Sesión 03

EXCEPCIONES PERSONALIZADAS

Curso Virtual

Fundamentos de Programación

Ing. Aristedes Novoa

anovoa@galaxy.edu.pe

GALAXY TRAINING

ÍNDICE

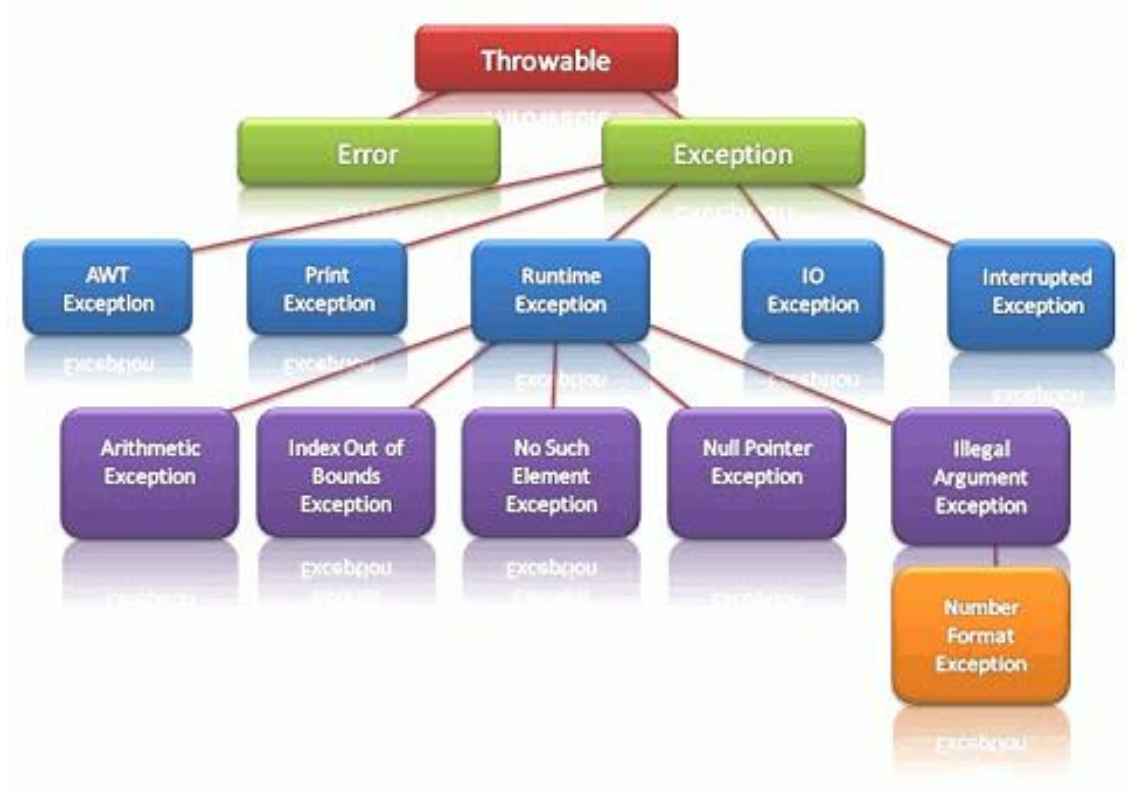
Las excepciones y errores son el medio que ofrece Java para gestionar los problemas que pueden suceder cuando ejecutamos un programa.

Jerarquía de Clases

Clase Throwable
Es la superclase de todos los Errores y excepciones.

Clase Error
Indica problemas graves que un programa normalmente no debería intentar detectar.

Clase Exception
Indica las excepciones normales que un programa podría querer detectar. Pueden ser verificadas y no verificadas.



Jerarquía de Excepciones y Errores

Se implementa usando las clausulas throw y throws a nivel de métodos que generan la excepción o invocan métodos que implementan excepciones

throws.- Permite definir el tipo de error o excepción que lanzara un determinado método

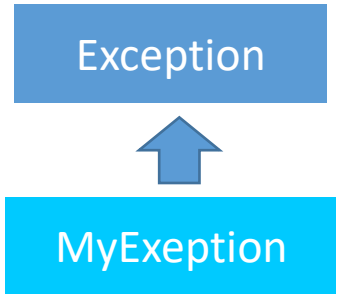
throw.- Permite lanzar el error o excepción en un método

Throws y Throw

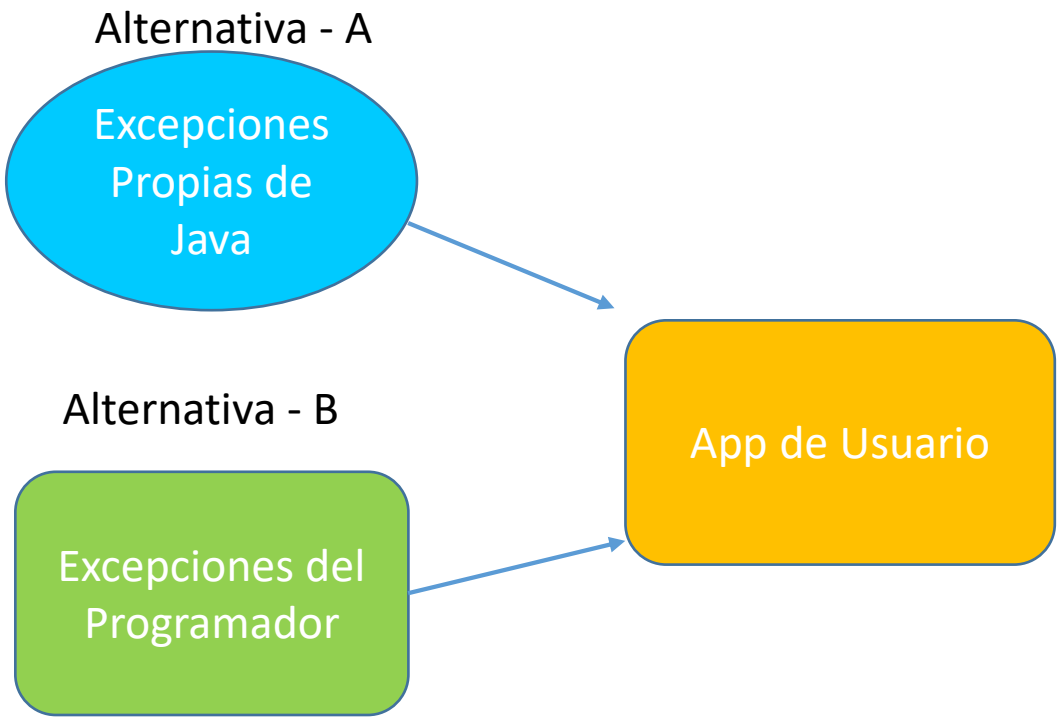
```
try {
            // código
} catch (Exception1 e ){
             // código que realiza el tratamiento
            // de la excepción o error reportado
} catch (Exception2 e ){

}finally{
            // código  que se ejecuta si o si
}
```

Estructura Try.. Catch

Java permite crear excepciones personalizadas en base a la jerarquía ya existente con la finalidad de realizar tratamientos especializados de acuerdo a las necesidades del programador.



Excepciones personalizadas



Excepciones personalizadas- Alternativas

```java
public class ProductoException extends Exception {

    public ProductoException() {

    }

    public ProductoException(String message) {
        super(message);

    }

    public ProductoException(Throwable cause) {
        super(cause);

    }

    public ProductoException(String message, Throwable cause) {
        super(message, cause);

    }

    public ProductoException(String message, Throwable cause, boolean enableSuppression, boolean writableStackTrace) {
        super(message, cause, enableSuppression// TODO Auto-generated constructor stub
    }

}
```

```java
@Data
@NoArgsConstructor
@ToString
public class Producto {

    private Long codigo;
    private String nombre;
    private Double precio;
    private Integer stock;
    private boolean activo;


    public void setPrecio(Double precio) throws ProductoException{
        if (precio<0) {
            throw new ProductoException("El precio no puede ser negativo");
        }
        this.precio= precio;
    }

    public Producto(Long codigo, String nombre, Double precio, Integer stock, boolean activo) throws ProductoException{
        super();
        this.codigo = codigo;
        this.nombre = nombre;

        this.setPrecio(precio);

        this.stock = stock;
        this.activo = activo;
    }

}
```

```java
public class AppProducto {

    public static void main(String[] args) {

        try {
            Producto producto= new Producto();

            producto.setCodigo(100L);
            producto.setNombre("Proyector ViewSonic");
            producto.setPrecio(-1500.00);
            producto.setStock(8);
            System.out.println(producto);

            Producto producto2= new Producto(200L, "Aire Acondicionado LENOX", 2300.75,4,true);

            System.out.println(producto2);

            System.out.println(new Producto(300L, "ASUS VivoBook Pro", -4200.00,4,true));

        } catch (ProductoException e) {
            System.err.println(e.getMessage());
        }

    }

}
```

GALAXY

TRAINING