Argentina
programa
4.0

Excepciones y Colecciones en Java

"Desarrollador Java Inicial"

Tipos de errores

Argentina programa 4.0

¿ Qué tipos de errores pueden ocurrir durante la ejecución de un programa / aplicación?

- Situación inesperada
- Error de negocio / Flujos alternativos

En Java ambos se implementan a través de excepciones o Exceptions

Situaciones que tengo en cuenta durante el desarrollo:

- Login incorrecto
- Datos en formato incorrecto
- Modificar los datos de una persona dada de baja
- El archivo necesario no existe
- El mensaje no puede llegar a destino
- etc...

Códigos de error

Argentina programa 4.0

Primero vamos a ver las siguientes implementaciones

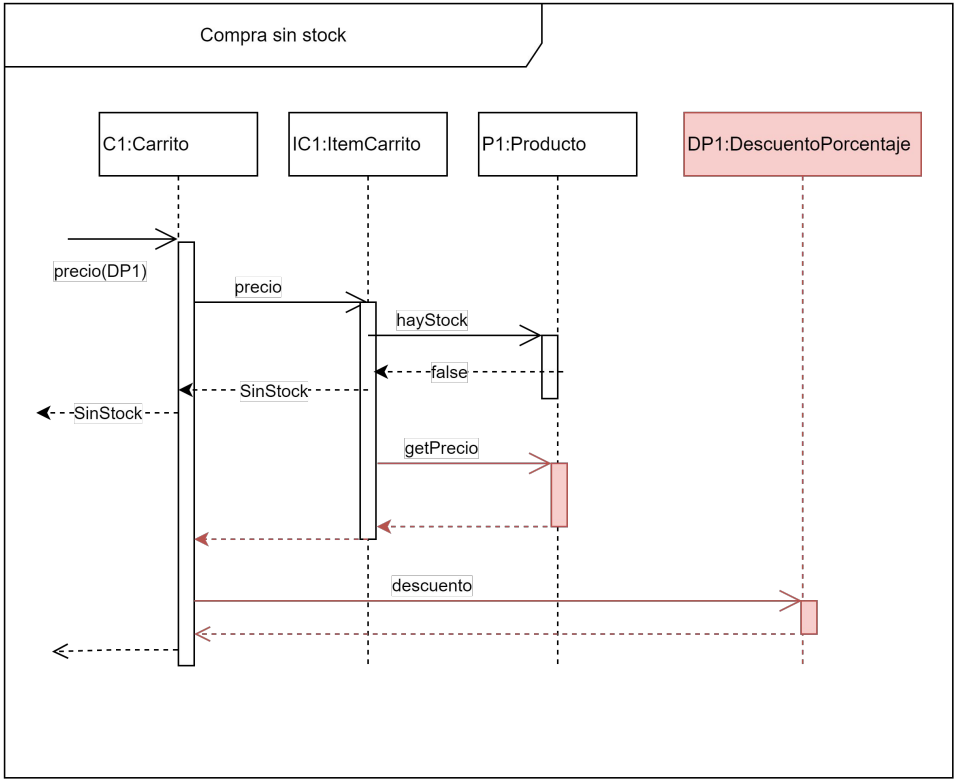
- Login#ingresar(usuario,password): int -> 1 Login ok 2 login error
- RepoUsuario#modificar(usuario):String -> ok, usuario no existe, nombre en formato incorrecto, apellido vacío , ... etc

Este es un mecanismo indicado para cuestiones sencillas (por ejemplo en la mayoría de los SO un proceso que termina correctamente retorna cero y si hay algún error cualquier otro número), pero cuando las situaciones de error son más complejas son poco adecuadas

Excepciones

Mecanismo utilizado por diversos lenguajes de programación para generar y manejar los errores durante la ejecución de un proceso.

En este ejemplo, lo rojo nunca se ejecuta y el programa se corta ante la excepción de SinStockException



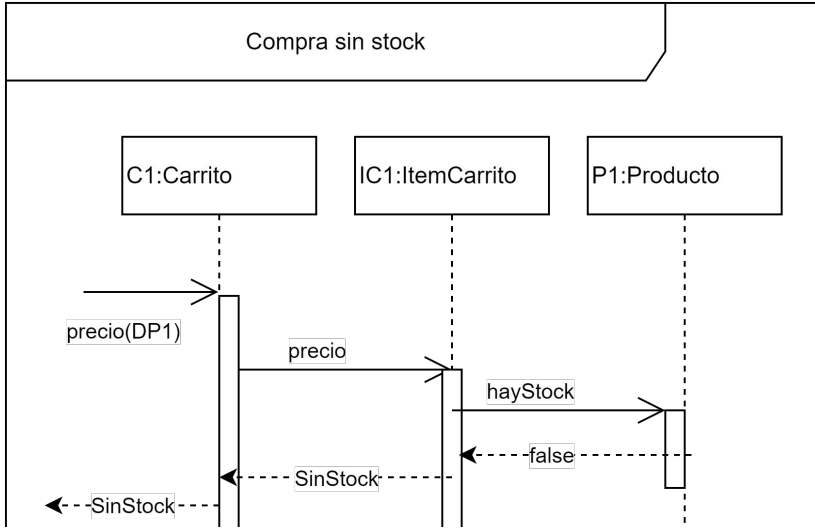
Implementación

1. Crear una clase que herede de Exception o RuntimeException

```
public class NoHayStockException extends Exception {
	private final Producto producto;
	public NoHayStockException(Producto producto) {
		this.producto = producto;
	}}
```

1. Arrojar la Excepción

```
public class ItemCarrito {
	public float precio() throws NoHayStockException {
		if (!producto.hayStock()) {
			throw new NoHayStockException(producto);
		}
		return this.cantidad * this.producto.getPrecio();
	}}
```



Implementación

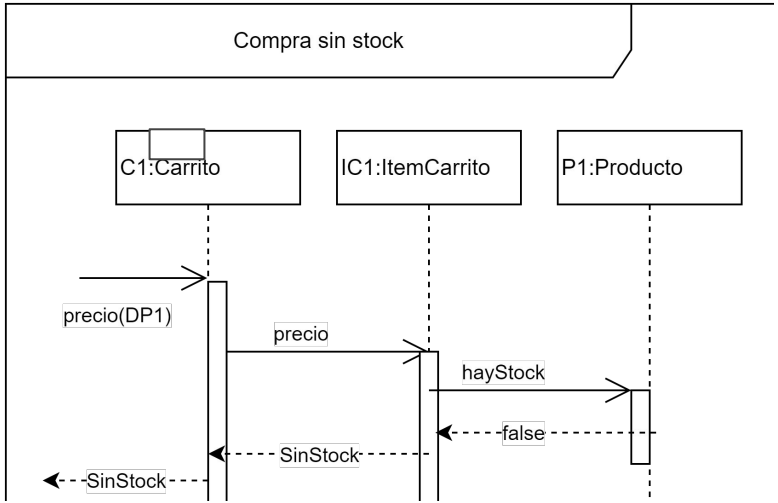
Argentina programa 4.0

3. “Subir” la Exception

```
public class CarritoCompra {
            public float precio() throws
      NoHayStockException {
                  return item1.precio() + item2.precio()
            + item3.precio();
            }
}
```

4. Tratar la excepción

```
public static void main(String[] args) {
      CarritoCompra carrito = new CarritoCompra();
      // Se arma el carrito …
      try {
            Float precio = carrito.precio();
            System.out.println("el precio del carrito es: "+ precio.toString());
      } catch (NoHayStockException e) {
            System.out.println("No hay stock de al menos uno de los productos");
      }
}
```



Excepciones Comunes

Argentina programa 4.0

Java y algunas librerías base, ya establecieron excepciones comunes y algunas de ellas deberían ser utilizadas o heredadas antes de crear otras.

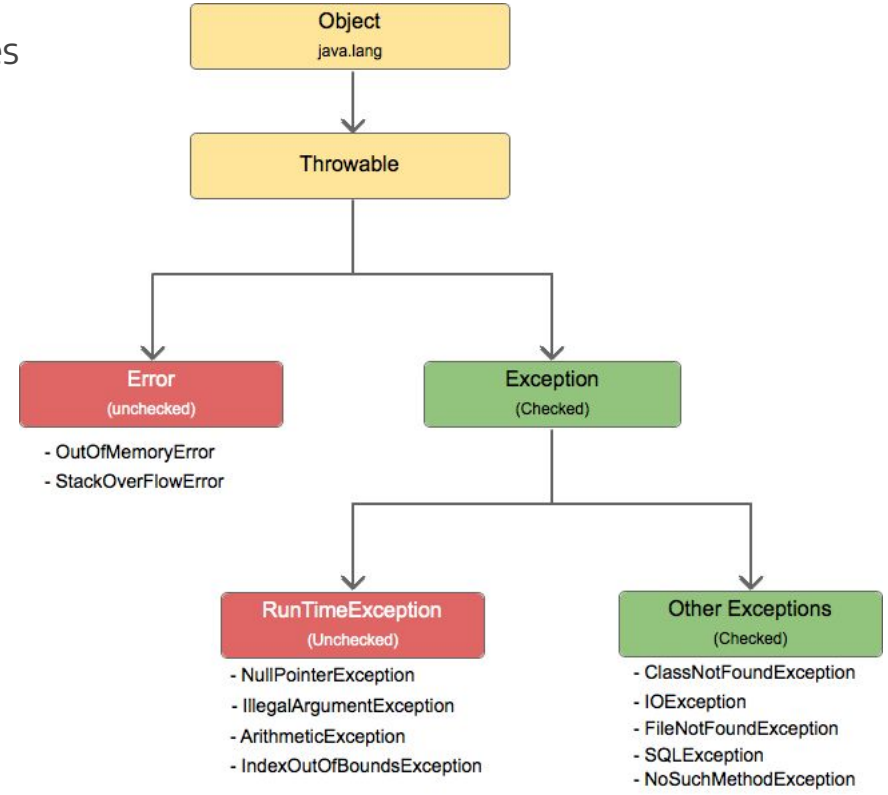
No chequeadas

- NullPointerException
- DivisionByCeroException
- IllegalStateException

Chequeadas

- FileNotFoundException
- IOException

....



Para tener en cuenta

Argentina programa 4.0

- No siempre es fácil determinar cuándo crear una excepción
- Depende un poco de mi marco de referencia, pero en general hay que pensar que errores un “usuario” puede llegar a cometer (olvidarse de la UI)
- ¿Qué tan específico ser con las excepciones?
    - Mientras más preciso se es, más fácil es saber QUÉ SALIÓ MAL
    - También tener en cuenta que las excepciones pueden tener datos asociados
- ¿Cuántas Crear?
    - Tampoco sirve tener muchos tipos de error...
    - No se puede tener en cuenta todo