Exceptions

- Qu'est-ce que c'est?
- Exemples d'Exceptions Courantes
- Gestion des exceptions
- Gestion précises vs générique

Qu'est-ce que c'est?

- **Définition** : Une exception est une situation inattendue qui se produit pendant l'exécution d'un programme.
- **Objectif** : Permet de gérer les erreurs de manière contrôlée sans interrompre brusquement le programme.
- Les exceptions proviennent généralement d'une opération faites par l'utilisateur.

Exemples d'exceptions Courantes

1. DivideByZeroException

```
int a = 10;
int b = 0;
int result = a / b; // Erreur ici
```

2. NullReferenceException

```
string text = null;
Console.WriteLine(text.Length); // Erreur ici
```

3. IndexOutOfRangeException

```
int[] numbers = [ 1, 2, 3 ];
Console.WriteLine(numbers[5]); // Erreur ici
```

4. FormatException

```
int number = Convert.ToInt32("abc"); // Erreur ici
```

5. ArgumentNullException

```
void PrintLength(string str) {
    Console.WriteLine(str.Length); // Erreur ici si str est null
}
string str;
PrintLength(str);
```

6. ArgumentOutOfRangeException

```
List<int> list = [42];
int value = list[1]; // Erreur ici
```

7. StackOverflowException

```
void Recursive() {
    Recursive(); // Appel infini
}
```

8. CannotImplicitlyConvertType

```
int number = 10;
double result = number; // Pas d'erreur ici, conversion implicite
string text = number; // Erreur ici, conversion implicite impossible
```

Gestion des exceptions

Plutôt que de laisser le programme *planter*, nous pouvons utiliser un try/catch pour gérer les exceptions.

- try : Contient le code susceptible de générer une exception.
- catch : Contient le code pour gérer l'erreur.
- finally : Bloc optionnel qui s'exécute toujours, qu'une exception se produise ou non.

```
try {
    // Code qui peut générer une exception
} catch (Exception e) {
    // Code pour gérer l'erreur
} finally {
    // (Optionnel) Code qui s'exécute toujours
}
```

Gestion des exceptions (suite)

- Par exemple:

```
Console.WriteLine("Écrivez deux chiffres:");
int a = Convert.ToInt32(Console.ReadLine());
int b = Convert.ToInt32(Console.ReadLine()); // Si l'utilisateur inscrit 0 ici.
try {
    int result = a / b;
} catch (DivideByZeroException e) {
    Console.WriteLine("Erreur : Division par zéro !");
}
```

Gestion des exceptions (suite)

- Les exceptions peuvent être géré de façon précise ou générique.
  - Gestion Précise : Ce code attrape uniquement les exceptions de type DivideByZeroException, permettant d'afficher un message spécifique à cette erreur. Cela permet de traiter ce cas particulier de manière appropriée.
  - Gestion Générique : Ce code attrape toutes les exceptions, peu importe leur type. Bien qu'il soit plus flexible, il peut rendre difficile le diagnostic des problèmes, car il ne fournit pas de détails sur le type d'erreur survenue.
- Utiliser une gestion précise est généralement recommandé lorsque vous savez quelles exceptions spécifiques peuvent se produire, tandis qu'une gestion générique peut être utile pour capturer des erreurs inattendues.

Exemple de gestion précise

```
try {
    int a = 10;
    int b = 0;
    int result = a / b; // Tentative de division par zéro
} catch (DivideByZeroException e) {
    Console.WriteLine("Erreur : Division par zéro !");
}
```

Exemple de gestion générique

```
try {
    int a = 10;
    int b = 0;
    int result = a / b; // Tentative de division par zéro
} catch (Exception e) {
    Console.WriteLine($"Erreur : {e.Message}");
}
```

Gestion de plusieurs exceptions.

- C'est possible de gérer plusieurs exceptions d'un coup en ajoutant plusieurs *catch*.
- L'exception sera alors capturée par le *catch* correspondant, en priorisant le plus précis.
- Par exemple:

```
try {
    int[] numbers = [4,8,6,7,3,2];
    int index = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine($"Le mot à l'indice {index} est : {numbers[index]}");
}
catch (IndexOutOfRangeException) {
    Console.WriteLine("Erreur : L'indice fourni est hors des limites du tableau.");
}
catch (FormatException) {
    Console.WriteLine("Erreur : Vous n'avez pas entré un nombre entier valide.");
}
```