

Wysiwyg Rich Text Editor

الصندوق الذهبي

مشروع بناء محرر نصوص WYSIWYG مخصص مبدئيا للعمل على IE 5.5

جميعنا يرى محررات النصوص في المنتديات ، ولربما أغلبنا يعرف ما هو مصطلح WYSIWYG والذي يعني أن التعليمات التي تضعها في مكان ما بخصوص مظهر البرنامج ستظهر تماما كما وضعتها (ما تراه هو ما يتحصل عليه)

أي أن ضغطنا لزر إمالة الخط | سيجعل الخط مائلا فورا بدلا من إدراج وسم `[b]` أو `<b></b>` حوله

المتعة في التعامل مع محررات WYSIWYG (WYSIWYG Rich Text Editors) هي أنك ترى النص تماما كما سيظهر في المنتدى... وفي الحقيقة كانت عملية الحصول على WYSIWYG محترمة تتطلب أكوادا برمجية كثيرة وعملا شاقا ، إضافة إلى الإلمام بلغة HTML وتنسيقاتها المختلفة. ومع ظهور IE 5.5 وما بعده بدأت شركة مايكروسوفت تسمح للمطورين ببرمجة الانترنت اكسبلورر باستخدام الكائنات Objects والدالات Functions والأدوات مسبقة الصنع BuiltIn tools مما سهل كثيرا على المطورين بناء التطبيقات وساعد في رفع شعبية انترنت اكسبلورر مقارنة بغيره من المتصفحات، ناهيك عن كونه مجاني ويتوفر مع ويندوز تلقائيا.

من الجميل لزائر موقعك أن يستطيع إبداء ملاحظاته والكتابة في سجل الزوار مثلا او ادراج موضوع او تنسيق ملفه الشخصي باستخدام محرر نصوص WYSIWYG قوي، والكثير من البرامج فقدت شعبيتها بسبب فقدانها لهذا المحرر، فمثلا يعتبر فقدان WYSIWYG مناسب في منتدى ASP.net الذي تطوره مايكروسوفت من الأسباب التي دفعت الكثير من مصممي المواقع على اختيار vBulletin من شركة JelSoft وبالتالي التحول من لغة ASP.net إلى Php

سنحاول في هذا المشروع بناء محرر WYSIWYG متواضع ومن ثم نحسن أدائه رويدا رويدا حتى الوصول به إلى مستوى جيد . ومن ثم سنحاول بناء paraser له ليستطيع صاحب المنتدى أو الموقع استخدام أكواد vBCode او BBCode (وهي تلك الوسوم tags البديلة التي نراها في محرر vBulletin من مثل `[b]` أو `[center]` والتي تستخدم كبديل عن وسوم HTML كون استخدام الثانية يعرض الموقع لخطر إدراج سكريبت سري)

الهدف من هذا المشروع تعليمي بالأساس، وذلك لتتعلم سوية بناء مشروع جافاسكريبت من الصفر. بعض التوابع التي أدرجتها في هذا الموضوع توابع مأخوذة من برامج مجانية أو محررات نصوص مفتوحة المصدر على النت

open source WYSIWYG

وطالما أن الهدف تعليمي فلا بأس في ذلك

أغلب التعليمات التي سأكتبها مخصصة لانتترنت إكسبلورر، والسبب في ذلك أمرين

- ١ - التبسيط: والابتعاد عن تعقيدات اللعب على المتصفحات مختلفة النواة (مثل IE مقارنة Firefox و Safari)
- ٢ - جهلي بالفروقات الدقيقة بين المتصفحات: وطالما أن المشروع أساسا تعليمي، فلماذا لا أتعلم أنا أيضا من مشاركات الأعضاء

من المهم معرفة أنني مجرد هاو، لست محترفا لجافاسكريبت، بل شأني شأن الكثير من مطوري الويب، نقوم باستعارة الأكواد المجانية (وغير المجانية أحيانا) وتعديلها بما يناسب استخدامنا الشخصي، لذلك أرجو أن لا يتوقع أحد أن لدي جميع الأجوبة، كما أرجو من السادة القراء تأجيل الأسئلة حتى نهاية كل مرحلة من المراحل.. ولربما أفتح صفحة خاصة بكل مرحلة من مراحل التطوير بحيث تنحصر النقاشات فيها في هذه المرحلة بالذات..

رجائي الخاص أن لا ييخل علينا أي من الأخوة بالإضافات والملاحظات والتطويرات التي يرونها مناسبة حتى لو أدت لنسف المشروع الذي قمت به وكتابته من جديد، فكما قلت فأنا مجرد هاو لا أكثر

والرجاء الأخير: عدم تشتيت الموضوع بالسؤال حول متطلبات شخصية مثلا (أنا عندي موقع وأرجو ان تساعدني في تحسينه... أريد محرر WYSIWYG حالا يقوم بعمل كذا وكذا..). فهذا لن يفيد الأعضاء

المرحلة الأولى:

الهدف من المرحلة: بناء محرر WYSIWYG مبسط كواجهة للرد السريع بحيث يقوم بالأساسيات من مثل جعل الخط عريض او مائلا أو ادراج رابط او صورة الخ

تمهيد:

بدأت مايكروسوفت بأدراج أدوات تعين مطوري الجافاسكريبت على التحكم بالصفحة بشكل شبه كامل، بحيث ترفع أداء انتترنت اكسبلورر وتجعله متصفح المهام الصعبة..

من الإضافات الهامة التي إضافتها مايكروسوفت إمكانية التحكم بالكائنات، حيث يعتبر كل وسم tag كائنا يمكن التعامل معه بشكل مستقل.. ويمكن استدعاؤه عبر أداة الوثيقة document والتي يمكن عبرها إجراء تعديلات على محتويات صفحة الويب من مثل تغيير التنسيق لعنصر ما (كالصورة أو العنوان) أو إخفاءها الخ

أضفت مايكروسوفت على أداة الوثيقة إمكانية إجراء التعليمات، ومن أهم التعليمات التي يمكن إجراؤها تعليمة execCommand والتي تمكننا من العمل بمرونة مع الصفحة.

صيغة هذه التعليمة

bSuccess = **object.execCommand**(sCommand [, bUserInterface] [, vValue])

حيث

- **bSuccess** يعبر عن القيمة المنطقية الناتجة عن تنفيذ التعليمة، فتكون قيمته **true** إذا تمت العملية بنجاح و **0** في حال فشلها
- **sCommand** ويعبر عن طبيعة الأمر المعطى، وهناك لائحة من الأوامر التي يمكن تمريرها عبر هذا المتحول سنتحدث عنها بعد قليل. مثلا لو مررت **bold** فالنص المختار سيصبح عريضا.
- **bUserInterface**: قيمة منطقية تعبر عن فيما إذا توجب على التعليمة إظهار واجهة إدراج بارامترات إضافية للمستخدم، فمثلا في حال إدراج صورة ستظهر نافذة تطلب إدخال مسار الصورة فيما لو إسندت القيمة **1** إلى **bUserInterface** ، وقد يتطلب المبرمج ان تكون واجهة المستخدم ذات خيارات أوسع، أو أن لا تكون هناك خيارات (مثلا إدراج الايقونات الباسمة **smilies** لا تتطلب إدراج مسار لان مسارها معرف سابقا) فيمنع ظهور واجهة الاستفسار هذه.
- **vValue**: وهي القيمة المسندة للتعليمة في بعض الحالات، فمثلا لو اسندنا القيمة **FontName** إلى **sCommand** ، يمكننا اسناد اسم الخط "Traditional Arabic" إلى **vValue**.

مثال:

`document.execCommand('insertimage', 0, imgSrc)`

تقوم بإدراج صورة مسارها **imgSrc** في موقع الفأرة.

أهم البارامترات التي يمكن إسنادها للمتحول **sCommand** هي

- ١ - **ForeColor**: لون الخط..حيث يمكن عبر هذه التعليمة معرفة لون خط النص المحدد أو تغييره
- ٢ - **BackColor**: لون تظليل الخط..حيث يمكن عبر هذه التعليمة معرفة لون تظليل خط النص المحدد أو تغييره
- ٣ - **Bold**: تحويل النص المحدد إلى عريض
- ٤ - **Italic**: تحويل النص المحدد إلى مائل
- ٥ - **Underline**: إضافة خط أسفل النص المحدد
- ٦ - **Copy**: نسخ النص المحدد إلى الذاكرة
- ٧ - **Cut**
- ٨ - **Paste**
- ٩ - **InsertHorizontalRule**: إدراج خط فاصل أفقي

١٠ - **InsertImage** إدراج صورة ، وباختيار قيمة **bUserInterface** تساوي الواحد تظهر لنا واجهة يمكن عبرها

تحديد المسار والنص البديل **alt** والحدود **borders** الخ

١١ - **InsertMarquee** تحويل النص المختار لشريط متحرك **Marquee**

١٢ - **InsertSelectDropDown** يضيف قائمة منسدلة للمنطقة المختارة

١٣ - **Print** أمر الطباعة

١٤ - **Refresh**

١٥ - **CreateLink** إدراج رابط إلى النص المختار، والنافذة التي تظهر في حال **bUserInterface=1** يمكن عبرها

اختيار نوع البروتوكول **http, ftp, mailto:** إضافة لإدراج الرابط

١٦ - **InsertUnorderedList** إدراج قائمة منقطة

١٧ - **InsertorderedList** إدراج قائمة مرقمة

الإضافة الثانية التي أضافتها مايكروسوفت هي الاطارات الداخلية **iframes** وجعلها ديناميكية وقابلة للبرمجة والتعديل عبر خاصية **designMode**

وحتى لا نفصل كثيرا في الموضوع لنفتح برنامج تحرير صفحات الويب ولنكتب الأسطر التالية:

```
<html> <head>
<title> Using execCommand to bold text </title>
<script language="JavaScript">
```

```
function Init() {
WYSIWYG.document.designMode = 'off';
}
</script>
```

```
<body onLoad="Init()">
```

```
<iframe id="WYSIWYG" style="width: 100px; height:100px"></iframe>
```

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

حاول وضع الفأرة في الإطار الناتجة والكتابة..ستجد أن الصفحة لا تستجيب

غير السطر **WYSIWYG.document.designMode = 'off'**؛ إلى

WYSIWYG.document.designMode = 'on'

وكرر المحاولة ستجد أنك أصبحت قادرا على كتابة نص داخل الإطار الظاهر..

تهانينا...لقد بدأت المشوار..

<script> أضف التابع التالي داخل المنطقة

```
function doBold() {  
wysiwyg.document.execCommand('bold', 0, null);  
}
```

والسطر التالي بعد وسوم iframe

<input type="button" onClick="doBold()" value="Make Me Bold!">

بحيث يصبح الناتج

```
<html> <head>  
<title> Using execCommand to bold text </title>  
<script language="JavaScript">
```

```
function Init() {  
WYSIWYG.document.designMode = 'on';  
}
```

```
function doBold() {  
wysiwyg.document.execCommand('bold', 0, null);  
}
```

```
</script>  
<body onLoad="Init()">
```

<iframe id="WYSIWYG" style="width: 100px; height:100px"></iframe>

```
<input type="button" onClick="doBold()" value="Make Me Bold!">  
</body> </html>
```

الآن اكتب نصا داخل الإطار ومن ثم قد بتحديدده واضغط الزر ستلاحظ أن النص أصبح عريضا

italic, underline, subscript, Superscript

doItalic(), doUnderline(), doSubscript(), doSuperscript()

ثانينا ..لقد حصلت على أول محرر نصوص من صنعك..

في المرحلة التالية سنحاول تنسيق المحرر وإظهاره بصورة أكثر احترافية باستبدال الأزرار بصورة وجعلها تستجيب للأحداث
onmouseover, onmousedown

تبدو الأزرار العادية مملة في واجهة محرر النصوص، فما رأيكم باستبدالها بصورة؟

التعليمة التالية تجعل الزر يتحول إلى صورة

```

```

نلاحظ في السطر عدة توابع وتعريفات

الأمر الأول هو alt="Bold" وهو النص البديل الذي يظهر في حال عدم تحميل الصورة أو تلميح الأيقونة وهو النص الذي يظهر
عند اقتراب الفأرة من الأيقونة

Class="buttClass" معرف يستخدم لتعريف جميع الأزرار بحيث أنها تخضع لنفس التنسيق (وسنقوم لتعريف التنسيق لاحقا في
الجزء style من الصفحة

src="img/bold.gif" عنوان الصورة

onMouseOver="selOn(this)"

onMouseOut="selOff(this)"

onMouseDown="selDown(this)"

onMouseUp="selUp(this)"

أحداث تغيير شكل الأيقونة عند تحرك الفأر فوقها

وأخيرا "doBold ()" onClick= وهو الحدث الرئيس للزر

الآن في المجال HEAD ادخل التعليمات التالية

```
<script type="text/javascript">

function selOn(ctrl) {

    ctrl.style.borderColor = '#000000';

    ctrl.style.backgroundColor = '#B5BED6';

    ctrl.style.cursor = 'hand';

}

function selOff(ctrl) {

    ctrl.style.borderColor = '#DBE6F3';

    ctrl.style.backgroundColor = '#DBE6F3';

}

function selDown(ctrl) {

    ctrl.style.backgroundColor = '#8492B5';

}

function selUp(ctrl) {

    ctrl.style.backgroundColor = '#B5BED6';

}
```

```
function doBold() {
    iView.document.execCommand('bold', 0, null);
    if (EditorMode == 1)
        doVbCode();
}
</script>
```

```
<style type="text/css">
    .butClass {
        border: 1px solid;
        border-color: #DBE6F3;
    }
</style>
```

التوابع التي أدخلناها تقوم بتغيير التنسيق **style** للأزرار باختلاف وضعية الفأرة.. أعتقد أن التعليمات مفهومة ولا داع للتفصيل فيها

تحدون في المرفق ملف فيه بعض الأيقونات التي سنستخدمها مؤقتا..

على أي حال.. دعونا الآن نبسط الأمور ونرسم جدولا ندخل فيه الأيقونات الضرورية ونسند لكل حدث يجري عليها تابعا مناسباً

```
<html>
<head>
<title> WYSIWYG Rich Text Editor </title>
<script type="text/JavaScript" language="JavaScript">
```

```
function Init() {
    wysiwyg.document.designMode = 'On';
}

function selOn(ctrl) {
    ctrl.style.borderColor = '#000000';
    ctrl.style.backgroundColor = '#B5BED6';
```

```
    ctrl.style.cursor = 'hand';
}

function selOff(ctrl)  {
    ctrl.style.borderColor = '#DBE6F3';
    ctrl.style.backgroundColor = '#DBE6F3';
}

function selDown(ctrl)  {
    ctrl.style.backgroundColor = '#8492B5';
}

function selUp(ctrl)  {
    ctrl.style.backgroundColor = '#B5BED6';
}


function doBold()  {
    wysiwyg.document.execCommand('bold', 0, null);
}

function doItalic()  {
    wysiwyg.document.execCommand('italic', 0, null);
}

function doUnderline()  {
    wysiwyg.document.execCommand('underline', 0, null);
}

function doLeft()  {
    wysiwyg.document.execCommand('justifyleft', 0, null);
}

function doCenter()  {
    wysiwyg.document.execCommand('justifycenter', 0, null);
}

function doRight()  {
    wysiwyg.document.execCommand('justifyright', 0, null);
}

function doOrdList()  {
    wysiwyg.document.execCommand('insertorderedlist', 0,
null);
```

```

    }

    function doBulList()  {
        wysiwyg.document.execCommand('insertunorderedlist', 0,
null);
    }

    function doLink()  {
        wysiwyg.document.execCommand('createlink',1, null);
    }

    function doImage()  {
        var imgSrc = prompt('Enter image location', '');

        if(imgSrc != null)
            wysiwyg.document.execCommand('insertimage', 0,
imgSrc);
    }

    function doRule()  {
        wysiwyg.document.execCommand('inserthorizontalrule', 0,
null);
    }

</script>

<style type="text/css">
    .butClass  {
        border: 1px solid;
        border-color: #DBE6F3;
    }

    .tdClass  {
        padding-left: 3px;
        padding-top:3px;
    }
</style>

<body onLoad="Init()">
    <table id="tblCtrls" width="415px" height="30px"
border="0" cellspacing="0" cellpadding="0"
bgcolor='#DBE6F3'>
    <tr>

```

```

        <td class="tdClass">
            
            
            

            
            
            

            
            

            
            
    

    </td>
</tr>
</table>
<iframe id="wysiwyg" style="width: 415px;
height:205px"></iframe>

</body>
</html>

```

المحرر الذي نمتلكه الآن يقوم بالأغراض الأساسية لمحررات WYSIWYG لكن أحيانا قد تتداخل التعليمات فتصبح الصفحة مشوهة ..وهنا لا بد من التدخل لإجراء عمل جراحي على مستوى HTML.. فكيف نسمح للمحرر أن يعرض لنا كود HTML؟

للأداة iframe خاصتان يمكن خلالهما تحويل النص إلى تعليمات HTML والعكس وهما innerHTML و innerText

فمثلا التعليمة

```
wysiwyg.document.body.innerHTML = "<b>السلام عليكم</b>";
```

ستعطينا " السلام عليكم" بخط عريض..بينما التعليمة:

```
wysiwyg.document.body.innerText = "<b>السلام عليكم</b>";
```

فستعطينا النص السلام عليكم كما هو.

ميزة هاتين الخاصتين أنهما قابلتان للقراءة والكتابة فمثلا

```
wysiwyg.document.body.innerHTML = "<b>السلام عليكم</b>";
```

```
var MyHTML = wysiwyg.document.body.innerHTML;
```

```
wysiwyg.document.body.innerText= MyHTML;
```

ستعطينا النص **السلام عليكم** كما هو.وبالتالي للحصول على نص تعليمات HTML كل ما علينا هو ان نحصل على قيمة innerHTML ونسندھا إلى innerText .. وبعد التعديل نقوم بالعكس

الآن لنضيف زرا يقوم بالتعديل المناسب

أضف في آخر قائمة الأزرار الزر التالي

```

```

وأضف التابع التالي للمنطقة script في بدايتها

```
var ISHTML = 1; // WYSIWYG

function ViewHTML()
{
    if(ISHTML == 1)
    {
        iHTML = wysiwyg.document.body.innerHTML;
        wysiwyg.document.body.innerText = iHTML;
        document.getElementById("HTMLbtn").src =
'img/mode.gif';
        wysiwyg.focus();

        viewMode = 2; // HTML
    }
    else
    {
        iText = wysiwyg.document.body.innerText;
        wysiwyg.document.body.innerHTML = iText;
        document.getElementById("HTMLbtn").src = 'img/
richtext.gif';
        wysiwyg.focus();

        ISHTML = 1; // WYSIWYG
    }
}
```

لنحلل الآن التابع خطوة خطوة

المتحول ISHTML يعبر عن حالة المحرر، عندما تكون قيمته "1" فالمحرر يكون في حالة **wysiwyg** أي أنك لو ضغطت على الزر **bold** فإن النص المختار سيصبح عريضاً، بينما عندما تكون قيمته "2" فإن المحرر سيعرض تعليمات **HTML** الداخلية.

في الحالة الابتدائية وعند تحميل صفحة المحرر سيكون في وضع **ISHTML=1** أو **wysiwyg**، عند ضغط الزر سيطلب التابع **ViewHTML** حيث يفحص قيمة المتحول **ISHTML** فإذا كان **1** فإن سيسند قيمة **innerHTML** إلى **innerText** ويغير صورة الزر إلى **richtext** ثم يعيد التمكين للمحرر ويغير قيمة المتحول **ISHTML** إلى **2**

والحالة الأخرى يجري فيها الأمر بالعكس.. .. لقد حلت المشكلة وأصبحنا جاهزين لاستخدام المحرر المشكلة.

الآن أننا لو كنا في وضع **HTML** واخترنا نصاً ما وضغطنا زر **BOLD** فإن هذا النص سيصبح عريضاً، إلا أن هذا التنسيق سيختفي بمجرد ضغط الزر ثانية.. فكيف الحل

ببساطة سندرج التابع التالي في المنطقة **script**

```
function doHTML()  
{  
  
    var iHTML = wysiwyg.document.body.innerHTML;  
  
    iHTML = iHTML.replace( /&lt;/g, '<' ) ;  
  
    iHTML = iHTML.replace( /&gt;/g, '>' ) ;  
  
    wysiwyg.document.body.innerText = iHTML;  
  
    wysiwyg.focus();  
  
}
```

ملاحظة : سنتحدث عن التعليمة replace بالتفصيل لاحقا، ما يهم معرفته الآن أنها تستبدل قيمة الرمزين < , > بما يوافقهما من وسوم HTML

ندرج السطر التالي في كل تابع لزر

```
if (ISHTML ==2)
```

```
doHTML();
```

بحيث يصبح تابع doBold كالتالي

```
function doBold()
```

```
{  
wysiwyg.document.execCommand('bold', 0, null);
```

```
if (ISHTML==2)
```

```
doHTML ();
```

```
}
```

السطر الأخير يجبر المحرر على إعادة قراءة محتوى innerHTML وإسنادها إلى innerText بحيث تتحول التعليمة bold إلى نص..

بعض المحررات تقوم بتغيير panel كلها فتخفي الأزرار وتظهر أزرار أخرى تقوم بلف wrap النص المختار بالتعليمة المناسبة (أي إضافة وسم قبل الكلمة و بعدها) لكنني وجدت أن هذه الطريقة أسهل

تجدون في المرفق المحرر كاملا حتى نهاية هذه المرحلة

النص النهائي سيكون

```
<html>  
<head>  
<title> Browser Based HTML Editor </title>  
<script type="text/JavaScript" language="JavaScript">  
  
var ISHTML = 1; // WYSIWYG  
function ViewHTML()  
{
```

```

    if(ISHTML == 1)
    {
        iHTML = wysiwyg.document.body.innerHTML;
        wysiwyg.document.body.innerText = iHTML;
        document.getElementById("bHTML").src =
"img/richtext.gif";
        wysiwyg.focus();

        ISHTML = 2; // HTML
    }
    else
    {
        iText = wysiwyg.document.body.innerText;
        wysiwyg.document.body.innerHTML = iText;
        document.getElementById("bHTML").src = 'img/mode.gif';
        wysiwyg.focus();

        ISHTML = 1; // WYSIWYG
    }
}

```

```

function doHTML()
{
    var iHTML = wysiwyg.document.body.innerHTML;
    iHTML = iHTML.replace( /&lt;/g, '<' ) ;
    iHTML = iHTML.replace( /&gt;/g, '>' ) ;
    wysiwyg.document.body.innerText = iHTML;
    wysiwyg.focus();
}

```

```

function Init()
{
    wysiwyg.document.designMode = 'On';
}

```

```

function selOn(ctrl)
{
    ctrl.style.borderColor = '#000000';
    ctrl.style.backgroundColor = '#B5BED6';
    ctrl.style.cursor = 'hand';
}

```

```

function selOff(ctrl)
{

```

```
ctrl.style.borderColor = '#DBE6F3';
ctrl.style.backgroundColor = '#DBE6F3';
}

function selDown(ctrl)
{
    ctrl.style.backgroundColor = '#8492B5';
}

function selUp(ctrl)
{
    ctrl.style.backgroundColor = '#B5BED6';
}

function doBold()
{
    wysiwyg.document.execCommand('bold', 0, null);
    if (ISHTML==2)
        doHTML\(\);
}

function doItalic()
{
    wysiwyg.document.execCommand('italic', 0, null);
    if (ISHTML==2)
        doHTML\(\);
}

function doUnderline()
{
    wysiwyg.document.execCommand('underline', 0, null);
    if (ISHTML==2)
        doHTML\(\);
}

function doLeft()
{
    wysiwyg.document.execCommand('justifyleft', 0, null);
    if (ISHTML==2)
        doHTML\(\);
}
```

```

function doCenter()
{
    wysiwyg.document.execCommand('justifycenter', 0, null);
    if (ISHTML==2)
        doHTML\(\);
}

function doRight()
{
    wysiwyg.document.execCommand('justifyright', 0, null);
    if (ISHTML==2)
        doHTML\(\);
}

function doOrdList()
{
    wysiwyg.document.execCommand('insertorderedlist', 0,
null);
    if (ISHTML==2)
        doHTML\(\);
}

function doBulList()
{
    wysiwyg.document.execCommand('insertunorderedlist', 0,
null);
    if (ISHTML==2)
        doHTML\(\);
}

function doLink()
{
    wysiwyg.document.execCommand('createlink',1, null);
    if (ISHTML==2)
        doHTML\(\);
}

function doImage()
{
    var imgSrc = prompt('Enter image location', '');

    if(imgSrc != null)

```

```

        wysiwyg.document.execCommand('insertimage', 0,
imgSrc);
        if (ISHTML==2)
            doHTML();
    }

    function doRule()
    {
        wysiwyg.document.execCommand('inserthorizontalrule', 0,
null);
        if (ISHTML==2)
            doHTML();
    }

```

</script>

<style type="text/css">

.butClass

{

border: 1px solid;

border-color: #DBE6F3;

}

.tdClass

{

padding-left: 3px;

padding-top:3px;

}

</style>

</head>

<body onLoad="Init()">

<table id="tblCtrls" width="415px" height="30px"

border="0" cellspacing="0" cellpadding="0"

bgcolor='#DBE6F3'>

<tr>

<td class="tdClass">


```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

        
    </td>
</tr>
</table>
<iframe id="wysiwyg" style="width: 415px;
height:205px"></iframe>

</body>
</html>

```

سأقوم بإضافة عدد من الأزرار التي تستخدم نفس النهج `execCommand` وأقوم بترتيب الملفات المرفقة بحيث أضع الصور في مجلد خاص بها والسكربتات في مجلد آخر وهكذا

الملف المرفق يحتوي الشكل النهائي للمحرر في هذه المرحلة.

الجزء الأول: يختار اسم الخط الذي يكتب به، ويتم ذلك عبر قائمتين منسدلتين لتحديد نوع الخط وقياسه عند تغييرهما ينشطان التابع الذي يقوم بتمرير قيمتهما للنهج `execCommand` المناسب.

حدث التغيير

```

1- <select name="selFont"
    onChange="doFont(this.options[this.selectedIndex].value)" ">
2- <select name="selSize"
    onChange="doSize(this.options[this.selectedIndex].value)" ">
3- <select name="selHeading"
    onChange="doHead(this.options[this.selectedIndex].value)" ">

```

والتوابع هي

```

function doFont(fName)
{

```

```

        if(fName != '')
            wysiwyg.document.execCommand('fontname', false,
fName);
            if (ISHTML==2)
                doHTML();
    }

    function doSize(fSize)
    {
        if(fSize != '')
            wysiwyg.document.execCommand('fontsize', false,
fSize);
            if (ISHTML==2)
                doHTML();
    }

    function doHead(hType)
    {
        if(hType != '')
        {
            wysiwyg.document.execCommand('formatblock', false,
hType);

            doFont(selFont.options[selFont.selectedIndex].value);
            if (ISHTML==2)
                doHTML();
        }
    }
}

```

الجزء الثاني : أدوات التحرير الاعتيادية كالنسخ والقص واتجاه التحرير وغيرها..ولا داع للتفصيل فيها فهي تشبه سابقاتها

الجزء الثالث :نافذة تحديد لون الخط ولون التظليل **highlight** حيث سنحتاج لنافذة يتم عبرها اختيار اللون

قمت باستخدام نافذة اختيار لون جاهزة، ولكننا سنحاول إلقاء الضوء على الكود الخاص بها فيما يلي:

الجزء الأول الكود الذي يستدعي نافذة اختيار اللون.. والتي تقوم بدورها بتمرير قيمة اللون المختار إلى التابع بشكل ستعشري

```

function doForeColor()

```

```

{
    var posX      = event.screenX;
    var posY      = event.screenY ;
    var screenW   = screen.width;           // screen
size
    var screenH   = screen.height - 20;     // take
taskbar into account
    var wPosition = 'dialogLeft:' +posX+ ' ; dialogTop:'
+posY;

    var newcolor = showModalDialog("popups/color.html",
'', "resizable: no; help: no; status: no; scroll: no;" +
wPosition);
    if (newcolor != null) {
wysiwyg.document.execCommand('forecolor', false,
"#"+newcolor); }
    if (ISHTML==2)
        doHTML() ;
}

```

الجزء الأول من التابع يحدد إحداثيات الفأرة (وبما أن الفأرة ضغطت على الزر فعلمنا هو يحدد إحداثيات الزر المرتبط بالحدث) ثم يمرر قيمة هذه الإحداثيات إلى التابع الذي سيفتح النافذة بحيث تظهر الزاوية العليا اليسرى من هذه النافذة عند رأس الفأرة تقريبا

النهج `showModalDialog` يقوم بفتح النافذة مع تحديد خصائصها.. الصيغة التي يكتب وفقها هذا النهج هي:

```
var = window.showModalDialog(sURL [, vArguments] [, sFeatures])
```

sURL: هو مسار النافذة المطلق او النسبي

vArguments : متحول من أي نوع (حتى لو كان مصفوفة) يمكن تمريره للنافذة المفتوحة، ويتم الوصول إليه من النافذة

المفتوحة عبر الخاصية `dialogArguments` من الكائن `window` .. يمكننا تمرير المتحول `oldcolor` وهو لون الخط القديم ليتم عرضه في نافذة اختيار اللون كلون مبدئي. لكننا اختصرنا هذا العمل للتبسيط

sFeatures: متحول آخر يصف شكل النافذة المفتوحة، ويحتوي على عدة خيارات منها

أبعاد النافذة: `dialogWidth` ، `dialogTop` ، `dialogLeft` ، `dialogHeight` ،

حالة النافذة :

متمركزة { `yes` | `no` | `1` | `0` | `on` | `off` } `center:`

resizable:{ yes | no | 1 | 0 | on | off } قابلة لتغيير الحجم

scroll:{ yes | no | 1 | 0 | on | off } عرض زر التصفح الأفقي

status:{ yes | no | 1 | 0 | on | off } عرض زر الحالة

window.returnValue النافذة التي تظهر يمكن أن تمرر للتابع الذي استدعاها قيمة ما عبر الأمر

وبالتالي فالمتحول newcolor يحتوي على قيمة اللون الجديد المختار

لنلق نظرة على التوابع في النفاذة المفتوحة color.html

التابع الأول والثاني تابعا التنشيط init ومهمتهما جعل الصفحة تستجيب لحدث ضغط الزر Esc فتغلق نفسها بنفسها،

```
function Init() {  
  // run on page load  
  document.body.onkeypress = _CloseOnEsc;  
}
```

```
function _CloseOnEsc() {  
  if (event.keyCode == 27) { window.close(); return; }  
}
```

التابع الثالث تابع عرض اللون المختار في نافذة العرض الصغيرة

```
function View(color) { // preview color  
  document.all.ColorPreview.style.backgroundColor = '#' +  
color;  
  document.all.ColorHex.value = '#' + color;  
}
```

التابع الرابع مهمته التأكد من صحة الترميز اللوني بحيث أن المستخدم لو أدخل لونا ما مباشرة في خانة إدخال اللون وكان هذا اللون خطأ (كأن يكتب اسم اللون مثلا أو يكتب رقما بأكثر من ست خانات) لم تستجب الصفحة

```
function ValidateColor(string) { // return  
valid color code  
  string = string || '';  
  string = string + '';  
  string = string.toUpperCase();  
  chars = '0123456789ABCDEF';
```

```

out    = '';

for (i=0; i<string.length; i++) {                // remove
invalid color chars
    schar = string.charAt(i);
    if (chars.indexOf(schar) != -1) { out += schar; }
}

if (out.length != 6) { return null; }              // check
length
return out;
}

```

وأخيرا تابع إرسال اللون إلى الصفحة الأم بعد التأكد منه وعرضه في نافذة استعراض اللون

```

function Set(string) {                            // select color
    color = ValidateColor(string);
    if (color == null) { alert("Invalid color code: " +
string); }    // invalid color
    else {
// valid color
        View(color);                            // show selected
color
        window.returnValue = color;              // set return
value
        window.close();                          // close dialog
    }
}

```

ثم تتدرج الألوان المختلفة في خلايا جدول محدد بحيث تستجيب لحدث مرور الفأرة فوقها فتعرض اللون الخاص بها في نافذة العرض أو تستجيب لحدث النقر فتُرسل اللون إلى النافذة الأم وتغلق النافذة المنبثقة.

زر إضافة بحث جوجول أو ويكيبيديا:

لو أنك كتبت في مستكشف الانترنت العبارة التالية:

<http://www.google.com/search?q=KEYWORD>

فإنك ستحصل على صفحة جوجول وفيها النتائج المتعلقة بالكلمة المفتاح **KEYWORD**، وبالتالي لو أننا أخطنا كلمة ما برابط يحتوي على الجملة السابقة لأصبحت الكلمة **hyperlink** تحصل على تفاصيل عنها بمجرد الضغط عليها..

التابع googleit() يقوم بهذه المهمة ومثله التابع doWiki()

```
function doGoogleit() {  
    var vKeyWord = prompt('Enter KeyWord ','');  
    if(vKeyWord != null && vKeyWord != '')  
    {    iView.focus ();  
        vKeyWord = 'http://www.google.com/search?q=' + vKeyWord;  
        iView.document.execCommand('createlink', false ,vKeyWord);  
    } else {  
        alert("A keyword must be specified.");  
    }  
  
    if (EditorMode == 1)  
        doVbCode();  
    if (EditorMode ==2)  
        doHTML();  
}
```

في المرحلة القادمة سنحاول إدراج أزرار تقوم بأعمال خارج نطاق النهج . .execCommand

من المناسب الآن أن نقوم بنقل جميع أكواد الجافاسكريبت إلى ملف خاص بها وذلك تسهيلا للعمل ومنعا لاختلاط اكواد HTML
باكواد Javascript

انسخ جميع الأكواد بين الوسمين <script></script> إلى ملف نصي فارغ من نوع *.txt ومن ثم سمه wysiwyg.js
مغيرا لاحقته من .txt إلى .js. ستلاحظ أن أيقونته تحولت للون الأصفر .. ضع الملف الجديد في مجلد جديد اسمه Js

الآن استبدل الوسمين <script></script> ومحتوياتهما بالوسم التالي:

```
<script type="text/javascript" src="js/wysiwyg.js"></script>
```

إدراج وسوم HTML برمجيا

بسبب محدودية إمكانيات النهج `execCommand` فمن المهم إيجاد وسيلة لإدراج وسوم HTML التي نريدها في المحرر بدون الاعتماد عليه. التابع التالي (وجدته في أحد المحررات المجانية على الانترنت) يقوم بالعمل.. لنحاول تحليل التابع خطوة خطوة

```
\* ----- */* -----
----- *\
```

Function : `editor_insertHTML`

Description : insert string at current cursor position in editor. If
two strings are specifed, surround selected text with them.

Usage : `editor_insertHTML(objname, str1, [str2], reqSelection)`

Arguments : `objname` - ID of textarea

`str1` - HTML or text to insert

`str2` - HTML or text to insert (optional argument)

`reqSelection` - (1 or 0) give error if no text selected

```
\* ----- */
```

للتابع الصيغة العامة التالية

`editor_insertHTML(objname, str1, [str2], reqSelection)`

حيث `objname` هو اسم الكائن الذي سيجري عليه التغيير

`str1` النص المراد إدراجه عند نقطة الإدراج (المكان الذي نقرت عليه الفأرة آخر مرة قبل الضغط إلى الزر)

مثلا لإدراج صورة ندرج تعليمة الـ HTML الكاملة

``

Str2 نص متمم للنص المراد إدراجه عند نقطة الإدراج (في حالة الرغبة بإدراج وسم يجعل النص يظهر كرابط فعال نكتب

Str1 =

Str2=

reqSelection يحدد ضرورة اختيار نص أو جزء من النص قبل تشغيل التابع، فمثلا لو أردت إدراج صورة فلا داع لوجود نص يتم اختياره (قيمة هذا المتحول 0)، بينما إذا رغبتنا بجعل نص ما عريضا، ما فيجب اختيار **select** هذا النص أولا (قيمة هذا المتحول 1).

صيغة التابع كالتالي:

```
function editor_insertHTML(objname, str1,str2, reqSel) {  
    var editor_obj = document.all[objname] ; // editor object  
    if (str1 == null) { str1 = ""; }  
    if (str2 == null) { str2 = ""; }  
    // for non-wysiwyg capable browsers just add to end of textbox  
    if (document.all[objname] && editor_obj == null) {  
        document.all[objname].focus();  
        document.all[objname].value = document.all[objname].value + str1 + str2;  
        return;  
    }  
    // error checking  
    if (editor_obj == null) { return alert("Unable to Do Command. Invalid object name  
    "" +objname+ ""."); }  
    editor_focus(editor_obj);  
    var tagname = editor_obj.tagName.toLowerCase();  
    var sRange;
```

```

// insertHTML for wysiwyg iframe

if (tagname == 'iframe') {

    var editdoc = editor_obj.contentWindow.document;

    sRange = editdoc.selection.createRange();

    var sHtml = sRange.htmlText;

    // check for control ranges

    if (sRange.length) { return alert(" Unable to Do Command. Try highlighting
content instead of selecting it."); }


    // insert HTML

    var oldHandler = window.onerror;

    window.onerror = function() { alert(" Unable to Do Command for current
selection."); return true; } // partial table selections cause errors

    if (sHtml.length) { // if content selected

        if (str2) { sRange.pasteHTML(str1 + sHtml + str2) } // surround

        else { sRange.pasteHTML(str1); } // overwrite

    } else { // if insertion point only

        if (reqSel) { return alert(" Please select something text first."); }

        sRange.pasteHTML(str1 + str2); // insert strings

    }

    window.onerror = oldHandler;

}

// insertHTML for plaintext textarea

else if (tagname == 'textarea') {

```

```

editor_obj.focus();

sRange = document.selection.createRange();

var sText = sRange.text;

// insert HTML

if (sText.length) { // if content selected

    if (str2) { sRange.text = str1 +sText+ str2; } // surround

    else { sRange.text = str1; } // overwrite

} else { // if insertion point only

    if (reqSel) { return alert("Please select something text first."); }

    sRange.text = str1 + str2; // insert strings

}

}

else { alert("Unable to insert HTML. Unknown object tag type '" +tagname+ "'."); }


// move to end of new content

sRange.collapse(false); // move to end of range

sRange.select(); // re-select

}

```

دراسته بالتفصيل كالتالي

نمرر إلى التابع أربع متغيرات: الكائن الذي سيجري عليه التغيير (هنا هو الـ `iframe`) والسلسلة النصية الأولى والثانية، والمتحول الذي يحدد ضرورة اختيار نص يطبق عليه التابع.. (أي يقوم بإدراج `Str1` قبله و `str2` بعده)

```
function editor_insertHTML(objname, str1,str2, reqSel) {
```

```
var editor_obj = document.all[objname] ; // editor object
```

```
if (str1 == null) { str1 = ""; }
```

```
if (str2 == null) { str2 = ""; }
```

تحويل التمكين إلى الكائن (التابع يميز بين المستعرضات التي تمتلك خاصية **wysiwyg** من غيرها)

```
// for non-wysiwyg capable browsers just add to end of textbox
```

```
if (document.all[objname] && editor_obj == null) {
```

```
    document.all[objname].focus();
```

```
    document.all[objname].value = document.all[objname].value + str1 + str2;
```

```
    return;
```

```
}
```

معالجة الأخطاء: في حالة عدم تمرير اسم الكائن أو تمريره بصورة خاطئة

```
// error checking
```

```
if (editor_obj == null) { return alert("Unable to Do Command. Invalid object name  
" + objname + "."); }
```

```
editor_focus(editor_obj);
```

```
var tagname = editor_obj.tagName.toLowerCase();
```

```
var sRange;
```

إضافة وسوم HTML

1- في حالة كان الكائن 'iframe' وكان هناك نص مختارا قابلا لتوصيف بوسوم HTML (مثلا عند اختيار جدول بدلا من النص فإن التابع لا يستطيع إجراء أي عمل عليه)

```
// insertHTML for wysiwyg iframe
```

```
if (tagname == 'iframe') {
```

```

var editdoc = editor_obj.contentWindow.document;

sRange = editdoc.selection.createRange();

var sHtml = sRange.htmlText;

// check for control ranges

if (sRange.length) { return alert(" Unable to Do Command. Try highlighting
content instead of selecting it."); }

// insert HTML

var oldHandler = window.onerror;

window.onerror = function() { alert(" Unable to Do Command for current
selection."); return true; }

```

آلية العمل هي يجعل النص في متغير نصي string ومن ثم إضافة المتغيرين str1 & str2 قبله وبعده

```

// partial table selections cause errors

if (sHtml.length) { // if content selected

    if (str2) { sRange.pasteHTML(str1 + sHtml + str2) } // surround

    else { sRange.pasteHTML(str1); } // overwrite

} else { // if insertion point only

    if (reqSel) { return alert(" Please select something text first."); }

    sRange.pasteHTML(str1 + str2); // insert strings

}

window.onerror = oldHandler;

}

```

2- في حال كان الكائن حقل نص textarea فإن التعديل يتم بإضافة Str1 & str2 كنص وليس كوسم HTML

```

// insertHTML for plaintext textarea

```

```

else if (tagname == 'textarea') {
    editor_obj.focus();

    sRange = document.selection.createRange();

    var sText = sRange.text;

    // insert HTML

    if (sText.length) { // if content selected

        if (str2) { sRange.text = str1 +sText+ str2; } // surround

        else { sRange.text = str1; } // overwrite

        insertion point 3- إذا لم يكن هناك نص مختار فيتم لصق str1 &str2 في نقطة الإدراج

    } else { // if insertion point only

        if (reqSel) { return alert("Please select something text first."); }

        sRange.text = str1 + str2; // insert strings

    }

}

```

4- إذا كان مكان الإدراج ليس ifraem ولا textarea فتجاهل الأمر:

```

else { alert("Unable to insert HTML. Unknown object tag type '" +tagname+ "'."); }

// move to end of new content

sRange.collapse(false); // move to end of range

sRange.select(); // re-select

}

```

التابع السابق يمنحنا الكثير من الإمكانيات التي تعجز عنها العبارة **execCommand** وسنرى فيما يأتي تطبيقات هذا التابع...

زر الاقتباس

لنبدأ بمثال بسيط جدا وهي علامة الاقتباس (أو إحاطة نص ما بإطار)، أضف الزر التالي إلى قائمة الأزرار

```

```

وأضف التابع التالي إلى الملف wysiwyg.js

```
function doQuote()
{
    iView.focus ();

    editor_insertHTML('wysiwyg','<FIELDSET><legend>&nbsp;Quote:
&nbsp;</legend>','</FIELDSET>',1);

    if (ISHTML==2)
        doHTML();
}
```

قم الآن بتشغيل المحرر وكتابة أي نص في نافذته ومن ثم اختياره والضغط على زر الاقتباس.. ستلاحظ إحاطة النص بإطار..

لاحظ انك لو نقرت على زر الاقتباس أولا فستظهر رسالة الخطأ.

زر إضافة ملف فيديو:

قم بإضافة السطر التالي لنهاية قائمة الأزرار

```



```

وأضف التابع التالي إلى الملف wysiwyg.js

```
function doVideo(val)

{   var Media = showModalDialog("popups/video.html?" + val, window,
"resizable: yes; help: no; status: no; scroll: no; dialogWidth:350px ;
dialogHeight:120px ");

    if(Media != null)

        iView.focus ();

        editor_insertHTML('wysiwyg', Media ,",0);

    if (ISHTML==2)

        doHTML();

}
```

والملف video.HTML إلى المجلد popups

يحتوي الملف video.HTML على واجهة يمكن عبرها تحديد مسار وأبعاد ملف الفلاش المدرج..

سنستخدم نفس النافذة لاحقاً في إدراج ملفات فيديو من نوع: Media Player Video WMV

و ملفات Real Player RM

بالإضافة للملفات Google و YouTube

يقوم الزر flash بتمرير القيمة "flash" إلى التابع doVideo والذي يقوم بدوره بتمريرها إلى النافذة video.html و ينتظر منها أن تمرر له قيمة وسم الـ HTML الذي سيستخدمه..

طريقة تمرير القيمة Flash هو بوضعها بعد اسم الصفحة وإشارة الاستفهام

```
popups/video.html?" + val
```

التابع الرئيس في الملف video.html هو Doit

```
function Doit()    {

var vType = location.search.substring(1,location.search.length);
```

```

var vTag;

var vURL = ""

var vHeight=0

var vWidth =0

vURL = document.forms(0).txtUrl.value;

vHeight = document.forms(0).txtHeight.value;

vWidth = document.forms(0).txtWidth.value;

    if (vURL != null && vURL != "http://") {

        if (vHeight ==0 && vWidth ==0) {

            switch(vType)    {

                case 'Flash': vTag = '[Flash]' + vURL + '[/Flash]';

                                break;

                case 'Google': vTag = '[Google]' + vURL + '[/Google]';

                                break;

                case 'YouTube': vTag = '[YouTube]' + vURL + '[/YouTube]';

                                break;

                case "": vTag = ""

                                }

            window.returnValue = vTag ;

            window.close ();

        } else if (isInteger(vHeight) && isInteger(vWidth))    {

            switch(vType){

                case 'Flash': vTag = '[Flash width=' +vWidth + ' height=' + vHeight +']'

+ vURL + '[/Flash]';

```

```

        break;

        case 'Google': vTag = '[Google width=' + vWidth + ' height=' + vHeight
+ ']' + vURL + ' [/Google]';

        break;

        case 'YouTube': vTag = '[YouTube width=' + vWidth + ' height=' +
vHeight + ']' + vURL + ' [/YouTube]';

        break;

        case "": vTag = "
        }

        window.returnValue = vTag ;

        window.close ();

    }else{

        alert(" Please Insert Valid Dimensions");

    }

}

}

```

```

function isInteger(val)      {

    if(isNaN(val)) { return false; }

    else{

        if ((val < 1) || (val >= 600) || (val== null) ) return false;

        return true;
    }
}

```

```
    }  
}
```

لنحلل التابع خطوة بخطوة :

```
function Doit() {
```

يقوم التابع باستخلاص القيمة الممررة إليه من عنوان الصفحة Query String عبر استخدام النهج `.search.substring`.

```
    var vType = location.search.substring(1,location.search.length);
```

```
    var vTag;
```

```
    var vURL = ""
```

```
    var vHeight=0
```

```
    var vWidth =0
```

يقوم التابع بالحصول على تعريفات الملف الذي سيدرجه (المسار والأبعاد) من النموذج الموجود في الصفحة نفسها

```
    vURL = document.forms(0).txtUrl.value;
```

```
    vHeight = document.forms(0).txtHeight.value;
```

```
    vWidth = document.forms(0).txtWidth.value;
```

معالجة الأخطاء:

التأكد أن المستخدم لم ينس إدراج المسار، و في حال لم يدخل المستخدم أبعاد الملف فلا تدرج تعريف الأبعاد في الكود

```
if (vURL != null && vURL != "http://") {
```

```
    if (vHeight ==0 && vWidth ==0) {
```

```
        switch(vType) {
```

```
            case 'Flash': vTag = '[Flash]' + vURL + ' [/Flash]';
```

```
                break;
```

```
            case 'Google': vTag = '[Google]' + vURL + ' [/Google]';
```

```
                break;
```

```
case 'YouTube': vTag = '[YouTube]' + vURL +  
['/YouTube'];
```

```
break;
```

```
case "": vTag ="
```

```
}
```

```
window.returnValue = vTag ;
```

```
window.close ();
```

إما في حال أدرج المستخدم الأبعاد فتأكد من أنه لم يتم بإدخال قيمة غير رقمية أو قيمة غير مقبولة في خانة الأبعاد (التابع `isInteger` والذي سنشرحه لاحقاً).

```
else if (vURL != null && vURL != "http://")
```

```
{ if (isInteger(vHeight) && isInteger(vWidth))
```

```
{
```

عبر هذه الحلقة سنحدد نوع الملف المدرج `flash, Media, YouTube` الخ ومن ثم نحدد التعليمة المناسبة

```
switch(vType) {
```

```
case 'Flash': vTag = '[Flash width=' + vWidth + ' height=' + vHeight + ']' + vURL +  
['/Flash'];
```

```
break;
```

```
case 'Google': vTag = '[Google width=' + vWidth + ' height=' + vHeight + ']' + vURL  
+ ['/Google'];
```

```
break;
```

```
case 'YouTube': vTag = '[YouTube width=' + vWidth + ' height=' + vHeight + ']' +  
vURL + ['/YouTube'];
```

```
break;
```

```
case "": vTag ="
```

```
}
```

```
window.returnValue = vTag ;
```

```
window.close ();
```

```
}else{ alert(" Please Insert Valid Dimensions");}
```

```
}else{ alert(" Please Insert a Valid URL");}
```

```
}
```

لاحظ أننا لم ندرج وسم الفلاش الحقيقي بل أدرجنا نصا هو

```
[Flash=URL] WIDTH =? Height=? [/Flash]
```

والسبب في ذلك أننا لو أدرجنا وسم HTML مناسب فإن المحرر ستوقف عن العمل حتى استكمال فتح ملف الفلاش أو الفيديو، ومن ثم سيصبح بطيئا جدا في العمل.. وهذا أمر غير مقبول.

ما سنقوم به هو برجة ما يسمى Code Paraser وهي صفحة الـ ASP.net التي سنستخدمها لتفسير النص وإدراجه في قاعدة البيانات أو في المكان الذي سيحفظ فيه، حيث ستقوم هذه الصفحة باستبدال السطر

```
[Flash=URL] WIDTH =width HEIGHT= height [/Flash]
```

بالوسم

```
<OBJECT classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'  
codebase='http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.  
cab#version=6,0,0,0' width=" width " height=" height ">
```

```
<param name='movie' value=" URL ">
```

```
<param name='quality' value="high">
```

```
<param name='bgcolor' value='#FFFFFF'>
```

```
<param name='loop' value="true">
```

```
<EMBED src=" URL" quality='high' bgcolor='#FFFFFF' width=" width "
```

```
height=" height " loop="true" type='application/x-shockwave-flash'
```

```
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Pr  
od_Version=ShockwaveFlash'>
```

```
</EMBED> </OBJECT>
```

كما يمكن الاكتفاء بالوسم في نسخ IE الحديثة

```
<EMBED src=" URL" quality='high' bgcolor='#FFFFFF' width=" width " height=" height " loop="true" type='application/x-shockwave-flash'  
pluginspage='http://www.macromedia.com/shockwave/download/index.cgi?P1_Pr  
od_Version=ShockwaveFlash'> </EMBED>
```

التابع الثاني في video.html هو isInteger ومهمته التأكد من كون أبعاد الفيلم المدخلة هي أبعاد حقيقية مقبولة.

```
function isInteger(val){  
if(isNaN(val)) { return false; }  
else{  
if ((val < 1) || (val >= 600) || (val== null) ) return false;  
return true;}  
}
```

في الملف المرفق تجدون المحرر وفيه بقية الأزرار الخاصة بإدراج الأفلام المختلفة .

زر إضافة ملف Real أو Media:

التابع يستدعي النافذة media

```
function doMedia(val)  
{ var Media = showModalDialog("popups/media.html?" + val, window,  
"resizable: yes; help: no; status: no; scroll: no; dialogWidth:420px ;  
dialogHeight:120px ");  
if(Media != null)
```

```

iView.focus ();

editor_insertHTML('wysiwyg', Media ,",0);

if (ISHTML==2)

    doHTML();

}

```

ملاحظة: يمكن دمج التابعين `doVideo` و `doMedia` في تابع واحد.. لكنني آثرت فصلهما للتبسيط.

النافذة `media.html` تحتوي تابعا مشاهدا للسابق لكن فيها خيار تحديد كون الملف ملف مرئي أم صوتي.

حيث تظهر خيارات الأبعاد في حال اختيار ملف مرئي.

أزرار التأثيرات `glow, Shadow, Blur`

تشبه تماما الأزرار السابقة لكنها تضيف تأثيرا للنص

1- Glow

```
<SPAN style="FONT-WEIGHT: bold; FONT-SIZE: 20px; FILTER: glow(color=#??,
strength=7); HEIGHT: 20px"> Text Goes here </SPAN>
```

2- Shadow

```
<SPAN style="FONT-WEIGHT: bold; FONT-SIZE: 20px; FILTER: shadow(color=#??,
direction=225); HEIGHT: 20px"> Text Goes here </SPAN>
```

3- Blur

```
<SPAN style="FONT-WEIGHT: bold; FONT-SIZE: 20px; FILTER: blur(add=0,
direction=200, strength=??); HEIGHT: 20px"> Text Goes here </SPAN>
```

يتم استدعاء نوافذ مناسبة لتحديد المتحولات `Color` و `strength` في هذه الفلاتر

زر إضافة الوسم `CODE`:

مهمة هذا الزر أن يحيط جزءا من النص بالوسم `code` لمنع تفسيره من قبل المفسر. حيث أنه يعامل كنص وليس كوسم فتحول الأجزاء `< >` منه إلى `<` و `>` وبالتالي يعرض كنص بعد خزنه في قاعدة البيانات

```
function doCode() {  
    wysiwyg.focus ();  
  
    editor_insertHTML('wysiwyg','<FIELDSET><legend>&nbsp;CODE:&nbsp;</legend  
> [CODE]<br>' , '<br>[/CODE]</FIELDSET>',1);  
  
    if (ISHTML==2)  
        doHTML();  
}
```

زر إضافة الوجوه الباسمة

مهمة هذا الزر فتح نافذة الوجوه الباسمة، ولهذه النافذة قصة طويلة سنشرحها بالتفصيل

```
function doSmilies(){  
    var mCommand, ulInterface, vValue;  
    var posX  = event.screenX;  
    var posY  = event.screenY ;  
    var screenW = screen.width;           // screen size  
    var screenH = screen.height - 20;     // take taskbar into  
account  
    var wPosition  = 'dialogLeft:' +posX+ ' ; dialogTop:' +posY;  
    var newimage = showModalDialog('popups/smilies.html', '', 'resizable:  
yes; help: no; status: no; scroll: yes; ' + wPosition);  
    if (newimage == null) return;
```

```

mCommand = 'insertimage';

vValue = newimage;

ulInterface = false;

editor_insertHTML('iView', newimage ,",0);

}

```

تم بيان جزء كبير من التابع .. الجزء الأول منه يفتح النافذة **smilies.html** قريبا من أيقونة الزر **smilies** ..

الجزء الثاني يفتح النافذة وينتظر تلقي المتحول **newimage** منها.. وأخيرا يتم تحرير المتحول للتابع **editor_insertHTML** السابق

النافذة **smilies.html**

ستعلمنا هذه النافذة التعامل مع ملفات **xml** عبر الجافاسكريبت..

لمن لا يعرف ملفات الـ **xml** فهي نوع من قواعد البيانات النصية التي يتم تعريفها من قبل المستخدم

فهي تشبه الجداول والأعمدة.. مثلا لإنشاء قاعدة بيانات اسمها **db** وفيها جدول اسمه **Author** وهذا الجدول فيه أعمدة

AuthorID , Name, Book نكتب التالي في مستند نصي عادي ومن ثم نحول لاحقته إلى **xml**.

(ملاحظة: ليس من الضرورة أن تكون اللاحقة **xml**.. فأحيانا يمكن توليد جدول الـ **xml** داخل صفحة **aspx**. وقراءته وكأنه

(**xml**)

<db>

< author>

< authorID > 1</ authorID >

<name> Muhammad </name>

<book> I can fly </book>

< /author>

< author>

< authorID > 2</ authorID >

<name> Khaled </name>

```
<book> I cannot fly </book>
< /author>
</db>
```

يبدو واضحاً أنه علينا تكرار اسم الجدول لكل سطر..

ما تمنحه **xml** هو أننا يمكننا تعشيش **Nesting** الجداول في بعضها، بما يشبه ربط الجداول في **access**.. مثلاً

```
<db>

< author>
  < authorID > 2</ authorID >
  <name> Khaled </name>
  <books>
    <book> I can fly </book>
    <book> I cannot fly </book>
    <book> I can swim </book>
    <book> I cannot swim </book>
  </books>
< /author>
</db>
```

طبعا هذه لمحة مبسطة عن **xml** وليس المجال هنا متاحاً للتفصيل فيها

سنقوم نحن بصنع ملف **xml** نخزن فيه الوجوه الباسمة ومساراتها ومختصراتها.. وهذا جزء منه:

```
<emoticons>
  <smilie>
    <path>group1/angry.png</path>
    <emeID>:{</emeID>
  </smilie>
```

```

<smilie>
    <path>group1/blink.png</path>
    <emeID>:2</emeID>
</smilie>
</emoticons>

```

ميزة هذا العمل أننا لا نحتاج لتعديل ملف **smilies.html** لإدراج صورة فيه، بل يقوم تابع جافاسكريبت بقراءة الملف **emoticons.xml** ورسم الجدول المناسب لعرض الوجوه داخله، وبالتالي كلما أضفنا سطرا للملف، أضيف وجه باسم للصفحة بشكل تلقائي..

التابع الذي يقوم بهذا – والموجود في الجزء **body** وليس **head** من الصفحة – هو:

```

var xmlDoc=null;
var perrow = 4 ;// how many smilies to present in each row
if (window.ActiveXObject) { // code for IE
xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
} else if (document.implementation.createDocument) { // code for Mozilla, Firefox,
Opera, etc.
xmlDoc=document.implementation.createDocument("", "", null);
} else { alert('Your browser cannot handle this script'); }
if (xmlDoc!=null) {
    xmlDoc.async=false;
    xmlDoc.load("emoticons.xml");
    var j =0
    var x=xmlDoc.getElementsByTagName("smilie");
    for (j=0;j<perrow ;j++) {
        document.write("<TD class='tdClass'><strong>Clip </strong> </TD>");
        document.write("<TD class='tdClass'><strong>Shortcut </strong> </td>");
    }
}

```

```

j=0

document.write("<tr>");

for (i=0;i<x.length;i++){

    document.write("<TD class='style1'><img class='img' ID='"+
x[i].getElementsByTagName("emID")[0].childNodes[0].nodeValue + "' src='smilies/" +
x[i].getElementsByTagName("path")[0].childNodes[0].nodeValue + "'");

    document.write(" OnClick=Set('" +
x[i].getElementsByTagName("path")[0].childNodes[0].nodeValue + "'," +
x[i].getElementsByTagName("emID")[0].childNodes[0].nodeValue + "')></TD>"
);

    document.write("<TD class='tdClass'>" +
x[i].getElementsByTagName("emID")[0].childNodes[0].nodeValue + "</td>");

j++;

if (j== perrow ) {

    j=0;

    document.write("</tr><tr>");

    }

}

}

```

لنشرح التابع خطوة خطوة

تعريف الملف xml ولتغير يحدد عدد الوجوه الباسمة في كل سطر

```
var xmlDoc=null;
```

```
var perrow = 4 ;// how many smiles to present in each row
```

كود تعريف ان المتحول xmlDoc هو ملف xml وفقا لـ IE والذي يقبل ActiveXObject

```
if (window.ActiveXObject) { // code for IE
```

```
xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
```

الكود السابق لا تفهمه بقية المستعرضات، لهذا نبحث في وجود النهج `createDocument` الذي يميز مستعرضات Firefox & Mozilla:

```
} else if (document.implementation.createDocument) { // code for Mozilla, Firefox, Opera, etc.
```

```
xmlDoc=document.implementation.createDocument("", "", null);
```

إذا كان المستعرض من نوع آخر.. فأنس الموضوع.. وعقوبة مستعمل هذا المستعرض الحرمان من الوجوه الباسمة!

```
} else { alert('Your browser cannot handle this script'); }
```

نقوم الآن بفتح الملف `emoticons.xml` بكائن `xmlDoc` الذي أنشأناه.

```
if (xmlDoc!=null) {
```

```
    xmlDoc.async=false;
```

```
    xmlDoc.load("emoticons.xml");
```

نقرأ الملف باعتبار أسماء الجداول والأسطر والأعمدة وكأنها وسوم HTML، وبالتالي نصل إليه عبر

`getElementsByTagName`، ومن ثم وعبر حلقة `for` نرسم الجدول المناسب لاحتواء الوجوه.

```
var j =0
```

```
var x=xmlDoc.getElementsByTagName("smilie");
```

```
for (j=0;j<perrow ;j++) {
```

```
    document.write("<TD class='tdClass'><strong>Clip </strong> </TD>");
```

```
    document.write("<TD class='tdClass'><strong>Shortcut </strong> </td>");
```

```
}
```

```
j=0
```

```
document.write("<tr>");
```

```
for (i=0;i<x.length;i++){
```

```
    document.write("<TD class='style1'><img class='img' ID='"+
```

```
x[i].getElementsByTagName("emeID")[0].childNodes[0].nodeValue + " src='smilies/" +
```

```
x[i].getElementsByTagName("path")[0].childNodes[0].nodeValue + "'");
```

```

document.write(" OnClick=Set('" +
x[i].getElementsByTagName("path")[0].childNodes[0].nodeValue + "','" +
x[i].getElementsByTagName("emelD")[0].childNodes[0].nodeValue + "')></TD>"
);

document.write("<TD class='tdClass'>" +
x[i].getElementsByTagName("emelD")[0].childNodes[0].nodeValue + "</td>");

j++;

if (j== perrow ) {

    j=0;

    document.write("</tr><tr>");

    }

}

}

```

لنفسر النقطة الأهم في التابع أعلاه: المتغير **x** هو مصفوفة **array** تحتوي جميع الأسطر **smile** في ملف

```
var x=xmlDoc.getElementsByTagName("smile");
```

ولكن كيف نصل لمحتويات الأعمدة **emelD** و **path** ؟

ببساطة نعامل **x** على أنها صفحة **html** ونقرأ **emelD** و **path** وكأنها وسوم **HTML** عبر **getElementsByTagName**

```
x[i].getElementsByTagName("emelD")[0].childNodes[0].nodeValue
```

```
x[i].getElementsByTagName("path")[0].childNodes[0].nodeValue
```

أما الجزء **childNodes[0]** فيما لو كان هناك جدول معشش داخل إحدى الوسوم..مثلا

```

<smilie>

    <path>group1/blink.png</path>

    <emelD>

        <ID1> 1</ID1>

        <ID2> 2</ID2>

    </emelD>

```

</smilie>

حيث نصل لـ ID1 عبر childNodes[0] ولـ ID2 عبر childNodes[1]. وهكذا.

أعتقد أن التابع أصبح واضحا الآن..

المرحلة الثانية

إنشاء محرر النصوص عبر تابع javascript

رغم وجود بعض الأزرار الإضافية إلا أنني لن أناقشها خوفا من الملل

السؤال الآن: هل يتوجب علينا نسخ كل كود HTML البرمجي إلى كل صفحة نريد إدراج المحرر فيها؟

ماذا لو أحب أحد المستخدمين تعطيل زر أو حذفه؟ هل عليه أن يدخل إلى صفحة الـ HTML وحذف الصورة التي تمثل الزر؟

سؤال آخر... لو أننا أردنا عرض المحرر بشكلين.. مثلا محرر للرد السريع وآخر للرد المفصل.. فهل علينا بناء محررين؟!

للإجابة عن هذه الأسئلة نقول: لا!

يمكننا بناء تابع يقوم برسم المحرر في أي مكان نريده.. ويقوم أيضا بتحديد نمط المحرر (نمط مبسط أم موسع) مع تحديد طبيعة الأزرار في كل من النمطين

كل ما علينا هو رصف الأزرار في مصفوفة، ورسم هذه المصفوفة عبر تابع معين.

أغلب المحررات الآن تستخدم مفهوم الـ framework والذي يمكننا من كتابة توابع الجافاسكريبت كـ classes تماما كما هو الحال في لغات البرمجة المتطورة.. لكننا لن ندخل في هذه المعمعة لأن هدفنا تعليمي بالأساس.

لو راقبنا سطر إضافة زر في المحرر سنجد أنه يتألف من ثلاث متحولات

النص alt، مسار الصورة، والتابع الذي يستدعى عند الحدث onClick.

فلو أننا جمعنا هذه البيانات في مصفوفة لأصبحنا قادرين على بناء المحرر عبر تعليمة document.write

وحلقة for.

دعونا نجرب هذا.

المصفوفات في جافاسكريبت:

يتم الإعلان عن المصفوفات في جافاسكريبت بشكل عام عبر السطر البرمجي التالي

```
var Name_of_Array= new Array(number_of_elements);
```

يمكن تعبئة المصفوفات بالبيانات يدويا..فمثلا

```
var names= new Array(5)
names [0]="Ahmad.";
names [1]="Ali.";
names [2]="Khalid.";
names [3]="Sami ";
names [4]="Mustafa.";
```

وكما ترون فالترقيم يبدأ من الرقم صفر...وبنفس الشكل يمكن قراءتها

```
var MyName = names [3];
```

كما يمكن تعبئتها عبر حلقة for

```
var names= new Array(5);
var x=0;
for (x=0; x<5; x++)
{
names[x]=prompt('Enter a name!',' ');
}
```

وبالمثل يمكن القراءة منها.

للمصفوفات في javascript نوعين هامين:

المصفوفات التقليدية والتي بينها في المثال السابق، وهي مصفوفات يشار إلى عناصرها بالأرقام.

والمصفوفات المرتبطة associative array وهي مصفوفات يشار إلى عناصرها بأسماء، بما يشبه ملفات الـxml

مثال:

```
var my_family= new Array()
my_family["father"]="Ahmad";
my_family["mother"]="Mona";
```

```
my_family["son"]="Ali";
my_family["daughter"]="Salwa";
```

ويتم استدعاء العنصر باسمه. هذه المصفوفات أسهل في حالة الرغبة بقراءتها كعناصر فردية، كحالة الشرط مثلا ، لكن لا يمكن المرور عبرها باستخدام حلقات **for**.

```
document.write('My Father is : ' + my_family["father"]+'
and My Mother is:' + my_family["mother"] +' we are a happy
family');
```

يمكن عمليا جعل مصفوفة ما عنصرا في مصفوفة أخرى. مثال:

```
var my_family= new Array()
my_family["father"]="Ahmad";
my_family["mother"]="Mona";
my_family["son"]="[Ali , Kahlid, Ihsan]";
my_family["daughter"]="Salwa";
```

ويمكن تعريف عناصر المصفوفة مباشرة بين القوسين. مثلا

```
var my_family= new Array('Ahmad', 'Ali' , 'Khalid')
var my_family= new Array('Ahmad', 'Ali' , [Khalid, Yaseen,
Salwa])
```

حيث تتضمن المصفوفة الأخيرة عنصرا هو مصفوفة بحد ذاته.

طالما أننا نحتاج لمصفوفة أزرار، وكل زر يحتاج لتعريفه مصفوفة تتألف من ثلاث عناصر، إذا فمصفوفة الأزرار ستكون بالشكل التالي (أنته أن جافاسكريبت حساسة لحالة الأحرف)

```
var toolbar = new Array(
[alttext1, imgURL1, onClick_Function1],
[alttext1, imgURL1, onClick_Function1],
..... )
```

فعناصر مصفوفة الأزرار هي مصفوفات أيضا تعرف **alt text** و مسار الصورة والتابع **onClick**

مثال:

```
var tool_bar1=new Array(
['startline',,,] //don't comment this line
,['Add Image','img/Image.gif','doImage()']
```

```
,['Horizontal Rule','img/rule.gif','doRule()']
,['Add Flash','img/Flash.gif','doVideo("Flash")']
,['Add Google Video','img/Google.gif','doVideo("Google")']
,['Add YouTube Video','img/YouTube.gif','doVideo("YouTube")']
,['Add RealPlaye (ram,ra, rm, rpm)','img/real.gif','doMedia("Real")']
,['Insert Media Player(wav,mp3, wma, wmv)','img/Media.gif','doMedia("Media")']
,['startline',,,]
,['Make A Text Glowing','img/glow.gif','doGlow()']
,['Drop Shadow For Text','img/Shadow.gif','doShadow()']
,['Blur Text','img/Blur.gif','doBlur()']
,['Insert Symbol','img/Symbol.gif','doSymbol()']
,['Insert Word Art','img/wordart.gif','doWordart()']
,['Insert Smilies','img/smile.gif','doSmilies()']
,['Insert Table','img/Table.gif','doTable()']
,['startline','img/.gif','do()']
,['Add Code','img/Code.gif','doCode()']
,['startline',,,]
,['Add Google Search','img/GoogleIlt.gif','doGoogleIlt()']
,['Add Wikipedia Search','img/wiki.gif','doWiki()']
)
```

العنصر ['startline',,,] يدرج صورة الفاصل بين الأيقونات ويقسمها لمجموعات..

لن نفصل في بناء هذه المصفوفات حيث أنكم ستجدونها في المرفقات.. في مشاركة لاحقة

مصفوفات القوائم المنسدلة:

قوائم نوع الخط وقياسه ونوع الترويسة مهمة أيضا، وهي يجب أن تكون قابلة للتغيير من الكود وليس من التصميم. وهي تشبه ما سبق..

```
var lstFont = new Array ( // font face Array
```

```
'Arial' //don't comment this line
```

```
,'Courier'
```

```
,'Sans Serif'
```

```
,'Tahoma'
```

```
,'Traditional Arabic'
```

```
,'Verdana'
```

```
,'Wingdings'
```

```
)
```

```
var lstSize = new Array ( // font size Array
```

```
['1','Very Small'] //don't comment this line
```

```
,['2','Small']
```

```
,['3','Medium']
```

```
,['4','Normal']
```

```
,['5','Large']
```

```
,['6','Larger']
```

```
,['7','Very Large']
```

```
)
```

قد يخطر سؤال للبعض..ما فائدة جعل كل شيء ضمن مصفوفات؟

الجواب..لو أنك وضعت علامة التعليق // قبل أي عنصر من المصفوفة (شرط أن يكون كل عنصر في سطر مستقل كما سبق) فإن هذا العنصر سيختفي . يمكنك وضع الأزرار وإزالتها أو إضافة خطوط وقياسات جديدة للخط بمجرد إضافة سطر.فمثلا إضافة

```
,'DecoType Naskh'
```

كسطر جديد في المصفوفة الخاصة بالخطوط سيضيف هذا الخط إلى قائمة الخطوط دون الاضطرار لتعديل كود الـ HTML وبالتالي

أصبحنا قادرين على التحكم بالمحرر من خلال جافاسكريبت

بناء تابع رسم المحرر والتابع المساعدة

لنبدأ بإنشاء ملف نسميه **wysiwyg-settings.js** ولنضع فيه مصفوفات الأزرار ،لكننا سنقوم بتكرار مصفوفة كل مجموعة أزرار **"toolbar"** ونمنحها اسمين **fm_Aray** و **bm_Array** حيث سنستخدم المصفوفة الأولى في بناء المحرر في الوضع الكامل **Full Mode** والأخرى للوضع المبسط **Basic Mode**

وبالطبع لن تحتوي المصفوفات من نوع **bm_** على جميع الأزرار، ولكنني لن أحذفها من المصفوفة بل سأسبقها بعلامة التعليق **//** فتصبح وكأنها غير موجودة.

سنضع بعض الثوابت في بداية الملف لتعريف المحرر

المتحول **wysiwyg_textarea** يحتوي اسم الـ **textarea** التي نريد تحويلها لمحرر

المتحول **wysiwyg_Mode** يبين فيما إذا كان الوضع المستخدم هو الوضع الكامل أو المبسط

المتحول **wysiwyg_Width** يعبر عن عرض المحرر بالبكسل

Sepline و **startline** تعبر عن أيقونتي بدء شريط الأدوات في كل سطر و الفاصل بين الأدوات ذات الطبيعة المختلفة

HTMLcode يحتوي تعليمة إضافة الزر الذي يحول بين نمط **wysiwyg** ونمط كود الـ **HTML**.

TblStart تعليمات ترويسة الجدول و **RowStart** تعليمات بدء سطر جديد و **TblClose** تعليمات إنهاء الجدول

```
var wysiwyg_textarea = ""; // The name of the text area
var wysiwyg_Mode = 0; // 0=full || 1=basic
var view_HTML = 0; // 0 = present HTML source code button , 1 = don't present it
var wysiwyg_Width= "610px" //pxls
var sepline = "<img alt='Separator' class='butClass' src='img/toolbar.separator.gif' >"
var startline = "<img alt='Start' class='butClass' src='img/toolbar.Start.gif' >"
var HTMLcode = "<img alt='HTML Code' id='HTMLbtn' class='butClass'
src='img/mode.gif' onMouseOver='selOn(this)' onMouseOut='selOff(this)'
onMouseDown='selDown(this)' onMouseUp='selUp(this)' onClick='doHTMLView()'>"
var RowStart = "</td><td></td></tr><tr><td class='tdClass' nowrap >"
var TblStart="";
var TblClose = "";
```

تابع إنشاء القائمة المنسدلة للخطوط

```
function listFont(){  
  
var fonttxt = "<select name='selFont'  
onChange='doFont(this.options[this.selectedIndex].value)'> <option value=''>-- Font --  
</option>";  
  
    for (x=0; x<lstFont.length ; x++)  
  
        { fonttxt += "<option value="+lstFont[x]+">"+lstFont[x]+"</option>"; }  
  
fonttxt += "</select>";  
  
return fonttxt ; }
```

أعتقد أن حلقة **for** واضحة المعالم هنا ، تابع إنشاء قائمة الحجم والترويسات لا يختلف كثيرا عن هذا.

تابع إنشاء شريط الأدوات

يقوم هذا التابع بقراءة المصفوفة الخاصة بكل شريط أدوات (والتي نمررها له كمتحول) وتحويلها إلى كود HTML مع تحويل الرموز الخاصة **sepline** و **selFont** و الخ.. إلى مقابلاتها من نصوص HTML

```
function doToolbar(tbArray) {  
  
var toolbar ="";  
  
var x=0;  
  
for (x=0; x<tbArray.length ; x++) {  
  
    if (tbArray[x][0] == "sepline")  
  
        { toolbar += sepline ;    }  
  
    else if (tbArray[x][0] == "startline")  
  
        { toolbar += startline ; }  
  
    else if (tbArray[x][0] == "selFont")  
  
        { toolbar += listFont(); }
```

```

else if (tbArray[x][0] == "selSize")
    { toolbar += listSize();    }

else if (tbArray[x][0] == "selHeading")
    { toolbar += listHead(); }

else

    { toolbar += "<img alt='"+ tbArray[x][0] + "' src='"+ tbArray[x][1] + "'
onClick='"+ tbArray[x][2] + "' class='butClass' onMouseOver='selOn(this)'
onMouseOut='selOff(this)' onMouseDown='selDown(this)'
onMouseUp='selUp(this)'>";}

}

return toolbar ;

}

```

تابع إنشاء المحرر.

نقوم في هذا التابع بترتيب السلاسل النصية التي أنشأناها من خلال التتابع السابقة بشكل جدول ، حيث نمرر للتابع اسم منطقة النص **textarea** التي سيقوم المحرر بتنسيق النص فيها، يقوم بإخفاءها (بوضع الخاصية **style.display** إلى **none**) ومن ثم يضيف السلاسل تباعا

الشرط **if (wysiwyg_Mode==1)** يحدد فيما إذا كنا نريد بناء محرر مبسط أم كامل، ومن السهل معرفة أن ضبط هذا المتحول عبر واجهة **HTML** سيحدد لنا نوع المحرر دون أن نتدخل في بنية **HTML** (سنرى تطبيق ذلك لاحقا)

```
function initiate(VbTextarea, vWidth, vHTML, vMode){
```

```
function initiate(VbTextarea, vWidth, vHTML, vMode){
```

```
if (VbTextarea != null) {
```

```
wysiwyg_textarea = VbTextarea;
```

```

} else {

alert (" Text area id must be assigned" ) ;

return;

}

if (vMode!=null) wysiwyg_Mode = vMode;

if (vHTML!=null) view_HTML = vHTML;

if (vWidth!=null) wysiwyg_Width =vWidthTblStart ="<table id='tblCtrls' width='"+
wysiwyg_Width +" height='30px' border='0' cellspacing='0' cellpadding='0'
bgcolor='#DBE6F3'><tr><td class='tdClass'>"

TblClose ="</td></tr></table><iframe id='wysiwyg' name='iframe' style='width: " +
wysiwyg_Width +"; height:200px' ></iframe>"

document.getElementById(wysiwyg_textarea).style.display = "none";

var wysiwyg_code= TblStart ;

if (wysiwyg_Mode==1 ) //basic mode

{
    wysiwyg_code+= doToolbar(bm_Edit);

    wysiwyg_code+= doToolbar(bm_Add);

    if (view_HTML == 0 )

    { wysiwyg_code+= HTMLcode ; }

    wysiwyg_code+= RowStart ;

    wysiwyg_code+= doToolbar(bm_Font);

    wysiwyg_code+= TblClose

} else {    // Full Mode

    wysiwyg_code+= doToolbar(fm_Edit);

    wysiwyg_code+= RowStart ;

```

}

لاحظ ان التابع يسمح لك بتخطي جميع التعريفات عدا اسم الـ `textarea`

جرب ان تستبدل التابع

```
initiate("mytextarea","550px",0,0);
```

بالتابع

```
initiate("mytextarea");
```

لاحظ أن المحرر يظهر بصورة طبيعية

إنشاء المحرر في صفحة HTML

في النموذج الذي تريد أن تجعل `teaxarea` فيه محرراً، وعلى فرض أن اسمها هو `mytextarea`

```
<form name="myform" id="myform" action="myform.aspx">
```

```
<textarea name="mytextarea" cols="20" rows="2"></textarea>
```


</form>

سنكتب التالي تحت هذا العنصر مباشرة:

```
<script type="text/javascript" >
```

```
// Definition: initiate("mytextarea area ID", editor width (px or %) ,HTML_view , Editor Mode);
```

```
initiate("mytextarea","550px",0,0); </script>
```

وكما هو واضح فإننا نتحكم بطبيعة المحرر من هذا السكريبت بتمرير المتغيرات التي نحتاجها للتابع `initiate`

قم بتغيير خانة Editor Mode إلى 1 ولاحظ الفرق

التعديل الهام الذي يجب أن نضيفه إلى كل التوابع التي أنشأناها والمتعلقة بتغيير كود الـ HTML هو أن نسند قيمة الـ

InnerHTML للمحرر إلى `mytextarea`.

استبدل الصيغة

```
if (ISHTML==2)
```

```
doHTML();
```

بالصيغة التالية أينما وجدت

```
if (ISHTML==2) {doHTML();}
```

```
document.getElementById(wysiwyg_textarea).innerText =
```

```
wysiwyg.document.body.innerHTML;
```

على كل حال قمت بتصحيح بعض الأشياء في التوابع ولا مجال لذكرها هنا، كما أضفت تابعا للصفحة بحيث تظهر القيمة التي سترسل لقاعدة البيانات. تجدون في المرفقات كامل المحرر.

في المرحلة القادمة سننشئ صفحة الـ **asp.net** التي تقوم بمهمة تفسير و تخزين البيانات في قاعدة بيانات **access**

المرحلة الثالثة

صفحة المفسر **asp.net**

بعد أن كنت قد قررت إنشاء صفحة تخزين البيانات في قاعدة بيانات وجدت أن هذا العمل سيكون خارج نطاق مهمتنا في إنشاء المحرر، لذلك سأكتفي بإنشاء المفسر ووضع القيمة النهائية التي ستخزن في قاعدة البيانات بشكل متحول نصي.

على كل حال كنت ومنذ فترة أقوم بتعريب المنتدى DMG وتطويره بحيث تكون عمليا أكثر ، وقد قمت بتطوير المحرر الذي ندرسه سوياً لأجل ذلك المنتدى بالتحديد، ولعلني أبداً بمناقشة آلية تطوير هذا المنتدى في منتدى مطوري ASP.NET لاحقاً ، وسيرى من يتابع الموضوع كيفية استخدام المحرر عملياً.

يقوم المحرر الذي أنشأته بتحويل نصوص HTML إلى كود UBBcode الشهير الذي يستخدم في المنتديات، ولكن وكما هو واضح أصبح هذا الموضوع ضحماً جداً ولم أجد التفاعل الكاف من الأعضاء، لذلك لن أشرح كيفية تحويل المحرر ليصبح قادراً على إصدار محتوياته مباشرة مرمزاً بكود UBBCode دفعا للملل، ولعلني أفرد لهذا موضوعاً آخر لاحقاً.

مهمة المفسر أن يقوم باستبدال الأكواد التالية بمقابلاتها.

[Flash] URL[/Flash]

[Flash width=X height=Y] URL [/Flash]

[Google width=X height=Y] URL [/Google]

[Google] URL [/Google]

[YouTube] URL [/YouTube]

[YouTube width=X height=Y] URL [/YouTube]

[Real] URL [/Real]

[Real width=X height=Y] URL [/Real]

[Media] URL [/Media]

[Media width=X height=Y] URL [/Media]

فمثلاً يجب أن يستبدل الكود

[Flash width=X height=Y] URL [/Flash]

بالمقابل:

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" width="X" height="Y" ><param name=""movie"" value="URL"><embed src="URL" width="X" height="Y"></embed></object>
```

الدالة التي تقوم بالاستبدال هي **Replace** ويمكننا وبما أننا نريد استبدال جزء محدد من النص بجزء آخر محدد، فلا يمكننا القيام بالاستبدال بناءً على موقع الحارف (يعني لا يمكننا استبدال الحرف 14 في العبارة الأولى بالحرف 69 في العبارة الثانية لأن الرقم X متغير قد يتألف من خانتين أو ثلاث خانات)

لحل هذه المعضلة سنستخدم ما يسمى **regular expressions** والتي تحدد لنا نمطا ما مكررا في السلسلة الحرفية وتسمح لنا باستبدال نمط معين بنمط آخر.

مثال للتوضيح:

في النص التالي

```
[Flash width=X height=Y] URL [/Flash]
```

هناك ثمان أنماط مكررة. سأحصر كل نمط بين قوسين لتوضيح المعنى

```
([Flash])(width=)(X)( height=)(Y)(])(URL)([/Flash])
```

وفي العبارة الثانية هناك أيضا 14 نمطا مكررا مهما كان نوع ملف الفلاش المدرج (أي أي وسم الـ HTML سيحتوي هذه الأجزاء أو الأنماط حصرا)

```
(<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000")( width=)("X")( height=)("Y") (><param name=""movie"" value=)("URL")(><embed src=)("URL")( width=)("X") (height=)("Y")(></embed></object>)
```

عبارات **regular expressions** تسمح لنا باستبدال نمط بآخر، وبالتالي يمكننا استبدال النمط رقم 3 في العبارة الثانية بالنمط رقم 3 في العبارة الأولى..

لنفترض أن العبارة الأولى كانت مخزنة في متحول نصي اسمه **Result_Text**، العبارة التي ستقوم بهذا العمل هي:

```
Result_Text = Regex.Replace(Result_Text, "(Pattern1)(Pattern2)(Pattern3)",  
"Replacement Text ")
```

ولو افترضنا أن العبارة **Replacement Text** هي:

```
"I want to have X books"
```

وأننا نريد استبدال النمط Pattern2 بالرقم X فالعبارة ستصبح:

```
Result_Text = Regex.Replace(Result_Text, "(Pattern 1)(Pattern 2)(Pattern 3)", "I want to have $2 books")
```

في المثال السابق الجزء Flash هو الجزء الأول حصرا بينما الجزء width فليس الجزء الثاني بالضرورة فقد يأتي تحديد الارتفاع height قبل العرض، من ناحية أخرى المسافة بين كلمة Flash و Width ليست محرفا واحدا بالضرورة بل قد تكون محرفين (فراغين) أو أكثر

فمثلا الأكواد التالية متشابهة تماما

```

```

```

```

```
<img SRC="..">
```

```
<img src='..' >
```

```
..etc
```

فما الحل؟

الحل باستعمال عبارة الاستبدال عدة مرات من جهة لتنسيق المتحول النصي من جهة، واستعمال عبارات (Regular RE Expressions) من جهة أخرى

كود استبدال العبارة الأولى بالثانية في مثالنا السابق هي:

```
Result_Text = Regex.Replace(Result_Text, "([Flash\\])(\\.\\n)*?)([\\VFlash\\])", "<object  
classid=""clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"" width=""425""  
height=""350""><param name=""movie"" value=""$2""><embed src=""$2""  
width=""425"" height=""350""></embed></object>")
```

لنحاول شرح الكود (رغم أن الدخول في RE صعب يحتاج لتفصيل طويل) مع إعطاء لمحة عما يحصل

العبارة الأولى أصبحت

```
([Flash\\])(\\.\\n)*?)([\\VFlash\\])
```

ونلاحظ بوضوح أننا عوضنا النمط الثاني ((\\.\\n)*?) وهو الذي يعبر عن رابط ملف الفلاش باستخدام الرمز \$2 في العبارة

<embed src=""\$2"" width=""425"" height=""350""></embed>

ضم الأنماط بين قوسين () تسمى التجميع grouping وهي عملية تساعد المبرمج على تحديد رقم النمط من جهة وعلى تنسيق عمله من جهة أخرى. وبالتالي فلدينا ثلاثة أنماط رئيسية [flash] و URL و [/flash]

يستخدم الحرف \ لتجاوز دلالة الأحرف المحجوزة (مثل * و / و < و > والأقواس [] و { } وغيرها)

فالعبرة \ تخبر التابع أن / ليست عملية تقسيم بل حرف عادي وأن * ليست عملية ضرب وأن ما بعد الفاصلة العلوية (') ليس تعليقا بل تنمة النص..

أمثلة:

"I can't go" is written as "I can\t go"

"Paragraph tag is <p>" is written as "Paragraph tag is \<p>"

وبالتالي:

[flash] is written as \[flash\]

بالنسبة لبقية الرموز فهذه لمحة عنها:

المقارنة

^ :يعبر عن بداية السلسلة المقارنة

\$:يعبر عن نهايتها

. :يرمز لأي محرف

[. - .] :أي حرف من الحرف الأول إلى الثاني مثلا [A-Z] يبحث عن وجود أي حرف كبير Capital.

الرموز الحرفية

\. :يجعل الحرف "." يفقد معناه الخاص كما أسلفنا سابقا*\ تعني البحث عن رمز النجمة وبالتالي فرمز النجمة فقد معناه كدلالة على عملية الضرب

يعبر عن رقم ثماني octal بثلاث خانوات

\x## يعبر عن رقم ست شعري بخانتين

التجمعات

(.*) تقابل كل ما بين قوسين وترمز له بالرمز \$x حيث x هو رقم النمط

مثال: في حال البحث عن التاريخ من نمط 22/12/2008 نكتب

(.*)/(.*)/(.*)

وتعبر \$1 عن الأيام و \$2 عن الأشهر و \$3 عن السنة

بينما في التاريخ من نمط September 12/ 2008 فنكتب

(.*)/(.*)/(.*)

فالمحول \$1 عن الشهر و \$2 عن الأيام و \$3 عن السنة

*|. تعني مقارنة أحد النصين: مثال (<i>|em </i>) تبحث عن وجود أي من الوسمين المخصصين لإمالة النص

i & em

* تعبر عن البحث عن تكرار النمط 0 مرة أو أكثر

+ تعبر عن البحث عن تكرار النمط مرة واحدة على الأقل

? تعبر عن عدم تكرار النمط أو تكراره مرة واحدة

أمثلة:

مثال:

foo(.*)bar في الجملة the food is barbecued in the barn
نجد

\$1 = "d is barbecued in the "

لأن النمط foo(.*)bar تكرر مرتين

في حال استخدام العبارة `foo(.*?)bar` في الجملة `the food is barbecued in the barn`

```
$1 = "d is "
```

على الرغم من أن النمط `foo(.*?)bar` تكرر مرتين لكن ؟ حصرت في اختيار التكرار الأول فقط، وبالمثل في حال استخدام العبارة `foo(.*?)bar` في الجملة `the food is barbecued in the barn` نجد

```
$1 = "d is "
```

`{n}` يبحث عن تكرار النمط `n` مرة ويضع الجميع في متحول واحد (مثلا رقم يتألف من ثلاث خانات)

`{n,}` يبحث عن تكرار النمط على الأقل `n` مرة ويضع الجميع في متحول واحد (مثلا رقم يتألف من ثلاث خانات على الأقل)

`{n,m}` يبحث عن تكرار النمط على الأقل `n` مرة ولكن ليس أكثر من `m` مرة ويضع الجميع في متحول واحد (مثلا رقم يتألف من `n` خانة على الأقل و `m` خانة على الأكثر)

وهناك الرمز الخاصة مثل `\n` وتعبّر عن بداية سطر جديد، و `\r` وتعبّر عن بداية مقطع جديد وهكذا..

هذه لمحة عن RE بشكل عام، وبالتالي فالعبارة

```
(\[Flash\])(.*\n)?(\[\/Flash\])
```

تقول: أبحث عن ثلاثة أنماط تكرر

النمط الأول يحتوي النص `[Flash]` والثاني قد يحتوي أحرفا وقد يحتوي بداية سطر.. المهم أن تحصر جميع الأحرف بين النمطين الأول والثالث وتجعلها في متحول واحد

وفق لما سبق فإن العبارة

```
[Flash] URL [\/Flash]
```

و العبارة

```
[Flash]
```

```
URL
```

```
[\/Flash]
```

[FLASH] URL [/FLASH]

لن تفسر وستمر كما هي..

هناك وسائل محددة تجعل RE غير حساس لحالة الحرف.. لكنها خارج نطاق بحثنا

التابع الذي سيقوم بتفسير الأكواد السابقة

[Flash] URL[/Flash]

[Flash width=X height=Y] URL [/Flash]

[Google width=X height=Y] URL [/Google]

[Google] URL [/Google]

[YouTube] URL [/YouTube]

[YouTube width=X height=Y] URL [/YouTube]

[Real] URL [/Real]

هو

```
Dim Result_Text As String = mytextarea.Text
```

```
Result_Text = Regex.Replace(Result_Text, "([Flash\])([.|\n]*?)([VFlash\])",  
"<object classid=""clsid:D27CDB6E-AE6D-11cf-96B8-444553540000""  
width=""425"" height=""350""><param name=""movie"" value=""$2""><embed  
src=""$2"" width=""425"" height=""350""></embed></object>")
```

```
Result_Text = Regex.Replace(Result_Text, "([Flash])( width\=)([.|\n]*?)(  
height\=)([.|\n]*?)\)([.|\n]*?)([VFlash\])", "<object classid=""clsid:D27CDB6E-  
AE6D-11cf-96B8-444553540000"" width=""$3"" height=""$6"" ><param  
name=""movie"" value=""$9""><embed src=""$9"" width=""$3""  
height=""$6""></embed></object>")
```

```
Result_Text = Regex.Replace(Result_Text, "([Google\])([.|\n]*?)([VGoogle\])",  
"<embed style=""width=""425""; height=""350"" id=""VideoPlayback""  
type=""application/x-shockwave-flash"" src=""$2""></embed>")
```

```
Result_Text = Regex.Replace(Result_Text, "([Google])( width\=)([.|\n]*?)(  
height\=)([.|\n]*?)\)([.|\n]*?)([VGoogle\])", "<embed style=""width=""$3"";
```

```

height=""$6"" id=""VideoPlayback"" type=""application/x-shockwave-flash""
src=""$9""></embed>")
    Result_Text = Regex.Replace(Result_Text,
"(\[YouTube\])(http:\\\\www.youtube.com\\watch?v=)((.\\n)*?)(\\[\\VYouTube\\])",
"<object width=""425"" height=""350""><param name=""movie""
value=""http://www.youtube.com/v/$3""></param><param name=""wmode""
value=""transparent""></param><embed src=""http://www.youtube.com/v/$3""
type=""application/x-shockwave-flash"" wmode=""transparent"" width=""425""
height=""350""></embed></object>",
System.Text.RegularExpressions.RegexOptions.IgnoreCase)
    Result_Text = Regex.Replace(Result_Text, "(\[YouTube\])( width\\=)((.\\n)*?)(
height\\=)((.\\n)*?)(\\])(http:\\\\www.youtube.com\\watch?v=)((.\\n)*?)(\\[\\VYouTube\\])",
"<object width=""$3"" height=""$6""><param name=""movie""
value=""http://www.youtube.com/v/$10""></param><param name=""wmode""
value=""transparent""></param><embed src=""http://www.youtube.com/v/$10""
type=""application/x-shockwave-flash"" wmode=""transparent"" width=""$3""
height=""$6""></embed></object>",
System.Text.RegularExpressions.RegexOptions.IgnoreCase)
    Result_Text = Regex.Replace(Result_Text, "(\[Real\])(.\\n)*?)(\\[\\VReal\\])",
"<embed src=""$2"" autostart=true hidden=false></embed>")
    Result_Text = Regex.Replace(Result_Text, "(\[Real\])( width\\=)((.\\n)*?)(
height\\=)((.\\n)*?)(\\])(.\\n)*?)(\\[\\VReal\\])", "<embed src=""$9"" width=""$3""
height=""$6"" nojava=true autostart=true controls=ImageWindow
console=one></embed><br /><embed width=""$3"" height=""$6""
pluginspage=""http://www.real.com/player"" type=""audio/x-pn-realaudio-plugin""
autostart=""true"" controls=""controlpanel,statusbar"" ></embed>")
    Result_Text = Regex.Replace(Result_Text, "(\[Media\])(.\\n)*?)(\\[\\VMedia\\])",
"<embed src=""$2"" autostart=true hidden=true>")
    Result_Text = Regex.Replace(Result_Text, "(\[Media\])( width\\=)((.\\n)*?)(
height\\=)((.\\n)*?)(\\])(.\\n)*?)(\\[\\VMedia\\])", "<embed type=""application/x-mplayer2""
src=""$9"" name=""mediaplayer"" width=""$3"" height=""$6"" showcontrols=""1""
showstatusbar=""1"" showdisplay=""1"" autostart=""0""></embed>")

```

حاول أن تدرس كل عبارة على حدة، ولمزيد من التفاصيل عن RE الرجاء زيارة الرابط التالي:

<http://www.regular-expressions.info>

تحدون في المرفقات المحرر كاملا مع أضرار الإرسال وغيرها. إما قضية التخزين في قاعدة البيانات فهذه خارج نطاق جافاسكريبت..

شكرا لمرافقتكم لي..