INSTITUT THOS I CODINA

M03
Programació

CFGS Desenvolupament d'Aplicacions Web

INSTITUT THOS I CODINA

Distribució

UF4: POO. Fonaments.

UF5: POO. Llibreries de classes fonamentals

UF6: POO. Introducció a la persistència en BD

INSTITUT THOS I CODINA

Continguts

Tema 1. Control d'excepcions

1.1. Captura d'excepcions.
1.2. Captura front delegació.
1.3. Llançament d'excepcions.
1.4. Excepcions i herència.

INSTITUT THOS I CODINA

Continguts

2. Aplicació de les estructures d'emmagatzematge en la programació orientada a objectes

2.1. Estructures de dades avançades .
2.2. Creació d'arrays.
2.3. Arrays multidimensionals.
2.4. Cadenes de caràcters.
2.5. Col·leccions i iteradors.
2.6. Classes i mètodes genèrics
2.7. Manipulació de documents XML. Expressions regulars de cerca

INSTITUT THOS I CODINA

Continguts

3. Interfícies gràfiques d'usuari

3.1. Creació i ús d'interfícies gràfiques d'usuari simples.
3.2. Concepte d'esdeveniment. Creació de controladors d'esdeveniments.
3.3. Paquets de classes per al disseny d'interfícies.

INSTITUT THOS I CODINA

Continguts

4. Lectura i escriptura d'informació

4.1. Tipus de fluxos. Fluxos de bytes i de caràcters.
4.2. Classes relatives a fluxos. Utilització de fluxos.
4.3. Entrada/Sortida. Llibreries associades
4.4. Fitxers de dades. Registres.
4.5. Gestió de fitxers
    4.5.1. Modes d'accés.
    4.5.2. Lectura/escriptura
    4.5.3. Utilització dels sistemes de fitxers.
    4.5.4. Creació i eliminació de fitxers i directoris

INSTITUT THOS I CODINA

Resultats d'aprenentatge

1. Gestiona els errors que poden aparèixer en els programes, utilitzant el control d'excepcions facilitat pel llenguatge.

2. Escriu programes que manipulin informació seleccionant i utilitzant els tipus avançats de dades facilitats pel llenguatge

3. Desenvolupa interfícies gràfiques d'usuari simples, utilitzant les llibreries de classes adequades.

4. Realitza operacions bàsiques d'entrada/sortida de informació, sobre consola i fitxers, utilitzant les llibreries de classes adequades.

INSTITUT THOS I CODINA

Criteris d'avaluació

1. Gestiona els errors que poden aparèixer en els programes, utilitzant el control d'excepcions facilitat pel llenguatge.

1.1. Reconeix els mecanismes de control d'excepcions facilitats pel llenguatge.
1.2. Implementa la gestió d'excepcions en la utilització de classes facilitades pel llenguatge.
1.3. Implementa el llançament d'excepcions en les classes que desenvolupa.
1.4. Reconeix la incidència de l'herència en la gestió d'excepcions.

INSTITUT THOS I CODINA

Criteris d'avaluació

2. Escriu programes que manipulin informació seleccionant i utilitzant els tipus avançats de dades facilitats pel llenguatge

2.1. Escriu programes que utilitzin taules (arrays)
2.2. Reconeix les llibreries de classes relacionades amb la representació i manipulació de col·leccions
2.3. Utilitza les classes bàsiques (vectors, llistes, piles, cues, taules de Hash) per emmagatzemar i processar informació.
2.4. Utilitza iteradors per recórrer els elements de les col·leccions.
2.5. Reconeix les característiques i avantatges de cada una de les col·leccions de dades disponibles.
2.6. Crea classes i mètodes genèrics.
2.7. Utilitza expressions regulars en la recerca de patrons en cadenes de text.
2.8. Identifica les classes relacionades amb el tractament de documents XML.
2.9. Dissenya programes que realitzen manipulacions sobre documents XML

INSTITUT THOS I CODINA

Criteris d'avaluació

3. Desenvolupa interfícies gràfiques d'usuari simples, utilitzant les llibreries de classes adequades.

3.1. Utilitza les eines de l'entorn de desenvolupament per crear interfícies gràfiques d'usuari simples.
3.2. Programa controladors d'esdeveniments.
3.3. Escriu programes que utilitzin interfícies gràfiques per a l'entrada i sortida d'informació.

INSTITUT THOS I CODINA

Criteris d'avaluació

4. Realitza operacions bàsiques d'entrada/sortida de informació, sobre consola i fitxers, utilitzant les llibreries de classes adequades.

4.1. Utilitza la consola per realitzar operacions d'entrada i sortida d'informació.
4.2. Aplica formats en la visualització de la informació.
4.3. Reconeix les possibilitats d'entrada / sortida del llenguatge i les llibreries associades.
4.4. Utilitza fitxers per emmagatzemar i recuperar informació.
4.5. Crea programes que utilitzen diversos mètodes d'accés al contingut dels fitxers.

INSTITUT THOS I CODINA

Avaluació

La nota de cada unitat formativa s'obté com a mitja ponderada de les notes de les seves activitats.
Una d'aquestes activitats serà un examen final que englobarà tots els continguts de la unitat i tindrà un pes més gran.
La resta d'activitats consistiran en exercicis o treballs amb un pes en la nota proporcional a la càrrega d'hores o a la seva dificultat.

INSTITUT THOS I CODINA

Avaluació

Segons la programació, el pes de les activitats pel curs 2023-2024 serà:

Qualificació dels resultats d'aprenentatge

Instruments d'avaluació (%)						
AEA		A1	A2	A3	A4	A5
Qualificació dels resultats d'aprenentatge	%					
RA1. Escriu programes que manipulin informació seleccionant i utilitzant els tipus avançats de dades facilitats pel llenguatge	35		75			25
RA2. Gestiona els errors que poden aparèixer en els programes, utilitzant el control d'excepcions facilitat pel llenguatge.	15	75				25
RA3. Desenvolupa interfícies gràfiques d'usuari simples, utilitzant les llibreries de classes adequades.	35			75		25
RA4. Realitza operacions bàsiques d'entrada/sortida de informació, sobre consola i fitxers, utilitzant les llibreries de classes adequades.	15				75	25

La qualificació de la UF5 (Q_{UF5}) s'obté segons la següent ponderació:

$$Q_{UF5} = 0{,}35\ Q_{RA1} + 0{,}15\ Q_{RA2} + 0{,}35\ Q_{RA3} + 0{,}15\ Q_{RA4}$$

tot i que el professor es reserva el dret de modificar entregues i ponderacions en funció del desenvolupament de la unitat formativa.

INSTITUT THOS I CODINA

Avaluació

M03	E01: Arbles (No dual	E02: Conjunt	E03: LlistaCircular	E04: Arbre binari	E05: Cues i piles	E07: SAX	E08: Calculador	E09: Black jack		Examen
	5	10	5	10	10	10	15	15		20
Pes de l'activitat	5	10	5	10	10	10	15	15		20

La prova pràctica, valida la realització de les pràctiques de forma individual i l'assoliment dels RAs, per tant, serà necessari una nota mínima de 4 per fer mitja amb les pràctiques.

INSTITUT THOS I CODINA

Avaluació

No s'acceptaran lliuraments d'activitats fora dels terminis establerts ni que no segueixin les instruccions donades per a aquest lliurament.
La detecció d'un treball copiat comporta un zero per a totes les parts implicades.
La unitat formativa estarà aprovada quan la nota així obtinguda i sense arrodonir sigui igual o superior a 5.
La no realització d'una activitat per falta d'assistència injustificada oficialment suposa un zero d'aquesta activitat i no hi haurà possibilitat de repetir-la. No és obligatòria la realització o aprovació de totes les activitats per fer mitja.

INSTITUT THOS I CODINA

Avaluació

L'assistència a classe és obligatòria i si aquesta és inferior al 70% de les hores impartides l'alumne perd el dret a l'avaluació en la primera convocatòria.
El mòdul no estarà aprovat fins que estiguin aprovades totes les seves unitats formatives.
Llavors, la nota final del mòdul serà la mitjana ponderada de les notes de les unitats formatives.

UF1 25%
UF2 25%
UF3 10%
UF4 15%
UF5 15%
UF6 10%

Recuperació

Si l'alumne no ha estat avaluat o no ha superat la unitat formativa es podrà presentar a la segona convocatòria que consistirà en una prova amb una part escrita i altra a realitzar amb ordinador. En casos particulars, i si el professor ho considera oportú, es pot substituir aquesta prova per la realització d'un treball o una sèrie d'exercicis.

INSTITUT THOS I CODINA

Excepcions

INSTITUT THOS I CODINA

Objectiu

- Permet la detecció i correcció d'errors en temps d'execució.
- Permet al programa recuperar-se i continuar.
- Separar codi normal del tractament d'errors.
- Es propaguen els errors enrere (pila de crides).
- S'agrupen errors segons naturalesa

```
errorCodeType readFile (  ) {
        initialize errorCode = 0;
        open the file;
        if (theFileIsOpen) {
                determine the length of the file;
                if (gotTheFileLength) {
                        allocate that much memory;
                        if (gotEnoughMemory) {
                                read the file into memory;
                                if (readFailed) {
                                        errorCode = -1;
                                }
                        } else {
                                errorCode = -2;
                        }
                } else {
                        errorCode = -3;
                }
                close the file;
                if (theFileDidntClose && errorCode==0) {
                        errorCode = -4;
                } else {
                        errorCode = errorCode and -4;
                }
        } else {
                errorCode = -5;
        }
        return errorCode;
}
```

```
readFile ( ) {
    try {
        open the file;
        determine its size;
        allocate that much memory;
        read the file into memory;
        close the file;
    } catch (fileOpenFailed) {
        doSomething;
    } catch (sizeDeterminationFailed) {
        doSomething;
    } catch (memoryAllocationFailed) {
        doSomething;
    } catch (readFailed) {
        doSomething;
    } catch (fileCloseFailed) {
        doSomething;
    }
}
```

INSTITUT THOS I CODINA

Error vs Excepció

Un **error** es refereix a problemes greus en la màquina virtual de Java, com ara errors en enllaçar amb alguna llibreria. Normalment en els programes Java no es tractaran aquest tipus d'errors.

Una **excepció** és un esdeveniment que ocorre durant l'execució d'un programa que trenca el flux normal d'execució sense ser crític. Quan es parla d'excepcions ens referim a *esdeveniment excepcional*.

INSTITUT THOS I CODINA

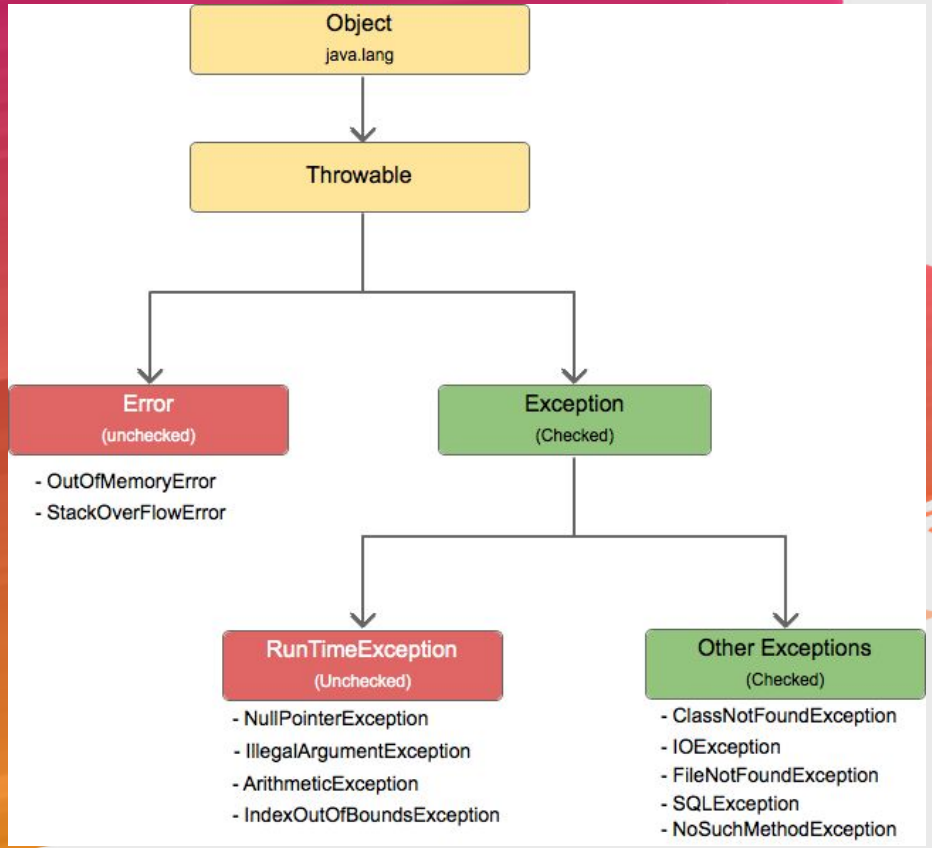
Error

Hi ha una classe, la **java.lang.Error** i les seves subclasses, que serveixen per a definir els errors irrecuperables més seriosos. Aquests errors causen parada en el programa, per la qual cosa el programador no fa falta que els manipuli. Aquests errors els produeix el sistema i són incontrolables per al programador.

java.lang.Error a Oracle

El paquet **java.lang.Exception** i els seus subpaquets contenen tots els tipus d'excepcions.

L'objectiu és que el programa no acabi incontroladament.



INSTITUT THOS I CODINA

Excepció

Quan ocorre un error dins d'un mètode, Java crea un objecte i li ho lliura a l'entorn d'execució.

L'objecte creat és un objecte de la classe Exception o alguna subclasse, i conté informació sobre l'error: causa, tipus, i estat del programa en el moment que va ocórrer l'error.

INSTITUT THOS I CODINA

Excepció

Quan es crea un objecte de tipus Exception el sistema d'execució busca un manegador d'excepcions (Exception handler).

Si no ho troba en el mètode on ocorre l'error, buscarà en el mètode que va cridar a aquest mètode, i així successivament fins a arribar al mètode main.

Si en el mètode main no s'ha controlat l'excepció, la màquina virtual de Java pararà l'execució del programa.

INSTITUT THOS I CODINA

Excepció

Els objectes llençats han de ser instàncies de classes derivades de Throwable.

```
Throwable e = new IllegalArgumentException("Stack underflow");
throw e;

throw new IllegalArgumentException("Stack underflow");
```

Throwable a Oracle

INSTITUT THOS I CODINA

Excepció

La classe Exception és la superclase de tots els tipus d'excepcions. Això permet utilitzar una sèrie de mètodes comuns a totes les classes d'excepcions:

String getMessage().

String toString().

void printStackTrace().

Exception a Oracle

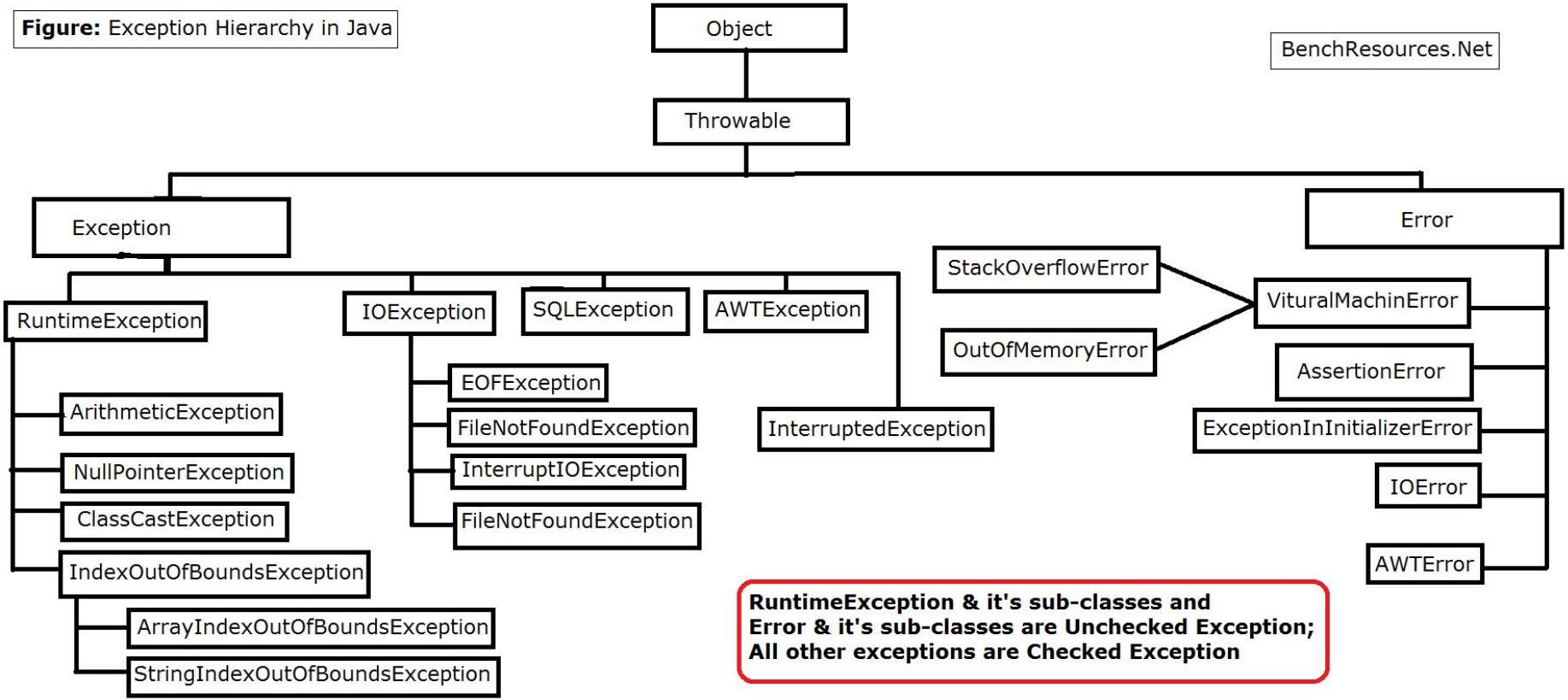
INSTITUT THOS I CODINA

Tipus

checked : gairebé totes les excepcions que hereten d'Exception. Obligatori controlar-les.

unchecked : les excepcions de tipus Error, Runtime Excepcion i les seves subclasses. No és obligatori controlar-les.

Figure: Exception Hierarchy in Java



INSTITUT THOS I CODINA

Unchecked

Les excepcions unchecked (no controlades) es deuen a problemes aliens als mètodes i classes de Java, normalment errors del codi creat pel programador. El compilador no ens obliga a controlar-les.

El compilador ens deixa usar-les sense control d'errors, i serà el programador l'encarregat de gestionar el seu ús.

Unchecked

Exemples d'aquestes excepcions són StringIndexOutOfBoundsException, ArrayIndexOutOfBoundsException, NumberFormatException, NullPointerException, etc.

Les produeixen mètodes de la classe String com charAt() o substring() quan ens sortim de rang. També el mètode parseInt() de la classe Integer, quan no li passem un String parseable.

Checked

Les excepcions checked (controlades) són tipus d'errors dels quals un programa hauria de poder-se recuperar. Ocorren sobretot a l'hora de treballar amb fitxers. El compilador de Java ens obliga a controlar-les.

Si un mètode produeix alguna excepció d'aquest tipus, la crida a aquest mètode donarà un error de compilació (estarà subratllada en vermell) fins que usem algun mecanisme de control d'excepcions.

INSTITUT THOS I CODINA

Checked

Existeixen dues maneres de controlar les excepcions checked dins d'un mètode:

- Amb la sentència try-catch.
- Propagant-les: és a dir, desentenent-se del problema en el mètode actual perquè el resolgui el següent mètode.

INSTITUT THOS I CODINA

Handler

```
try{
} catch(Tipus_excepcio ){
} finally{ //Codi que s'executa en qualsevol cas!
```

El try-catch recorda l'estructura if-else.

INSTITUT THOS I CODINA

Handler

En la part del **try**, s'executa el programa i es llencen les possibles excepcions.

El **catch** recull i processa aquestes excepcions i les de les funcions que hagi pogut cridar i aquestes ho hagin processat.

El codi **finally** és executat al final de tot, no tenint en compte si el codi va acabar normalment, per una excepció un return o un break.

INSTITUT THOS I CODINA

Handler

BLOC1

try {

    BLOC2

} catch (AException e) {

    BLOC3

} finally {

    BLOC 4

}

BLOC 5

Normal

1 → 2 → 4 → 5

Amb AException

1 → 2* → 3 → 4 → 5

Amb altres Exceptions

1 → 2* → 4

```
try {
    FileInputStream infile = new FileInputStream(argv[0]);
    FileOutputStream outfile = new FileOutputStream(argv[1]);
    File tmp_file = File.createTempFile("tmp_", ".tmp");
    String tmp_name = tmp_file.getName();
    System.out.println("Created temporary file " + tmp_name);
} catch (IOException e) {
    e.printStackTrace();
}
```

INSTITUT THOS I CODINA

Propagació

Hi ha la possibilitat és fer que l'excepció la manegi el codi que va fer la crida. Això significa que el mètode en el qual es produeix l'error es desfà de l'excepció i se la passa al mètode que li va cridar a ell. Així el mètode actual es desentén d'ella, i serà aquest altre mètode l'encarregat de controlar-la.

Aquest procés es pot repetir fins a arribar al mètode principal d'un programa (main()), que és l'últim punt on hem de controlar una excepció. No hem de deixar que les excepcions surtin del mètode main.

INSTITUT THOS I CODINA

Propagació

Això es fa afegint la paraula **throws** després de la primera línia d'un mètode. Després d'aquesta paraula s'indica quines excepcions pot provocar el codi del mètode

INSTITUT THOS I CODINA

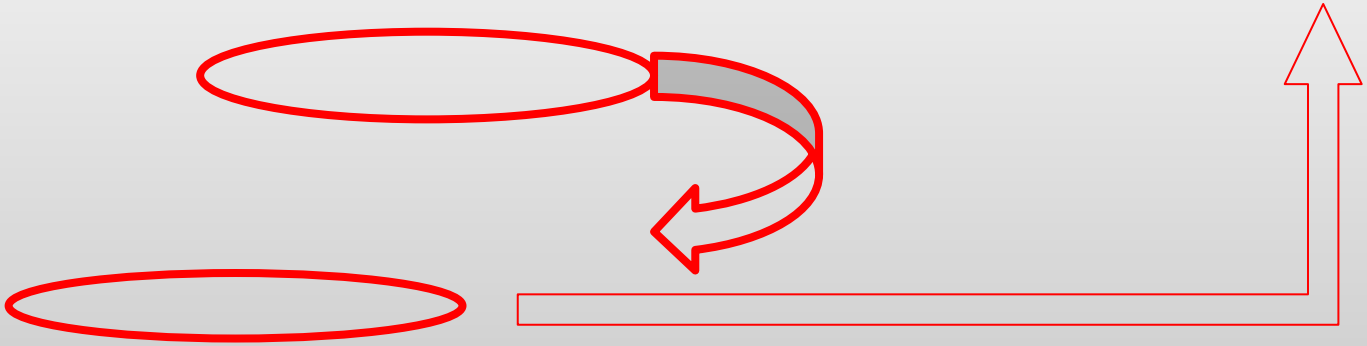
Propagació

Els nostres mètodes també llançar excepcions.

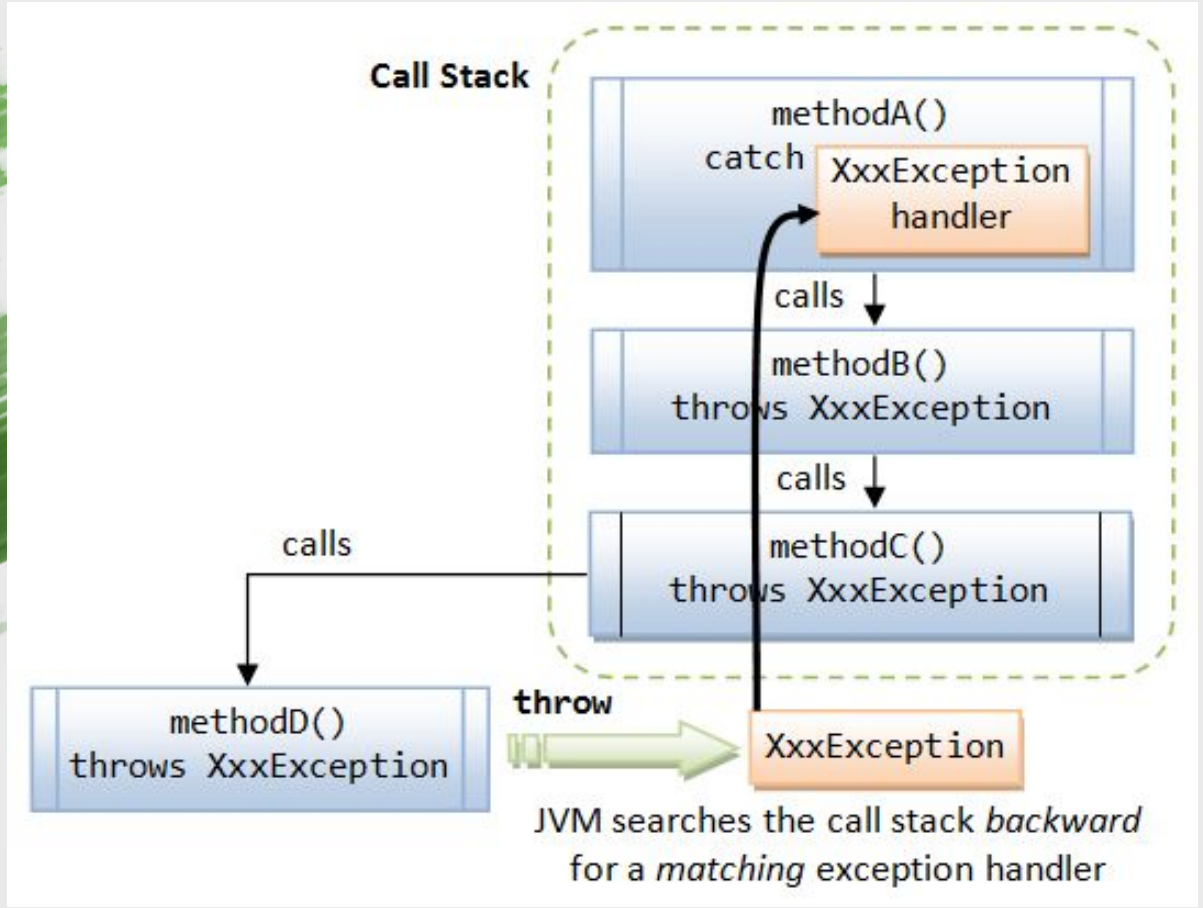
Per a llançar un objecte de tipus Excepció, hem de crear-lo i després llançar-lo amb la comanda **throw**.

```
public static void doIt(InputStream in, OutputStream out) throws IOException {
    int e;
    while ((e = in.read()) != -1) {
        out.write(Character.toLowerCase((char) e));
    }
}
```

```
public static void doio (InputStream in, OutputStream out) throws Throwable {
    try {
        int c;
        while ((c = in.read()) >= 0) {
            out.write(toLowerCase((char) c));
        }
    } catch (Throwable t) {
        throw t;
    }
}
```



Cerca del handler



INSTITUT THOS I CODINA

Problemes

El problema de les excepcions és que en moltes APIs de Java es llancen *checked* quan haurien de ser *unchecked*.

Per exemple, en el API de JDBC en executar una consulta mitjançant *executeQuery()* es llança l'excepció *java.sql.SQLException* que és de tipus *Checked* si la SQL és erronea.

Potser això no és un error de programació i per tant hauria de ser de tipus unchecked? És a dir que tenim un problema ja que haurem de tractar excepcions Checked però no es poden fer cap tractament per a solucionar el que va generar aquesta excepció.

Problemes

Això ens obliga a capturar aquestes excepcions i haver de tractar-les.

Només hi ha una manera de fer-ho, és transformant les excepcions Checked en unchecked.

```
try {
    double c=matematiques.dividir(-1.6, 0);
} catch (Exception ex) {
    throw new RuntimeException("Error divisió",ex)
}
```

INSTITUT THOS I CODINA

Problemes

També hi ha excepcions unchecked , que segons el seu significat, haurien d'haver estat del tipus checked ja que no són degut a errors del programa (el contrari del cas anterior).

L'excepció *ConstraintViolationException* es llança quan alguna validació d'una entitat no s'ha complert. Se sol utilitzar per a indicar que l'usuari ha introduït alguna dada errònia en l'aplicació.

Què NO hem de fer

Mai hem de perdre l'excepció original.

```
try {
    double c=matematiques.dividir(-1.6, 0);
} catch (Exception ex) {
    throw new RuntimeException("Error divisió" +
ex.getMessage());
}
```

és millor

```
try {
    double c=matematiques.dividir(-1.6, 0);
} catch (Exception ex) {
    throw new RuntimeException("Error divisió", ex);
}
```

Exception a Oracle

Què NO hem de fer

No hem de crear excepcions inútils, o crear excepcions genèriques que no aporten res.

Què NO hem de fer

No hem de declarar que llança excepcions checked (throws) quan haurien de ser unchecked.... això només fa que expandir el problema sense solucionar-ho.

Què NO hem de fer

Mai no hem d'ignorar l'excepció.

```
try {
    double c=matematiques.dividir(-1.6, 0);
} catch (Exception ex) {

}
```

Què NO hem de fer

Fer un log.... també és ignorar l'excepció:.

```
try {
    double c=matematiques.dividir(-1.6, 0);
} catch (Exception ex) {
    Logger.getLogger(Main.class.getName()).lo
 g(Level.SEVERE, null, ex);
}
```

Què NO hem de fer

o fer una traça per consola.... també és ignorar l'excepció:.

```
try {
    double c=matematiques.dividir(-1.6, 0);
} catch (Exception ex) {
    ex.printStackTrace();
}
```

Què NO hem de fer

o imprimir el missatge.... també és ignorar l'excepció:.

```
try {
    double c=matematiques.dividir(-1.6, 0);
} catch (Exception ex) {
    System.out.println("Falla al dividir");
}
```