

Handling the Unexpected: Exception programming with



Jason Porter

twitter: @lightguardjp

Red Hat - Senior Software Engineer

Exception Handling



Exception Handling

exception

1. special conditions that change the normal flow of program execution

πρόσθ ειδικών εκθεσπρών
πρ ειδικών συνθηκών ππρ αμάρξο πρ νόμοι



Exception Handling

Dealing with these "special conditions" is
exception handling



Exception Handling

Q: What does this mean to us
developers?



Exception Handling

A: `try / catch`



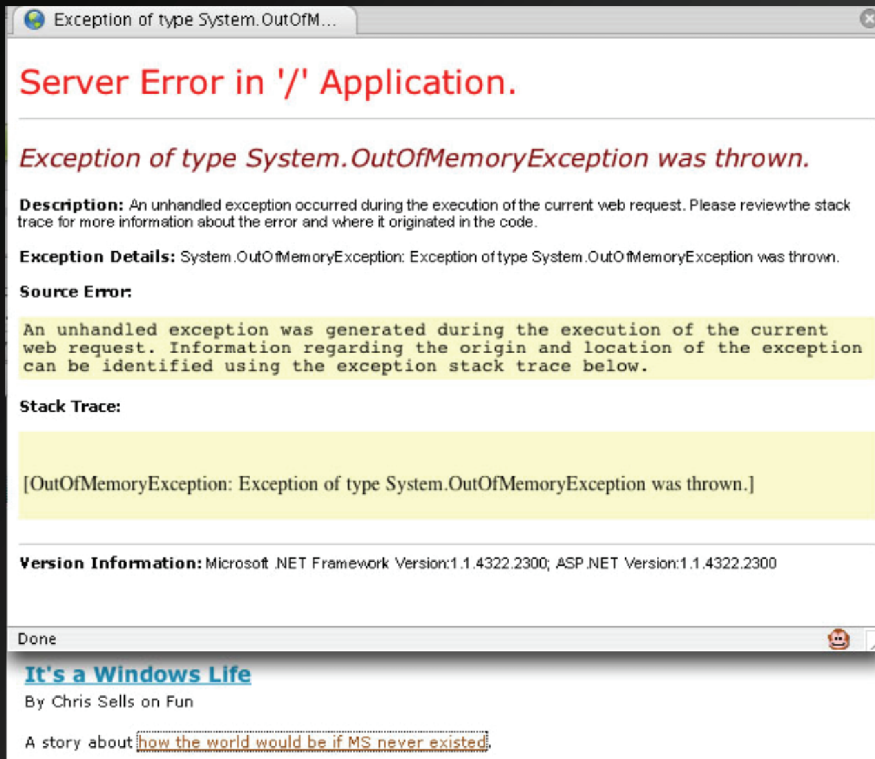
Exception Handling

A:

```
x libata dock scsi_mod [last unloaded: nvidia]
Pid: 15249, comm: setiathome-5.27 Tainted: P M (2.6.26.2 #2)
EIP: 0060:[<c013b532>] EFLAGS: 00010046 CPU: 2
EIP is at clockevents_program_event+0xf6/0x139
EAX: 27fe3473 EBX: 00000000 ECX: 000f2267 EDX: 00000000
ESI: 09443490 EDI: c3627140 EBP: c3627140 ESP: ee4c7f24
DS: 007b ES: 007b FS: 0048 GS: 0033 SS: 0060
Process setiathome-5.27 (pid: 15249, ti=ee4c6000 task=f569f390 task.ti=ee4c6000)
Stack: 09443490 000177cd c0136556 00064c0f 00000001 09443490 000177cd c3627140
c013c3b6 093e1229 000177cd c36280e0 c3628058 093dfb3e 000177cd c0136703
ee4c7f64 09443490 000177cd 093dfb3e 000177cd c3628028 c3628024 00000002
Call Trace:
[<c0136556>] ktime_get+0xd/0x2c
[<c013c3b6>] tick_program_event+0x38/0x5c
[<c0136703>] hrtimer_interrupt+0x18e/0x1bb
[<c0110ef9>] smp_apic_timer_interrupt+0x53/0x03
[<c01030b0>] apic_timer_interrupt+0x28/0x30
=====
Code: 00 0f 07 5e ff ff ff b8 c2 ff ff ff 00 c4 10 5b 5e 5f 5d 84 a2 00 00 00 00
c3 39 c8 ff 03 67 ff ff ff 0d b4 7e 00 00 00 00 eb ad <39> 00 8d 74 26 00 0f 86
45 00 ff ff 66 90 eb b0 a1 b4 5c 41 c0
EIP: [<c013b532>] clockevents_program_event+0xf6/0x139 SS:ESP 0060:ee4c7f24
Kernel panic - not syncing: Fatal exception in interrupt
```

Exception Handling

A:



Exception of type System.OutOfMemoryException

Server Error in '/' Application.

Exception of type System.OutOfMemoryException was thrown.

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.OutOfMemoryException: Exception of type System.OutOfMemoryException was thrown.

Source Error:

An unhandled exception was generated during the execution of the current web request. Information regarding the origin and location of the exception can be identified using the exception stack trace below.

Stack Trace:

[OutOfMemoryException: Exception of type System.OutOfMemoryException was thrown.]

Version Information: Microsoft .NET Framework Version:1.1.4322.2300; ASP.NET Version:1.1.4322.2300

Done

[It's a Windows Life](#)
By Chris Sells on Fun
A story about [how the world would be if MS never existed.](#)

Exception Handling

A:



Exception Handling Landscape

Common Language Features

try / catch

method declaration



Exception Handling Landscape

Web Applications

web.xml



Exception Handling Landscape

Framework Support

JAX-RS, JSF2,
Struts config, etc.



Exception Handling Landscape

Spring

Spring Exception
Resolvers



Exception Handling Landscape

Uncommon JDK Feature

Thread.UncaughtExceptionHandler



Exception Handling Landscape



A Better Way: Seam Catch

Catch allows you to build your own
handling programming model

Declarative
portable behavior
built on top of Java EE



A Better Way: Seam Catch

Catch allows you to build your own handling programming model

Smart, Contextual Handlers

Application State Injection

Specific or broad targeted handlers



A Better Way: Seam Catch

Based on standards: CDI Events

@Observes



A Better Way: Seam Catch

Based on standards: CDI Events

@Observes

```
void executeHandlers(@Observes @Any ExceptionToCatch payload,  
                    final BeanManager bm) {  
    // ...  
}
```



A Better Way: Seam Catch

Based on standards: CDI Events

Firing events



A Better Way: Seam Catch

Based on standards: CDI Events

Firing events

```
public class blah {
    @Inject Event<blah> event;
    @Inject BeanManager bm;

    void someMethod() {
        bm.fireEvent(new blah());
        // ...
        event.fire(new blah());
    }
}
```



A Better Way: Seam Catch

Exception unwrapping

including SQLException family!



A Better Way: Seam Catch

Exception filtering / mapping

Happens before exception handling,
after event firing

Able to change the exception



A Better Way: Seam Catch

Stack trace filtering

<https://gist.github.com/933229#gistcomment-27961>



A Better Way: Seam Catch

Stack trace filtering

"Flipping" the trace
Dropping elements
"Folding" elements
Re-writing the trace



A Better Way: Seam Catch

Easy integration

Uses CDI events

Already taken care of if using Seam
Servlet, Seam Faces, Seam Rest

Simply fire the event your self

Interceptor coming in next version



Anatomy of a Handler

```
@HandlesExceptions
public class StandardHandlers {
    public void rollbackTransaction(@Handles CaughtException<PersistenceException> event,
                                   UserTransaction tx) {
        try {
            tx.rollback();
        } catch (SystemException e) {
            // Something here
        }
        event.handled();
    }

    public void logExceptions(@Handles(during = TraversalMode.BREADTH_FIRST, precedence = Precedence.HIGH)
                              CaughtException<Throwable> evt, Logger log) {
        log.error("Exception caught: " + evt.getException().getMessage());
    }

    public void endConversation(@Handles CaughtException<MyBaseApplicationException> evt,
                                Conversation conv) {
        if (!conv.isTransient())
            conv.end();
    }
}
```



DEMO

