

# Namespace MotsGlissés

## Classes

### [Dictionnaire](#)

Classe représentant un dictionnaire. Elle ne contient que le chemin vers le fichier représentant le dictionnaire.

### [Extras](#)

Cette classe contient des méthodes utiles pour le projet. Elles ne sont pas directement liées au jeu, mais rendent son développement plus facile. Elles sont statiques pour pouvoir être appelées sans instancier la classe, et appelables depuis n'importe où.

### [Jeu](#)

Classe représentant une partie de Mot Glissé

### [Joueur](#)

Classe représentant un joueur

### [Plateau](#)

## Structs

### [Extras.Position](#)

Represents a position in a two-dimensional space.

# Class Dictionnaire

Namespace: [MotsGlissés](#)

Assembly: MotsGlissés.dll








Classe représentant un dictionnaire. Elle ne contient que le chemin vers le fichier représentant le dictionnaire.

```
public class Dictionnaire
```

## Inheritance

[object](#)  ← Dictionnaire

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#) 

# Constructors

## Dictionnaire(string)

Constructeur de la classe Dictionnaire

```
public Dictionnaire(string chemin)
```

## Parameters

**chemin** [string](#) 

Chemin vers le fichier représentant le dictionnaire

# Methods

## RechDichoRecuratif(string)

Recherche un mot dans le dictionnaire, via un stream, en accédant itérativement à la ligne correspondante au premier caractère du mot Nécessite que le dictionnaire soit trié, et que le format du

fichier soit correct et en accord avec le sujet

```
public bool RechDichoRecuratif(string input)
```

## Parameters

input [string](#)

Mot recherché

## Returns

[bool](#)

bool: Vrai si la mot a été trouvé, faux si le mot n'est pas trouvé, ou est nul, de taille nulle, ou ne contient pas que des lettres

## Tri\_Fusion()

Effectue un Tri Fusion sur chacune des lignes du dictionnaire Possible de le faire grâce à des streams  
Implémentation de Fusion dans Extras.cs

```
public void Tri_Fusion()
```

## toString()

Représente le dictionnaire sous forme de chaîne de caractères Affiche la langue du dictionnaire (français par défaut ), puis le nombre de mots par lettre Implémenté avec un stream Assume que le dictionnaire soit conforme au format donné dans le sujet

```
public string toString()
```

## Returns

[string](#)

string: la chaîne de caractère

# Class Extras

Namespace: [MotsGlissés](#)

Assembly: MotsGlissés.dll








Cette classe contient des méthodes utiles pour le projet. Elles ne sont pas directement liées au jeu, mais rendent son développement plus facile. Elles sont statiques pour pouvoir être appelées sans instancier la classe, et appelables depuis n'importe où.

```
public class Extras
```

## Inheritance

[object](#)  ← Extras

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

# Methods

## Fusion(string[])

Implémentation récursive du tri fusion sur un tableau de string. Le polymorphisme est simple à ajouter, mais n'est pas nécessaire pour le projet.

```
public static string[] Fusion(string[] tab)
```

## Parameters

tab [string](#)  []

Tableau à trier

## Returns

[string](#)  []

string[]: le tableau qui a été trié

# ReadLine(TimeSpan)

Equivalent de ReadLine mais avec un timeout

```
public static string? ReadLine(TimeSpan timeout)
```

## Parameters

**timeout** [TimeSpan](#)

La durée maximale d'attente pour une entrée.

## Returns

[string](#)

La chaîne de caractères saisie par l'utilisateur, ou null si aucune saisie n'a été effectuée dans le délai imparti.

# Split(string[])

Méthode permettant de couper en deux un tableau de chaînes de caractères. Le polymorphisme est simple à ajouter, mais n'est pas nécessaire pour le projet.

```
public static (string[], string[]) Split(string[] tab)
```

## Parameters

**tab** [string](#)[]

Le plateau à couper

## Returns

([string](#)[], [string](#)[])

(string[], string[]): un tuple des tableaux coupés, dans l'ordre de coupe

# Struct Extras.Position

Namespace: [MotsGlissés](#)

Assembly: MotsGlissés.dll

Represents a position in a two-dimensional space.

```
public struct Extras.Position
```

## Inherited Members

[ValueType.GetHashCode\(\)](#), [ValueType.ToString\(\)](#), [object.Equals\(object, object\)](#), [object.GetType\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Constructors

### Position(int, int)

Constructeur de la structure Position

```
public Position(int x, int y)
```

## Parameters

x [int](#)

La position en x

y [int](#)

La position en y

## Properties

### X

Propriété en lecture seule pour lire la position X, qui se réfère à la largeur de la console.

```
public int x { get; }
```

Property Value

[int](#)

## Y

Propriété en lecture seule pour lire la position Y, qui se réfère à la hauteur de la console.

```
public int Y { get; }
```

Property Value

[int](#)

## Methods

### Equals(object?)

Override de la méthode Equals pour tester l'égalité de deux positions

```
public override bool Equals(object? obj)
```

Parameters

**obj** [object](#)

Objet à tester

Returns

[bool](#)

bool: vrai si les positions sont égales en x et y, faux dans le cas contraire, ou si elle est nulle.

# Operators

## operator ==(Position, Position)

Override de l'opérateur == pour tester l'égalité de deux positions

```
public static bool operator ==(Extras.Position left, Extras.Position right)
```

### Parameters

**left** [Extras.Position](#)

Première position

**right** [Extras.Position](#)

Seconde position

### Returns

[bool](#)

bool: vrai si les deux positions sont égales

## operator !=(Position, Position)

Override de l'opérateur != pour tester l'inégalité de deux positions

```
public static bool operator !=(Extras.Position left, Extras.Position right)
```

### Parameters

**left** [Extras.Position](#)

Première position

**right** [Extras.Position](#)

Seconde position

### Returns



[bool](#) 

bool: vrai si l'inégalité est vérifiée, faux snn

# Class Jeu

Namespace: [MotsGlissés](#)

Assembly: MotsGlissés.dll








Classe représentant une partie de Mot Glissé

```
public class Jeu
```

## Inheritance

[object](#)  ← Jeu

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#) 

## Constructors

Jeu(Dictionnaire, Plateau, List<Joueur>, TimeSpan, TimeSpan)

Constructeur de la classe Jeu

```
public Jeu(Dictionnaire dico, Plateau plateau, List<Joueur> joueur, TimeSpan tempsJoueur,  
TimeSpan tempsJeu)
```

## Parameters

dico [Dictionnaire](#)

Dictionnaire utilisé pour valider

plateau [Plateau](#)

joueur [List](#)  <[Joueur](#)>

tempsJoueur [TimeSpan](#) 

tempsJeu [TimeSpan](#) 

# Methods

play()

Méthode

```
public void play()
```

# Class Joueur

Namespace: [MotsGlissés](#)

Assembly: MotsGlissés.dll

Classe représentant un joueur

```
public class Joueur
```

## Inheritance

[object](#)  ← Joueur

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#) 

# Constructors

## Joueur(string)

Cosntructeur simple de la classe Joueur Tout est mis à 0

```
public Joueur(string nom)
```

## Parameters

nom [string](#) 

Nom du joueur

## Joueur(string, int, List<string>)

Constructeur de la classe Joueur

```
public Joueur(string nom, int score, List<string> listeMotTrouver)
```

## Parameters

**nom** [string](#)

Nom du joueur

**score** [int](#)

Score initial du joueur

**listeMotTrouver** [List](#) <[string](#)>

Liste initiale des mots trouvés

## Properties

### Nom

Propriété Nom en lecture seule

```
public string Nom { get; }
```

### Property Value

[string](#)

### Score

Propriété Score en lecture seule

```
public int Score { get; }
```

### Property Value

[int](#)

## Methods

## Add\_Mot(string)

Ajoute un mot à la liste des mots trouvés par le joueur, seulement si ce mot n'a pas déjà été trouvé

```
public void Add_Mot(string mot)
```

### Parameters

mot [string](#)

Mot qu'il faut ajouter à la liste des mots trouvés

## Add\_Score(int)

Ajoute un score au score du joueur Le score est calculé depuis plateau, à partir du poids des lettres et d'un bonus de 5 points pour avoir trouvé un mot

```
public void Add_Score(int val)
```

### Parameters

val [int](#)

Valeur à ajouter

## Contient(string)

Test si un mot a déjà été trouvé par le joueur

```
public bool Contient(string mot)
```

### Parameters

mot [string](#)

Le mot qu'on teste

### Returns

[bool](#)

vrai si le mot a déjà été trouvé, faux sinon

## toString()

Condense les informations du joueur en une chaîne de caractère

```
public string toString()
```

Returns

[string](#)

La chaîne de caractère représentant le joueur

# Class Plateau

Namespace: [MotsGlissés](#)








Assembly: MotsGlissés.dll

```
public class Plateau
```

## Inheritance

[object](#)  ← Plateau

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#) 

# Constructors

## Plateau()

Crée un plateau de maniere aleatoire à partir d'un fichier .txt conforme au sujet Prochaine implémentations : sélection du fichier et de la taille du plateau

```
public Plateau()
```

## Plateau(string)

Constructeur qui crée un plateau à partir d'un fichier .csv Se référer à ToRead

```
public Plateau(string filePath)
```

## Parameters

filePath [string](#) 

# Properties



# NbLettres

Propriété en lecture seule représentant le nombre de lettres restantes sur le plateau

```
public int NbLettres { get; }
```

Property Value

[int](#)

## Methods

### GetScore(string)

Récupère le score d'un mot en fonction des valeurs des lettres

```
public int GetScore(string mot)
```

Parameters

mot [string](#)

Mot dont on cherche le scor associé

Returns

[int](#)

int: valeur Scrabble du mot choisi

### Maj\_Plateau(Stack<Position>)

Mise à jour de la matrice représentant le plateau selon les positions données Remplace les positions par des espaces et fait descendre les lettres Les positions sont données dans une stack pour éviter les modifications intempestives des positions

```
public void Maj_Plateau(Stack<Extras.Position> positions)
```

## Parameters

**positions** [Stack](#) <[Extras.Position](#)>

Pile des positions à modifier

## Recherche\_Mot(string)

```
public (bool, Stack<Extras.Position>) Recherche_Mot(string mot)
```

## Parameters

**mot** [string](#)

## Returns

([bool](#), [Stack](#) <[Extras.Position](#)>)

## ToFile(string)

Sauvegarde l'instance du plateau dans un fichier en respectant la structure précisée

```
public void ToFile(string filePath)
```

## Parameters

**filePath** [string](#)

Chemin du fichier d'écriture

## ToRead(string)

Lecture d'un fichier .csv et remplissage du plateau

```
public void ToRead(string filePath)
```

## Parameters

`filePath` [string](#) 

Chemin vers le fichier

## toString()

Retourne une chaîne de caractères qui représente le plateau

```
public string toString()
```

## Returns

[string](#) 

Chaîne de caractère représentant le tableau de jeu