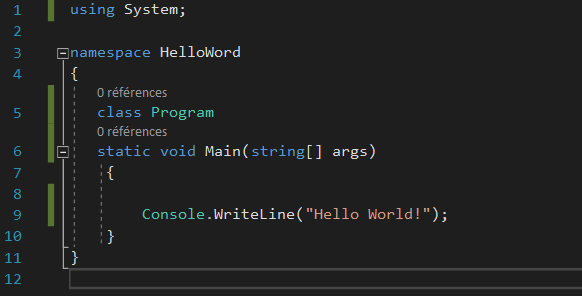
***INTRODUCTION :***

*Le C# est un langage de programmation orienté objet, il est un dérivé du C++ et très proche du JAVA, des notions ont été néanmoins ajouté tel que la surcharge des opérateurs par exemple. Enfin il est utilisé pour développer des applications web.*

***Débuter en C# :***



***Main***

*La méthode « Main » crée des objets et exécute les autres méthodes, c’est la fonction principale du code comme son nom l’indique. C’est aussi le point d’entrée c’est donc la première méthode appelait. Celui-ci est déclaré à l’intérieur d’une « classe » ou d’un « struct ».*

***Structure Générale d’un programme C#***

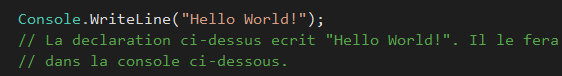
*Les programmes C# se composent d’un ou plusieurs fichiers. Chaque fichier peut contenir zéro ou plusieurs espaces de noms. Un espace de noms peut contenir des types tels que des classes, des structs, des interfaces ou encore des énumérations.*

***Convention de Codage***

*Comme dans la plupart des langages, il existe des conventions de codage. Elles ont plusieurs objectif tel que :*

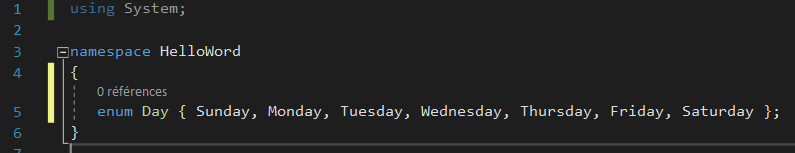
* *Une cohérence dans le code.*
* *Une rapidité de compréhension pour le lecteur.*
* *Et enfin elle facilite les modifications du code.*

*Prenons l’exemple du commentaire :*

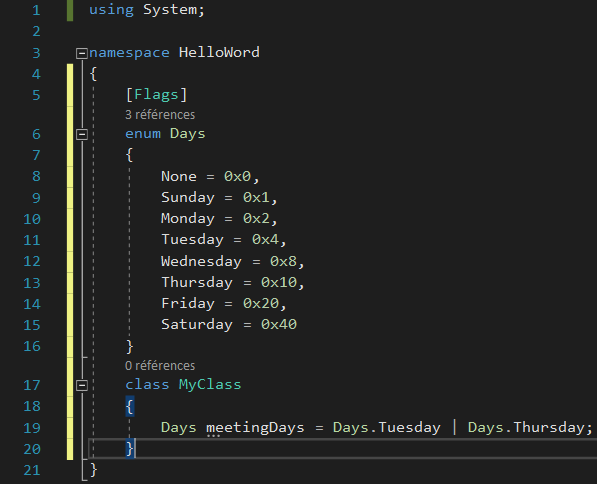
**

*On ici plusieurs conventions tel que tout le commentaire n’est pas sur la même ligne, ou encore que celui-ci commence par une majuscule et termine par un point. Enfin il est nécessaire de mettre un espace entre le délimiteur de commentaire (//) et le texte du commentaire.*

***Types énumération :***

**

*Le type énumération est très utilise en programmation et en particulier en C#, il permet par exemple de définir un ensemble constant pouvant être assigné à une variable. Si par exemple on souhaite* définir une variable dont la valeur représente un jour de la semaine. Il n’y a que sept valeurs significatives qui pourront être stockées par cette variable. On utilise ainsi un type énumération, qui est déclaré à l’aide du mot clé « enum ».

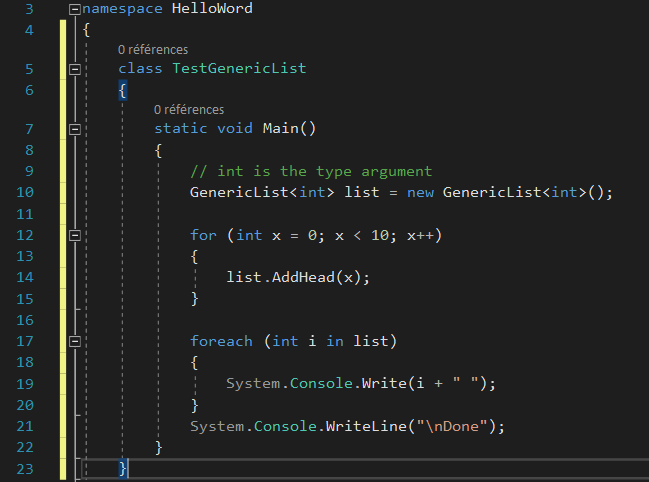


Enfin un autre point très important de l’utilisation du type enum est que l’on peut avoir deux valeurs attribuées à une variable comme on peut le voir dans l’exemple ci-dessus grâce à l’indicateur binaire « OU » --> « | » en C#.

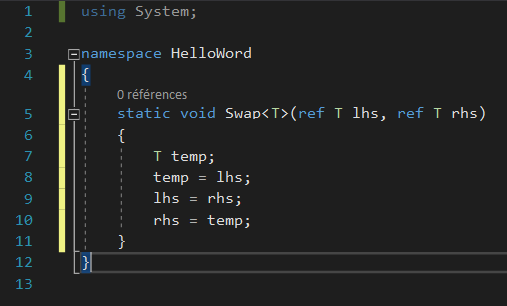
***Générique :***

Les méthodes et les classes génériques combinent la réutilisabilité, la cohérence des types et l’efficacité, ce que ne peuvent pas faire leurs équivalents non génériques. Les génériques sont plus fréquemment utilisés dans des collections et des méthodes qui agissent sur eux.

Bien sûr, vous pouvez également créer des types et des méthodes génériques personnalisés pour fournir des solutions et des modèles de conception généralisés qui soient efficaces et de type sécurisé. L’exemple de code suivant montre une classe de liste liée générique simple.



Les génériques constituent la solution à une limitation dans les versions antérieures du **Common Language Runtime** et du langage C# dans lesquelles la généralisation s’effectue par un cast de types vers et depuis le type de base universel [Object](https://docs.microsoft.com/fr-fr/dotnet/api/system.object). En créant une classe générique, vous pouvez créer une collection qui est de type sécurisé au moment de la compilation.



***Lexique***

**Package** : Ensemble de logiciels munis d'une documentation, conçus pour répondre à des besoins spécifiques et permettre une utilisation autonome.

**Garbage collection :** gère l’allocation et la libération de mémoire pour votre application. Chaque fois que vous créez un objet, le Common Langage Runtime alloue de la mémoire pour l’objet à partir du tas managé. Aussi longtemps que de l'espace d'adressage est disponible dans le tas managé, le Runtime continue à allouer de l'espace pour de nouveaux objets.

**Programmation de type objet**: En [programmation orientée objet](https://fr.wikipedia.org/wiki/Programmation_orient%C3%A9e_objet), un objet est créé à partir d'un modèle appelé [classe](https://fr.wikipedia.org/wiki/Classe_(informatique)) ou [prototype](https://fr.wikipedia.org/wiki/Programmation_orient%C3%A9e_prototype), dont il hérite les comportements et les caractéristiques. Les comportements et les caractéristiques sont typiquement basés sur celles propres aux choses qui ont inspiré l'objet : une personne (avec son [état civil](https://fr.wikipedia.org/wiki/%C3%89tat_civil)), un dossier, un produit.

**Common Language Runtime (CLR)** c’est le nom choisi par [Microsoft](https://fr.wikipedia.org/wiki/Microsoft) pour le composant de [machine virtuelle](https://fr.wikipedia.org/wiki/Machine_virtuelle) du [framework .NET](https://fr.wikipedia.org/wiki/Framework_.NET). Il s'agit de l'implémentation par Microsoft du standard [Common Language Infrastructure](https://fr.wikipedia.org/wiki/Common_Language_Infrastructure) (CLI) qui définit l'[environnement d'exécution](https://fr.wikipedia.org/wiki/Environnement_d%27ex%C3%A9cution) des codes de programmes. Le CLR fait tourner une sorte de [bytecode](https://fr.wikipedia.org/wiki/Bytecode" \o "Bytecode) nommé [Common Intermediate Language (CIL)](https://fr.wikipedia.org/wiki/Common_Intermediate_Language). Le compilateur à la volée transforme le code CIL en code natif spécifique au [système d'exploitation](https://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27exploitation). Le CLR fonctionne sur des systèmes d'exploitation [Microsoft Windows](https://fr.wikipedia.org/wiki/Microsoft_Windows).