EXEPCIONS (II)

Una Excepció en JAVA és un error en temps d'execució.

Podem **capturar excepcions** (amb l'estructura TRY-CATCH-FINALLY), però també podem **llançar** excepcions i inclús **crear-les**, per posteriorment **capturar-les.**

En Programació Orientada a Objectes, és la Classe la responsable de la lògica dels seus objectes**. És en la classe on s'ha d'assegurar que les dades siguen valides**, és on hem de controlar tot allò que **està permés i el que no**.

Quan és recomanable **llançar** excepcions?

- **En els constructors.** Quan s'intenta instanciar una classe (crear objectes) amb valors incorrectes. Per exemple, crear un objecte Persona amb un DNI no vàlid, o una edat negativa, o al crear un objecte CompteBancari amb saldo negatiu, etc.
- **En els mètodes.** Quan els **arguments** que passem a un mètode **no són correctes o no compleixen certs requisits** o en qualsevol altre mètode que intente fer algo no permés o viole la integritat d'un objecte, com per exemple, quan volem retirar diners de un CompteCorrent sense saldo suficient.
- **En els *"setters"***. Si el **valor** introduït **no és vàlid**.

Llançar una excepció és simplement avisar d'un error, qui crida al mètode serà responsable de gestionar l'excepció per a que el programa no pare (es a dir capturar-la amb Try-Catch)

Exemple:

Declarar una classe anomenada '**PersonaAdulta**' amb els atributs **nom** i **edat**. **Llançar una excepció** de tipus IOException en cas que arribe al constructor una edat menor a 18 anys.

Es a dir, volem validar i llançar una excepció quan creem un objecte de tipus **'PersonaAdulta'**, amb una edat menor que 18 anys.

```
public class PersonaAdulta {
    private String nom;
    private int edat;

    public PersonaAdulta(String nom, int edat) throws Exception {
        this.nom = nom;
        if (edat < 18)
            throw new Exception(nom + " no és adult/a perquè te " + edat + " anys.");
        this.edat = edat;
    }

    public void fijarEdad(int edat) throws Exception {
        if (edat < 18)
            throw new Exception(nom + " no és adult/a perquè te " + edat + " anys.");
        this.edat = edat;
    }

    public void imprimir() {
        System.out.println(nom + " - " + edat);
    }

    public static void main(String[] ar) {
        try {
            PersonaAdulta persona1 = new PersonaAdulta("Jaume", 44);
            persona1.imprimir();
            PersonaAdulta persona2 = new PersonaAdulta("Isabel", 17);
            persona2.imprimir();
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }
}
```

THROW:

Per llançar una excepció hem d'utilitzar la paraula clau **'throw'** i seguidament la referència **d'un objecte de la classe 'Exception'** o d'una subclasse de 'Exception':

THROWS:

En el mètode que volem llançar una excepció amb **'throw'** hem d'enumerar en la declaració després de la paraula clau **'throws'** els noms d'excepcions que pot llançar aquest mètode (poden ser més d'un)

TRY-CATCH:

Quan creem objectes de la classe 'PersonaAdulta' hem de **capturar obligatòriament** l'excepció mitjançant un bloc **try / catch**

```
Jaume - 44
Isabel no és adult/a perquè te 17 anys.
```

THROW - THROWS

- ## Throw

Per llançar l'excepció s'utilitza la paraula reservada **throw** seguit d'un objecte de tipus Exception (o alguna de les seves subclasses com ArithmeticException, NumberFormatException, ArrayIndexOutOfBoundsException, etc.). **Com les excepcions són objectes, han de instanciarse amb "New"**.

Per tant, podem llançar una excepció genèrica així:

```
Exception e = new Exception();
throw e;
```

O de manera abreviada;

```
throw new Exception();
```

El constructor d'**Exception** permet (opcionalment) un argument **String** **per donar detalls sobre el problema**. Si l'excepció no es maneja i el programa es para, el missatge d'error es mostrarà per la consola (això és molt útil per a depurar programes)

```
throw new Exception ( “No pot ser menor de 18 anys” );
```

En lloc de llançar excepcions pròpies de Java (ArithmeticException, NumberFormatException, ArrayIndexOutOfBoundsException, etc.) normalment és preferible llançar excepcions genèriques 'Exception' o millor encara, utilitzar les nostres pròpies excepcions

- ## Throws

És **obligatori** indicar en la **capçalera del mètode** que **pot llançar excepcions i quines**.

Per a això cal afegir, a la dreta de la capçalera i abans de les claus, la paraula reservada **throws** seguit **del tipus d'excepció que pot llançar** (si pot llançar diferents tipus d'excepcions han d'indicar totes separades per comes).

```
public Persona(String dni, int edad) throws InvalidDniException, InvalidEdadException {

        if (!dni.matches(“^[0-9]{8}[A-Z]”)) {
        throw new InvalidDniException("DNI no válido: " + dni);
        }
        if (edad < 0) {
        throw new InvalidEdadException("Edad no válida: " + edad);
        }
        this.dni = dni;
        this.edad = edad;
}
```

En qualsevol cas , és millor llançar excepcions genèriques “EXCEPTION” o utilitzar les nostres pròpies excepcions com en l’exemple anterior.

OBJECTE “EXCEPTION”

Quan creem un objecte de tipus **“exception”** o qualsevol més específic que herete de **“exception”** Este objecte tindrà informació detallada sobre el error.

Esta informació pot ser interesant tant per a que l’usuari sàpiga que ha passat, i per a que el programador puga depurar i corregir el codi.

En la clàusula catch tenim accés a l'objecte en cas que vulguem utilitzar-lo

```
try {
    // instrucions
} catch (Exception ex) {
    //instruccions
}
```

Els dos **mètodes** de la classe **“Exception”** més utilitzats són:

- **getMessage() :** Ens torna un String amb un text simple sobre el error.
- **printStackTrace():** Més complet, Ens mostra el tipus d’excepció, el missatge d’error simple i la pila de cridades als mètodes (És el que fa JAVA per defecte quan troba una excepció i acaba el programa).

```
public static void main(String[] ar) {
    try {
        PersonaAdulta persona1 = new PersonaAdulta("Jaume", 44);
        persona1.imprimir();
        PersonaAdulta persona2 = new PersonaAdulta("Isabel", 17);
        persona2.imprimir();
    } catch (Exception ex) {

        //Mostrem el missatge de l'excepció.

        System.out.println("\nError1:" + ex.getMessage());
        System.err.println("\nError2: "+ ex.getMessage());

        // Mostrem tota la informació de l'excepció
        System.out.println("");
        ex.printStackTrace();

    }
}
```

```
Jaume - 44

Error1:Isabel no és adult/a perquè te 17 anys.

Error2: Isabel no és adult/a perquè te 17 anys.

java.lang.Exception: Isabel no és adult/a perquè te 17 anys.
        at Exemples.PersonaAdulta.<init>(PersonaAdulta.java:10)
        at Exemples.PersonaAdulta.main(PersonaAdulta.java:28)
```

NOTA: Els objectes de tipus “EXCEPTION” tenen sobrecarregat el mètode ***toString(),*** i per tant, també es poden imprimir directament amb ***println():***

```
System.out.println(ex);
```

DEFINIR EXCEPCIONS PRÒPIES

Quan creem les nostres pròpies classes, és habitual que es puguen produir excepcions que no estiguin definides en Java, o simplement les volem personalitzar.

Per a crear una excepció pròpia, definirem una classe derivada de la classe ***"Exception"***, es a dir, **crearem una subclasse que heretarà de la superclasse *"Exception"*.**.

Exemple: Anem a veure un exemple senzill de definició i ús d'un nou tipus d'excepció anomenada **ExcepcioPropia** que utilitzarem quan a li passem al "**mètode**" un valor superior a 10.

```
public class ExcepcioPropia extends Exception{

    private int numero;

    // Creem el constructor de la nostra excepció

    ExcepcioPropia(int numero){
        this.numero=numero;
    }

    // Sobrecarreguem el métode toString de la classe Exception

    public String toString() {
        return "Excepció Pròpia[" + this.numero + "]";
    }
}
```

```
public class MainExcepcionsPropies {

    public static void main(String[] args) {

        try
        {
            metode(1);
            System.out.println("\n ******** ");
            metode(20);
        }
        catch (ExcepcioPropia expropia)
        {

            System.out.println("Capturant l'excepció " + expropia);
        }
    }

    static void metode (int n) throws ExcepcioPropia{

        System.out.println(" Estic cridant al mètode amb el valor: " + n + " metode("+n+")");

        if (n>10)
            throw new ExcepcioPropia(n);

        System.out.println("Finalització Normal!!! ");
    }

}
```

```
 Estic cridant al mètode amb el valor: 1 metode(1)
Finalització Normal!!!

********
 Estic cridant al mètode amb el valor: 20 metode(20)
Capturant l'excepció Excepció Pròpia[20]
```

Com es pot observar l'excepció es llança quan el número és més gran que 10 i **serà tractada per la nova classe creada (ExcepcioPropia) que hereta de Exception.**

A més es sobreescriu el mètode toString que és l'encarregat de mostrar el missatge associat a l'excepció.