**الفصل التاسع عشر**

**مقدّمة لبرمجة الإنترنت**

**Introduction To Web Programming**

* [**ما هو الإنترنت؟:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20ما%20هو%20الإنترنت.htm)
* [**لغة HTML:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20HTML%20لغة.htm)
* [**ASP.NET:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\3-%20ASP.htm)
* [**بناء تطبيقات الإنترنت:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\4-%20بناء%20تطبيقات%20الإنترنت.htm)
* [**التفاعل مع تطبيق الإنترنت:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\5-%20التفاعل%20مع%20تطبيق%20الإنترنت.htm)
* [**أدوات الإنترنت Web Controls:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\6-%20أدوات%20الإنترنت.htm)
* [**إبقاء الحالة Maintaining State:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\7-%20إبقاء%20الحالة.htm)
* [**كائنات ASP.NET:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\8-%20ASP%20كائنات.htm)
* [**استخدام البيانات المؤقّتة Cookies:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\9-%20استخدام%20البيانات%20المؤقّتة.htm)
* [**التعامل مع أكثر من نموذج (صفحة):**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\1-%20مقدّمة%20لبرمجة%20الإنترنت\10-%20التعامل%20مع%20أكثر%20من%20نموذج.htm)

**ما هو الإنترنت:**

تسمّى أجهزة الكمبيوتر التي تستضيف مواقع الإنترنت Web Sites بالخوادم Servers، حيث تمنح الخدمات للعملاء Clients الذين يدخلون هذه المواقع.. وطبعا تكون هذه الأجهزة خاصّة مثل أجهزة الكمبيوتر الرئيسيّة Main Frames، وهي أكبر في الحجم والسرعة ومساحة التخزين من الكمبيوتر الشخصيّ.

ولكن ما هو الإنترنت؟

إنّه عبارة عن شبكة كمبيوتر هائلة، تستخدم بروتوكولات معيّنة للتفاهم فيما بين أجزائها.. ومن أهمّ هذه البروتوكولات، Transmission Control Protocol/Internet Protocol (TCP/IP)، وهو بروتوكول بسيط، قادر على التعامل مع كلّ أجهزة الكمبيوتر وكلّ أنظمة التشغيل.

ولا تجزع هكذا.. لسنا بحاجة لشرح هذا البرتوكول أو سواه.. إنّ كلّ هذه التفاصيل تتمّ في الخفاء بعيدا عنك، وكلّ ما عليك لاستغلال الإنترنت، هو وجود متصفّح الإنترنت Internet Explorer على جهازك، حيث يقوم بفتح مواقع الإنترنت وتحميل الصفحات المطلوبة إلى جهازك (هذا بافتراض أنّ جهازك يحتوي على كارت إنترنت، وأنّه متّصل بالهاتف).

إنّ بناء تطبيقات الإنترنت قريب جدّا من بناء تطبيقات الويندوز، ولكنّ هناك العديد من الاختلافات التي يجب مراعاتها.

وفي هذا الفصل سنرى كيف نطبّق كلّ ما تعلّمناه عن VB في تطبيقات الإنترنت.. ولكن لا بدّ لنا من بعض المعلومات الأساسيّة عن لغة صفحات الإنترنت Hypertext Markup Language (HTML)، وعن كيفيّة تفاعل العملاء مع الخوادم على الإنترنت.

**لغة HTML:**

* [**ما هي:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20HTML%20لغة\1-%20ما%20هي.htm)
* [**عناصر الكود في لغة HTML:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20HTML%20لغة\2-%20عناصر%20الكود.htm)
* [**لغة XHTML:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20HTML%20لغة\3-%20لغة%20XHTML.htm)
* [**أدوات HTML:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20HTML%20لغة\4-%20أدوات%20HTML.htm)

**ما هي:**

إنّ HTML هي اللغة التي يتخاطب بها الخوادم والعملاء لنقل الوثائق على الإنترنت.

وليست الوثيقة إلا ملفّا نصّيّا Text file، يحتوى على مراجع تشير إلى بعض الصور ليتمّ عرضها، وبعض الأصوات ليتمّ تشغيلها، وبعض الوصلات Links تشير إلى صفحات ووثائق أخرى.

ويمكنك أن تنشئ صفحات HTML بأيّ برنامج محرّر نصوص مثل Notepad، حيث تكتب فيه أوامر اللغة يدويّا.. لقد كان هذا هو الوضع قبل إصدار التطبيقات المسمّاة WYSIWYG (ما تراه هو ما تحصل عليه what you see is what you get)، مثل تطبيق Microsoft FrontPage، بل مثل تطبيق Microsoft Office 2000 وما يليه من إصدارات.. في هذه التطبيقات تستخدم واجهةً مرئيّةً لإدراج عناصر الصفحة وتنسيق ألوانها وخطوطها وصورها، دون أن تكتب حرفا واحدا من لغة HTML.

وتوجد صفحات الإنترنت على الخوادم، التي تعرضها لكلّ من يطلبها، حيث يمكن في هذه الحالة أن يحفظ نسخة منها على جهازه.

ولكلّ خادم عنوان يسمّى Uniform Resource Locator (URL)، يكون على شاكلة:

[http://www.example.com](http://www.example.com/)

حيث http هو اسم البروتوكول المستخدم في التعامل مع الخادم، و www.example.com هو عنوان الخادم على الإنترنت.

إنّ لكلّ جهاز على الإنترنت عنوان لبرتوكول الإنترنت Internet Protocol (IP) Address، يكون على شاكلة 193.22.103.18.. ونظرا لأنّه من الصعب علينا أن نحفظ مثل هذه الأرقام، فإنّ كلّ خادم يحتفظ بجدول به هذه الأرقام والعناوين النصّيّة التي تناظرها.. وعندما تقوم بكتابة العنوان www.example.com فإنّ الخادم يبحث عن الرقم المناظر له ويستخدمه.

**عناصر الكود في لغة HTML:**

تتكوّن لغة HTML من مجموعة من عناصر تنسيق النصوص Text-formatting Tags، توضع كلّ منها بين علامتي < >.. وفي الغالب يظهر كلّ عنصر تنسيق Tag في صورة زوج: عنصر التنسيق الأوّل يمثّل بداية التنسيق، والآخر يمثّل نهايته.. فمثلا، لكي تعرض بعض الكلمات بخطّ سميك Bold، ضعها بين زوج من عنصر التنسيق <B> كالتالي:

Some <B>words</B> in the sentence are formatted in <B>bold</B>.

طبعا لاحظت أن عنصر تنسيق النهاية يسبقه الرمز "/": (</B>).

الجملة السابقة ستظهر في المتصفّح كالتالي:

Some **words** in the sentence are formatted in **bold**.

ولترى مثالا لصفحة انترنت، انسخ الجمل التالية إلى ملفّ نصّ عاديّ، واحفظه.. غيّر امتداد الملفّ إلى .HTM .. انقر الملفّ بعد ذلك مرّتين بالفأرة، ليتمّ عرضه في متصفّح الإنترنت:

<HTML>

<HEAD>

<TITLE>هذا هو عنوان الصفحة</TITLE>

</HEAD>

<BODY>

هذا هو نصّ الصفحة: <B>مرحبا بك</B>

</BODY>

</HTML>

وتستخدم بعض السمات Attributes مع عناصر التنسيق، وهي بعض الكلمات الخاصّة، التي تؤدّى وظيفة محدّدة مع عنصر التنسيق.. فعلى سبيل المثال، يستخدم عنصر التنسيق <A> لإدراج وصلة فائقة Hyperlink في الوثيقة.. هذا العنصر يستخدم السمة HREF.. انظر للجملة التالية، التي تضيف للصفحة وصلة لموقع ميكروسوفت على الإنترنت:

This <A HREF="http://www.microsoft.com">link</A> leads to Microsoft's home page.

إنّ النصّ الموجود بين علامتي </A> <A> سيعرض على أنّه وصلة فائقة، كالتالي:

This [link](http://www.microsoft.com) leads to Microsoft's home page

حيث تخبر السمة HREF المتصفّح بالعنوان URL الذي سيتمّ الانتقال إليه عند ضغط الوصلة (ولا يغرض هذا العنوان كنصّ).

ولمزيد من التفاصيل حول لغة HTML، يمكنك الرجوع [للملحق 4](file:///C:\Users\Administrator\Desktop\05-%20الملاحق\HTML.htm)، في نهاية هذا الكتاب.

**لغة XHTML:**

في عام 2000 ظهر أوّل إصدار من لغة HTML الممدّدة Extensible، وذلك حتّى تتجاوب مع قواعد لغة XML.. وتستخدم هذه اللغة الجديدة معظم عناصر تنسيق HTML، ولكن مع بعض القيود:

- يجب كتابة أسماء عناصر التنسيق بحروف صغيرة Small (وإن كنّا نكتبها في هذا الفصل كبيرة للإيضاح.. ولكنّك ستجدها في مشاريع هذا الفصل مكتوبة بحروف صغيرة).

- يجب وضع قيم السمات Attributes بين علامتي تنصيص.

- يجب إغلاق كلّ مقطع تنسيق بعنصر نهاية التنسيق.

**أدوات HTML:**

تتعامل لغة HTML مع بعض الأدوات البسيطة، مثل مربّعات النصّ Text Boxes، أزرار التحويل Radio Buttons، مربّعات الاختيار Check Boxes، الأزرار Buttons، وغيرها.

ويتمّ عرض هذه الأدوات على صفحة الإنترنت، حيث يمكن للمستخدم أن يختار أو يدخل بعض القيم ثمّ يضغط زرّ إرسالها إلى الخادم.. ويقوم الخادم بمعالجة هذه البيانات، وإعداد صفحة جديدة فيها ناتج المعالجة، حيث يرسلها ليتمّ عرضها على جهاز المستخدم.

وللتعرّف على هذه الأدوات، عليك بقراءة [الملحق رقم 4](file:///C:\Users\Administrator\Desktop\05-%20الملاحق\HTML\7-%20Forms%20and%20Controls.htm).. وإنّ كان ASP.NET سيغنيك عن ذلك، حيث ستشعر أنّك تكتب تطبيق VB عاديّا تماما وتصمم أدواته بواجهة مرئيّة، في حين أنّ كلّ هذه التفاصيل ستكون مختفيةً بمعزل عنك.

**ASP.NET:**

العيبُ الرئيسيّ في صفحات الإنترنت هو أنّها ثابتة المحتوى، ممّا يعنى صعوبة تحديثها.. هنا كان لا بدّ من ظهور تقنية Common Gateway Interface (CGI)، التي تسمح بإضافة محتويات مرنة إلى الإنترنت، بحيث تكون قابلة للتعديل.. فمثلا، يمكن للعميل أن يستعلم عن قائمة الأسعار الخاصّة ببعض المنتجات اليوم.. في هذه الحالة لن يعيد الخادم صفحة HTML ثابتة المحتوى، لكنّه سينفّذ تطبيقا ما يجمع المعلومات المطلوبة، وبعد ذلك يحوّلها لتنسيق صفحة HTML ويرسلها للعميل.. وبهذا يرى العميل أحدث المعلومات.

ولكن تتبقّى مشكلة التعقيد في كتابة تطبيقات CGI وتطويرها.. هنا قدّمت ميكروسوفت الحلّ في صورة صفحة الخادم النشيطة ASP (Active Server Page)، وهي عبارة عن صفحة إنترنت تحتوي على برنامج مكتوب في الغالب بلغة VBScript، بحيث يمكنه أن يتفاعل مع العميل ومع المكوّنات الأخرى على الخادم.. فمثلا، يمكن لصفحة ASP أن تتفاعل مع قواعد البيانات للحصول على بعض البيانات المطلوبة.

والآن نحن نتعامل مع أحدث إصدار من ASP هو ASP.NET، الذي يمكّنك من كتابة برامج تعمل على الخادم بلغة VB.NET.

إنّ هناك ميزات هائلة لاستخدام VB.NET في كتابة هذه البرامج، فبخلاف توفير وقت التعلّم علينا، وسهولة البرمجة بـ VB، فإنّ ذلك يجعل التطبيق يعمل بسرعةٍ أكبر على الخادم.. هذا بالإضافة إلى استغلال كلّ إمكانيّات بيئة التطوير المتكاملة IDE الخاصّة بـ VS.NET، واستخدام مجموعة جديدة من أدوات الإنترنت Web Controls التي صمّمت لعلاج القصور في أدوات HTML.

أمّا الميزة الكبرى، فهي إحساسك بأنّك تكتب كود VB عادي تماما، مبتعدا معظم الوقت عن مفاهيم HTML و XML بل و ASP نفسه.

**ملحوظة:**

ليست لغة VB وحدها هي التي تستطيع بها كتابة تطبيقات ASP.NET، بل إنّ بإمكانك كتابتها بأيّ لغة تعمل على VS.NET مثل C#، J#، وCOBOL، إلى آخر قائمة اللغات التي تنضمّ كلّ حينٍ إلى VS.NET.

إنّ تطبيقك يعمل على الخادم حيث توجد أدوات الإنترنت Web controls.. فكيف إذن تظهر هذه الأدوات على جهاز العميل؟

في هذه الحالة يقوم ASP.NET بتحويل أدوات الإنترنت إلى أدوات HTML وإرسالها كصفحة HTML لجهاز العميل ليتمّ عرضها.

الخلاصة، أنّك تتعامل مع نسخة رائعة من ASP.NET تشعرك بأنّك تكتب تطبيق ويندوز عاديّ، بينما تقوم هي في الخلفيّة بإنتاج صفحات الإنترنت المطلوبة.

**بناء تطبيقات الإنترنت:**

ابدأ تطبيق ASP.NET جديدا (باختيار النوع ASP.NET Web Application من مشاريع VB).. سمّ هذا التطبيق Register (يجب أن يظهر الاسم في خانة الموضع Location كالتالي: http://localhost/Register).

**ملحوظة هامّة:**

حتّى تستطيع إنشاء أو فتح تطبيقات الإنترنت، يجب أن تفتح متصفّح الإنترنت Internet Explorer ومن قائمة ملفّ Flle تتأكّد أنّ الخيار Work offline غير فعّال.. فلو كانت بجواره علامة (✓) فاضغطه بزرّ الفأرة مرّة لإزالتها.. بغير هذا ستحصل على رسالة من VS تطالبك بتنفيذ هذه الخطوات، قبل السماح بفتح تطبيقات الإنترنت.

أين سيتمّ حفظ هذا التطبيق؟.. ستجده على المسار الافتراضيّ c:\Inetpub\wwwroot.. فإذا أردت تشغيل أمثلة هذا الفصل، فعليك بنقلها إلى هذا المجلّد.. بغير هذا لن تعمل.

ويجب أن تعرف هنا، أنّ تطبيق الإنترنت لا يتمّ ترجمته إلى ملفّ تنفيذي .EXE، بحيث يمكنك تشغيله بنقره مرّتين بالفأرة.. إنّ تطبيق الإنترنت يتمّ ترجمته إلى صفحة إنترنت بامتداد .ASPX، حيث يمكن تشغيلها بكتابة عنوانها واسمها في متصفّح الإنترنت.. (ولو ضغطت F5 في VS فسيتمّ تشغيل متصفّح الإنترنت وعرض الصفحة عليه تلقائيّا).

وكما اعتدنا في تطبيقات الويندوز، ستجد أن نموذجا افتراضيّا قد أضيف للتطبيق.. هذا النموذج يشبه صفحة الإنترنت، لأنّه لن يظهر على سطح المكتب، وإنّما سيتمّ عرضه في متصفّح الإنترنت.. وطبعا هذا النموذج من نوع نماذج الإنترنت Web Form، واسمه الافتراضيّ WebForm1.

افتح صندوق الأدوات ToolBox.. ستجد به شريطين يحتويان على الأدوات التي سنعمل بها: شريط Web Controls، وشريط HTML Controls.. ويمكنك تحويل أيّ أداة HTML إلى أداة إنترنت، بوضع نسخة منها على النموذج، وضغطها بزرّ الفأرة الأيمن، واختيار الأمر Run As Server Control من القائمة الموضعيّة.

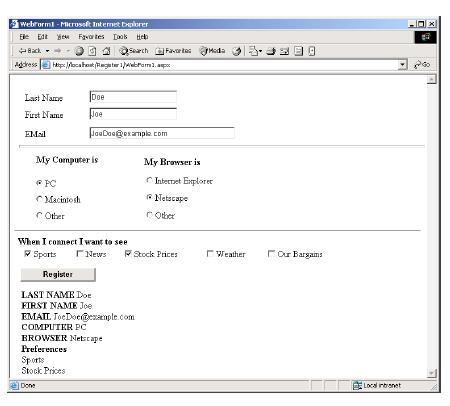
وطبعا لا يوجد أيّ اختلاف في كيفيّة تصميم نموذج الإنترنت عن نموذج الويندوز.. فقط، اسحب الأدوات إلى النموذج، وغيّر خصائصها من نافذة الخصائص.

هذا، ولا تمتلك أدوات الإنترنت عددا كبيرا من الخصائص كالذي تمتلكه أدوات الويندوز، ولكنّه بالتأكيد أكبر من عدد الخصائص الذي تمتلكه أدوات HTML.

**ملحوظة:**

تقوم خاصيّة ID بنفس وظيفة خاصيّة Name في أدوات الويندوز، حيث تمثّل الاسم البرمجيّ للأداة.. إنّه مجرّد اختلاف في الأسماء فحسب، وهو له ما يبرّره.. إنّ مبرمجي VB ضيوفٌ على برمجة الإنترنت، أمّا مبرمجو الإنترنت، فقد اعتادوا كتابة كود HTML لسنوات.. لهذا يجب مراعاة المفاهيم والمصطلحات والأسماء التي اعتادوا عليها طوال هذه السنوات.. هذا بخلاف أنّ مظهر نموذج الإنترنت ما هو إلا كود HTML، يتمّ إرساله لمتصفّح الإنترنت لعرض الصفحة.. لا بأس.. القليل من الاختلافات عمّا اعتدناه لن تضرّ أحدا!

والآن ضع بعض الأدوات على النموذج، ليبدو كما في الصورة:



تلاحظ طبعا أنّ واجهة الاستخدام إنجليزيّة.. إنّ أوّل مطبّ سيقابلك في تطبيقات الإنترنت، هو أنّ أدوات الإنترنت لا تحتوى على خاصيّة RightToLift.. لا تخف: هناك حلّ.. إنّ النموذج يمتلك خاصيّة الاتجاه Dir، التي تأخذ القيمتين التاليتين:

|  |  |
| --- | --- |
| rtl | وهي تعمل على محاذاة النموذج وما عليه من أدوات من اليمين إلى اليسار. |
| ltr | وهي تعمل على محاذاة النموذج وما عليه من أدوات من اليسار إلى اليمين. |

**ملحوظة:**

قد تؤدّى المحاذاة من اليمين إلى اليسار إلى حدوث بعض الشعوذة في أداء النموذج، كاختفاء بعض اللافتات أو اختفاء عناوينها، خاصّة لو حاولت نقلها من مواضعها.. عامّةً لا داعي لتبخير التطبيق ولا لتحضير (شمهورش)!!.. هكذا تبدأ ميكروسوفت أيّ إصدار جديد من إصداراتها، حيث ترهق المبرمجين العرب إلى أقصى حدّ في التعامل مع هذه الإصدارات، إلى أن يمطروها بمطالب التعديل، التي سرعان ما تظهر في الإصدارات التالية وعليك خير!!.. لهذا أنصحك بأن تصمّم النموذج كلّه جيّدا أولا، وتضع كلّ أداة في موضعها، وتكتب عناوينها، ثمّ بعد ذلك تغيّر خاصيّة Dir للنموذج.. فإذا اختفت بعض العناوين، فعليك بتغيير حجم اللافتة ليعود عنوانها للظهور، وإذا اختفت بعض الأدوات عند تحريكها، فحرّك المنزلق الأفقيّ، فقد تكون هذه الأدوات قد انتقلت إلى أقصى أقصى اليسار.. ومن الأفضل تحريك هذه الأدوات عن طريق الخصائص الموجودة في نافذة الخصائص!!!

ولكن: ماذا لو أردت أن تغيّر المحاذاة لأداة بعينها دون غيرها، كأن تجعل محاذاة مربّع النصّ الذي يستقبل العنوان البريديّ من اليسار لليمين على عكس باقي الأدوات؟

سيحتاج هذا للقليل من العمل، فالأدوات للأسف لا تمتلك خاصيّة Dir.. اتبع هذه الخطوات:

- في أسفل النموذج ستجد اختيارين: عرض التصميم Design، وعرض HTML.. اضغط هذا الاختيار الأخير.

- الآن ترى كود HTML الذي يعبّر عن النموذج.. ابحث عن المقطع التالي في الكود، وهو ذلك الذي يعرّف مربّع النصّ TxtEmail وخصائصه:

<asp:textbox id="txtEMail" style="Z-INDEX: …………………>

وأبسط طريقة للعثور على هذا التعريف، هي ضغط Ctrl + F من لوحة المفاتيح، ليظهر مربّع حوار البحث.. استخدمه للبحث عن النصّ txtEMail في الصفحة.

- الآن كلّ ما عليك هو أن تضيف هذه السمة Atrtribute المسمّاة Dir إلى التعريف السابق كالتالي:

<asp:textbox dir="ltr" id="txtEMail" style="Z-INDEX: …………………>

* ستلاحظ ظهور خطّ متعرّج تحت كلمة Dir ولو حلّقت فوقه بالفأرة فسيظهر لك تلميح يخبرك بأنّ هذه السمة غير معرّفة لمربّع النصّ.. كبّر دماغك.. كلّ شيءٍ سيعمل على ما يرام!

أعتقد أنّك تستطيع الآن تعريب واجهة التطبيق كما يحلو لك!

الآن، تعال نضيف بعض الكود للزرّ.. كلّ ما سنفعله، هو أن نقرأ القيم التي أدخلها المستخدم، وسنعرضها في اللافتة الموجودة في أسفل النموذج (واسمها Values).. وطبعا لن نعقّد الموضوع بإجراء أيّ عمليّات على هذه القيم، فنحن ما زلنا نحبو في بداية الدرب!

انقر الزرّ مرّتين بالفأرة، ليظهر لك الحدث المستجيب لضغط الزرّ.. ستجد فيه ما يلي:

**Private Sub Button1\_Click(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles Button1.Click**

**End Sub**

جليٌّ جدّا أنّك تألف هذه الصيغة!

والآن كلّ ما عليك هو كتابة كود VB المعتاد.

ولكن.. هل سيوضع هذا الكود في الصفحة؟

لا.. إنّ الزرّ سيعرض على الصفحة، ولكن عندما يضغطه المستخدم، فإنّ تنبيها سيرسل للخادم، حيث سيقوم بإطلاق حدث الضغط وتنفيذ ما به من كود، وإعادة الناتج للعميل.

والآن، اكتب ما يلي في حدث ضغط الزرّ:

**Private Sub Button1\_Click(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles Button1.Click**

**Values.Text = "الاسم الأخير: " & txtLName.Text**

**Values.Text &= "<BR>" & "الاسم الأوّل: " & txtFName.Text**

**Values.Text &= "<BR>" & "العنوان البريدي: " & txtEMail.Text**

**Values.Text &= "<BR>" & "الكمبيوتر: " & \_**

**radioHardware.SelectedItem.ToString**

**Values.Text &= "<BR>" & "المتصفّح: " & \_**

**radioBrowser.SelectedItem.ToString**

**Dim i As Integer**

**Values.Text &= "<BR><B>الاختيارات</B>"**

**For i = 0 To chkView.Items.Count - 1**

**If chkView.Items(i).Selected Then**

**Values.Text &= "<BR>" & \_**

**chkView.Items(i).Value.ToString**

**End If**

**Next**

**End Sub**

أعتقد أنّك تألف هذا الكود فهو لا يحتاج لأيّ شرح.. الملاحظة الوحيدة التي تحتاج لإيضاح، هي أنّ اللافتة لا تستقبل نصوصا فقط، بل تستقبل عناصر تنسيق HTML أيضا، مثل عنصر بداية الفقرة <BR>، وعنصر الخطّ السميك <B>.. إنّ هذا يعني أمرين في غاية الأهميّة:

- أنّ اللافتة تمنحك قدرات فائقة تماثل قدرات مربّع النصّ المنسّق RichTextBox في تطبيقات الويندوز، حيث تسمح لك بتنسيق كلّ جزء من أجزاء النصّ بخطّ وحجم وسمك ولون مختلف، وغير ذلك من سمات التنسيق.

- أنّك يجب أن تتقن لغة HTML لأنّها عظيمة الفائدة كما ترى.. [إذن فعليك بملحق 4](file:///C:\Users\Administrator\Desktop\vb.net\05-%20الملاحق\HTML.htm).

والآن اضغط F5 واختبر التطبيق.. ستظهر لك نافذة متصفّح الإنترنت، حيث سيتمّ تحميل الصفحة عليها.. أسدل القائمة View واضغط الأمر Source.. سيظهر لك تطبيق Notepad حيث سيعرض كود HTML المناظر لهذه الصفحة.

لاحظ أنّ مربّع النصّ الأوّل موضوع على الصفحة بجملة HTML التالية:

<INPUT NAME="txtLast" TYPE="text" ID="txtLast" TABINDEX="1"

STYLE="z-index: 102; left: 120px; position: absolute; top: 12px" />

أغلق Notepad، وأدخل بعض القيم في صفحتنا.. اكتب في مربّع النصّ الأوّل كلمة "محمود"، ثمّ اضغط الزرّ.. سيتمّ ملء اللافتة بالقيم.

والآن افتح مصدر HTML مرّة أخرى.. ستجد أنّ الجملة التي تعرّف مربّع النصّ الأوّل قد صارت:

<INPUT NAME="txtLast" TYPE="text" VALUE="محمود" ID="txtLast" TABINDEX="1" STYLE="z-index: 102; left: 120px; position: absolute; top: 12px" />

طبعا لاحظت أنّ الفارق الوحيد هو إضافة السمة VALUE، التي وضع فيها النصّ الموجود في مربّع النصّ.

**التفاعل مع تطبيق الإنترنت:**

هذا هو ما يحدث عند تنفيذ تطبيق الإنترنت:

- يتركّب تطبيق الإنترنت من نموذج إنترنت أو أكثر.. ويتكوّن النموذج من كود HTML وكود VB.. وبينما يتمّ تنفيذ كود VB على الخادم، فإنّ كود HTML يتمّ إرساله إلى متصفّح العميل ليقوم بعرض الصفحة على جهازه.. إنّ هذا التقسيم يقدّم للعميل العديد من المزايا: فهو لا يحتاج لشراء التطبيق، ولا يحتاج لإعداده على جهازه، ولا يستهلك مساحة تخزين لديه، ولا يقلق بخصوص تحديثه والإصدارات الجديدة منه.

- يتمّ ترجمة Compile كود التطبيق في أوّل مرّة يتمّ تنفيذه فيها.. هذا هو السبب في البطء النسبيّ للتنفيذ في أوّل مرّة.. بعد ذلك لا يتمّ ترجمة الكود طالما أنّ التطبيق يعمل.

- يقوم VB بإنشاء خليّة Class لتتفاعل مع متصفّح الإنترنت.. هذه الخليّة تُنشأ وتترجم في وقت التنفيذ.

- تقوم هذه الخليّة بقراءة قيم المتغيّرات المرسلة من المتصفّح، وتعالج الاستعلامات المرسلة، ثمّ تُنتج كود HTML وترسله إلى العميل.. معنى هذا أنّك تستطيع تغيير قيم الأدوات على صفحات الإنترنت، كما تستطيع إنشاء صفحات جديدة بها القيم الجديدة للأدوات وإرسالها للعميل.

- ويجب أن تضع في اعتبارك أنّ الكود يتمّ تنفيذه على الخادم Server، بمعنى أنّه لا يرى القيم الموجودة في الأدوات إلا حينما يرسلها له المستخدم (بضغط زرّ ما).. إذن فالتطبيق لا يعمل على الخادم طوال الوقت، وإنّما يتمّ تشغيل الكود اللازم عندما يطلب العميل ذلك.. وتسمّى عمليّة إرسال كلّ قيم الأدوات إلى الخادم بإرجاع البيانات Postback.

**أدوات الإنترنت Web Controls**

* [**هي أدوات حقيقيّة؟:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\6-%20أدوات%20الإنترنت\1-%20هي%20أدوات%20حقيقيّة.htm)
* [**أدوات التحقّق من الصحّة Validation Controls:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\6-%20أدوات%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة.htm)
* [**أداة التقويم Calendar Control:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\6-%20أدوات%20الإنترنت\3-%20أداة%20التقويم.htm)
* [**الشجرة TreeView Control:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\6-%20أدوات%20الإنترنت\4-%20الشجرة.htm)
* [**أداة الملفّ File Control:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\6-%20أدوات%20الإنترنت\5-%20أداة%20الملفّ.htm)

**هي أدوات حقيقيّة:**

قلنا مرارا إنّ كود HTML هو المسئول عن رسم الصفحات على متصفّح الإنترنت.. إذن ما هي حكاية أدوات الإنترنت هذه؟

في الواقع، هي ليست أدوات حقيقيّة، إذ يتمّ تحويلها إلى كود HTML ليتمكّن المتصفّح من عرضها على الصفحة.

أعتقد أنّك قد بدأت تألف اللعبة.. إنّ ميكروسوفت قد وفّرت لك بيئة افتراضيّة تشبه تلك التي اعتدت التعامل معها في تطبيقات الويندوز، ولكنّها في الحقيقة تقوم بإنتاج كود HTML التقليديّ، لأنّ هذه هي الطريقة التي تفهمها متصفّحات الإنترنت.

ولكن كيف يتمّ إنتاج كود HTML؟

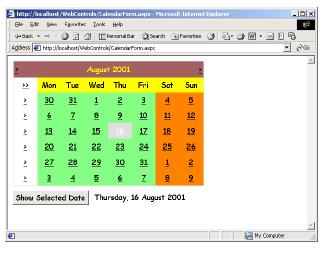
عند تشغيل التطبيق، يتمّ إنشاء ملفّ DLL يحمل اسم التطبيق، يوضع في المجلّد Debug أو المجلّد Bin.. هذا الملفّ يحتوي على الخليّة Class التي تقوم بإنتاج كود HTML المناسب.

تعال نتعرّف على أهمّ أدوات الإنترنت.. ولكن نظرا لأنّك تجيد بالفعل التعامل مع الكثير منها، مثل مربّع النصّ واللافتة، فستقتصر مناقشتنا هنا حول الأدوات الجديدة.

**أداة التقويم Calendar Control:**

تقوم هذه الأداة بنفس الوظيفة التي تقوم بها الأداة MonthCalendar في تطبيقات الويندوز، فهي تعرض تقويم الشهر الحالي، وتتيح للمستخدم اختيار اليوم أو الأسبوع بالفأرة.

وبينما تبدو لك هذه الأداة على نموذج الإنترنت كأداة مستقلّة، فإنها تبدو على صفحة الإنترنت في صورة جدول.



وباستخدام خاصيّة "طراز الاختيار" SelectionMode يمكنك أن تسمح للمستخدم باختيار مجموعة من الأيّام بدلا من يوم واحد.. وتأخذ هذه الخاصيّة القيم التالية:

|  |  |
| --- | --- |
| None | لا يمكن للمستخدم أن يختار أيّ تاريخ. |
| Day | يمكن للمستخدم أن يختار يوما واحدا فقط من التقويم. |
| DayWeek | يمكن للمستخدم أن يختار يوما واحدا فقط أو أسبوعا بكامله من التقويم. |
| DayWeekMonth | يمكن للمستخدم أن يختار يوما واحدا فقط أو أسبوعا بكامله أو شهرا كاملا من التقويم. |

ولكي تتفاعل مع اختيار المستخدم للتاريخ، فإنّ الأداة تطلق الحدث SelectionChanged كلّما تغيّر التاريخ المحدّد بها (بواسطة المستخدم وليس من الكود).

ولكي تحصل على معلومات حول التاريخ المحدّد، استخدم الخاصيتين التاليتين:

|  |  |
| --- | --- |
| SelectedDate | اليوم الذي اختاره المستخدم في الأداة.. فإذا اختار المستخدم مجموعة من الأيّام، فإنّ هذه الخاصيّة تمثّل اليوم الأوّل منها. |
| SelectedDates | مجموعة Collection تمثّل كلّ الأيّام المحدّدة في الأداة.. وأعتقد أنّك خبير بما يكفي في التعامل مع المجموعات Collections. |

ولكي تتحكّم في مظهر التقويم، لديك مجموعة من الخصائص من نوع الكائن "طراز عنصر الجدول" TableItemStyle، وهي:

|  |  |
| --- | --- |
| DayStyle | طراز الأيّام في الشهر الحالي. |
| OtherMonthDayStyle | طراز أيّام الشهور الأخرى. |
| DayHeaderStyle | طراز الصفّ الذي يحوي أسماء أيّام الأسبوع. |
| NextPrevStyle | طراز الوصلتين اللتين تسمحان لك باختيار الشهرين السابق والتالي. |
| SelectedDayStyle | طراز اليوم المحدّد. |
| SelectorStyle | طراز العمود الأيسر، الذي يظهر عندما تسمح للمستخدم باختيار أسبوع بأكمله. |
| TitleStyle | طراز عنوان التقويم. |
| TodayDayStyle | طراز اليوم الحاليّ. |
| WeekendDayStyle | طراز يوم عطلة نهاية الأسبوع. |

ولكن بم تفيدنا هذه الخصائص؟

قلنا إنّ هذه الخصائص كائنات من النوع TableItemStyle، وبهذا فهي تمتلك الخصائص التالية:

|  |  |
| --- | --- |
| BackColor | لون الخلفيّة. |
| ForeColor | لون النصّ. |
| BorderColor | لون الحافّة. |
| BorderStyle | طراز الحافّة. |
| BorderWidth | عرض الحافّة. |
| Font | خطّ النصّ. |
| Height | ارتفاع العنصر. |
| Width | عرض العنصر. |
| HorizontalAlign | المحاذاة الأفقيّة. |
| VerticalAlign | المحاذاة الرأسيّة. |
| Wrap | التفاف السطر. |

ممّا يتيح لك تحكما كبيرا في تنسيق وأداء أداة التقويم.

**أدوات التحقّق من الصحّة Validation Controls:**

تمنحك اللغة بعض الأدوات، لتساعدك في التحقّق من صحّة البيانات التي يُدخلها المستخدم، حيث تعرض بعض هذه الأدوات رسالة خطأ للمستخدم بمجرّد مغادرته لمربّع النصّ، بينما يعرض بعضها رسالة الخطإ عند محاولة ضغط أيّ زر.. وطبعا لو كانت البيانات غير صحيحة، لا يتمّ تنفيذ كود هذا الزرّ.

* [**المتحقّق من الخانات المطلوبة RequiredFieldValidator:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\1-%20المتحقّق%20من%20الخانات%20المطلوبة.htm)
* [**المتحقّق من المقارنة CompareValidator:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\2-%20المتحقّق%20من%20المقارنة.htm)
* [**المتحقّق من النطاق RangeValidator:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\3-%20المتحقّق%20من%20النطاق.htm)
* [**متحقّق التعبير النمطيّ RegularExpressionValidator:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\4-%20متحقّق%20التعبير%20النمطيّ.htm)
* [**المتحقّق المخصّص CustomValidator:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\5-%20المتحقّق%20المخصّص.htm)
* [**ملخص التحقّق من الصحّة ValidationSummary:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\6-%20ملخص%20التحقّق%20من%20الصحّة.htm)
* [**بعض الخصائص المشتركة:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\7-%20بعض%20الخصائص%20المشتركة.htm)
* [**مثال عمليّ:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\8-%20مثال%20عمليّ.htm)
* [**هل توجد أدوات ويندوز للتحقّق من الصحّة؟:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\9-%20هل%20توجد%20أدوات%20ويندوز%20للتحقّق%20من%20الصحّة.htm)
* [**ولكن.. كيف تعمل هذه الأدوات؟**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\2-%20أدوات%20التحقّق%20من%20الصحّة\10-%20ولكن.htm)

**المتحقّق من الخانات المطلوبة RequiredFieldValidator:**

للتأكّد من أنّ الأدوات بها قيم ولم تترك فارغة.

فمثلا: إذا أردت أن تتأكّد أنّ أحد مربّعات النصّ ليس فارغا، فضع نسخة من الأداة RequiredFieldValidator على النموذج، واربطها بمربّع النص هذا، واكتب في خاصيّة Text الخاصّة بها "لا يمكنك ترك هذه الخانة فارغة".

**المتحقّق من المقارنة CompareValidator:**

لمقارنة قيم بعض الأدوات بقيمة معيّنة (سواء أكانت ثابتا أم كانت قيمة أداة أخرى أم كانت قيمة حقل في قاعدة بيانات).

وتمتلك الأداة CompareValidator خاصيّة ControlToCompare لتسمح لك باختيار أداة أخرى لتقارن بقيمتها قيمة الأداة التي تتحقّق منها، وخاصيّة ValueToCompare لتسمح لك بتحديد القيمة التي تريد المقارنة بها.. أهمّ شيء هو تحديد نوع المقارنة (أكبر من، أصغر من، يساوي، ... إلخ).. هذا ما تتيحه لك خاصيّة المعامل Operator.

**المتحقّق من النطاق RangeValidator:**

للتأكّد من أنّ القيمة المدخلة في الأداة محصورة في نطاق معيّن (بين قيمتين معيّنتين).

وتمتلك الأداة RangeValidator الخاصيّتين "أصغر قيمة" MinimumValue، و"أكبر قيمة" MaximumValue، وذلك لتحدي المجال الذي تريد أن تقع القيمة فيه.

**متحقّق التعبير النمطيّ RegularExpressionValidator:**

تقارن القيمة الموجودة في الأداة بصيغة معيّنة.. هذا الصيغة يتمّ تكوينها تبعا لقواعد محرّك التعبير النمطيّ Regular Expression Engine، وهو موضوع كبير، سنكتفي منه بذكر هذه القواعد:

- النقطة "." تمثّل حرفا واحدا (لا يمكن أن يكون رقما).

- العلامة "\*" تمثّل أيّ عدد من الحرف (حتّى لو كان هذا العدد صفرا).

- \d تمثّل رقما (من 0 إلى 9).

- [1-4] تمثّل أيّ رقم بدءا من 1 إلى 4.. طبعا هذا مثال.. ويمكنك أن تستخدم أيّ رقمين بدلا من 1، 4.

- [A-Z] تمثّل أيّ حرف كبير Capital بين حرفي A و Z (بما في ذلك هذان الحرفان).

- [a-e] تمثّل أيّ حرف صغير Small بين من a إلى e.

- [ا-ت] تمثّل أيّ حرف من الألف إلى التاء.

مثال:

النصّ ABC99 يتوافق مع الصيغة [...\d\d] (ثلاثة حروف ورقمان).. بينما النصّ 9AA99 لا يتوافق معها، لأنّه يبدأ برقم.

ولكن.. ماذا لو أردت أن تكوّن صيغة متكرّرة؟.. مثلا، كيف نكتب صيغة تحتوي على خمس أرقام كلّ منها محصور بين 2 و 6؟.. أوّل ما سيجول بخاطرك هو أن تكتبها كالتالي:

[2-6][2-6][2-6][2-6][2-6]

ماشي.. ولكن ماذا لو لم تكن خمسة أرقام؟.. ماذا لو كانت 20 رقما مثلا؟!

هنا يجدر بك استخدام الصيغة التالية:

[2-6]{5}

إن الرقم 5 بين القوسين المجعّدين يعنى أنّ الصيغة التي تسبقه مكرّرة لخمس مرّات.. سلسٌ جدّا إذن أن تضع بدلا من الرقم 5 أيّ رقم آخر: 10، 20، .... إلخ.

**المتحقّق المخصّص CustomValidator:**

تمنحك القدرة على تكوين قاعدة التحقّق من الخطإ الخاصّة بك.. وإن كان هذا يعني أنّك ستكتب بعض الكود في أحداث الأدوات التي تريد التحقق من قيمها.

**ملخص التحقّق من الصحّة ValidationSummary:**

عند ضغط أيّ زرّ، تعرض هذه الأداة تلقائيّا، ملخّصا لكلّ الأخطاء التي تمّ اكتشافها بواسطة باقي أدوات التحقّق من الصحّة.. ويمكنك اعتبار هذه الأداة مجرّد لافتة تعرض رسائل الخطإ، وإن كنتَ لا تحتاج لكتابة أيّ كود لها.

**بعض الخصائص المشتركة:**

**ControlToValidate:**

استخدم هذه الخاصيّة لكي تحدّد لأداة التحقٌّ من الصحّة، الأداة التي ستتحقّق من صحّتها، حيث تضع فيها اسم الأداة المعنيّة (ستظهر لك أسماء الأدوات الموجودة على النموذج في قائمة منسدلة في خانة القيمة لهذه الخاصيّة في نافذة الخصائص).. ولا مانع من أن تربط أكثر من أداة تحقّق من الصحّة بأداة واحدة.

**Text:**

تمثّل النصّ الذي ستعرضه أداة التحقّق من الصحّة لو لم تكن القيمة صالحة.

**رسالة الخطإ ErrorMessage:**

قيمة هذه الخاصيّة هي التي تعرضها الأداة ValidationSummary.. ويمكنك أن تجعل لخاصية ErrorMessage نفس نصّ خاصيّة Text.

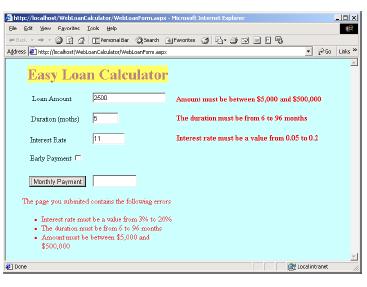
**مثال عمليّ:**

مللت الكلام المجرّد.. هه؟

حسنا.. تعال نطوّر تطبيق WebLoanCalculator الذي أنشأناه مبكّرا في [الفصل الثاني](file:///C:\Users\Administrator\01-%20أساسيّات%20اللغة\02-%20المشاريع\3-%20تشغيل%20محاسب%20القروض%20على%20الإنترنت.htm) من هذا الكتاب، بحيث نضيف إليه أدوات التحقّق من الصحّة.. سنسمّى التطبيق المعدّل WebLoanCalculator1، وستجده في مجلّد برامج هذا الفصل.

وسواء نسخت تطبيق WebLoanCalculator من برامج الفصل الثاني وطوّرته، أو قمت بإنشاء مشروع جديد، في كلتا الحالتين لا تنسَ أن تضع المشروع في مجلّد تطبيقات الإنترنت الافتراضيّ على جهازك (وهو في الغالب C:\Inetpub\wwwroot).

ضع ثلاث نسخ من الأداة RequiredFieldValidator على النموذج، بحيث تكون كلّ منها مجاورةً لمربّع نصّ، كما في الصورة التالية:



اجعل خاصيّة ControlToValidate لكلّ أداة تشير إلى مربّع النصّ المجاور لها.

ثمّ أضف للنموذج ثلاث نسخ من الأداة RangeValidator، وضع كلا منها بحيث تكون منطبقةً على إحدى أدوات RequiredFieldValidator.. غيّر خصائص هذه الأدوات تبعا للجدول التالي:

|  |  |  |  |
| --- | --- | --- | --- |
| الخاصيّة | RangeValidator1 | RangeValidator2 | RangeValidator3 |
| ControlToValidate | txtAmount | txtDuration | txtInterest |
| MinimumValue | 5000 | 6 | 0.05 |
| MaximumValue | 500000 | 96 | 0.2 |
| Text | مقدار القرض يجب أن ينحصر بين 5000 و 500000 جنيه. | مدّة القرض يجب أن تنحصر بين 6 شهور و 96 شهرا. | معدّل الفائدة (الربا) يجب أن ينحصر بين 0.05 و 0.2 . |
| ErrorMessage | مقدار القرض يجب أن ينحصر بين 5000 و 500000 جنيه. | مدّة القرض يجب أن تنحصر بين 6 شهور و 96 شهرا. | معدّل الفائدة (الربا) يجب أن ينحصر بين 0.05 و 0.2 . |

الآن يمكنك أن تجرّب التطبيق، محاولا كتابة قيم صالحة وأخرى غير صالحة.

**هل توجد أدوات ويندوز للتحقّق من الصحّة؟:**

رائعة هي أدوات التحقّق من الصحّة.. أليس كذلك؟

أعتقد أنّك الآن تتمنّى لو كانت هناك مثل هذه الأدوات في نماذج الويندوز، فهي توفّر الكثيرَ من الوقت والمجهود.

اطمئنّ.. هناك فعلا أداة تفعل ذلك في تطبيقات الويندوز، وهي الأداة: ErrorProvider.. وإن كنت تحتاج لكتابة بعض الكود للتعامل مع هذه الأداة، فهي لا تعمل آليا مثل هذه الأدوات (أشبه ما يكون بالأداة CustomValidator.. ربّما تفهم من هذا لماذا لا يوجد منها إلا نوع واحد فقط، حيث إنّك تستطيع أن تكتب في الكود كلّ الشروط التي تريد التحقّق منها، ثمّ تستخدم الوسيلة SetError، التي تمنحها معاملين: اسم الأداة التي سيتمّ عرض أيقونة الخطإ بجوارها، والنصّ الذي سيُعرض كتلميح للمستخدم ToolTip حينما يحلّق بالفأرة فوق أيقونة الخطإ:

**ErrorProvider1.SetError(TextBox1, "لا تترك هذا المربّع فارغا")**

وسأترك لك استخدام ملفات المساعدة الخاصة باللغة لاستكشاف قدرات هذه الأداة الرائعة.

**ولكن.. كيف تعمل هذه الأدوات؟**

في الواقع، إنّ VB يقوم بإنتاج كود javascript داخل صفحة الإنترنت.

ستتحفّز منتصبا في جلستك، وتسألني:

- ما موضوع javascript هذا؟

- إصدار خاصّ من لغة برمجة تدعى Java.. هذه اللغة تستخدم قواعد الكود الخاصّة بـ C++، ولكنّها أبسط منها بكثير، ولاقت نجاحا وشهرةً كبيرين، خاصّةً فيما يتعلّق بالإنترنت، بسبب الإمكانيّات الكبيرة التي تمنحها.

- كأنّك تدعوني لتعلّمها!

- في الواقع لا.. فقد قامت ميكروسوفت باللازم، واقتبست لـ VS.Net معظم إمكانيّات هذه اللغة ومعظم طرائقها.. وليست هي فحسب.. يمكنك أن تقول إنّ ميكروسوفت قد اقتبست لـ VS.NET كلّ التسهيلات الموجودة في لغات البرمجة الأخرى، مثل C Builder و Delphi وغيرهما.. وعموما إنّ VS.NET تكتب كود javascript.Net، ومؤخّرا صدرت لغةٌ جديدة لتنضمّ لعائلة VS.NET وهي J# التي هي عبارة عن Visual Java .NET.. لن تخرج عن نطاق VS.NET إذن!

- جميل.. ولكنّي ما زلت لم أفهم موضوع الـ Scripts هذا؟

- الفكرة بسيطة.. تعرف أنّ هناك أدوات HTML مثل مربّعات النصوص واللافتات والأزرار.. طبيعيّ جدّا إذن أن تكون هناك وسيلة لكتابة الكود لهذه الأدوات.. هنا برزت javascript، ثمّ تلتها VBScript.

- معنى هذا أنّ صفحة HTML العاديّة يمكن أن تحتوي على كود.

- أجل.

- إذن ما الفرق بين صفحات ASP وصفحات HTML؟

- بالنسبة للقدرات، لا فارق (وإن كان هذا الفارق يظهر مع ASP.Net لأنّها تستخدم VB.Net و C# بكامل إمكانيّاتهما تقريبا).. يمكنك أن تنشئ صفحتين متماثلتين في المظهر تماما وتقومان بنفس الوظيفة، إحداهما بلغة HTML مضمّنا بها كود VBScript أو JavaScript، والأخرى مكتوبة كصفحة ASP.

- فلماذا إذن لم ننشئ صفحة محاسب القروض كصفحة HTML من البداية؟.. ما الفارق بينهما؟

- الفارق هو أنّ الكود الذي تكتبه في صفحة ASP سيعمل على الخادم.. إنّ هذا سيمنحك القدرة على تحديثه في أيّ وقت، كما سيعزل هذا الكود عن المستخدم، ممّا سيحفظ لك حقوقك كمبرمج ويضمن ألا يطّلع أحد على هذا الكود.. أمّا بالنسبة للكود الذي تكتبه في صفحة HTML فهو يمتاز بسرعة تنفيذه، لأنّه يعمل على جهاز المستخدم، ولكنّه لن يكون محميّا، فبإمكان أيّ شخص أن يفتحه ويقرأه وينقله إلى أيّ مكان لو أراد.. والآن نحن نستفيد من ميزات الطريقتين معا.. سنكتب الكود الذي نريد حمايته بـ ASP.Net ونجعله يعمل على الخادم، ونكتب الكود الذي لا يقلقنا أن يطّلع عليه المستخدم ـ مثل كود التحقّق من الصحّة ـ بلغة JavaScript، وبهذا نضمن سرعة تنفيذه.

والآن لو عرضت الصفحة في متصفّح الإنترنت، وفتحت كودها، لوجدت قرب نهايته المقطع التالي:

<SCRIPT LANGUAGE="javascript">

<!--

var Page\_ValidationActive = false;

if (typeof(clientInformation) != "undefined" &&

clientInformation.appName.indexOf("Explorer") != -1) {

if (typeof(Page\_ValidationVer) == "undefined")

alert("Unable to find script library

'/aspnet\_client/system\_web/1\_0\_2914\_16/WebUIValidation.js'. Try placing this file manually, or reinstall by running 'aspnet\_regiis -c'.");

else if (Page\_ValidationVer != "121")

alert("This page uses an incorrect version of WebUIValidation.js. The page expects version 121. The script library is " + Page\_ValidationVer + ".");

else

ValidatorOnLoad();

}

function ValidatorOnSubmit() {

if (Page\_ValidationActive) {

ValidatorCommonOnSubmit();

}

}

// -->

</SCRIPT>

هذا هو كود JavaScript.. وبالطبع لن نشرحه هنا، فهذا ليس موضوعنا.

ما يهمّ هنا هو أن تلاحظ أنّ الكود يتأكّد من نوع المتصفّح الذي يعمل عليه.. فإن كان Internet Explorer يتمّ تنفيذ الكود، وإن كان متصفّحا آخر، لا يتمّ تنفيذ هذا الكود.. إنّ من المتوقّع أن يتغيّر هذا التصّرف في المستقبل القريب، عندما تطوّر معظم الشركات متصفّحاتها.

ولكن.. هل يعني هذا أنّنا سنضطرّ لكتابة كود آخر للتحقّق من الصحّة للتعامل مع المتصفّحات بخلاف Internet Explorer؟

لحسن الحظّ لا.. إنّ كلّ المطلوب منك هو وضع أدوات التحقٌّ من الصحّة على النموذج.. فإذا كان المتصفّح هو Internet Explorer، فإنّ كود التحقٌّ من الصحّة سيتمّ على جهاز المستخدم.. وإن كان غير ذلك، فسيتمّ إرسال الصفحة للخادم، وهناك سيتمّ التحقّق من صحّة البيانات المدخلة.

**الشجرة TreeView Control:**

رغم أنّها لا تقارن بالشجرة التي تعاملنا معها في [تطبيقات الويندوز](file:///C:\Users\Administrator\Desktop\03-%20مواضيع%20متقدّمة\11-%20أدوات%20متقدّمة\3-%20الشجرة.htm)، إلا إنّ هذه الأداة ستيسّر عليك الكثير من العمل.

وأوّل ما يجب أن تضعه في ذهنك، هو أنّ كلّ حدث يؤدّيه المستخدم مع هذه الأداة سيتمّ إرساله إلى الخادم مباشرة، ممّا قد يعني بطء استجابتها في كثير من الأحيان.

لا تقلقنَّ.. يمكنك تغيير خاصيّة "الإرجاع التلقائيّ" AutoPostBack لهذه الأداة إلى False، وإن كان هذا سيضع على عاتقك وضع زرّ يقوم بإرسال التغييرات التي أحدثها المستخدم إلى الخادم.. لا أعتقد أنّ هذا سيروقك!.. أنت بالخيار.

عامّةً لن يكون الأمر بهذا السوء لو تمّ فتح الصفحة على متصفّح الإنترنت Internet Explorer، ففي هذه الحالة ستحتوي الصفحة على كود JavaScript ليتولّى مسئوليّة إسدال وتقليص فروع الشجرة، ممّا سيوفّر الكثير من الوقت، لأنّ تنفيذه سيتمّ على جهاز المستخدم.

ولحسن الحظّ، لن تجد اختلافات برمجيّة كبيرةً بين شجرة الإنترنت وشجرة الويندوز.. فمثلا، يمكنك وضع عناصر الشجرة باستخدام خاصيّة Nodes، سواء من نافذة الخصائص أو من الكود ([راجع فصل الأدوات المتقدّمة](file:///C:\Users\Administrator\Desktop\03-%20مواضيع%20متقدّمة\11-%20أدوات%20متقدّمة\3-%20الشجرة\5-%20إضافة%20العناصر%20للشجرة%20في%20وقت%20التصميم.htm)).

خلافات طفيفةٌ فقط هي التي ستقابلك.. فمثلا يمتلك كلّ فرع Node خاصيّة ID التي تميّزه عما سواه، وخاصيّة NodeData وهي مماثلة لخاصيّة Tag، حيث يمكنك أن تضع بها أيّ بيانات إضافيّة تهمّك.. هذا بالإضافة إلى خاصيّة "طلب العنوان" NavigateUrl، التي تسمح لك بربط كلّ عنصر في الشجرة بعنوان موقع، يتمّ فتحه عند ضغط هذا العنصر.

وكما تعرف، يمكنك أن تستخدم حدث SelectedIndexChanged للتفاعل مع ضغط المستخدم لعناصر الشجرة.. ويمكنك معرفة رقم العنصر المحدّد في الشجرة عن طريق خاصيّة SelectedNodeIndex.. ولكي تعرف العنصر من رقمه، استخدم خاصيّة GetNodeFromIndex.. والتعبير التالي يعيد لك نصّ العنصر المحدّد في الشجرة TV1:

**TV1.GetNodeFromIndex(TV1.SelectedNodeIndex).Text**

**أداة الملفّ File Control:**

هذه الأداة من أدوات HTML، وإن كان من الممكن أن تجعلها تعمل على الخادم، بضغطها بزرّ الفأرة الأيمن، وضغط الأمر Run As Server Control.

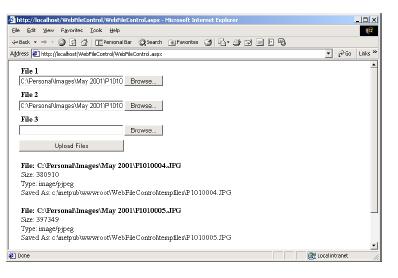
**ملحوظة:**

طبعا أصبحت تدرك الفارق بين أدوات الإنترنت Web Controls وأدوات HTML: فالأولى أدوات تعمل على الخادم وترسل أحداثها له، ممّا يعني أنّ الكود الذي تكتبه لها محميٌّ وبعيد عن متناول المستخدم.. أمّا أدوات HTML فهي تعمل على جهاز المستخدم، ولن تستطيع كتابة كود VB.Net لها.. تستطيع فقط استخدام JavaScript أو ما شابهها.. ورغم أنّك قد لا تحبّذ هذا، إلا إنّه قد يكون ضروريّا لتطوير أداء صفحاتك، خاصّةً في المواضع التي لا تهتمّ فيها بحماية الكود.

ولكن ما هو دور أداة الملفّات؟

نفس دور أداة الويندوز OpenFileDialog، حيث تتيح للمستخدم اختيار ملفّ من على جهازه.. بعد هذا يمكنك تحميل هذا الملفّ إلى الخادم Upload، بكتابة الكود المناسب في أحد الأزرار.. فإذا تمّ هذا، صار بإمكانك كتابة الكود الذي يتعامل مع هذا الملفّ من على الخادم، عن طريق خاصيّة "الملفّ المرسل" FilePosted، التي تمنحها لك أداة الملفّات.

وتبدو هذه الأداة على الصفحة كزرّ ومربّع نصّ.. الزرّ يعرض مربّع حوار فتح ملفّ، ومربّع النصّ يستقبل اسم الملفّ المحدّد.



ويمكنك التدريب على استخدام هذه الأداة في تطبيق WebFileControl في مجلّد برامج هذا الفصل.

لاحظ أنّك تحتاج لتعديل نصّ التنسيق الخاصّ بالنموذج <FORM> tag بإضافة السمة التالية يدويّا:

**ENCTYPE="multipart/form-data"**

إنّ هذا يخبر النموذج بأنّ عليه تحميل upload كلّ أنواع الملفّات إلى الخادم.. هكذا يجب أن يبدو نصّ تنسيق النموذج:

**<FORM ID="Form1" ENCTYPE="multipart/form-data" RUNAT="server">**

تعال نرى كيف نتعامل مع أحد الملفّات التي يختارها المستخدم:

في البداية يجب أن نحدّد الموضع الذي سنحمّل الملفّ إليه على الخادم.. هذا الموضع نستطيع أن نحصل عليه باستخدام الوسيلة "خريطة المسار" MapPath الخاصّة بالكائن [Server](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\8-%20ASP%20كائنات\4-%20كائن%20الخادم.htm)، حيث سنستخدمها لإنشاء مجلّد احتياطيّ اسمه tempfiles في المجلّد الذي يحتوي على تطبيقك على الخادم.. هذا هو الكود الذي يرجع لك مسار هذا المجلّد الجديد:

**Dim UploadFolder As String = Server.MapPath("tempfiles")**

الآن سنشرع في استخدام خاصيّة PostedFile للتعامل مع الملفّ الذي حدّده المستخدم.. هذه الخاصيّة تحتوي على المعلومات اللازمة عن الملفّ، مثل اسمه، حجمه، نوعه... إلخ.

وطبعا يجب أن تتأكّد أولا من أنّ المستخدم قد اختار ملفّا بالفعل.. وفي الكود التالي سنكتب اسم الملفّ حجمه ونوعه ومساره في لافتة:

**If File1.PostedFile.FileName <> "" Then**

**Dim fileName As String = \_**

**System.IO.Path.GetFileName(File1.PostedFile.FileName)**

**Dim filePath As String = UploadFolder & "\" & fileName**

**Label4.Text = "<B>الملفّ: " & File1.PostedFile.FileName & "</B>"**

**Label4.Text &= "<BR>حجمه: " & File1.PostedFile.ContentLength**

**Label4.Text &= "<BR>نوعه: " & File1.PostedFile.ContentType**

**Label4.Text &= "<BR>تم حفظه باسم: " & filePath**

**End If**

ولكن.. هل تمّ حفظ الملفّ بالفعل على الخادم؟

نعم تمّ ذلك آليّا، ولكنّ هذا الملفّ تمّ وضعه في مجلّد ملفّات الإنترنت المؤقّتة، وليس في المجلّد الذي أنشأناه، ممّا يعني أنّنا لن نستطيع التحكّم فيه.

إذن ما الحل؟

بسيطة.. استخدم الوسيلة SaveAs الخاصّة بالكائن PostedFile، لحفظ الملفّ في المسار الذي تريده على الخادم، كالتالي:

**File1.PostedFile.SaveAs(filePath)**

الآن يمكنك أن تفتح هذا الملفّ وتتعامل معه كيفما يحلو لك.

**إبقاء الحالة Maintaining State:**

لقد وصلنا الآن للمرحلة التي يجب أن نتساءل عندها: كيف يمكن الاحتفاظ بقيم المتغيّرات الخاصّة بالصفحة، إذا كان الكود لا يعمل على الخادم إلا لفترات محدودة ثمّ يتوقّف، ممّا يعني محو كلّ البيانات بين الاستعلامات وبين عمليات التحديث لكلّ صفحة؟

إنّ المعلومات التي تريد إبقاءها على صفحة أو أكثر تسمّى "حالة التطبيق" Application's state.

افترض أنّ المستخدم ضغط زرّا ما ليعالج البيانات التي أدخلها في بعض الأدوات.. ما هي إلا لحظات وتظهر النتائج في الأدوات المخصّصة لذلك.. ولكن ماذا عن البيانات التي أدخلها في أدوات الإدخال؟.. ستزول هذه البيانات وتبدو الأدوات فارغة!

وسيحدث نفس الأمر لو انتقل المستخدم من صفحة لأخرى، ثمّ عاد للصفحة الأولى.. لن يجد أيّ بيانات ممّا أدخلها، وكأنّه فتح الصفحة لأّول مرّة.

إذا كان هذا حال الأدوات، فما بالنا بالمتغيّرات؟

طبعا لن تحتفظ بقيمها بين الصفحات المختلفة، أو بين التحديثات المختلفة لنفس الصفحة!

لا أعتقد أنّ هذا ما تريده.. اطمئنّ، فلدينا وفرة من الحلول:

1- جعل خاصيّة "تمكين عرض الحالة" EnableViewState الخاصّة بكلّ أداة بالقيمة True.. ولحسن الحظّ هذه هي القيمة الافتراضيّة.. وبهذا ستحتفظ الأداة بالقيمة التي تعرضها إلى أن يتمّ إغلاق التطبيق نهائيّا.

2- استخدام الحقول الخفيّة Hidden fields لتخزين قيم المتغيّرات.. افتح صندوق الأدوات، واضغط شريط HTML.. ستجد ضمن أدواته الأداة Hidden.. ضع نسخةً منها على النموذج.. هذه الأداة لن تظهر على الصفحة عند عرضها على متصفّح الإنترنت.. إنّها فقط مخزن غير معتاد للقيمة التي تريد بقاءها.. استخدم خاصيّة Name لتحديد اسم هذا المتغيّر (وليكن \_\_VIEWSTATE)، وخاصيّة Value لتحديد قيمته.. ولا تنس تحديد ID للأداة، لاستخدامه في البرمجة.. الآن ستظهر الجملة التالية في كود HTML للصفحة:

**<INPUT TYPE="hidden" NAME="\_\_VIEWSTATE" VALUE="dDwtMTMzNTkzMTE2Mjs7Pg==" />**

3- هناك طريقة ثالثة، تقوم بنفس دور الطريقتين السابقتين.. وذلك باستخدام حقيبة الحالة StateBag لتعريف بعض الحقول الخفيّة.. كالتالي:

**ViewState("TempData") = 25**

"ولكن ماذا لو كان هناك متغيّر بنفس هذا الاسم؟".. لا ريب أنّك ستتساءل.

"بالطبع ستتمّ تغيير قيمته القديمة للقيمة الجديدة".. لا ريب أنّني سأجيب.

ويمكن قراءة هذا المتغيّر كالتالي:

**Dim X As Integer = Str(ViewState("TempData"))**

جدير بالذكر أنّ استخدام هذه الطريقة يضيف للصفحة عند إرسالها للمتصفح حقلا خفيّا كوده كالتالي:

**<input type="hidden" name="\_\_VIEWSTATE" value="dO…NnU=" />**

4- جميل.. لكنّ كلّ الحلول التي طرحت حتّى الآن تبقي الحالة عبر صفحة واحدة.. فماذا لو كنا نريد إبقاء الحالة عبر عدّة صفحات؟.. في هذه الحالة يمكن استخدام الطريقة التي يألفها مبرمجو ASP، وهي تخزين البسكويت Cookies.. ولا أعني بهذا أنّ عليك إحضار بعض البسكويت وتخزينه في مكانٍ أمين!!.. إنّني أعنى تخزين حالة التطبيق على جهاز المستخدم (وبهذا يمكن إبقاء بعض البيانات على جهازه، حتّى بعد إغلاق الاتصال).. إنّ الأمريكيّين بارعون للغاية في استخدام الكنايات والتشبيهات، وينحتون مصطلحاتهم بهذه الطريقة التي ترهقنا في الترجمة!.. واضح أنّهم يعنون أنّك تستطيع تخزين بعض المعلومات الخفيفة سريعة الإعداد والاستحضار على جهاز المستخدم.. ولا ريب أنّهم أسموها كذلك حتّى يسهل على المستخدم تقبّل فكرة أنّ صفحات الإنترنت تخزّن المعلومات على جهازه، إذ لا بدّ أنّه سيقلق بخصوص المساحة.. وهذا هو ما يحدث بالفعل، إذ كثيرا ما يقوم المستخدم بإغلاق مخازن البسكويت هذه!!.. أقصد يقوم بعدم تمكينك من تخزين بياناتك على جهازه.. معنى هذا أنّ هذا الحلّ لن يعمل في كلّ الأحيان.. عامّةً سنشرح هذه الـ Cookies بالتفصيل [لاحقا في هذا الفصل](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\9-%20استخدام%20البيانات%20المؤقّتة.htm).

5- كما يمكن إنشاء مجموعة من المتغيّرات عبر كائن الجلسة Session object.. هذا الكائن يتمّ إنشاؤه لكلّ مستخدم جديد يستخدم التطبيق، ويتمّ إنهاؤه عندما ينهي المستخدم اتصاله بالتطبيق نهائيا.. انظر كيف نعرّف المتغيرين Items و UID لكلّ الجلسة ونمنحهما القيمتين 3 و "مجهول":

**Session("Items") = 3**

**Session("UID") = "مجهول"**

لاحظ أنتّ هذه المتغيّرات ستحتفظ بالمعلومات لكلّ جلسة، وليس للصفحة فقط.. ولكن لمستخدم واحد فقط.. بحيث تضيع هذه المعلومات عندما ينهي المستخدم تعامله مع التطبيق.

ولقراءة أيّ متغيّر، يمكنك استخدام كود كهذا:

**Dim ID As String = Session("UID")**

6- ويمكن استخدام معرّف متفرّد Unique ID لكلّ جلسة (اتصال) من المستخدمين Session.. ويمكنك الحصول على هذا المعرّف باستخدام خاصيّة SessionID الخاصّة بالكائن Session.. بعد هذا يمكنك التعامل مع هذا المعرّف عن طريق متغيّر (كائن) من النوع SessionState**.**HttpSessionState، وبهذا يمكنك تمريره بين الخادم والعميل، ليمثّل حالة التطبيق طالما كان العميل متصلا.

7- لعلك لاحظت أنّ كلّ الطرق السابقة تختصّ بحفظ حالة صفحة واحدة، أو بحفظ حالة عميل واحد.. فماذا إذن لو أردنا حفظ الحالة عبر التطبيق كلّه، وللمستخدمين جميعا؟

هنا يمكننا استخدام كائن التطبيق Application لتعريف بعض المتغيّرات، بطريقة مشابهة لتعريف متغيرات كائن الجلسة:

**Application("App\_StartTime") = Now**

ويمكن قراءة هذا المتغيّر كالتالي:

**Dim T As Date = CDate(Application("App\_StartTime"))**

لاحظ أنّ هذا المتغيّر سيحتفظ بقيمته طوال عمل التطبيق.. مع الأخذ في الاعتبار أنّ تطبيق الإنترنت يعمل على الخادم ليل نهار!

ولكن علينا أن نحترس لموقف خطير: هو التسابق من أكثر من مستخدم على قراءة أو تغيير المعلومات المخزنة في المتغيّرات المعرّفة عبر كائن التطبيق.

افترض أنّ لدينا متغيّر عبر التطبيق يسمى X..

وهناك مستخدمان الآن يستخدمان التطبيق..

قام الأوّل بتنفيذ عمليّة معينة، جعلت قيمة X = 5.. وكان كود العمليّة في تطبيقك يحاول أن يضيف 1 على هذه القيمة لإجراء حسابات معيّنة..

ولكن في هذه اللحظة قام المستخدم الثاني بتنفيذ عمليّة غيّرت قيمة X إلى 3.. وبالتالي سيحصل المستخدم الأوّل على نتائج خاطئة!

لهذا لو كنت تخشى من حدوث مثل هذه المشكلة، فعليك إغلاق متغيّرات التطبيق أثناء تعاملك معها من الكود، ثمّ إعادة فتحها بعد انتهاء المهمّة، وذلك كالتالي:

**Application.Lock**

**'……………….**

**Application("X") += 1**

**'……………….**

**Application.Unlock**

**كائنات ASP.NET**

لا أعتقد أنّك تستطيع كتابة التطبيقات التي تريدها، دون أن تتعرّف على الكائنات التالية:

* [**كائن الصفحة Page Object:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\8-%20ASP%20كائنات\1-%20كائن%20الصفحة.htm)
* [**كائن الاستجابة Response Object:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\8-%20ASP%20كائنات\2-%20كائن%20الاستجابة.htm)
* [**كائن الاستعلام Request Object:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\8-%20ASP%20كائنات\3-%20كائن%20الاستعلام.htm)
* [**كائن الخادم Server Object:**](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\8-%20ASP%20كائنات\4-%20كائن%20الخادم.htm)

**كائن الصفحة Page Object:**

يمثّل هذا الكائن الصفحة المعروضة على جهاز العميل.. ولهذا الكائن ـ مثل باقي الكائنات ـ حدثان: حدث التحميل Load، وحدث الإغلاق Unload.

وأهمّ خاصيّة لكائن الصفحة، هي خاصيّة "تمّ إرجاعه" IsPostBack، التي تعطيك القيمة True إذا تمّ إرسال بيانات الصفحة إلى الخادم.. وتفيد هذه الخاصيّة في تحديد إذا ما كانت الصفحة تحتاج لوضع القيم الابتدائيّة في أدواتها أم لا.. وكما اتفقنا سابقا، من المحبّذ الاحتفاظ بمدخلات المستخدم بعد إرسال بياناتها إلى الخادم.

ولتجدنّ الإجراء التالي في معظم تطبيقات ASP.NET:

**Private Sub Page\_Load(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles MyBase.Load**

**If Not Page.IsPostBack Then**

**' الكود اللازم لوضع القيم الابتدائيّة في الأدوات**

**End If**

**End Sub**

**كائن الاستجابة Response Object:**

هذا الكائن مشتقّ من الخليّة HttpResponse.. وهو يمكّنك من إرسال النتائج من الخادم إلى العميل.

**أهمّ الخصائص:**

**تخزين Buffer:**

لو جعلت قيمة هذه الخاصيّة True، يقوم الخادم بتخزين النتائج المطلوبة إلى أن تكتمل، ثمّ يرسلها دفعةً واحدةً إلى العميل.. بخلاف ذلك، يتمّ إرسال النتائج مجزّئةً، كلّما اكتمل منها جزء.. والوضع الأخير في الغالب أفضل، لأنّه يقلّل الوقت الذي ينتظره المستخدم.

**البيانات المؤقّتة Cookies:**

مجموعة Collection تمثّل البيانات التي يخزّنها الخادم على جهاز العميل ـ إن كان العميل يتيح هذه الإمكانيّة.. (وسنناقش ذلك بالتفصيل [لاحقا في هذا الفصل](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\9-%20استخدام%20البيانات%20المؤقّتة.htm)).

**الزوال Expires:**

تقرأ هذه الخاصيّة أو تغيّر المدّة الزمنيّة (بالدقيقة)، التي ستبقى فيها الصفحة مهيأة Cached على جهاز العميل.. إنّ الصفحات المهيّأة لا يتمّ تحميلها للعميل في كلّ مرّة، ممّا يوفّر الكثير من الوقت.

**الزوال المحدّد ExpiresAbsolute:**

مماثلة للخاصيّة السابقة، ولكنّها لا تتعامل بالدقائق، بل بالتاريخ والوقت الذي سيزول بعدهما الصفحة المهيأة من على جهاز العميل.

**هل العميل متّصل IsClientConnected:**

تعيد هذه الخاصيّة True إذا كان العميل ما زال متّصلا بالخادم.. استخدم هذه الخاصيّة في البرامج التي تأخذ وقتا طويلا في التنفيذ، فمن الممكن أن يغلق العميل الصفحة (من فرط الملل!!)، وبهذا يصبح استمرارك في تنفيذ الكود بلا طائل.

**الوسائل:**

**محو المحتوى ClearContent:**

تزيل كلّ المخرجات الموجودة في مخزن المخرجات Output Buffer Stream.

**محو العناوين ClearHeaders:**

تمحو كلّ العناوين Headers من مخزن المخرجات.

**إغلاق Close:**

تغلق الاتصال مع العميل.

**إنهاء End:**

تنهي تنفيذ المهمّة المطلوبة حاليّا، ويتمّ إرسال المخرجات الموجودة في المخزن Buffer للعميل.

**محو Flush:**

ترسل المخرجات الموجودة في المخزن Buffer للعميل.. استخدم هذه الوسيلة لإرسال البيانات للعميل على فترات، إذا استغرق التنفيذ وقتا طويلا.

**إعادة التوجيه Redirect:**

تقبل هذه الوسيلة عنوان أحد المواقع URL، وتصل المستخدم به.. وفي الغالب، تستخدم هذه الوسيلة لعرض الصفحات في التطبيقات التي تتكوّن من أكثر من صفحة.

**كتابة Write:**

استخدم هذه الوسيلة لإضافة نصوص (أو حتّى كائنات) للمخرجات التي سيتمّ إرسالها للعميل.

**كتابة ملفّ WriteFile:**

استخدم هذه الوسيلة لإرسال محتويات ملفّ موجود على الخادم إلى العميل.

**كائن الاستعلام Request Object:**

هذا الكائن مشتقّ من الخليّة HttpRequest.. وهو يسمح لك بالتعامل مع الاستعلامات الواردة من العميل إلى الخادم.. وفيما يلي أهمّ خصائص هذا الكائن:

**مسار التطبيق ApplicationPath:**

تعيد المسار الذي يعمل عليه التطبيق على الخادم.

**طول المحتوي ContentLength:**

طول البيانات المرسلة من العميل (بالبايت).

**البيانات المؤقّتة Cookies:**

تعيد مجموعة Collection تحتوي على البيانات المؤقّتة المرسلة من العميل.

**العنوان الخام RawUrl:**

تعيد العنوان URL المطلوب حاليّا.

**مسار الملفّ FilePath:**

تعيد المسار الافتراضيّ Virtual Path للاستعلام الحاليّ.

**المسار الفعليّ PhysicalPath:**

تعيد مسار الصفحة التي تمثّل العنوان URL المطلوب.

**هل الاتصال مؤمّن IsSecureConnection:**

تعيد هذه الخاصيّة True إذا كان الاتصال يستخدم وصلات مؤمّنة secure sockets، مثل HTTPS.

**متغيّرات الخادم ServerVariables:**

تعيد مجموعة Collection تحتوي على متغيّرات الخادم.

**حجم البيانات TotalBytes:**

تعيد المجموع الكلّيّ للبيانات الموجودة في الاستعلام (بالبايت).

**العنوان Url:**

تعيد معلومات عن العنوان المطلوب حاليّا.

**العنوان السابق UrlReferrer:**

تعيد العنوان السابق، الذي ضغط من خلاله المستخدم الوصلة الفائقة التي فتحت العنوان الحاليّ.

**عنوان المضيف UserHostAddress:**

تعيد معرّف عنوان العميل IP.

**اسم المضيف UserHostName:**

تعيد الاسم DNS name الخاصّ بالعميل.

**لغة العميل UserLanguages:**

تعيد مصفوفة، تحتوي على أسماء اللغات المفضّلة لدى العميل.

**المتصفّح Browser:**

هذه الخاصيّة عبارة عن كائن يمنحك المعلومات اللازمة عن المتصفّح الذي يعمل على جهاز العميل.. ومن خصائص هذا الكائن:

|  |  |
| --- | --- |
| AOL | تعيد True إذا كان المتصفّح America Online. |
| BackgroundSounds | تعيد True إذا كان المتصفّح يسمح بإصدار الأصوات في خلفيّة التشغيل. |
| Beta | تعيد True إذا كان المتصفّح إصدارا تجريبيّا. |
| Cookies | تعيد True إذا كان المتصفّح يستطيع تخزين البيانات المؤقّتة على جهاز العميل.. ولا علاقة لقيمة هذه الخاصيّة بتفعيل المستخدم أو إغلاقه لإمكانيّة تخزين البيانات المؤقّتة Cookies على جهازه. |
| Crawler | تعيد True إذا كان المتصفّح محرّك بحث Crawler Search Engine. |
| Frames | تعيد True إذا كان جهاز العميل يستطيع التعامل مع إطارات HTML. |
| JavaApplets | تعيد True إذا كان المتصفّح يستطيع التعامل مع Java Applets. |
| VBScript | تعيد True إذا كان المتصفّح يستطيع التعامل مع الكود المكتوب بلغة VBScript. |
| JavaScript | تعيد True إذا كان المتصفّح يستطيع التعامل مع الكود المكتوب بلغة JavaScript. |
| MajorVersion | رقم الإصدار الرئيسيّ للمتصفّح. |
| MinorVersion | رقم الإصدار الفرعيّ للمتصفّح. |
| Platform | اسم نظام التشغيل الذي يعمل عليه المتصفّح. |
| Tables | تعيد True إذا كان المتصفّح يستطيع التعامل مع جداول HTML. |

**كائن الخادم Server Object:**

يمنحك هذا الكائن الكثير من المعلومات المهمّة عن الخادم.. فمثلا، يمنحك خاصيّة "اسم الجهاز" MachineName، التي تعيد اسم الجهاز الذي يعمل عليه الخادم.

**وسائل كائن الخادم:**

**GetLastError:**

تعيد آخر استثناء Exception حدث أثناء التنفيذ.

**ClearError:**

تمحو بيانات آخر خطإ قد حدث.

**خريطة المسارات MapPath:**

تعيد المسار الكامل للملفّ المرسل لها كمعامل.. ولقد رأينا كيفيّة استخدامها [من قبل](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\6-%20أدوات%20الإنترنت\5-%20أداة%20الملفّ.htm).

**تنفيذ Execute:**

تنفّذ كود الصفحة، التي تمّ إرسال عنوانها كمعامل، دون أن تعرض هذه الصفحة.. ويتمّ عرض ناتج الصفحة الجديدة على الصفحة القديمة ([كما سنرى لاحقا](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\10-%20التعامل%20مع%20أكثر%20من%20نموذج.htm)).

**انتقال Transfer:**

تلغي تنفيذ الصفحة الحاليّة وتنتقل إلى صفحة أخرى، دون أن تتمّ العودة للصفحة السابقة.

**تشفير العنوان UrlEncode:**

لو جرّبت الكود التالي:

**X = Server.UrlEncode("who am I?")**

فستحصل على النصّ:

who+am+I%3f

**فكّ تشفير العنوان UrlDecode:**

تفكّ تشفير النصّ الذي تحصل عليه من الوسيلة السابقة.

**HtmlEncode:**

تعمل على تشفير كود HTML المرسل لها كمعامل.

**HtmlDecode:**

تعمل على فكّ تشفير النصّ الناتج من الوسيلة السابقة.

**تشفير مسار العنوان UrlPathEncode:**

تشفّر الجزء الخاصّ بالمسار في عنوان الموقع.

**استخدام البيانات المؤقّتة Cookies:**

قلنا إنّ بإمكانك التعامل مع مجموعة البيانات المؤقّتة Cookies collection الموجودة حاليّا على جهاز المستخدم، عبر خاصيّة Cookies، التي يمنحها لك كائنا الاستعلام Request والاستجابة Response.

وكأيّ مجموعة Collection، يمكنك إضافة المزيد من البيانات المؤقّتة لجهاز المستخدم.. ابدأ أولا بتعريف هذه المجموعة كالتالي:

**Dim cookie As New HttpCookie("User")**

حيث User هو المفتاح الذي تشير به إلى هذه البيانات المؤقّتة (يمكن أن يكون أيّ نصّ آخر).

الآن يمكنك استخدام خصائص كائن البيانات المؤقّتة التالية:

|  |  |
| --- | --- |
| Value | البيانات المؤقّتة التي ستخزّنها على جهاز المستخدم. |
| Values | مجموعة Collection تستخدم في حالة وضع بيانات عديدة في نفس الكائن.. في هذه الحالة يكون لكلّ قيمة مفتاح Key يتمّ التعامل معها من خلاله. |
| Expires | التاريخ الذي سيقوم جهاز العميل بمحو هذه البيانات عنده.. ولو تحدّد هذا التاريخ، فسيتمّ محو هذه البيانات فور إغلاق التطبيق. |
| Path | المسار الذي سيتمّ تخزين هذه البيانات عليه. |

تعال نضع قيمة في الكائن الذي عرّفناه:

**Cookie.Value = txtCookie1.Text**

وكان من الممكن اختصار الخطوة السابقة، بتعريف كائن البيانات المؤقّتة منذ البداية كالتالي:

**Dim cookie As New HttpCookie("User", txtCookie1.Text)**

الآن تعال نضيف هذا الكائن لمجموعة البيانات المؤقّتة الموجودة على جهاز العميل:

**Response.Cookies.Add(cookie)**

**ملحوظة:**

رغم أنّ باستطاعتنا إضافة البيانات المؤقّتة لكائن الاستعلام Request، إلا إنّ ذلك لن يؤثّر على جهاز المستخدم.. كائن الاستجابة Response فقط هو الذي يؤثّر على جهاز المستخدم.. إذن خذها كقاعدة: لإضافة البيانات المؤقّتة، استخدم كائن الاستجابة، ولقراءتها استخدم كائن الاستعلام.

ولحذف بعض البيانات المؤقّتة، استخدم الوسيلة Remove، كالتالي:

**Request.Cookies.Remove("User")**

ولمزيد من التدريب، عليك بتطبيق Cookies في مجلّد برامج هذا الفصل.

**التعامل مع أكثر من نموذج (صفحة):**

أبسط طريقة لعرض صفحة من خلال أخرى، هي استخدام أداة الوصلة الفائقة Hyperlink Control، مع وضع اسم النموذج الذي تريد عرضه في خاصيّة NavigateUrl لهذه الأداة.. ولو أردت إبقاء الصفحة المعروضة حاليّا، وعرض الصفحة الجديدة في نافذة مستقلّة، فحدّد خاصية Target، ومن القائمة المنسدلة اختر لها القيمة:

\_blank

ورغم سهولة هذه الطريقة، إلا إنّها لن تنفعك إذا أردت تمرير بعض المعاملات للصفحة الجديدة بطريقة مباشرة.. في هذه الحالة يمكنك استخدام الطرق التي شرحناها في مقطع "[إبقاء الحالة](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\7-%20إبقاء%20الحالة.htm)"، مثل تعريف بعض المتغيّرات عبر كائن الجلسة Session Object.

أو بإمكانك استخدام إحدى الوسيلتين [Response.Redirect](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\8-%20ASP%20كائنات\2-%20كائن%20الاستجابة.htm) أو [Server.Execute](file:///C:\Users\Administrator\Desktop\vb.net\f-%20%20برمجة%20الإنترت\19-%20مقدّمة%20لبرمجة%20الإنترنت\8-%20ASP%20كائنات\4-%20كائن%20الخادم.htm)، اللتين ناقشناهما سابقا.

تعال نرى كيف نستخدم الوسيلة Redirect.. إنّ هذه الوسيلة تقبل كمعامل، عنوان الموقع (الصفحة) التي تريد تحميلها، حيث سيتمّ عرض الصفحة الجديدة في نفس نافذة المتصفح.. وهذا معناه انتهاء عرض الصفحة الحاليّة.. ولهذا علينا أن نحتفظ ببعض القيم التي تهمنا منها، قبل تجميل الصفحة الجديدة.. هنا يمكن أن تضيف المعاملات للنصّ الذي يمثّل العنوان كالتالي:

- أضف علامة استفهام إنجليزية ?بعد اسم الموقع لتفصله عن المعاملات.

- أضف أزواج المعاملات (الاسم والقيمة) يفصل بينهما علامة =.

- افصل بين كلّ زوجين متتاليين من المعاملات بعلامة &.

- عليك بعد كلّ هذا تشفير النصّ الناتج باستخدام الوسيلة Server.UrlEncode.

انظر للمثال التالي:

**Dim params As String**

**params = "Var1=" + Server.UrlEncode(TextBox1.Text) + " & "**

**params += "Var2=" + Server.UrlEncode(TextBox2.Text) + " & "**

**params += "Var3=" + Server.UrlEncode(TextBox3.Text)**

**Response.Redirect("WebForm3.aspx?" & params)**

ولكن كيف سنقرأ هذه المعاملات؟

يمكنك في هذه الحالة استخدام خاصيّة QueryString في حدث تحميل النموذج كالتالي:

**Private Sub Page\_Load(ByVal sender As System.Object, \_**

**ByVal e As System.EventArgs) Handles MyBase.Load**

**' استخدم المعاملات في أيّ شيءٍ تريده**

**Response.Write(Request.QueryString("Var1"))**

**Response.Write(Request.QueryString("Var2"))**

**Response.Write(Request.QueryString("Var3"))**

**End Sub**

أمّا الوسيلة Execute، فهي مختلفة إلى حدّ بعيد، ذلك أنّها لا تعرض الصفحة الجديدة، ولكنّها تنفّذ ما بها من كود، وتضيف الناتج للصفحة الأولى!

لهذا فإنّ أسلم طريقة لاستخدام هذه الوسيلة، هي قراءة الناتج منها في كائن من النوع StringWriter، ووضع الناتج في لافتة على النموذج الحاليّ.

انظر للمثال التالي:

**Dim pageOut As New System.IO.StringWriter()**

**Server.Execute("ShowBasket.aspx", pageOut)**

**Response.Write("<H2>تحتوي سلّتك على هذه العناصر:</H2>"**

**Response.Write("<BR>")**

**Response.Write(pageOut.ToString)**

التعامل مع أدوات نموذج آخر:

مقارنةً بتطبيقات الويندوز، يبدو أنّ تمرير المتغيّرات بين الصفحات عمليّة صعبة المراس!

بداية يجب أن أحدثك عن الخاصية Request.Form، وهي تعيد مجموعة Collection من النوع NameValueCollection (تشبه الجدول المختلط HashTable ولكنّ رموزها Keys وقيمها Values نصّيّة).. هذه المجموعة تحتوي على أسماء كلّ الأدوات الموجودة على النموذج بالإضافة إلى أيّ حقول خفيّة Hidden Fields يحتوي عليها النموذج.. حيث توضع أسماء هذه العناصر في المجموعة (في صورة Keys وقيمها المناظرة (في صورة Values).... وبهذا يمكن معرفة النص الموجود في مربّع نصّ عن طريق الصيغة:

**Request.Form.Item("TextBox1")**

والكود التالي يريك كيفية المرور عبر كلّ عناصر النموذج، والحصول على قيمها، وكتابتها في كائن الاستجابة (سيؤدي هذا لعرضها كنص على خلفية الصفحة):

**Dim key As String**

**Response.Write(Request.QueryString)**

**For Each key In Request.Form.Keys**

**Response.Write(key + " = " + Request.Form.Item(key))**

**Response.Write("<BR>")**

**Next**

جميل.. نريد الآن أن نستخدم هذه الطريقة من نموذج آخر.. في هذه الحالة علينا أن نستخدم حيلة مشابهة لما قمنا به مع نماذج ويندوز:

على مستوى النموذج الأوّل عرّف متغيّرا عامّا مشتركا، لنضع فيه مجموعة عناصر النموذج:

**Public Shared Elem As Collections.Specialized.NameValueCollection**

ضع مربّع نصّ وزرّا على النموذج.. في حدث ضغط الزرّ اكتب ما يلي:

**Elem = Me.Request.Form**

**Response.Redirect("WebForm2.aspx")**

حيث وضعنا مجموعة عناصر النموذج في المتغيّر، ثمّ طلبنا عرض النموذج الثاني.

في حدث تحميل النموذج الثاني، اكتب ما يلي:

**Dim key As String**

**For Each key In WebForm1.Elem.Keys**

**Response.Write(key + " = " + WebForm1.Elem.Item(key))**

**Response.Write("<BR>")**

**Next**

وبهذا ستجد نصا على النموذج الثاني يسرد أسماء عناصر النموذج الأوّل وقيمها..

ويمكنك أن تضع مربّع نصّ على النموذج الثاني.. ولكي تعرض فيه الكلام الموجود في مربّع النصّ الخاصّ بالنموذج الأوّل، يمكنك استخدام هذه الجملة في حدث تحميل النموذج الثاني:

**Me.TextBox1.Text = WebForm1.Elem("TextBox1")**

وعلى فكرة.. يمكن تنفيذ نفس الفكرة السابقة باستخدام مجموعة الأدوات Controls Collection الخاصة بالنموذج.. ولكنّ هذه المجموعة لا تستطيع التعامل مع الأدوات بالاسم.. تتعامل معها فقط بالرقم.. لهذا يجب عليك كتابة كود إضافي للمرور عبر كلّ عناصر مجموعة الأدوات لمعرفة أيّها هو مربّع النصّ!

ولكن.. أرى السؤال في عينيك:

لماذا لا أعرّف متغيّرا عاما مشتركا للنموذج الأوّل نفسه كالتالي:

**Public Shared Frm1 As WebForm1**

وأضع فيه نسخة من النموذج قبل تحميل النموذج الثاني:

**Frm1 = Me**

**Response.Redirect("WebForm2.aspx")**

وبهذا أستطيع التعامل مع أدواته من النموذج الثاني مباشرة:

**Me.TextBox1.Text = WebForm1.TextBox1.Text**

فكرة معتبرة.. ولكنّ أدوات كلّ نموذج معرّفة على أنّها محميّة Protected.. وهذا يجعلنا عاجزين عن استخدامها من خارج النموذج بهذه الطريقة.. وللأسف.. لا توجد لأدوات الإنترنت خاصية Modifiers كما في أدوات الويندوز.. ولو غيّرت تعريف كلّ أداة يدويّا، فإنّ أيّ تعديل لتصميم النموذج سيلغي تعديلك ويعيد الأمور لوضعها السابق!

لهذا دعك من هذه المتاهة!