TP : Jeu de gestion, je créer mon village

HelloMyVillage

Exercice 00

Créer une classe `Village`,
qui aura comme attribut :

`string name` qui ne sera accessible que par la méthode `getName()` et qui sera instancier a la creation d'un nouveau village.

Give me some ressources

Dans un fichier Ressources.cs

Créer une classe `Ressources`
Elle possédera deux attributs priver :

- 📌 `int woods`
- 📌 `int stones`

Give me some ressources

Ces ressources ne seront accessible que par des getter respectif :

- 📌 `getWood()`
- 📌 `getStone()`

Give me some ressources

Rajoutez des fonctions pour utiliser vos ressources avec les prototype suivants :

- `public void useStone(int nbr)`
- `public void useWood(int nbr)`)

nbr est le nombre de ressources utilisées

!! attention on ne pourra pas utiliser plus de ressource que l'on en a.

Give me some ressources

Créer un constructeur `Ressources` qui ne prendra aucun paramètre et qui initialisera woods a 10 et stones a 10.

Une fois fait;

Rajoutez a votre village un attribut priver `myRessources` de type `Ressources`
Attribut qui sera instancier dans le constructeur de `Village` en appelant le constructeur de `Ressources`.

Give me some ressources

Rajoutez des getters

- 📌 `getWood()`
- 📌 `getStone()`

qui appellerons, `getWood()`, `getStone()` de myRessources.

EXERCICE 01 : What is a kingdom without subject ?

Exercice 1

Créer un fichier House.cs
Dans lequel vous créerez une classe public House
Elle aura 3 attributs :

- 📌 `stone_needed` (coût en pierre)
- 📌 `wood_needed` (coût en bois)
- 📌 `villageois` (nbr de villageois que ce bâtiment apportera)

Exercice 1

C'est attribut seront accessible uniquement avec la class House en non avec un object de classes.
Ils ne seront pas pas instantiable.
Stone_needed et wood_needed renverront toujours 3.
Villageois renverra toujours 10.

Exercice 1

Tester son code :

```
Console.WriteLine(House.wood_needed) // affichera 3
Console.WriteLine(House.stone_needed) // affichera 3
// Console.WriteLine(House.stone_needed++) // --> Erreur
House justAHouse = new House()
// justAHouse.wood_need // --> Erreur
// justAHouse.villageois // --> Erreur
// justAHouse.villageois = 20 // --> Erreur
```

Exercice 1

Une fois fait, ajoutez a la class Village votre maison,

```
public House chefHome;
```

Ainsi que le nombre de sujet de votre village :

```
private int villageois = 0;
```

Initialiser cette variable dans le constructeur de village,

N'oubliez pas de rajouter le nombre de villageois disponible dans la chefHome au nombre de villageois de votre village.

Exercice 2 : One house is not suffisant for a village

Exercice 2

Dans votre class Village,
Creer un attribut `listHouse` qui sera une un tableau de maison public,
Dans le constructeur de village, vous l'initialiserez;
apres avoir initialisez `chefHome`, telque votre `listHouse` continient `chefHome`.

Exercice 2

Test son code,

```
Village myVillage = new Village("Victor le createur");

myVillage.getName(); // affichera Victor le createur
Console.WriteLine(myVillage.listHouse.Length); // affichera 1
```

Exercice 2

Ajoutez des maison a votre list,

Creer une methode public qui ajoutera une `House` a votre `listHouse`.
Prototype : `public void addHouse()`

Exercice 2

Test son code,

```
Village myVillage = new Village("Victor le createur");

myVillage.getName(); // affichera Victor le createur
Console.WriteLine(myVillage.listHouse.Length); // affichera 1
myVillage.addHouse();
myVillage.addHouse();
Console.WriteLine(myVillage.listHouse.Length); // affichera 3
```

Exercice 2

Une fois fait, on changera un peu l'attribut villageois.
Deja on le rendra public.
Ensuite pour savoir combien on a de `villageois` dans notre `Village`, on retournera le produit du nombre de maison contenue dans `listHouse` par le nombre de `villageois` par `House`.

Exercice 2

```
Village myVillage = new Village("Victor le createur");

myVillage.getName(); // affichera Victor le createur
Console.WriteLine(myVillage.listHouse.Length); // affichera 1
myVillage.addHouse();
myVillage.addHouse();
Console.WriteLine(myVillage.listHouse.Length); // affichera 3
Console.WriteLine(myVillage.villageois); // affichera 30
```

Exercice 2

On passera juste addHouse en priver a la fin quand les test seront valide.

Exercice 3 : Sujets au travail !!

Exercie 3

Vous avez 10 villageois, vous avez 10 pierres, vous avez 10 bois.
Mais ce n'est pas suffisant !!!
Mettez vos sujets au travails pour leurs chef (vous).

Creer un fichier Mine.cs
Avec une classe `Mine`

Exercie 3

Sur le meme model de `House`
Creer des les attributs suivant :

- 📌 gain_stone (retournera 10)
- 📌 stone_cost (retournera 2)
- 📌 wood_cost (retournera 1)

```
Console.WriteLine(Mine.gain_stone) // affichera 10
Console.WriteLine(Mine.stone_cost) // affichera 2
Console.WriteLine(Mine.wood_cost) // affichera 1
```

Exercie 3

Creer un construceur `Mine()`
qui affichera dans le terminal `"Mine created"`)

Creer une methode public `mineStone` qui prendra en argument un nombre de villageois, et qui renvera le produit de
`nombre de villageois * gain_stone`

Ajouter a la classes `Village` :

📌 un attribut `Mine` qui sera instancier dans le constructeur.

📌 une methode public `mineStone` qui prendra en parametre un nombre de villageois.

Lorsqu'un villageois utilise la methode `mineStone` celui-ci consomme `Mine.stone_cost` et `Mine.wood_cost`
Et celui-ci vous rapportera `Mine.gain_stone` (creer une methode dans la classes `Ressource` `public addStone(int nbr)` qui ajouteras nbr a stone).

`mineStone` devra etre protege contre :

- 📌 si en parametre, il y a plus de nbr de villageois que votre village n'en possede.
  Auquel cas vous ecrirez dans le terminal
  `"Il n'y a pas assez de villageois"`
- 📌 si l'operation coute plus de ressources que vous n'en possedez.
  Auquel cas vous ecrirez dans le terminal
  `"Il vous manque des ressources"`

Tester son code :

```
Village myVillage = new Village("Victor le createur");

myVillage.mineStone(50); // Affichera: Il n'y a pas assez de villageois

Console.WriteLine(myVillage.getStone()); // Affichera 10
Console.WriteLine(myVillage.getWood());// Affichera 10

myVillage.mineStone(6); // Affichera : Il n'y a pas assez de ressources
Console.WriteLine(myVillage.getStone()); // Affichera 10
Console.WriteLine(myVillage.getWood()); // Affichera 10
myVillage.mineStone(5);
myVillage.mineStone(5);
Console.WriteLine(myVillage.getStone()); // Affichera 90
Console.WriteLine(myVillage.getWood()); // Affichera 0
myVillage.mineStone(5); // Affichera : Il n'y a pas assez de ressources
```

Exercice 4 : Where is the Wood ?

Exercice 4

Vous avez de la pierre.
Maintenant on veux du bois.

A l'image de `Mine`, creer une classes `Forest` dans un fichier Forest.cs

Elle aura les attributs suivants (comme Mine en remplacant gain_stone par gain_wood) :

- 📌 `gain_wood`
- 📌 `stone_cost`
- 📌 `wood_cost`

Exercice 4

```
Console.WriteLine(Forest.gain_wood) //affichera 10
Console.WriteLine(Forest.stone_cost) //affichera 2
Console.WriteLine(Forest.wood_cost) //affichera 1

Forest test = new Forest();

// test.wood_cost // --> erreur
// test.gain_wood // --> erreur
// Forest.gain_wood = 123 // --> erreur
// test.gain_wood  = 329 // --> erreur
```

Ainsi qu'une methode `cutWood` qui prendra un int en parametre (nombre de villageois, toujours comme dans Mine).
Et qui renvera le produit du parametre de la methode et de `gain_wood`

Ajouter a `Village` une attribut `forest` du type `Forest` qui sera instancier dans le constructeur.

Ajouter une methode `cutWood` a `Village` qui, comme pour `mineStone`.
Elle prendra un `int` (representant des villageois) en parametre.
Cette methode utilisera les ressources demander par la classe `Forest` (`stone_cost` et `wood_cost`), et produira `gain_wood`; multiplier par le parametre d'entree.

Penser a proteger la methode !!!

Tester son code :

```
Village myVillage = new Village("Victor le createur");
myVillage.cutWood(50); // affichera Il n'y a pas assez de villageois
Console.WriteLine(myVillage.getStone()); // afficher 10
Console.WriteLine(myVillage.getWood()); // afficher 10
myVillage.cutWood(6); // affichera Il n'y a pas assez de ressource
Console.WriteLine(myVillage.getStone()); // afficher 10
Console.WriteLine(myVillage.getWood()); // afficher 10
myVillage.cutWood(5);
myVillage.cutWood(5); // affichera Il n'y a pas assez de ressource
Console.WriteLine(myVillage.getStone()); // afficher 0
Console.WriteLine(myVillage.getWood()); // afficher 55
myVillage.cutWood(5); // affichera Il n'y a pas assez de ressource
```

Exercice 5 : From a village to a kingdom

Exercice 5

Du bois,
De la pierre,
Maintenant la population dois s'agrandir.

Vous avez tout pour produire des ressources a l'infinie,
vous avez meme creer une methode pour ajouter des maison a votre village.
Maintenant passons de la theorie a la pratique et mettons la mains a la bourse.

Exercice 5

Creer une methode pour pour construire des maisons :

- 📌 nom de la methode `buildHouse`
- 📌 parametre `int` , qui representera le nombre de maison que vous voulez construire.

Evidement ces maisons ne sont pas gratuites.

tester son code :

```
Village myVillage = new Village("Victor le createur");
Console.WriteLine(myVillage.getName());

myVillage.cutWood(2);
myVillage.mineStone(2);
myVillage.cutWood(4);
myVillage.mineStone(4);

Console.WriteLine(myVillage.getWood()); // affiche 58
Console.WriteLine(myVillage.getStone()); // affiche 46

myVillage.buildHouse(2);
Console.WriteLine(myVillage.listHouse.Length); // affiche 3
Console.WriteLine(myVillage.villageois); // affiche 30

myVillage.cutWood(10);
myVillage.mineStone(10);
myVillage.cutWood(13);
myVillage.mineStone(13);
myVillage.cutWood(16);
myVillage.mineStone(16);
Console.WriteLine(myVillage.getWood()); // affiche 364
Console.WriteLine(myVillage.getStone()); // affiche 274
```

```
Village myVillage = new Village("Victor le createur");
Console.WriteLine(myVillage.getName());

myVillage.cutWood(2);
myVillage.mineStone(2);
myVillage.cutWood(4);
myVillage.mineStone(4);

Console.WriteLine(myVillage.getWood()); // affiche 58
Console.WriteLine(myVillage.getStone()); // affiche 46

myVillage.buildHouse(2);
Console.WriteLine(myVillage.listHouse.Length); // affiche 3
Console.WriteLine(myVillage.villageois); // affiche 30

myVillage.cutWood(15);
myVillage.mineStone(15); // affiche Il n'y a pas assez de ressources
Console.WriteLine(myVillage.getWood()) // affiche 187
Console.WriteLine(myVillage.getStone()); // affiche 10
myVillage.buildHouse(4); // affiche Il n'y a pas assez de ressources
```

Exercice 6 : Tester son code

Exercice 6

Faire un fichier de test avec l'ex05 pour tester les fonctions et méthodes de son code.