

# **Exceptions**

**Tristan Crolard**

Laboratoire CEDRIC  
Equipe « Systèmes Sûrs »

**`tristan.crolard@cnam.fr`**

**`cedric.cnam.fr/sys/crolard`**

# Exceptions

Les exceptions permettent de traiter les *situations exceptionnelles*.

Il existe deux sortes de situations exceptionnelles.

- ▶ celles qui **peuvent se produire exceptionnellement** (*checked exceptions*)  
Pour modéliser ces situations, on utilise la classe `Exception`
- ▶ celles qui **ne devraient pas se produire** (*unchecked exceptions*) :
  - Pour réagir aux situations inattendues dûes à des erreurs de programmation, on utilise la classe `RuntimeException`
  - Pour réagir aux situations inattendues dûes à des défaillances matérielles ou logicielles, on utilise la classe `Error`

**Remarque.** Vous êtes obligés de traiter ou propager les *checked exceptions* mais pas les *unchecked exceptions* (elles sont supposées être fatales).

# Syntaxe

- Déclaration d'une classe d'exceptions

```
class MyException extends Exception
```

- Méthode qui lève une « *checked* » exception :

```
public void f(...) throws MyException
```

- Levée d'exception :

```
throw new MyException()
```

- Traitement d'une exception :

```
try { ... f(...); ... } catch MyException(e) { ... }
```

**Note :** obligatoire pour une *checked exception* si elle n'est pas propagée (et donc listée dans la clause `throws`).