Module 8:
Utilisation de Référence des variables de type

Vue d'ensemble

- Utilisation de variables de type référence
- Utilisation des types référence communs
- La hiérarchie des objets
- Espaces de noms dans le Framework .NET
- Conversions de données

Utilisation de Référence des variables de type

- Comparaison entre types valeur et référence
- Déclaration et suppression des variables de référence
- Références invalides
- Références multiples à un même objet
- Utilisation de références comme paramètres de méthode

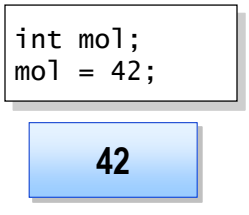
technofutur TIC

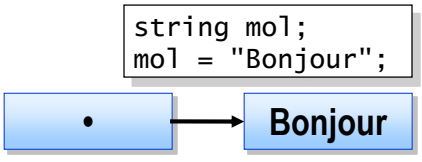
WAVENET
Intelligent IT Solutions

Comparaison Types valeur de référence Types

- Les types de valeur
  - La variable contient la valeur directement
  - Exemples: **char**,**int**

- types de référence
  - La variable contient une référence aux données
  - Les données sont stockées dans une zone de mémoire séparée





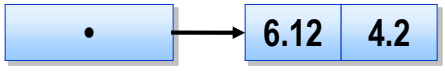
technofutur TIC

WAVENET
Intelligent IT Solutions

Déclarer et libération variables de référence

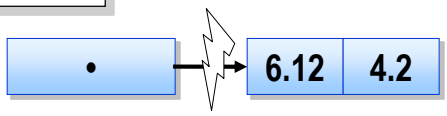
- Déclaration des variables de référence

```
coordinnate c1;
c1 = new coordinnate ();
c1.x = 6.12;
c1.y = 4.2;
```



- Libérer les variables de référence

```
c1 = null;
```

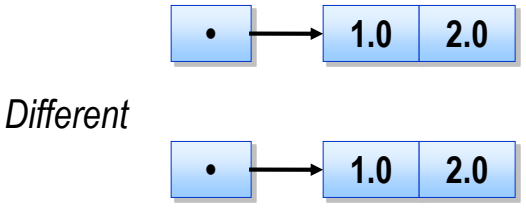


Références non valides

- Si vous avez des références invalides
  - Vous ne pouvez pas accéder aux membres ou variables
- Références non valides au moment de la compilation
  - Compilateur détecte l'utilisation de références non initialisées
- Références non valides au moment de l'exécution
  - Système va générer une Exception

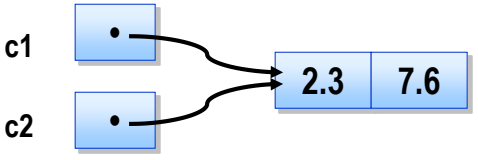
Comparaison de valeurs et comparaison Références

- Comparaison des types de valeur
  - == Et != Comparer les valeurs
- En comparant les types référence
  - == Et != Comparer les références, pas les valeurs



Références multiples à un même objet

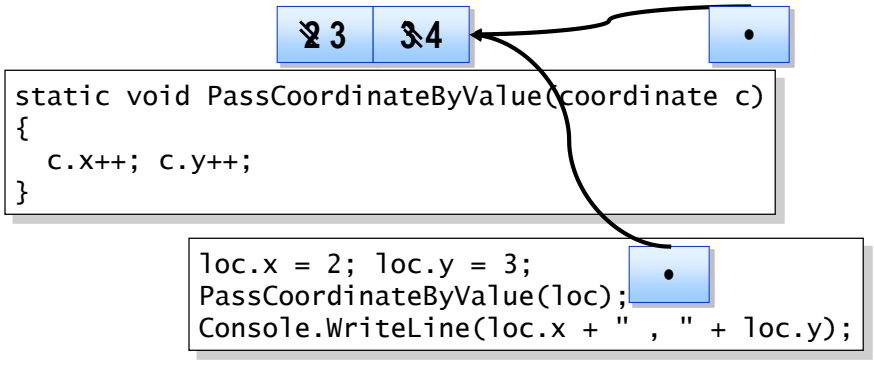
- Deux références peuvent se référer au même objet
  - Deux façons d'accéder au même objet en lecture / écriture




```
coordinate c1= new coordinate( );
coordinate c2;
c1.x = 2.3; c1.y = 7.6;
c2 = c1;
Console.WriteLine(c1.x + " , " + c1.y);
Console.WriteLine(c2.x + " , " + c2.y);
```

Utilisation de références comme paramètres de méthode

- Les références peuvent être utilisées comme paramètres
  - Lorsqu'il est passé par valeur, les données étant référencés peuvent être modifiés



technofutur TIC

WAVENET Intelligent IT Solutions

Utilisation des types communs de référence

- La classe Exception
- La classe String
- Méthodes communes de la classe String, opérateurs et Propriétés
- Comparaisons de strings
- Les opérateurs de comparaison de strings

technofutur TIC

WAVENET Intelligent IT Solutions

La classe Exception

- Exception est une classe
- Les objets d'exception sont utilisés pour lever des exceptions
  - Créer une **Exception** à l'aide **new**
  - Jetez l'objet en utilisant **throw**
- Les types d'exception sont des sous-classes de Exception

Classe String

- Tableau de caractères Unicode
- Raccourci pour System.String
- Immuable

```
string s = "Bonjour";

s [0] = 'c'; / / erreur de compilation
```

Méthodes communes cordes, opérateurs et Propriétés

- []
- Propriété Length
- Méthode Copy
- Méthode Concat
- Méthode Trim
- Méthodes de ToUpper et ToLower

Comparaisons de chaînes

- Methode Equals
  - Comparaison de valeur
- Méthode Compare
  - Plus de comparaisons
  - Option : insensible à la casse
- Options de comparaison liées à la localisation

Les opérateurs de comparaison de chaînes

- Le == et != sont des Opérateurs surchargés pour les string
- Ils sont équivalents à String.Equals et !String.Equals

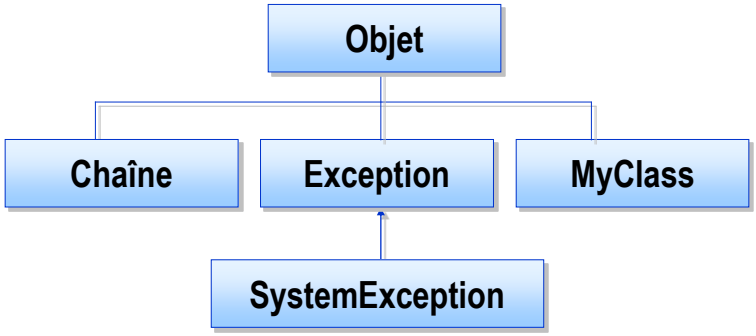
```
string a = "Test";
string b = "Test";
if (a == b) ...   // Returns true
```


◆ La hiérarchie de l'objet

- Le type Objet
- Méthodes communes
- Réflexion

Le type objet

- Synonyme de System.Object
- La classe de base pour toutes les classes



Méthodes communes

- Méthodes communes pour tous les types reference:
  - **ToString**
  - **Equals**
  - **GetType**
  - **GetHashCode**
  - **Finalize** //Inaccessible

Réflexion

- Vous pouvez interroger le type d'un objet
- Espace de noms System.Reflection
- L'opérateur typeof renvoie le type d'une classe
  - Uniquement pour les classes disponibles a la compilation
- GetType dans System.Object
  - Informations disponibles à l'éxécution

◆ Espaces de noms dans le. NET Framework

- System.IO, espace de noms
- System.Xml
- System.Data
- D'autres espaces de noms utiles

System.IO, espace de noms

- Accès au système de fichiers d'entrée / sortie
  - File, Directory
  - StreamReader, StreamWriter
  - FileStream
  - BinaryReader, BinaryWrite

System.Xml

- support de XML
- Diverses normes liées à XML

System.Data

- System.Data.SqlClient
  - SQL Server NET Data Provider.
- System.Data
  - Se compose principalement des classes qui constituent l'architecture ADO.NET

D'autres espaces de noms utiles

- Espace de noms System
- Espace de noms System.Net
- Espace de noms System.Net.Sockets
- Espace de noms System.Windows.Forms

◆ Conversions de données

- Conversion des types valeur
- Conversions parent / enfant
- L'opérateur is
- L'opérateur as
- Les conversions et le type d'objet
- Conversions et Interfaces
- Boxing et unboxing

Conversion des types de valeur

- Les conversions implicites
- Les conversions explicites
  - Cast opérateur
- Exceptions
- Classe System.Convert
  - Gère les conversions interne

Conversions parent / enfant

- Conversion à la référence de la classe parent
  - Implicite ou explicite
  - Réussit toujours
- Conversion de référence de classe de l'enfant
  - Une conversion explicite requise
  - Vérifiera que la référence est du type correct
  - Élèvera **InvalidCastException** sinon

L'opérateur is

- Retourne true si une conversion peut être faite

```
Bird b;
if (a is Bird)
    b = (Bird) a; // Safe
else
    Console.WriteLine("Not a Bird");
```

L'opérateur as

- Conversion entre types référence
- En cas d'erreur
  - Renvoie null
  - Ne leve pas une exception

```
Bird b = a as Bird; // Convert

if (b == null)
Console.WriteLine("Not a bird");
```


Les conversions et le type d'objet

- L'objet est le type de base de toutes les classes
- Toute référence peut être attribué à une variable objet
- Toute variable objet peut être attribué à toute référence
  - Avec la conversion et les vérifications de type approprié
- Le type objet et l'opperateur as

```
object ox;
ox = a;
ox = (object) a;
ox = a as object;
```

```
b = (Bird) ox;
b = ox as Bird;
```

Conversion et Interfaces

- Une interface ne peut être utilise que pour accéder aux membres qu'elle définit
- Les autres méthodes et variables de la classe ne sont pas accessibles via l'interface

Boxing et unboxing

- Boxing
- Unboxing
- Appel de méthodes d'objets sur des types valeur