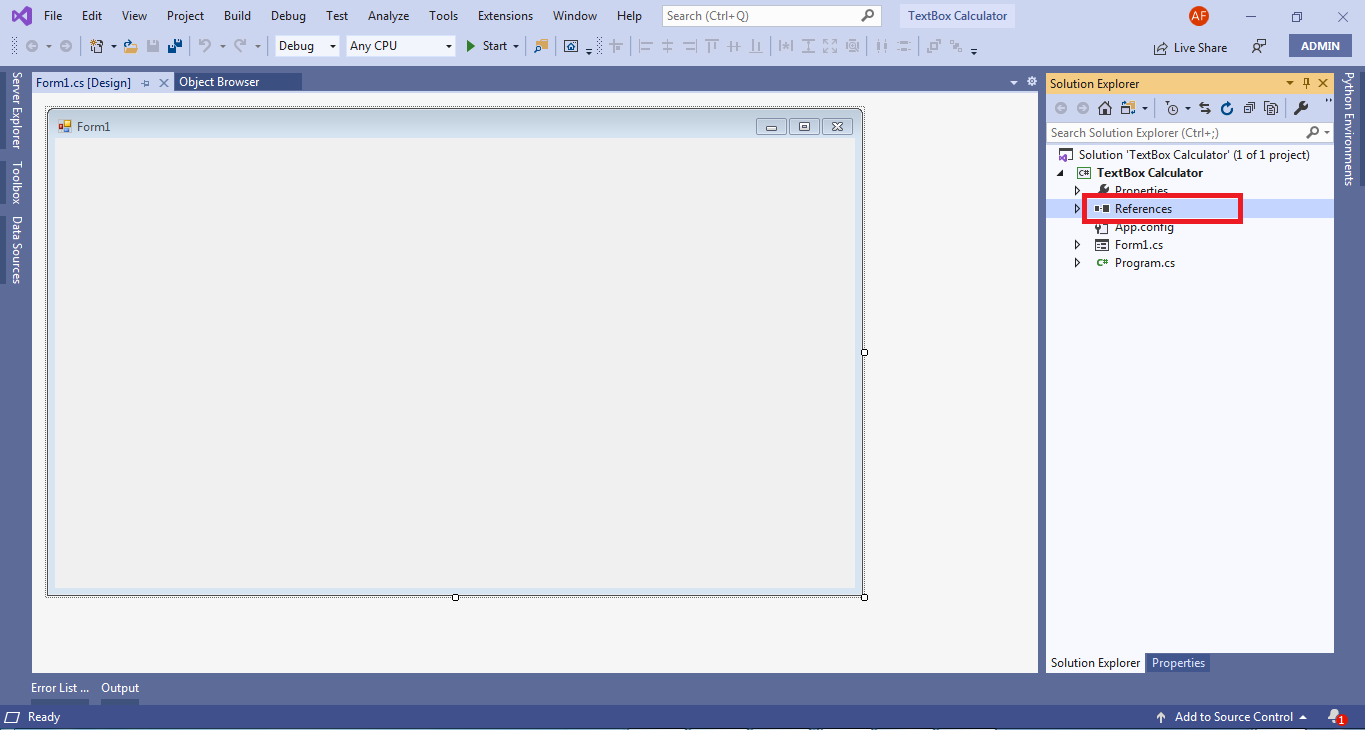
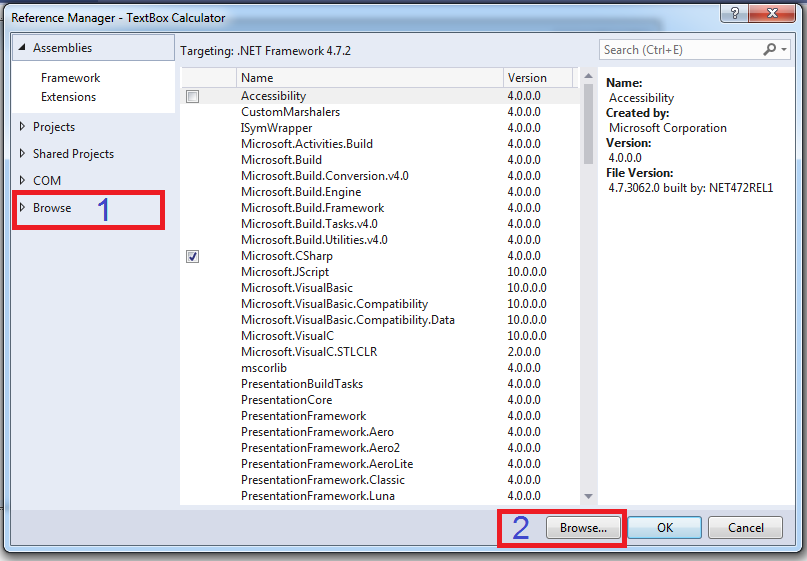
# معلومات أولية عن المكتبة:

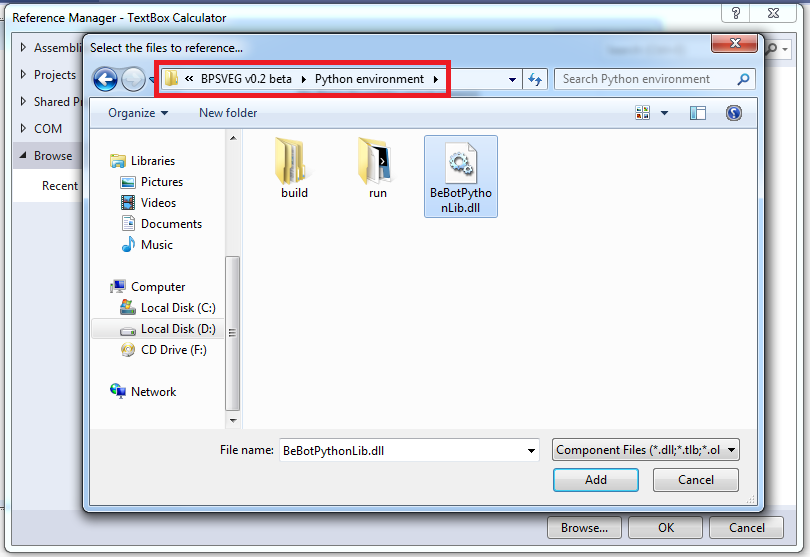
هي مكتبة تم بناءها من قبل Ahmad Fijr و فريق BeBot، مكتوبة بلغة سي شارب وهي تمثل API للتحكم والوصول لبيئة بايثون المصغرة المنشئة بواسطة برنامج BPSVEG من داخل برنامج C#.

# كيف تضيف المكتبة لمشروعك:

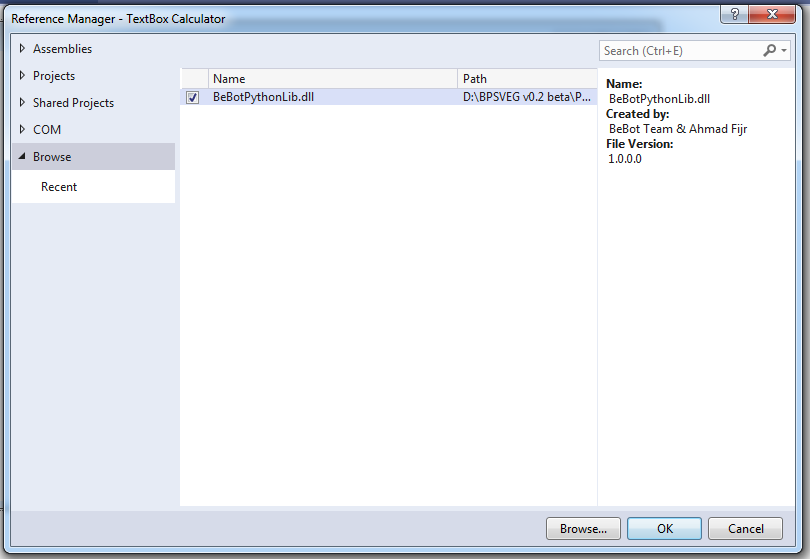
بعد إنشاء المشروع، نضغط بالزر الأيمن للفأرة على References ثم نختار Add References



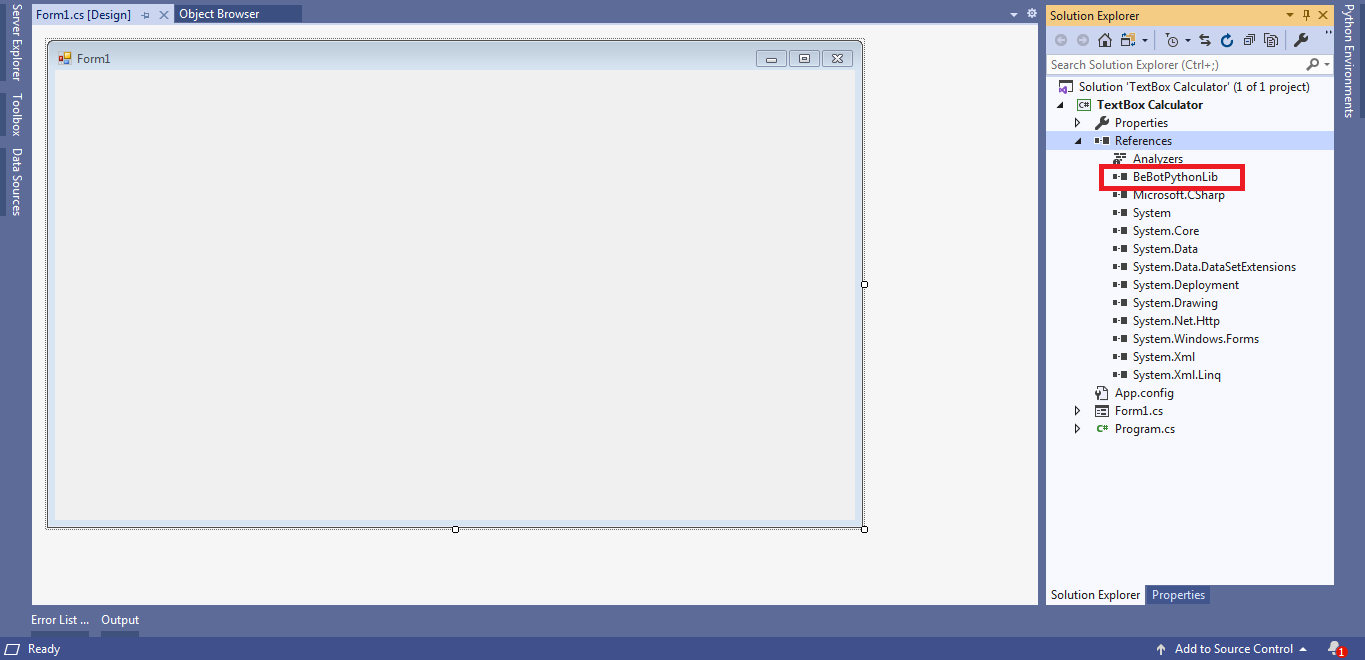
بعدها ستظهر هذه النافذة، نضغط على Browse كما في الصورة



نذهب إلى مسار البيئة المصغرة التي أنشأناها ونختار ملف **BeBotPythonLib.dll**



نضغط ok لإضافتها



تمت إضافتها بنجاح

# كيفية استخدام المكتبة:

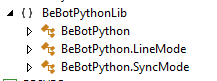
## استدعاء المكتبة:

using BeBotPythonLib;

استدعاء المكتبة للتمكن من استخدامها

## أنواع الدوال بعد الاستدعاء:

يوجد 3 دوال يمكنك استخدامها



## دالة BeBotPython:

تحتوي على بعض الدوال المفيدة في الدمج ودوالها تقابل دوال مكتبة ToolCS لبايثون

طريقة التعريف:

اختيار أي اسم

BeBotPython beBotPython = new BeBotPython();

الدوال التي تقع تحتها:

**Delete\_File(string)**

beBotPython.Delete\_File("text.txt");

تستخدم لحذف ملف وفي داخله تكتب أسم الملف

**Get\_File(string)**

beBotPython.Get\_File("text.txt"); string TEXT =

تستخدم لقراءة ملف وفي داخله تكتب اسم الملف ويقوم بإرجاع محتوى الملف

**Set\_File(string, string)**

beBotPython.Set\_File("name.txt", "I Love BeBot Python");

تستخدم لكاتبة ملف وفي داخله تكتب اسم الملف ثم محتواه

**Get\_OutputPy();**

string TEXT = beBotPython.Get\_OutputPy();

تستخدم لقراءة ملف نصي اسمه “OutputPy.txt” وإرجاع محتوى النص

**Set\_InputToPy(string)**

beBotPython.Set\_InputToPy("say Hello Now");

تستخدم لكتابة ملف نصي اسمه “InputToPy.txt” وفي داخله تكتب محتواه

## دالةBeBotPython.LineMode :

دالة لاستخدام بايثون في الوضع الخطي، الوضع الخطي بإمكانك استخدامه عندما تريد أن تتم العمليات بشكل الخطي (أي تفتح بايثون تنفذ المهمة ثم تغلق) تكون عادة أسهل في الاستخدام، وعندما تريد الربط بين الواجهة الرسومية المكتوبة بلغة بايثون مع الواجهة المكتوبة بلغة سي شارب، وعندما لا تريد ربط بايثون ببرنامج سي شارب أي بقاء عمل بايثون حتى بعد إغلاق برنامج سي شارب.

طريقة التعريف:

اختار أي اسم

BeBotPython.LineMode pyL = new BeBotPython.LineMode();

الدوال التي تقع تحتها:

**Py\_Arguments**

pyL.Py\_Arguments = "argv1 'ar gv 2' argv3 ";

تستخدم لإرسال حجج لبايثون، وكأنك ترسل الحجج بهذه الطريقة على موجه الأوامر

Example.py argv1 'ar gv 2' argv3

وتكتب قبل دالة **Py\_RunPython ، لن تعمل إذا كتبت بعدها – أي تكتب قبل تشغيل بايثون، وقيمة افتراضية لها لا شيء.**

**Py\_WindowType**

pyL.Py\_WindowType= "Normal";

**تستخدم لتحديد نوع نافذة بايثون، سوء تريدها ظاهرة أو مخفية، القيم الخاصة بها (**Hidden – Normal – Minimized – Maximized) **، تكتب قبل** دالة **Py\_RunPython ، لن تعمل إذا كتبت بعدها، وقيمة الافتراضية لها هي** Hidden**.**

**PY\_WantTo\_UseWaitForExit**

pyL. PY\_WantTo\_UseWaitForExit = true;

بشكل عام في الوضع الخطي عند كتابة، **Py\_RunPython** **تقوم بتشغيل بايثون والانتظار حتى ينتهي ثم تنتقل للسطر الذي بعده ولكن عند جعل قيمتها true يعني انه سيشكل خيط جديد (أي إنه لن ينتظر بايثون حتى ينتهي وينفذ الأسطر التي بعده بالتوازي مع عمل بايثون) و**ستتمكن من استخدام دوال الإغلاق من (**Py\_WaitForExit** - **Py\_Kill** - **Py\_CloseMainWindow - Py\_Close**) ، وتكون قيمتها الافتراضية false.

**Py\_RunPython**

pyL.Py\_RunPython(@"python\run.exe");

لتشغيل بايثون وفي داخلها تكتب مسار البيئة المصغرة

**Py\_WaitForExit**

pyL.Py\_WaitForExit();

**تستخدم هذه الدالة حصرا بعد Py\_RunPython** وهي **لتحديد المكان الذي ننتظر به بايثون حتى ينتهي، لكن إذا كان كود بايثون لا ينتهي يعني أن البرنامج سيتجمد ولن تتمكن من إغلاقه إلا من خلال task manger.**

**Py\_Kill**

pyL.Py\_Kill();

**تستخدم لقتل وإنهاء مهمة بايثون حالا.**

**Py\_Close**

pyL.Py\_Close();

**تستخدم لإغلاق نافذة بايثون ممكن أن تكون نافذة طرفية.**

**Py\_CloseMainWindow**

pyL. Py\_CloseMainWindow();

تستخدم لإغلاق النافذة الرئيسية لبايثون، بإضافة لجميع النوافذ الطرفية.

ملاحظة دائما **دوال الإغلاق تستخدم بعد Py\_RunPython**

## دالة BeBotPython.SyncMode:

دالة لاستخدام بايثون في وضع المزامنة الآنية بين اللغتين، فيها تعقب لأخطاء بايثون ، يتم تبادل البيانات في الخلفية ، وتكون أسرع من الوضع الخطي.

طريقة التعريف:

اختار أي اسم

BeBotPython.SyncMode pyA = new BeBotPython.SyncMode();

الدوال التي تقع تحتها:

**Py\_Arguments**

pyL.Py\_Arguments = "argv1 'ar gv 2' argv3 ";

تستخدم لإرسال حجج لبايثون، وكأنك ترسل الحجج بهذه الطريقة على موجه الأوامر

Example.py argv1 'ar gv 2' argv3

وتكتب قبل دالة **Py\_RunPython ، لن تعمل إذا كتبت بعدها – أي تكتب قبل تشغيل بايثون، وقيمة افتراضية لها لا شيء.**

**Py\_PrintCS**

**متغير نصي يحوي أخر قيمة ل دالة PrintToCS المكتوبة بلغة بايثون، ولكن لا تستخدمها حاليا ، ستكون مفيدة أكثر في التطويرات القادمة ، لذا استخدم دالة Py\_Get\_PrintCS** بدلا منها

**Py\_use\_RedirectInput**

pyA.Py\_use\_RedirectInput = true;

إذا كان false يعني أنه سيحدث خطأ عند استخدام دالة **Py\_Set\_Input** ، القيمة الافتراضية له هي true

**WorkingDirectory**

pyA.WorkingDirectory= @"any path";

تكتب قبل **Py\_RunPython ، وتستخدم لتحديد مسار الذي نريد أن نكون موجودين فيه حاليا، قيمته الإفتراضية لا شيء**

**Py\_RunPython**

pyL.Py\_RunPython(@"python\run.exe");

لتشغيل بايثون وفي داخلها تكتب مسار البيئة المصغرة

**Py\_WaitForExit**

pyA.Py\_WaitForExit();

**تستخدم هذه الدالة حصرا بعد Py\_RunPython** وهي **لتحديد المكان الذي ننتظر به بايثون حتى ينتهي، لكن إذا كان كود بايثون لا ينتهي يعني أن البرنامج سيتجمد ولن تتمكن من إغلاقه إلا من خلال task manger.**

**Py\_Set\_Input**

pyA.Py\_Set\_Input("I love python");

**تستخدم لإرسال مدخل لبايثون، بنفس طريقة ادخال المستخدم العادي للinput الموجود في بايثون ولكن عن طريق سي شارب.**

**Py\_Get\_PrintCS**

PyA.Py\_Get\_PrintCS(300);

في داخلها تكتب زمن الانتظار ويكتب في ملي ثانية ، تقوم بإرجاع البيانات المرسلة من بايثون بواسطة دالة **PrintToCS**

**Py\_Kill**

pyA.Py\_Kill();

**تستخدم لقتل وإنهاء مهمة بايثون حالا.**

**Py\_Close**

pyL.Py\_Close();

**تستخدم لإغلاق نافذة بايثون ممكن أن تكون نافذة طرفية.**

**Py\_CloseMainWindow**

pyL. Py\_CloseMainWindow();

تستخدم لإغلاق النافذة الرئيسية لبايثون، بإضافة لجميع النوافذ الطرفية.

ملاحظة دائما **دوال الإغلاق تستخدم بعد Py\_RunPython**