

La Programmation Orientée Objet (POO) avec Java

Module 7 – Les exceptions

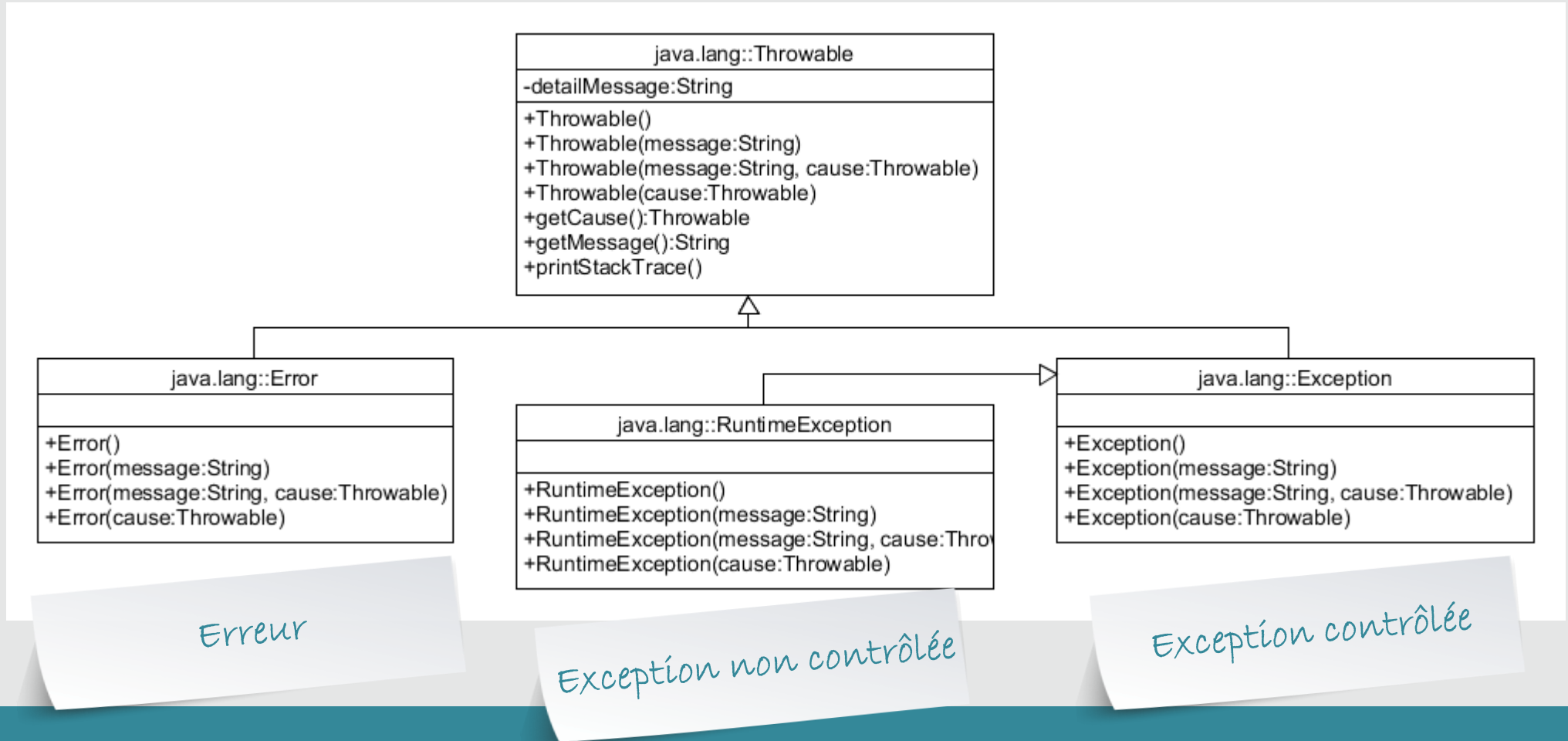


Objectifs

- Comprendre plus en profondeur le mécanisme des exceptions
- Savoir créer des exceptions personnalisées

Les exceptions

Les exceptions et l'héritage



Les erreurs

- Erreurs graves et irrécupérables de la machine virtuelle ou de java
- Arrêt du programme
- Exemples :
 - Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
at fr.eni.ecole.monopoly.Monopoly.main([Monopoly.java:109](#))
 - Exception in thread "main" java.lang.StackOverflowError
at fr.eni.ecole.monopoly.Monopoly.main([Monopoly.java:109](#))
- Correction du code ou modification du paramétrage de la machine virtuelle java

Les exceptions non contrôlées

- Erreurs de programmation détectées à l'exécution
- Exemples :
 - Exception in thread "main" [java.lang.NullPointerException](#)
at fr.eni.ecole.monopoly.Monopoly.main([Monopoly.java:109](#))
 - Exception in thread "main" [java.lang.ArrayIndexOutOfBoundsException: 5](#)
at fr.eni.ecole.monopoly.Monopoly.main([Monopoly.java:109](#))
 - Exception in thread "main" [java.lang.ArithmeticException: / by zero](#)
at fr.eni.ecole.monopoly.Monopoly.main([Monopoly.java:109](#))
- À corriger par le programmeur

Les exceptions

Les exceptions non contrôlées personnalisées

```
public class JeuException extends RuntimeException {  
  
    private static final long serialVersionUID = 1L;  
  
    public JeuException(String message) {  
        super(message);  
    }  
}
```

Exception non contrôlée :
Hérite de **RuntimeException**

Les exceptions non contrôlées personnalisées

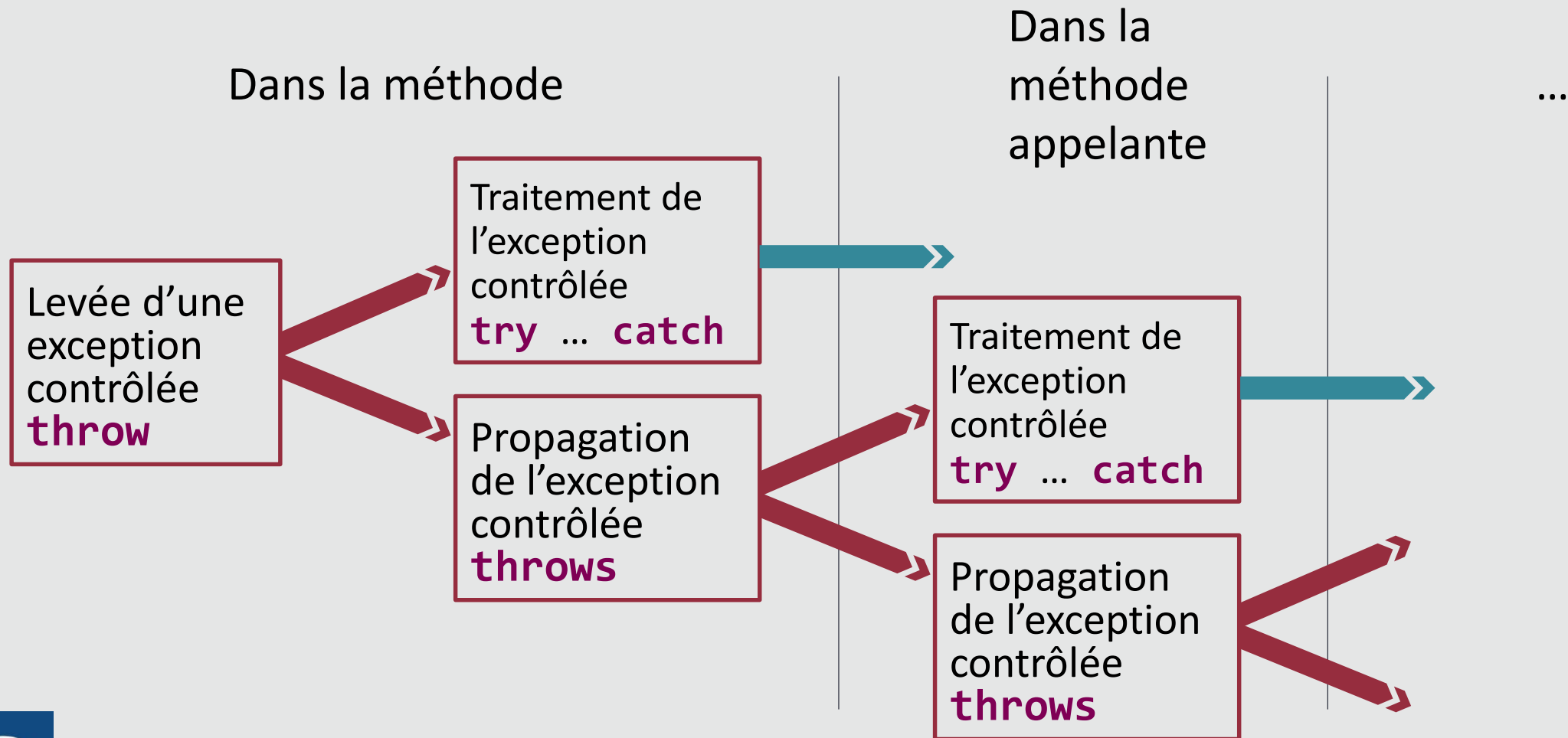
```
public class De {  
    private int nbFaces;  
    ...  
    public void setNbFaces(int nbFaces) {  
        De.verifNbFaces(nbFaces);  
        this.nbFaces = nbFaces;  
    }  
  
    private static void verifNbFaces(int nbFaces) {  
        if (nbFaces <= 1)  
            throw new JeuException("Un dé doit avoir au moins deux faces");  
    }  
    ...  
}
```

Ni bloc **try** ... **catch**,
ni déclaration de
propagation d'exception
avec **throws**

Les exceptions contrôlées

- Erreurs d'utilisation
- Exemple :
 - Exception in thread "main" [java.io.FileNotFoundException: inexistant.txt \(Le fichier spécifié est introuvable\)](#)
 - at java.io.FileInputStream.open0([Native Method](#))
 - at java.io.FileInputStream.open(Unknown Source)
 - at java.io.FileInputStream.<init>(Unknown Source)
 - at java.io.FileInputStream.<init>(Unknown Source)
 - at java.io.FileReader.<init>(Unknown Source)
 - at fr.eni.ecole.monopoly.Monopoly.main([Monopoly.java:109](#))

Les exceptions contrôlées



Les exceptions contrôlées personnalisées

- Classe héritant de Exception mais pas de RuntimeException
- Exemple :

```
public class FailliteException extends Exception {  
  
    private static final long serialVersionUID = 1L;  
    private Joueur joueur;  
  
    public FailliteException(String message, Joueur j) {  
        super(message);  
        this.joueur = j;  
    }  
  
    public Joueur getJoueur() {  
        return joueur;  
    }  
}
```

Exception contrôlée :
Hérite de **Exception**

Les exceptions contrôlées personnalisées

```
private static void jouer() throws AllerEnPrisonException {  
    ...  
    try {  
        if (Monopoly.de1.lancer() == Monopoly.de2.lancer()) {  
            nbDoubles++;  
            if (nbDoubles == 3) {  
                System.out.printf("%s a fait %d et %d aux dés%n", joueurCourant.get(),  
                                   Monopoly.de1.getFaceTiree(), Monopoly.de2.getFaceTiree());  
                throw new AllerEnPrisonException("3 doubles d'affilée");  
            }  
        }  
        joueurCourant.get().jouer(Monopoly.de1.getFaceTiree(), Monopoly.de2.getFaceTiree());  
    } catch (FailliteException e) {  
        Monopoly.joueurs.retirer(e.getJoueur());  
        System.err.println(e.getMessage());  
    }  
    ...  
}
```

➡ Monopoly.jouer() line: 132
Monopoly.main(String[]) line: 107



Les exceptions contrôlées personnalisées

```
public class Joueur {
    ...
    public void jouer(int de1, int de2) throws FailliteException, AllerEnPrisonException {
        //////////////////////////////////// déplacement ////////////////////////////////////
        System.out.printf("%s a fait %d et %d aux dés%n", this.nom, de1, de2);
        if (position.get().joueurPart(this)) {
            for (int i = 0; i < de1 + de2 - 1; i++) {
                position = position.suivant();
                position.get().joueurPasse(this);
            }
            position = position.suivant();
            position.get().joueurArrive(this);
        }
        ...
    }
    ...
}
```





➡

- Joueur.jouer(int, int) line: 98
- Monopoly.jouer() line: 132
- Monopoly.main(String[]) line: 107

Les exceptions contrôlées personnalisées

```
public abstract class Propriete extends Case implements Detenable {
    ...
    @Override
    public void joueurArrive(Joueur j) throws FailliteException, AllerEnPrisonException {
        super.joueurArrive(j);
        if (this.proprio == null) {
            if (Outils.ouiNon(String.format("Voulez-vous acheter %s pour %d€ ?", this.nom,
                                             this.prixAchat))) {

                j.debiter(this.prixAchat);
                this.setProprio(j);
            }
        } else {
            if (j.equals(this.proprio))
                System.out.printf("%s est à domicile%n", j);
            else
                this.payerLoyer(j, this.proprio);
        }
    }
}
```



➡

- Gare(Propriete).joueurArrive(Joueur) line: 56
- Joueur.jouer(int, int) line: 98
- Monopoly.jouer() line: 132
- Monopoly.main(String[]) line: 107

Les exceptions contrôlées personnalisées

```
public class Gare extends Propriete {
```

```
    public Gare(String nom) {  
        super("Gare " + nom, 200, Groupe.GARE);  
    }
```

```
@Override
```

```
    protected void payerLoyer(Joueur passager, Joueur proprietaire) throws FailliteException {  
        int loyer = this.txComplGroupe;  
        if(loyer==75)  
            loyer = 100;  
        else if(loyer==100)  
            loyer = 200;  
        System.out.printf("%s possède %d gare%s\n", p, this.txComplGroupe*4/100, loyer>25?"s":"");  
        passager.payeA(proprietaire, loyer);  
    }  
}
```



```
Gare.payerLoyer(Joueur, Joueur) line: 38  
Gare(Propriete).joueurArrive(Joueur) line: 56  
Joueur.jouer(int, int) line: 98  
Monopoly.jouer() line: 132  
Monopoly.main(String[]) line: 107
```



Les exceptions contrôlées personnalisées

```
public class Joueur {  
    ...  
    public void debiter(int somme) throws FailliteException {  
        this.argent -= somme;  
        if (this.argent < 0) {  
            throw new FailliteException(this.nom + " n'a plus d'argent, il quitte la partie !", this);  
        }  
    }  
    ...  
    public void payeA(Joueur joueur, int somme) throws FailliteException {  
        System.out.printf("%s paye %d€ à %s%n", this.nom, somme, joueur);  
        this.debiter(somme);  
        joueur.crediter(somme);  
    }  
    ...  
}
```



- Joueur.payeA(Joueur, int) line: 286
- Gare.payerLoyer(Joueur, Joueur) line: 38
- Gare(Propriete).joueurArrive(Joueur) line: 56
- Joueur.jouer(int, int) line: 98
- Monopoly.jouer() line: 132
- Monopoly.main(String[]) line: 107

Les exceptions contrôlées personnalisées

```
public class Gare extends Propriete {
```

```
    public Gare(String nom) {  
        super("Gare " + nom, 200, Groupe.GARE);  
    }
```

```
@Override
```

```
    protected void payerLoyer(Joueur passager, Joueur proprietaire) throws FailliteException {  
        int loyer = this.txComplGroupe;  
        if(loyer==75)  
            loyer = 100;  
        else if(loyer==100)  
            loyer = 200;  
        System.out.printf("%s possède %d gare%s\n", p, this.txComplGroupe*4/100, loyer>25?"s":"");  
        passager.payeA(proprietaire, loyer);  
    }  
}
```






```
Gare.payerLoyer(Joueur, Joueur) line: 38  
Gare(Propriete).joueurArrive(Joueur) line: 56  
Joueur.jouer(int, int) line: 98  
Monopoly.jouer() line: 132  
Monopoly.main(String[]) line: 107
```



Les exceptions contrôlées personnalisées

```
public abstract class Propriete extends Case implements Detenable {  
    ...  
    @Override  
    public void joueurArrive(Joueur j) throws FailliteException, AllerEnPrisonException {  
        super.joueurArrive(j);  
        if (this.proprio == null) {  
            if (Outils.ouiNon(String.format("Voulez-vous acheter %s pour %d€ ?", this.nom,  
                                             this.prixAchat))) {  
                j.debiter(this.prixAchat);  
                this.setProprio(j);  
            }  
        } else {  
            if (j.equals(this.proprio))  
                System.out.printf("%s est à domicile%n", j);  
            else  
                this.payerLoyer(j, this.proprio);  
        }  
    }  
}
```



```
≡ Gare(Propriete).joueurArrive(Joueur) line: 56  
≡ Joueur.jouer(int, int) line: 98  
≡ Monopoly.jouer() line: 132  
≡ Monopoly.main(String[]) line: 107
```

Les exceptions contrôlées personnalisées

```
public class Joueur {  
    ...  
    public void jouer(int de1, int de2) throws FailliteException, AllerEnPrisonException {  
        /////////////////////////////////// déplacement ///////////////////////////////////  
        System.out.printf("%s a fait %d et %d aux dés%n", this.nom, de1, de2);  
        if (position.get().joueurPart(this)) {  
            for (int i = 0; i < de1 + de2 - 1; i++) {  
                position = position.suivant();  
                position.get().joueurPasse(this);  
            }  
            position = position.suivant();  
            position.get().joueurArrive(this);  
        }  
        ...  
    }  
    ...  
}
```



```
≡ Joueur.jouer(int, int) line: 98  
≡ Monopoly.jouer() line: 132  
≡ Monopoly.main(String[]) line: 107
```

Les exceptions contrôlées personnalisées

```
private static void jouer() throws AllerEnPrisonException {  
    ...  
    try {  
        if (Monopoly.de1.lancer() == Monopoly.de2.lancer()) {  
            nbDoubles++;  
            if (nbDoubles == 3) {  
                System.out.printf("%s a fait %d et %d aux dés%n", joueurCourant.get(),  
                                   Monopoly.de1.getFaceTiree(), Monopoly.de2.getFaceTiree());  
                throw new AllerEnPrisonException("3 doubles d'affilée");  
            }  
        }  
        joueurCourant.get().jouer(Monopoly.de1.getFaceTiree(), Monopoly.de2.getFaceTiree());  
    } catch (FailliteException e) {  
        Monopoly.joueurs.retirer(e.getJoueur());  
        System.err.println(e.getMessage());  
    }  
    ...  
}
```



Monopoly.jouer() line: 132
Monopoly.main(String[]) line: 107



Plusieurs catch

```
try {  
    val = Outils.s.nextInt();  
    ok = val >= min && val <= max;  
} catch (InputMismatchException e) {  
    ok = false;  
} catch (NoSuchElementException e) {  
    ok = false;  
    System.err.println("Aucune saisie");  
} catch (Exception e) {  
    ok = false;  
    System.err.println("Erreur");  
} finally {  
    Outils.s.nextLine();  
}
```

N'intercepte que les Exceptions de
type **InputMismatchException**

N'intercepte que les Exceptions de
type **NoSuchElementException**

Intercepte toutes les exceptions

L'ordre d'interception
est important si une
classe est héritée
d'une autre

Catch pour plusieurs types d'exceptions

```
try {
    val = Outils.s.nextInt();
    ok = val >= min && val <= max;
} catch (InputMismatchException|IllegalStateException e) {
    ok = false;
} finally {
    Outils.s.nextLine();
}
```

Intercepte les Exceptions de type
InputMismatchException et de type
IllegalStateException

Les exceptions

TP

