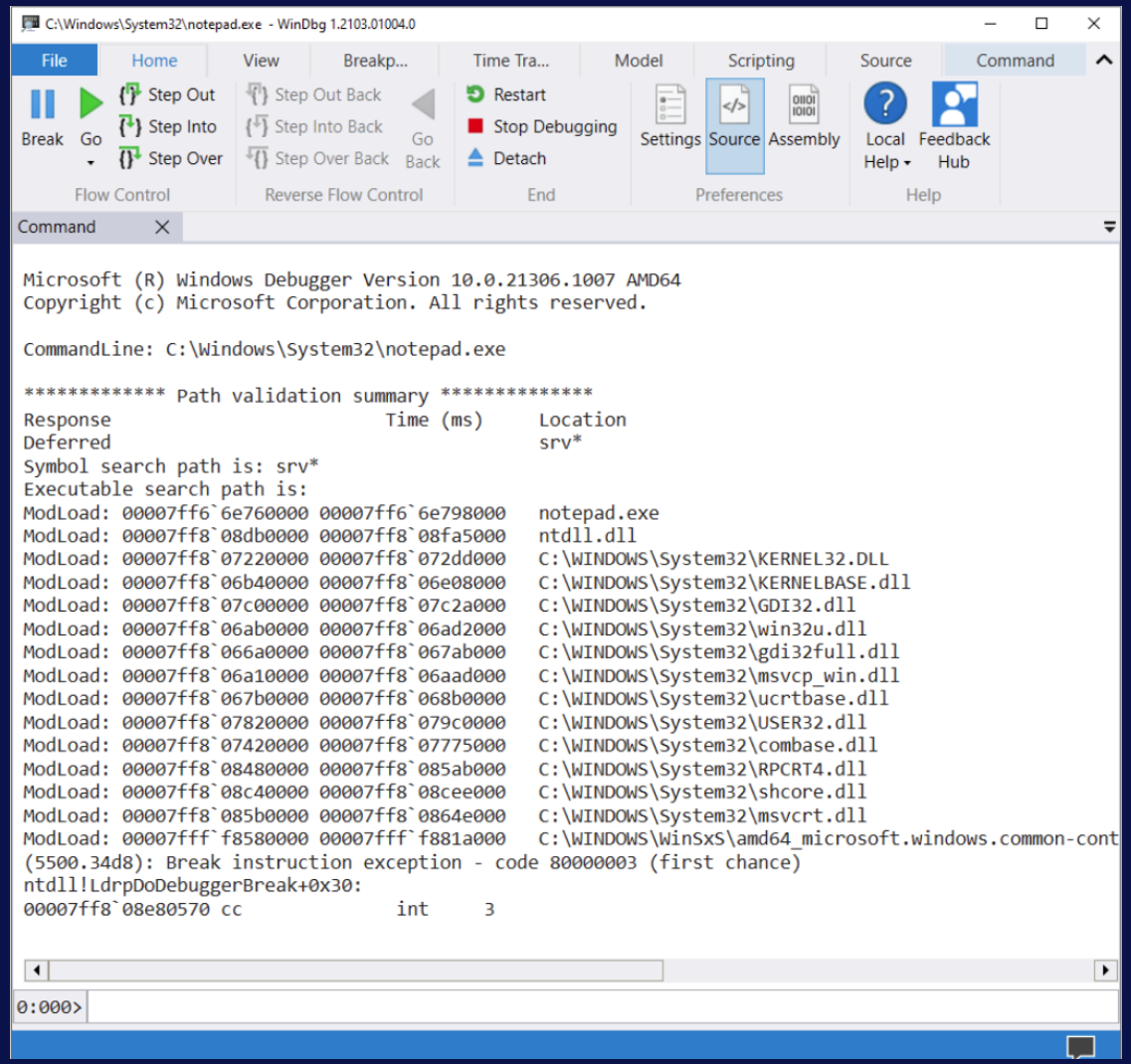
- 📒Dump文件（1.0.0.*_fail.dmp）
- 📒驱动信息（driverInfo）
- 📒操作系统信息/硬件信息（systeminfo）
- 📒系统事件日志（systemlog.evtx）

异常的解析

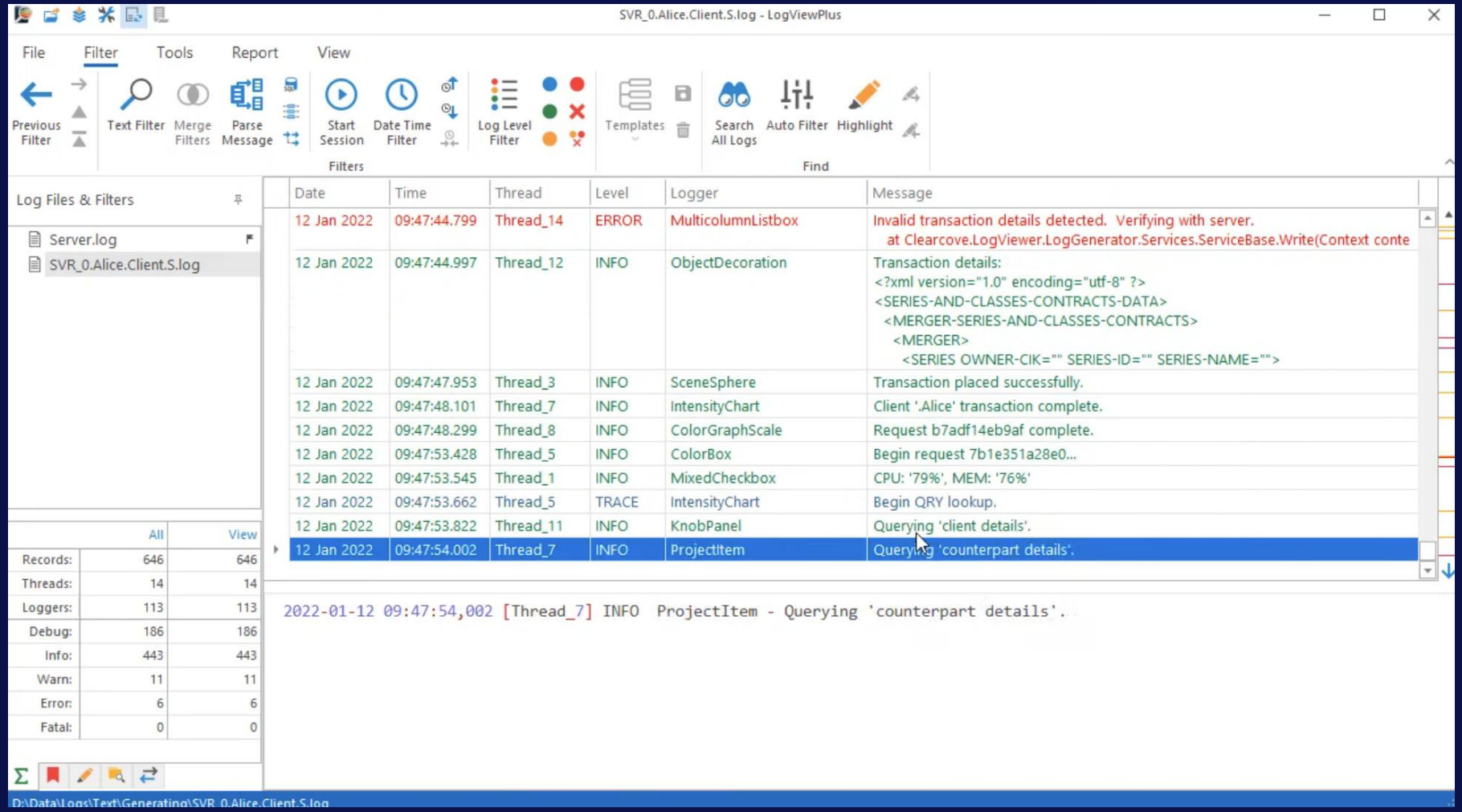
DebugView


DebugView on \\BOWL (local)
File Edit Capture Options Computer Help
Time Debug Print

WinDbg


C:\Windows\System32\notepad.exe - WinDbg 1.2103.01004.0
File
Home
View
Breakp...
Time Tra...
Model
Scripting
Source
Command
Break
Go
Step Out
Step Into
Step Over
Step Out Back
Step Into Back
Step Over Back
Go Back
Restart
Stop Debugging
Detach
Settings
Source
Assembly
Local Help
Feedback Hub
Flow Control
Reverse Flow Control
End
Preferences
Help
Command
Microsoft (R) Windows Debugger Version 10.0.21306.1007 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.
CommandLine: C:\Windows\System32\notepad.exe
************* Path validation summary **************
Response Time (ms) Location
Deferred srv*
Symbol search path is: srv*
Executable search path is:
ModLoad: 00007ff6`6e760000 00007ff6`6e798000 notepad.exe
ModLoad: 00007ff8`08db0000 00007ff8`08fa5000 ntdll.dll
ModLoad: 00007ff8`07220000 00007ff8`072dd000 C:\WINDOWS\System32\KERNEL32.DLL
ModLoad: 00007ff8`06b40000 00007ff8`06e08000 C:\WINDOWS\System32\KERNELBASE.dll
ModLoad: 00007ff8`07c00000 00007ff8`07c2a000 C:\WINDOWS\System32\GDI32.dll
ModLoad: 00007ff8`06ab0000 00007ff8`06ad2000 C:\WINDOWS\System32\win32u.dll
ModLoad: 00007ff8`066a0000 00007ff8`067ab000 C:\WINDOWS\System32\gdi32full.dll
ModLoad: 00007ff8`06a10000 00007ff8`06aad000 C:\WINDOWS\System32\msvcp_win.dll
ModLoad: 00007ff8`067b0000 00007ff8`068b0000 C:\WINDOWS\System32\ucrtbase.dll
ModLoad: 00007ff8`07820000 00007ff8`079c0000 C:\WINDOWS\System32\USER32.dll
ModLoad: 00007ff8`07420000 00007ff8`07775000 C:\WINDOWS\System32\combase.dll
ModLoad: 00007ff8`08480000 00007ff8`085ab000 C:\WINDOWS\System32\RPCRT4.dll
ModLoad: 00007ff8`08c40000 00007ff8`08cee000 C:\WINDOWS\System32\shcore.dll
ModLoad: 00007ff8`085b0000 00007ff8`0864e000 C:\WINDOWS\System32\msvcrt.dll
ModLoad: 00007fff`f8580000 00007fff`f881a000 C:\WINDOWS\WinSxS\amd64_microsoft.windows.common-cont
(5500.34d8): Break instruction exception - code 80000003 (first chance)
ntdll!LdrpDoDebuggerBreak+0x30:
00007ff8`08e80570 cc int 3
0:000>

LogViewer


SVR_0.Alice.Client.S.log - LogViewPlus
File
Filter
Tools
Report
View
Previous Filter
Text Filter
Merge Filters
Parse Message
Start Session
Date Time Filter
Log Level Filter
Templates
Search All Logs
Auto Filter
Highlight
Filters
Find
Log Files & Filters
Server.log
SVR_0.Alice.Client.S.log
Date
Time
Thread
Level
Logger
Message
12 Jan 2022 09:47:44.799 Thread_14 ERROR MulticolumnListbox Invalid transaction details detected. Verifying with server.
at Clearcove.LogViewer.LogGenerator.Services.ServiceBase.Write(Context conte
12 Jan 2022 09:47:44.997 Thread_12 INFO ObjectDecoration Transaction details:
<?xml version="1.0" encoding="utf-8" ?>
<SERIES-AND-CLASSES-CONTRACTS-DATA>
<MERGER-SERIES-AND-CLASSES-CONTRACTS>
<MERGER>
<SERIES OWNER-CIK="" SERIES-ID="" SERIES-NAME="">
12 Jan 2022 09:47:47.953 Thread_3 INFO SceneSphere Transaction placed successfully.
12 Jan 2022 09:47:48.101 Thread_7 INFO IntensityChart Client '.Alice' transaction complete.
12 Jan 2022 09:47:48.299 Thread_8 INFO ColorGraphScale Request b7adf14eb9af complete.
12 Jan 2022 09:47:53.428 Thread_5 INFO ColorBox Begin request 7b1e351a28e0...
12 Jan 2022 09:47:53.545 Thread_1 INFO MixedCheckbox CPU: '79%', MEM: '76%'
12 Jan 2022 09:47:53.662 Thread_5 TRACE IntensityChart Begin QRY lookup.
12 Jan 2022 09:47:53.822 Thread_11 INFO KnobPanel Querying 'client details'.
12 Jan 2022 09:47:54.002 Thread_7 INFO ProjectItem Querying 'counterpart details'.
All
View
Records: 646 646
Threads: 14 14
Loggers: 113 113
Debug: 186 186
Info: 443 443
Warn: 11 11
Error: 6 6
Fatal: 0 0
2022-01-12 09:47:54,002 [Thread_7] INFO ProjectItem - Querying 'counterpart details'.
D:\Data\Logs\Text\Generating\SVR_0.Alice.Client.S.log

总结

异常管理

良好的异常管理，增加了响应速度、弱化现场人员依赖、减少排查成本。

.NET Conf China 2025

改变世界 改变自己

THANK YOU