AGASHE

الطريق لاحتراف تطوير المواقع

ج6

تأليف

م/ محمد يوسف

الفهرس

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

السلام عليكم و رحمة الله تعالى و بركاته

أخواني الأعزاء متابعين سلسلة " الطريق لاحتراف تطوير المواقع" ها نحن نلتقي مجددا في مغامرة شيقة على طريقنا لاحتراف تطوير المواقع ☺ ، رحلتنا هذه المرة تأخذنا إلى لغة مهمة و مفيدة في العمل سواء لمطور المواقع أو للمبرمجين بشكل عام ، و هي في نفس الوقت تعتبر بسيطة و سهلة و لا تحتاج الكثير من الفهم ، كل ما نحتاجه هو التركيز .

ما هي لغة XML ؟؟؟

هي Extensible Markup Language و التي تعني بالعربية لغة الترميز الممتدة أو واسعة النطاق ، كثيرا ما نسمع عن استعمال الشركات لعدة لغات في نفس الوقت لنفس المشروع مثلا C++ و PHP و Java ، فكيف إذا ترتبط كل هذه اللغات داخل نفس المشروع ؟؟؟ أحد هذه الطرق هو باستعمال XML ، هذه اللغة التوصيفية (لا تعتبر لغة برمجة فعلية) مهمتها هي نقل البيانات فقط بصورة ثابتة بين جميع الأنظمة المختلفة ، يعني بطريقة أبسط لنفرض أن مشروعك جزء منه مبرمج PHP و جزء Python في هذه الحالة يمكنك تنفيذ الجزء الخاص ب PHP و وضعه في ملف XML و من ثم يقوم الجزء المبرمج بـ python بمتابعة عملية المعالجة للبيانات ، طبعا هذا أحد الاستخدامات أما في العموم فلغة XML تستخدم على نطاق واسع لحل مشكلة نقل البيانات في مختلف المواقع دون تعقيد كما سنرى في الفصول القادمة ، ناهيك عن موضوع API و كيفية الحصول على معلومات من مواقع أخرى .

كما أشرنا سابقا فهذه اللغة أشبه بملف نصي عادي تماما لكن هنا نجد المزيد من الترتيب للبيانات المنقلة بواسطتها ، فهنا لن ترى حلقات تكرار أو اتخاذ أوامر أو كتابة دوال ... الخ ، ربما يتبادر لذهنك الآن سؤال جميل و هو ما الفرق بين XML و HTML فكليهما لغة توصيفية و لا تحتوي على أي شكل من أشكال البرمجة ، لماذا لا نستعمل HTML و حسب ؟؟؟

ربما اللغتين متقاربيتين من حيث المبدأ لكن هناك اختلاف جذري في طبيعة عمل كل منهم ، HTML لغة تصف مكونات أي صفحة في موقع من حيث المكونات التي يراها المستخدم كالأزرار و الصور و الجداول و ...الخ ، أما XML فهي تصف محتويات هذه العناصر على سبيل المثال عند تسجيلك في أحد المواقع فكل ما أمامك من مربعات إدخال النص و الأزرار ..الخ هو HTML بينما البيانات التي أدخلتها فمن الممكن أن توضع في ملف XML و أن يتم إرسالها لتخزن أو تعالج في عدة برامج مكتوبة بلغات مختلفة

خلاصة القول أعزائي XML مهمة و يجب على كل مطور مواقع أن يكون على دراية بها☺

بعد هذه المقدمة الجميلة ندخل صلب الموضوع ، قم بفتح محررك و متصفحك أيا كان نوعهم حتى لو

Notpad و internet explorer 8

و نتوكل على بركة الله

الفصل الأول / الصيغة الأساسية لملفات XML :

تماما مثل أختها HTML فستجد أن XML أيضا تستعمل نظام الوسوم (Tags) لكتابة المعلومات ، المميز هنا هو كونك أنت من تحدد الوسوم لنلقي نظرة على المثال التالي :

```
<?xml  version="1.0"  encoding="utf-8"?>
<parent>
        <child1>
                <subchild1>.....</subchild1>
        </child1>
        <child2>.....</child2>
</parent>
```

ما تراه أمامك هو صيغة أي ملف XML ، دعنا نشرحه سطر تلو الآخر :

1 - سطر تعريف الملف و يحتوي على خاصيتين :

الاولى : version و هي المسؤولة عن تحديد إصدار اللغة (المعتمد حاليا حتى تاريخ صدور هذا الكتاب هو1.00).

الثانية : encoding و هو نوع تشفير النص و هو طبعا يعتمد على نوع اللغة التي كتبت بها البيانات ، و في حالة عدم كتابة هذه الخاصية تكون utf-8 هي القيمة الإفتراضية .

2 - تعريف الوسم الأب (parent element) للملف و هو أول وسم يكتب في الملف و يعبر عن نوع البيانات التي تنقلها .

3 - تعريف وسم ابن (child element) و هي جميع الوسم التي تلي الوسم الجذر في الملف مثل child1 و child2 .

4 - تعريف وسم حفيد (subchild) و هي جميع الوسوم التي تتفرع من وسم الإبن مثل subchild1 .

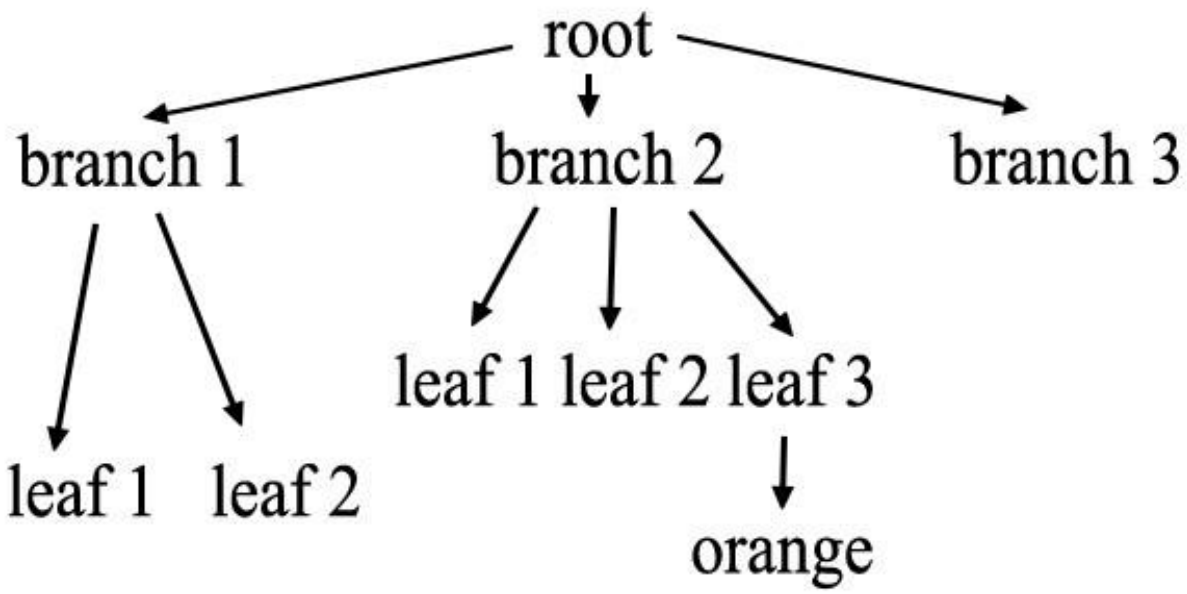
5 - نغلق وسم child1 .

6 - وسم child2 و إغلاقه.

7 – نغلق وسم parent.

ما شاء الله ما قصة الأسرة الكريمة ☺ آباء و أبناء و أحفاد .... الخ ، فيXML يتم استعمال نظام ترتيب شجري يبدأ من الجذور إلى الفروع ثم الأوراق و هكذا يسمى (XML DOM) ، كما في الصورة التالية :

```
<?xml  version=”1.0”  encoding=”utf-8”?>
<root>
        <branch1>
                <leaf1>……</leaf1>
                <leaf2>……</leaf2>
        </branch1>
        <branch2>
                < leaf1>……</ leaf1>
                <leaf2>……</leaf2>
                <leaf3>
                        <orange>……..</orange>
                </leaf3>
        </branch2>
        <branch3>…….</branch3>
</root>
```



على هذا النحو يمكنك ترتيب عناصر الصفحة أو عناصر قاعدة بيانات أو أي شيء تريد نقله أو حتى الاحتفاظ به بشكل منظم ، و عند الحاجة للرجوع لعنصر كل ما عليك فعله هو تحديد الفرع الذي تتبع له .

تتيح لك XML كتابة الوسوم بالطريقة التي تعجبك و فق متطلبات مشروعك ، فالوسومroot و parent و كل هذه المسميات يمكنك استبدالها بأي أسماء آخرى لها علاقة بمشروعك .

نأخذ عدة أمثلة لملفات XML :

هذه صفحة HTML يتم تخزينها داخل ملف XML .

```
<?xml version="1.0" encoding="utf-8"?>
<page>
    <header>
        <logo>images/smile.gif</logo>
        <nav>
            <button>home</button>
            <button>articles</button>
            <button>contact</button>
        </nav>
    </header>
    <container>
        <article>hello world ! </article>
    </container>
    <footer>
        <text>thank you</text>
    </footer>
</page>
```

معرض سيارات يخزن بيانات السيارات :

```
<?xml version="1.0" encoding="utf-8"?>
<motor_show>
```

```
        <car>
                <manufactory>skoda</manufactory>
                <model_year>2011</model_year>
                <engine_hp>60hp</engine_hp>
                <price>11,500$</price>
        </car>
        <car>
                <manufactory>nissan</manufactory>
                <model_year>2013</model_year>
                <engine_hp>80hp</engine_hp>
                <price>14,000$</price>
        </car>
        <car>
                <manufactory>kia</manufactory>
                <model_year>2014</model_year>
                <engine_hp>136hp</engine_hp>
                <price>36,000$</price>
        </car>
</motor_show>
```

الرسائل التي ترسلها بالبريد الإلكتروني :

```
<?xml  version=”1.0”  encoding=”utf-8”?>
<message>
        < from>home</ from>
        < to>articles</ to>
        < msg_content>contact</ msg_content>
        < date_and_time>contact</ date_and_time>
</message>
```

قم بحفظ الملف بإمتداد XML و قم بتشغيله كأي ملف HTML عادي في المتصفح .

قواعد XML :

كما رأينا كل ما نفعله هو فرض مسميات الوسوم بنفسنا و لتجنب المشكلات أثناء التسمية تم وضع مجوعة من القواعد يجب مراعتها

1 - اللغة حساسة لحالة الأحرف : لذا يجب أن تكون الوسوم بنفس الكلمة سواء في البداية أو النهاية :

```
<message>.....</message>
```

فمن الخطأ أن نكتبها :

```
<Message>.....</message>
```

أو

```
<message>.....</MESSAGE>
```

2 - جميع الوسوم يجب أن يكون لها وسم إغلاق : بمعنى أنه في HTML كان من الممكن كتابة الوسوم دون إغلاقها مثل

```
<p>hello world !<br>
```

في XML هذا الأمر غير جائز إطلاقا و كل الوسوم لابد من إغلاقها .

3 - حافظ على ترتيب الوسوم : يعني في HTML كنا نتغاضى عن مثل هذا الشيء :

```
<li><a  href="###"><center><bold>Go to link </li></bold></a></center>
```

لكن في XML يجب أن تكون جميع الوسوم مرتبة حسب ترتيب الاستخدام في السطر كل وسم يكتب يجب أن يناظره وسم الإغلاق الخاص به بنفس الترتيب ، فالمثال السابق لا يجوز إطلاقا .

4 - التعليقات : لكتابة تعليق في XML نستعمل الوسم التالي :

```
<!-- التعليق -->
```

5 - الرموز عموما في XML قد تتسبب بالمشكلات أثناء نقل البيانات لذا من الأفضل تجنبها باستثناء (_) فيمكنك استخدامها في تسمية الوسوم :

```
<the_book_name>.....</ the_book_name >
```

بعض الرموز لا يمكن استعمالها نظرا لانها تسبب التعارض لذا يمكن الاستعاضة عنها بالأكواد التالية :

“	&quot;
‘	&apos;
&	&amp;
>	&gt;
<	&lt;

إذا أردنا كتابة الوسم التالي :

```
<link_to_abc_website> ”go to website” >> ‘www.abc.com’ </link_to_abc_website>
```

طبعا لايمكن كتابته بهذه الطريقة مباشرة فيجب أن نستعمل الأكواد الخاصة بكل رمز منهم فيصبح شكل الوسم :

```
<link_to_abc_website>
 &amp;quot; go to website &amp;quot; &amp;gt; &amp;gt; &amp;apos; www.abc.com &amp;apos;
 </link_to_abc_website>
```

6 - ترك الفراغات بين وسمي البداية و النهاية لا طائل منه فاللغة لا تتعامل معه :

```
<text>          hello world !          </text>
```

هي نفسها لو كتبت :

```
<text> hello world ! </text>
```

7 - و هي لا تعتبر قاعدة لكن يستفضل أن تختار أسماء الوسوم ذات دلالة واضحة و مفهومة يعني مثل هذا الملف لا فائدة منه

```
<x>
<y><z> what is this??? </z></y>
</x>
```

خصائص الوسوم (Attribute) :

تماما كما في HTML تمكنك XML من فرض خصائص للعناصر حسب احتياجاتك ، و طبعا لابد من وضع قيمتها بين علامتي تنصيص سواء مزدوجة أو مفردة ففي المثال التالي :

```
<?xml version="1.0" encoding="utf-8"?>
<message>
        < from>Ahmed</ from>
        < to>Mohamed</ to>
        < msg_content>text</ msg_content>
        < date_and_time>contact</ date_and_time>
</message>
```

يمكن كتابته باستعمال الخصائص :

```
<?xml version="1.0" encoding="utf-8"?>
<message date_and_time ="12/5/2015">
        < from>Ahmed</ from>
        < to>Mohamed</ to>
        < msg_content>text</ msg_content>
</message>
```

أو أن تكتب كامل الملف في الخصائص ☺

```
<?xml version="1.0" encoding="utf-8"?>
<message date_and_time ="12/5/2015" from=" Ahmed" to=" Mohamed" msg_content="
text">
</message>
```

بكل الاحوال استعمال الخصائص مع العناصر غير مستحب لأنه غير مرتبط بالترتيب الشجري للملف و بالتالي ربما تواجهك مشاكل في إرساله مع الملف ، لكن بوسعك إستعمالها لغرض الترتيب بمعنى أنه في المثال السابق لو كان لدينا 10 رسائل فمن الممكن ترتيبهم بعنصر بخاصية number او message_id مثلا .

مشكلة اختلاط أسماء الوسوم (Namespace conflict) :

كما قلنا سابقا XML تعطيك كامل الحرية لاختيار الوسوم و كيفية ترتيبها ، لكن أحيانا ربما تواجه وجود خلط بين وظيفة الوسم و اسمه ، أشهر مثال على ذلك الوسم table فلو نظرنا للمثالين التاليين بتعمق :

```
<table>
        < tr>
                < td>pages</ td>
        < /tr>
        < tr>
                < td>links</ td>
        < /tr>
</table>
```

```
<table>
        < color>brown</ color>
        < width>1.1m</ width>
        < height>1.3m</ height>
        < length>1.2m</ length>
</table>
```

في المثال الاول نجد الوسم table يعبر عن جدول في صفحة في أحد المواقع ، بينما على النقيد في المثال الثاني يحتوي على مواصفات طاولة فإذا تم إستدعاء كلا الملفين معا عندها سوف يحدث خلط بين البيانات و لن يتمكن البرنامج من معرفة أيا منهما يطبع .

لذا و بصفة عامة في حالة وجود تشابه بين أسماء الوسوم الجذر نستعمل البادئة (prefix) و هي عبارة عن كلمة أو إسم أو حرف يوضع قبل اسم الوسم بغرض التفريق بين الوسوم المتشابهة .

```
<a:tag   xmlns:a = “URI” >
……..
</a:tag>
```

حيث :

a:tag --> الوسم بعد إضافة البادئة

Xmlns:a --> كلمة محجوزة لتعريف البادئة حيث أن a هي البادئة يمكن طبعا كتابة أي كلمة .

URI --> حيث أنه عنوان مصدر الصفحة و هو لتمييز البادئات المختلفة داخل الصفحة .

و يجب أن تحتوي كل الوسوم داخل الوسم الجذر على هذه البادئة ، لذا لنرى الأمثلة السابقة بعد التعديل :

```
<x:table  xmlns:x=”www.abc.com/pages”>
        < x: tr>
                < x: td>pages</ x: td>
        < / x:tr>
        < x: tr>
                < x:td>links</ x: td>
        < / x:tr>
</ x:table>
```

```
<y:table  xmlns:y=”www.abc.com/furniture”>
        < y:color>brown</ y:color>
        < y: width>1.1m</ y: width>
        < y: height>1.3m</ y:height>
        < y:length>1.2m</ y: length>
</ y:table>
```

أو أن تضع كليهما في وسم جذر واحد :

```
<table_elements   xmlns:x=”www.abc.com/pages”   xmlns:y=”www.abc.com/furniture”>
        <x:table>
                < x: tr>
                        < x: td>pages</ x: td>
                < / x:tr>
                < x: tr>
                        < x:td>links</ x: td>
                < / x:tr>
        </ x:table>

        <y:table>
                < y:color>brown</ y:color>
                < y: width>1.1m</ y: width>
                < y: height>1.3m</ y:height>
                < y:length>1.2m</ y: length>
        </ y:table>
</table_elements >
```

أيضا يوجد حل جميل و مختصر و هو عن طريق استعمال URI فقط مع الخاصية xmlns دون اللجوء لاستعمال البادئات فيمكن كتابة المثال السابق مباشرة هكذا :

```
<table  xmlns =”www.abc.com/pages”>
        < tr>
                < td>pages</ td>
        < /tr>
        < tr>
                < td>links</ td>
        < /tr>
</table>
```

```
<table  xmlns  =”www.abc.com/furniture”>
        < color>brown</ color>
        < width>1.1m</ width>
        < height>1.3m</ height>
        < length>1.2m</ length>
</table>
```

إذا نستنتج من كل ما سبق أنه لدينا ثلاث طرق للتعامل مع حالات تشابه أسماء الوسوم الجذر المختلفة :

1 - استعمال البادئات مع تعريف المصدر لكل وسم جذر على حدة .

2 - استعمال البادئات مع تعريف جميع المصادر داخل وسم جذر وحيد يشمل باقي الوسوم الجذور .

3 - استعمال خاصية تعريف عينات الاسماء الرئيسية (Defualt Namespaces) لتجنب تكرار البادئة داخل باقي الوسوم الأبناء المتفرعين من الوسم الجذر .

ما هو URI ؟؟؟؟؟

ربما تعتقد أن هو نفسه URL لكن مع الأسف فكليهما مختلف تماما فالأول اختصار لمعرف المصادر المنتظمة (uniform resources identfer) و هو المسؤول عن تخزين بيانات الموقع ، بينما الثاني وهو اختصار لموجه المصادر المنتظمة (uniform resources locator) و هو نظاق الموقع (domain) .

الفصل الثاني / التحقق من ملفات XML :

بسبب عملية نقل الملفات بين المواقع و لأن ملفات XML يمكن أن تكتبها و ترسلها بنفسك للموقع أو للتطبيق ، لذا كان لابد من تطوير آلية أو مرجع لطرق كتابة ملفات XML بمعنى أنه لو طلب منا القيام بكتابة ملف XML عندها سنجد أن بوسع كل مبرمج أن يكتب الملف بالطريقة التي تعجبه و عندها يصبح هناك ارتباك في البيانات ، لذا يوجد لدينا طريقتين لوضع معيار أساسي لملفات XML التي يتم إرسالها أو استقبالها .

1 - تعريف نوع المستند (document type definition) و تختصر ب DTD :

في هذه الطريقة نقوم بكتابة تعريف لكل الوسوم داخل الصفحة كالآتي :

نفرض أنه لدينا الملف التالي :

```
<?xml  version="1.0"  encoding="utf-8"?>
<message>
        < from>Ahmed</ from>
        < to>Mohamed</ to>
        < msg_content>text</ msg_content>
</message>
```

نقوم بكتابة DTD :

```
<?xml  version="1.0"  encoding="utf-8"?>
<!DOCTYPE message [
<!ELEMENT message (from, to, msg_content)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT msg_content (#PCDATA)>
]>

<message>
        < from>Ahmed</ from>
        < to>Mohamed</ to>
        < msg_content>text</ msg_content>
</message>
```

كل ما يهمنا هو الجزء الجديد الذي أضفناه بخصوص DTD :

```
<!DOCTYPE message [
<!ELEMENT message (from, to, msg_content)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT msg_content (#PCDATA)>
]>
```

في السطر الأول : نجد أننا قمنا بتعريف message كوسم جذر .

في السطر الثاني : عرفنا ثلاث وسوم هم from , to , msg_content و هذا معناه أن الوسم الجذر يحتوي على ثلاث وسوم متفرعة .

في السطر الثالث : عرفنا وسم from و العبارة #PCDATA تعني أنه يقبل القيم النصية .

في السطر الرابع : عرفنا وسم from و العبارة #PCDATA تعني أنه يقبل القيم النصية .

في السطر الخامس : عرفنا وسم from و العبارة #PCDATA تعني أنه يقبل القيم النصية .

في السطر السادس : أغلقنا وسم تعريف الملف .

عند تشغيل الملف في المتصفح ستلاحظ أن أكواد DTD لا تظهر إلا عند استعراض الكود المصدري للصفحة .

أيضا يمكنك كتابة صيغة DTD في ملف منفصل و تضمنيه في ملفات XML بعد ذلك لكي لا تعيد تكرار الكود كل مرة ، و يتم حفظ الملف بصيغة DTD .

كود التضمين :

```
<?xml  version="1.0"  encoding="utf-8"?>
<!DOCTYPE  message  SYSTEM "file.dtd">
<message>
        < from>Ahmed</ from>
        < to>Mohamed</ to>
        < msg_content>text</ msg_content>
</message>
```

حيث أن "file.dtd" SYSTEM هو الرابط لمكان الملف .

2 - تعريف مخطط ملف XML (XML schema definition) :

طريقة من طرق تعريف ملفات XML ، مشابهة إلى حد ما إلى البادئات (prefix) ، و لكنها تستعمل معيار خاص موضوع من قبل منظمة w3 ، و المعيار عبارة عن ملف DTD عادي لكن به تفاصيل عامة ، يمكنك الإطلاع عليه من خلال هذا الرابط :

https://www.w3.org/2001/XMLSchema

الفرق بين schema و ملف DTD الذي تصنعه بنفسك ، هو أن shema يعتبر مرجع عام لجميع حالات الملفات و البيانات بين مختلف المواقع ، أما ملف DTD الذي تكتبه للبيانات الموجودة في موقعك فهو يعتبر مرجع خاص بالبيانات داخل موقعك فقط و خارجه لا اعتراف بها .

نأخذ الآن مثال على استعمال هذه الطريقة ، نفرض وجود الملف التالي :

```
<?xml  version=”1.0”  encoding=”utf-8”?>
<book>
        < name>الطريق لاحتراف تطوير المواقع ج6</ name>
        <date>2016  – 2017 </date>
        < num_of_pages>100</ num_of_pages>
        < size>5.4 MB</ size>
        <auther>Mohamed yousef</auther>
</book>
```

يمكنك كتابته بواسطة XML schema :

```
<?xml  version=”1.0”  encoding=”utf-8”?>
<xs:schema    xmlns:xs="http://www.w3.org/2001/XMLSchema">
        <xs:element name="book">
                <xs:complexType>
                        <xs:sequence>
                                <xs:element name="name" type="xs:string"/>
                                <xs:element name="date" type="xs:string"/>
                                <xs:element name=" num_of_pages " type="xs:string"/>
                                 <xs:element name=" size " type="xs:string"/>
                                <xs:element name=" auther " type="xs:string"/>
                         </xs:sequence>
                </xs:complexType>
        </xs:element>
</xs:schema>
```

الفصل الثالث / ربط ملفات XML مع ملفات CSS :

بالرغم من أن XML ليست مخصصة لعرض البيانات ، إلا أنها مثل أختها HTML يمكن ربطها بملفات CSS و التعديل على شكل الصفحة ، نتذكر مثال معرض السيارات :

```
<?xml  version=”1.0”  encoding=”utf-8”?>
<motor_show>
        <car>
                <manufactory>skoda</manufactory>
                <model_year>2011</model_year>
                <engine_hp>60hp</engine_hp>
                <price>11,500$</price>
        </car>
        <car>
                <manufactory>nissan</manufactory>
                <model_year>2013</model_year>
                <engine_hp>80hp</engine_hp>
                <price>14,000$</price>
        </car>
        <car>
                <manufactory>kia</manufactory>
                <model_year>2014</model_year>
                <engine_hp>136hp</engine_hp>
                <price>36,000$</price>
        </car>
</motor_show>
```

لربط ملف CSS بملف XML ما علينا سوى إضافة السطر التالي بعد سطر تعريف الملف :

```
<?xml  version=”1.0”  encoding=”utf-8”?>
<?xml-stylesheet type="text/css" href="file.css"?>
```

حيث :

Xml-stylesheet : عبارة لتعريف ملف CSS .

type=”text/css” : تحديد نوع الملف الذي سوف يتم ربطه .

href=”file.css” : اسم الملف و إمتداده .

نضيف سطر التعريف للمثال السابق :

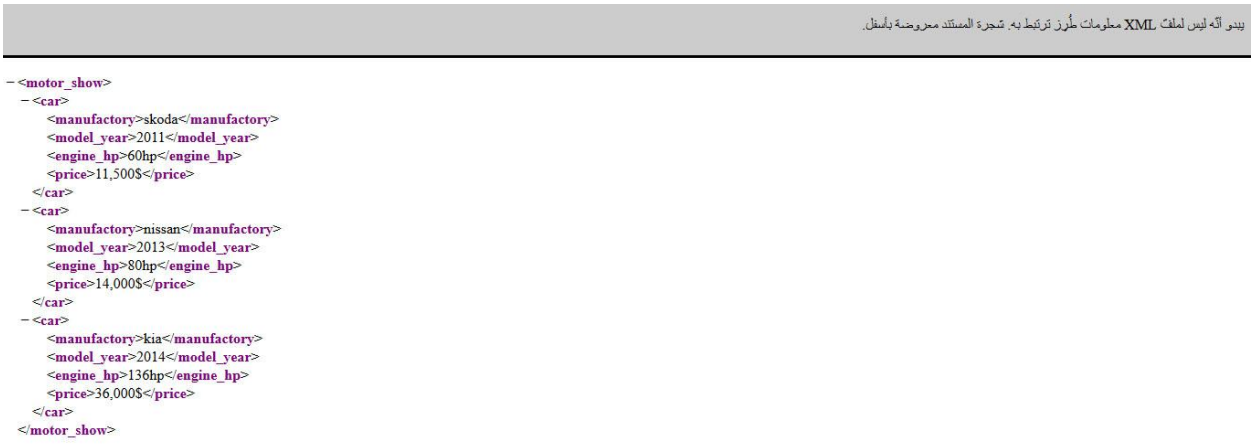
```
<?xml  version=”1.0”  encoding=”utf-8”?>
<?xml-stylesheet type="text/css" href="style.css"?>
<motor_show>
        <car>
                <manufactory>skoda</manufactory>
                <model_year>2011</model_year>
                <engine_hp>60hp</engine_hp>
                <price>11,500$</price>
        </car>
        <car>
                <manufactory>nissan</manufactory>
                <model_year>2013</model_year>
                <engine_hp>80hp</engine_hp>
                <price>14,000$</price>
        </car>
        <car>
                <manufactory>kia</manufactory>
                <model_year>2014</model_year>
                <engine_hp>136hp</engine_hp>
                <price>36,000$</price>
        </car>
</motor_show>
```

و بالنسبة لملف style.css فيمكن مثلا كتابة :

```
car{
border:2px solid #000;
display:block;
width:250px;
height:150px;
margin:10px;
padding:5px;
}
manufactory , model_year , engine_hp , price{
display:block;
Font-size:16pt;
```

```
Font-weight:bold;
Color:blue;
margin-bottom:30px;
line-height:0.5;
}
```

عند فتح الصفحة قبل ربط ملف CSS يكون الناتج هو مجموعة الوسوم العادية :



لكن بعد الربط مع ملف CSS :

الفصل الرابع / استخراج البيانات من ملف XML بواسطة JavaScript :

يعتبر هذا الفصل هو أهم فصل في الكتاب ، فهنا سوف نشعر بفائدة ملفات XML في استقبال البيانات من المواقع المختلفة ، بداية دعونا نتذكر كيفية التحكم بمحتوى عنصر داخل صفحة HTML باستعمال JavaScript .

لنفرض وجود صفحة HTML التالية :

```
<html>
<head><script src="control.js"></script></head>
<body>
        <p id="name"></p>
</body>
</html>
```

كل ما نريده الآن هو ملئ العنصر p باسم المستخدم لكن عن طريق JS ، لذا يمكننا استعمال الكود التالي في ملف control.js :

```
document.getElementById("name").innerHtml = "ahmed";
```

بواسطة هذا السطر اللطيف ستتغير قيمة العنصر p و الذي اشرنا له بالمعرف name ((id =" name")) إلى الإسم ahmed

بنفس هذه الكيفية سوف نستخرج محتويات ملف XML و نعرضها للمستخدم في مختلف العناصر التي تحتويها الصفحة .

لكن وقبل أن نتجه إلى عملية الاستخراج لابد من معرفة بعض الأمور :

1 - معرف طلبات XML أو (XMLhttpRequest) : لربط أي ملف XML بالكود لابد من تعريف كائن من هذا الصنف في بداية الكود .

```
var x = new XMLhttpRequest( );
```

2- استعمال الدالة open : و هي من إسمها مسؤولة عن فتح الملف, تأخذ ثلاث معاملات

- طريقة الإرسال و هي تعتمد على مدى أهمية البيانات و تحتمل طريقتين POST أو GET .
- رابط الملف و هو طبعا مكان تخزينه و إسمه مثل .../data/user.xml
- تفعيل التزامن (synchronous) و هي خاصية مهمة في حالات البيانات الكثيرة ففي حالة التزامن يكون الإرسال و الإستقبال معا في نفس الوقت بينما في حالة الإرسال الغير متزامن (asynchronous) يكون هناك انتظار لحين اكتمال تحميل الصفحة و لهذا تفعيل التزامن غير مستحب ففي حالة سرعات الإنترنت البطيئة ستصبح النتائج غير معروفة . لتفعيل خاصية الإرسال المتزامن نضع القيمة ب false و لتفعيل خاصية الإرسال الغير متزامن true .

```
x.open("GET" , "user.xml" , true);
```

الإرسال الغير متزامن نضع true ..... للتأكيد ☺

3 - الدالة send : و هي للسماح للبيانات بالانتقال من ملف XML لصفحة HTML .

4 – الدالة responseXML : ة هي للإستجابة لطلب نقل البيانات و يستفضل ربطها بمتغير

5 - استخراج قيمة أي وسم من ملف XML :

```
var y = x.getElementsByTagName("user")[0].childNodes[0].nodeValue;
```

حيث :

- Var y : متغير سيحمل القيمة من الوسم داخل ملف XML و ممكن يكون عنصر داخل صفحة HTML .
- X : كائن من الصنف XMLhttpRequest .
- getElementByTagName(“user”)[0] : دالة لتحديد العنصر باسم الوسم ، هنا فرضنا وجود وسم داخل ملف XML إسمه user ، [0] و هذه معناها أول عنصر في الصفحة ، طيب لو إفترضنا أنه لا يوجد غير وسم وحيد إسمه user هل نضعها ؟؟؟ نعم لأن الملفات غالبا تعامل كمصفوفة عناصر .
- childNodes[0].nodeValue : أول قيمة داخل الوسم ، و هنا الصفر إلزامي طالما العنصر لا يحتوي سوى على قيمة نصية .

طبعا لتأكيد كل هذا الكلام لابد من مثال دسم ☺

لدينا ملف XML إسمه users_data.xml يحتوي على البيانات التالية :

```
<users_data>
        <user>
                <user name>mohamed</username>
                <password>12345</password>
                <email>Mohamed@abc.com</email>
        </user>
</users_data>
```

و لدينا ملف HTML إسمه users_page.html يحتوي على العناصر التالية :

```
<html>
<body>
        <script></script>
```

```
        <p  id="username"></p>
        <p  id="password"></p>
        <p  id="email"></p>
</body>
</html>
```

كل ما عليك فعله هو كتابة كود لوضع كل قيمة داخل الوسوم في ملف XML في عنصر p المناسب يعني usename يوضع في العنصر p ذو id="username" و هكذا ....الخ

حسنا داخل وسمي script نكتب الكود التالي :

```
xmlhttp = new XMLHttpRequest();
xmlhttp.open("POST"," users_data.xml",true);
xmlhttp.send();
doc = xmlhttp.responseXML;
document.getElementById("username ").innerHTML=
doc.getElementsByTagName("username")[0].childNodes[0].nodeValue;
document.getElementById("password").innerHTML=
doc.getElementsByTagName("password")[0].childNodes[0].nodeValue;
document.getElementById("email ").innerHTML=
doc.getElementsByTagName("email")[0].childNodes[0].nodeValue;
```

- عرفنا xmlhttp كائن من الصنف XMLHttpRequest .
- إستعملنا الدالة open لفتح الملف و مررنا ثلاث قيم لمعاملاتها الثلاث :
  الأول وهو نوع الإرسال و هنا إستعملنا POST و ذلك لسرية البيانات فإرسال اسم المستخدم و كلمة السر يجب أن يكون مؤمن لأقصى درجة ممكنة .
  الثاني و هو إسم الملف users_data.xml .
  الثالث و هو تفعيل خاصية عدم التزامن و وضعنا لها القيمة true .
- دالة الإرسال send .
- عرفنا متغير إسمه doc ليحمل البيانات المستقبلة .
- السطور الثلاث الأخيرة هي مجرد نقل القيمة من كل وسم في ملفXML إلى العنصر ذو id المناسب .

الفصل الخامس / استخراج البيانات من ملف XML بواسطة Visual Basic 6 :

في هذا الفصل نحاول تصليت الضوء على استعمال آخر للXML لكن هذه المرة ليس مع مواقع الإنترنت بل للبرامج الموجودة على الأجهزة أيضا ، في هذا الفصل سنستعمل بيئة Visual Basic 6 كمثال على استعمال البرامج أيضا لملفات XML ، طبعا أغلب اللغات الموجهة لبرامج سطح المكتب أو حتى تطبيقات الهواتف المحمولة مثل C# و Java تدعم التعامل مع ملفات XML سواء بالقراءة أو الكتابة ، على هذا النحو يمكنك إرسال التحديثات و بيانات البرنامج إلى كل المستخدمين بكل سهولة و يسر ، طبعا إذا كنت لا تجيد VB6 فلا تلقي بالا لهذا الفصل و إنتقل للفصل التالي مباشرة .

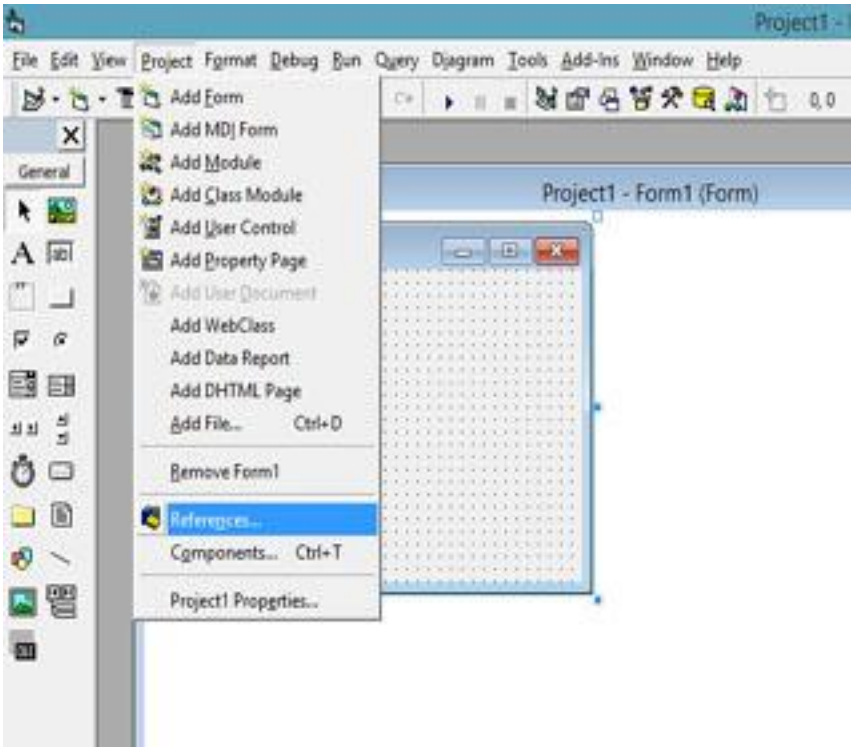
كيف نربط ملف XML ببرنامج مصمم ب VB6 ؟

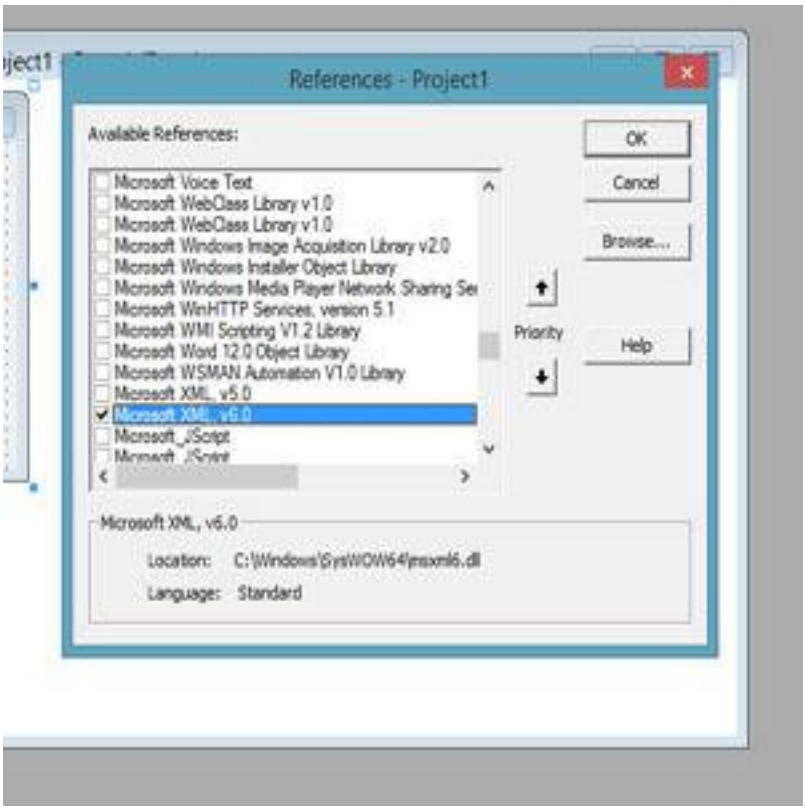
بعد فتح بيئة VB6 و نحدد نوع المشروع الجديد كـ standard EXE نتجه لقائمة project و منها نختار references ثم نختار Microsoft XML, version 2.0 ، في حالة عدم العثور عليها يمكنك تحميلها من الرابط التالي :

https://www.microsoft.com/en-us/download/details.aspx?id=3988

و تثبيتها لا يحتاج لأي شرح فقط إفتح الملف ☺ .

النسخة التي قمنا بتحميلها هي 6 و ليست 2 ، لكن لا مشكلة سواء حملت 3 ، 5 ، 4 جميعهم يعملون بنفس الكود فلا داعي للقلق .





بعد أن انتهينا من تحميل المكتبة و تنصيبها و ربطها بالبرنامج ، كل ما علينا فعله الآن هو تعريف بعض الأكود :

1 - لتعريف عنصر جديد من هذه الأداة :

```
Dim xml_doc As MSXML2.DOMDocument
```

2 - لفتح ملف XML نستعمل الدالة Load :

```
xml_doc.Load("../folder/file.xml")
```

و هنا نجد شيء مهم جدا و هو أن الدالة Load تقبل مكان الملف في الجهاز أيضا يمكن وضع رابط لموقع على شبكة الإنترنت

لكن هنا يجب وضع البادئة http:// قبل عنوان الموقع .

```
xml_doc.Load("http://www.abc.com/folder/file.xml")
```

و بهذا يستقبل البرنامج بيانات التحديثات من الموقع مباشرة دون الحاجة لتحميلها على جهاز المستخدم .

3 - لقراءة محتويات أي وسم داخل ملف XML نسعمل الدالة :

```
Dim xml_tag As IXMLDOMElement
Set xml_tag = xml_Doc.selectSingleNode("title")
```

عرفنا متغير يحمل قيم الوسوم داخل ملف XML و بعدها استعملنا الدالة selectSingleNode للحصول على قيمة وسم معين .

بعد أن عرفنا هذه الدوال نذهب لنطبق ما تعلمناه على مثال و كالعادة سنفرض ملف XML خفيف إسمه update.xml يحتوي على نص سوف نقوم بطباعته في البرنامج :

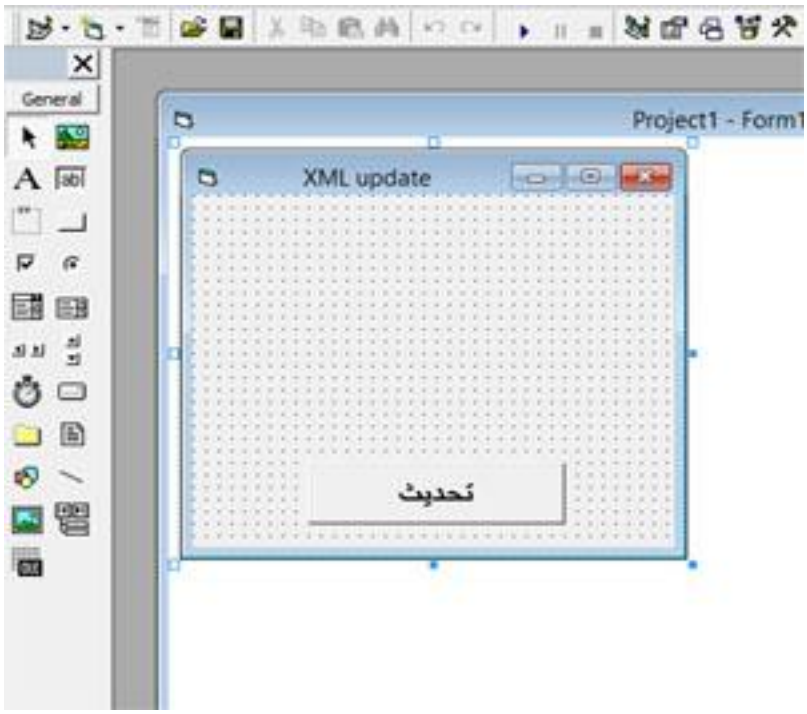
```
<?xml version="1.0" encoding="utf-8"?>

<title>

The program was successfully updated !

</title>
```

و في VB6 نصمم برنامج يحتوي فقط على زر مكتوب عليه كلمة تحديث :



نكتب بعد ذلك الكود :

في منطقة General و قبل كتابة أي أمر نعرف متغير و ليكن إسمه x ليحمل قيمة الوسم :

```
Dim x As String
```

في حدث بدء تشغيل البرنامج (form_load) نكتب كود الربط مع ملف XML :

```
Private Sub Form_Load()
Dim xml_doc As New MSXML2.DOMDocument
xml_doc.Load ("update.xml")
Dim xml_tag As IXMLDOMElement
Set xml_tag = xml_doc.selectSingleNode("title")
x = xml_tag.Text
End Sub
```

و في حدث الضغط على زر التحديث نكتب كود لطباعة نص جديد و طبعا هذا مجرد مثال نعتبر أن طباعة هذا النص هو التحديث ، لكن طبعا يمكنك عمل الكثير من الأمور كتغيير الخطوط و أحجام العناصر ....الخ بل و حتى إضافة ملفات DLL للبرنامج :

```
Private Sub Command1_Click()
Print x
End Sub
```

الفصل السادس / مواضيع متفرقة في XML :

في هذا الفصل نتحدث عن مجموعة من التقنيات و الملاحظات التي ترتبط بلغة XML و التي قد تمر عليك ، فالأفضل على الأقل أن تكون على دراية و لو حتى بمعنها .

2 - لغة تحويل الأنماط الممتدة (Extensible Stylesheet Language Transformations - XSLT) :

تقوم هذه اللغة بتحويل ملفات XML العادية إلى صفحات HTML تعرض في المتصفح ، و هي تعتبر بديل عن ربط ملف XML لملفات CSS ، لكنها تمتاز بالقدرة على التحكم بالبيانات داخل الملف من حيث عرضها و إخفائها إضافة العناصر و إزالتها و الكثير من الأمور إذا كنت تعرض ملفات XML مباشرة للمستخدم إذا فهي الخيار الأفضل لك .

3 - مسار XML (XML Path - XPATH) :

تقوم هذه التقنية بتسهيل عملية تحديد الوسوم داخ ملفات XML و لابد من إستعمالها إذا كنت تستعمل XSLT .

4 - روابط و مؤشرات XML (XLink , XPointer) :

XLink تقوم بتحويل أي وسم داخل الملف إلى رابط تماما مثل الوسم a في HTML .

XPointer تقوم بتحديد أجزاء محددة من الصفحة و يتم ربطها بـ XLink .

5 - استعلام XML (XQuery) :

نستعمل هذه التقنية إذا أردنا أيضا استخراج البيانات من ملفات XML كاستعلامات ، و الفرق بين هذه التقنية و بين XPath أن XQuery تستعمل مع ملفات XML الكبيرة و التي قد تستعمل كقواعد بيانات للبرامج فهي أكثر إحترافية من XPath .

6 - لغة الترميز الاسلكية (Wireless Markup Language - WML) :

هي لغة مشتقة من XML بغرض نقل البيانات بين الأجهزة التي تستعمل الشبكات الاسلكية .

7 - لغة الترميز الكيميائية (Chemical Markup Language - CML) :

و من إسمها هي لغة تم إشتقاقها من XML لتخزين بيانات المركبات الكيميائية .

8 - لغة الترميز الحسابية (MathML) :

و هي مخصصة لتخزين و تبادل المعادلات و العمليات الحسابية بشكل موحد على شبكة الإنترنت .

9 - لغة الترميز الموسيقية الممتدة (MusicXML) :

طبعا لتخزين الألحان الموسيقية .

و على هذا النحو ستجد عشرات اللغات المخصصة لنقل أنواع مختلفة للملفات بل هناك أيضا الكثير من المؤسسات تقوم بعمل لغات ترميزية خاصة بها لتخزين بياناتها .

بهذا نكون أحبائي قد وصلنا لنهاية هذا الفصل أرجوا من الله أن أكون قد وفقت في الشرح و كما أقول دائما في حالة وجود أي لبس أو أي أمر مستعصي ، ملاحظات ، أخطاء .....الخ ، لا تترددوا في التواصل معي فهذا سيكون من دواعي سروري .

طبعا هذا الكتاب أو أي كتاب كما نعرف لا يوصل للإحتراف ، لذا أحرصوا على العمل فمثلا قوموا بعمل مشروع بواسطة XML و حاولوا دمجها مع معرفتكم السابقة بهذا تزيد خبرتكم و يرتفع مستواكم ، هذا و أتمنى كل التوفيق لي و لكم .

أخوكم المحب /

محمد يوسف محمد

Modi401@hotmail.com

أيضًا

https://www.facebook.com/mohamed.yossef.583

أو تفضلوا بزياراتنا على الموقع الإلكتروني

www.AGASHE.pe.hu

AGASHE

2015 - 2016