كيفية برمجة Microsoft Agent

تأليف: Mohammad Elsheimy

نظرة خاطفة

درسنا في هذا اليوم يتكلم عن طريقة برمجة Microsoft Agent في برنامج الويندوز Windows Application الخاص بك. تعتبر Microsoft Agentتقنية ليس لها مثيل فهي تمكنك من خلق شخصيات متحركة تقوم بالتحادث مع المستخدم وإرشاده. مثال على شخصيات Microsoft Agent هو مساعد الأوفيس Office Assistant وهو الذي يظهر في أوفيس 2003 والإصدارات السابقة. وهو عبارة عن شخصية تقوم بمساعدة المستخدم وتشرح لها النوافذ والشاشات والأوامر التي يتعرض لها.

سنبدأ أولا بشرح مقدمة عن .Microsoft Agent ثم سنتكلم عن الشخصيات المتوفرة والتي يمكن استخدامها. بعد ذلك سوف نتطرق عن المكتبات التي ستحتاجها وكيفية استخدامها في برنامجك. بعد ذلك سوف نشرع في شرح كل نقاط Microsoft Agent من أوامر وأحداث. سنتعلم كيفية تحريكه وكيفية جعله يتكلم وغيرها.

بالإضافة إلى هذا كله، يحتوي هذا الدرس على برنامج يقوم بتقسيم الملفات كبيرة الحجم إلى أكثر من جزء ثم يقوم بدمجها بعد ذلك. يحتوي هذا البرنامج على شخصية ظريفة من شخصيات Microsoft Agent تتفاعل مع المستخدم ومع حركاته خلال البرنامج وتقوم بشرح أجزاء البرنامج بطريقة مبسطة وظريفة.

مقدمة

تعتبر تقنية Microsoft Agent هي أحد التقنيات الفريدة التي تساعدك على إنشاء واجهة Interface ذكية تقوم بالتفاعل مع المستخدم، فتقوم بالتحرك والتحادث معه. وتعتبر هذه التقنية سهلة من خلال التعامل معها برمجيا حيث توفر لك العديد من الأوامر البسيطة التي تمكنك من التحكم الكامل بهذه التقنية بطريقة بسيطة وسهلة.

توفر لك تقنية Microsoft Agent إمكانية استخدام شخصيات متحركة سواء في برامج الويندوز أو مواقع الويب. حيث يمكنك تحريك هذه الشخصيات وجعلها تتكلم سواء بكلام تحدده أنت أو تقوم بتسجيله بصوتك. كما يمكنك التفاعل أكثر مع المستخدم من خلال الأحداث الخاصة بهذه الشخصيات. حيث يمكنك جعل هذه الشخصيات تقوم بالتحدث مع المستخدم مثلا عند الضغط عليها. كما يمكنك استخدام الأحداث الخاصة بالبرنامج مثلا في جعل هذه الشخصيات تقوم بشرح نقطة معينة في البرنامج عند وصول المستخدم لها أو عند ضغوطه زر المساعدة.F1

كما تلاحظ أنه يمكننا استخدام Microsoft Agent وشخصياتها في العديد من الأماكن والأدوار التي لا حصر لها ومن هذه الأماكن التي يمكنك استخدام هذه التقنية فيها هو إنشاء مساعد للمستخدم وهذا المساعد Assistant يمكننا استغلاله في أدوار كثيرة منها:

- القيام بشرح البرنامج أو الموقع للمستخدم عند دخوله أول مرة عليه.
- القيام بشرح عمليات خاصة في البرنامج ومساعدة المستخدم على استكمالها خطوة بخطوة.
- القيام بإبلاغ المستخدم عند حدوث أمر معين مثلا عند استقباله بريدا إلكترونيا جديدا مثلا.
- القيام بعمل عمليات معينة مثلا البحث على الإنترنت بدلا من المستخدم حيث يقوم المستخدم مثلا بإعطائه نص البحث ويقوم هذا المساعد بالبحث تلقائيا. أو أن يقوم المساعد بالتفاعل مع المستخدم فعند دخوله شاشة أو نافذة معينة مثلا يقوم المساعد بعمل بحث عن المواقع أو الملفات التي يمكنها مساعدة المستخدم.

شخصيات Microsoft Agent

يمكنك مشاهدة تقنية Microsoft Agent وشخصياتها Characters في العديد من الأماكن والبرامج، ومن أشهرها الـ Office الإصدار 2003 أو الإصدارات السابقة حيث يقوم باستغلال تقنية Microsoft Agent في إنشاء مساعد الأوفيس Office Assistant الشخصية التي تقوم بمساعدة المستخدم أثناء تعامله مع برامج الأوفيس. كما تجد أيضا هذه الشخصيات وهذه التقنية في العديد من مواقع الإنترنت. شكل 1 و 2 عبارة عن اثنان من شخصيات Microsoft Agent الشهيرة والاثنان تجدهم مع الأوفيس الأول هو المشبك Clippit والثاني هو الساحر.Merlin

هناك العديد من الشخصيات التي يمكنها الوصول إلى أكثر من 50 شخصية موجودين وجاهزين للتحميل من مواقع الإنترنت ومن الأماكن المختصة. ومن هذه الشخصيات الساحر Merlin وهو يأتي بشكل أساسي مع الويندوز. كما أن هناك العديد من الشخصيات التي تأتي مع الأوفيس.

هذه الشخصيات عبارة عن ملفات لها الامتداد acs. الموقع الرسمي لتقنية Microsoft Agent والذي سوف تجد به العديد والعديد من الشخصيات الجاهزة للتحميل مجانا هو Microsoft Agent Ring.

متطلبات Microsoft Agent

كي يمكنك استخدام تقنية Microsoft Agent في برنامجك يجب أن تكون بعض الأشياء متوفرة:

1. **المكتبات الأساسية Microsoft Agent SDK**

وهي التي تحتوي على جميع الأوامر من إظهار الشخصيات وتحريكها وغيرها من الأوامر الهامة الخاصة بالشخصية والتقنية نفسها Microsoft Agent كما تسمى هذه المكتبات -أو يمكننا القول التقنية- تسمى الخادم Agent Server لأن التقنية نفسها عبارة عن خادم لبرنامج حيث تقوم أنت بإعطائه الأمر كي يقوم بإظهار الشخصية أو تحريكها وهكذا.

2. **الشخصية Character**

وهي عبارة عن ملف من نوع acs يحتوي هذه الشخصية. بالطبع يمكنك تحميل الشخصيات التي تريدها أو حتى إنشاء الشخصيات بنفسك. وفكرة فصل الخادم) Server التقنية) عن الشخصية تعتبر فكرة فريدة من نوعها حيث أنه توفر لنا طريقة واحدة للتعامل مع كل الشخصيات بغض النظر عن إمكانيات كل شخصية أو شكلها أو الحركات الخاصة بها.

3. **محرك تحويل النص إلى صوت Text-to-Speech (TTS) Engine**

وهذا في حالة أنك تريد أن تقوم الشخصية بقول نص معين تقوم أنت بتحديده. وهذا المحرك يختلف باختلاف اللغة. فهناك مثلا محرك للغة الإنجليزية ومحرك آخر للفرنسية وآخر للإسبانية وآخر لليابانية وهكذا. فإذا كنت تريد استخدام لغة معينة فحينها يمكنك تنصيب محرك هذا اللغة على جهاز المستخدم. **لاحظ أنه للأسف ليس هناك محركا للغة العربية. (سوف نتعلم كيفية التغلب على هذه المشكلة لاحقا)**

هذه المعطيات التي حددناها يجب توافرها على جهاز المستخدم وكذلك على جهاز المطور أثناء استخدامه لهذه التقنية. ولحسن الحظ، فهي متوفرة للتحميل من خلال صفحة Microsoft Agent الموجودة على موقع Microsoft والتي يمكنك زيارتها من هنا .كما يمكنك أيضا من خلال هذه الصفحة تحميل برنامج Microsoft Agent Editor والذي يمكنك من خلاله إنشاء الشخصيات الخاصة بك. كما يمكنك أيضا من خلال هذه الصفحة تحميل المساعدة Documentation الخاصة بـ Microsoft Agent وهي ستشرح لك كل شيئ عن هذه التقنية وكيفية استخدامها.

بالطبع يمكنك إضافة هذه المتطلبات إلى المنصب Installer الخاص ببرنامجك كي يقوم بتنصيبها تلقائيا.

ملفات Microsoft Agent

عند تنصيبك لمكتبات Microsoft Agent ستجد أنها تقوم بالتنصيب تلقائيا في المسار .%windir%\MSAgent هناك ستجد جميع الملفات الخاصة بهذه التقنية. لحسن الحظ، لا نحتاج منها إلا الملف AgentCtl.dll ولكن بالطبع **الخادم) Agent Server التقنية) يحتاج إلى كل هذه الملفات**.

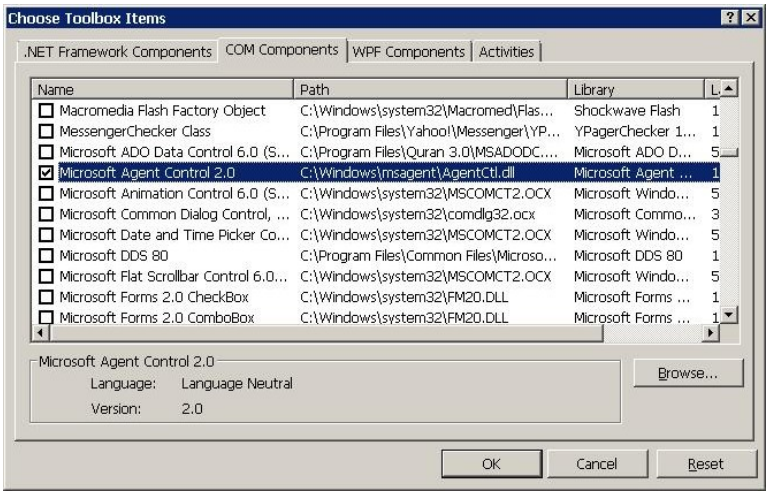
في هذه الملفات أيضا ستجد البرنامج AgentSvr.exe وهو المسؤول كما قلنا عن التقنية كلها أي هو المسؤول عن الأوامر الخاصة بجميع الشخصيات التي تعمل على الجهاز. ولهذا عند تشغيلك لأي برنامج يحتوي على هذه التقنية ستجد أن هذا الملف يعمل في خلفية الجهاز Background فلهذا يمكنك مشاهدة عمله من خلال التبويب Process الموجود في .Task Manager وعند إغلاقك لهذا الملف تغلق جميع الشخصيات المفتوحة.

لاحظ أن المجلد %windir%\MSAgent يحتوي على مجلد داخلي آخر وهو chars وهو يحتوي على جميع الشخصيات الأساسية التي تم تحميلها مع المكتبات. وهي في الغالب شخصية واحدة وهي الساحر.Merlin

استخدام Microsoft Agent

درسنا في هذا اليوم يتكلم عن طريقة استخدام Microsoft Agent في مشروع الويندوز Windows Application الخاص بك. للأسف درسنا لا يتكلم عن كيفية استخدامه في مشروع الويب .Web Application ولكن طريقة استخدامه بسيطة ومماثلة كثيرا. تختلف فقط في كيفية إضافة المكتبة إلى المشروع. يمكنك القراءة عن مشاريع الويب في المساعدة Documentation الخاص بـ Microsoft .Agentأما مشاريع XAML Application فهي شبيهة جدا بالويندوز Windows Application ولكن الفرق هو في كيفية إضافة أداة Microsoft Agentإلى النموذج Form أو النافذة Window التي تريدها. فقط قم بإضافة عنصر من النوع WindowsFormsHost كي يحوي هذه الأداة.

الآن قم بفتح مشروع جديد من نوع .Windows Application بعد فتح المشروع الجديد قم بإضافة أداة Microsoft Agent إلى شريط الأدوات Toolbox عن طريق الضغط عليه بالزر الأيمن واختيار .Choose Toolbox Items لتظر شاشة Choose Toolbox Items شكل 3.



بما أن الأداة Microsoft Agent موجودة في المكتبة AgentCtl.dll كما قلنا. فإن المكتبة AgentCtl.dll ليست مكتبة دوت نت أي لم يتم إنشائها باستخدام بيئة .NET Framework. بدلا من ذلك فإنه تم إنشائها باستخدام تقنيات سابقة ولذلك تسمى هذه المكتبة COM Library. ولهذا فلإضافة هذه الأداة سوف نقوم بالتحويل إلى التبويب COM Components ثم سوف نبحث في القائمة عن الأداة Microsoft Agent Control ونختارها كي يتم إضافتها. انظر الشكل السابق. إن لم تجد الأداة في القائمة قم بضغط زر Browse لتحديد مكان المكتبة وهي موجودة في %windir%\MSAgent.

بعد إضافة الأداة Microsoft Agent Control إلى المشروع ستجد أنه تمت إضافة مكتبتين إلى مراجع References المشروع الخاصة بك وهما AgentObjects و AxAgentObjects ولكن لم تتم إضافة المكتبة AgentCtl.dll! لماذا؟ هذا لأن كما قلنا هذه المكتبة هي COM Library ولذلك فإنك لا تستطيع التعامل معها مباشرة من خلال الكود. بدلا من ذلك فإن Visual Studio يقوم بإنشاء مكتبتين دوت نت عبارة عن وسطاء بينك وبين المكتبة الأصلية وبهما جميع عناصر المكتبة الأصلية. ولهذا يمكنك التعامل معها مباشرة كأنك تتعامل مع المكتبة الأصلية.

الآن قم بإضافة الأداة إلى النموذج Form الذي تريده. انظر شكل 4.

لاحظ أنك لا تحتاج إلى إضافة هذه الأداة في كل النماذج. بل يكفي إضافتها إلى نموذج واحد فقط (وهو في الغالب شاشة البرنامج الرئيسية) وبالتالي ستعمل الشخصية في كل النماذج والشاشات الخاصة بالمشروع.

إذا كنت تستخدم C# قم بإضافة هذا السطر إلى الدالة :Main()

```
[System.STAThreadAttribute]
        void Main()
      {
          ...
      }
```

قمنا بإضافة الخاصية Attribute المسماة بـ STAThreadAttribute إلى الدالة Main() وذلك لأن المكتبة الخاصة بنا هي لها الخاصية

STA (Single-Threaded Apartment)وهذا معناه أنه لا يمكنك التعامل مع هذه المكتبة إلا بـ Thread واحد فقط. وهذا بعكس المكتبات.(MDA (Multi-Threaded Apartment

إظهار الشخصية

يمكننا الآن جعل Agent Server أو يمكننا القول تقنية أو أداة Microsoft Agent يمكننا الآن إظهار الشخصية أو الشخصيات التي نريدها وذلك عن طريق الخاصية Characters الخاصة بهذه الأداة. لاحظ الكود التالي والذي يقوم بتحميل شخصية الساحر Merlin من الملف الخاص بها وإعطائها الاسم Merlin كي نستطيع التعامل معها ولكي نميزها عن الشخصيات الأخرى في حالة استعمالنا لأكثر من شخصية. لاحظ أيضا كيفية الحصول على الشخصية المحملة وكيفية إظهارها. لاحظ أنه هناك فرق بين تحميلها في الذاكرة وبين إظهارها. فيجب تحميل الشخصية أولا ثم إظهارها.

```
// C# Code

      this.axAgent1.Characters.Load("Merlin", @"C:\Windows\MSAgent\chars\merlin.acs");
      AgentObjects.IAgentCtlCharacterEx character =
this.axAgent1.Characters.Character("Merlin");
character.Show(null);
```

```
' VB.NET Code

Me.axAgent1.Characters.Load("Merlin", "C:\Windows\MSAgent\chars\merlin.acs")
Dim character As AgentObjects.IAgentCtlCharacterEx =
Me.axAgent1.Characters.Character("Merlin")
character.Show(Nothing)
```

يحتوي عنصر الشخصية Character على الدالة Hide وذلك لإخفائه. كما تحتوي الأداة Microsoft Agent على الدالة Unload الموجودة في الخاصية Characters وذلك لإلغاء العنصر وإغلاقه. لاحظ أيضا أنه هناك فرق بين مجرد الإخفاء وبين الإغلاق تماما.

جعل الشخصية تتكلم

يمكنك جعل الشخصية تتكلم إن نص تحدده أنت وإما ملف صوتي موجود على الجهاز. وفي حالة تحديد نص سوف يقوم Agent Serverباستخدام Text-to-Speech (TTS) Engine لتحويل النص إلى صوت. وسوف يقوم بتحريك فم الشخصية حتى تحاكي عملية التكلم. وهذا أيضا في حالة الملف الصوتي. لاحظ أنه كما قلنا هناك TTS Engine لأكثر من لغة. ولكن للأسف ليس من ضمنها اللغة العربية فلهذا يمكننا التغلب على هذا عن طريق تسجيل الصوت وجعل الشخصية تقوم هذا الصوت. لاحظ الكود التالي والذي يقوم في السطر الأول بجعل الشخصية تتكلم بنص معين وفي السطر الثاني بجعل الشخصية تتكلم بملف صوتي.

```
// C# Code

character.Speak("Hello everybody, I'm Merlin. I'll guide you throuh the application
windows.", null);
character.Speak(null, @"D:\Recorded Introduction.wav");
```

```
' VB.NET Code

character.Speak("Hello everybody, I'm Merlin. I'll guide you throuh the application
windows.", Nothing)
```

```
character.Speak(Nothing, "D:\Recorded Introduction.wav")
```

لاحظ شكل 5 والذي يظهر الساحر Merlin أثناء التكلم.

هناك أيضا الدالة Think() والتي تجعل الشخصية تظهر كأنها تفكر وليس تتكلم.

تحريك الشخصية

السطر التالي يوضح كيفية تحريك الشخصية. إلى النقطة 100، 100 (س، ص) من الشاشة من أعلى أيسر الشاشة.

```
// C# Code
character.MoveTo(100, 100, null);
```

```
' VB.NET Code
character.MoveTo(100, 100, Nothing)
```

لاحظ أن الدالة MoveTo لها 3 مدخلات الأول هو الإحداثي س (x) من يسار الشاشة. الثاني هو الإحداثي ص (y) من أعلى الشاشة. الثالث هو سرعة الحركة. إذا قمت بتحديد السرعة كـ null (Nothing في VB.NET) كما في المثال فهذا معناه أن السرعة سوف تكون متوسطة أي 1000. إذا قمت بتحديد رقم أعلى من 1000 سوف تكون السرعة أعلى. أو رقم أقل سوف تكون السرعة أبطأ. إذا قمت بتحديد 0 سوف تختفي الشخصية من مكانها وتظهر في المكان الجديد بدلا من انتقالها تدريجيا.

حركات الشخصية

لاحظ الفرق الذي نريده بين التحريك Move وهو عبارة عن انتقال الشخصية من مكانها وبين الحركة Animation وهي المقصودة هنا وهي أي حركة تقوم الشخصية بعملها مثلا كما في المشبك Clippit عندما يتحول إلى شكل دراجة أثناء ظهوره وهكذا. فالـ Animation لا تجعل الشخصية تتحرك من مكانها.

لكل حركة Animation اسم. ولكل شخصية الحركات الخاصة بها ولكنها قد تتشابه في بعض الحركات.ولكي تقوم بالبرمجة باستخدام شخصية معينة يجب عليك معرفة أسماء الحركات الخاصة بها وذلك لتفادي محاولتك لتشغيل حركة معينة قد تكون هذه الشخصية لا تدعمها أو حتى لا تكون باسم مختلق. فلكي تحصل على أسماء جميع الحركات يمكنك ذلك عن طريق الخاصة AnimationNames الخاصة بالشخصية. الكود التالي يوضح كيفية إضافة أسماء جميع الحركات الخاصة بالشخصية إلى قائمة ListBox موجودة على الفورمة.

```
// C# Code

IEnumerator enumerator = character.AnimationNames.GetEnumerator();
while (enumerator.MoveNext())
```

```
    this.listBox1.Items.Add(enumerator.Current.ToString());
```

```
' VB.NET Code

Dim enumerator As IEnumerator = character.AnimationNames.GetEnumerator()
While enumerator.MoveNext()
    Me.listBox1.Items.Add(enumerator.Current.ToString())
End While
```

والآن سوف نقوم بعمل بعض السحر باستخدام الساحر Merlin وسوف نقوم بجعل الساحر يتكلم أثناء عمله لهذه الحركة. لاحظ أن الحركة التي نريدها تسمى.DoMagic1

```
// C# Code
character.Play("DoMagic1");
character.Speak("I'm better than Harry Potter. Am not I?", null);
```

```
' VB.NET Code

character.Play("DoMagic1");
character.Speak("I'm better than Harry Potter. Am not I?", Nothing)
```

شكل 6 يظهر Merlin وهو يستعرض مواهبه. لاحظ أن الحركة سوف تنتهي في خلال ثواني.

لاحظ أن هناك بعض الحركات التي تحتوي على اتجاهات مثلا Left و Right مثل الحركتين GestureRight و GestureLeft والتي تجعل الشخصية تنظر إلى اليمين واليسار. لاحظ أن اليمين واليسار هنا هما يمين ويسار الشخصية وليس أنت. فلهذا يسار الشخصية يعتبر يمينك ويمينها هو يسارك!

هناك أيضا الدالة GestureAt() وهي تقوم بجعل الشخصية تنظر إلى اتجاه معين.

هناك أيضا الدوال Stop() و StopAll() الخاصة بالشخصية. الأولى تقوم بإيقاف حركة معينة. والثانية تقوم بإيقاف حركات من نوع خاص. فالدالة StopAll() تأخذ مدخلا واحدا وهو إما "Move" وهذا إذا كنت تريد إيقاف التحركات Movement الخاصة بالشخصية، أو "Play" وذلك لإيقاف الحركاتAnimations ، أو "Speak" وذلك لإيقاف التكلم، أو null (Nothing في VB.NET) وذلك لإيقاف أي حركات من أي نوع.

خواص عنصر الشخصية

التالي هو قائمة بأشهر الخواص Properties الموجودة في عنصر Object الشخصية:

- **AutoPopupMenu**
إذا قمت بإعطائها القيمة True فإن القائمة المنسدلة (Popup Menu (Context Menu الخاصة بالشخصية سوف تظهر في حالة ضغط المستخدم للزر الأيمن على الشخصية. هذه القائمة لا تحوي إلا الأمر Hide لإخفاء الشخصية.

- **Balloon**
هي خاصية للقراءة فقط Read-Only أي أنه لا يمكنك تعديلها. وتحتوي على خواص البالون الذي يظهر فوق الشخصية عند التكلم. وهذه الخواص منها لون الخلفية ولون نص الكتابة. طبعا لتعديل هذه الخواص يمكنك ذلك عن طريق خصائص سطح المكتب من اختيارات .Appearance فقط قم بتغيير خصائص الملاحظات.Tooltip

- **Left**و**Top**
تحتوي على الإحداثيات، بمعنى آخر مكان، الشخصية على الشاشة. قم باستخدام الدالة ()MoveTo أفضل وذلك لعمل حركة Animationأثناء التحرك.Move

- **SoundEffectsOn**
قم بإعطاء هذه الخاصية True لتشغيل أصوات التأثيرات Effects للشخصية أو False لإلغائها. لاحظ أن أصوات التأثيرات مختلفة عن أصوات التكلم.

- **Speed**
هي خاصية للقراءة فقط Read-Only أي أنه لا يمكنك تعديلها. وتحتوي على سرعة الشخصية. بعض الدوال مثل ()MoveToتأخذ قيمة لتحديد السرعة.

أحداث أداة Microsoft Agent

الآن سوف نلقي الضوء على بعض الأحداث Events الخاصة بالأداة .Microsoft Agent Control بشكل غريب، تحتوي هذه الأداة على أحداث من المفترض أن تكون موجودة في عنصر الشخصية نفسه ولكنها موجودة في الأداة نفسها وهذا يدل على أن المسؤول عن التحركات وجميع عمليات الشخصية ليست الشخصية نفسها بل هو الخادم.Agent Server

- **BallonShow**و**BallonHide**
تحدث عند ظهور البالون فوق شخصية أو إختفائه.

- **ClickEvent**و**DblClick**
تحدث عند الضغط بالفأرة ضغطة واحدة أو مرتان على شخصية.

- **DragStart**و**DragComplete**
تحدث عند بدأ تحريك شخصية وعند الانتهاء.

- **ShowEvent**و**HideEvent**
تحدث عند ظهور شخصية وعند اختفائها.

- **MoveEvent**تحدث عند تحريك شخصية

إنشاء مدير الشخصيات

عند تعاملك مع Microsoft Agent وخصوصا عندما يكون عندك نوافذ عدة في برنامجك وعندما يكثر التعامل مع الشخصية ستجد انه من الصعوبة أن تقوم كل مرة باسترجاع الشخصية أو الشخصيات والتعامل معها بسهولة وخصوصا مع كثرة الدوال وكثرة الحركات التي تريدها فستجد أنه من الصعوبة تذكر أسماء الحركات كاملة. ولذلك فإن الحل هو في أن تقوم بإنشاء عنصر Class معينة تقوم باحتواء هذه الشخصية وتقوم بإضافة الأوامر الخاصة بك إليها وكذلك الأوامر التي تحوي أوامر متعددة. ونسمي هذه الكلاس المدير أو المتحكم .Controller لاحظ الكود التالي والذي يظهر لنا Controller متميز لهذه الشخصية.

```
// C# Code

      internal sealed class AgentController
      {
          private AgentObjects.IAgentCtlCharacterEx _char;
          private AxAgentObjects.AxAgent _agent;

          public AgentController(AxAgentObjects.AxAgent agent, string
characterLocation)
          {
              _agent = agent;
              _agent.Characters.Load("CHAR", characterLocation);
              _char = _agent.Characters.Character("CHAR");
              _char.Show(null);
          }

          public bool IsVisible() { return _char.Visible; }
          public void Animate(string animation, bool stopAll)
          {
              if (stopAll)
                  _char.StopAll(null);
              _char.Play(animation);
          }
          public void Speak(string text, bool stopAll)
          {
              if (stopAll)
                  _char.StopAll(null);
              _char.Speak(text, null);
          }
          public void MoveTo(int x, int y, bool stopAll)
          {
              if (stopAll)
                  _char.StopAll(null);
              _char.MoveTo((short)x, (short)y, null);
          }

          public void Surprise()
          {
              _char.StopAll(null);
              _char.Play("Alert");
              _char.Play("Sad");
          }
      }
```

```
' VB.NET Code

      Friend NotInheritable Class AgentController
          Private _char As AgentObjects.IAgentCtlCharacterEx
          Private _agent As AxAgentObjects.AxAgent

          Public Sub New(ByVal agent As AxAgentObjects.AxAgent, ByVal
characterLocation As String)
                 _agent = agent
```

```
            _agent.Characters.Load("CHAR", characterLocation)
            _char = _agent.Characters.Character("CHAR")
            _char.Show(Nothing)
        End Sub

        Public Function IsVisible() As Boolean
            Return _char.Visible
        End Function

        Public Sub Animate(ByVal animation As String, ByVal stopAll As Boolean)
            If (stopAll) Then
                _char.StopAll(Nothing)
            End If
            _char.Play(animation)
        End Sub

        Public Sub Speak(ByVal text As String, ByVal stopAll As Boolean)
            If (stopAll) Then
                _char.StopAll(Nothing)
            End If
            _char.Speak(text, Nothing)
        End Sub

        Public Sub MoveTo(ByVal x As Integer, ByVal y As Integer, ByVal stopAll As
Boolean)
            If (stopAll) Then
                _char.StopAll(Nothing)
            End If
            _char.MoveTo(x, y, Nothing)
        End Sub

        Public Sub Surprise()
            _char.StopAll(Nothing)
            _char.Play("Alert")
            _char.Play("Sad")
        End Sub
    End Class
```

لاحظ في الكود السابق أنه لجعل الشخصية تقوم بعمل حركة كأنها تفاجأت بشيئ، يجب أن نقوم بإيقاف الحركة السابقة -إن كانت تعمل- ثم تشغيل الحركة Alert ثم تشغيل الحركة Sad وبالتالي سنجعل الشخصية تنتبه ثم تقوم تحزن كأنها تفاجأت بشيئ كأن خطأ معين مثلا حدث في البرنامج.

لاحظ بالطبع أنه من الأسهل عليك النداء على الدالة Surprise بدلا من النداء على الثلاثة دوال التي بداخلها في حالة أنك تريد عمل هذه الحركة.
لاحظ أيضا أنك يمكنك إضافة دوالك وأحداثك.

إنهاء Microsoft Agent

بما أن مكتبات Microsoft Agent كما قلنا هي COM Libraries وبما أن Microsoft Agent تقنية تقوم باستهلاك موارد من الجهاز فإنك تريد حالة انتهائك من استخدامها إزالة هذه الموارد حتى لا تبقى في الذاكرة لفترات طويلة. بالطبع وكما تعرف أنه يمكنك استخدام الدالة Characters.Unload الموجودة في الأداة Microsoft Agent Control وذلك لإنهاء الشخصية من الذاكرة بعد انتهاء التعامل معها. بالإضافة إلى ذلك يجب عليك إنهاء الأداة نفسها من الذاكرة وهي عبارة عن كود بسيط وهو كالتالي:

```
// C# Code
while (System.Runtime.InteropServices.Marshal.FinalReleaseComObject(_char) > 0);
```

```
' VB.NET Code
While System.Runtime.InteropServices.Marshal.FinalReleaseComObject(_char) > 0
End While
```

يمكنك إضافة هذه الأسطر إلى أي مكان لإنهاء الأداة كما يمكنك إضافتها إلى المدير Controller الخاص بك. حيث يمكنك تطبيق الـ Interfaceالمسماة بـ IDisposable على المدير وتطبيق الدالة الوحيدة الخاصة بها وهي Dispose() وإضافة هذا الكود بها. ويمكنك بعد ذلك إضافة سطر النداء على الدالة Dispose() بدلا من الأسطر نفسها في المكان الذي ترغب فيه في الانتهاء من الأداة والشخصيات. لاحظ الكود التالي.

الكود مختصر للوضوح.

```
// C# Code
      internal sealed class AgentController : IDisposable
      {
          ............
          public void Dispose()
          {
              _char.StopAll(null);
              _char.Hide(null);
              _agent.Characters.Unload("CHAR");
      while (System.Runtime.InteropServices.Marshal.FinalReleaseComObject(_char) > 0);
          }
      }
```

```
' VB.NET Code

      Friend NotInheritable Class AgentController
          Implements IDisposable

          ............

          Public Sub Dispose() Implements IDisposable.Dispose
              _char.StopAll(Nothing)
              _char.Hide(Nothing)
              _agent.Characters.Unload("CHAR")
              While
System.Runtime.InteropServices.Marshal.FinalReleaseComObject(_char) > 0
              End While
          End Sub
      End Class
```

كما يمكنك إضافة كود النداء على هذه الدالة في كود الدالة Dispose() أيضا ولكن الخاصة بالنموذج وهكذا تضمن إزالة الأداة من الذاكرة فور انتهاء النموذج أي انتهاء برنامجك (في الغالب). لاحظ الكود التالي في النموذج:

```
// C# Code

      Protected override void Dispose(bool disposing)
      {
          try
          {
              if (disposing)
              {
```

```
                this.axAgent1.Dispose();
                if (components != null)
                    components.Dispose();
            }
        }
        finally { base.Dispose(disposing); }
    }
```

```
' VB.NET Code

    Protected Overrides Sub Dispose(ByVal disposing As Boolean)
        Try
           If (disposing) Then
                Me.axAgent1.Dispose()
                If (components IsNot Nothing) Then
                    components.Dispose()
                End If
            End If
        Finally
            base.Dispose(disposing)
        End Try
    End Sub
```

يمكنك الوصول إلى الدالة Dispose() الخاصة بالنموذج من خلال شاشة تحرير الكود Code Editor من خلال القائمة Members الموجودة في أعلى يمين الشاشة.

ليس فقط إزالة هذا العنصر من الذاكرة هو المطلوب، بل هذا مطلوبا في جميع العناصر بجميع الأنواع وهي التي توفر لنا الخاصية Dispose() أما عناصر COM فيمكننا إزالتها بالطريقة السابقة.

المراجع

بعض المراجع الهامة في تقنية:Microsoft Agent

- **مساعدة Microsoft Agent**
  يمكنك تحميلها من صفحتها بـ .Microsoft انظر القسم "استخدام."Microsoft Agent
- **كتاب Microsoft Agent Software Development Kit :**
  ISBN: 0-7356-0567-X
- **كتاب Developing for Microsoft Agent :**
  ISBN: 1-5723-1720-5

المثال

المثال ليس عبارة عن فقط مثال. بل هو برنامج كامل يقوم بتقسيم الملفات كي يتم حفظها على أقراص مرنة Floppy Disks أو اسطوانات مدمجة CDs أو حتى إرسالها بالبريد الإلكتروني أو غيرها. كما يقوم البرنامج بدمجها مرة أخرى.

يحتوي هذا البرنامج برنامج !Geming PartIt على شخصية الساحر Merlin والذي يقوم بشرح خطوات البرنامج بطريقة بسيطة وسلسة.

قم بتحميل البرنامج من هنا
قم بتحميل الكود من هنا

خاتمة

تعلمنا في هذا الدرس الكثير والكثير من الأفكار المدهشة والتي يمكنك تطبيقها في برنامجك لتوفير واجهة Interface جميلة وظريفة يمكن للمستخدم التعامل معها ببساطة. في دروس أخرى سنتعلم أفكار أخرى وتقنيات أخرى .