JCON 2022
GERMANY
ONLINE


# Swallowed Exceptions

# CONSULTANT

ELMAR DOTT

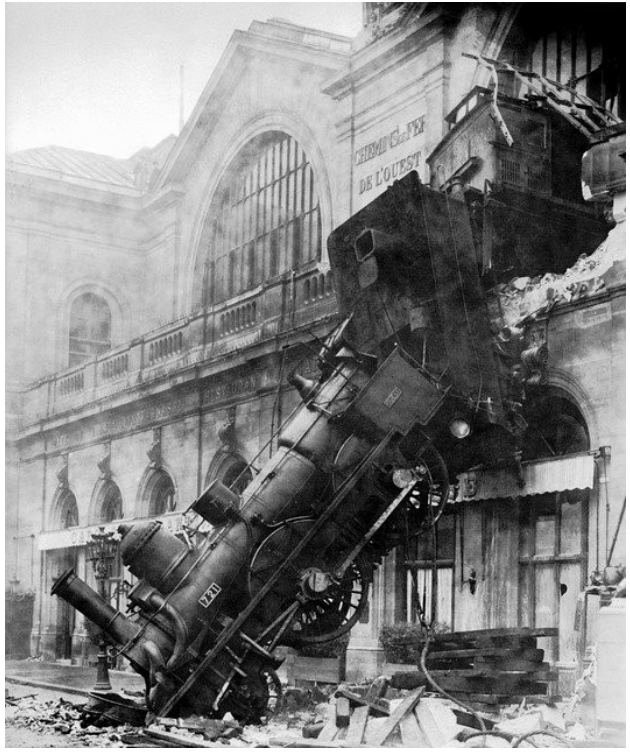
(M. Schulz) studierte an der HS Merseburg Diplominformatik und twittert regelmäßig über alle möglichen technischen Themen. Seine Schwerpunkte sind hauptsächlich Build und Konfiguration Management, Software Architekturen und Release Management.

Seit über fünfzehn Jahren realisiert er in internationalen Projekten für namhafte Unternehmen umfangreiche Webapplikationen. Er ist freier Consultant / Trainer. Sein Wissen teilt er mit anderen Technikbegeisterten auf Konferenzen, wenn er nicht gerade wieder einmal an einem neuen Fachbeitrag schreibt. https://elmar-dott.com

+ Consultant + Writer + Speaker + Trainer +

# AGENDA

- Ein Déjà vu
- Die Struktur von Ausnahmen
- Logging & Testen
- Wartbarkeit
- Informationsfluss
- Microservices



# DEFINITION

Oracle: Eine Ausnahme ist ein Ereignis, das während der Ausführung eines Programms auftritt und den normalen Ablauf der Programmanweisungen unterbricht.

> **Ausnahmen sind Sollbruchstellen im Programmfluss die dabei helfen eine Routine in einem vorausgesagten Fehler - Szenario sicher zu terminieren.**

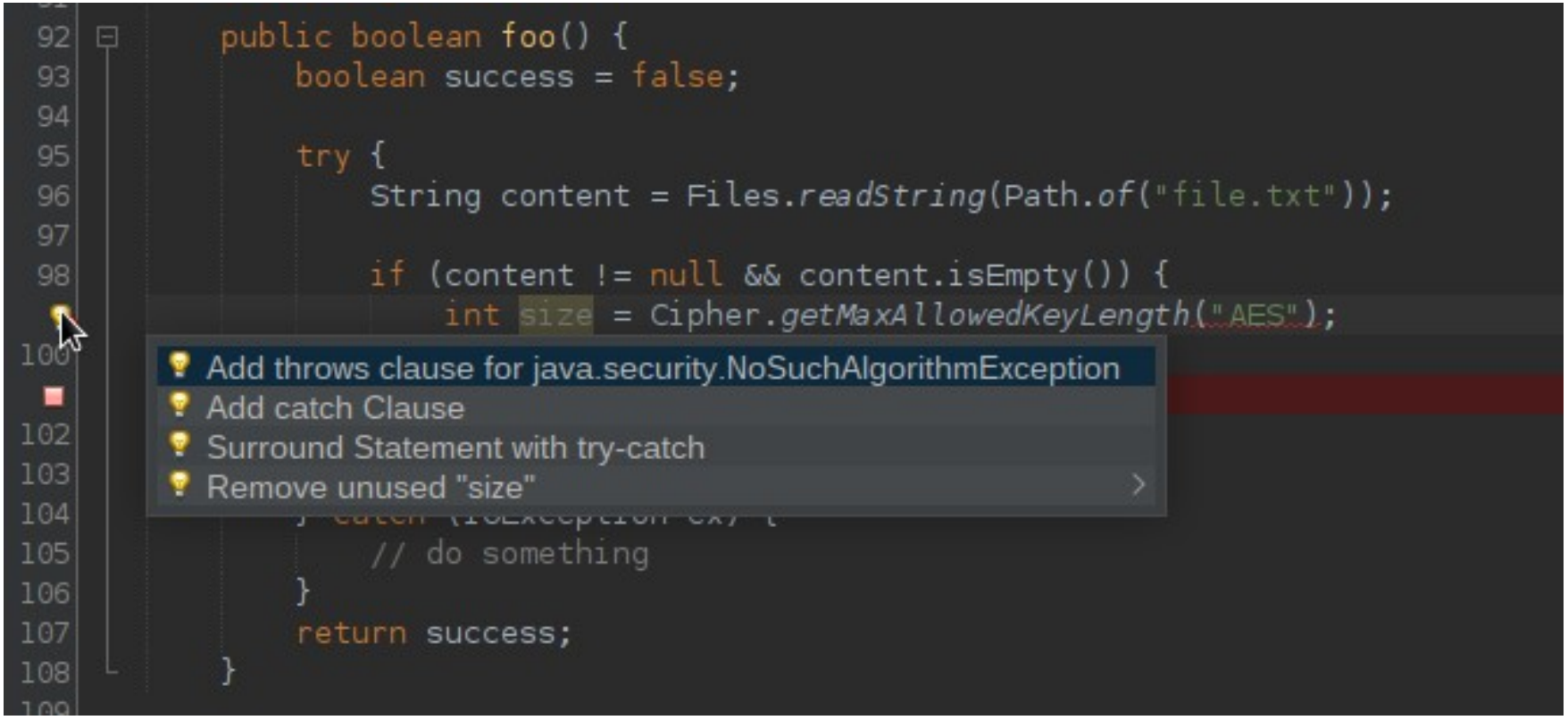
https://docs.oracle.com/javase/tutorial/essential/exceptions/definition.html

```
public void foo() {
    try {
        String content = Files.readString(Path.of("file.txt"));

        try {
            int size = Cipher.getMaxAllowedKeyLength("AES");
        } catch (NoSuchAlgorithmException ex) {
            // do something
        }

    } catch (IOException ex) {
        // do something
    }
}
```

# EDITOR HINTS



```
public void foo() throw UnexpectedException {
    try {
        if(fail) {
            throw new MyOwnException('message');
        }
    } catch (Exception ex) {
        System.err.println(ex.getMessage());
    } finally {
        // clean up
    }
}
```

```
@Test
void check() throws MyException {
    assertThrows(MyException.class, () -> {
        MyObject.foo();
    });
}
```

# Logging & Test Driven Development

## Maven / Junit 5 / Logback

```
 INFO 2022-09-20 16:12:03 org.europa.together.service.MailClientService     | instance class
ERROR 2022-09-20 16:12:03 org.europa.together.application.JdbcActions       | PSQLException: ERROR: duplicate key value violates unique constraint
  Detail: Key (idx)=(88888888-4444-4444-4444-ccccccccccc) already exists.
DEBUG 2022-09-20 16:12:03 org.europa.together.application.JdbcActions       | File (org/europa/together/sql/configuration-test.sql): INSERT INTO AP
DEBUG 2022-09-20 16:12:03 o.e.together.application.ConfigurationHbmDAOTest | TEST CASE: updateNotExist
TRACE 2022-09-20 16:12:03 o.e.together.application.ConfigurationHbmDAOTest | TEST CASE TERMINATED.
DEBUG 2022-09-20 16:12:03 org.europa.together.utils.StringUtils            | generateUUID() da790e2b-5618-470e-bab7-f854ed22da7c
DEBUG 2022-09-20 16:12:03 org.europa.together.application.JdbcActions       | File (org/europa/together/sql/configuration-test.sql): INSERT INTO AP
```

# Wartbarkeit - API Design

- Entkoppeln von Abhängigkeiten
- Pattern: Proxy, Wrapper & Facade
- Auch Ausnahmen von Bibliotheken sollten gekapselt werden

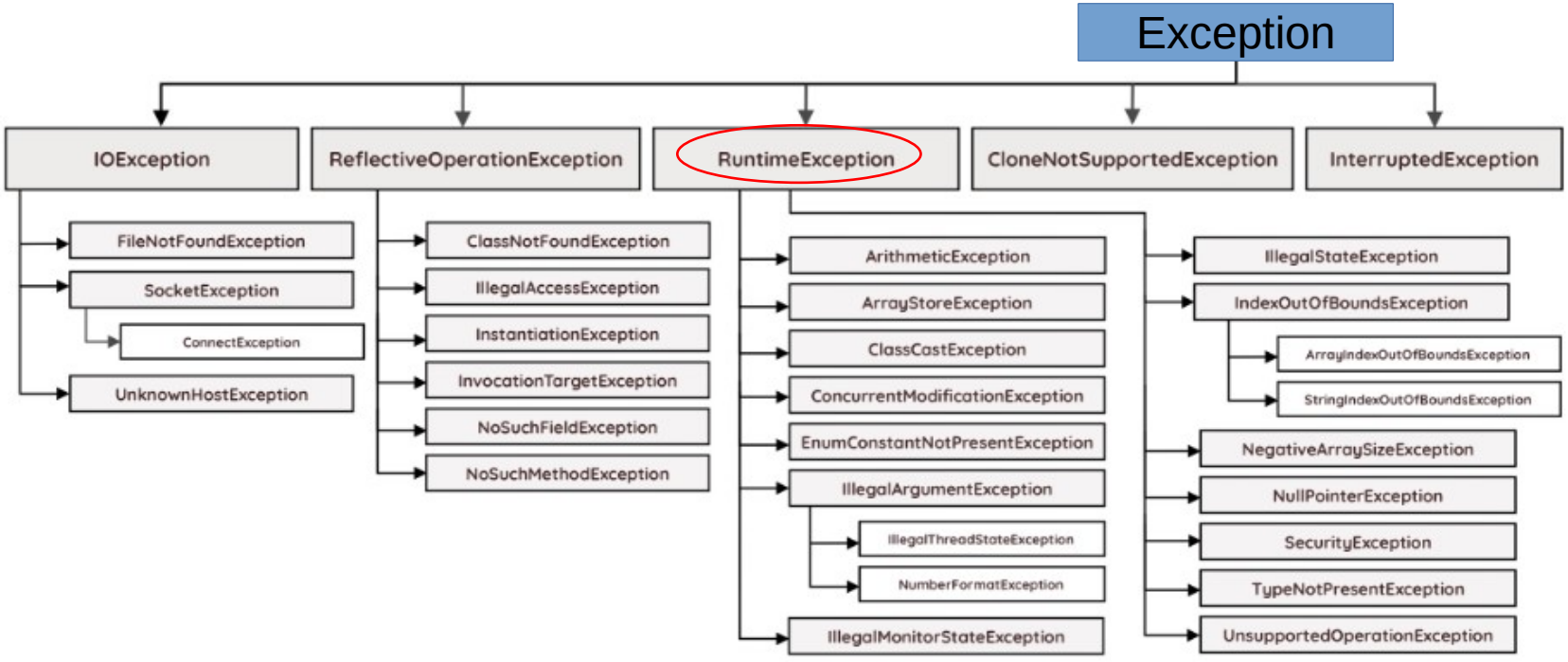
```
public String serializeAsJsonObject(final T object)
                                throws MyJsonProcessingException {
    try {
        ObjectMapper mapper = new ObjectMapper();
        return mapper.writeValueAsString(object);
    } catch (JsonProcessingException ex) {
        throw new
          MyJsonProcessingException(ex.getOriginalMessage());
    }
}
```

# My Own Private - Exceptions

```
public class MyException extends Exception {
    public MyException(final String msg) {
        super(msg);
    }
}
```

```
@Test()
void exceptionObject() {
    MyException ex =
                    new MyException("MyException");
    assertEquals("MyException", ex.getMessage());
}
```

# HIERARCHIE



- sg. ungeprüfte Ausnahmen
- stellen schwere Fehler dar **OutOfMemoryError**
- terminiert das Programm
- können auch ‚gefangen‘ werden

Nicht jede Methode ist geeignet um Exceptions zu fangen bzw. zu werfen.

- Regel 1: Low Level Methoden werfen Exceptions
- Regel 2: High Level Methoden fangen Exceptions
- Regel 3: alles dazwischen leitet ‚nur‘ durch

# BEISPIEL

- Hibernate ORM – EntityManager  // werfen
- DAO Klassen oder ähnliche Konstrukte delegieren die Ausnahme nach ‚oben‘
- es werden dadurch keine Informationen „verschluckt“
- Service Klassen / UI etc. leiten dann die Fehlerbehandlung ein

# ANWENDUNG

```
public void foo() throw UnexpectedException {
    try {
        if(fail) {
            throw new MyOwnException('message');
        }
    } catch (Exception ex) {
        System.err.println(ex.getMessage());
    } finally {
        // clean up
    }
}
```

# NULL POINTER EXCEPTION

```
public List<Integer> foo() {
    List<Integer> result = new ArrayList<>();

    // do something

    return result;
}
```

```
public Response fetchRes(@PathParam("res") String resName) {
    Response response = null;
    try {
        ResourcesDO resource = resourceDAO.find(resName);
        response = Response.status(Response.Status.OK)
                .type(MediaType.APPLICATION_JSON)
                .entity(resourceDAO.asJson(resource))
                .encoding("UTF-8")
                .build();
    } catch (EmptyResultDataAccessException | NoResultException ex) {
        response = Response.status(Response.Status.NOT_FOUND).build();
    } catch (Exception ex) {
        response = Response.status(Response.Status.INTERNAL_SERVER_ERROR).build();
    }
    return response;
}
```

# REFERENZEN


[1] https://www.yegor256.com/2022/08/30/dont-group-exception-catchers.html

[2] Clean Code, 2009, Robert C. Martin, ISBN 0-13-235088-2

[3] Effective Java 3rd Edition, 2018, Joshua Bloch, ISBN-10: 0-13-468599-7

Marco Schulz, 2021, Continuous Integration mit Jenkins, Rheinwerk, ISBN: ISBN 978-3-8362-7834-8 https://www.rheinwerk-verlag.de/continuous-integration-mit-jenkins/


# CREDENTIALS


```
-------------------------------------------------------------
   Homepage      : https://elmar-dott.com
   GitHub        : https://github.com/ElmarDott
   AnchorFM      : https://anchor.fm/elmar-dott
   Twitter       : https://twitter.com/ElmarDott
   Speaker Deck  : https://speakerdeck.com/elmardott

   Lbry          : https://lbry.tv/@elmar.dott:8
   BitChute      : https://www.bitchute.com/channel/3IyCzKdX8IpO/
-------------------------------------------------------------
```

## Danke / thank you / Gracias




open source

ubuntu MATE

Libre Office
The Document Foundation

BITCHUTE

HECHO EN MÉXICO