

INSTITUTO FEDERAL DA BAHIA

CAMPUS FEIRA DE SANTANA

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Alef de Araujo Rocha

Fabio Fernandes dos Santos Novaes

Gabriel Santiago Carneiro

**ATIVIDADE AVALIATIVA EM TRIO  
PESQUISA SOBRE JAVA 7**

Feira de Santana

2025

Alef de Araujo Rocha

Fabio Fernandes dos Santos Novaes

Gabriel Santiago Carneiro

**ATIVIDADE AVALIATIVA EM TRIO  
PESQUISA SOBRE JAVA 7**

Pesquisa sobre Java 7 buscando entender quais inovações essa versão trouxe para a linguagem java e discorrer sobre os mesmos.

Orientador: Prof. Cleber Jorge Lira de Santana

Feira de Santana

2025

1. **Multi-catch**

Antes do lançamento da versão 7 do Java, para o tratamento de exceções, era necessário, caso houvesse a possibilidade de um bloco levar a mais de um tipo de exceção, a criação de um bloco catch para cada exceção, como pode ser visto abaixo:

catch (IOException ex) {

logger.log(ex);

throw ex;

catch (SQLException ex) {

logger.log(ex);

throw ex;

}

}

Contudo, a partir do Java 7, é possível a criação de um único bloco catch com todas as exceções que faz-se necessário tratar, o que é benéfico para o programador (uma vez que elimina a necessidade de duplicar blocos catch); além disso, o "[...] bytecode gerado pela compilação de um bloco catch que manipula múltiplos tipos de exceção será menor (e, portanto, superior) do que a compilação de vários blocos catch que manipulam apenas um tipo de exceção cada." (Oracle, 2020, tradução nossa). O código mostrado anteriormente ficaria com uma barra vertical separando os tipos de exceção, ficando como é mostrado no código abaixo:

catch (IOException|SQLException ex) {

logger.log(ex);

throw ex;

}

1. **Try-with-resources**

Try-with-resources é um recurso sintático que surgiu para diminuir o tratamento manual da abertura e fechamento de recursos, presente até o Java 6. A funcionalidade garante um melhor tratamento das conexões com bancos de dados, redes, arquivos e referências, garantindo que sejam devidamente fechadas. Os recursos no try devem implementar a interface AutoCloseable para que o close() seja automaticamente adicionado ao fim do try.

Exemplo:

class Principal {

public void metodoQueRelancaExcecoes() throws MeuException {

try (Teste t = new Teste()) {

// ...

}

}

public void metodoQueTrataExcecoes() {

try (Teste t = new Teste()) {

// ...

} catch (MeuException e) {

// Trata a exceção.

}

}

}

1. **Precise rethrow**

O precise rethrow no Java 7 permite especificar tipos de exceção mais específicos na cláusula throws de uma declaração de método.

Exemplo:

static class FirstException extends Exception { }

static class SecondException extends Exception { }

public void rethrowException(String exceptionName) throws Exception {

try {

if (exceptionName.equals("First")) {

throw new FirstException();

} else {

throw new SecondException();

}

} catch (Exception e) {

throw e;

}

}

O bloco "try" deste exemplo pode lançar "FirstException" ou "SecondException". Suponha que você queira especificar esses tipos de exceção na cláusula "throws" da "rethrowExceptiondeclaração" do método. Em versões anteriores ao Java SE 7, isso não é possível. Como o parâmetro de exceção da cláusula "catch", "e", é do tipo Exception, e o bloco catch relança o parâmetro de exceção "e", você só pode especificar o tipo de exceção "Exception" na cláusula "throws" da declaração "rethrowException" do método.

Exemplo usando o Precise Rethrow:

public void rethrowException(String exceptionName)

throws FirstException, SecondException {

try {

// ...

}

catch (Exception e) {

throw e;

}

}

Com o uso do "Precise rethrows" o bloco "catch" captura uma exceção, mas ela não faz parte da cláusula "throws". O compilador Java 7 analisa o bloco "try" completo para verificar quais tipos de exceções são lançadas e, em seguida, relê-las a partir do bloco "catch".

Referências bibliográficas:

DIGITALOCEAN. **Java Catch Multiple Exceptions and Rethrow Exception**. Digital Ocean, 2024. Disponível em: https://www.digitalocean.com/community/tutorials/java-catch-multiple-exceptions-rethrow-exception. Acesso em: 12 ago. 2025.

ORACLE. **Catching Multiple Exception Types and Rethrowing Exceptions with Improved Type Checking**. Oracle, 2014. Disponível em: https://docs.oracle.com/javase/7/docs/technotes/guides/language/catch-multiple.html. Acesso em: 12 ago. 2025.

ORACLE. **Try-with-resources Statement**. Oracle, 2014. Disponível em: https://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html. Acesso em: 12 ago. 2025.

STACK OVERFLOW. **Como funciona o try-with-resources**. Stack Overflow em português, 2018. Disponível em: https://pt.stackoverflow.com/questions/172909/como-funciona-o-try-with-resources. Acesso em: 12 ago. 2025.