**Exception**

**Exception** มี 3 ประเภท

1. Exception extends มาจาก Throwable (Checked)

Ex. IOException, ParseException, SQLException

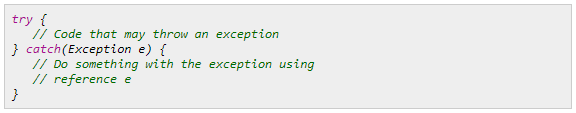
2. RuntimeException extends มาจาก Exception

Ex. IOException, ParseException, SQLException

3. Error extends มาจาก Throwable เป็น Exception ที่ไม่ปกติที่โปรแกรมไม่ควรจัดการ

Ex. AssertionError, IOError, LinkageError, VirtualMachineError

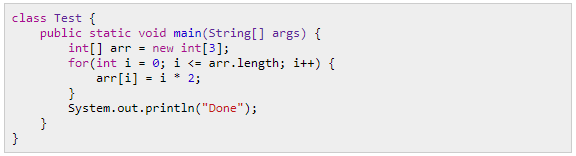
**Try-Catch Block**

Syntax : 

try -> จะ throw เมื่อโปรแกรมมีข้อผิดพลาด

catch -> จัดการกับ exception ซึ่งจะมีการ define ประเภทของ exception และสามารถอ้างอิงถึง exception นั้นได้

ตัวอย่างโปรแกรมที่ไม่ได้ใส่ try catch



เมื่อรันโปรแกรมจะเกิดอะไรขึ้น ???

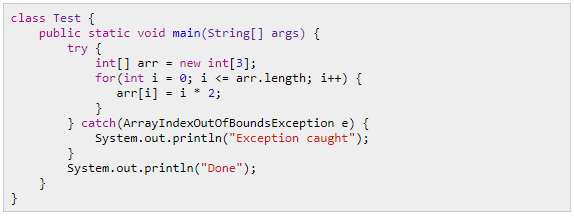
Answer : loop รอบสุดท้าย i จะเป็น 3 แต่ได้กำหนด array ขนาดเท่ากับ 3 ไว้เลยทำให้เกิด exception ตอน runtime

Output : Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 3 at com.example.Test.main(TestException.java:8)

ถ้า exception ไม่ถูก handle JVM จะมีให้มีการ default exception handle ดังนี้

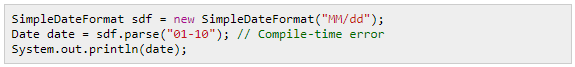
1. ปริ้นท์รายละเอียดของ exception
2. ปริ้นท์ stack trace (ลำดับชั้นของ method ที่ทำให้เกิด exception)
3. Terminate program

ตัวอย่างโปรแกรมที่ใส่ try catch



Output : Exception caught  
 Done

Example



\*\*signature ของ parse method จะ throws ParseException (Extends from Exception)

public Date parse(String source) throws ParseException

“throws” เป็น keyword ที่แสดงว่า exception สามารถ throw, checked exception เท่านั้นที่ require เมื่อประกาศใช้ method parse

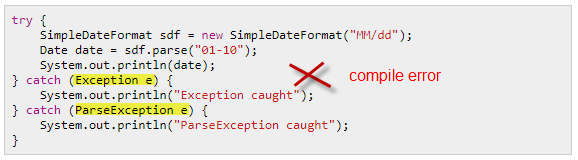


สามารถ catch เป็น superclass ได้

หรือจะ catch แบบนี้ก็ได้



\*\*ถ้ามีหลาย catch ห้ามเอา superclass ไปอยู่บนสุดก่อน subclass

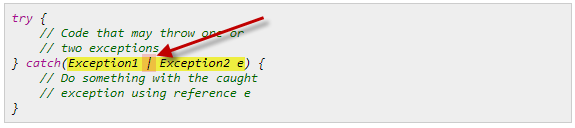


\*\*ใน catch block ถ้ามีการ catch exception ที่ไม่ถูกประเภทจะทำให้ compile error ด้วย



**Multi- Catch**

Syntax :



**Finally**

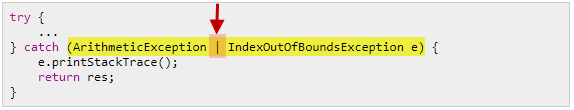


\*\*catch block เป็น optional จะมีทั้ง catch block หรือ finally block ก็ได้

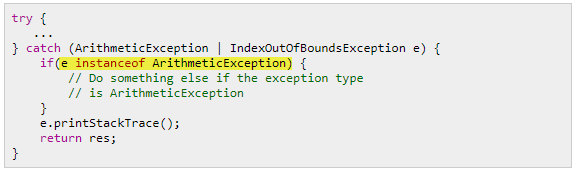
finally block จะถูก excute เสมอไม่ว่าจะเกิด exception หรือไม่เกิด แต่ถ้าสั่ง System.exit(); โปรแกรมจะถูก terminate ทันทีและไม่ทำงานใน finally โดยทั่วไปแล้วเคสนี้จะเกิดขึ้นยากเพราะไม่ค่อยทำกัน

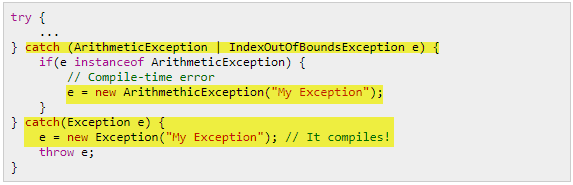
**Mutiple-Catch and Finally**

เป็นการเอาประเภทของ exception ที่อยู่ในแต่ละ catch block มารวมอยู่ใน catch block เดียวกัน

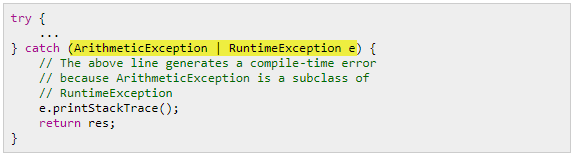


“|” หมายถึง OR จากรูปไม่ว่าจะ throw ArithmeticException หรือ IndexOutOfBoundsException ก็จะเข้ามาทำงานใน catch นี้ แต่ถ้าอยากรู้ว่าเป็น exception ประเภทไหนก็สามารถใช้ instanceof ตรวจสอบได้



\*\*ตัวแปรที่อยู่ใน multi-catch ถือว่าเป็น final ไม่สามารถเอาไป new ได้

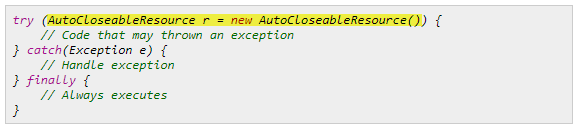
\*\*ไม่สามารถรวม subclass และ superclass ให้อยู่ใน multi-catch block เดียวกันได้



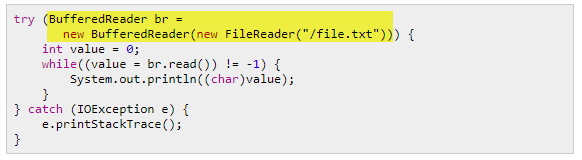
**Try-With-Resources**

* Resource จะถูกปิดโดยอัตโนมัติหลังจากที่ทำงานใน try block

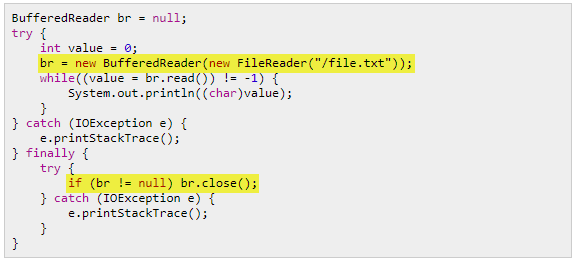
version ก่อนหน้านี้จะใช้ finally block มาใช้ในการ close resources และตั้งแต่ Java 7 ก็มีการใช้ try-with-resources block มาใช้ในการ close resource โดย 1 resources หรือมากกว่าที่ประกาศใน try block จะถูก close resource โดยไม่ต้องใช้ finally



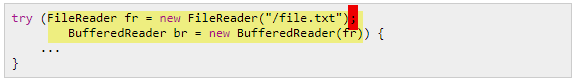
Example



BufferReader จะปิดเมื่อหลังจากทำใน try block เสร็จซึ่งจะเทียบเท่ากับการ close resourceใน finally block ตามตัวอย่างด้านล่าง

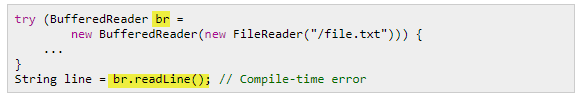


ถ้าต้องการประกาศมากกว่า 1 resource ก็ให้คั่นระหว่าง resource ด้วยเครื่องหมาย ;

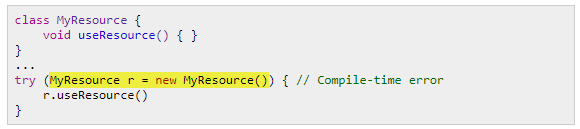


\*\*resource ที่ประกาศใน try-with resources จะไม่สามารถใช้นอก try block ได้เพราะว่า

1. Out of scope
2. Resource ปิดไปแล้วหลังจากหลุดจาก try block



ไม่ใช่ทุก class ที่จะใช้ try-with-resource ได้



ถ้าจะใช้ try-with-resource ต้อง implement

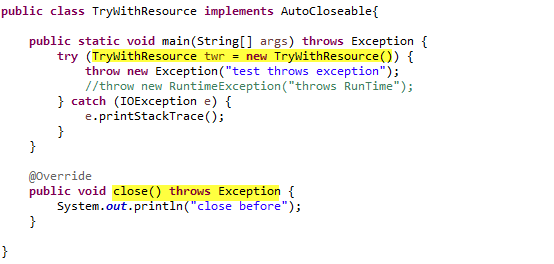
* java.lang.AutoCloseable -> extends Exception

9-16-2017 5-15-02 PM.png

* java.lang.Closeable -> extends IOException

9-16-2017 5-14-12 PM.png

Example

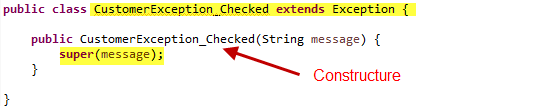


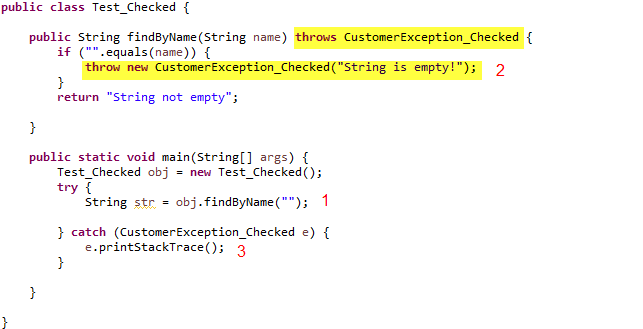
**Custom exceptions**

เป็น class ที่เขียนขึ้นมาเองเพื่อจัดการกับ exception ที่เกิดขึ้นแบ่งออกเป็น Checked, Unchecked

* checked extends จาก Exception หรือ subclass
* unchecked extends จาก RuntimeException หรือ subclass

Example: checked





**Key Points**

* Java มี exception 3 ประเภท

1. java.lang.Exception
2. java.lang.RuntimeException
3. java.lang.Error

* RuntimeException และ subclass ไม่จำเป็นต้อง catch (unchecked)
* Exception และ subclass รู้กันว่าเป็น checked exception เพราะ compiler ต้องตรวจสอบว่ามีคำสั่ง try-catch อยู่หรือไม่
* ถ้า exception ถูก catch ได้มากกว่า 1 block, exception จะ catch แค่ block แรกที่ type ตรง
* ลำดับการ catch ห้าม superclass อยู่ก่อน subclass จะทำให้ complie error
* ถ้า code มีการเรียก method ที่มี throws และไม่ได้อยู่ใน try-catch block ตรงส่วนที่เรียกใช้ต้องประกาศ throws ด้วย
* ต่อเนื่องจากข้อก่อนหน้า การเรียกใช้ method จะต้อง catch หรือ ประกาศ throws
* multi-catch block ยอมให้มีมากกว่า 2 exception ที่ไม่เกี่ยวข้องกันใน 1 catch block
* finally block is ALWAYS executed
* ใน try-with-resources block , resource ที่ประกาศไว้จะ close auto หลังจากทำงานใน try-block โดยการ implement AutoCloseable หรือ Closeable
* เมื่อใช้ try-with-resources block, catch และ finally คือ optinal เราสามารถสร้าง exception เองโดยการ extends Exception หรือ RuntimeException