Premiers programmes en C# "étape par étape" avec une carte BrainPad v1

[Mise à jour le : 11/07/2018]

1. Préambule

Pour mener à bien ce tutoriel vous devez disposer d'une carte **BrainPad V1**. Les outils logiciels nécessaires à sa programmation doivent être installés sur le PC. Le firmware de la carte doit être à jour. Si ce n'est pas le cas : suivez le « **Guide d'installation des logiciels** » ici.

Les vidéos du cours sur les fondamentaux du langage C#, accessibles sur le site MVA sont un excellent préalable ou un complément à ce tutoriel.

On trouvera également des tutoriels en anglais sur le site de la société GHI Electronics.

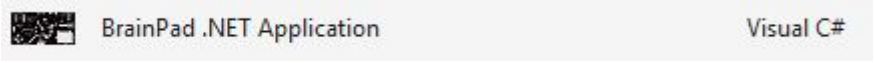
2. Premier programme : Blink

Cahier des charges

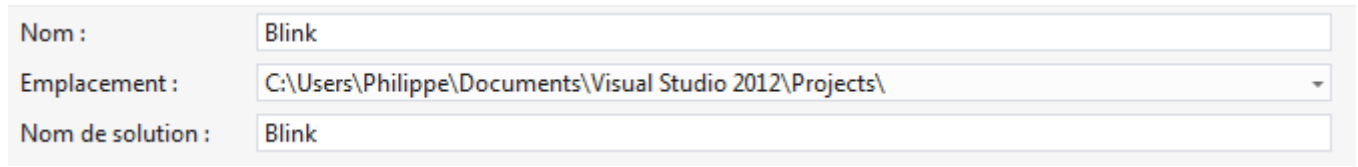
Faire clignoter la LED RVB de la carte BrainPad et afficher le texte "Hello, world!" sur l'écran graphique.

2a. Créer un projet avec le template BrainPad .NET Application

- Ouvrez l'**IDE Microsoft Visual Studio 2015** en cliquant sur l'icône suivante : puis sélectionnez : **Fichier (File) → Nouveau projet (New Project)** ou **[Ctrl+Maj+N]**,
- Dans la boîte de dialogue "Nouveau projet" (New Project) sélectionnez: **Modèles(Templates) → Visual C# → Micro Framework puis BrainPad .NET Application**.

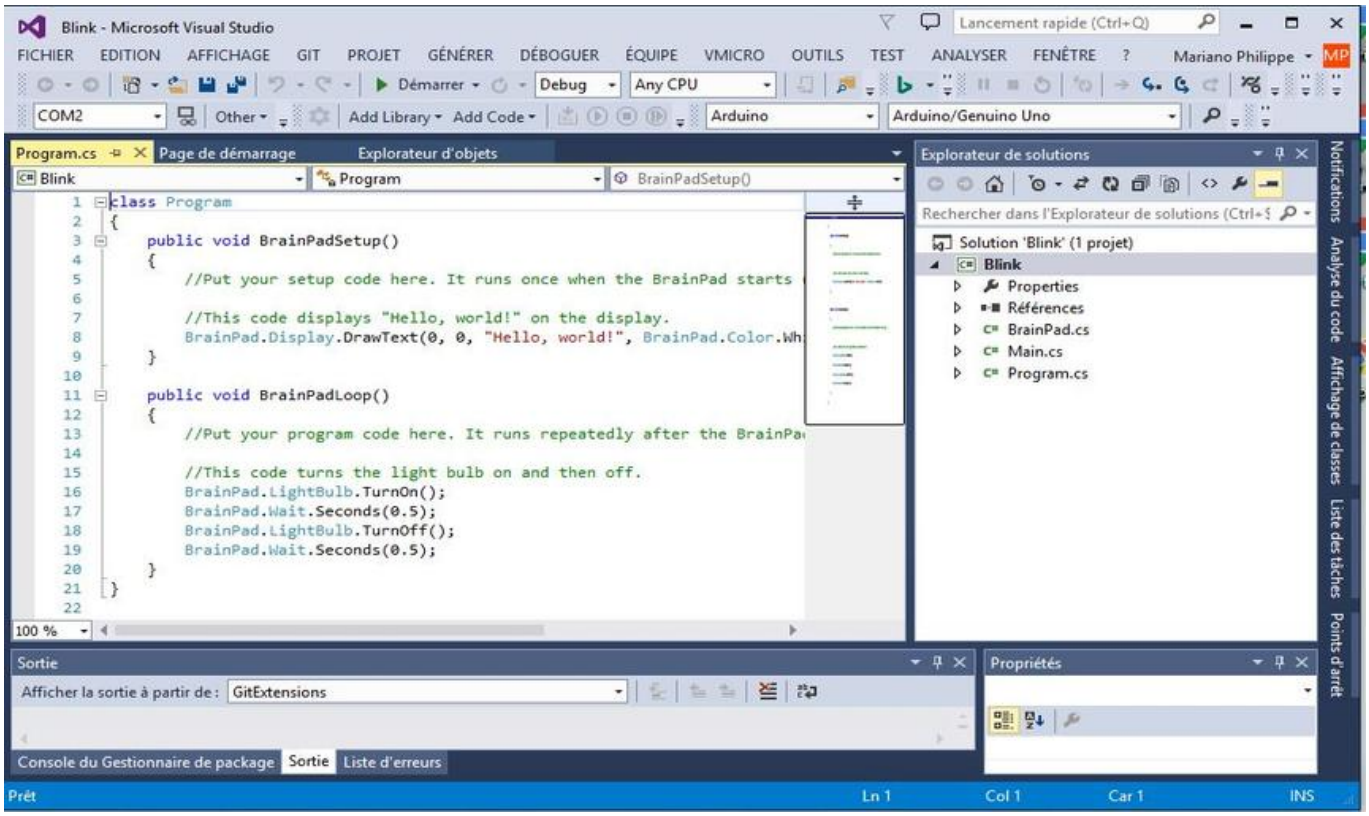


- Donnez le nom "**Blink**" à l'application puis cliquez sur **Ok**.



Remarque: L'emplacement (Location) identifié ci-dessous peut changer en fonction de la version du logiciel et de l'arborescence des répertoires du PC.

L'IDE est alors configuré comme sur la copie d'écran ci-dessous (ou un équivalent selon sa version) :



Le projet ***Blink*** est contenu dans la solution ***Blink***. Vous allez écrire le code dans le fichier **Program.cs**.

2b. Organisation d'un programme créé avec le template "BrainPad .NET Application"

Comme pour un projet **Arduino**, un projet utilisant le **template BrainPad .NET Application** contient deux parties :

- Le **Setup**, nommé ici **BrainPadSetup**, contient le code ne devant s'exécuter qu'**une seule fois** au démarrage de la carte.
- La boucle **Loop**, nommée ici **BrainPadLoop**, contient le code devant s'exécuter **indéfiniment** après le démarrage de la carte.

2c. Téléchargement et exécution du programme Blink

La carte étant connectée au PC avec le câble USB, le programme est compilé puis transféré en cliquant sur le bouton Démarrer ou sur **F5**. La LED RVB **Light Bulb** clignote et le texte **“Hello, world!”** s'affiche sur l'écran comme ci-dessous.

Étude du fonctionnement de l'exemple

L'application démarre en exécutant une seule fois le code placé entre les accolades de la **méthode BrainPadSetup()**.

```
//This code displays "Hello, world!" on the display.
BrainPad.Display.DrawText(0, 0, "Hello, world!", BrainPad.Color.White);
```

La méthode **DrawText()** de l'objet Display placé sur la carte BrainPad affiche en blanc le texte “Hello, world” à la position 0,0 de l'écran.

Puis, le code placé entre les accolades de la méthode BrainPadLoop() s'exécute indéfiniment.

```
//This code turns the light bulb on and then off.
BrainPad.LightBulb.TurnOn();
BrainPad.Wait.Seconds(0.5);
BrainPad.LightBulb.TurnOff();
BrainPad.Wait.Seconds(0.5);
```

La Led Bulb est éclairée puis éteinte successivement pendant 0,5s.

Exercice 1 : Modifiez le programme pour que la LED émette un flash de 100ms toutes les 1s et que votre nom (prémom) apparaissent en jaune sur l'afficheur.

Remarque : l'afficheur n'accepte pas les caractères accentués.

---

3. L'objet BrainPad

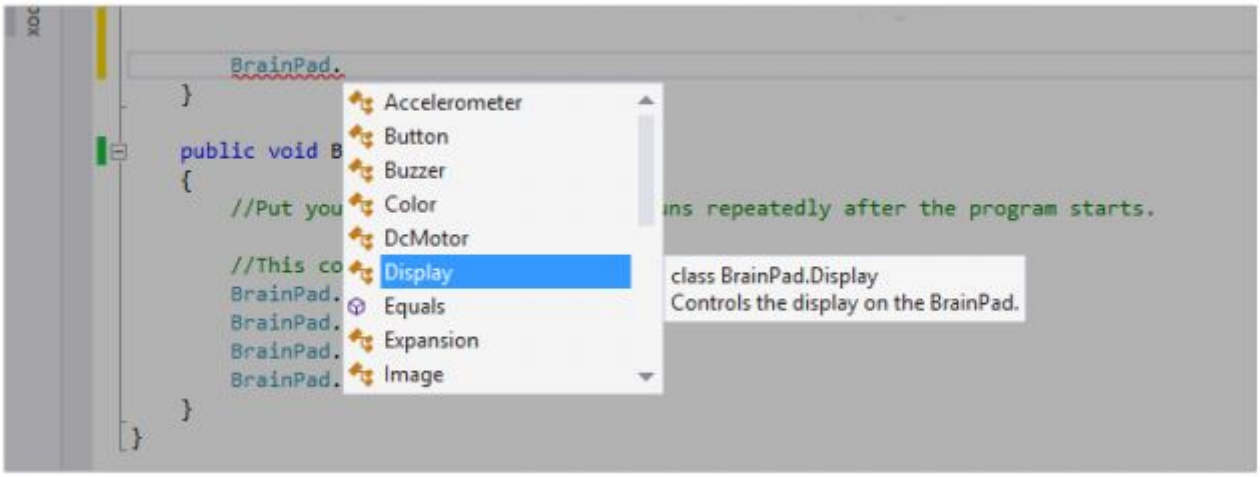
Dans le monde “réel”, les voitures, les tables, les téléviseurs sont des **objets**. Un objet du monde réel peut être décrit par son **état** (la voiture est une Citroën bleue) et par son comportement (la voiture avance).

Dans le monde “virtuel” tout peut être assimilé à un objet : même une personne ! Les concepts d'**objet**, d'**état** et de **comportement** son repris dans les langages de programmation dit “Objet”

comme le **C#** utilisé pour programmer la carte BrainPad.

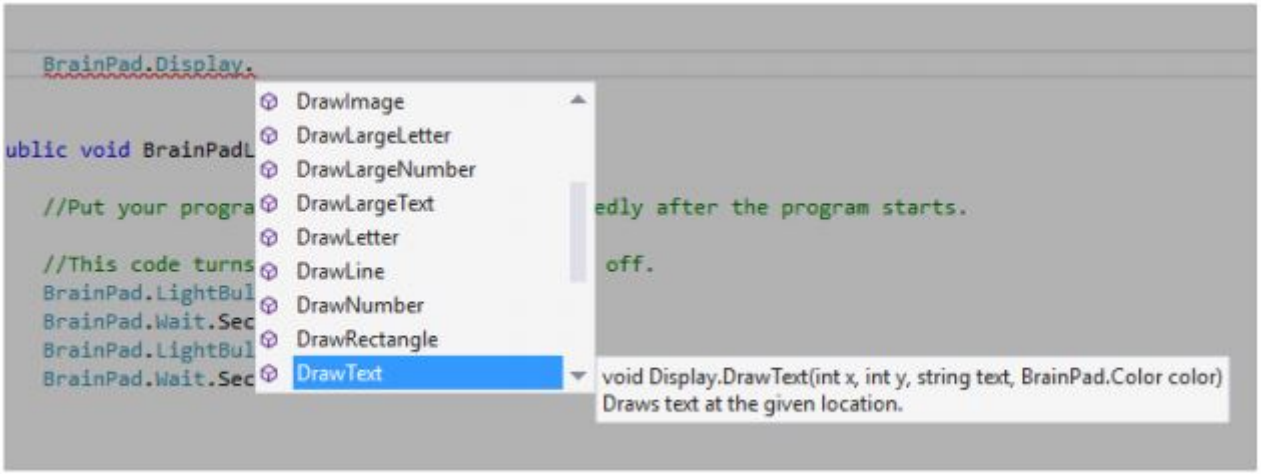
La carte BrainPad réelle est contrôlée par l'objet BrainPad virtuel. Cet objet est décrit par du code développé par la société GHI Electronics. Il est contenu dans le fichier Brainpad.cs.

L'accès aux fonctionnalités de la carte BrainPad nécessite d'entrer le mot **BrainPad chaque fois qu'on y fait référence**. En faisant suivre ce mot par un **point** on obtient la liste des objets qui “composent” la carte tels que : **Display**, **Buzzer**, **Accelerometer** etc.



Tous ces “sous-objets” sont accessibles à partir de leur nom. Il suffit d'en sélectionner un pour accéder à la liste de ses fonctionnalités.

Par exemple, en sélectionnant Display on obtient la liste ci-dessous :



L'IDE Visual Studio simplifie la programmation en proposant automatiquement la liste des fonctionnalités d'un objet. Vous allez utiliser cette aide pour écrire le prochain programme.

4. Second programme : TrafficLight

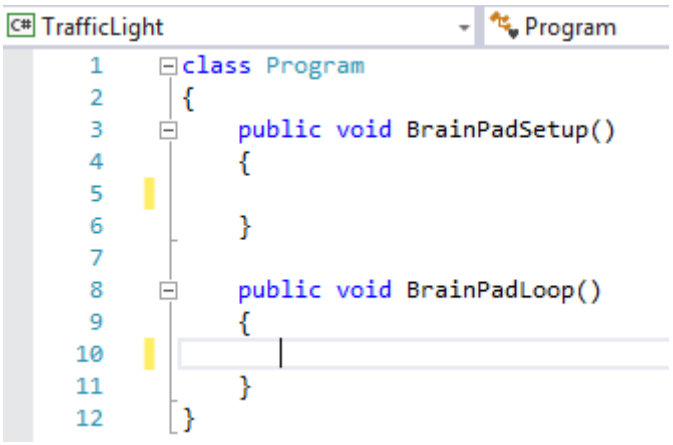
Cahier des charges : Faire clignoter la Led verte de la zone TrafficLight.

Remarque : Vous allez écrire une partie du code et simuler un problème de programmation. Sa

résolution vous permettra de découvrir le mode de fonctionnement pas à pas.

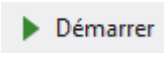
Etape 1. **Créez** un nouveau projet BrainPad et nommez-le TrafficLight

Etape 2. **Supprimez** le code produit automatiquement. Votre programme doit ressembler à la copie d'écran ci-dessous.



Etape 3. **Copiez** l'extrait de code ci-dessous entre les accolades de **BrainPadSetup()**.

```
BrainPad.TrafficLight.TurnGreenLightOff();
BrainPad.TrafficLight.TurnGreenLightOn();
BrainPad.TrafficLight.TurnGreenLightOff();
BrainPad.TrafficLight.TurnGreenLightOn();
```

Etape 4. **Téléchargez** et exécutez le code en cliquant sur 

ou en appuyant sur **F5**.

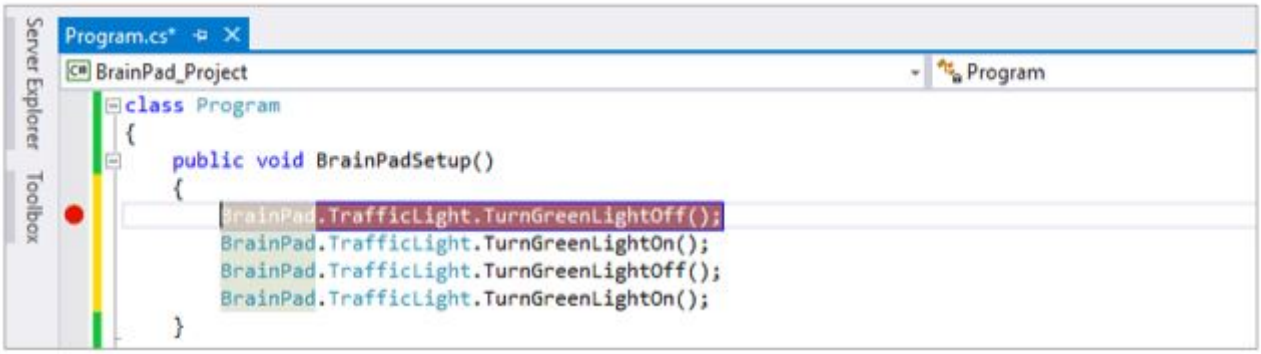
Problème : la Led reste constamment éclairée au lieu de clignoter. Ceci vient du fait que le programme fonctionne très rapidement. Il est nécessaire de le ralentir.

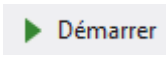
Pour visualiser ce que fait le programme lorsqu'il s'exécute lentement, nous allons le faire fonctionner en mode pas-à-pas.

Etape 5 : **Test** du programme en mode pas-à-pas Ce mode de fonctionnement permet de mettre un programme “au point” (déboguer). Dans ce mode, vous pouvez l'arrêter, le redémarrer ou le mettre en pause.

1. **Cliquez** sur l'icône “Arrêter”.

2. **Placez** un **point d’arrêt** en cliquant dans la marge à gauche de la première ligne comme ci-dessous.



3. **Relancez** le programme (touche F5 ou 

. Celui-ci s'arrête. Vous pouvez exécuter le code ligne par ligne (**mode pas-à-pas**) en appuyant sur la touche **F10**. (Un appui exécute une ligne) **Lorsque vous atteignez l'accolade fermante : arrêtez le programme**.

Exercice 2 : Modifiez le programme pour que la Led jaune clignote 2 fois par seconde. Par précaution, on éteindra les trois Leds au démarrage de la carte.

Exercice 3 : Créez un programme de simulation de feu de carrefour à partir de l'algorithme ci-dessous. (Créez un nouveau projet BrainPad et nommez-le Carrefour).

1. Turn the red light off.
2. Turn the yellow light off.
3. Turn the green light on.
4. Wait 5 seconds.
5. Turn the green light off.
6. Turn the yellow light on.
7. Wait 1 seconds.
8. Turn the yellow light off.
9. Turn the red light on.
10. Wait 5 seconds.

---

5. Button : un troisième programme pour se familiariser avec la structure si... alors ... sinon... fin si

Algorithme

```
si (condition) alors Action1 sinon Action2 fin si
```

L'action 1 est exécutée si la *condition* est vraie. L'action 2 est exécutée si la *condition* est fausse. (L'action 2 peut être omise !).

Cahier des charges du programme à réaliser Si le boutton Down est appuyé la led verte s'éclaire.

---

1. Créez un nouveau projet BrainPad et nommez-le Button.

2. Copiez le code ci-dessous et exécutez-le :

```
public void BrainPadSetup()
{
    BrainPad.TrafficLight.TurnGreenLightOff();
}

public void BrainPadLoop()
{
    if (BrainPad.Button.IsDownPressed())
    {
        BrainPad.TrafficLight.TurnGreenLightOn();
    }
}
```

Il semble qu'il y ait un problème : la Led ne s'éteint pas lorsqu'on relâche le bouton. Pour cela : il faut lui dire !

3. Modifiez votre programme comme ci-dessous.

```
public void BrainPadSetup()
{
    BrainPad.TrafficLight.TurnGreenLightOff();
}

public void BrainPadLoop()
{
    if (BrainPad.Button.IsDownPressed())
    {
        BrainPad.TrafficLight.TurnGreenLightOn();
    }
    else
    {
        BrainPad.TrafficLight.TurnGreenLightOff();
    }
}
```

6. Synthèse : "Jouer une partition avec le buzzer"

Les méthodes décrites dans le tableau ci-dessous permettent de contrôler le buzzer.

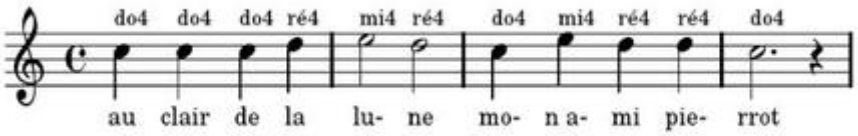
	Synthaxe	Description
S	*void* **PlayFrequency**(*int* Frequency)	Joue une fréquence.
S	*void* **Stop**()	Coupe le son.

Remarque : void signifie que la méthode ne renvoie pas d'informations. int signifie que la valeur à placer entre les parenthèses doit être un entier

Exemple : BrainPad.Buzzer.PlayFrequency(523);

Vous allez utiliser ces méthodes pour écrire le programme **AuClairDeLaLune** dont la partition est donnée ci-dessous (ou une autre de votre choix) .

Partition



Les fréquences correspondant aux notes et le rythme à attribuer à une note sont précisés dans les ressources ci-dessous.

Ressources Fréquence des notes : lien Le rythme des notes de musique (ronde, blanche, noire...): lien

---

Exercice 4

Version 1a : La carte joue seule la partition (une fois). (Nom du projet : **ClairLuneV1a**)

Version 1b : La carte joue seule la partition (une fois) et le texte de la chanson s'affiche sur l'écran. (Nom du projet : **ClairLuneV1b**) [Video]

Remarques : Une noire dure 0,5s

---

pour aller plus loin...

Version 2 : L'utilisateur joue la partition avec le clavier. (Nom du projet **ClairLuneV2**)

Indications : Utiliser des évènements pour gérer les boutons-poussoir (**voir prof**)

Events

Many modules generate useful events. Type +=<tab><tab> to add a handler to an event, e.g.: button.ButtonPressed +=<tab><tab>

Sources

Les sources des exemples et des exercices (compilés avec **Visual Studio 2015 Community**) sont téléchargeables ici.

---

7. Les classes de la bibliothèque Brainpad

Accessibles à partir de ce lien