Fiche procédure C#

Table des matières

Installation de C#

Sous Windows

- Installe Visual Studio Community ou Visual Studio Code avec l'extension C#.
- Installe le SDK .NET pour pouvoir compiler et exécuter des programmes C#.

Sous macOS et Linux

- Installe Visual Studio Code avec l'extension C#.
- Installe le SDK .NET via le terminal :
- bash

```
# macOS
brew install --cask dotnet-sdk

# Linux (Debian/Ubuntu)
sudo apt update
sudo apt install dotnet-sdk-7.0
```

-

Pour vérifier l'installation :
bash

```
dotnet --version
```

Création d'un projet C#

1. Ouvre un terminal ou Visual Studio.
2. Pour créer un projet console :

3. bash

```
dotnet new console -n MonProjetCSharp
cd MonProjetCSharp
```

4.
5. Pour compiler et exécuter :
6. bash

```
dotnet run
```

7.

Utilisation de C#

- C# est un langage orienté objet, utilisé pour des applications console, desktop, web, mobiles, etc.
- Un programme C# commence par une méthode `Main`.
- Utilise des classes, méthodes, variables, boucles, conditions, collections, etc.

Exemple de structure de base :
csharp

```
using System;

namespace MonProjet
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Bonjour, C#!");
        }
    }
}
```

Bonnes pratiques

- Utilise les conventions de nommage : PascalCase pour les classes et méthodes, camelCase pour les variables.
- Applique le principe DRY (Don't Repeat Yourself) : évite la duplication de code, factorise dans des méthodes ou classes réutilisables.

- Commente et documente ton code pour faciliter la maintenance et la compréhension.
- Gère les exceptions avec des blocs `try-catch` uniquement là où des erreurs sont attendues (lecture de fichiers, réseau, etc.), et capture des exceptions spécifiques plutôt que générales.
- Utilise les interfaces et abstractions pour rendre ton code flexible et évolutif.
- Structure ton code en petites méthodes claires et simples.
- Privilégie les types de données C# (`int`, `string`, etc.) et utilise `var` uniquement si le type est évident.
- Nomme clairement tes variables et méthodes pour décrire leur usage.
- Documente les classes et méthodes avec des commentaires XML si besoin.
- Teste ton code et utilise des outils de logs pour surveiller le comportement de l'application.

Exemple de code C#

csharp

```csharp
using System;

namespace Exemple
{
    class Program
    {
        // Fonction qui retourne la TVA calculée
        static double CalculerTva(double montant)
        {
            return montant * 0.2;
        }

        static void Main(string[] args)
        {
            Console.WriteLine("Entrer un montant :");
            double montant =
Convert.ToDouble(Console.ReadLine());
            double tva = CalculerTva(montant);
            Console.WriteLine($"La TVA sur {montant} € est de
{tva} €.");
        }
    }
}
```