Handling the Unexpected: Exception programing with

Jason Porter

twitter: @lightguardjp

Red Hat - Senior Software Engineer

Exception Handling

Seam Catch

Exception Handling

exception

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

Seam Catch

Exception Handling

Dealing with these "special conditions" is exception handling

Exception Handling

Q: What does this mean to us developers?

SeamCatch

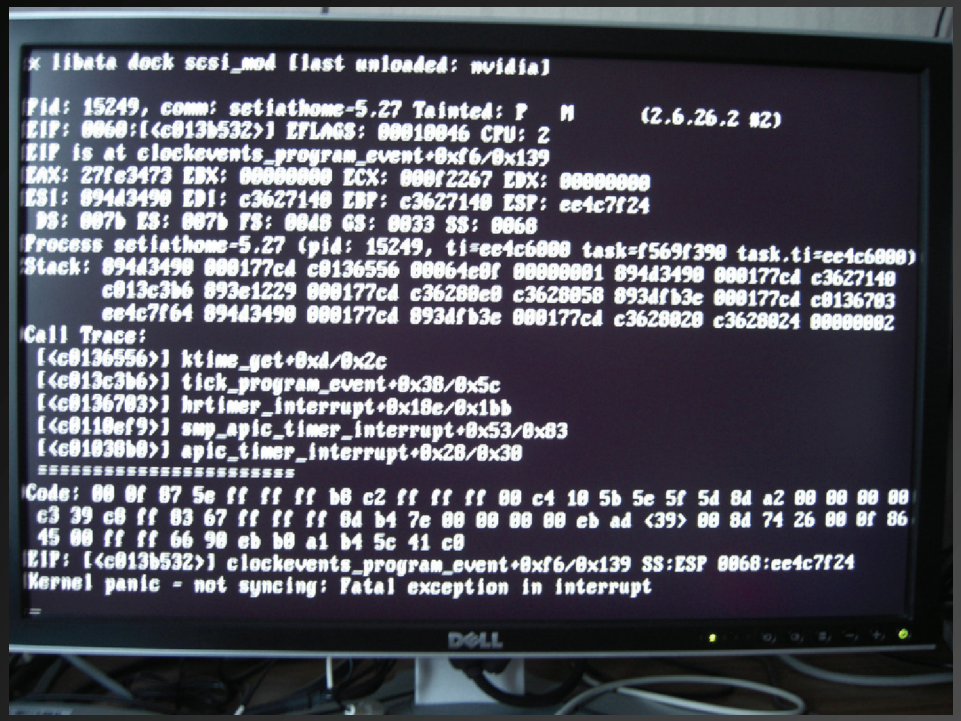
Exception Handling

A: try / catch

SeamCatch

Exception Handling

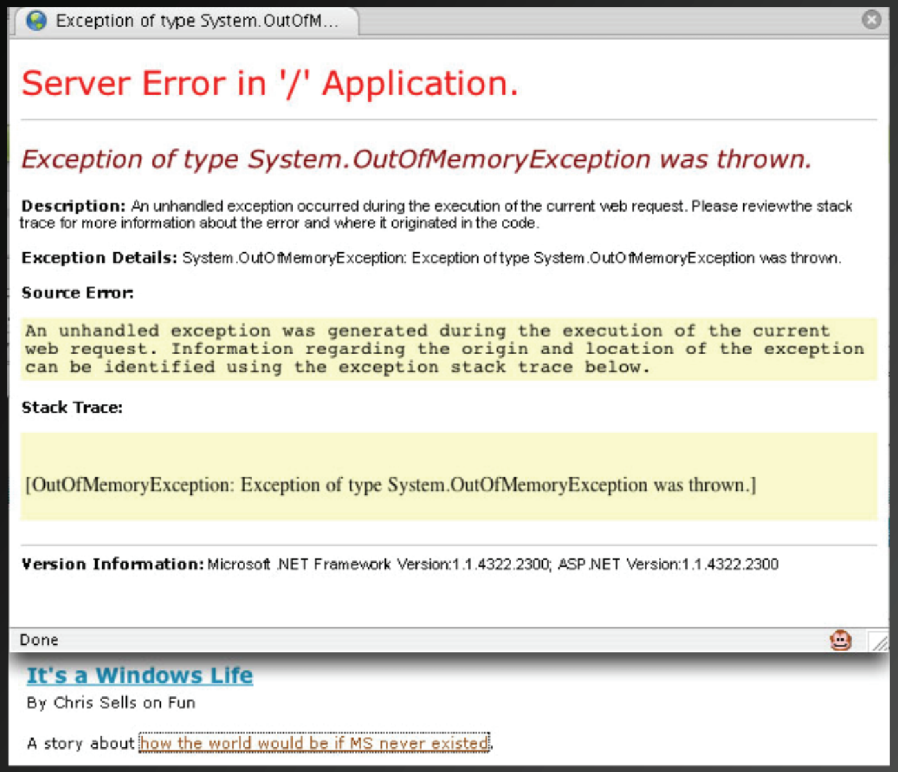
A:



SeamCatch

Exception Handling

A:



Exception Handling

A:

Exception Handling Landscape

Common Language Features

try / catch

method declaration

Exception Handling Landscape

Web Applications

web.xml

Seam Catch

Exception Handling Landscape

Framework Support

JAX-RS, JSF2,
Struts config, etc.

Seam Catch

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 programing model

Declarative

portable behavior

built on top of Java EE

A Better Way: Seam Catch

Catch allows you to build your own handling programing model

Smart, Contextual Handlers

Application State Injection

Specific or broad targeted handlers

A Better Way: Seam Catch

Based on standards: CDI Events

@Observes

Seam Catch

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());
    }
}
```

Seam Catch

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();
    }
}
```

Seam Catch

DEMO