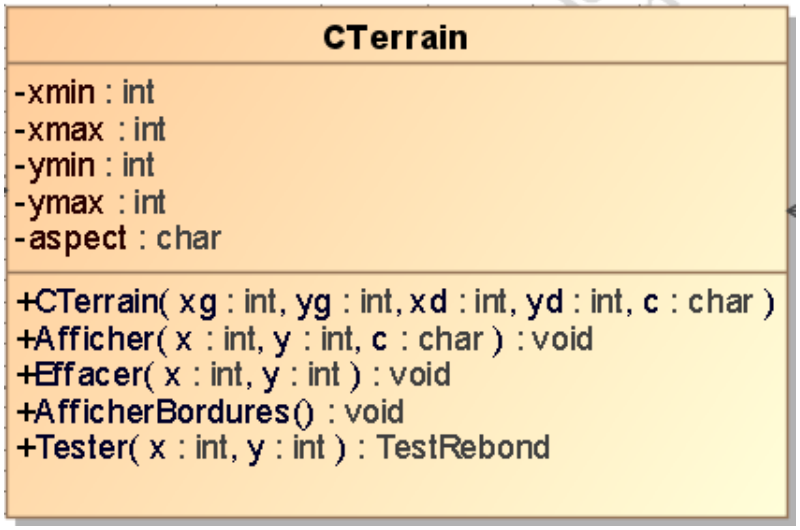
Sommaire

2 . Class Cterrain

2.1 Diagramme de classe de Cterrain



2.2 Rôle de CTerrain :

Cette Classe permet d'effectuer la gestion d'un terrain dans lequel peut évoluer des balles ou des raquettes. Le terrain sera un rectangle définit par les coordonnées de son angle supérieur droit et de son angle inferieur gauche. Un caractère permettra de représenter les lignes du terrain.
Ce terrain sera dessiné dans une console

2.3 Définition des attributs

xmin ,xmax, ymin et ymax (entiers) qui sont les limites du terrain.
aspect (char) qui fera les lignes du terrain.

2.4 Constructeur

```
CTerrain::CTerrain(int xg, int yg, int xd, int yd, char c)
{
	xmin = xg;
	xmax = xd;
	ymin = yg;
	ymax = yd;
	aspect = c;
}
```

2.5 Méthode Afficher()

2.5.1 Description :

La méthode Afficher permet d'afficher un caractère dans la zone du terrain délimité par les attributs xmin, ymin, xmax, ymax. Si les coordonnées sont hors zone, le caractère n'est pas affiché.

2.5.2 Paramètres :

Entrée : x est un Entier abscisse du caractère à afficher
y est un Entier ordonnée du caractère a afficher
c est un Caractère code ASCII du caractère à afficher
Sortie : rien
E/S : rien
Retourné : rien .

2.5.3 Test unitaire d'Afficher().

Le programme de test unitaire remplit l'écran d'un caractère '.' . Une fois rempli le programme essaye d'afficher 'O' sur tout l'écran à l'aide de la méthode Afficher (). Seule la zone définie par l'objet de type CTerrain doit être remplie de 'O'.
Différents terrains sont utilisés. Ils sont répertoriés dans le tableau suivant.

xmin	ymin	xmax	ymax	Caractere	Affichage attendu	Affichage obtenu
0	0	79	24	+	*La zone doit être couverte de X*	*La zone est couverte de X*
10	5	19	14	A	*La zone doit être couverte de A*	*La zone doit être couverte de A*
40	12	40	12	O	*Le terrain se résume en 1 seul caractère O*	*Le terrain se résume en 1 seul caractère O*
60	5	40	14	I	*Pas d'affichage (mauvaise définition de terrain)*	Pas d'affichage
40	14	60	5	T	*Pas d'affichage (mauvaise définition de terrain)*	Pas d'affichage

2.5.5 Pseudo code :

Debut
	Si les coordonnées sont dans la zone du terrain
		Afficher le caractère
	Fin SI
Fin

2.5.6 Prototype de la méthode en C++:

void Afficher(int x, int y, char c);

2.5.7 Code C++ de la méthode Afficher()

```
void CTerrain::Afficher(int x, int y, char c)
{
    if (x >= xmin && x <= xmax && y >= ymin && y <= ymax) {
        Console::SetCursorPosition(x, y);
        Console::CursorVisible = false;
        cout << c;
    }
}
```

2.5.8 Code du test unitaire

```
#ifdef TEST_CTERRAIN_AFFICHER
/* Test unitaire de Afficher:
Le test consiste à remplir l'écran de '.' puis d'appeler la methode afficher 'O' avec toutes les coordonnées possibles de l'écran
Seule la surface de la zone gérée par le terrain doit comporter des 'O'.
*/
#define NBLIGNES 80 // correspond à y
#define NBCOL 25 //correspond à x
void main()
{
    //création des terrains qui serviront au test
    CTerrain t1(0, 0, 79, 24, 'x');
    CTerrain t2(10, 5, 19, 14, '*');
    CTerrain t3(40, 12, 40, 12, '-');
    CTerrain t4(60, 5, 40, 14, '+');
    CTerrain t5(40, 14, 60, 5, '/');
```

```cpp
    //remplissage du terrain à l'aide d'un '.'
    for (int j = 0;j < NBCOL;j++)
    {
        for (int i = 0;i < NBLIGNES;i++)
        {
            cout << '.';
        }
        cout << endl;
    }
    _getch();
    // Test de la methode Afficher sur tout l'ecran
    for (int i = 0;i < NBLIGNES;i++) {
        for (int j = 0;j < NBCOL;j++) {
            t1.Afficher(i, j, '+');
        }
    }
    _getch();
    // Test de la methode Afficher sur tout l'ecran
    for (int i = 0;i < NBLIGNES;i++) {
        for (int j = 0;j < NBCOL;j++) {
            t2.Afficher(i, j, 'A');
            t3.Afficher(i, j, 'O');
            t4.Afficher(i, j, 'I');
            t5.Afficher(i, j, 'T');
        }
    }
    _getch();
}
#endif
```

2.5.9 Application du test unitaire.

D:\Snir1\Eric\Pingpong\Pingpong\Debug\Pingpong.exe

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++O+++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++AAAAAAAAAA++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

2.6 Méthode Effacer()

2.6.1 Description :

La méthode Effacer permet d'effacer un caractère dans la zone du terrain délimité par les attributs xmin, ymin, xmax, ymax. Si les coordonnées sont hors zone, le caractère n'est pas effacé.

2.6.2 Paramètres :

*Entrée : **x** est un Entier abscisse du caractère à afficher*
***y** est un Entier ordonnée du caractère a afficher*
Sortie : rien
E/S : rien
Retourné : rien

2.6.3 Visibilité :
Publique.

2.6.4 Test unitaire d'Effacer ().

Le programme de test unitaire remplit l'écran d'un caractère '.' . Une fois rempli le programme essaye d'effacer sur tout l'écran à l'aide de la méthode Effacer (). Seule la zone définie par l'objet de type CTerrain doit être vide.

xmin	ymin	xmax	ymax	Affichage attendu	Affichage obtenu
0	0	79	24	*Le terrain complet doit être effacé*	*Le terrain complet doit être effacé*
10	5	19	14	*La zone doit être effacée*	*La zone doit être effacée*
40	12	40	12	*Le terrain se résume en 1 seul caractère*	*Le terrain se résume en 1 seul caractère*
60	5	40	14	*Pas d'effacement (mauvaise définition de terrain)*	*Mauvaise définition*
40	14	60	5	*Pas d'effacement (mauvaise définition de terrain)*	*Mauvaise definition*

2.6.5 Pseudo code :

Debut
	Afficher(x,y,' ')
Fin

2.6.6 Prototype de la méthode en C++:

void Effacer(int x, int y);

2.6.7 Code C++ de la méthode Effacer()

void CTerrain::Effacer(int x, int y)
{
Afficher(x, y , ' ') ;
}

2.6.8 Code du test unitaire

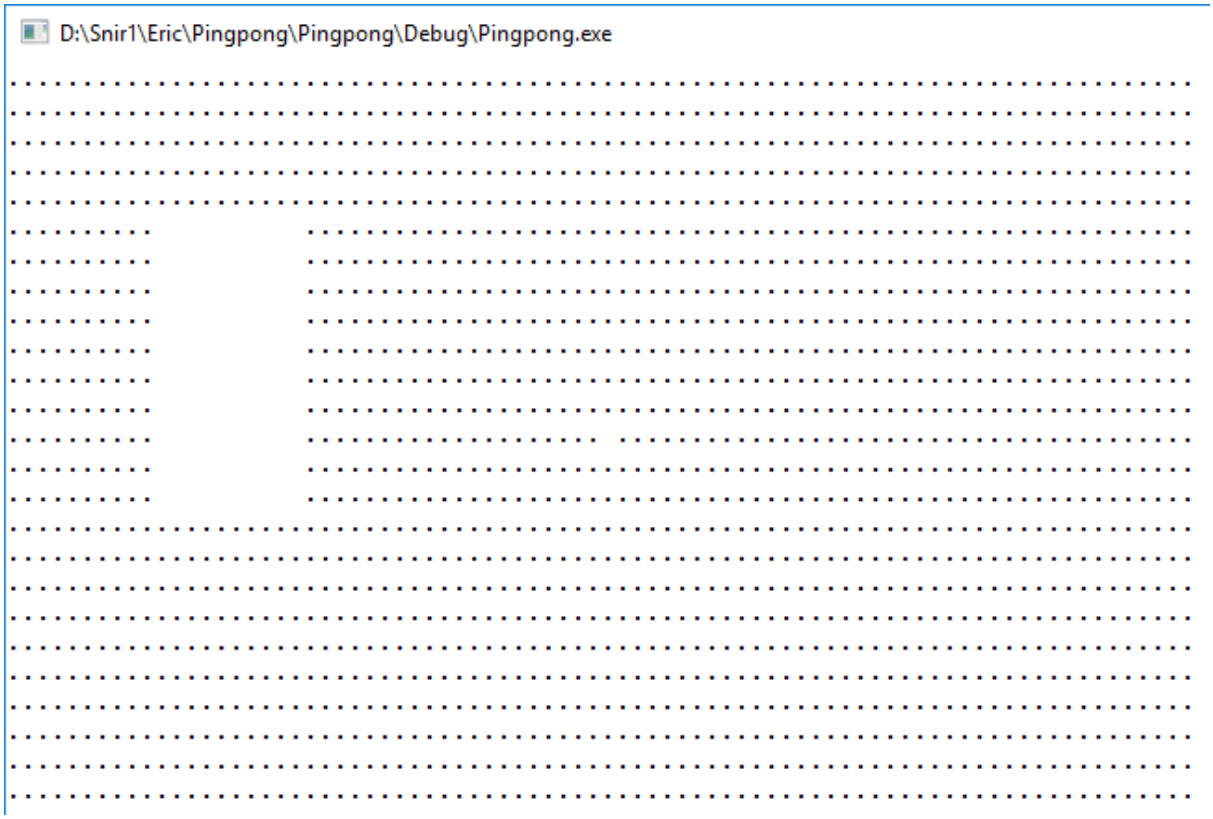
```
#ifdef TEST_CTERRAIN_EFFACER
/* Test unitaire de Effacer:
Le test remplit l'écran de '.' puis appelle la methode Effacer() avec toutes les coordonnées possibles de l'écran
Seule la surface de la zone gérée par le terrain doit être éffacée.
*/
#define NBLIGNES 80 // correspond à y
#define NBCOL 25 //correspond à x

void main()
{
    CTerrain t1(0, 0, 79, 24, 'x');
    CTerrain t2(10, 5, 19, 14, '*');
    CTerrain t3(40, 12, 40, 12, '-');
    CTerrain t4(60, 5, 40, 14, '+');
    CTerrain t5(40, 14, 60, 5, '/');
    //remplissage du terrain à l'aide d'un '.'
    for (int j = 0;j < NBCOL;j++)
    {
        for (int i = 0;i < NBLIGNES;i++)
        {
            cout << '.';
        }
        cout << endl;
    }
    _getch();
    // Test de la methode Effacer sur tout l'ecran
    for (int i = 0;i < NBLIGNES;i++) {
        for (int j = 0;j < NBCOL;j++) {
            t1.Effacer(i, j);
        }
    }
```

```
    //remplissage du terrain à l'aide d'un '.'
    Console::SetCursorPosition(0, 0);
    for (int j = 0;j < NBCOL;j++)
    {
        for (int i = 0;i < NBLIGNES;i++)
        {
            cout << '.';
        }
        cout << endl;
    }
    _getch();
    // Test de la methode Effacer sur tout l'ecran
    for (int i = 0;i < NBLIGNES;i++) {
        for (int j = 0;j < NBCOL;j++) {
            t2.Effacer(i, j);
            t3.Effacer(i, j);
            t4.Effacer(i, j);
            t5.Effacer(i, j);
        }
    }
    _getch();
}

#endif
```

2.6.9 Application du test unitaire



2.7 Méthode AfficherBordures()

2.7.1 Description :

La méthode AfficherBordures permet de dessiner sur l'écran les bords du terrain à l'aide d'un caractère dont le code ASCII est enregistré dans l'attribut aspect.

2.7.2 Paramètre :

Entrée : rien
Sortie : rien
E/S : rien
Retourné : rien

2.7.3 Visibilité :

publique

2.7.4 Test unitaire d'AfficherBordures () :

Le programme de test unitaire remplit l'écran d'un caractère '.' . Une fois rempli le programme essaye d'afficher les bords du terrain.
Différents terrains sont utilisés. Ils sont répertoriés dans le tableau suivant.

xmin	ymin	xmax	ymax	Caractere	Affichage attendu	Affichage obtenu
0	0	79	24	+	*La zone doit être couverte de +*	*La zone est couverte de +*
10	5	19	14	A	*La zone doit être couverte de A*	*La zone doit être couverte de A*
40	12	40	12	O	*Le terrain se résume en 1 seul caractère O*	*Le terrain se résume en 1 seul caractère O*
60	5	40	14	I	*Pas d'affichage (mauvaise définition de terrain)*	Pas d'affichage
40	14	60	5	T	*Pas d'affichage (mauvaise définition de terrain)*	Pas d'affichage

2.7.6 Prototype de la méthode en C++:

```
void CTerrain::AfficherBordures()
```

2.7.7 Code C++ de la méthode AfficherBordures()

```
void CTerrain::AfficherBordures()
{
	int i;
	for (i = xmin; i != xmax; i++)
	{
		Console::SetCursorPosition(i, ymin);
		cout << aspect;
	}

	for (i=ymin; i != ymax; i++)
	{
		Console::SetCursorPosition(xmin,i);
		cout << aspect;
	}

	Console::SetCursorPosition(xmax,ymax ); // Placement pour le deuxieme point
nt
	for (i=xmax ; i != xmin-1 ; i--)
	{
		Console::SetCursorPosition(i,ymax );
		cout << aspect;
	}


	for (i=ymax; i !=ymin-1 ; i--)
	{
		Console::SetCursorPosition(xmax,i );
		cout << aspect;
	}

	Console::SetCursorPosition(0, ymax + 1);
}
```

2.7.8 Code du test unitaire

```
#ifdef TEST_UNITAIRE_TERRAIN
void main()
{
	CTerrain t(0, 0, 60, 25, '°');
	t.AfficherBordures();
}
#endif
```

2.8 Méthode Tester()

2.8.1 Description :

La méthode Tester permet de tester les coordonnées passées en argument par rapport aux limites du terrain. Le résultat de ce test sera retourné sous la forme d'un type énuméré TestRebond

2.8.2 Paramètre :

*Entrée : **x** est un Entier abscisse du caractère à afficher*
***y** est un Entier ordonnée du caractère a afficher*
Sortie : rien
E/S : rien
Retourné : Résultat du test de type TestRebond

2.8.3 Visibilité :
publique

2.8.4 Test Unitaire de Tester():

Dans ce Test, on balaye toutes les combinaisons possibles de l'écran, puis on affiche la valeur retournée par la méthode Tester() conformément à TestRebond:
TestRebond { Rien, Haut, Droite, Bas, Gauche, HautGauche,HautDroite , BasDroite, BasGauche };
0 1 2 3 4 5 6 7 8
On doit obtenir un motif de ce type

00000000000000000000
*00005111111111116**0000***
*00004000000000002**0000***
*00004000000000002**0000***
*00004000000000002**0000***
*00004000000000002**0000***
*00004000000000002**0000***
*00008333333333337**0000***
00000000000000000000

2.8.6 Prototype de la méthode en C++:

```
TestRebond CTerrain::Tester(int x, int y)
```

2.8.7 Code C++ de la méthode Tester()

```
TestRebond CTerrain::Tester(int x, int y)
{
	if (x == xmin && y == ymin)
		return(HautGauche);

	else if (x == xmin && y == ymax)
		return(BasGauche);

	else if (x == xmax && y == ymin)
		return(HautDroite);

	else if (x == xmax && y == ymax)
		return(BasDroite);

	else if (xmin < x && x < xmax && y == ymin)
		return(Haut);
```

```
        else if (ymin < y && y < ymax && x == xmin)
                return(Gauche);

        else if (xmin < x && x < xmax && y == ymax)
                return(Bas);

        else if (ymin < y && y < ymax && x == xmax)
                return(Droite);
        else
                return(Rien);
}
```

2.8.8 Code du test unitaire

```
#ifdef TEST_CTERRAIN_TESTER
/*Test Unitaire de Tester
Dans ce Test, on balaye toutes les combinaisons possible de l'ecran, puis on affiche la valeur retournée par la fonction
conformement à TestRebond:
TestRebond { Rien, Haut, Droite, Bas, Gauche, HautGauche,HautDroite , BasDroite, BasGauche };
              0   1     2     3     4       5         6           7         8
On doit obtenir un motif de ce type
000000000000000000000
000051111111111160000
000040000000000020000
000040000000000020000
000040000000000020000
000040000000000020000
000083333333333370000
000000000000000000000
*/
void main()
{
    CTerrain t1(10, 5, 70, 25, '*');
    for (int i = 0;i < 30;i++)
    {
        for (int j = 0;j < 100;j++)
        {
            cout << '.';
        }
        cout << endl;
    }
    _getch();
    t1.AfficherBordures();
    _getch();
    cout << endl;

    char c;
    for (int i = 0;i < 100;i++) {
        for (int j = 0;j < 30;j++) {
            c = '0' + t1.Tester(i, j);
            Console::SetCursorPosition(i, j);
            Console::CursorVisible = false;
            cout << c;
        };
        cout << endl;
    }
    _getch();
}
#endif
```

2.8.9 Application du test unitaire.

```
#ifdef TEST_TESTREBOND
void main()
{
	CTerrain t(0, 0, 25, 10, '°');
	for (int i = 0; i != 79; i++)
	{
		for (int j = 0; j != 19; j++)
		{
			Console::SetCursorPosition(i, j);
			cout << t.Tester(i, j);
		}
	}//remplir la console de point -- ecran 80*25
}
#endif
```

3 Classe CBalle

3.1 Diagramme de Classe de Cballe

3.3 Définition des Attributs :

Entiers : x, y, vitesse
Char : aspect
Tsens sens

3.4 Constructeur paramétré :

```
CBalle(int x, int y, Tsens sens, char aspect, int vitesse);
```

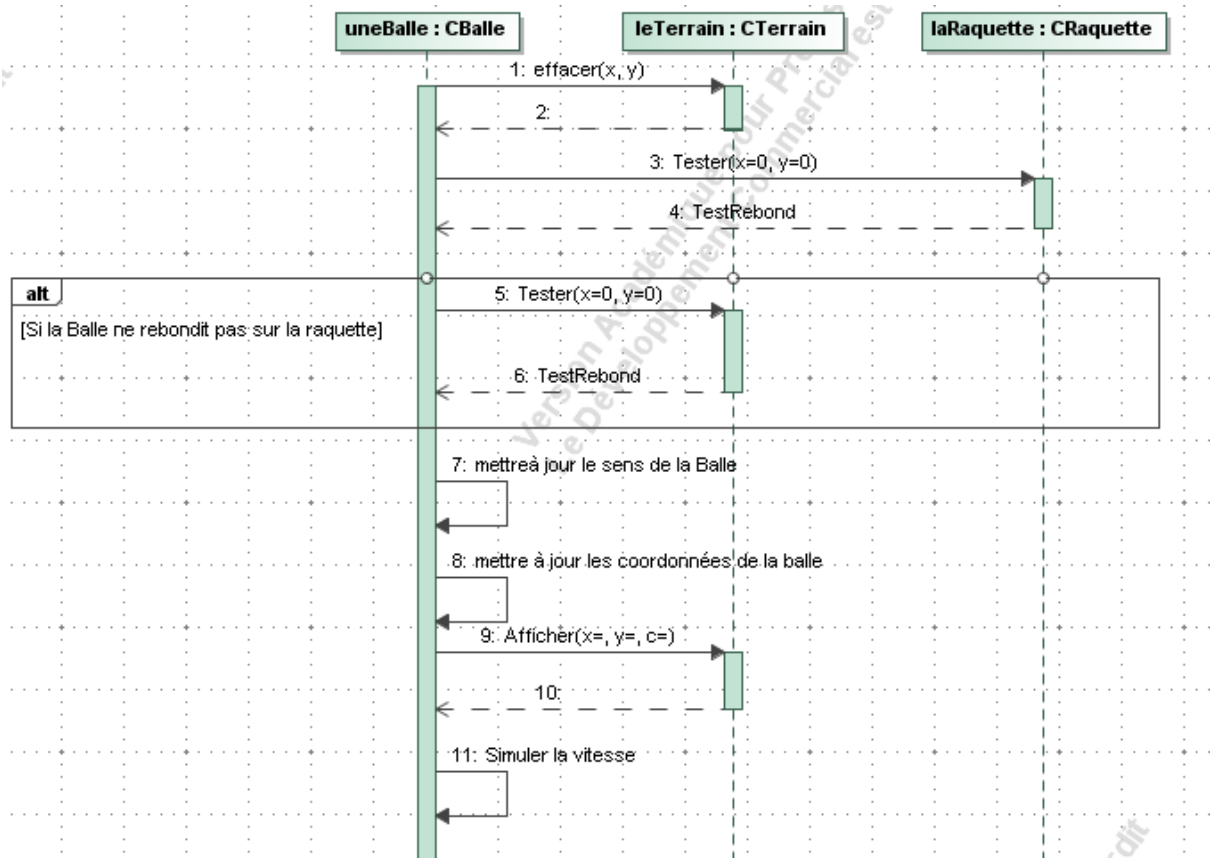
méthode Deplacer()

3.5.1 Description :

La méthode déplacer permet de déplacer la balle à l'intérieur du terrain. Cette méthode met à jour le sens de la balle en fonction de ses coordonnées et du résultat de l'appel de la méthode Tester() du Terrain.

HG	*HD*	*BG*	*BD*
Rien	*HG*	*HD*	*BG*	*BD*
Haut	*BG*	*BD*	*BG*	*BD*
Bas	*HG*	*HD*	*HG*	*HD*
Droite	*HG*	*HG*	*BG*	*BG*
Gauche	*HD*	*HD*	*BD*	*BD*
HautGauche	*BD*	*BD*	*BD*	*BD*
HautDroite	*BG*	*BG*	*BG*	*BG*
BasGauche	*HD*	*HD*	*HD*	*HD*
BasDroite	*HG*	*HG*	*HG*	*HG*

3.5.2 Diagramme de Séquence



3.5.5 Test unitaire de Tester () :

Pour tester la position, il faut pouvoir positionner une balle dans une position déterminée avec un sens défini, puis faire évoluer la balle. Elle doit répondre à toutes les conditions énumérées dans le tableau

Conditions sur le terrain : taille de 11 par 6
Xg =1 yg=1 xd= 11 yd=6 aspect ''*

Conditions initiales de la Balle

X	Y	Sens	Bords touchés estimés	Bords réelement touchés
4	3	BD	*Bas, Droite, Haut, Gauche*	Bas, Droite, Haut, Gauche
4	3	HD	*Haut, BasDroite, BasGauche*	Haut, BasDroite, BasGauche
4	4	HG	*HautGauche, Bas, HautDroite*	HautGauche, Bas, HautDroite

3.5.9 Code du test unitaire

```
#ifdef TEST_CBALLE_AVANCER
/*Pour tester la position, il faut pouvoir positionner une balle dans une position déterminée avec un sens défini,
puis faire évoluer la balle.
Elle doit répondre à toutes les conditions énumérées dans le tableau

Conditions sur le terrain : taille de 11 par 6
xg = 1    yg = 1     xd = 11    yd = 6    aspect '*'

Conditions initiales de la Balle
X   Y   Sens      Bords Touchés estimés         Bords réellement touchés.
4   3    BD     Bas, Droite, Haut, Gauche
4   3    HD     Haut, BasDroite, BasGauche
4   4    HG     HautGauche, Bas, HautDroite
*/
void main()
{
    CTerrain t1(1, 1, 11, 6, '*');
    CBalle b1(5, 3, BD, 'O', 50);
    b1.setTerrain(&t1);
    CTerrain t2(1+20, 1, 11+20, 6, '*');
    CBalle b2(4+20, 3, HD, 'O', 50);
    b2.setTerrain(&t2);
    CTerrain t3(1+40, 1, 11+40, 6, '*');
    CBalle b3(4+40, 4, HG, 'O', 50);
    b3.setTerrain(&t3);
    do
    {
        b1.Avancer();
        b2.Avancer();
        b3.Avancer();
    }while (_kbhit() == 0);
}
#endif
```