Action Script 2.0

من الصفر

المؤلف

مبرمج

مايكل نبيل اخنوخ

Micheal240001@yahoo.com

micheal_nabil@hotmail.com

إهداء

اهدي هذا الكتاب بباقة ورد عطرة
الى صديقى العزيز/ شريف محمد الشاذلى
هو اول من علمنى حرفا فى برنامج الفلاش و لغة الاكشن اسكربت و له الفضل فى اهتمامى بهذا المجال.
و ايضا الى خطيبتى الغالية التى ساعدتنى فى تنسيقة و اعدادة و مراجعة الاكواد و تطبيقها .

ملحوظة :

المادة العلمية لهذاالكتاب قد تم مراجعتها و لكنة غير مسئول عن الاخطاء التى قد تحدث من سوء التطبيق او السهو و الخطأ مع مراعاة ان الكتاب قد تم تحريرة على برنامج الورد لذلك قد تجد مسافات زائدة فى الكود لذلك فى حالة أكتشاف أى اخطأ برجاء ارسال بريد الكترونى للكاتب توضح بة الخطأ و الصفحة .

جميع الاراء الموجودة فى هذا الكتاب هى اراء تعبر عن راى الكاتب الشخصى حتى لو لم توثق بمراجع .

<u>الملاحق</u> تعتبر خارج نطاق الكتاب و قد وضعت كمرجع للقارئ و ليست من صنع الكاتب لذلك ليس على الكاتب ادنى مسئولية من التطبيق الخاطئ لهذة المعلومات و قد وضعت اسماء المواقع التى تم اقتباس هذة الملاحق منها حفاظا لحقوق اصحابها و ليقوم القارئ بزيارتها .

أرسل ملاحظاتِك وتعليقاتِك وآراءك واقتراحاتِك وأسئلتَك على العنوانِ البريديِّ التالي:

Micheal240001@yahoo.com

ويُفضّل جعل عنوان الرسالة "تعليق على الكتاب"، حتّى يمكنني تمييزها عن باقي الرسائل.

في الانتظار..

الــهـدف

أن الهدف من هذا الكتاب هو توفير العلم بطريقة رخيصة و غير مكلفة و ايضا تبادل المعلومات و الخبرات فى هذا المجال من خلال التعليق على الكتاب و ارسال التعليقات عبر الاميل للكاتب فيتم الرد عليها و نشرها فى الكتاب القادم أما الملاحق التى تم نقلها من موقع الموسوعة العربية للكمبيوتر و الانترنت فذلك لان الموضوعات التى يتناولها الكتاب لها علاقة وثيقة بهذة المقالات و لكنها خارج الهدف من الكتاب .
فالهدف من الكتاب هو تعليم Action script 2.0 من البداية و حتى الاحتراف بأسلوب بسيط و سهل و دون تعقيد و بعيدا عن اسلوب الكود المحفوظ لتنفيذ عمليات معينة لاننى للأسف وجدت نسبة كبيرة من كورسات Action script 2.0 تعتمد على هذة الطريقة و بعد انتهاء الكورس تجد الطالب لا يستطيع انشاء برامج خاصة به.

المحتويات

الـفـهـرس

مـقـدمـةالـكـتـاب

هذا الكتاب موجة لمن لدية خبرة و لو قليلة ببرنامج الفلاش كبرنامج للجرافيك و الرسم و قد قمت فى بداية الكتاب بوضع فكرة سريعة عن برنامج الفلاش بصفة عامة و لكن لن نشرح برنامج الفلاش بالتفصيل بل فقط لغة الاكشن اسكربت ٢ .

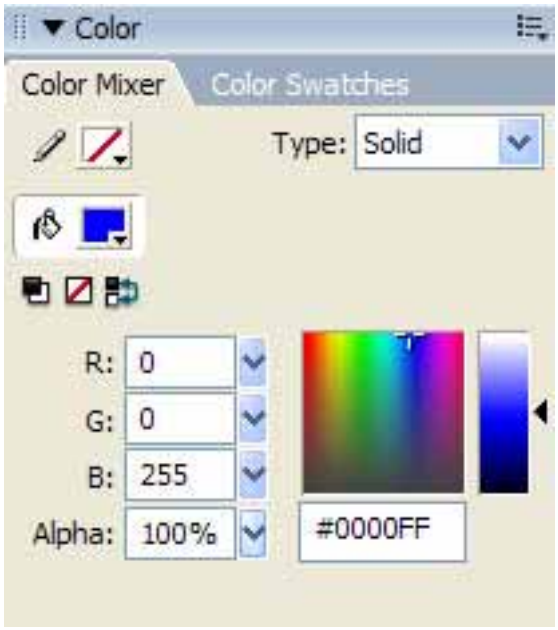
أدوات برنامج فلاش

ستتعرف على معظم الأدوات المستخدمة في برنامج فلاش .

لوحة الأدوات الرئيسية : تحتوي على أهم الأدوات المستخدمة في هذا البرنامج ويحتوي هذا اللوح على الأدوات مثل أداة التحديد، أداة النص، أداة سطل التلوين، أداة رسم المستطيل، أداة رسم الدوائر، أداة الفرشاة، أداة القلم،أداة الممحاة ، أداة التحجيم ، أداة التكبير والتصغير .... ألخ ، سيتم شرح مهمة هذه الأدوات تدريجياً مع الدروس كما أنه يمكنك معرفة مهمة كل أداة من خلال وضع الماوس عليها ، الشكل رقم (١) يبين لوحة الأدوات الرئيسية .

الشكل (١)

لوحة التحكم بالالوان: الوان العمل أو بألوان الأشكال أو الكتابات وهي مقسمة إلى قسمين ، القسم الأول لاختيار لون خط الرسم الثاني لاختيار لون الشكل أو التحكم في إظهار اللون من عدم إظهاره وغيرها ..... الشكل رقم (٢)

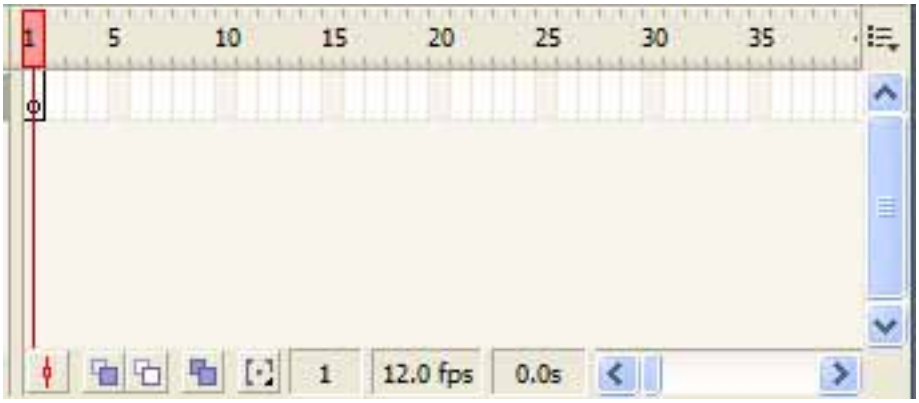


الشكل (٢)

نافذة الطبقات : **وهي مهمة جدا ففيها يتم وضع الكائنات في طبقات محددة يتم التحكم بتغيير أسمها بالضغط نقرتين على نفس الطبقة وترتيبها وكذلك إخفائها أو حمايتها من التعديل أو حذفها ..... الشكل (٣) .**

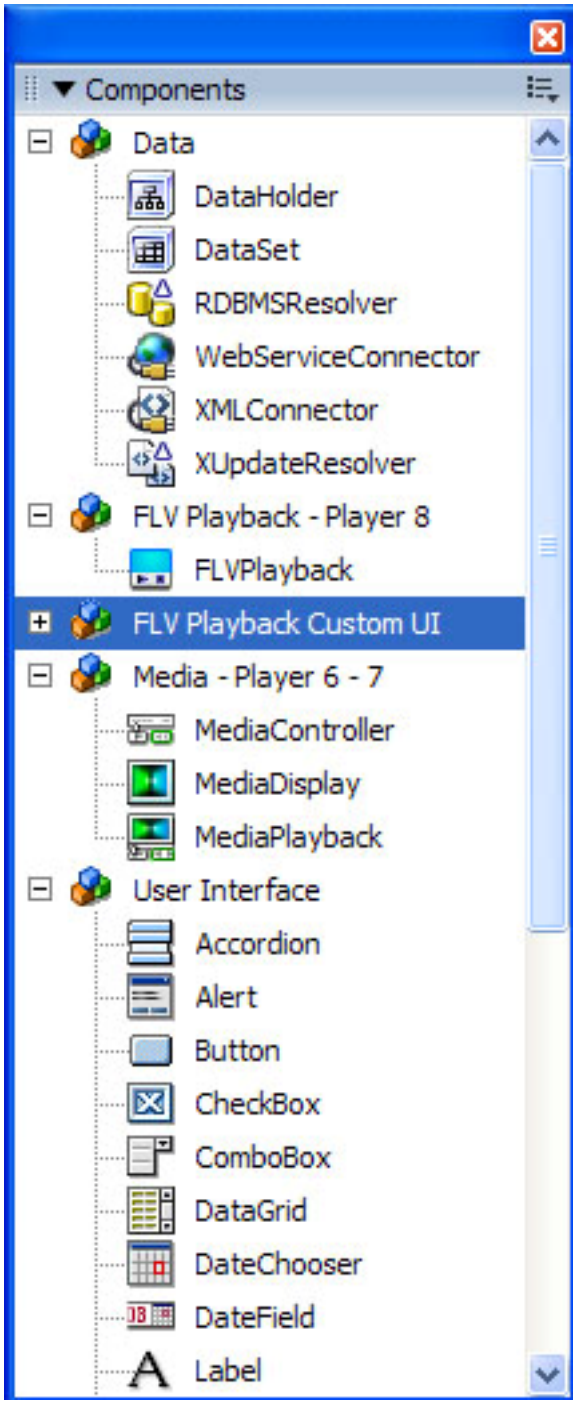
الشكل (٣)

شريط الوقت Time Line : **حيث يقسم الوقت إلى إطارات Frames ، افتراضياً كل ١٢ إطار يمثل ثانية واحدة ، حيث يتم من خلاله تنسيق حركة الإطارات وتنظيم فيلم الفلاش ... الشكل (٤) .**

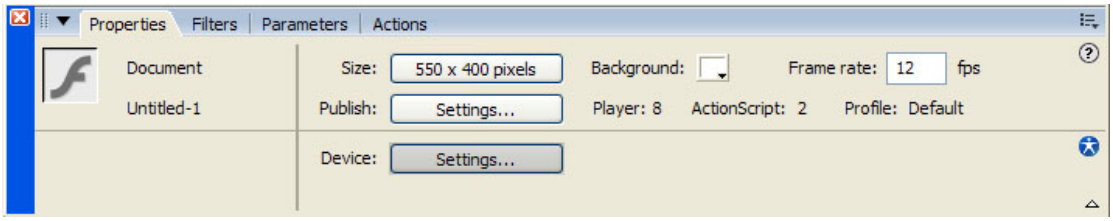


الشكل (٤)

لوح Components : **وهي تحوي العناصر الأكثر استخداماً في نظام ((التربيط التبادلي)) ، الشكل (٥) .**

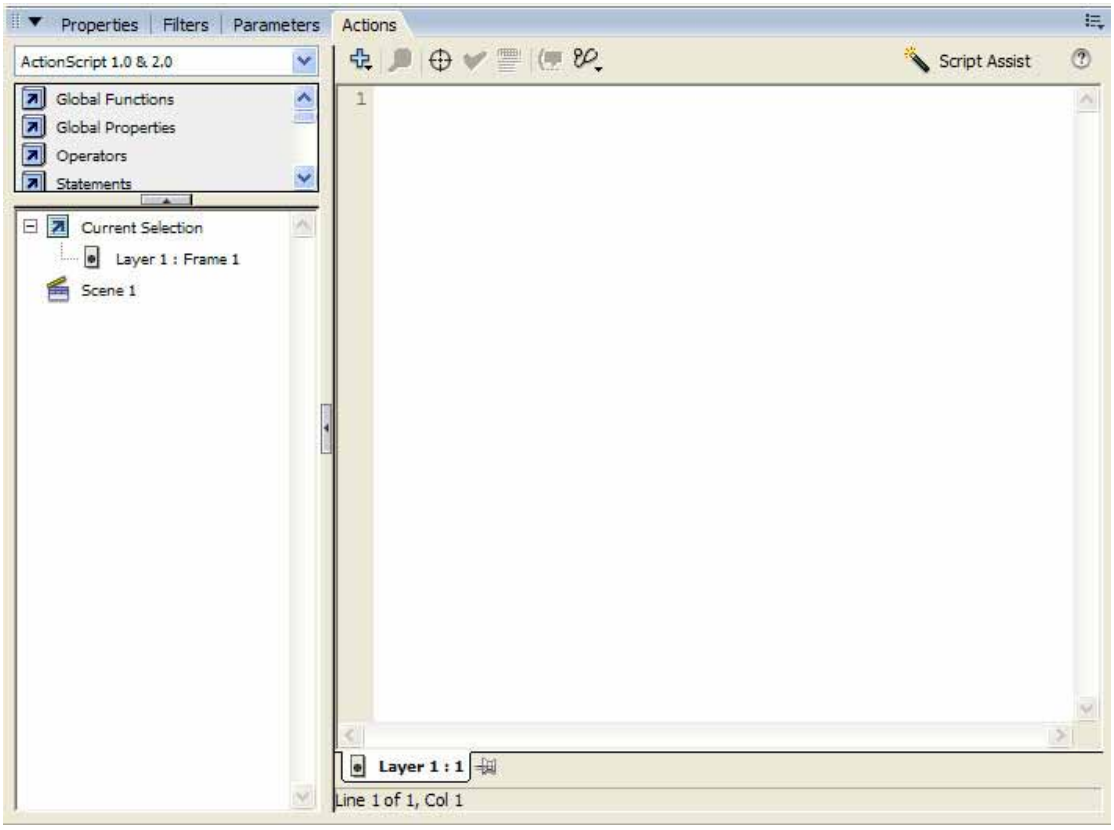


لوحة الخصائص Properties : وهي التي تتحكم بمعظم خصائص ملف فلاش والأشكال الموجودة ضمنه وستتعرف عليها خلال الدروس القادمة ، الشكل (٦) .



الشكل (٦)

لوحة الأحداث Actions : وهي تستخدم من أجل إضافة الأوامر البرمجية لفيلم الفلاش ، الشكل (٧) .



الشكل (٧)

الـكـتـابـة بـالـلـغـة الـعـربـيـة

من أهم مواضع القصور في برنامج فلاش للمستخدم العربي عدم دعمه للغة العربية ، بالتالي لا يمكن الاعتماد عليه وحده لتطبيق أدواته على نصوص باللغة العربية. إلا انه بالإمكان تجاوز هذا القصور عن طريق استخدام برنامج Corel Draw الذي يدعم اللغة العربية لإعداد النص المطلوب ومن ثم تصديره بصيغة ملف Adobe Illustrator إلى برنامج فلاش أو عن طريق برنامج الرسام العربي .

برنامج Corel Draw :-

١ - شغل برنامج الكوريل ثم استخدم أداة النص لكتابة العبارة المطلوبة يمكنك أيضا اختيار حجم الخط ولونه واسمه حسب الرغبة .

٢ - الآن سنستخدم أداة خاصة في Corel Draw لتحويل النص إلى انحناءات ، بعد تظليل العبارة التي كتبتها أختر Convert To Curves<Arrange من القائمة الرئيسية ، ستلاحظ أن تحويل النص إلى انحناءات قد قسم وباعد بين الحروف ، لا تهتم لذلك سنصلح هذا الخلل ضمن فلاش .

٣ - بعد ذلك من القائمة الرئيسية File أختر Export لتصدير النص ومن ثم ستحصل على مربع حواري يطلب منك تسمية الملف المراد تصديره واختيار نوعية التصدير ، أكتب اسما للملف ثم اختر Adobe Illustrator و اختصاره AI من القائمة المنسدلة ثم اضغط Export ثم OK ، هذا كل ما عليك فعله في الكوريل ...

٤ - اذهب الآن إلى برنامج فلاش وقم باستيراد النص الذي أعددناه في الخطوات السابقة من القائمة الرئيسية لبرنامج فلاش اختر Import من قائمة File ثم حدد الملف الذي حفظته .

٥ - لاحظ أن النص يظهر مقطع، من أجل تصليح هذا الأمر من القائمة الرئيسية Modify أختر Break Apart ، اضغط مفتاح Shift في لوحة المفاتيح ثم استخدم الزر الأيسر للماوس وانقر مرة واحدة على أي حرف ثم

استخدم السهم الأيمن في لوحة المفاتيح وقم بإزاحة بقية النص إلى الجهة اليمنى حتى تلتصق الحروف مع هذا الحرف ، كرر نفس الطريقة مع بقية الأجزاء المقسمة الأخرى حتى تحصل على نص متكامل بنوعية جيدة .

** يمكنك الآن تغير لون الخط ومقياسه حسب الرغبة ، فقط اضغط Ctrl+A في لوحة المفاتيح ثم استخدام أداة التلوين لتغير لون الخط وأداة القياس لتحديد المقياس .

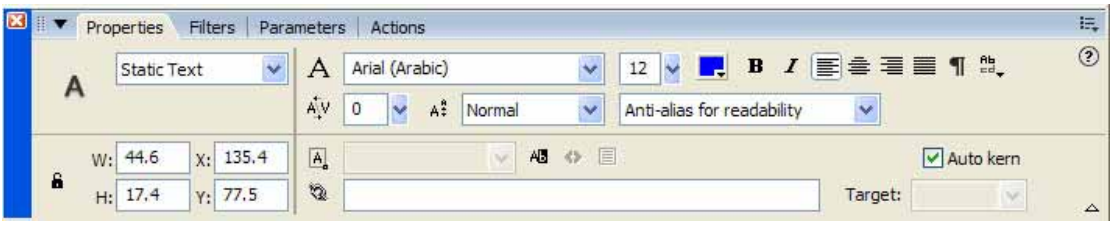
برنامج الرسام العربي :

١ - اكتب النص الذي تريد في واجهة برنامج الرسام العربي الرئيسية .

٢ - من القائمة الرئيسية Edit اختر Copy & Minimize سيحفظ جهاز الكمبيوتر نسخة من النص الذي كتبته في الخطوة الأولى ضمن الذاكرة .

٣ - عد الآن إلى برنامج فلاش استخدم أداة النص وانقر بها حيث المكان الذي ترغب في إضافة النص فيه ، الآن استخدم لوحة المفاتيح واضغط على Ctrl+V لإلصاق النص المحفوظ في الذاكرة ، وطالما إن برنامج فلاش لا يدعم اللغة العربية فستلاحظ ظهور النص على شكل رموز غير مقروءة .

٤ - من لوح الخصائص اختر القائمة المنسدلة التي تحتوي على قائمة بالخطوط (Fonts)الموجودة في جهازك ، ستجد مجموعة من الخطوط تبدأ بالأحرف AXt الإنجليزية . هذه هي الخطوط الخاصة ببرنامج الرسام العربي أختر الخط الذي تريد ولاحظ بأنك حصلت على النص العربي المقروء ، طبق الآن ما تريده من حجم الخط ولونه وذلك من خلال لوح الخصائص الشكل (١) .



الشكل (١)

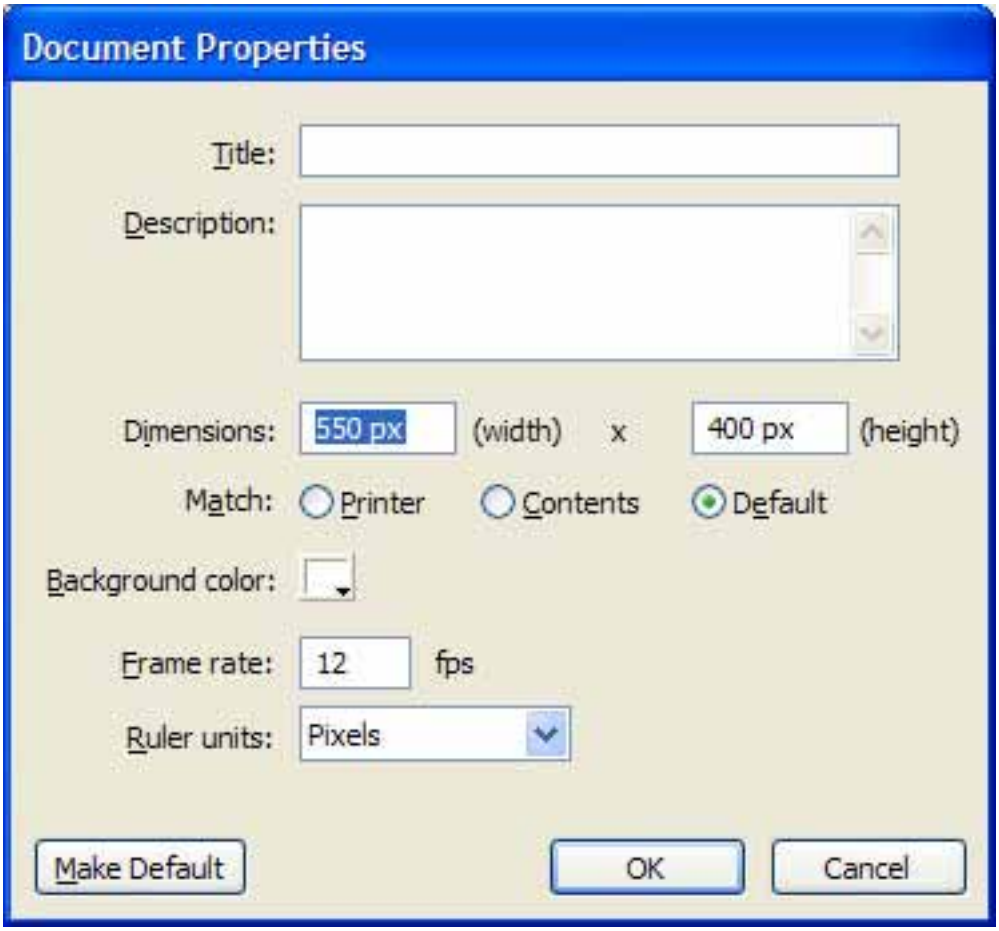
وبذلك تكون قد عرفت كيف تكتب باللغة العربية في برنامج فلاش .

إنشاء الحركة في فلاش ٨

سنتعلم من هذا الدرس كيفية كيفية تحريك الأشكال أو الصور من نقطة في الفيلم إلى نقطة أخرى حيث أنه هناك طريقتين لإنشاء الحركة في فلاش وهما (طريقة الحركة البينية للأطرات، طريقة إطار تلو إطار) وسوف نقوم بشرح كلتا الطريقتين مع الأمثلة المناسبة .

قبل أن تبدأ : عليك أن تتعلم كيف تخصص فيلم الفلاش من حيث أبعاده ولون الخلفية وعدد الإطارات في الثانية وما الى ذلك .

من القائمة الرئيسية Modify اختر Document أو اضغط على Ctrl + M يظهر لك مربع حوار Document Properties ومن ثم تقوم بتخصيص الفيلم كما تشاء كما هو موضح بالشكل (أ) :-



Width: العرض ، height الطول ، Background color لون الخلفية ، frame rate : عدد الاطارات بالثانية ،ruler units وحدة القياس .

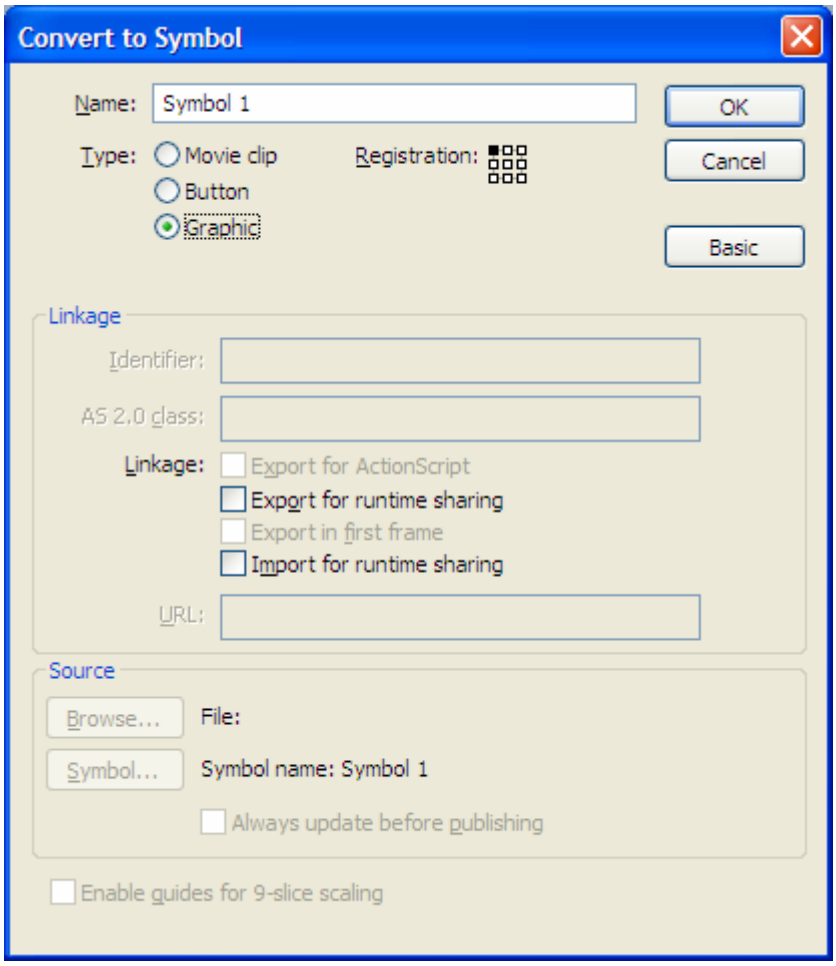
الشكل (أ)

أولا- طريقة الحركة البينية للأطر :

على فرض أننا سوف نقوم بتحريك دائرة من أول المشهد إلى آخره فتكون الخطوات :

١. اختر أداة الدائرة من شريط الأدوات ثم بعد تحديد لون التعبئة والخط الخارجي نقوم برسم الدائرة في الجانب الأيسر من المشهد، لاحظ أنك حصلت على دائرة سوداء في شريط الزمن عند الإطار رقم١ .

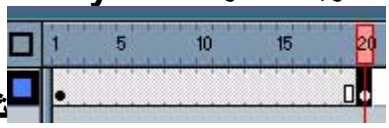
٢. قم بالنقر المزدوج على الدائرة من أجل تحديد لون التعبئة والخط الخارجي ثم نحولها إلى رمز من خلال الأمر Convert to Symbol من قائمة Insert أو من خلال الضغط على زر F8 فيظهر مربع الحوار الشكل(١) نحدد الخيار Graphic ثم نضغط OK .



الشكل (١)

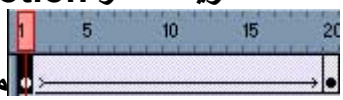
بعد تحويل الشكل إلي رمز يقوم فلاش بحفظ الرسم في المكتبة الخاصة للمشروع يمكن رؤية ذلك من خلال الأمر Library من قائمة Window ((سنقوم بشرح التعامل مع المكتبات في الدروس القادمة)) .

٣. الآن باستخدام الماوس نقوم بتحديد الإطار ٢٠ من شريط الزمن لنفس الطبقة بالزر الأيمن وتختار من القائمة السريعة الأمر Insert Keyframe أو نضغط الزرF6 حيث سيصبح شريط الزمن هكذا



ثم تقوم بتحريك الدائرة في مربع الرسم إلى الجهة المقابلة باستخدام الماوس .

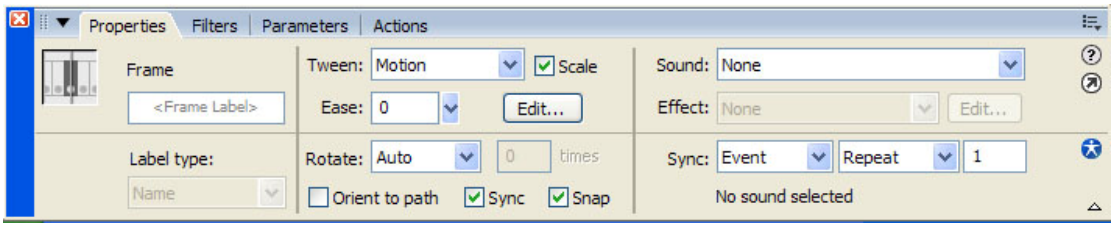
٤. والآن سوف نقوم بإضافة الحركة إلى الفيلم وذلك باستخدام الماوس بالزر الأيمن نحدد الإطار الأول في شريط الزمن ونختار من القائمة السريعة الأمر Tween Create Motion فيصبح شريط الزمن على شكل سهم له خلفية زرقاء باهته



مع العلم أنه لا تعمل الحركة البينية إلا مع الرموز أو الكائنات المجمعة .

٥. الآن نقوم باختبار ما قمنا به من خلال الضغط على زري Ctrl+Enter يجب أن تكون الحركة كما في الشكل (٢) :

الشكل (٢)



الشكل (٣)

يبين الاختيار Tween خيارات الحركة البينية حيث يمكن تحديد معدل الدوران حسب اتجاه عقارب الساعة أو بالعكس أيضا يمكن إلغاء الحركة البينية ويمكن أيضا تحديد خيارات السهول لسرعة الانطلاق أو سرعة الوصول .

ثانياً- طريقة الإطار تلو الإطار:

تعتمد هذه الحركة على تغير محتويات صفحة العرض في كل إطار وهي مناسبة للأشكال المعقدة (مع العلم أن طريقة الإطار تلو الإطار تحتاج لحجم ملف أكبر من الحركة البينية) .

على فرض أننا سنقوم بإنشاء ملعب لكرة السلة وفكرة المثال هي انطلاق الكرة ودخولها في الدائرة كما ترون إن هذا الشكل معقد بالنسبة للحركة الأولى ولقد استخدمنا طريقة الإطار تلو إطار بعد رسم الملعب طبعاً حيث عند كل إطار نقوم بإضافة إطار مفتاحي ثم نحرك الكرة قليلاً وهكذا حتى دخولها الدائرة وسقوطها على الأرض (ولا ننسى تحويل الكرة إلى رمز) .

التغير البيني للأشكال (تأثير المورفنج)

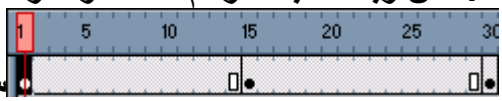
سنقوم بهذا الدرس بتعلم كيفية تغير الأشكال إلى أشكال أخرى تختلف في بنيتها عن طريق التداخل في بنيتها وسوف نقوم بشرح طريقتين لهذا التغير الأولى بالطريقة العادية والثانية باستخدام نقاط التحويل وهي مفيدة جداً من أجل ضبط التغير والتحكم به وسوف نشاهد ذلك من خلال المثالين التاليين عن كل طريقة بحيث يشرح كل مثال كيفية التحول بين الأعداد ١و٢و٣:

أولاً- الطريقة العادية:

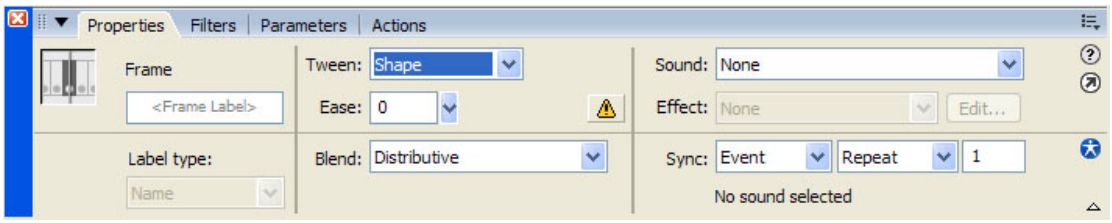
١. نختار أداة النص A ونقوم بكتابة الرقم١ في منطقة الرسم ثم نقوم بتكبيره ليشغل ثلثي منطقة الرسم من خلال الأداة عن طريق سحب مقابض التحجيم مع استمرار الضغط على مفتاح Shift من أن يكون التحجيم متوازي والمحافظة على الشكل، ثم نقوم بكسر العناصر من خلال الأمر Break Apart من قائمة Modify.

٢. عند الإطار١٥ ننقر عليه بالماوس ثم نضغط F6 لإضافة إطار مفتاحي ثم نضغط الزر Delete لحذف الرقم١ ثم نقوم بكتابة الرقم٢ ونجري عليه نفس الخطوة السابقة وكذلك الرقم٣ عند الإطار٣٠.

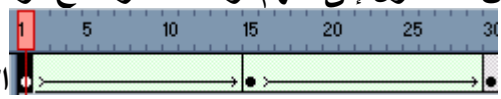
٣. الآن وبعد كتابة الأرقام الثلاثة وكسر عناصرها سوف يصبح شريط الزمن عندك هكذا



سوف نقوم الآن بإجراء التغير البيني للأرقام الثلاثة، اختر الإطار رقم واحد من شريط الزمن ومن لوح Properties نختار الحركة من نوع Shape من القائمة Tween كما في الشكل رقم (١).



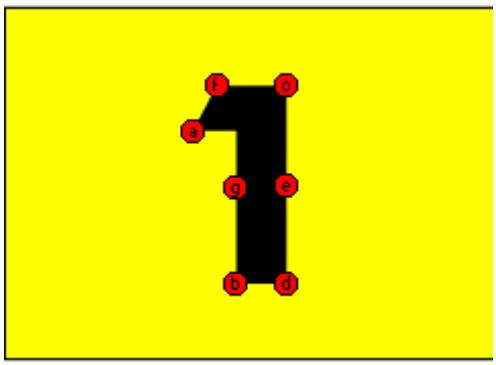
الشكل رقم (١)

سوف تلاحظ أن شريط الزمن قد تحول إلى سهم لونه أخضر فاتح، ونكرر نفس الخطوة عند الإطار رقم١٥ فيصبح شريط الزمن عندك هكذا  **الآن نقوم باختبار الفيلم بالضغط على الزر Ctrl+Enter**

لاحظنا في المثال السابق كيف أن هناك تشويه وعدم تركيز أثناء التغير البيني للأرقام الثلاثة وبالتالي هنا يأتي دور نقاط التحويل للتحكم بتلك التغيرات.

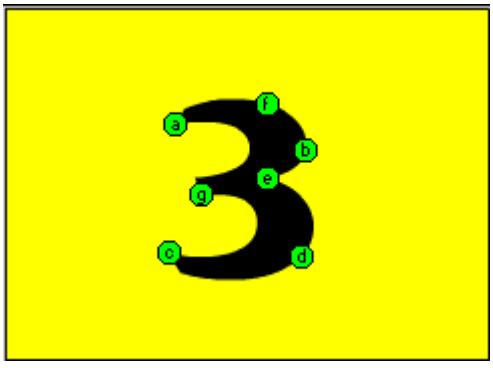
<u>ثانياً- طريقة نقاط التحول:</u>

بالعودة إلى المثال السابق عند آخر نقطة وصلنا لها نقوم بإضافة سبعة نقاط تحول عند الإطار١ من خلال الأمر من القائمة Modify القائمة الفرعية Shape الخيار Hint Add Shape نفس الخيار لكل نقطة أو من خلال الضغط على Ctrl+Shift+H، فيظهر عندنا سبعة نقاط لونها أحمر فوق بعضها البعض نقوم بواسطة الماوس بسحبها إلى زوايا مختلفة كما في الشكل رقم (٣):



الشكل (٣)

الآن نذهب إلى الإطار رقم١٥ ونقوم بسحب النقاط إلى زوايا مختلفة وعند إلغاء التحديد نلاحظ أنه قد تحول لون النقاط إلى الأخضر والنقاط ضمن الإطار رقم١ إلى اللون الأصفر ((دائماً يكون لون النقاط صفراء في إطار البداية وخضراء في إطار النهاية)) ولإظهار نقاط التحويل في حال عدم ظهورها نفعّل الخيار Hints Show Shape من القائمة View، ونكرر نفس الخطوة السابقة عند الإطار١٥ والإطار٣٠ كما في الشكل رقم (٤):



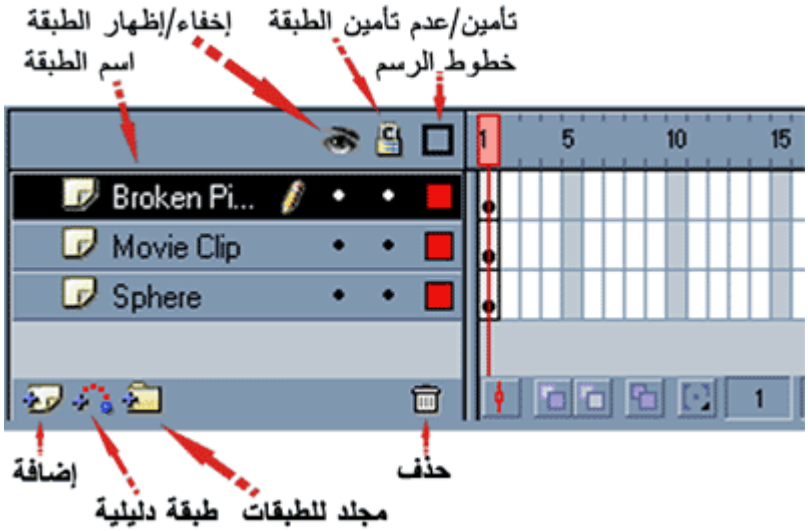
الشكل رقم (٤)

الآن نقوم بتنفيذ العمل بالضغط على الزر Ctrl+Enter.

ꕥ ꕥ ꕥ ꕥ ꕥ ꕥ ꕥ

الطبقات في فلاش ٨

إن استخدام الطبقات مهم جداً في فلاش لتنظيم العمل الفني ، وعندما تقوم بتحرير العناصر على الطبقة ، فإن عمليات التحرير التي تجريها لا تؤثر على العناصر الموجودة على الطبقات الأخرى ، كل طبقة تنشئها لها شريط الوقت الخاص بها كما في شكل رقم (١):



الشكل رقم (١)

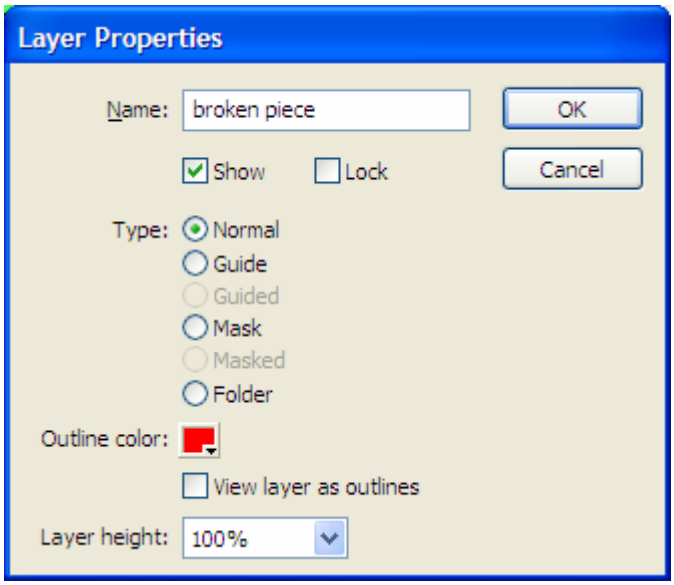
أنواع الطبقات:

هناك أربعة أنواع للطبقات وهي: (الطبقة العادية ـ الطبقة القناع ـ الطبقة الدليلية ـ طبقة المجلد) حيث سنقوم بشرحها بالتفصيل مع الأمثلة إن لزم الأمر:

أولاً- الطبقة العادية:

تفيد تلك الطبقة فقط في فصل العناصر عن بعضها فعند رسم عنصرين متداخلين على نفس الطبقة يصبحان شكل واحد لذلك يفضل عند رسم أي عنصر أن يتم رسمه ضمن طبقة منفصلة، ويمكن تغيير اسم أي طبقة عن طريق النقر

المزدوج على الاسم، ويمكن أيضا تعديل خصائص تلك الطبقة أو تحويلها لطبقة دليلية أو قناع من خلال النقر بالزر الأيمن على الطبقة واختيار الأمر **Properties** الشكل رقم (٢):



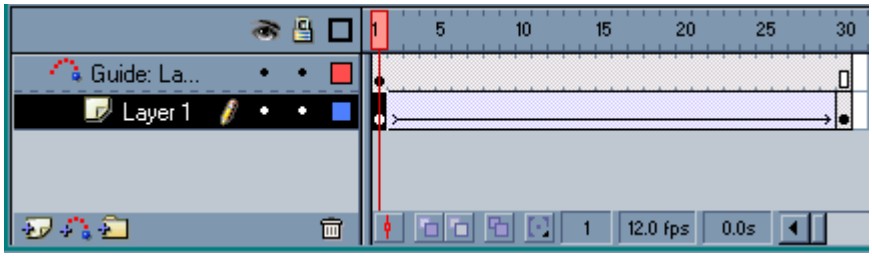
الشكل رقم (٢)

ثانياً- الطبقة الدليلية:

نقوم بإنشاء الطبقات الدليلية لاستخدامها كمرجع عند محاذاة وضبط مواضع العناصر ضمن الأفلام وطبقات دليل الحركة لإنشاء المسار الذي تتبعه العناصر أثناء حركتها ((مع العلم أن كل ما يرسم أو يكتب ضمن الطبقة الدليلية لا يظهر عند عرض الفيلم))، وكمثال على ذلك نفترض أننا سنقوم برسم كرة وتحريكها بمسارات عشوائية فتكون الخطوات:-

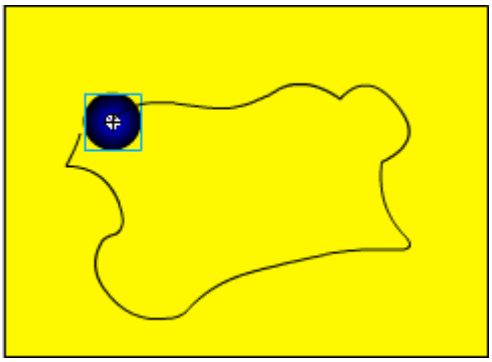
١. نقوم برسم الكرة ونحددها بالماوس بالنقر المزدوج لتحديد لون الملء والإطار معاً ثم نحولها إلى رمز من خلال الضغط على الزر F8 واختيار Graphic.

٢. ندرج إطار مفتاحي عند الإطار ٣٠ نحدد طبقة العمل ثم نضغط على زر إضافة طبقة دليلية يجب أن يكون شريط الزمن والطبقات مثل الشكل رقم (٣):



الشكل رقم (٣)

٣. نحدد الإطار رقم١ من الطبقة الدليلية وباستخدام قلم الرصاص نقوم برسم خط لا على التعين حيث سيكون ذلك الخط هو المسار الذي ستسير عليه الكرة أثناء العرض وليكن الخط كما في الشكل رقم (٤):



الشكل رقم (٤)

٤. نقوم الآن بتفعيل ميزة القفز إلى الدلائل اختر أمر دلائل الرسم guides من قائمة أمر العرض view ، ثم اختر أمر القفز إلى الدلائل snap to guides، ننتقل الآن إلى الطبقة الرئيسية ونحدد الإطار رقم١ ثم نقوم بسحب الكرة إلى رأس الخيط وعند الإطار رقم٣٠ نسحب الكرة إلى رأس الخيط المقابل بحيث يجب في الحالتين أن تظهر قرب الماوس دائرة سوداء تدل على ارتباط الدائرة بالخيط، الآن نضيف الحركة من خلال النقر بزر الماوس الأيمن واختيار الأمر Create Motion Tween، ثم ننفيذ الفيلم بالضغط على Ctrl+Enter .

ثالثاً- طبقة القناع:

هذا النوع من الطبقات مفيد جداً لعمل الخدع والأعمال الفنية حيث يتم وضع طبقة القناع فوق طبقة أو عدة طبقات أخرى لإخفاء أو إظهار أجزاء معينة من تلك الطبقة، فالمناطق المملوءة من طبقة القناع هي فقط التي ستظهر من الطبقة المقنعة، وتكون دائماً طبقة القناع فوق الطبقة المقنعة، وسنقوم بشرح مثال بسيط عن ذلك خطواته هي:

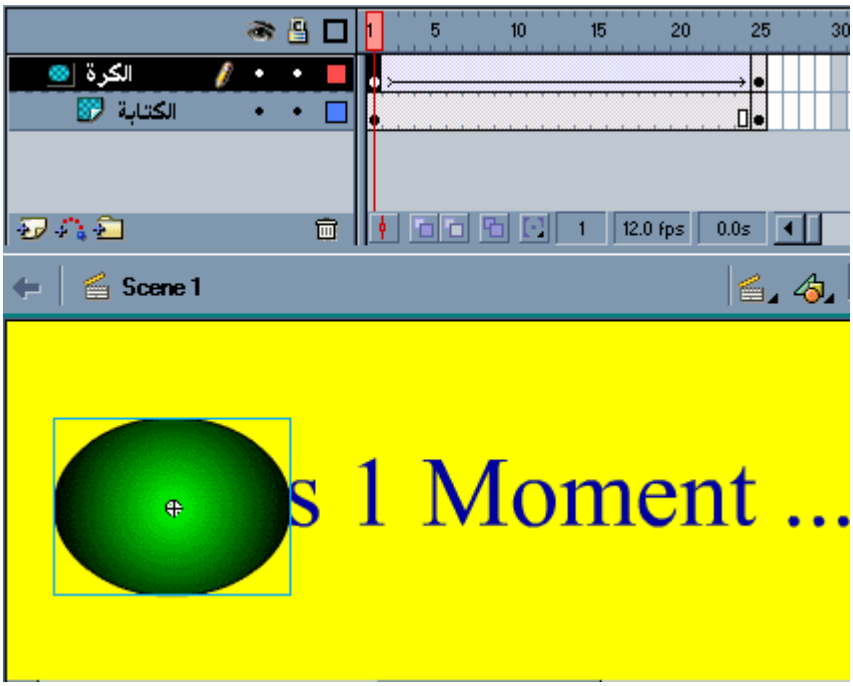
١. نحتاج إلى طبقتين الأولى للكتابة عليها والثانية لجعلها قناع، نقوم بإضافة طبقة ثانية من خلال الضغط على زر إضافة طبقة الأولى نسميها الكرة بالنقر المزدوج على اسمها والثانية نسميها كتابة.

٢. نختار طبقة الكتابة وعند الإطار١ باختيار أداة الكتابة نكتب النص وليكن One Moment Please!ونكبره حتى يشغل معظم مساحة الرسم ثم نحوله إلى رمز بالضغط على زر F8 من نوع Graphic ثم نقوم بإضافة إطار مفتاحي عند الإطار٢٥ بالضغط على الزرF6.

٣. الآن ننتقل إلى الطبقة الكرة نحددها بالنقر عليها وعند الإطار١ نقوم برسم الكرة عند بداية النص حيث لا يهم لون التعبئة لأنه لا يظهر عند العرض كما ذكرنا سابقاً، ثم نحول الكرة إلى رمز بالضغط على زر F8 من نوع Graphic ثم نقوم بإضافة إطار مفتاحي عند الإطار٢٥ بالضغط على الزرF6 نحرك الكرة بواسطة الماوس إلى نهاية النص.

٤. نقوم الآن بإضافة الحركة إلى الكرة حيث نضغط على الإطار الأول من طبقة الكرة بالزر الأيمن للماوس ونختار الأمر Craete Motion Tween .

٥. الآن نقوم بإضافة القناع لطبقة الكرة بالضغط عليها بزر الماوس الأيمن واختيار الأمر Mask وبذلك يتغير رمز الطبقة كما في الشكل رقم (١):



الشكل رقم (١)

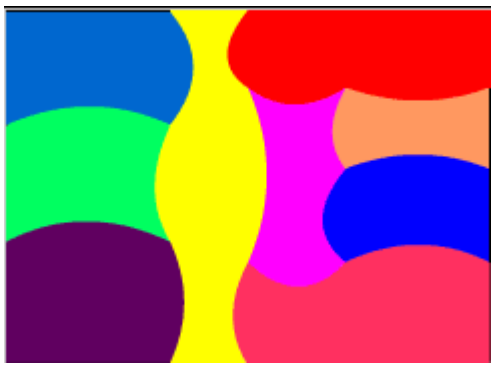
٦. الآن نقوم باختبار الفيلم بالضغط على Ctrl+Enter .

ملحوظة:- يمكن معاينة المشهد من شاشة الرسم من خلال تأمين طبقة القناع والطبقة المقنعة والتنفيذ بالضغط على زر Enter.

- سنقوم الآن بشرح مثال آخر عن تطبيقات طبقة القناع نظراً لأهميتها البالغة في جميع الأعمال الفنية، وفكرة مثالنا هي بسيطة جداً حيث تدور حول فكرة كتابة نص وتكبيره وبالتالي تغير الألوان ضمنه باستمرار وخطوات الدرس هي:

١. ننشئ طبقتين الأولى طبقة قناع Mask نسميها Text والثانية مقنعة Masked نسميها Colors .

٢. نختار طبقة Colors ونقوم برسم مستطيل ونلونه بألوان عشوائية حسب الرغبة مثلاً كما في الشكل رقم (٣) ونضيف إطار مفتاحي لنفس الطبقة عند الإطار ٤٥ بالضغط على مفتاح F6 ثم نقوم بتأمين تلك الطبقة لسلامتها:

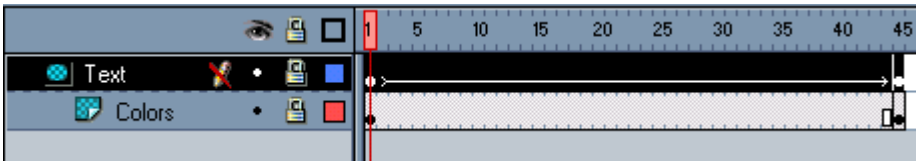


الشكل رقم (٣)

٣. الآن نختار الطبقة Text ونقوم بكتابة نص وليكن COLOR ثم نحوله إلى رمز بالضغط على مفتاح F8 من نوع Graphic ونسحبها بواسطة الماوس إلى فوق نافذة الرسم، ثم نضيف إطار مفتاحي أيضا عند الإطار ٤٥ بالضغط على مفتاح F6 ثم نسحب النص إلى أسفل نافذة الرسم.

٤. نقوم بإضافة الحركة للنص عن طريق النقر بالماوس بالزر الأيمن على الإطار ١ لطبقة Text ونختار الأمر Create Motion Tween .

٥. الآن نقوم بإضافة القناع لطبقة Text بالضغط عليها بزر الماوس الأيمن واختيار الأمر Mask وبذلك يتغير رمز الطبقة كما في الشكل رقم (٤):



الشكل رقم (٤)

٦. الآن نقوم باختبار الفيلم بالضغط على Ctrl+Enter .

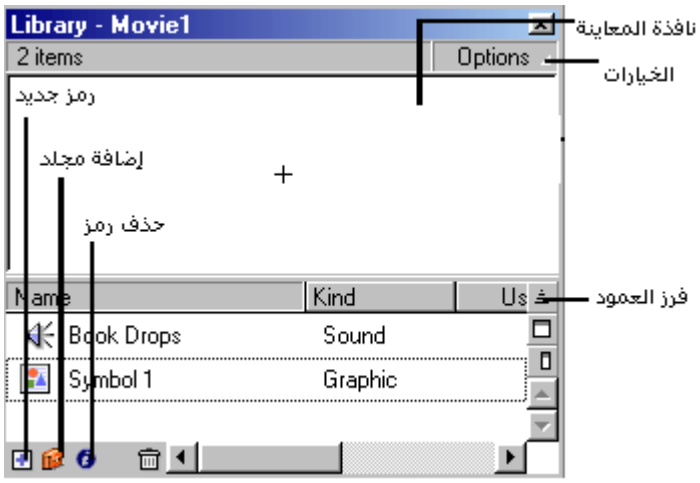
رابعاً- الطبقة المجلد:

فائدتها فقط هي تصنيف الطبقات ضمن مجلد ليسهل الوصول إليها .

التعامل مع المكتبات

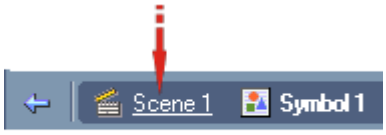
ستعلم في هذا الدرس بعض الأمور التي تسهل على المصمم العمل حيث تستخدم المكتبات لحفظ واستخدام جميع الوسائط، وكذلك استخدام القوالب المصممة مسبقاً وكذلك التعامل مع المكتبات المشتركة أعتبرها في نظري مهمة جداً، مع العلم أن استخدام الرموز يقلل من حجم الملف لأن فلاش لا يأخذ بعين الاعتبار عدد النسخ التي يتم توليدها من الرمز.

سنتعرف أولاً على نافذة المكتبة من الشكل رقم (١) التي تظهر من خلال الأمر **Library** من قائمة **Window** أو الضغط على **Ctrl+I**:



الشكل رقم (١)

يفضل دائماً إنشاء الرمز من خلال المكتبة من زر **Options** نختار الأمر **New Symbol** وذلك من أجل تحرير الرمز في نافذة مستقلة عن نافذة المشهد كما في الشكل رقم (٢) حيث نستفيد من وجود طبقات وشريط زمن مستقل لكل رمز مما يقلل من تعقيد أي فيلم، ثم نقوم بنسخها من المكتبة إلى نافذة الرسم:



الشكل رقم (٢)

و للانتقال إلى المشهد نضغط على **Scene** كما في الشكل السابق، ولا يمكن إضافة أي رسمة مستوردة أو منشأة ضمن فلاش إلى مكتبة المشروع إلا عن طريق الأمر **Convert to Symbol** من القائمة **Insert**،

ـ ولاستخدام عنصر موجود ووضعه ضمن الفيلم نختار أمر المكتبة **Library** من قائمة أمر النافذة **window** ، ثم نسحب العنصر المطلوب من المكتبة إلى نافذة الرسم.

وكافة الرموز المأخوذة من المكتبة لا يمكن تعديلها إلا عن طريقين:

١. كسر عناصر الرمز من خلال الأمر **Break Apart** من قائمة **Modify** أو الزر **Ctrl+B**.

٢. تعديل الرمز الرئيسي من نافذة المكتبة بالنقر المزدوج عليه مع ملاحظة أنه عند تغيير الرمز الرئيسي من المكتبة سوف يؤدي هذا الأمر إلى تعديل كافة النسخ المأخوذة لذلك الرمز.

ـ من أجل التحرير باستخدام برنامج خارجي:

١. انقر على العنصر المطلوب لتحديده.
٢. اختر الأمر **edit with** من قائمة **options** يظهر مربع حوار انتقاء برنامج التحرير الخارجي **select external editor** .
٣. ابحث عن البرنامج المناسب ، ثم انقر **open** .
٤. قم بتحرير العنصر ثم احفظه، يقوم فلاش بتحديث العنصر ضمن المكتبة .

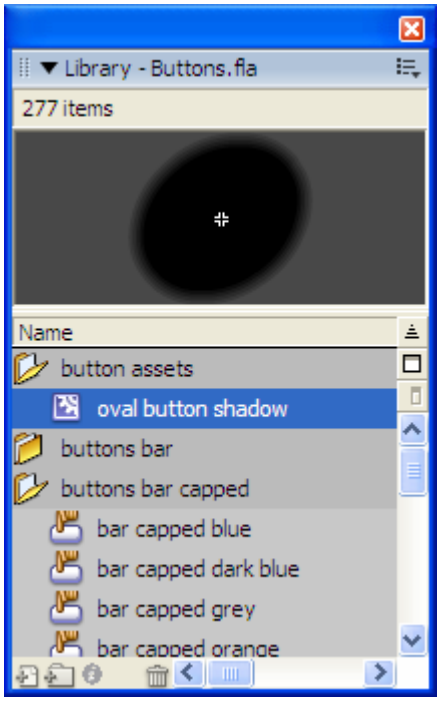
ـ تحديث العناصر المستوردة ضمن المكتبة :ـ

١. اختار الملف المستورد ضمن المكتبة .

٢. اختر أمر تحديث update من قائمة options .
٣. يظهر مربع التحديث update media، انقر على زر التحديث update .

المكتبة المشتركة في فلاش ٨:

يتضمن فلاش ٨ مجموعة كبيرة من الرسوم واللقطات والأزرار الجاهزة والفنية والأصوات ويمكن استعراض تلك الرسوم من خلال الأمر Common Libraries من قائمة Window كما في الشكل رقم (٣)



الشكل رقم (٣)

حيث يمكن استخدام المكاتب المشتركة من داخل أي مشروع من خلال سحب العناصر منها إلى مكتبة المشروع .

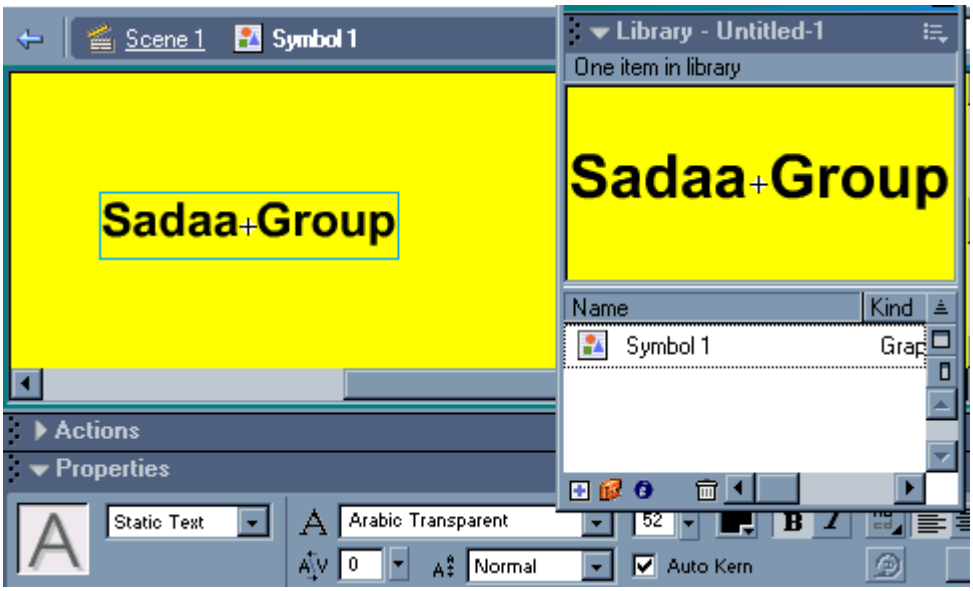
ـ إنشاء مكتبة خاصة أو جعل مكتبة المشروع مكتبة مشتركة لاستخدامها في مشاريع أخرى :-

و ذلك باتباع الخطوات التالية :-
١. قم بإنشاء أو استيراد العناصر التي تريد وضعها في المكتبة الخاصة .
٢. اختر أمر save من قائمة file يظهر مربع حوار حفظ الملفات باسم save as .
٣. ابحث عن مجلدات المكتبة libraries ، وهو المجلد الموجود ضمن المجلد الأساسي.
٤. اكتب اسماً للمكتبة ثم انقر save، حيث يتم حفظ المكتبة ضمن القائمة الفرعية Common Libraries من قائمة Window .

ـ لمعاينة اللقطة السينمائية المنشأة ضمن المكتبة والمنسوخة لنافذة الرسم من خلال الأمر Test Movie من قائمة Control.

مثال شامل فكرته تقوم على إنشاء نصوص بمقاييس مختلفة تتلاشى كلما كبرت ولكن سوف ننشئ هذا المثال مع الاستفادة من خصائص المكتبات وإنشاء القوالب وسوف يكون درساً ممتعاً لنبدأ:

١. نقوم بفتح نافذة المكتبة من خلال الضغط على Ctrl+l ومن قائمة Option التابعة لنافذة المكتبة نختار الأمر New Symbol تختار الرمز من نوع Graphic تفتح لنا نافذة مستقلة لتحرير الرمز نقوم بكتابة نص ما باستخدام أداة النص A وليكن النص Sadaa Group وذلك كما في الشكل رقم (١) حيث يمكن تغيير أعدادات النص من قائمة Text:

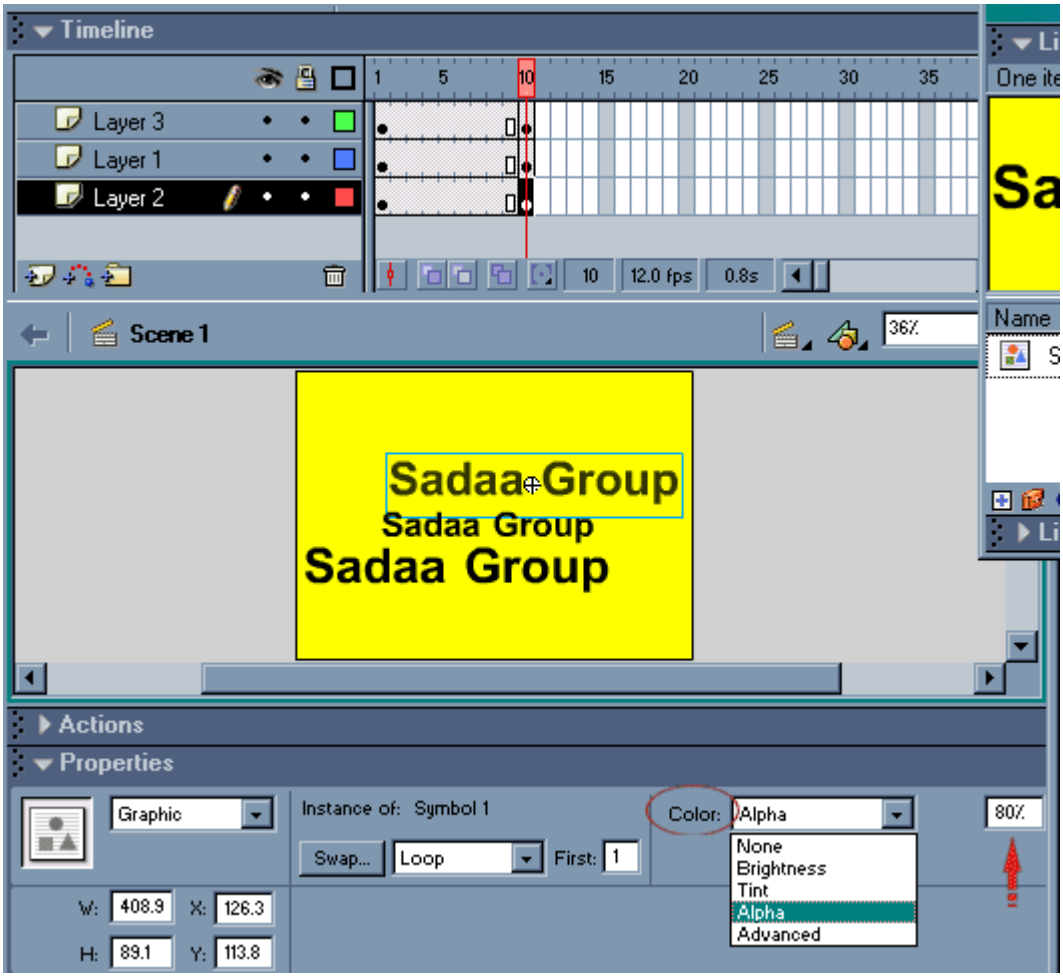


الشكل رقم (٢)

٢. نقوم بالعودة إلى نافذة تحرير المشهد بالضغط على رز **Scene1** ثم نقوم بسحب الرمز من المكتبة إلى نافذة الرسم ونقوم بوضعه في وسط النافذة ثم نضيف إطار مفتاحي عند الإطار ١٠ بالضغط على مفتاح **F6**.

٣. نضيف الآن طبقتين ونجعل الطبقة التي أنشأنا عليها النص طبقة متوسطة بين الطبقتين الجديدتين وعلى كل طبقة نقوم بإضافة الرمز من المكتبة عند الإطار الأول ونحاول أن نجعل النصوص الثلاثة على مستوى واحد.

٤. نختار الآن الطبقة الدنيا عند الإطار ١٠ ونضيف إطار مفتاحي بالضغط على مفتاح **F6** ثم نكبّر النص عند نفس الإطار من خلال أداة التحجيم نقوم بالسحب إلى الأعلى واليمين بمضاعفة حجم النص وكذلك الطبقة العليا ولكن إلى الأسفل واليسار كما في الشكل رقم (٣) نم نقوم بإضافة تأثير **Alpha** لكل من الطبقة العليا والدنيا عند الإطار ١٠ من لوح **Properties** ونختار من قائمة **Color** الأمر **Alpha** ونضع القيمة تساوي الصفر كما في الشكل رقم (٣):



الشكل رقم (٤)

٥. الآن نضيف الحركة من نوع Motion من لوح Properties لكل من الطبقة العليا والدنيا عند الإطار ١ ثم ننفذ الحركة بالضغط على الزر Ctrl+Enter .

☜ إن إحدى الفوائد من استخدام الكائنات النصية في فلاش هو إمكانية إعادة استخدام لمؤثر نصي تم بنائه مسبقاً دون الحاجة لإعادة إنشاؤه وتكون الخطوات هي:

١. نفتح المؤثر المنشأ مسبقاً وليكن المثال الذي شرحناه سابقاً مع مراعاة أن يكون قد إنشأ بنفس ما قمنا بشرحه.

٢. ثم نقوم بفتح نافذة المكتبة بالضغط على Ctrl+l يظهر لنا الرمز Symbol1 كما في الشكل السابق رقم (٢) نقوم بالنقر المضاعف بالماوس على ذلك الرمز لتحريره ضمن نافذته المستقله كما في الشكل رقم (٢).

٣. باستخدام أداة النص A نقوم بالنقر على النص فيتيح لنا فلاش إمكانية إعادة كتابة النص نقوم بحذف النص وكتابة النص الجديد وليكن Yasin Fares ثم ننتقل إلى نافذة المشهد بالضغط على زر Scene1.

٤. نرى أن المشهد قد تحول إلى النص الجديد مع الحفاظ على نفس الحركة نقوم باختبار المشهد بالضغط على Ctrl+Enter .

إدراج ملف الفلاش في صفحة ويب

إن إدراج فيلم الفلاش في صفحة ويب يعتبر من الأمور المهمة جداً ، حيث أن الفلاش أصبح يستخدم وبشكل كبير من أجل إضافة التأثيرات الحركية والجمالية في موقع الانترنت إضافة إلى الأكشن سكريبت وما تعطيه من إمكانيات تقنية

... ولذلك فإن تعلم هذا الأمر يعتبر من الأمور الأساسية وهذا الأمر بسيط جداً وهو ينقسم إلى قسمين إدراج ملف الفلاش عن طريق Front Page وإدراجه عن طريقة لغة html ....

أدراج ملف الفلاش في Front page :-

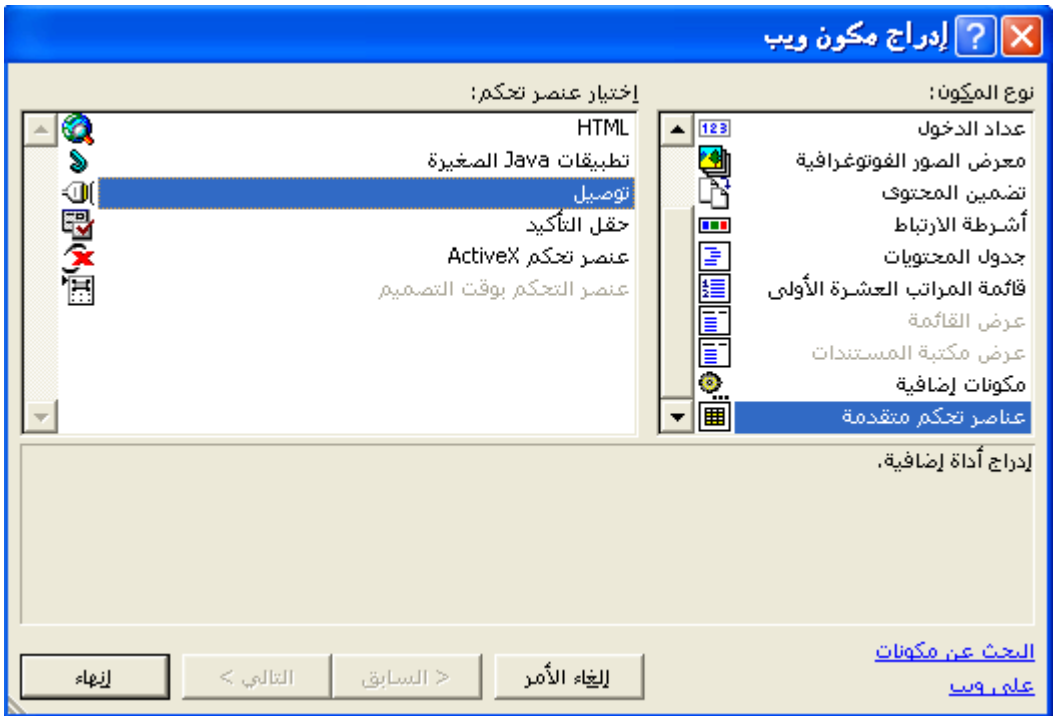
١ - قم بعمل فيلم فلاش .

٢ - من القائمة الرئيسية File للفلاش أختر Export Movie احفظ الملف في القرص الصلب باسم sadaa على سبيل المثال بامتداد swf اضغط Save ثم OK ثم OK مرة أخرى .

***لاحظ بأن البرنامج قد قام بتصدير الفيلم على هيئة Flash Player باسم sadaa.swf الملفات ذات الامتداد swf هي التي يمكن أن تدرج في صفحات الويب وتعرض في الإنترنت .*

٣ - الآن نقوم بفتح برنامج Front page ثم نفتح الصفحة اللي سنصدر العمل إليها ثم نختار من القائمة الرئيسة insert ← web component ← advansed ← Plug-in

أما بالنسبة لمستخدم Front page باللغة العربية يختار من القائمة الرئيسية إدراج ← مكون ويب ← عناصر تحكم متقدمة ← توصيل ← الشكل (١) .



الشكل (١)

٤ -الآن فتح لنا مربع حوار نقوم بتحديد الملف الذي سندرجه في Front page وذلك بالضغط على Browser ثم نختار الملف المطلوب ثم OK بعد ذلك قم بتحديد أبعاد الفيلم من طول وعرض .... الشكل (٢) .... هذا كل شيء في Front page ....

الشكل (٢)

إدراج ملف الفلاش عن طريق لغة html :-

طبعاً أولاً عليك القيام بالعمليتين السابقتين ١ ، ٢ حتى تقوم بحفظ الملف بامتداد swf .

<embed width="550" height="400"
src="http://www.sadaagroup.com/sadaa.swf"

type="application/x-shockwave-flash">

انسخ الكود أعلاه وألصقه في المكان الذي تريد أن تدرج فيه الفيلم في صفحة الويب ، لا تنسى تغير وصلة الملف لتتلاءم مع عنوان موقعك الشخصي والمجلد الذي نقلت إليه ملف الفلاش إضافة إلى أبعاد فيلم الفلاش الخاص بك حيث أننا هنا وضعنا الأبعاد الافتراضية وهي ٤٠٠ X ٥٥٠ بكسل (لا تنسى الأبعاد بالبكسل) .

وهكذا تعلمت كيف تدرج فيلم فلاش في صفحة ويب وقد لاحظت أن هذا الأمر بغاية السهولة وخاصة إذا تم في

Front page

التعامل مع الأصوات في فلاش

إحدى المزايا التي يتمتع بها برنامج فلاش ويدعمها بقوة هي إمكانية إضافة الأصوات للأفلام حيث يتعامل فلاش مع نوعين من الأصوات (أصوات الأحداث، الأصوات المتدفقة) :-

١ـ حيث تستخدم أصوات الأحداث للتركيز على أحداث معينة في الرسم ولا تعمل حتى يحمل الموقع والصوت بالكامل .

٢ـ أما الأصوات المتدفقة فتعمل مع عرض الرسم المتحرك وعند انتهاء عرض الرسم يتوقف الصوت فوراً حيث يحمل الصوت على دفقات وهذا النوع من الأصوات هو الأنسب لتصميم مواقع الإنترنت ولا انصح باستخدام الخيار

Loop مع الأصوات المتدفقة الذي سوف نتعرف عليه لاحقاً ولإضافة صوت متدفق من خلال اللوح Properties نجعل خيار المزامنة Sync = Stream .

- ولاستيراد الصوت إلى فلاش من قائمة File نختار الأمر Import ونبحث على الملف المطلوب من نوع wav أو أي صيغة أخرى للصوت ثم نضغط Open، ويتم عرض الملف الصوتي ضمن المكتبة التي يمكن إظهارها من خلال الضغط على Ctrl+l ويضاف إلى اللوح Properties ضمن قسم Sounds لإضافته إلى الفيلم فيما بعد، ويفضل وضع كل صوت على طبقة منفصلة.

لانتقاء الصوت من المكتبة:

١. انتق الصوت عن طريق النقر على اسمه .
٢. استمع للصوت عن طريق النقر على زر التشغيل ply ضمن نافذة المكتبة .
٣. أضف الصوت إلى الفيلم عن طريق سحبه من مكتبة الأصوات إلى منطقة العمل بعد تحديد الإطار الذي تريد عنده إضافة الصوت نضيف إطار مفتاحي فارغ عن طريق الضغط على مفتاح F5.
٤. عندما تقوم بتعيين الصوت يظهر الشكل المتموج للصوت ضمن الإطار المفتاحي ثم يمتد عبر شريط الوقت ل ليغطي العدد الدقيق من الأطر التي سينطلق الصوت عبرها ، كما في شكل رقم (٢).

الشكل رقم (٢)

حيث يمكن تشغيل أصوات مختلفة في وقت واحد من خلال الإدراج ضمن إطارات متعددة كم في الشكل السابق .

التعامل مع الأزرار

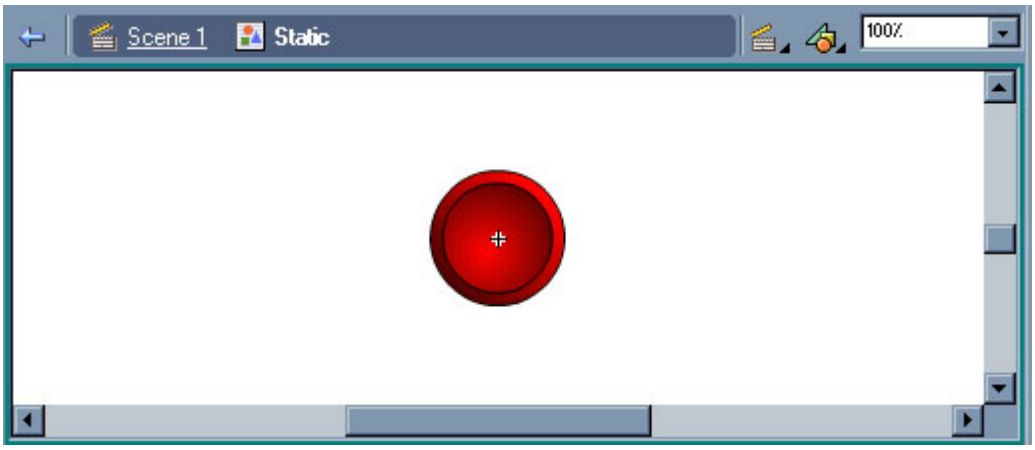
سنتعلم في هذا الدرس كيفية مبادئ تحويل الفيلم في فلاش من الشكل السكوني إلى الشكل التفاعلي حيث يستطيع المشاهد التحكم ببعض الأمور التي سوف تظهر في الفيلم وكل ذلك عن طريق الأزرار، ومن الأفضل دائماً أن تقوم بتصميم الجزء التفاعلي من بداية المشروع.

سنقوم أولاً بتعلم كيفية إنشاء زر متغير ثم كيفية ربط ذلك الزر بحدث معين من لوح Actions أو عن طريق استخدام أزرار جاهزة من المكتبة المشتركة Common Libraries.

إنشاء زر متغير:-

لإنشاء زر متغير نتبع الخطوات التالية:-

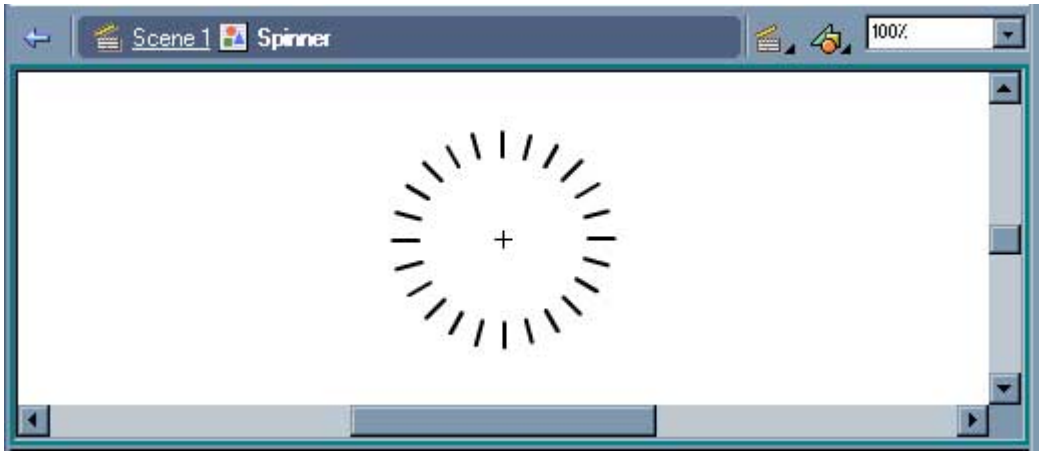
١. بعد فتح مشروع جديد وتجهيز أبعاده حسب المطلوب نقوم بإدراج رمز جديد من نوع Graphic من قائمة Insert نختار الأمر New Symbol ونسميه باسم Static ثم نقوم برسم زر أياً كان شكله مثلاً دائرة مجوفة كما في الشكل رقم (١):-



الشكل رقم (١)

ثم نقوم بالانتقال إلى المشهد الحالي بالضغط على زر Scene1 وبذلك نكون قد حفظنا الرمز ضمن مكتبة المشروع.

٢. الآن نقوم برسم التوهج للزر أثناء مرور الماوس من فوقه عن طريق إدارج رمز كما في السابق ولكن نسميه Spinner كما في الشكل رقم (٢):

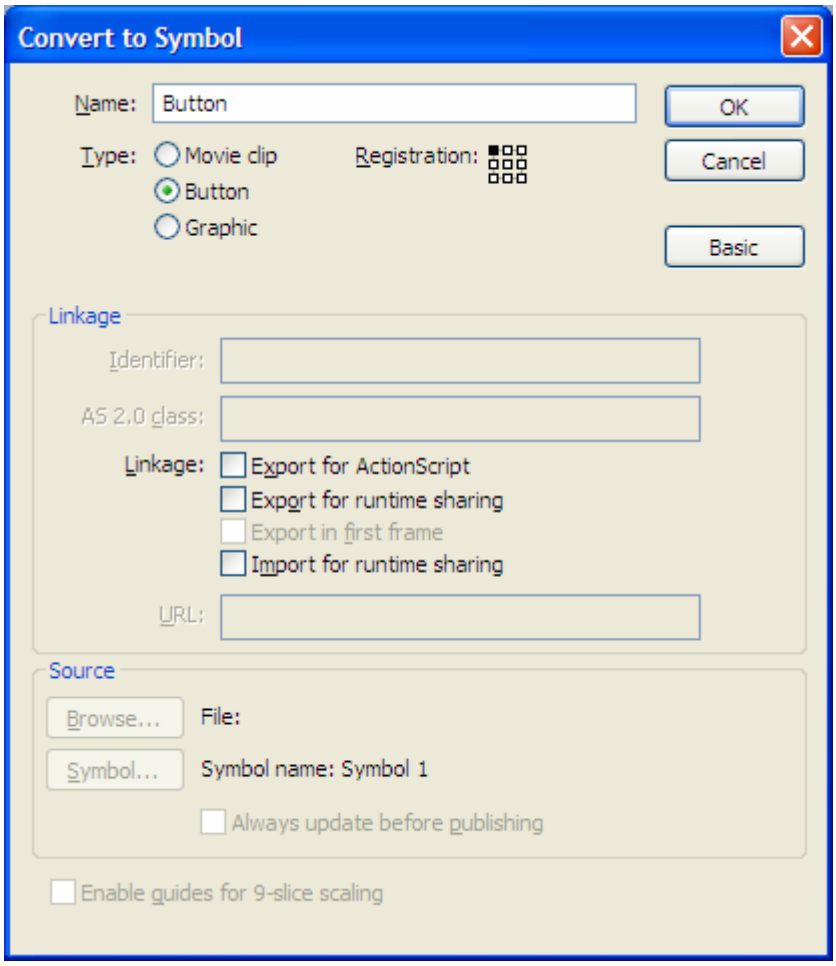


الشكل رقم (٢)

ثم نقوم بالانتقال إلى المشهد الحالي بالضغط على زر Scene1 وبذلك نكون قد حفظنا الرمز ضمن مكتبة المشروع.

٣. الآن نقوم بإنشاء لقطة سينمائية لحالة التوهج للزر عن إدراج رمز ولكن من نوع Movie Clip باسم Spinner Movie نفتح لنا النافذة الخاصة لإنشاء اللقطة السينمائية حيث نقوم بفتح نافذة المكتبة عن طريق الضغط على Ctrl+I ونختار رمز التوهج Spinner ونقوم بسحبه إلى نافذة اللقطة السينمائية عن طريق الماوس ونحاول توسيطة عند إشارة + ثم نقوم بإضافة إطار مفتاحي عند الإطار ١٠عن طريق الزر F6 ونضيف الحركة للرمز عن طريق النقر بالزر الأيمن للماوس عند الإطار ١ ونختار الأمر Create Motion Tween وعند الإطار ١٠ نحدد الرمز ونقوم بتدويره ربع دورة من خلال أداة التحجيم من خارج الإطار عندما يصبح شكل الماوس دائري نقوم بالتدوير وبذلك نكون قد أنهينا لقطة التوهج ننتقل إلى المشهد Scene1 .

٤. سوف نقوم الآن بإنشاء الزر المتغير وذلك عن طريق إدراج رمز من نوع Button من قائمة Insert نختار الأمر New Symbol ونسميه Button كما في الشكل رقم (٣):-



الشكل رقم (٣)

بذلك يتم فتح نافذة تحرير الزر ويكون في شريط الزمن أربع أطر نقوم بشرح مبسط عن كل إطار،

الحالات أربعة للأزرار:

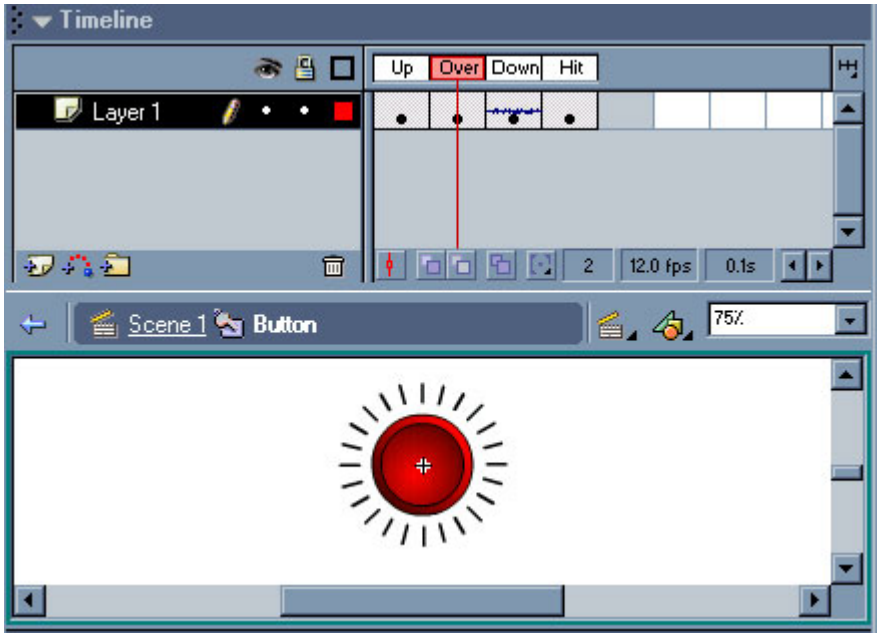
- حالة Up: غير منضغط ولا يكون مؤشر الماوس فوقه.

- حالة Over: تحدد الشكل الذي سيظهر عليه الزر عندما يكون مؤشر الماوس فوق المنطقة النشطة الخاصة بالزر.

- حالة Down: تحدد شكل الزر عندما يكون منضغطاً للأسفل.

- حالة Hit: تشير إلى المنطقة النشطة للزر المحدد ويجب أن يكون الشكل أكبر من باقي الأشكال ليتم تضمين باقي الأشكال فيها وعلى كل حال لن تكون تلك المنطقة مرئية في الفيلم النهائي، وإذا لم نحدد تلك المنطقة فإن فلاش يستخدم صورة الإطار Up لإطار Hit.

٥. نختار الإطارUp عن طريق الماوس ثم نفتح نافذة المكتبة ونسحب الرمز Static إلى نافذة تحرير الزر ونحاول توسيطة ثم نضيف أطر مفتاحية عند كل إطار عن طريق الزر F6، وعند الإطار Over نقوم بسحب اللقطة السينمائية Spinner Movie ونحاول توسيطها، وعند الإطار Down نقوم بقلب الزر لإعطاء حالة الإنضغاط للزر عن طريق أداة التحجيم نقوم بتدوير الدائرة نصف دورة ويمكن إضافة صوت من المكتبة المشتركة كما شرحنا في الدرس السابق عن طريق سحبه ووضعه ضمن إطار Down أيضاً، وبذلك تصبح نافذة تحرير الزر كما في الشكل رقم (٤):-



الشكل رقم (٤)

وبهذا نكون قد أنهينا الدرس نخرج الآن إلى المشهد Scene1 ثم نقوم بمعاينة العمل بالضغط على Ctrl+Enter .

استيراد الصور النقطية والتعامل معها

هذا الدرس يفتقر إلى التطبيق العملي إلا أن المعلومات التي يتضمنها بالغة الأهمية فإن فلاش يسمح لك بإنشاء وتحريك الصور، أيضاً يسمح لك باستيراد ومعالجة الصور النقطية والصور المتجهة التي تم إنشاؤها في تطبيقات أخرى فمن المستحيل أن نقوم برسم كافة الرسوم التي نحتاجها وأحياناً نكون مضطرين لاستيراد بعض الصور اللازمة للفيلم، حيث أن معظم البرامج الرسومية تتعامل مع نوعين من الصور وهي (الصور المتجهة Vector graphics والصور النقطية Bitmaps) وسنوضح المصطلحين السابقين بشيء من الإيجاز وهما:

أولا- الصور المتجهة: **هي الصور التي تستخدم الخطوط والمنحيات التي تشكل المظهر الخارجي للصورة ويتم تحديد اللون من خلال لون خطوط الرسم، ويمكن تغيير خصائص الخطوط والمنحنيات التي تصف الشكل بدون أن يؤثر ذلك على جودة المنظر، حيث أن الصور المتجهة تملك خاصية استقلالية الدقة.**

ثانياً- الصور النقطية: **يتم توصيف الصورة النقطية باستخدام النقاط الملونة والتي تسمى بيكسل Pixels والتي يتم توزيعها على شبكة، وعند تحرير صورة نقطية يمكن تغيير البكسلات وليس الخطوط والمنحنيات، إن هذا النوع من الصور يملك خاصية عدم استقلال الدقة لآن المعلومات التي تصف الصورة مثبتة إلى شبكة بقياس محدد وبالتالي عند تحرير الصورة النقطية يمكن أن تتغير جودتها.**

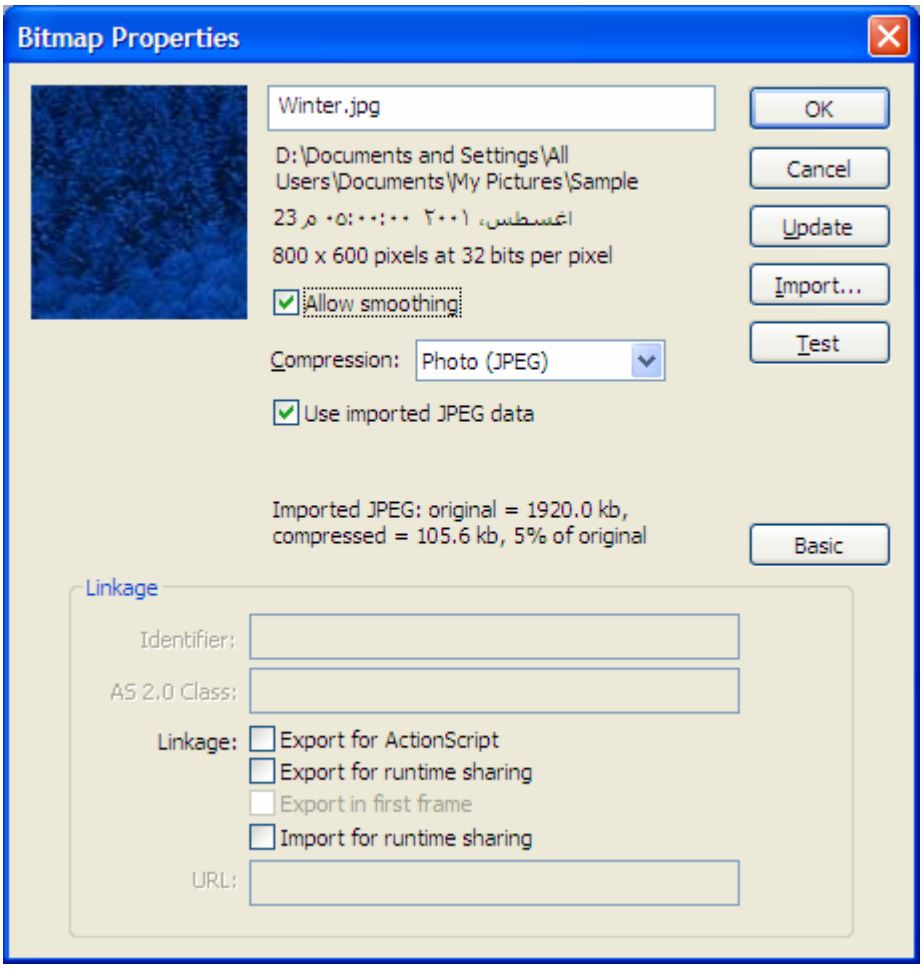
لاستيراد صورة نقطية إلى فلاش:-

١. اختر أمر استيراد Import من قائمة File .
٢. ابحث عن الصورة الذي تريد استيراده .
٣. أنقر على زر الفتح Open لاستيراد الملف .
عند استيراد الصور تظهر في مكتبة المشروع.

- إن استيراد الصور النقطية يؤدي إلى زيادة حجم الملف التي يمكن تصغير حجمها بطريقتين:
أولاً- خيار الضغط من المكتبة من قائمة Options نختار الأمر Properties Bitmap.
لتحرير إعدادات تصدير الصورة النقطية:-

١. انتق الصورة المستوردة ضمن مكتبة المشروع.
٢. افتح قائمة خيارات لوح المكتبة Option Library ثم اختر أمر الخصائص Properties

انظر الشكل رقم (١).



الشكل رقم (١)

٣. خيار السماح بالنعومة **Allow smoothing** يكون فعالً بشكل افتراضي وهو يؤدي إلى تطبيق مفعول النعومة على الصورة .

٤. انقر على السهم الموجود في الطرف الأيمن من قائمة خيارات الضغط **Compression** ثم اختر **Photo (JPEG)** أو **Lossless(PNG/GIF)** .

٥. إذا انتقيت خيار الضغط الملائم للصور الفوتوغرافية فسيظهر قسم الجودة **quality** ضمن مربع الحوار ، أزل اشارة التحديد من مربع خيار استخدام الجودة الافتراضية للمستند **use document default quality** .

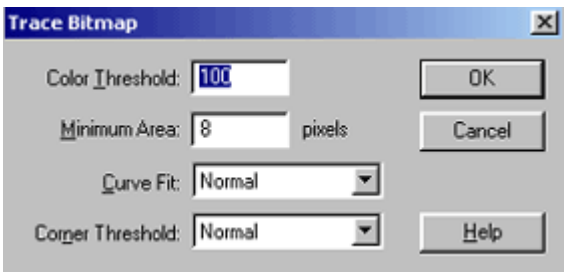
٦. أنقر على زر الاختبار **test** حيث يتم تحديث الصورة المصغرة ضمن نافذة المعاينة .

ثانياً- تحويلها إلى صورة متجهة نختار الأمر Trace Bitmap من قائمة Modify.

لتحويل الصور النقطية إلى عناصر رسومية:

١. حدد الصورة التي تريد تحويلها .

٢. اختر أمر تتبع الصورة **Bitmap Trace** من قائمة أمر المعالجة **Modify** لفتح مربع حوار تتبع الصورة ك كما في شكل (٢).



الشكل رقم (٢)

٣. ضمن حقل نطاق الألوان Threshold Color ، اكتب قيمة تتراوح بين ١ و ٥٠٠ . وهذه القيمة هي التي تحدد مدى التقارب اللوني بين البكسلات المتجاورة في الصورة .
٤. ضمن حقل الحد الأدنى Area Minimum ، أكتب قيمة تتراوح بين ١ و ١٠٠٠ ، وهذه القيمة هي التي ت تحدد مقدار البكسلات المتجاورة .
٥. انتق خياراً من قائمة ملاءمة المنحنى Fit Curve . والخيارين المناسبين لمعظم الحالات Normal ، Smooth .
٦. انتق خياراً من قائمة نطاق الزاوية Threshold Corner ، هذا الإعداد هو الذي يحدد كيفية قيام فلاش ب برسم الزوايا . .
٧. أنقر موافق Ok لتحويل الصورة إلى عناصر رسومية .
يوضع الصورة النقطية الأصلية إلى اليسار ونسختين مستخلصتين إلى اليمين .

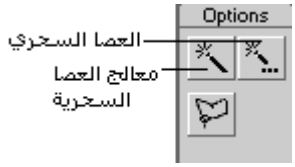
- وأفضل برنامج لعملة التحويل تلك هو برنامج (Free Hand)، مع تجنب تحويل الصور النقطية المعقدة لعدم وضوحها، ونلاحظ هنا أنه عند تحويل الصورة إلى رسوم متجهة أننا نستطيع تحرير الصورة وإزالة الأجزاء غير المرغوب بها كما سنرى في الفقرة التالية.

استخدام أمر الكسر والتفريق:-

الكسر والتفريق Break apart يعتبر طريقة أخرى من طرق تحويل الصورة النقطية إلى عناصر رسومية قابلة للتحرير .

لكسر وتفريق الصورة النقطية:

١. حدد الصورة النقطية التي تريد كسرها وتفريقها .
٢. اختر أمر الكسر والتفريق Apart Break من قائمة أمر المعالجة Modify ، بعد تفعيل الأمر ، يختفي صندوق الإحاطة الذي يحدد أطراف الصورة ، استخدم معالج العصا السحرية Wand Magic لانتقاء وتحديد المناطق اللونية المختلفة ، أو أداة القطارة لتحويل الصورة النقطية إلى لون داخلي .
٣. لانتقاء وتحديد المناطق اللونية المتماثلة بواسطة العصا السحرية قم بكسر وتفريق الصورة نقطية .
٤. انتق أداة الحبل lasso  ، ثم انقر على magic wand كما في الشكل رقم (٣).



الشكل رقم (٣)

٥. انقر على زر magic wand properties لفتح مربع حوار إعدادات العصا السحرية .
٦. ضمن حقل النطاق Threshold ، اكتب قيمة تتراوح بين ٠ و ٢٠٠ وهذه القيمة تحدد مدى التقارب اللوني بين البكسلات المتجاورة .
٧. انتق أحد الخيارات من قائمة النعومة Smoothing وهذا الخيار هو الذي يحدد مدى نعومة حواف المنطقة المحددة .
٨. انقل الأداة نحو منطقة العمل ، لاحظ أن مؤشر الأداة يتحول من مؤشر الحبل إلى مؤشر العصا السحرية عند د دخوله منطقة الصورة النقطية ، اضغط على مفتاح Shift ثم انقر على المزيد من المناطق اللونية لإضافتها إ ا الى التحديد، ثم ننقر نقراً مزدوجاً لإغلاق المضلع وبعد تفريغ الصورة يمكن إضافة حد خارجي للرسم لتغطية ا ا النتوءات الحاصلة باختيار لون الحد المناسب باستخدام أداة زجاجة الحبر 

☞ أخيرا نقوم بتجميع الجزء المحرر باختيار الأمر Group من قائمة Modify.

ولتحويل الصورة النقطية إلى رمز عن طريق الأمر Convert To Symbol من قائمة Insert .

لتحويل الصورة النقطية إلى لون داخلي :-

١. حدد الصورة النقطية التي تريد استخدامها كعينة للون الداخلي .

٢. اختر أمر الكسر والتفريق Break apart من قائمة Modify .

٣. انتق أداة القطارة DROPPER  ، ثم انقر على الصورة التي تم كسرها وتفريقها تتحول أداة القطارة إلى ا اللون الداخلي.

٤. بعد اخذ عينة من الصورة النقطية التي تم كسرها وتفريقها ، احذف الصورة ، وتذكر أن فلاش قد حفظ الصورة الأصلية ضمن مكتبة المستند .

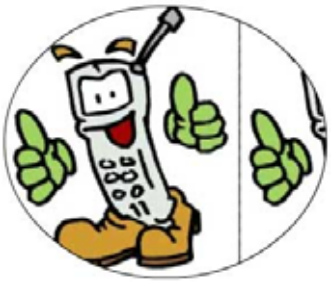
لتطبيق مفعول اللون الداخلي المأخوذ من الصورة النقطية مع إمكانية تغير أبعاد الصورة، اتبع أحد الطرق التالية :

- بعد إنشاء اللون الداخلي من الصورة النقطية ، قم برسم عنصر ما على مسرح العمل ، وذلك إما أداة البيضاوي OVAL أو المستطيل rectangle ،يقوم فلاش بملء العنصر بلون العينة كما في الشكل رقم (٤).

- بعد إنشاء اللون الداخلي من الصورة النقطية انتق أداة الفرشاة ثم انقل الأداة نحو منطقة العمل لرسم بعض ا ا الضربات اللونية بالفرشاة وبلون النقش الذي تم استخلاصه من الصورة النقطية .

- لتعبئة العنصر الموجود باللون الداخلي المستخلص من الصورة النقطية ، أنتق أداة اللون الداخلي ثم انقر على ا العنصر.

الصورة الأساسية



الصورة بعد تحويلها إلى لون داخلي

الشكل رقم (٤)

وبذلك نكون قد تعلمنا كيفية التعامل مع الصور النقطية وتحريها والاستفادة منها في إعطاء الأفلام التي نقوم تصميمها حركة جمالية وفنية باستخدام الصور النقطية .

لغة الاكشن اسكربت Action Script

لغة الاكشن اسكربت Action Script هى مجموعة من الاجراءات و الاحداث المتسلسلة التى تخبر الكمبيوتر ماذا يفعل بالضبط .

نستفيد من لغة الاكشن اسكربت فى عمل الحركات Animation ، التنقل داخل الفيلم Navigation ، التعامل مع مدخلات المستخدم لملف الفلاش User Input ، التعامل مع البيانات Get Data ، عمل العمليات الحسابية Calculations ، العمل مع ملفات الصوت Play Sounds، و عمليات اخرى كثيرة .

بداية نتسأل فى اى عناصر داخل الفلاش يتم كتابة اكواد لغة الاكشن اسكربت ؟

١- الخط الزمنى Scripts in the Timeline .

٢- الازرار Scripts Attached to Buttons.

٣- الموفى كليب.Scripts Attached to Movie Clips.

تاريخ لغة الاكشن اسكربت:

ActionScript ١.0 من الاصدار الرابع من برنامج فلاش و حتى الاصدار السادس

Action Script 2.0 من الاصدار السابع حتى الاصدار الثامن .

Action Script 3.0 من الاصدار التاسع .

ملاحظة : **قد نرمز للغة الاكشن اسكربت بالرمز AS 3.0**

الفصل الاول

اساسيات الاكشن اسكربت Action Script 2

يمكننا كتابة كود الاكشن اسكربت من خلال محرر الكود الخاص بهذة اللغة و هو مدمج داخل البرنامج و يمكننا فتحة من خلال النافذة actions و لفتح اللوحة Actions انقر على القائمة windows اختر actions او اضغط f9 من لوحة المفاتيح .

و يتم تنفيذ الكود من خلال اضافتة الى الكادرات Frames او القطع الحركية MovieClip او الازرار التى يتم انشائها داخل العمل .

لكن لاحظ ان يتم تنفيذ الامر عند وقوع حدث معين على هذا الزر مثال على هذة الاحداث الخاصة بالازرار Buttons :-

الحدث Event	شـــرح الـحـــدث
Press	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح الفأرة الايسر و قبل عودتة للوضع الطبيعى .
Release	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح الفأرة الايسر ثم ترك مفتاح الفأرة مع وجود المؤشر فوق المفتاح
releaseOutside	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح الفأرة الايسر ثم ترك مفتاح الفأرة ابتعاد المؤشر عن المفتاح
keyPress	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح فى لوحة المفاتيح .
rollOver	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بتمرير المؤشر فوق الزر .
rollOut	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بتمرير المؤشر فوق الزر و الابتعاد بالمؤشر عن الزر.
dragOver	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على الزر و سحب الزر مع الضغط فوقة .
dragOut	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على الزر اثناء وجود المؤشر فوق الزرثم تحريك المؤشر بعيدا عن الزر

الاحداث الخاصة بال movieClip

الحدث Event	شـــرح الـحـــدث
Load	حدث تحميل القطع الحركية
Unload	حدث خروج القطع الحركية من مسرح العمل
enterFrame	عند بداية تحميل الفريم الاول للموفى كليب و يستمر حتى ينتهى الموفى كليب .
mouseDown	عند الضغط على زر الفأرة الايسر
mouseUp	عند تحرير زر الفأرة الايسر
mouseMove	عندما يتحرك مؤشر الفأرة
keyDown	عند الضغط على مفتاح من لوحة المفاتيح
keyUp	عند تحرير الضغط عن مفتاح من لوحة المفاتيح
Data	

- اعتقد انة حان الوقت لكى نجرب اول اسكربت لنا مع هذة اللغة:
- قم بعمل فيلم جديد ثم بعد ذلك قم بعمل ثلاث كادرات key frames و ذلك بالضغط على الفريم الاول فى الشريط الزمنى ثم الضغط على f7 من لوحة المفاتيح مرتين و باستخدام مهاراتك العادية التى اكتسبتها من العمل على برنامج فلاش قم بعمل ثلاث اشكال مختلفة كل شكل فى فريم خاص بة من الثلاث فريمات ، الان قم بعمل تشغيل تجريبى للفيلم من خلال الضغط على مفتاحى ctrl+enter من لوحة المفاتيح ، سوف تشاهد الفيلم يعيد تكرار نفسة كلما انتهى.
- ارجع للملف الاصلى و قف بالمؤشر عند الفريم الثانى و اضغط f9 فيظهر لك نافذة محرر كود الاكشن اسكربت و اكتب stop(); . و اغلق النافذة و قم بتجربة العمل

سترى ان الفيلم لم يعرض منة الا الفريم الاول و الثانى ثم بعد ذلك توقف و لم يكرر نفسة من جديد و من هنا نكون رأينا اول فائدة من هذة اللغة و هى التحكم فى الفيلم .

كيف يفكر الكمبيوتر؟؟

فى الحقيقية الكمبيوتر لا يفكر بل يأخذ التعليمات و ينفذها بحذافيرها اى لا يضيف او يفكر هل التعليمات التى قمت بأعطائها له منطقية ام لا .
و لغة الاكشن اسكربت هى مجموعة من التعليمات تكتب فى اسكربت داخل الفلاش لكى تخبر الكمبيوتر ماذا يفعل بالضبط .
و لكى تستطيع تعلم البرمجة يجب ان تفكر كالكمبيوتر كل امر تكتبة فكر بة كأنك انت الكمبيوتر او بمعنى ادق كأنك برنامج فلاش و تصور تسلسل هذة الاوامر و التعليمات و فكر كيف تنفذها و ما هو ناتجها .
المبرمج الحقيقى ليس فقط من يستطيع كتابة كود برنامجة من البداية للنهاية بل ايضا من يستطيع فهم اكواد الاخرين و تعديلها للأفضل يجب ان تتعلم من الاخرين و تحلل الافكار لكى تكون قادرا على طرح افكارا جديدا و لكى لا ترهق نفسك فى التفكير فى حلول لمشاكل سبقك اخرين لحلها .
مثال سوف نقوم بتحليل كود قمت بكتابتة و سوف تستفيد من هذا التحليل كثيرا عندما تنتهى من قراء ة الكتاب ككل

```
on (press) {
   var myVariable:Number = 7;
   var myOtherVariable:String = "Macromedia";
   for (var i:Number=0; i<10; i++) {
      trace(i);
      if (myVariable + 3 == 5) {
         trace(myOtherVariable);
      }
   }
}
```

السطر الاول تحليلة اننى قمت بعمل زر فى الفلاش و قمت بكتابة كود لا يتم تنفيذة الا عند الضغط على الزر والامر المسئول عن ذلك هو on (press) و لا حظ انه لا يمكن استخدام الامر السابق الا مع الازرار .
اما كل ما هو مكتوب بين العلامتين { } هو كود يتم تنفيذة عند الضغط
السطر الثانى قمت بتعريف متغير رقمى اسمة myVariable و اسندت لة القيمة ٧ .
السطر الثالث قمت بتعريف متغير نصى اسمة myOtherVariable و اسندت لة القيمة Macromedia .
السطر الرابع استخدمت التكرار من خلال عبارة for .
السطر الخامس استخدمت الدالة trace لرؤية المخرجات .
السطر السادس استخدمت التحكم فى الكود من خلال if .
فى النهاية قم بتجربة الكود و سترى النتيجة تظهر فى شاشة المخرجات التى نستخدمها لرؤية تأثير الكود على الفيلم و سوف نناقش الاسكريبت السابق فى نهاية الكتاب و سترى بنفسك انك بعد الدروس التالية ستكون قادرا على تحليل الكود و شرحة و تطويرة بنفسك.

الـفـصـل الـثـانـي

Data and Data Types البيانات و أنواعها

هذا الفصل سوف نتحدث فيه عن مبادئ لغة الاكشن اسكربت ٢ و سوف نتعلم فى هذا الفصل كيف نتعامل مع البيانات و ما هو نوع البيانات التى يمكننا التعامل معها من خلال هذة اللغة.
البيانات عموماً هى مجموعة من الحروف و الارقام و المعلومات و يمكنك توصيف او تحديد نوع البيانات من خلال المتغيرات و هذا يدفعنا للتسائل ***ما هو المتغير ؟!***

المتغيّرات Variables:

كما في أيّ لغة برمجة، تقوم المتغيّرات بتخزين القيم أثناء تنفيذ البرنامج.. وطبعًا سُمّيت متغيّراتٍ، لأنّك تستطيع تغيير قيمها في أيّ لحظةٍ أثناء تنفيذ البرنامج.
وللمتغيّر اسمٌ وقيمة.. *فمثلا:* المتغيّر "اسم المستخدم" UserName يمكن أن تُوضع به القيمة "مايكل".. والمتغيّر "الخصم" Discount يمكن أن تُوضع به القيمة ٠,٤٠.
تلاحُظ هنا أنّ القيمتين "مايكل" و ٠,٤٠ مختلفتان، فالأولى نص String لهذا تمّ وضعها بينَ علامتَيْ تنصيص، بينما الثانية قيمة رقميّة Numeric Value.
المتغيّرات في الاكشن اسكربت ٢ ليست مجرّد أسماءٍ أو مخازن للقيم.. إنّها كذلك كِيانات ذكيّة لتخزين وإجراء العمليات على القيم.. باختصار: إنّها كائنات Objects، لها وسائلها وخصائصها الخاصة بها.

أنواع البيانات Types of data

String النّصّيّة: تم شرحها فى المثال السابق.

Number الرقميّة:-تم شرحها فى المثال السابق .

Boolean المنطقية :
البيانات المنطقيّة تخزّن واحدةً فقط من القيمتين: "صواب" True و"خطأ" False، وهي في الأساس أعداد صحيحة، فالقيمة "صواب" تعادل ١، والقيمة "خطأ" تعادل صفرا.. وفي الواقع، أي قيمة غير صفرية، تعتبر True.

Object الكائنات:-
MovieClip مقطع فيلم لة خصائص و أحداث
Null قيمة فارغة
Undefined قيمة غير معرفة

تعريف المتغيرات Declaring Variables

لماذا نعرّف المتغيّر؟؟؟؟؟
في معظم لغاتِ البرمجة، يجبُ تعريفُ المتغيّراتِ أولا قبلَ استخدامها.. إنّ هذا يجعل الأمرَ أيسرَ بالنسبة لمترجم الكود Compiler، ففي كلّ مرّة يصادف المترجم متغيّرا، عليه أن يُنشئه في الذاكرة، ونتيجة لاعتباراتٍ في تنظيم الذاكرة، فإن مثل هذه العملية تستهلك بعض الوقت، ممّا يُبطئ البرنامج.. ولكن لو كان المترجم يعرف كل متغيرات البرنامج وأنواعها سلفا قبل أن يبدأ ترجمة البرنامج، ففي هذه الحالة سيتحسّن الأداء لأقصى درجة.

لقد كانت من أشهر سمات اكشن اسكربت (١)، عدم إرغامه للمبرمج على تعريف كلّ المتغيرات.. لقد صارت هذه السمة منتقدةً الآن بشدّةٍ، ليس فقط للأسباب المتعلّقة بسرعة الترجمة وكفاءة الأداء، ولكن أيضا لأن تعريف المتغيّر يُمكّن المترجم من اصطياد أخطاء كثيرةٍ، سواء في وقت التصميم Design Time أو وقت الترجمة Compile Time، بدلا من أن تُفاجئكَ في وقت التشغيل Runtime.

الشروط الواجب توافرها عند تعريف المتغيرات:
- ألا يكون كلمة من كلمات اللغة الأساسيّة (تلك التي تراها باللون الأزرق عند فتح لوحةactions).

و منها :

- break
- case
- class
- continue
- default
- delete
- dynamic
- else
- extends
- for
- function
- get
- if
- implements
- import
- in
- instanceof
- interface
- new
- private
- public
- return
- set
- static
- switch
- this
- typeof
- var
- void
- while
- with

- ألا يزيد عن ٢٥٥ حرفًا، وهو رقم كبير بالفعل بما يكفي.
- أن يتكون من كلمة واحدة لا تتخللها المسافات.. ويمكن استخدام الشرطة المنخفضة "_" للفصل بين مقاطع الكلمة بدلا من المسافات.
- لا يبدأ بأرقام، وإن كان من الممكن أن تتوسطه أرقام، أو ينتهيَ بها.
- لا يحتوي على أي من: علامات التنصيص أو الأقواس أو النقطة "."، ولا علامات العمليات الحسابية أو علامات المقارنة الحسابية أو المنطقية، فكل هذه العلامات محجوزة لوظائف أخرى.
- غير مسموح بتكرار اسم المتغيّر داخل نفس النطاق، فلا يمكن تعريف متغيّرين متماثلين في الاسم داخل نفس الإجراء، وإن كان من الممكن تكرار نفس اسم المتغيّر لكن في إجراءات مختلفة.

-والمتغيرات في لغة الاكشن اسكربت (٢) لا تتجاهل حالة الأحرف Case-insensitive، فالأسماء myAge و myage و MYAGE، ليست كلّها متكافئة معنى هذا أنّك تستطيع استخدام هذه الكلمات لتعريف ثلاثة متغيرات مختلفة، فكلّها لا تُعتبر اسما واحدا.

طريقة تعريف المتغير:-

Var نوع المتغير :اسم المتغير

كلمة Var هى المسئولة عن حجز مكان فى الذاكرة لمتغير ثم نقوم بأعطاء المتغير اسم و نوع بعد ذلك مثال:

```
var sProduct:String;
```

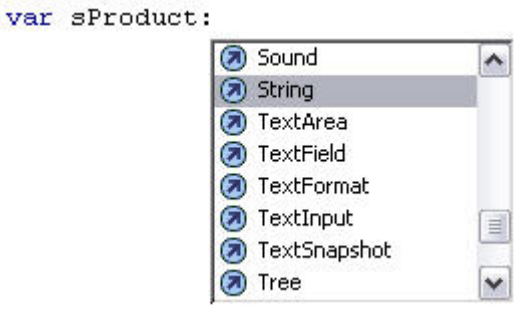
بعد تعريف المتغير يمكنك اسناد قيمة ما لهذا المتغير شرط ان تكون من النوع الذى عرفتة للمتغير

```
sProduct = "car";
```

و يمكن اختصار الجملتين فى جملة واحدة هكذا

```
var sProduct:String = "car";
```

لاحظ انه عندما تكتب var ثم اسم المتغير ثم : فيظهر لك نافذة بها أنواع البيانات التى قد تحتاجها



اذا كنت تريد عرض قيمة المتغير sProduct فى وضع الاختبار يمكنك استخدام الدالة trace() فسوف تقوم بارسال القيمة الى شاشة المخرجات Output panel فيصبح الكود كالتالى :

```
var sProduct:String = "car";

trace(sProduct);
```

الناتج

```
// car
```

لو قمت بتغيير قيمة المتغير sProduct الى ١٢٠ مثلا :-

```
sProduct = 120
```

سوف ترى الخطأ التالى :-

```
Type mismatch in assignment statement: found Number where String is

  required.
```

هذا الخطأ يخبرك انك قمت باسناد بيانات خطأ لمتغير معين . لاحظ ان علامة التنصيص (" ") the quotation marks ليست ضرورية فى حالة المتغيرات الرقمية

```
var numWrinkles:Number = 55;
```

و اذا كنت تريد تغير قيمة المتغير بعد ذلك يمكنك كتابة الكود التالى :-

```
numWrinkles = 60;
```

اذا كانت القيمة رقمية او منطقية (صح او خطأ) لا تستخدم (quotation marks). مثال :-

```
var age:Number = 38;

var married:Boolean = true;

var hasChildren:Boolean = false;
```

ما هى المصفوفة array :-

البرمجة الحقيقيّة تبدأ من هذه النقطة، فلقد صُنع الكمبيوتر أساسا، ليقوم بالعمليات الرتيبة المتكرّرة لآلاف أو ملايين المرّات، بسرعةٍ وبدون ملل.
افترض مثلا أنّك تريد حساب متوسط العمر لخمسين طالبا.. أوّل ما ستفكّر فيه، هو أن تعرّف خمسين متغيّرا وتجمعها معا وتقسم الناتج على ٥٠.. إنّ مثل هذه الطريقة ستستهلك منك شهرا على الأقلّ لكتابتها، وهي كفيلة بجعلك تكره البرمجة اساسا!
مع أنّك تستطيع أن تكتب هذا البرنامج في خمس سطور لا غير.. تخيّل!
والفكرة كلّها تعتمد على تخزين أعمار الطلبة في "تركيب ما"، يمكن للكمبيوتر أن يتعامل معه بطريقة آليّة رتيبة متكرّرة، لينفّذ عليه العمليّات التي نريدها.
هذا التركيب هو المصفوفة **Array**، التي تتكوّن من مجموعة من الخانات، كل خانة منها تحتفظ بقيمة معيّنة، بحيث يمكن الوصول لهذه القيمة عن طريق رقم خانتها **Index**.
مثال :

```
var childrenArr:Array = new Array("Pylon", "Smithers", "Gil");
```

لاحظ أن يمكنك كتابة الكود السابق بطريقة مختصرة كما يلى :-

```
var childrenArr:Array = ["Pylon", "Smithers", "Gil"];
```

و هناك طريقة اخرى لتعريف مصفوفة تحتوى اسماء الشهور :

```
var myArr:Array = new Array();

myArr[0] = "January";
myArr[1] = "February";
myArr[2] = "March";
myArr[3] = "April";
```

بنفس الطريقة يمكننا تعريف كائن جديد (object) مثال :

```
var myObj:Object = new Object();
myObj.firstName = "Steve";
myObj.age = 50;
myObj.childrenArr = new Array("Mike", "Robbie", "Chip");
// طريقة اخرى
var myObj:Object = {firstName:"Steve", age:50, childrenArr:["Mike",
  "Robbie", "Chip"]};
```

ملحوظة : يمكنك استخدام العلامة // فى نافذة محرر الكود لكى نوضح لبرنامج محرر الكود فى فلاش ان الكلمات التى تلى هذة العلامة ليست اكواد و لكنها تعليقات لتذكرة المبرمج بسبب كتابتة للكود اى هوامش لا يتم تنفيذها .

الـفـصـل الـثـالـث

ا لـــمـــعـــامـــلات Operators

انك تريد من برنامجك ان يفعل اشياء معينه عند تحقق شروط معينه ولكن كيف نصيغ هذه الاوامر هذا هو المهم انك ترى الشروط دائما وربما بدون ان تدري فأول شئ فعلته اليوم عند اتصالك بالانترنت هو ادخال كلمة السر – أليس كذلك – نعم هو كذلك ولا شئ غيره – ان برنامج الاتصال قد وضع شرط صحة كلمة السر لكي يمكنك من الدخول وربما اشياء اخرى مثل عدم انتهاء مدة الاشتراك وصحة اسم المستخدم وهكذا كلها شروط انها ايضا موضوعنا هذا اليوم ومعظم الشروط المستخدمه لقيمتين هي هل هما متساويتان ام احدهما اكبر او اصغر بالاضافة الى المعاملات البولينية Boolean operators

ومن درس منكم علم الجبر سوف يتذكرها بسرعة ام من نسى او لم يدرسها اصلا فالموضوع بسيط جدا لانك تمارسه في حياتك اليومية ربما دون ان تدري وهي

AND , OR , NOT

كمثال من الحياة اليومية :-

☞ **Ask Michael AND Nabil**

يجب ان تسأل مايكل و نبيل وليس واحد منهم فقط بل يجب ان يكون السؤال للاثنين

☞ **Ask Michae OR Nabil**

اي يجب ان تسأل مايكل او نبيل يكفي فقط واحد منهم

☞ **Ask Michael NOT Nabil**

اي يجب ان تسأل مايكل وليس نبيل فلو سألت نبيل فان الشرط غير متحقق

باختصار المعاملات تعتبر ضرورة بالنسبة لصياغة الشروط .

الجدول التالي يعبر عن طريقة صياغة المعاملات البولينية في الاكشن اسكربت :-

Operator	Description
+	الجمع
-	الطرح
!	Not ليس
++	لزيادة بواحدة و إسناد الناتج إلى المتغير
--	نقصان بواحدة و إسناد الناتج إلى المتغير

()	Grouping or function call
[]	Array element
.	Member access
++	Pre-increment
--	Pre-decrement
new	Allocation
delete	Deallocation
typeof	Type of object
void	Undefined type
*	الضرب
/	قسمة
%	باقى القسمة
<<	Bitwise shift left
>>	Bitwise shift right
>>>	Bitwise shift right (unsigned)
<	اقل من
<=	اقل من او يساوى
>	اكبر من
>=	اكبر من او يساوى
==	تساوى
!=	لا يساوى
&	Bitwise AND
^	Bitwise XOR
|	Bitwise OR
&&	Logical AND
||	أو
?:	Conditional
=	Assignment
*=, /=, %=, +=, -=, &=, |=, ^=, <<=, >>=, >>>=	Compound assignments

تذكر جيدا خطأ من السهل ان تقع فيه ان الشروط غير التخصيص بمعنى انك عندما تخصص قيمة لمتغير فإنك لابد ان تستخدم علامة يساوي واحدة فقط مثل :

A=5;

B=A;

C=18;

اما في صياغة شرط فإنك لابد ان تضع علامتين اذا اشترطت التساوي مثل :-

if (A==B)

if (A==18)

☝ ارجو ان تنتبه لذلك !!!!!!!!!!!!!!

يجب ان تعرف كيف تكتب المعادلات لانجاز الحسابات

هذا الموضوع بالذات تتفق فيه جميع لغات الكومبيوتر بلا استثناء وهناك بعض القواعد البسيطة التي يجب ان تعرفها مثل :-

١- لابد ان تحتوي المعادلة في طرفها الايسر على متغير واحد فقط واي عدد في الطرف الايمن

مثال :-

A = B + C * 5 –3

٢- جميع القيم الموجودة في الطرف الايمن تكون معلومة القيمة اي تم حسابها مثلا في معادلة سابقة

مثال :-

B = 5

C = 10

A = B + C * 5 –3

فقد تم تخصيص قيمة للمتغير B و للمتغير C

ثم دخل البرنامج الى المعادلة الاخيرة لحساب قيمة A وقد علم تماما قيم الطرف الايسر

اي قيم B و C

فاذا كتبت مايلي سيكون خطأ قاتل !!!!!!!!!!

B = 5

A = B + C * 5 –3

C = 10

لان البرنامج عرف قيمة المتغير B ثم دخل الى المعادله ليحسب A وهو لا يعرف قيمة المتغير C حيث سيعلمها فيما بعد فرتب خطواتك لتكون منطقية .....

· كل من درس مبادئ الرياضيات يعلم ان القسمة على الصفر خطأ فادح واذا حدثت سيتعطل البرنامج ويعطيك نظام التشغيل الرسالة المعروفة (هذا البرنامج قام بعملية ممنوعة)

مثال

B = 7

C = 7

D = B-C

A = B/D

قام البرنامج بالقسمة على صفر حيث ان قيمة B = ٧ وC = ٧ وD = B – C اي ٧-٧ وتساوي بذلك صفر ثم تأتي مرحلة انهيار البرنامج عندما يقسم بي على دي وقيمتها صفر بعد الحسابات السابقة (احذر ذلك)

كل ماهو خطأ رياضي سيكون خطأ في البرنامج مثل حساب الجذر التربيعي لعدد اقل من الصفر وهكذا

- حافظ دائما على ان يكون عدد الاقواس المفتوحة يساوي عدد الاقواس المغلقه

اقصد عدد الاقواس ذات الاتجاه اليمين = عدد الاقواس ذات الاتجاه اليسار

لان الكومبيوتر عندما تفتح له قوس ايسر معناها بدء عملية حسابية منفصلة حتى لوكانت داخل معادلة واحدة

فأذا لم تغلق بالقوس الايمن فمعنى ذلك عملية لم تغلق وتنهار المعادلة وبالتالي البرنامج.

مثال لاحظ فيه ان عدد الاقواس متساوي

A = (A+B) * (C – D) – (Q * M)

عملية باقي القسمة :-

نحن نعرف العمليات الحسابية الأربعة الجمع و الضرب و القسمة و قد اعتدنا عليها و لكن في بعض لغات البرمجة هناك عملية خامسة تضاف عليها و هي باقي القسمة حيث تضع بين أي عددين و تدل على باقي قسمة الأول على الثاني مثل :-

```
z = x % y ;
```

عملية الزيادة بواحد أو النقصان بواحد(++ ، --) :-

و تتلخص مهمة كلا من هذه العمليات على زيادة واحد على قيمة أي متغير و ثم تسند الناتج له أي إذا كان هناك متغير يساوي ٥ ووضعنا هذه العملية قبله أو بعده فسوف يصبح ٦ و لاحظ هنا أني قلت قبله أو بعد حيث هناك فرق بين الحالتين .

اين المثال ؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟

من الواضح أن المتغير y سوف يزيد بواحد أي تصبح قيمته ٦ و من ثم تسند هذه القيمة إلى المتغير z

لكن أنظر إلى الحالة التالية و لاحظ الفرق في المثال التالي :-

```
y = 5;

z = y++ ;
```

أما في هذه الحالة فسوف تسند قيمة y الأصلية إلى z و هي القيمة ٥ و من ثم تزاد قيمة y بواحدة .

Pre-Increment and Post-Increment (++)

قمت بتعريف متغير اعطيتة القيمة ١٠ و هذا السطر لن يتم تفيذة من البرنامج لانه تعليق //

```
myNumber = 10;
trace(++myNumber);
trace(myNumber++);
trace(myNumber);
```

الناتج

١١
١١
١٢

Pre-Decrement and Post-Decrement (--)

```
myNumber = 10;
result = --myNumber * 2;
trace(result);
myNumber = 10;
result = myNumber-- * 2;
trace(result);
```

الناتج

١٨
٢٠

مثال على عمل دالة لجمع رقمين :-

قم بعمل ٣ مربع نص و ليكن اسمهم input1 و input2 و total و زر اسمة btAdd
ثم ضع الكود التالى فى الفريم الاول

```
btAdd.onRelease = function(){
    var theTotal:Number = Number(input1.text)+Number(input2.text);
    total.text = String(theTotal);
}
```

الـــفـــصـــل الـــرابـــع

جـمـل الـتـحـكـّم فـي الـمـسـارFlow- control statement

إنّ البرمجة أعمق من أن تكون مجرّد تعريف متغيّرات.. إنّها تفكير منطقيّ يعتمد على حساب كل الاحتمالات، لاتخاذ الأفعال المناسبة لكل احتمال.. لهذا فلا بد أن توجد طرق نتحكّم بها فيما ينفّذ ومتى ينفّذ من البرنامج.

جملة الشرط ...If:

تستطيع أن تختبر حدوث شرط معيّن، فإذا كان صحيحا يتمّ تنفيذ مقطع الشرط، وإن كان خاطئا يقفز التنفيذ إلى جملة نهاية الشرط .

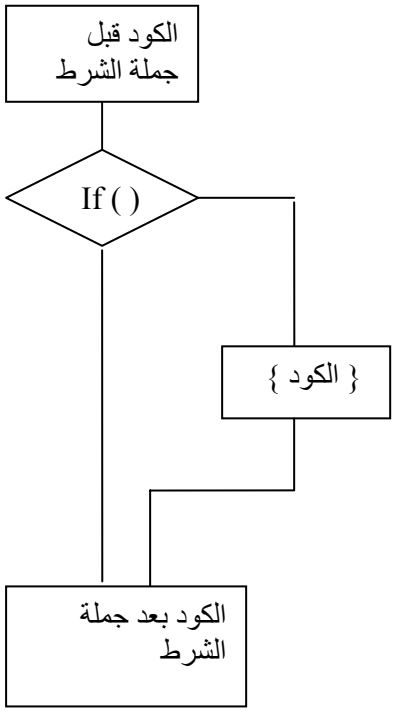
ولكن ما هو الشرط؟؟؟؟؟؟؟؟؟؟؟؟؟

الشرط هو افتراض معين يتوقف عليه عمليات اخرى فمثلاً تريد ان تضع شرط الا يدخل رقم موظف اكبر من الف لان عدد موظفين الشركة لايزيدون عن الف وبالتالي اذا ادخل من يعمل على البرنامج رقم موظف اكبر من الف يقوم البرنامج باصدار رسالة تفيد بذلك وهكذا لها حالات كثيرة حسب فكرة البرنامج

المهم انه تعبير يعطي نتيجة منطقيّة (True أو False)، مثل:

```
myNumber = 10;
if(myNumber < 20){
   trace("myNumber is less than 20");
}
```

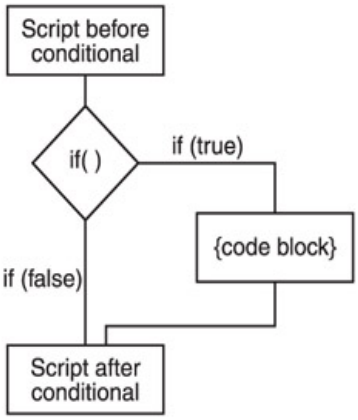
شرح المثال : لقد قمنا بتعريف متغير myNumber و اسندنا له القيمة ١٠ و قمنا بأختبار قيمة هذا المتغير اذا كانت اقل من ٢٠ فسوف نقوم بتنفيذ الكود الموجود بين القوسين { } و هو جملة trace اما اذا كانت قيمة المتغير اكبر من ٢٠ فلن يحدث شيء من الكود الموجود بين الاقواس { }



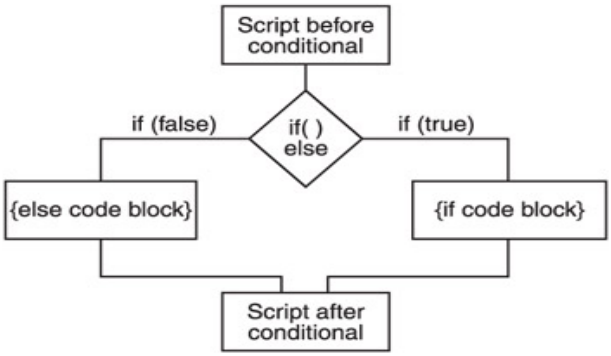
جملة الشرط ...else If:

مثال :

```
myNumber = 10;
if(myNumber > 20){
   trace("myNumber is greater than 20);
}
Else{
   trace("myNumber is less than or equal to 20);
}
```



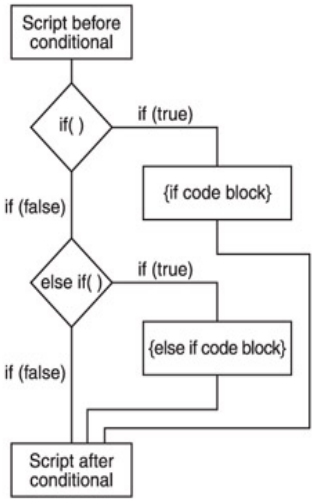
شرح المثال : لقد قمنا بتعريف متغير myNumber و اسندنا له القيمة ١٠ و قمنا بأختبار قيمة هذا المتغير اذا كانت اكبر من ٢٠ فسوف نقوم بتنفيذ الكود الموجود بين القوسين { } و هو جملة trace اما اذاقمنا بتغيير قيمة المتغيرلتصبح ٥٠ فسيتم تنفيذ الكود الموجود بعد كلمة else



جملة الشرط if...Else :-

يمكن دمج جملتى if من خلال استخدام التعبير else if

```
myNumber = 10;
if(myNumber < 20){
trace("myNumber is less than 20");
}

else if(myNumber < 50){
trace("myNumber is less than 50 but greater than or equal to 20");
}
```



استخدام if مع معاملات اخرى مثل and او or:-

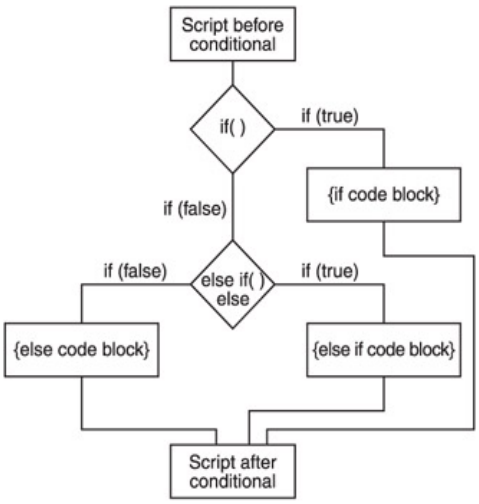
```
on (press) {
   if ((a == 7) and (b == 15)) {
    gotoAndPlay(20);
}
}
```

مثال اخر:-

```
on (press) {
   if ((a == 7) or (b == 15)) {
    gotoAndPlay(20);
}
}
```

كما يمكن دمج اكثر من تعبير فى جملة if كما يلى :-

```
myNumber = 10;
if(myNumber < 20){
   trace("myNumber is less than 20");
}
else if(myNumber < 50){
   trace("myNumber is less than 50 but greater than or equal to 20");
}
else{
   trace("myNumber is greater than or equal to 50");
}
```



الدالة الشرطية جملة switch .... Case:-

عندما يكون لدينا عدة خيارات و نكون نريد أن نخرج بواحد منهم و هو الذى نريده من بين الخيارات والذى سوف نخرج به سوف يحدده المتغير الذى سوف ندخله و الذى سوف يتفق مع واحدة من هذه الخيارات و يحققه ..

```
switch (المتغير){
   case الاحتمال الاول:
      المطلوب لهذا الاحتمال
      case الاحتمال الثانى:
      المطلوب لهذا الاحتمال
فى حالة عدم تطابق اى حالة يتم تنفيذ ال
   ...
   [default]
}
```

The switch Statement

```
switch (expression){
   case caseClause1:
      code block
   case caseClause2:
      code block
   ...
   [default]
}
```

مثال:-

فى المثال التالى عرفنا متغيرx و قيمته ١٠ وهناك حالتين لهما نفس قيمة شرط الحالة و هو ١٠ لذلك سوف تجد الناتج دائما هو الحالة الاولى case 1 و لن يطبع ابدا الحالة الثانية.

```
x = 10;
switch( x ){
   case 10:
      trace("case 1");
      break;
   case 10:
      trace("case 2");
      break;
}
```

مثال اخر:-

```
exp = "hello";
switch( exp ){
   case "hello":
      trace("case 1");
   case "hi":
      trace("case 2");
      break;
}
```

الناتج

case 1

مثال أخر يوضح التشابة بين فكرة استخدام if للتحكم فى البرنامج و استخدام switch :-

```
switch( exp ){
   case 1:
      //do task a
   case 2:
      //do task b
   case 3:
      //do task c
}
```

يمكن عمل نفس الكود السابق باستخدام if هكذا :-

```
if( x == 3 ){
   //do task c
}
else if( x == 2 ){
   //do task b
   //do task c
}
else if( x == 1 ){
   //do task a
   //do task b
   //do task c
}
```

مثال اخير يوضح جملة switch :-

```
switch( user_command_string ){
   case "move north":
   case "go north":
   case "north":
   case "n":
      trace("you have moved north");
      break;
   case "move south":
   case "go south":
   case "south":
   case "s":
      trace("you have moved south");
      break;
   default:
      trace("I'm sorry, I don't understand "+ user_command_string);
}
```

الخلاصة : اننا ندخل المتغير فى جملة (switch) للمقارنة مع الخيارات الموجودة بداخلها و عند مطابقة المتغير مع احدى الخيارات تصبح النتيجة (true) وسوف تنفذ الجملة المتعلقة بهذه المطابقة مع العلم ان واحد فقط من هذه الخيارات يعطى (true) و البقية (false) و عندما تكون كل الخيارات ليست مطابقة سوف ينفذ ما بداخل الـ(default) .

الـفـصـل الـخـامـس

تـراكـيـب الـتـكـرار Loop Structures

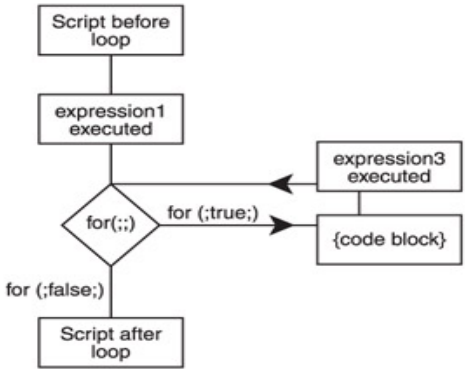
قدرة الكمبيوتر على تكرار أي جزء من الكود ـ خاصّة مع سرعته الفائقة ـ هي ما تجعله مريحا جدا للبشر، ليحمل عنهم عناء الرتابة والبطء والملل

جملة التكرار "من إلى" ...For:

الدوران او التكرار او عمل looping من الاوامر الاساسية في جميع لغات البرمجة فمثلا اذا كان البرنامج سيدخل اسماء الف موظف هل تعتقد انك ستكتب الف امر لادخال هذه الاسماء بالطبع ستكون حماقه ولكن لو وضعنا امر واحد فقط لادخال اسم الموظف وطلبنا من البرنامج الدوران الف مرة حول هذا الامر بالطبع سيكون شئ جميل ان يدخل الف بيان بمجهود بسيط نتيجة تسهيل اعطته لغة البرمجة

مثال

```
for(i = 0;i < 1000; i++){
   trace(i);
}
```



ياله من شئ جميل فعلاً طلبنا من الكومبيوتر انشاء عداد يعد من صفر الى١٠ ورمزنا لها بالرمز i ثم داخل العداد طلبنا منه ادخال الموظف رقم i وهو عداد متغير حتى يكتمل العداد بوصوله للالف ويكون قد تم تنفيذ الامر معه الف مرة بأدنى مجهود وكلما تغير العد من ١ الى ٢ الى ١٠٠٠ تغير معه رقم الموظف بنفس الطريقة

**** الجدول التالى يوضح صيغ التكرار المختلفة الخاصة بـFor وتفسيرها.....**

الـصـيـغـة	الـتـفـسـيـر
for	امر اللغة لعملية التكرار
(	قوس مفتوح يوضح بداخله بارامترات الامر
i=0;	المتغير = رقم بداية الحلقات
i<1000;	شرط نهاية الحلقات

i++	المتغير يزيد بمقدار واحد مع بداية كل حلقة – لاتضع بعده فاصلة منقوطة
)	قوس نهاية بارامترات الامر – لاتضع بعده فاصلة منقوطة
{	قوس بداية بلوك الاوامر المطلوب تكرارها
بداخل اقواس البلوك توضع الاوامر المطلوب تكرارها	
}	قوس نهاية بلوك الاوامر المطلوب تكرارها

بداخل بلوك الاوامر تم تنفيذ الامر

وهو امر يقوم بطبع قيمة x التي تتغير في كل مرة ابتداء من صفر حسب ماذكرت ان

i=0 وتزيد في كل مرة بمقدار ١ حسبما ذكرت ان ++i وذلك حتى يصل الى ٩٩ حسبما ذكرت ان i<100

وبالتالي ستكون مخرجات البرنامج كما يلي

0

1

2

3

4

.

.

.

99

لنستفيد اكثر من قوة الحلقات التكرارية وايضا نرى امكانات اخرى لها بوضع حلقة داخل حلقة كما يلى في البرنامج الذي يطبع جدول الضرب من جدول واحد حتى جدول ١٢

```
on(press){

for(var x:Number=1;x<13;x++) {

 for(var y:Number=1;y<13;y++) {

  var z:Number = x * y

       trace(x+"*"+y+"="+z)

 }}}
```

اعلنا عن ثلاث متغيرات لاعداد صحيحة الاول للحلقة الاولى والثاني للحلقة الثانية والثالث لاحتواء حاصل الضرب وهو الهدف من البرنامج.

بنفس ماسبق شرحه وضعنا حلقة تكرارية تبدء من رقم واحد وتنتهي برقم ١٢ وتزيد بمقدار واحد واوامر هذه الحلقة مكتوبة للتوضيح بخلفية صفراء حيث بدأت الحلقة بالصيغة السابق شرحها ثم قوس بداية البلوك وقوس نهاية البلوك حيث يوضع مابين القوسين سلسلة اوامر لتنفيذها كما سبق واتفقنا عليه

```
for(var x:Number =1;x<13;x++)
 {
اوامر مطلوب تنفيذها
{
```

وبداخل اقواس البلوك للحلقة الاولى مطلوب وضع اوامر للتنفيذ فكانت اوامر التنفيذ داخل البلوك عبارة عن حلقة اخرى ، وما المانع فالحلقات وغيرها هي نفسها اوامر والحلقه الاخرى كانت عبارة عن عداد يعد من الرقم ١ الى الرقم ١٢ وهي ايضا تحتاج الى صيغة معينة واقواس بلوك للتنفيذ وهي موضوعة في البرنامج بخلفية خضراء كما يلي :-

```
for(var y:Number=1;y<13;y++)  {
        var z:Number = x * y;

trace(x+"*"+y+"="+z)

}
```

وبداخل بلوك الاوامر للحلقة الثانية وضعنا عدة اوامر لتفى بالغرض الذي من اجله تم عمل البرنامج وهو جدول الضرب – عبارة عن معادلة لحساب حاصل ضرب المتغيرين x و y

ويوضع الناتج في المتغير z كما يلي :-

```
z = x * y;
```

يبدء البرنامج بالدخول الى الحلقه الاولى ويخصص للمتغير X القيمة واحد حيث انها قيمة عداد البداية ثم يدخل الى داخل اقواس بلوكه لتنفيذ ما بداخله فيجد حلقه اخرى فيبدء بتنفيذها وطلبت الحلقة ان يكون متغيرها Y يبدء بالقيمة واحد وينتهي بالقيمة ١٢ ثم تدخل الحلقة الثانية الى بلوك اوامرها فتجد معادلة يضرب فيها قيمة x التي هي واحد الآن وتتغير قيمة y فيها ثم طبع النتائج وتظل الحلقة الداخلية تنفذ بتغير Y مع ثبات X بالطبع وتكون النتائج كما يلى :-

1 * 1 = 1

1 * 2 = 2

1 * 3 = 3

.

.

.

1 * 12 = 12

وعند وصول نهاية عداد الحلقة الداخلية الى ١٢ وهو نهاية الحلقة ينتهي تنفيذ الحلقة فيستمر البرنامج فيجد امامه قوس نهاية الحلقه الخارجية ذو الخلفية الصفراء فيعود الى الحلقة الخارجيه حيث تزيد قيمة X بمقدار واحد وتصبح قيمتها = ٢ فيدخل الى بلوك اوامره كما سبق فيجد حلقة اخرى تبدء من واحد وتنتهي بـ ١٢ وبداخلها معادلة وامر طبع فيكون التنفيذ كما يلي :-

2 * 1 = 2

2 * 2 = 4

2 * 3 = 6

.

.

.

2 * 12 = 24

و هكذا يستمر التنفيذ الى ان تصل الحلقة الخارجية الى نهايتها بالرقم ١٢ واثناء ذلك تنفذ الحلقة الداخلية من بدايتها الى نهايتها اي من ١ الى ١٢ وتطبق المعادلة وامر الطبع ليكون اخر تنفيذ كما يلي :-

12 * 1 = 12

12 * 2 = 24

12 * 3 = 36

.

.

.

12 * 12 = 144

مثال اخر لعمل ١٠ نسخ من نفس الـ movieClip لاحظ انه تم تسميتهmyClip_mc

```
for (var i:Number=0; i<=10 ; ++i) {
  myClip_mc.duplicateMovieClip("myClip_mc" + i, i);
}
```

جملة التكرار The while Loop

فى هذه الطريقة يستمر بتنفيذ ما بداخل جملة التكرار ما دام الشرط متحقق فى كل مرة تريد فيها الدخول سوف يتحقق من الشرط اولا فاذا تحقق تقوم بالدخول الى داخل الجملة و تنفذ ما بداخلها الى ان يفشل و يخرج من جملة التكرار و لكن اذا كان هناك اوامر خارج جملة التكرار اى بعد جملة while سوف ينفذها .
لكن لاحظ ان جملة (while) يجب ان تحتوى على ما يلى:
١- متغير نضعه بالشرط لكى نتحقق من صحة الشرط .

٢- يجب وضع قيمة ابتدائية لهذا المتغير قبل جملة الـ(while) .
٣- يجب ان نذكر هذا المتغير و مقدار زيادته بداخل جملة الشرط سواءا قبل تنفيذ الجملة التى بداخل جملة التكرار او بعدها .

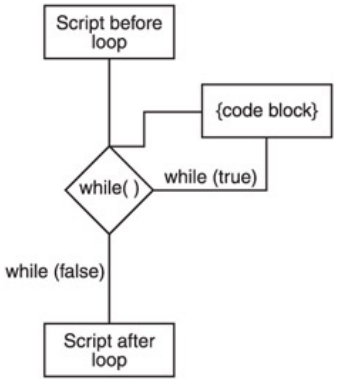
مثال :-

```
x = 0;
while( x < 100 ){
   trace(myNumber);
   x++;
}
```

** الجدول التالى يوضح صيغ عمليات التكرار المختلفة وتفسيرها .....

الصيغة	التفسير
while	امر اللغة لعملية التكرار
)	قوس مفتوح يوضح بداخله بارامترات الامر
X<100	المتغير مع شرط لنهاية الحلقة
)	قوس نهاية بارامترات الامر – لاتضع بعده فاصلة منقوطة
{	قوس بداية بلوك الاوامر المطلوب تكرارها
بداخل اقواس البلوك توضع الاوامر المطلوب تكرارها	
x++;	ولا تنسى عداد الزياده او النقصان ليتحقق الشرط لانهاء تنفيذ الحلقة
}	قوس نهاية بلوك الاوامر المطلوب تكرارها

وبعد الاعلان عن المتغير X خلاصة شرح الامر السابق (أعد تنفيذ ما بداخل البلوك طالما X او المتغير اقل من ١٠٠ ثم اقواس بلوك تضع ماشئت بداخله من اوامر وقوس نهاية البلوك ويزيد معنا فقط عداد للمتغير ليزيده بالمقدار الذي تريده ويوضع في اي مكان داخل الحلقة او حسب افكارك عن البرنامج المهم لاتنساه والا سوف يدور البرنامج داخل الحلقة الى الابد حيث ان شرط نهايتها ان تزيد X عن ١٠٠ وطالما لم تضع عداد زياده للمتغير فلن يتحقق الشرط وبالتالى لن تنتهي الحلقه الى الابد ويظل يعمل الكومبيوتر بلا نهاية للبرنامج ويميز العداد بالخلفية الصفراء واليك جدول صياغة الامر الذي تعودنا عليه.



حلقة التكرار The do while Loop

فى هذه الحالة يستمر تنفيذ ما بداخل الحلقة ما دام الشرط متحقق وهنا سوف يدخل الى داخل الحلقة و من ثم ينفذ الامر الذى بداخلها و بعد تنفيذها ينتقل ليتحقق من الشرط فاذا تحقق يعود مرة اخرى و اذا لم يتحقق يخرج من حلقة التكرار و لن يعود لها .

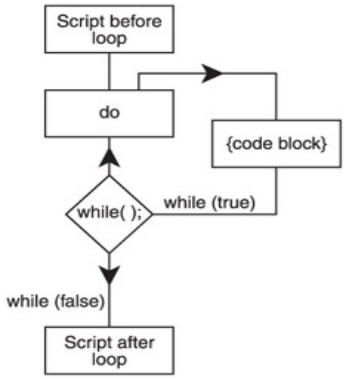
لكن لاحظ ان جملة (do while) يجب ان تحتوى على ما يلى:

١- متغير نضعه بالشرط لكى نتحقق من صحة الشرط .

٢- يجب وضع قيمة ابتدائية لهذا المتغير قبل جملة الـ(do while) .

٣- يجب ان نذكر هذا المتغير و مقدار زيادته بداخل حلقة الشرط سواءا قبل تنفيذ الجملة التى بداخل حلقة التكرار او بعدها .

```
myNumber = 99;
do{
   trace(myNumber);
}while(myNumber++ < 10);
```



قد نتسائل هنا ما الفرق بين while و جملة do while ؟؟؟؟؟

فى (while) نتحقق من الشرط قبل الدخول الى الحلقة اى اننا لا ننفذ اى شئ بداخلها ما دام الشرط لم يتحقق و هذا امر طبيعى لاننا لم ندخل الى الحلقة اصلا فكيف نعرف ما بداخلها و ننفذه أما فى (do while) كنا ندخل الى الحلقة و ننفذ امر ثم نفحص الشرط و لكن بعد ان نكون قد نفذنا هذا الامر و يجب التنبيه هنا فى حالة عدم تحقق الشرط لن نعود مرة اخرى الى الـ(do) اذن الفرق هو ان بالـ (do while) ينفذ على الاقل امر واحد فى داخل حلقة التكرار حتى لو كان الشرط غير متحقق على العكس الـ(while) الذى لا ينفذ اى امر مادام الشرط غير متحقق .

Nested Loops تداخل التكرار

```
var i:Number = 0;
while (++i <= 10) {
  var j:Number = 0;
  while (++j <= 10) {
    // perform these actions
  }
}
```

الان اعتقد ان من الممكن دمج بعض الجمل و التعبيرات لعمل مثال نستخدم فية المصفوفات و التكرار ...

```
// عرف مصفوفة جديدة
var myArr:Array = new Array();
// value1 قمنا بأسناد قيم لعناصر المصفوفة الاول قيمتة
myArr[1] = "value1";
myArr[0] = "value0";
// للمرور على العناصر الموجودة داخل المصفوفة و معرفة قيمتها سوف استخدم التكرار
var i:String;
for (i in myArr) {
```

```
  // طباعة القيم
  trace("key: " + i + ", value: " + myArr[i]);
}
```

الناتج

```
key: 0, value: value0
key: 1, value: value1
```

الـفـصـل الـســادس

functions and methods

الـدوال و الـخـصـائـص

الدوال **functions** هي شبه برنامج صغير اذا كان عندك سلسله عمليات متشابهه سوف تجريها داخل البرنامج كثيرا وتكررها كثيرا فلا داعي لاعادة كتابتها كل مرة بل يكفيك ان تكتبها مرة واحدة وتطلق عليها اسم وكلما ذكرت هذا الاسم داخل البرنامج تتم تلك العمليات ويصبح هذا الاسم كأنه من اوامر اللغة ، ايضا من فوائدها :-

- تقسيم البرنامج الى اجزاء صغيرة تستدعى وقت اللزوم .
- تقسيم البرنامج الى اجزاء يمكن اختبارها منفصلة لسرعة تحديد الخلل بالبرنامج .
- تقسيم العمل بين المبرمجين عند العمل في مشروع جماعي لانتاج برنامج كبير .
- تبادل الخبرات بين مطوري البرامج بنشر اجزاء يستخدمها الآخرون في برامجهم .

مثال اخر :-

لنفرض ان هناك اوامر لرسم جدول بأبعاد ومساحة معينة وتحتاج الى رسمة كثيرا فليس معنى هذا انك كل مرة تريد رسم الجدول ستعيد كتابة اوامره الكثيرة في كل مكان داخل البرنامج فقط جمع اوامر رسمه في دالة واعطية اسم وكلما ذكرت الاسم يتم رسم الجدول و نقوم بأعطاء هذة الدالة معطيات خاصة بها عند استدعائها انه شئ رائع اليس كذلك انا اظن ذلك.

الشكل الاساسى لكتابة الدالة هو:-

```
function myFunction (parameter1:DataType,parameter2:DataType,etc.) {
  //actions go here;
}
```

حيث **myFunction** اسم الدالة
parameter1 المعطى الاول للدالة و نوعة
و عندما نريد استدعاء هذه الدالة نقوم بكتابة الكود التالى :

```
myFunction(parameter1, parameter2);
```

لتطبيق ذلك دعنا نحضر البرنامج الذي تناولناه سابقا الخاص بجدول الضرب لنعطيه مزيد من القوة و التطوير باستخدام الدوال ،

ودعنا ننتهز فرصة الدوال لاعطاءمزيد من التحسينات على البرنامج حيث ان البرنامج ينتج فقط جدول الضرب من جدول ١ الى جدول ١٢ فلماذا لانجعلها دالة عامة تنتج مايحلو لنا من جداول ضرب من المؤكد انك تتفق معي في ذلك وبذلك نستفيد من قوة الدوال وتصبح دالة عامة غير موجهه لمهمة ثابته فنحن نريد ان نحدد نحن بداية الجداول ونهايتها فتارة تطبع من جدول ٧ الى جدول ٢٠ وتارة تطبع من جدول ١٧ الى ٤٠ وهكذا اليس هذا افضل سوف نمرر للدالة قيمة البداية وقيمة النهاية مع اجراء التعديلات اللازمة لذلك.

الصيغة العامة لكتابة الدوال :-

```
function اسم الدالة(المتغيرات التى سوف ترسل للدالة)
{
بداية الدالة العمليات التى تتم بداخل الدالة
Return النتيجة المعادة و;
}
```

```
on(press)
{
 function myFunction1(s1:Number,e1:Number)
 {for(var x:Number=s1;x<e1+1;x++)
{
 for(var y:Number=s1;y<e1+1;y++)
 {
  var z:Number = x * y
        trace(x+"*"+y+"="+z)
 }}}
//للأستدعاء الدالة نقوم بكتابة اسم الدالة و اعطائها قيمة المتغيرات بين القوسين
 myFunction1(1,5);
  }
```

لتسهيل الامر عليك وضعت لك لون اصفر على التعديلات التي اجريتها للوصول الى الهدف المنشود لقد وضعنا في اقواس **Prototype** اعلان عن تمرير متغيرين رقمين الى الدالة **(s1:Number,e1:Number)**
حيث **s1** هو الرقم الذى سيبدأ منه الجدول و المتغير **e1** هو الرقم الذى سينتهى فيه الجدول بعد ان ادخلت قيمة البداية والنهاية فكما ذكرنا تم تعديل الدالة لتمرير متغيرين لها يتغيران مع كل ادخال والآن لننادي الدالة

ممرين لها القيمتين الذين ادخلهما مستخدم البرنامج

```
 myFunction1(1,5);
```

مثال :دالة لحساب الوزن مقسوما على ٦,٠٤

```
function convertToMoonWeight (myWeight:Number){
 var weightOnMoon:Number = myWeight/6.04;
}
```

و لكى اقوم بأستدعاء الدالة السابقة نكتب التالى :-

```
convertToMoonWeight(165);
```

مثال اخر :-

```
function openWindow(url:String, window:String){
getURL(url, window);
}
```

لأستدعائها نكتب الكود التالى :-

```
openWindow("http://www.yahoo.com", "_blank");
```

مثال اخر :-

```
on (press) {
        function myFunction(num) {
   var newNum = num + 3;
   return newNum;
}
 trace(myFunction(7));
trace(myFunction(13));
```

```
trace(myFunction(2));
}
```

الناتج

```
10
16
5
```

* * *

الـفـصـل الـسـابـع

ا لـرسـم بـواسـطـة الاكـشـن اسـكـريـبـت

يمكننا الرسم باستخدام الدالة lineStyle() لكن هذة الدالة لها معطيات منها نوع الخط و درجة وضوحه و لونه

```
path.lineStyle(thickness, color, alpha)
```

مثال :-

```
_root.myClip_mc.lineStyle(10, 0x009900, 100);
```

للتحرك للمكان الذى سيبدأ منه الرسم نستخدم الدالة moveTo()

```
path.moveTo(x, y);
```

مثال :-

```
_root.myClip_mc.lineStyle(10,0x009900,100);
_root.myClip_mc.moveTo(100,100);
```

لنبدا الرسم من نقطة x الى نقطة y نستخدم الدالة lineTo()

مثال :-

```
_root.createEmptyMovieClip("canvas_mc",1);
_root.canvas_mc.lineStyle(2,0x009900,100);
_root.canvas_mc.moveTo(100,100);
_root.canvas_mc.lineTo(200,150);
```

مثال لعمل خط منحنى :-

```
this.createEmptyMovieClip("myMovieClip_mc", 1);
myMovieClip_mc.lineStyle(1, 0x000000, 100);
myMovieClip_mc.curveTo(0, 100, 100, 100);
```

مثال اخر لعمل مستطيل:-

```
_root.createEmptyMovieClip("rectangle_mc", 1);
rectangle_mc.lineStyle(1, 0x000000, 100);
// رسم الخطوط الاربعة المكونة للمستطيل
rectangle_mc.lineTo(100,  0);
rectangle_mc.lineTo(100, 50);
rectangle_mc.lineTo(  0, 50);
rectangle_mc.lineTo(  0,  0);
```

مثال لعمل دائرة:-

```
MovieClip.prototype.drawCircle = function (radius, x, y)
{
var angleDelta = Math.PI / 4;
  var ctrlDist = radius/Math.cos(angleDelta/2);
  var angle = 0;
  var rx, ry, ax, ay;
  this.moveTo(x + radius, y);
```

```
  for (var i = 0; i < 8; i++) {
    angle += angleDelta;
    rx = x + Math.cos(angle-(angleDelta/2))*(ctrlDist);
    ry = y + Math.sin(angle-(angleDelta/2))*(ctrlDist);
    ax = x + Math.cos(angle)*radius;
    ay = y + Math.sin(angle)*radius;
    this.curveTo(rx, ry, ax, ay);
  }
}
var ctrlDist = radius/Math.cos(angleDelta/2);
rx = x + Math.cos(angle-(angleDelta/2))*(ctrlDist);
ry = y + Math.sin(angle-(angleDelta/2))*(ctrlDist);
ax = x + Math.cos(angle)*radius;
ay = y + Math.sin(angle)*radius;
this.createEmptyMovieClip("circle_mc", 1);
circle_mc.lineStyle(1, 0x000000, 100);
circle_mc.drawCircle(100, 50, 75);
circle_mc.drawCircle(65);
```

مثال لعمل مربع و تلوينه باللون الاحمر :-

قمنا بعمل movieclip فارغ و قمنا بتسميته box_mc بالكود//

```
_root.createEmptyMovieClip("box_mc",1);
with (_root.box_mc) {
  lineStyle(0,0x000000,100);
  beginFill(0x990000,100);
  moveTo(0,0);
  lineTo(100,0);
  lineTo(100,100);
  lineTo(0,100);
  lineTo(0,0);
  endFill();
}
```

مثال اخر لعمل مستطيل به تدريج لونى :-

```
_root.createEmptyMovieClip("holder_mc", 1);
with (_root.holder_mc) {
  lineStyle(0, 0x000000, 0);
  rotation = 90 * (Math.PI/180);
  colors = [ 0x6666FF, 0xFF6600 ];
  alphas = [ 100, 100 ];
  ratios = [ 0, 255 ];
  matrix = {matrixType:"box", x:0, y:150, w:200, h:100, r:rotation };
  beginGradientFill( "linear", colors, alphas, ratios, matrix );
  moveTo(0,0);
  lineTo(550,0);
```

```
  lineTo(550,300);
  lineTo(0,300);
  lineTo(0,0);
  endFill();
}
```

ملحوظة :- بعد كتابة الكود قم بالضغط على مفتاحى **Ctrl + enter** لمشاهدة الشكل الذى قمت برسمه .

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

Using event handler methods استخدام مجيبات الأحداث

مجيب الحدث Event Handler هو إجراءٌ فرعيٌّ يتمّ استدعاؤه تلقائيًّا في كل مرة يحدث فيها الحدث.. فمثلاً في حالتنا هذه، الحدثُ هو ضغط الزر، لهذا في كل مرة يضغط فيها المستخدم هذا الزر، يتم استدعاء الإجراء onPress وما عليك الآن سوى كتابة الأوامر في هذا الإجراء، لتتفاعل مع حدث ضغط الزر.
ويتكون اسم الإجراء الذي يستجيب للحدث، من اسم الأداة واسم الحدث object.eventMethod .

الصيغة العامة:-

```
object.eventMethod = function () {
  // Your code here, responding to event.
}
```

مثال :

```
next_btn.onPress = function () {
  nextFrame();
}
```

صيغة اخرى :-

```
// Assign a function reference to button's onPress event handler.
next_btn.onPress = goNextFrame;
// Define goNextFrame() function.
function goNextFrame() {
  nextFrame();
}
```

لاحظ التالى!!

```
// خطا
next_btn.onPress = goNextFrame();
// صح
next_btn.onPress = goNextFrame;
```

انشـــاء مربع نـص بـالـكـود

لأنشاء مربع نص و تحديد نوعه بحيث يكون لادخال النصوص نستخدم الكود التالى :

```
this.createTextField("my_txt", 99, 10, 10, 200, 20);
my_txt.border = true;
my_txt.type = "input";
this.createTextField("myOther_txt", 100, 10, 50, 200, 20);
myOther_txt.border = true;
myOther_txt.type = "input";
myOther_txt.onSetFocus = function(my_txt:TextField) {
  my_txt.text = "I just lost keyboard focus";
        }
```

تحميل صورة داخل الفلاش بالاكشن اسكربت

نستخدم الكود التالى :-

```
loadMovie("myBitmap.jpg", "myClip_mc");
```

الـــفـــصـــل الـــتـــاســـع

ا لتحكم فى الصوت داخل الفيلم باستخدام الاكشن

نقوم بعمل كائن جديد للصوت بالطريقة التالية :-

```
var soundObjectName:Sound = new Sound (Target);
```

مثال :

اذا كان لدينا movieclip اسمه myMovieClip_mc و نريد التحكم فى الصوت الموجود بداخله نستخدم الكود التالى :

```
var mySound:Sound= new Sound ("myMovieClip_mc");
```

ثم بعد ذلك نتحكم فى مستوى الصوت من خلال الكود التالى :-

```
mySound.setVolume (50);
```

لتحميل ملف mp3 من خارج ملف الفلاش يجب عمل كائن للصوت

```
var myFavMP3:Sound = new Sound();
```

ثم تحميل الملف الصوت mySong.mp3

```
myFavMP3.loadSound("mySong.mp3", true);
```

لاحظ اننا استخدمنا المعطى true لكى يقوم ملف الصوت بالعمل بعد تحميل swf و لكن اذا كتبنا false فيصبح الكود كالتالى :

```
myFavMP3.loadSound("mySong.mp3", false);
```

لكن يجب ان نستخدم دالة start لكى يبدأ الصوت فى العمل

```
myFavMP3.start();
```

الـتـــحـــكـــم فـــى الـــفـــيـــلـــم

احيانا نحتاج لوقف الفيلم عند فريم معين فنستخدم الدالة

```
stop ();
```

اما للتنقل بين الفريم و الذهاب من فريم الى اخر نستخدم الامر التالى

```
gotoAndPlay(7);
gotoAndPlay(20);
// ممكن ان نستخدم اسم الفريم بدلا من رقمه
gotoAndPlay("my frame label")
```

اما للذهاب الى Scene اخرى

```
gotoAndPlay("My Scene","My Frame");
```

Moving and Changing Movie Clips تحريك و تعديل الـ

كل عنصر فى ساحة العمل له موقع محدد يقاس بالبكسيل pixels حيث ان الجزء الايسر العلوى من مسرح العمل يعتبر ٠,٠ و هى اول نقطة للألتقاء المحور الافقى و الرأسى لمسرح العمل .
و لكى نعرف مكان عنصر على مسرح العمل نستخدم الخاصيتين _x and _y

مثال لتحديد مكان movieclip :-

نقوم بعمل movieclip و نقوم بتسميته myclip1 ثم نكتب فى خانة instace الاسم الذى نختاره للتعامل مع هذا الـmovieclip و ليكن اسمه myClip ثم نكتب الكود التالى :

```
trace(myClip._x);
trace(myClip._y);
```

فيظهر الناتج التالى مثلا و هذا يتوقف على مكان الـmovieclip

```
268
115.5
```

و يمكنك تغيير مكانه من خلال الكود التالى :-

```
myClip._x = 200;
myClip._y = 250;
```

الـفـصـل الـعـاشـر

مـعرفة مكان الفأرة فى الفيلم Mouse Location

لكى نعرف موقع الفأرة على مسرح العمل نستخدم الخاصيتين_xmouse و _ymouse
مثال : قم بعمل Movie clip و قم بتحديده و اضغط f9 و اكتب الكود التالى :-

```
onClipEvent (enterFrame) {
   trace(_root._xmouse);
   trace(_root._ymouse);
   trace("");
}
```

فتلاحظ فى شاشة المخرجات انك كلما تحركت بالماوس تغيرت الاحداثيات لاننا استخدمنا الحدث enterFrame .
كود أخر لكن يجب عمل ٢ مربع نص و اسم الاول box1_txt و الثانى box2_txt و نضع الكود التالى فى اول فريم فى الفيلم :-

```
var mouseListener:Object = new Object();
mouseListener.onMouseMove = function() {
  // تقوم الدالة بأرجاع الاحداثى السينى و الصادى للماوس
// return X and y
  box1_txt.text = _xmouse;
  box2_txt.text = _ymouse;
};
Mouse.addListener(mouseListener);
```

❋ ❋

الـفـصـل الـحـادى عـشـر

عـمـل تـدويـر لـلـمـوفـى كـلـيـب Movei Clip Rotation

مثال : قم بعمل موفى كليب واعطيه الاسم myClip واضغط على الفريم رقم ١ و اضغط f9 و اكتب الكود التالى :

```
myClip._rotation = 90;
myClip._rotation++;
_root["myClip"]._rotation = 45;
this._rotation += 0.5;
```

★★★★★★★★★★★★★★★★★★★★

Stretching and Shrinking Movie Clips عمل تمدد للموفى كليب

*** باستخدام الخاصية _xscale ***

مثال قم بعمل موفى كليب و سوف نشير اليه فى الكود من خلال التعبير this .

```
onClipEvent (load) {
   // نعرف الاحداثيات الاصلية للموفى كليب
   origWidth = this._width;
   origHeight = this._height;
}
onClipEvent (enterFrame) {
//معرفة المسافة بين منتصف الموفى كليب و الفأرة
   dx = _root._xmouse-this._x;
   dy = _root._ymouse-this._y;
//حساب النسبة المئوية للتمدد
   sx = 100*dx/(origWidth/2);
   sy = 100*dy/(origHeight/2);
   //  تحديد قيمة التمدد و اسنادها للموفى كليب
   this._xscale = sx;
   this._yscale = sy;
}
```

☜ يمكننا عمل محاكاة لحركة الـ3D بالتمدد للموفى كليب من خلال الكود التالى :-

```
onClipEvent(load) {
   scaleAmt = 10;
   x = 525;
   y = 25;
}
onClipEvent(enterFrame) {
   scaleAmt++;
   x -= 5;
   y += 5;
   this._xscale = scaleAmt;
   this._yscale = scaleAmt;
   this._x = x;
   this._y = y;
}
```

** كما يمكن كتابة نفس الكود بطريقة اخرى ونختصر بعض السطور كما يلى :-

```
onClipEvent(load) {
   this._x = 525;
   this._y = 25;
}
onClipEvent(enterFrame) {
   this._xscale++;
   this._yscale++;
   this._x -= 5;
   this._y += 5;
}
```

التحكم فى رؤية العناصر الموجودة على مسرح العمل فى برنامج فلاش Visibility

```
myClip._visible = false;
```

الخاصية visible تأخذ معاملين اما true او false اى يظهر او لا .

```
myClip._alpha = 50;
```

الخاصية alpha شدة الوضوح تأخذ قيم من ٠ حتى ١٠٠
لعمل تأثير الـFading

```
onClipEvent(enterFrame) {
  if (this._alpha > 0) {
      this._alpha -= 5;
   }
}
```

السحب و الالقاء للموفى كليب Selecting and Dragging Movie Clips

ترتبط الخاصية hitTest بعملية السحب و الالقاء حيث ان هذه الخاصية هى المسئولة عن معرفة هل الفأرة قامت بلمس الموفى كليب على مسرح العمل ام لا
و لكن ذلك مع استخدام الحدث المناسب و هو onClipEvent(mouseDown) or onClipEvent(mouseUp) و يمكنك تجربة الكود التالى :

```
onClipEvent (mouseUp) {
   this.gotoAndStop(2);
}
```

و لكن بفرض ان هناك اكثر من موفى كليب على مسرح العمل فيمكنك معرفة اى منهم تم النقر علية بالفأرة من خلال الكود التالى :

```
onClipEvent (mouseUp) {
   if (this.hitTest(_root._xmouse, _root._ymouse)) {
      this.gotoAndStop(2);
   }
}
```

السحب و الافلات Dragging :-

```
onClipEvent (mouseDown) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.startDrag();
    }
}
onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();
    }
}
```

الفصل الثانى عشر

اصطياد المفاتيح Detecting Keypresses

معرفة المفتاح الذى تم الضغط علية فى لوحة المفاتيح من خلال اسناد الكود التالى الى موفى كليب :

```
on (keyPress "a") {
   trace("Key 'a' pressed.");
}
```

و هذة قائمة بالمفاتيح التى يمكن تجربة الكود السابق معها

<Left>	<End>	<PageUp>
<Right>	<Insert>	<PageDown>
<Up>	<Delete>	<Tab>
<Down>	<Backspace>	<Escape>
<Home>	<Enter>	<Space>

مثال أخر

```
on (keyPress "<Left>") {
   trace("Left pressed.");
}
```

أما اذا كنت تريد معرفة اذا كان المفتاح يتم الضغط علية حاليا و الضغط مستمر مثلما فى الالعاب عندما يضغط اللاعب على مفتاح للأمام لكى تستمر الحركة فى اللعبة مثال :

```
if (Key.isDown(Key.LEFT)) {
   trace("The left arrow is down");
}
```

الدالة Key.isDown الناتج الخاص بها هو true او false و فى حالة الضغط المستمر يكون true .

قائمة بالمفاتيح الاخرى التى يمكن تجربة الكود السابق معها :

Key.BACKSPACE	Key.ENTER	Key.PGDN
Key.CAPSLOCK	Key.ESCAPE	Key.RIGHT
Key.CONTROL	Key.HOME	Key.SHIFT
Key.DELETEKEY	Key.INSERT	Key.SPACE
Key.DOWN	Key.LEFT	Key.TAB
Key.END	Key.PGUP	Key.UP

عمل متصنت لحركة المفاتيح Key Listeners

بداية نقوم بأخبار الفلاش اننا قمنا بعمل متصنت لحركة المفاتيح

```
Key.addListener(_root);
```

ثم بعد ذلك نقوم بعمل دالة لتتبع حركة المفتاح عندما يرفع المستخدم اصبعة عن المفتاح لكى نعرف الكود المقابل للمفتاح فى الاسكى كود مثال

```
_root.onKeyUp = function() {
   trace(Key.getAscii());
}
```

اضغط على الحرف a و كود الاسكى المقابل لة هو ٩٧

و باستخدام الكود التالى يمكنك تحويل كود الاسكى للحرف المقابل لة فى لوحة المفاتيح

```
trace(String.fromCharCode(Key.getAscii()));
```

مثال : افترض انك تريد عمل لعبة عبارة عن سيارة صغيرة تسير من خلال مفاتيح الاتجاهات فى لوحة المفاتيح فما هو الكود الازم لعمل ذلك .

فكر قليلا يمكنك استخدام ما تعلمناة فى الدروس السابقة لعمل هذة اللعبة بسهولة

ارى انك اقتربت من الحل و لكنك بحاجة لبعض المساعدة لذلك سوف نفكر سويا

بداية نحن نحتاج لعمل موفى كليب السيارة و لنفترض اننا قمنا برسم مربع صغير ثم قمنا بالضغط على f8 لتحويلة الى موفى كليب ثم قمنا بالنقر على الموفى كليب لتحديدة و انقر على f9 لتظهر شاشة محرر الكود ثم اكتب التالى فى حدث تحميل الموفى كليب نريد ان نعرف احداثيات الموفى كليب و عرفنا متغير speed و اعطيناة القيمة ٥//

```
onClipEvent(load) {
    x = this._x;
    y = this._y;
    speed = 5;
}
onClipEvent(enterFrame) {

    if (Key.isDown(Key.LEFT)) {
        x -= speed;
    }
        if (Key.isDown(Key.RIGHT)) {
        x += speed;
    }
        if (Key.isDown(Key.UP)) {
        y -= speed;
    }
        if (Key.isDown(Key.DOWN)) {
        y += speed;
    }

    this._x = x;
    this._y = y;
}
```

و يمكن كتابة الكود السابق بطريقة اخرى ليعطى نفس النتيجة مع كتابة الاتجاة على مسرح العمل :
فى البداية قم بعمل موفى كليب السيارة و قم بتسميتة فى مربع : instance of
car_mc ثم قف على الفريم الاول للفيلم ثم اضغط f9 و اكتب فى نافذة الكود التالى :

```
// متغير رقمى لحساب المسافة
var distance:Number = 10;
```

انشأ مربع نص اثناء تنفيذ الكودو لاحظ انك تستخدم دالة انشاء مربع النص التى لها معطيات يجب ان تاخذها منها اسم مربع النص و العمق و مكانة x و الy والطول و العرض

```
this.createTextField("display_txt", 999, 0, 0, 100, 20);

var keyListener:Object = new Object();

keyListener.onKeyDown = function() {

  if (Key.isDown(Key.LEFT)) {
// لاحظ أستخدامنا للدالة Math و هى المسئولة عن العمليات الرياضية
//  و من خصائص هذة الدالة الخاصية max التى تعود بالقيمة الاكبر
    car_mc._x = Math.max(car_mc._x - distance, 0);

    display_txt.text = "Left";

  } else if (Key.isDown(Key.RIGHT)) {
```

```
// و من خصائص هذة الدالة الخاصية min التى تعود بالقيمة الاصغر
    car_mc._x = Math.min(car_mc._x + distance, Stage.width -
  car_mc._width);
    display_txt.text = "Right";
  } else if (Key.isDown(Key.UP)) {
    car_mc._y = Math.max(car_mc._y - distance, 0);
    display_txt.text = "Up";
  } else if (Key.isDown(Key.DOWN)) {
    car_mc._y = Math.min(car_mc._y + distance, Stage.height -
  car_mc._height);
    display_txt.text = "Down";
  }
};
Key.addListener(keyListener);
```

الفصل الثالث عشر

الوقت و التاريخ Dates and Times

معرفة الوقت و التاريخ الحالى

```
// Create a new Date object.
today = new Date(  );
// Displays client-side date and time
trace(today);
```

الناتج

```
Tue Jul 11 17:17:48 GMT+0300 2006
```

معرفة السنةو الشهر و اليوم و الساعة و الثانية

```
var dDateObject:Date = new Date(year, month, date, hour,
minute, second, millisecond);
```

معرفة السنة من تاريخ معين

```
var dWhen:Date = new Date(1978, 9, 13);
trace(dWhen.getYear()); // Displays: 78
trace(dWhen.getFullYear()); // Displays: 1978
```

الفصل الرابع عشر

Using Data استخدام البيانات و المعلومات

يمكننا تخزين البيانات فى ملف نصى و استدعائها داخل الفلاش مثال :
قم بعمل ملف نصى باستخدام المفكرة و عرف فية متغيرين و اكتب فية التالى

```
color=red&area=100
```

ثم قم بحفظ الملف باسم variables.txt
ثم افتح ملف فلاش جديد و قم بحفظة فى نفس المجلد الذى بة الملف النصى .
ثم اكتب الكود التالى فى اول فريم فى الملف

```
//نستخدم الدالةLoadVars لتحميل متغيرات اثناء عرض الفيلم
var lvSampleData:LoadVars = new LoadVars();
// ثم نقوم بعمل دالة لنعرف منها هل تم تحميل البيانات ام لا
lvSampleData.onLoad = function(bSuccess:Boolean):Void {
        trace("data loaded");
      };
lvSampleData.load('variables.txt');
```

و عند تجربة العمل تجد النتيجة اما true او false
و يفترض انها true اذا كنت و ضعت الملفين فى نفس المكان على الهارد ديسك .
أما اذا كان نريد معرفة قيمةالمتغيرين color و area فاننا نستخدم الكود التالى :

```
lvSampleData.onLoad = function(bSuccess:Boolean):Void {
        if(bSuccess) {
          trace('color = ' + this.color);
          trace('area = ' + this.area);
        }
      else {
          trace('There is an error with the data.');
        }
      };
lvSampleData.load('variables.txt');
```

الناتج
color = red :
area = 100

اما اذا كنت تريد عرض محتويات الملف النصى داخل الفلاش فيمكنك استخدام الكود التالى :

```
var lvExample:LoadVars = new LoadVars();
      lvExample.onData = function(sText:String):Void {
        trace(sText);
      };
      lvExample.load('variables.txt');
```

الناتج
color=red&area=100

مثال أخر
قم بعمل ملف نصى و قم بتسمية electricBill.txt و اكتب فية الكود التالى :

```
&electricBill=60
```

ثم قم بعمل ملف فلاش جديد و اكتب الكود التالى فى الفريم الاول

```
//انشأ مربع نص بالكود لكى نستقبل فية قيمة المتغير المكتوبة فى الملف النصى.
this.createTextField("my_txt", 99, 10, 10, 200, 20);
my_txt.border = true;
my_txt.type = "input";
```

استخدمنا الدالة Loadvars () لتحميل المتغير //

```
var externalData:LoadVars = new LoadVars();

externalData.onLoad = function(){
```

عند تحميل الملف النصى و قراءة محتواة تتغير قيمة مربع النص my_txt لتصبح قيمة المتغير electricBill .//

```
 my_txt.text = externalData.electricBill;

}
externalData.load("Electric_Bill.txt");
```

الناتج:
مربع نص مكتوب بة ٦٠

كيف يتعامل الفلاش مع ملفات التخزين الخارجية مثل لغة الXml

ملحوظة مرفق فى اخر الكتاب ملحق عن لغة الXml بالتفصيل يجب الرجوع الية قبل قراءة هذا الفصل و بعد ذلك يمكنك متابعة القراءة و تنفيذ الاكواد :

استخدام the XML class

تستخدم فى تحميل و ارسال البيانات الى السيرفر من خلال HTTP POST method

- تقوم بتحميل ملف اكس ام ال من رابط و تضعة داخل كود الاكشن اسكربت The load()
- تقوم بارسال البيانات و اعادة البيانات مرة اخرى The send()
- تقوم بارسال البيانات و تحميل الناتج The sendAndLoad()

الكائن The XML Object

هذا الكائن يحتوى على مجموعة من الدوال و الخصائص التى تساعدك على التعامل مع لغة الxml لذلك يجب فى البداية عمل متغير من هذا الكائن لنستطيع التعامل معة .

```
myXML = new XML();
```

و يمكننا كتابة كود الxml داخل كود الاكشن اسكربت بواسطة الامر parseXML الذى يمكنك من كتابة الxml كأنها string

```
myXML = new XML();
myXML.parseXML("<user><name>Gary</name><ID>47</ID></user>");
```

طريقة اخرى مختصرة

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
```

و لكن كيف نتأكد من اننا كتبنا الامر السابق بشكل صحيح دون ان نخطأ فى شئ بالنسبة لكود الxml بالطبع هناك دالة جاهزة لذلك و هى

```
trace(myXML.status);
```

اذا كان الناتج

-٩ معنى ذلك ان هناك تاج فتح و لم يقفل.

-١٠ معنى ذلك ان هناك تاج قفل و لم يكن مفتوح من قبل .

اما اذا كان الناتج0 فمعنى هذا ان الكود مكتوب بطريقة صحيحية .

استخراج البيانات من ملف xml

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.firstChild);
```

الناتج **"<user><name>Gary</name><ID>47</ID></user>".**

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.firstChild.firstChild);
```

الناتج <name>Gary</name>

طريقة اخرى باستخدام childNodes

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.childNodes[0].childNodes[0]);
```

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.childNodes[0].childNodes[0].childNodes[0]);
```

الناتج : Gary

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.childNodes[0].childNodes[1].childNodes[0].nodeValue);
```

النتاج : ٤٧

عمل ملف xml من البداية :

```
myXML = new XML();
newElement = myXML.createElement("user");
myXML.appendChild(newElement);
newElement = myXML.createElement("name");
myXML.childNodes[0].appendChild(newElement);
newText = myXML.createTextNode("Gary");
myXML.childNodes[0].childNodes[0].appendChild(newText);
newElement = myXML.createElement("ID");
myXML.childNodes[0].appendChild(newElement);
newText = myXML.createTextNode("47");
myXML.childNodes[0].childNodes[1].appendChild(newText);
```

If you want to change the value of a text node, you need to set the nodeValue property:

اذا كنت تريد تغيير قيمة النص فى عقدة يجب عليك اضاف خاصية nodevalue

```
myXML.childNodes[0].childNodes[1].childNodes[0].nodeValue = 53;
```

Attributes

```
<user>
   <name type="alias">Gary</name>
   <ID>47</ID>
</user>
```

مثال :

```
myXML = new XML("<user><name type='alias'>Gary</name><ID>47</user>");
trace(myXML.childNodes[0].childNodes[0].attributes.alias);
```

طريقة اخرى

```
trace(myXML.childNodes[0].childNodes[0].attributes["alias"]);
```

يمكنك معرفة عدد العقد فى ملف الxml

```
myXML = new XML("<user><name>Gary</name><ID>47</user>");
trace(myXML.childNodes[0].childNodes.length);
```

الناتج ٢

يمكنك معرفة اسم العقدة ايضا

```
myXML = new XML("<user><name>Gary</name><ID>47</user>");
trace(myXML.childNodes[0].childNodes[0].nodeName);
```

مثال على ملف xml

```
<book>
<title>ActionScript Arabic book</title>
<authors>Michael nabil</authors>
</book>
```

الان مثال على كيفية عرض البيانات من داخل ملف ال xml السابق

قم بحفظ ملف ال xml السابق باسم books.xml

ثم افتح ملف فلاش جديد و قم بحفظة فى نفس المجلد الذى بة الملف book.xml

ثم اكتب الكود التالى فى اول فريم فى الملف

```
// book_xml و اسمة xmlنقوم بعمل كائن جديد من الكلاس
var book_xml:XML = new XML();
//xml نجبر الفلاش على تجاهل المسافات البيضاء فى ملف
this.book_xml.ignoreWhite = true;

// نقوم بتحميل ملف ال xml
this.book_xml.load("book.xml");
```

جدول بة الخصائص والدوال التى سبق و استخدمناها فى الامثلة السابقة ارجو ان تقوم بتجربتها لتعرف ما هى فائدتها .

Properties, Methods, and Return Types of the XML Object

الخصائص و الدوال الخاصة بالكائنxml

PROPERTY AND METHODS	RETURN TYPES
Attributes	Returns an associative array containing all of the attributes of the specified node
ChildNodes	Read-only; returns an array containing references to the child nodes of the specified node
FirstChild	Read-only; references the first child in the list for the specified node
hasChildNodes()	Returns true if the specified node has child nodes; otherwise, returns false
LastChild	References the last child in the list for the specified node
nextSibling	Read-only; references the next sibling in the parent node's child list

Properties, Methods, and Return Types of the XML Object

الخصائص و الدوال الخاصة بالكائنxml

PROPERTY AND METHODS	RETURN TYPES
NodeName	The node name of an XML object
NodeType	The type of the specified node (XML element –type = 1, text node –type = 3)
NodeValue	The text of the specified node if the node is a text node
ParentNode	Read-only; references the parent node of the specified node
previousSibling	Read-only; references the previous sibling in the parent node's child list
RemoveNode()	Removes the specified node from its parent
toString()	Converts the specified node and any children to XML text

الفصل الخامس عشر

البرمجة بالكائنات

Object-Oriented Programming

Object : وتعني شئ أو كائن (حاجة) ... حيث يعتبر الكرسي Object ، والسيارة أيضاً Object أو شئ ... وهكذا ...

Oriented: مثلاً لو أننا تعرفنا على شخص ما وكان همه الوحيد في هذه الدنيا المال ، فأننا نستطيع أن نطلق عليه أنه Money-oriented person أو شخص تحركه المادة ، حيث أن المال هو المحرك الأساسي في حياته.

Object-oriented Programming: والمقصود بها أن المبرمج يحلل العمل المراد منه وبناءه على أساس أنه مجموعة من الأشياء منفصلة Objects ، ولكنها في نفس الوقت تتعامل مع بعض البعض على أسس منظمة ومحددة. مثلاً في منظومة المكتبة يمكننا أن نعتبر أن الطالب Object ، وله أسم ورقم ، والعمليات التي يمكن أن يقوم بها في المكتبة هي القراءة والاستعارة وترجيع كتب... وهكذا...

OOP هي أولاً وأخيراً طريقة تصميم وبرمجة ، ولا تقدم أي ميزة اضافية على المنتج النهائي للبرمجيات (على سبيل المثال Execution Files او DLL Files) ، ولكنها في نفس الوقت للمبرمج الذي يستخدمها تقدم ميزات تسهل عليه التعامل مع البرامج التي هو يكتبها أو غيره من المبرمجين ، من حيث سهولة إدارة البرامج وإعادة استخدامها من قبل فرق العمل ، أيضاً لسهولة تخيل البرامج لأنها مصممة من البداية على أسس OOP وتكمن سهولةOOP في محاكاتها للكائنات.

يولي البعض أهمية كبيرة لمفاهيم كبيرة وكثيرة للبرمجة الشيئية ، فهم من جهة لا يدركون أي أهمية لمصطلحاتها وتقنياتها ، ومن جهة أخرى فهم لا يستطيعون الفرار من قوقعة التفكير الهيكلي .... لذلك نجد أن كثيراً من البرامج التي كتبت والتي ما زالت حتى الآن مكتوبة بأسلوب هيكلي وليس بأسلوب شيئيي حتى وإن وجدت في الكود عبارات مثل: class و extends وغيرها ... تظل طريقة التفكير نفسها هي طريقة التفكير لدى تصميم أي برنامج آخر بواسطة الدوال ، الان أحاول ولو بشكل بسيط معالجة بعض المسائل النظرية المتعلقة بالبرمجة الشيئية دون أي تعمق في المسائل الكودية المختلفة بين لغات البرمجة.

+الفرق بين التفكير الهيكلي والتفكير الشيئي.
+التجريد.
+الكائنات والكبسلة.
+التماسك والتزاوج.
+الوراثة وتعدد الأوجه.
+أنماط التصميم البرمجية.

الفرق بين التفكير الهيكلي والتفكير الشيئي:

لحظة تفكير المبرمج بالبرنامج الجديد الذي يريد كتابته هي اللحظة الفاصلة التي تجعله يقرر بين إن كان يفكر بشكل شيئي أو هيكلي ، إذا افترضنا أن البرنامج الجديد هو عبارة عن نظام لإدارة مدرسة ابتدائية فإن غالبية قراء هذه المقالة سيبدؤون التفكير بتحديد مهمة هذا البرنامج ، ثم بعد ذلك يتم تقسيم البرنامج إلى مجموعة كائنات ، ويتم إسناد مهمة لكل كائن .... طريقة هذه التفكير هي طريقة التفكير الهيكلي .. إن السؤال الأساسي المفترض عند بداية تصميم أي برنامج ليس ماهي مهمة هذا البرنامج ... بل ما هو العالم الذي تريد مني نمذجته في هذا البرنامج ... وبالتالي فإن الإجابة ستكون نمذجة عالم المدرسة الابتدائية ... إذا رجعنا لطريقة التفكير الهيكلي فإن الإجابة ستكون مهمة هذا البرنامج إدارة هذه المدرسة .. وعند التعمق أكثر ستجد أن هذه الطريقة في التفكير تقسم البرنامج إلى مجموعة كائنات ، ستجد كائن المدير الذي سيؤدي مهمة ما ، ثم بعد ذلك كائن المرشد الطلابي الذي سيؤدي مهمة ما ، ثم حاوية تحوي كائنات الطلاب والذين سيؤدون مهمة ما ... إن هذه الطريقة في التفكير طريقة غير شيئية بل هي طريقة تفكير مشوهة .. فلا هي هيكلية ولا شيئية بل خليط لا طائل منه سوى تشويه أسلوب البرمجة.

من جهة أخرى سنجد أن طريقة التفكير الشيئي ستقسم البرنامج إلى مجموعة كائنات ، سيكون هناك كائن المرشد الطلابي وحاوية تحوي طلاب المدرسة وستكون العلاقة بين المرشد وهذه الحاوية هي علاقة صداقة لأن من حق المرشد الطلابي الإطلاع على أسرار الطلبة ، لاحظ أيضاً هناك أنه يوجد أكثر من حاوية ، وبالتالي فإن

الطالب الواحد سيكون موجوداً أكثر من حاوية ، هناك حاوية الفصل ، وحاوية للصف ، وحاوية للمدرسة ، عند انتهاء السنة الدراسية ستنتهي الفصول والصفوف وسيكون من الواجب إلغاؤها ... فهل سيكون من حق هذه الحاويات إلغاء الطلبة وبالتالي إلغاء المدرسة ... لذلك عليك التفكير عن العلاقة بين الحاوية وعناصرها .. هل سيكون من حق الحاوية امتلاك عناصرها وبالتالي إلغاءها أم لا ... لاحظ أيضاً هنا أن كائن المعلم سيكون من حقه مراقبة الطلبة وتقييمهم .... طريقة التفكير السابقة لن تفكر بالعلاقات بين الكائنات بهذا الشكل .. وستحل المشاكل السابقة "مشاكل الحاويات مثلاً" بطريقة هيكلية وليست شيئية .. لأن المبرمج الذي يفكر بطريقة هيكلية سيركز فقط على إيجاد الكائنات وسيعتبر وجود الكائنات كأنه وجود للمتغيرات الأساسية فقط وبالتالي فإنه سيحل المشكلة بطريقة هيكلية أما بالنسبة لطريقة التفكير الشيئية فهو سيركز أيضاً على إيجاد العلاقات بين الكائنات .. لأن العالم يحتوي على الكائنات وعلى العلاقات بينها ، وسيركز أيضاً على مزيد من التجريد **Abstraction** لكي يكون تركيزه على إيجاد حل لميدان المشكلة (سنصل لهذه النقطة لاحقاً).

معنى التجريد:

تعتبر لغة التجميع تجريداً للغة الآلة ، فكلمة **mov** هي تجريد لأمر التحريك الموجود في لغة الآلة ، حتى تفهم ميزة التجريد بشكل عام ، فلنقول أنك تريد برمجة برنامج بواسطة لغة الآلة ، تركيزك الأساسي هنا لن يكون على إيجاد حل للمشكلة التي تعمل عليها فقط .. بل أيضاً على إيجاد حلول لمشاكل أخرى مثل مشاكل الأخطاء البشرية عند إدخال الأوامر بصورتها الثنائية ، أيضاً يجب عليك معرفة التفاصيل الداخلية للجهاز الذي تعمل عليه ... أي يجب عليك التركيز على مشاكل أخرى غير ميدان المشكلة الذي تعمل عليه ... لنفرض مرة أخرى أنه طلب منك برمجة نفس البرنامج ولكن هذه المرة بواسطة لغة التجميع "الاسمبلي" ، أنت تعلم أن هذه اللغة هي تجريد للغة الآلة ... وبالتالي فأنت لن تهتم بشأن الأخطاء البشرية لأنه بإمكانك كتابة أوامرك بواسطة لغة الإنسان ، هذا الأمر سيجعلك تركز أكثر وأكثر على ميدان المشكلة الذي تريد حله وسيريحك بالتالي من التفكير في مشاكل أخرى .. صحيح أن لغة الاسمبلي ليست ذات مستوى متقدم من التجريد وستضطر في النهاية إلى التفكير ببعض المسائل غير ميدان مشكلة البرنامج الذي تريد حله ولكن في النهاية فالأمر أفضل من البرمجة بلغة الآلة .

لغة السي أتت أيضاً كتجريد للغة الاسمبلي ... في لغة الاسمبلي يجب عليك الاهتمام بالجهاز الذي تعمل عليه بينما هذا الأمر غير موجود في لغة السي ، فقط اكتب الكود الذي تريده ثم ترجمه على أي جهاز آخر وسيعمل لديك .. وهذا يعطيك مزيد من التفكير بشأن المشكلة التي تريد حلها في برنامجك ، ويقوم بإلغاء التفكير بشأن المشاكل الأخرى مثل ما هي التفاصيل الداخلية للجهاز الذي تعمل عليه.

نستطيع القول هنا أيضاً أن لغة الجافا هي تجريد آخر ، في لغة السي لا تستطيع أخذ الملف التنفيذي ونقله إلى جهاز آخر أو إلى نظام تشغيل آخر لأنه لن يعمل ... الملف التنفيذ الصادر عن لغة السي يعمل فقط على الجهاز الذي تمت ترجمة السورس كود عليه ، ويجب عليك الاهتمام هناك بتوفير مترجم للأجهزة الأخرى .. الأمر في لغة الجافا مختلف ففلسفتها هي: **Write Once, run every where** ، أي اكتب السورس كود وسيعمل الملف التنفيذي على كل جهاز وعلى كل نظام تشغيل.

فكرة التجريد مطبقة بشكل كبيرة في البرمجة الشيئية .

انواع البيانات التجريدية Abstract Data Type (ADT).

في لغة الاكشن اسكربت توجد أنواع معرفة مسبقاً من قبل مترجم اللغة. مثلاً number يعتبر نوع نقوم بإستخدامه لمعالجة Manipulate الارقام الصحيحة ، والمقصود بكلمة معالجة هنا هو إجراء العمليات المعتادة على هذا النوع مثل "- + * /" وهكذا ، لأن الجمع والطرح والضرب عمليات تجرى عادة على الأرقام.

السلاسل الحرفية Strings نستعملها عند وجود الحاجة الى معالجة مصفوفات من نوع حرف. وعلى العكس من نوع number الذي نستطيع أن نقوم بعمليتي الضرب والقسمة عليه ، فطبيعة البيانات التي من نوع السلاسل الحرفية لايمكن ضربها أو قسمتها.

ADT هي قاعدة الاساس في البرمجة بطريقة OO ، وهي الخطوة الاولى التي يتم فيها تصميم وتعريف أنواع جديدة. هذه الانواع الجديدة يتم تعريفها وترجمتها من اوصاف وأفعال النوع نفسه، فالاوصاف يتم التعبير عنها بمتغيرات Variables ، و الأفعال يتم التعبير عنها بدوال Functions. مثلاً :

الشئ / الكائن	أوصاف	أفعال
سيارة	لون ، عدد الابواب ، لون ، سرعة ، نوع ، إسم ، ...	تشغيل ، إيقاف ، تغيير سرعات ، ...
صالون	لون ، نوع القماش ،عدد	تركيب ، تنظيف ،

	الكراسي ، ...	فك...

الأوصاف والأفعال التي يتم تعريفها لهذه الأنواع الجديدة تختلف من برنامج الى أخر ، فمثلاً لو أردنا أن نقوم بتعريف نوع سيارة لإستخدامه في برنامج تحكم بالسرعات فإن لون السيارة سيكون غير مهم بالنسبة لنا.

يسمى هذا النوع من البيانات بالبيانات التجريدية وذلك لأنها لازالت تحتاج الى تعريف محدد لإستخدامها من قبل لغات البرمجة ، فنوع القماش مثلاً عبارة عن معلومة يجب ترجمتها حتى يمكن التعبير عنها ، فمثلاً :

```
number Fabric;  // 1=Blue, 2=Red....
```

او

```
String Fabric; // B=blue, R=red, ...
```

الكائنات والكبسلة:

للبرمجة الشيئية فلسفتها القائمة على تقسم البرنامج ليس إلى مهام كما هو الحال في البرمجة الهيكلية بل إلى أجسام أو كائنات في العالم الفيزيائي الحقيقي ، مفاهيم الكائنات هي على كل حال ما يلي:

الفئات classes :

تساعد الفئة على تمييز الكائن من خلال تعريفها ، إذا نظرنا مثلاً إلى فئة اسمها سائق ، ستجد أن هناك فرق بين السؤال: كيف يقود السائق ، وفرقاً بين السؤال: كيف يقود السائق أحمد ، الكائن أحمد هو سائق سباق سيارات رالي.

عند السؤال عن ما هو السائق، فإنك في الحقيقة تتكلم عن حالة عامة ، كل السائقين يعلمون كيفية تشغيل السيارة ، وكل السائقين لديهم رخصة قيادة (إلا في حالات شاذة طبعاً) ، وكل السائقين لديهم مخالفات مرورية ، وكل السائقين لديهم سيارة أيضاً؛ باختصار فإن الفئة **class** تقوم بكبسلة تصنيف معين لمجموعة من الكائنات **objects** .. وهو في مثالنا هذا كائنات السائقين.

ولكن حينما نتحدث عن سائق معين فنحن هنا ننتقل من مفهوم الفئة **class** إلى مفهوم الكائن **object** ...حينما نتحدث عن الكائن أحمد في المثال السابق ، فإنه وبسبب معرفتنا بأنه ينتمي إلى فئة السائقين فإننا نعرف أنه يملك رخصة قيادة ولديه سيارة ، ما لا نعرفه ربما بعض السمات والخصائص المتواجدة لدى السائق أحمد ... كم سيارة لديه ؟ ، ما مدى مهاراته في قيادة السيارات؟ ، ما نوع الرخصة المرورية التي لديه ... وبالتالي فإن الكائن **object** هو حالة خاصة **instance** من الفئة.

الخلاصة :ان الوراثة **Inheritance**.

هي إمكانية توريث صفات وأفعال **Class** معينة الى **Class** اخرى جديدة ، وذلك لتجنب عملية تكرار الصفات المشتركة بين الـ **classes**. حيث تسمى الـ **Class** الجديدة بالابن والاولى بالأب.

المكونات Components :

التفكير في الكائن عن طريق مكوناته الصغيرة أمر ضروري وأساسي للغاية في البرمجة الشيئية ، فئة السيارة مثلاً تتكون من المحرك وعجلة القيادة ومخزن البنزين والإطارات ... إلخ ؛ المكون في النهاية هو عبارة عن فئة **class** ولكنها أقل حجماً وأكثر تحديداً ، فئة السيارة إذا تم وصفها في الكود على أنها سيارة فقط فسيكون الكود أقل مقروئية ، ولكن بدلاً من ذلك ماذا لو تم وصف فئة السيارة على أنها فئة تدير وتتعامل مع مكونات أقل حجماً ، هذا سيسهل كثيراً من قراءة الكود ومن تصميم الكود ويجعلك قادراً على فهم الفئة **class** بشكل أوضح كما هي في العالم الحقيقي ، لاحظ أيضاً هنا أنه من الممكن أن تحتوي هذه المكونات على مكونات أخرى.

الخواص Properties:

السمات أو الخواص هو ما يفرق الكائنات عن بعضها البعض ، إذا نظرنا إلى العالم الحقيقي سنجد أن ما يفرق بين إنسان وإنسان آخر هو شكل الوجه مثلاً والطول وحتى العقيدة ومجموعة القيم التي تكون الإنسان والأخلاق .. وغير ذلك من الخواص الخاصة بكل إنسان.

بإمكانك التفكير عن الخواص في مستوى الفئات **classes** ، فمثلاً في مثال السائق ، سنجد أن جميع السائقين يجب أن يكونوا يملكون رخصة قيادة ، وبالتالي فهذه خاصية مشتركة وعامة بين جميع الكائنات من نوع فئة السائق.

السلوكيات Behaviors:

بإمكاننا التفكير عن السلوك على مستوى الكائن ومستوى الفئة ، في مثال السائق سنجد أن هناك سلوك اسمه: يقود (الموقع الذي تريده) ، وسنجد أن جميع الفئات ستشترك في فعل القيادة هذا ، ولكن الاختلاف هنا يمكن بين كل كائن وكائن ، فإذا كان الكائن أحمد سريعاً للغاية فإنه سيصل في وقت أسرع ولكن احتمالات وقوع حادث كبيرة ، بينما الكائن محمد سائق هادئ نوعاً ما فإنه سيصل في وقت أقل ولكن بدرجة أمان أكبر من صاحبه السابق.

الكبسلة Encapsulation:

الفئة والكائن تعتبران في الأساس تغليفاً للسلوكيات والخصائص في مجموعة واحدة أو قالب واحد ، هذه الكبسلة والتغليف ستسهل عليك البرمجة بشكل كبير فأنت الآن ستكون قادراً على التعامل مع هذه الكبسولة بشكل شامل ، فأي تغيير في إحدى خصائص كائن ما ، سيقوم بتغيير جميع خصائص هذا الكائن ، لنأخذ مثالاً واقعياً لنفرض أن الطالب أيمن رسب في اختبار إحدى المواد ، سيؤثر هذا على معدله العام وليس ذلك فحسب بل سيؤثر على خصائصه المتبقية فهو أولاً سيتأثر نفسياً وبالتالي سيؤثر هذا الأمر على حالته الجسدية العامة ، لكن إذا نظرنا إلى البرمجة الهيكلية فحينما تقوم بجعل درجة الطالب أحمد درجة سيئة ستكون مضطراً على تغيير كل خاصية من خصائصه مما سيجعل الكود طويلاً وليس ذا مقروئية عالية ، لكن الأمر يختلف في البرمجة الشيئية فبواسطة سطر واحد من الكود تستطيع فعل كل هذه التغييرات .

الخلاصة: ان الكبسلة "التجميع والتغليف": وهي تمثل عملية تجميع صفات وأفعال شئ معين ووضعها داخل مغلف وأخيراً تسميتها بأسم هذا الشئ ... مثلاً الـ Stack ، له صفات "Attributes" وهي حجم ، ونوع ، وحيز ... أيضاً الـ Stack له عمليات أو أفعال "Operations or Methods" يقوم بها وهي Pop و Push.

إخفاء البيانات Data Hiding :

من المفيد أن يكون هناك تجزئة في قطاع البرمجة إلى صانعي الفئات class creator وإلى مستخدمي الفئات أو المبرمجين الزبائن client programmers ، النوع الأول من المبرمجين هو من يقوم بإنشاء الفئات والنوع الثاني هو من يستخدم هذه الفئات ويجمعها لكي يبني من خلالها نظاماً أو برنامجاً متكاملاً . الوظيفة الأساسي لصانعي الفئات فهي بناء فئة يكشف فيها للمبرمج الزبون ما هو ضروري فقط، لماذا؟ .. حتى يسمح للمبرمج الزبون بالتفكير في ميدان المشكلة دون التورط في التفكير بشأن مسائل أخرى.

بالرغم من الفائدة الضرورية لعمل صانه الفئات إلا أنه سيفرض قيود على المبرمج الزبون عند استخدام فئة ما، فهو لا يستطيع الوصول إلى الأعضاء المخفيين private .. لكن إذا فكرت قليلاً ستجد أن هذه الأجزاء الداخلية مهمة للعمل الداخلي داخل الفئة وليس لها أي علاقة بتاتاً بالأعضاء العامة public أو واجهة الفئة .... في مثال السائق الذي يوجد فيه سلوك (أو دالة إن شئت) اسمها يقود ، ستجد أنك تستخدم هذا السلوك دون أي تفكير مسبق في كيف سيعمل هذا السلوك ، لمثال أوضح أنظر إلى هذا المثال المكتوب بواسطة لغة السي شارب:

```
Console.Write (" O O P\n");
```

هذا السطر يقوم بطباعة العبارة O O P ، لاحظ هناك أنك لا تهتم بما يحدث داخل الدالة Write بل كل ما فكرت فيه هو المشكلة التي تود حلها فقط وحينما احتجت طباعة عبارة ما ... فإنك قمت بفتح التوثيقات الخاصة بلغة السي شارب ووجدت أنه يوجد فئة اسمها Console لديها دالة ساكنة اسمها Write تقوم بطباعة أي كلام تريده على الشاشة ... ثم استخدمتها ولم تحتج بتاتاً لأي عمليات معقدة تقوم بها، لأن هذه العمليات المعقدة تحدث داخلياً داخل الفئة Console ، مثل هذا الأمر سيجعلك تفكر أكثر في المشكلة التي تريد حلها وليس في المشاكل الصغيرة التي ستتولد أثناء محاولتك حل المشكلة الكبيرة.

الخلاصة: ان إخفاء البيانات : وهي عملية إخفاء المتغيرات والدوال ، بالنسبة للمتغيرات يجب دائماً إخفائها عن طريق تحديدها بـ private اما الدوال فذلك يرجع لإحتياج مصمم البرنامج ، حيث يوجد ثلاث أنواع من الإخفاء public, protected and private ، سنؤجل شرح عمل الـ protected لعلاقتها بالوراثة.

- تعددية التشكل Polymorphism.

تعدد التشكل وهي منح المبرمج إمكانية تعريف دالتين أو أكثر بنفس الأسم ، ولكن يشترط أن تختلف كل من هذه الدوال في قائمة إرسال/إستقبال المتغيرات Parameters ، أياً كان هذا الإختلاف في العدد أو النوع أو كليهما.

التماسك والتزاوج Cohesion and Coupling :

نأتي الآن إلى أمر مهم يندر الحديث عنه رغم أهميته ولو أنه يتردد بكثرة في مواضيع هندسة البرمجيات ، وهو موضوع التماسك.

التماسك: هو قدرة المقطع الواحد على تنفيذ مهمة واحدة من دون الحاجة إلى استدعاء مقاطع برمجية أخرى.

من المعلوم أن أي برنامج يتم تقسيمه إلى مقاطع برمجية صغيرة ، وهو يقسم إما إلى فئات أو دوال أو إلى أي شيء آخر ... وهذا التقسيم هو ما يقصد به المقطع الواحد في التعريف السابق.

التزاوج هو عكس التعريف السابق ، وهو يعني أن أكثر من مقطع برمجي مرتبطة مع بعضها بشكل يجعلهم لا يستطيعون العمل إلا أثناء تواجدهم كلهم مع بعضهم في برنامج واحد.

من التعريفين السابقين نجد أنه كلما ازداد التماسك في الفئات قل التزاوج فيما بينهم ، هناك تقريباً سبعة أنواع من التماسك أقواها هو التماسك الوظيفي والذي يعني قدرة المقطع على أداء مهمة واحدة فقط من دون الاعتماد على أي مقطع برمجي آخر.

أما أضعفها فهو Coincidental Cohesion وهو حينما يقوم المقطع البرمجي بأداء أكثر من مهمة غير مترابطة.

لكي نطلق على لغة برمجة معينة بأنها تدعم البرمجة بطريقة OO يجب أن تتوفر فيها التقنيات الأتية:

١- انواع البيانات التجريدية Abstract Data Type (ADT).
٢- الكبسلة Encapsulation.
٣- إخفاء البيانات Data Hiding.
٤- تعددية التشكل Polymorphism.
٥- الوراثة Inheritance.

الان لنبدأ الدراسة العملية ل Object-Oriented Programming with ActionScript 2.0

ActionScript Classes

سوف نرمز للClasses فى حديثنا بالخلايا و احيانا قد تجدنى اكتبها كلاس بالعربية و ذلك للتسهيل و للتعود على المصطلح الاجنبى .

Class

وهي مكوّن أساسيّ لبناء الكود، ليس لها واجهة مرئيّة، ولكنّها تؤدّى وظائف معيّنة لبرنامجك.
وباختصار، فإنَّ الخليّة هي مجموعةٌ من الدوالّ والإجراءات التي يمكنُ إعادة استخدامها في أيّ مشروع، بحيثُ يوفّر المبرمجُ على نفسِه مشقّة إعادة كتابتها مرة أخرى.
لاحظ ان الخليّة هي برنامج لا يمكن تنفيذه بمفرده، إذ يجب أن يتمّ استخدامه بواسطة تطبيق آخر أو جزء آخر من نفس التطبيق الذي يحتوي على الخليّة.
ستسألني:

- أتعني أنّنا نستدعي الخليّة كما نستدعي الدوال والإجراءات الفرعيّة، باستخدام اسم الدالة.
- إطلاقا.. إنّ الخليّة تحتوي أساسا على العديد من الإجراءات والدوال والخصائص الخاصّة بها.
- إذن كيف نستخدم الخليّة؟
- أنّنا نعرّف متغيّرا من نفس نوع الخليّة، تماما كما نعرّف متغيرا من نفس نوع البيانات التى ينتمى لها المتغير .
- بالضبط.. وهذا يعني ببساطة، أنّ الخليّة هي إحدى أنواع المتغيرات التي يعرّفها المستخدم User Defined Types، وإن كانت هي أشملها وأقواها وأهمّها على الإطلاق، كما سنرى بعد قليل.

الآن تدرك بالطبع أنّك أوغلت في استخدام الخلايا، فهي ليست جديدة أو غريبة.. الجديد هو أنّك ستتعلّم هنا كيف تبني الخلايا الخاصّة بك.

- بقى ان تعرف ان الخلايا يتم تعريفها فى ملفات مستقلة و ليست فى الtime line و تكون هذة الملفات بامتداد as و يمكنك كتابة الكود الخاص بها فى المحرر الخاص بذلك الموجود فى برنامج فلاش النسخة الاحترافية او باستخدام النوت باد و حفظ الملف بامتداد as .
- لاحظ ايضاانة يجب ان يكون اسم ملف الخلية هو ذات اسم الخلية مثلا خلية اسمها Product يجب ان يكتب الكود الخاص بها فى ملف اسمةProduct.as و يكون هذا الملف فى نفس المجلد الذى سوف يحفظ فية ملف الفلاش الذى سوف يستخدم فية الخلية Product و يمكن ايضا ان يحفظ ملف الخلية فى مجلد فرعى داخل المجلد الرئيسى مثلا نقوم بعمل مجلد فرعى و ليكن اسمة Data و نحفظ فية ملف الخلية .

و للأستدعاء الخلية نقوم بكتابة الكود التالى فى ملف الفلاش :

```
var myProduct:Product = new Data.Product();
import Data.Product;
```

مثال يوضح طريقة كتابة الخلية :

```
class Product {

    var id:Number;

    var prodName:String;

    var description:String;


    function Product (id:Number, prodName:String, description:String)

    {
```

```
        this.id = id;
        this.prodName = prodName;
        this.description = description;
    }
}
```

لاحظ Public and Private Attributes

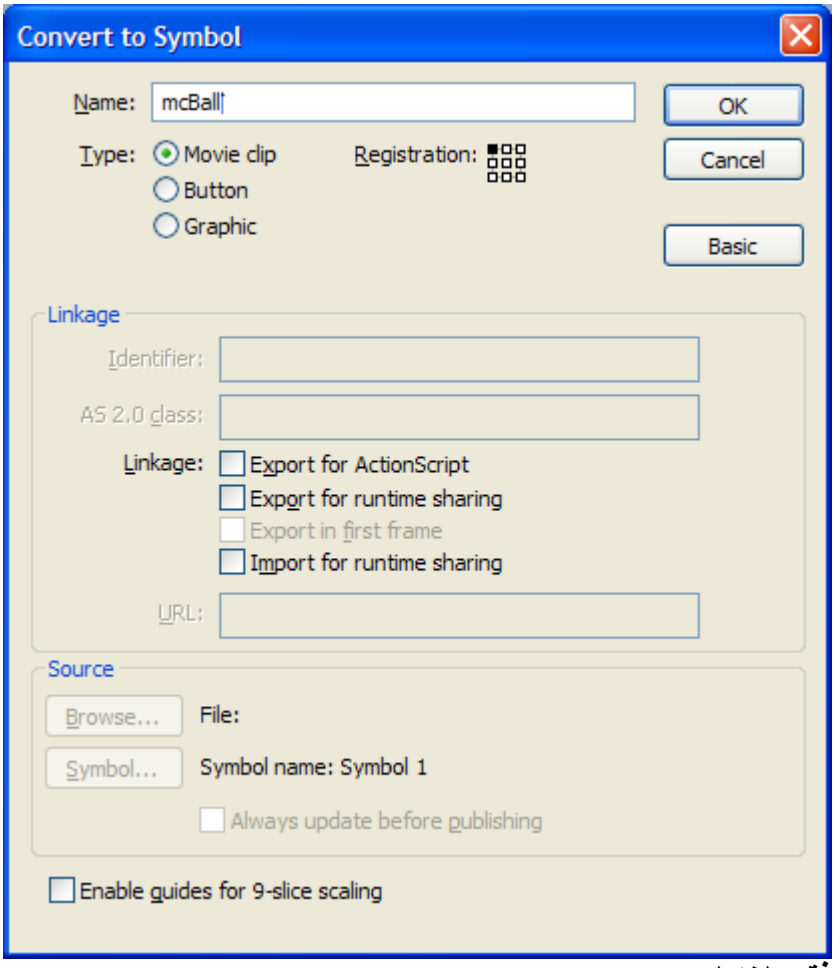
Public: ليصبح الخلية متاحا للاستخدام من أي موضع في مشروعك، أو من أي مشروع آخر يضيف مرجعا لمشروعك.
Private: ليصبح الإجراء متاحا للاستخدام، فقط من داخل الدالة أو الخليّة Class .

The extends Keyword

تستخدم لعمل وراثة لخصائص و دوال خلايا اخرى سبق تعريفها وهى public مثال يمكن وراثة الخلية MovieClip و استخدام كل خصائصها **فى عمل خلية جديدة و ليكن اسمها** class Drag **و فى هذة الحالة الخلية الجديدة يمكنها استعمال كلا من**
onPress(), onRelease(), startDrag(), and stopDrag(),
و كل ذلك بسبب وراثة الخلية MovieClip class .

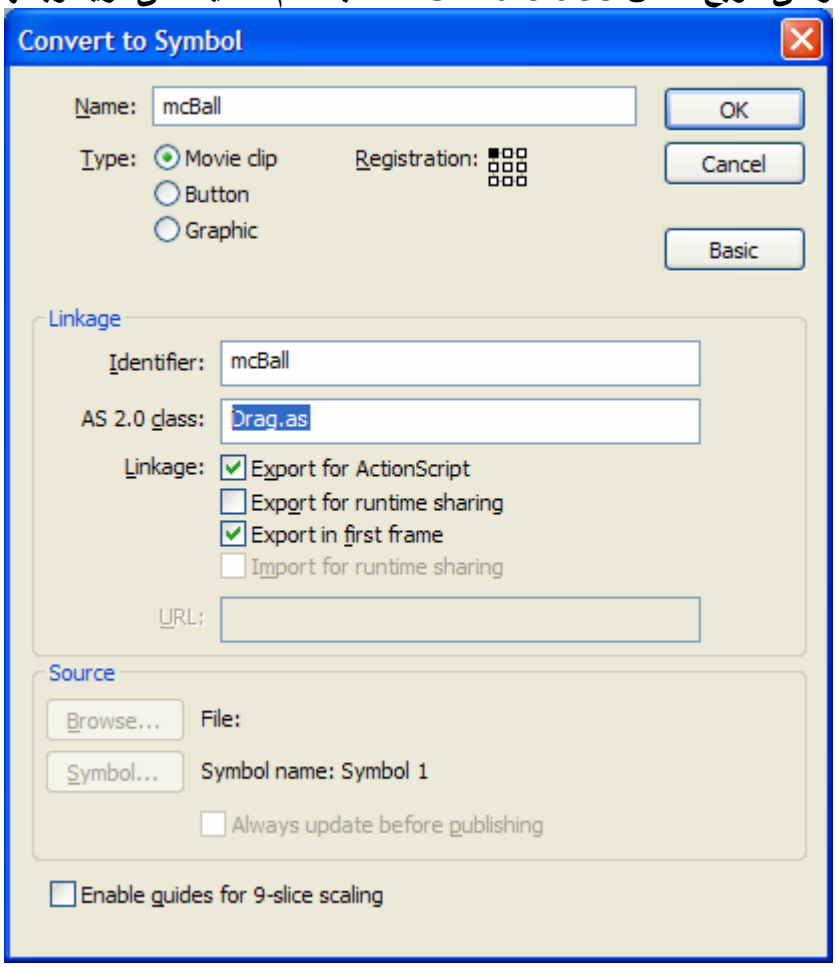
```
classclass Drag extends MovieClip {

    function Drag () {
        onPress = doDrag();
        onRelease = doDrop();
    }
    private function doDrag(){
        this.startDrag();
    }
    private function doDrop(){
            this.stopDrag();
    }
}
```

- **الان يمكن ربط خلية ما بعنصر على مسرح العمل مثل موفى كليب من خلال استخدام خاصية linkage مثال :**
- **مثال : قم بعمل دائرة ثم اضغط f8 لتحويلها لموفى كليب فتظهر لك هذة الصورة :**



فقم باختيار Export for ActionScript

و فى مربع النص AS 2.0 class اكتب اسم الخلية التى تريد ربطها مع هذا الموفى كليب و ليكن مثلا Drag.as

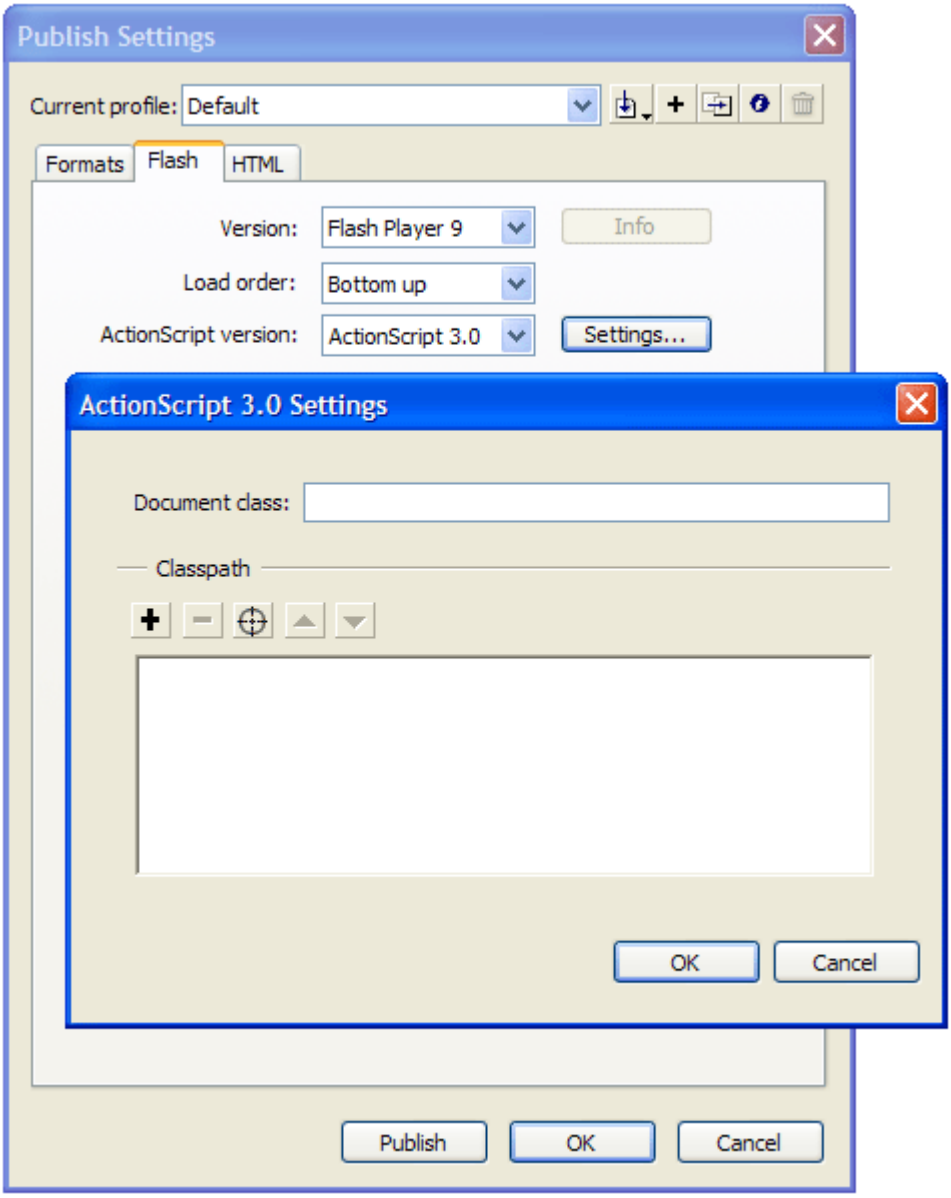


الان عندما تستخدم هذا الموفى كليب سيكون لة كل خصائص الخلية Drag.

موضوعات من الكتاب القادم

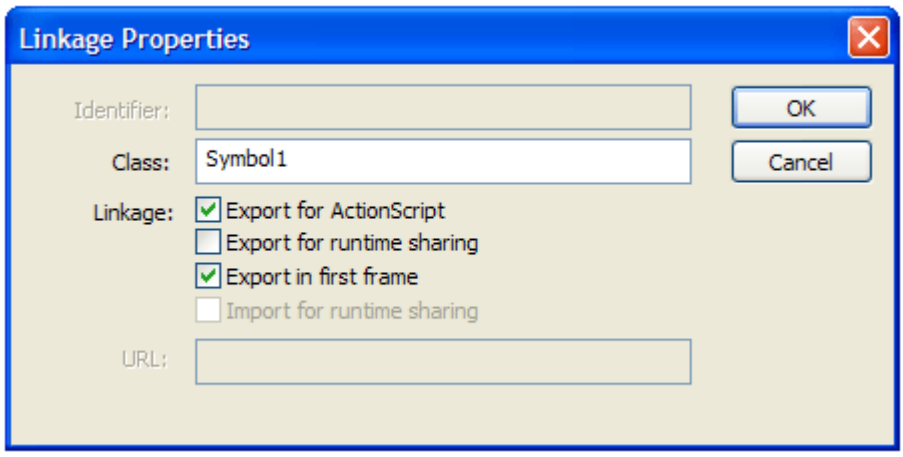
Document class: مفهوم جديد فى فلاش ٩ و معناة تعريف **class** فى الشريط الزمنى لملف الفلاش لكى يقوم ببناء الclass عند تشغيل ملف **swf** و يمكنك عمل ذلك من خلال النافذة **Property inspector** او من خلال مربع الحوار **ActionScript 3.0 Publish** .

- **File > Publish Settings > Flash tab > Settings button** (انظر الشكل).

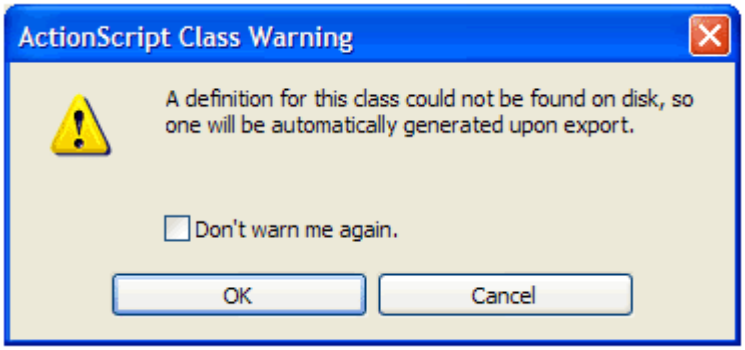


ActionScript 3.0 Settings dialog box

Symbol-class linkage: الان تستطيع ان تستخدم الترابط بين الclass لتقوم بتعريف **instances** من هذا الclass مباشرة.



ربما لا يستطيع الفلاش ايجاد الclass على الهارد ديسك لذلك سيقوم البرنامج بعمل class



مثال : عمل شكل يستجيب للنقر بالفأرة و للسحب .

١- قم بعمل ملف فلاش جديد و قم بتسمية simpleBall.fla ثم قم برسم دائرة على مسرح العمل ثم قم بتحويلها الى موفى كليب واسمة circle و مربع Instance Name قم بأعطاء الموفى كليب اسم و ليكن ball_mc ثم قم بالنقر على الفريم الاول فى الشريط الزمنى و اضغط f9

٢- ثم اكتب الكود التالى :

```
ball_mc.addEventListener(MouseEvent.CLICK, clickHandler);
function clickHandler(event:MouseEvent):void {
  trace("You clicked the ball");
}
```

٣- لقد قمنا بعمل مستجيب للحدث و الحدث هنا هو النقر على الموفى كليب و استخدمنا MouseEvent.CLICK بدلا من onPress() التى كانت مستخدمة فى AS2 .

٤- الان لكى نعطى المستخدم الاحساس بأن الكرة عبارة عن زر نكتب الكود التالى :

```
ball_mc.buttonMode = true;
```

٥- قم بتجربة الملف control+enter .

٦- الان لكى نستطيع سحب و افلات الكرة

سوف نستخدم الاحداث MouseEvent.MOUSE_DOWN، MouseEvent.MOUSE_UP

الكود:

```
ball_mc.buttonMode = true;
ball_mc.addEventListener(MouseEvent.CLICK, clickHandler);
ball_mc.addEventListener(MouseEvent.MOUSE_DOWN,
mouseDownListener);
ball_mc.addEventListener(MouseEvent.MOUSE_UP, mouseUpListener);
function clickHandler(event:MouseEvent):void {
```

```
    trace("You clicked the ball");
  }
  function mouseDownListener(event:MouseEvent):void {
    ball_mc.startDrag();
  }
  function mouseUpListener(event:MouseEvent):void {
    ball_mc.stopDrag();
  }
```

الان قم بتجربة العمل ستجد انك تسطيع تحريك الكرة فى اى مكان على مسرح العمل و تركة اى اصبحت الكرة تتمتع بخاصية السحب و الافلات و لكن تصور انك تريد عمل اشكال كثيرة تتمتع بنفس الخاصية ولا يعقل ان نقوم بتكرار الكود مع كل شكل لكن من الاسهل عمل class تكون و ظيفتها توفير خاصية السحب و الافلات لكل شكل نقوم بعملة فى الفيلم بعد ذلك سنقوم بتحويل الكود السابق لclass .

٧- قم بعمل ملف اكشن اسكربت جديد من خلال

File > New > ActionScript File

ثم قم بحفظة فى نفس المجلد الموجود بة الملف السابق و ليكن اسم الملف Ball.as

ثم اكتب الكود التالى فى ملف الاكشن

```
package {
  import flash.display.MovieClip;
  import flash.events.MouseEvent;
  public class Ball extends MovieClip {
    public function Ball() {
      trace("ball created: " + this.name);
      this.buttonMode = true;
      this.addEventListener(MouseEvent.CLICK, clickHandler);
      this.addEventListener(MouseEvent.MOUSE_DOWN,
mouseDownListener);
      this.addEventListener(MouseEvent.MOUSE_UP, mouseUpListener);
    }
    private function clickHandler(event:MouseEvent):void {
      trace("You clicked the ball");
    }
    function mouseDownListener(event:MouseEvent):void {
      this.startDrag();
    }
    function mouseUpListener(event:MouseEvent):void {
      this.stopDrag();
    }
  }
}
```

٨- الان قم بالعودة لملف الفلاش السابق و ارسم كرة و حولها لموفى كليب ثم اذهب الى مكتبة الملفات و انقر بالزر الايمن للفأرة على الموفى كليب الجديد و اختر الامر Linkage من القائمة ثم اختر Export و فى Class text box اكتب Ball ثم اضغط على ok

٩- الان جرب العمل ستجد ان الدائرة الجديدة اصبحت لها خاصية السحب و الافلات .

Error Messages رسائل الخطأ

يقدم برنامج فلاش مجموعة من رسائل الخطأ و منها التالى :

Error number	Message text
1093	A class name was expected.
1094	A base class name is expected after the 'extends' keyword.
1095	A member attribute was used incorrectly.
1096	The same member name may not be repeated more than once.
1097	All member functions need to have names.
1099	This statement is not permitted in a class definition.
1100	A class or interface has already been defined with this name.
1101	Type mismatch.
1102	There is no class with the name '<ClassName>'.
1103	There is no property with the name '<propertyName>'.
1104	A function call on a non-function was attempted.
1105	Type mismatch in assignment statement: found [lhs-type] where [rhs-type] is required.
1106	The member is private and cannot be accessed.
1107	Variable declarations are not permitted in interfaces.
1108	Event declarations are not permitted in interfaces.
1109	Getter/setter declarations are not permitted in interfaces.
1110	Private members are not permitted in interfaces.
1111	Function bodies are not permitted in interfaces.
1112	A class may not extend itself.
1113	An interface may not extend itself.
1114	There is no interface defined with this name.
1115	A class may not extend an interface.
1116	An interface may not extend a class.
1117	An interface name is expected after the 'implements' keyword.
1118	A class may not implement a class, only interfaces.
1119	The class must implement method 'methodName' from interface 'interfaceName'.
1120	The implementation of an interface method must be a method, not a property.
1121	A class may not extend the same interface more than once.
1122	The implementation of the interface method doesn't match its definition.
1123	This construct is only available in ActionScript 1.0.

Error number	Message text
1124	This construct is only available in ActionScript 2.0.
1125	Static members are not permitted in interfaces.
1126	The expression returned must match the function's return type.
1127	A return statement is required in this function.
1128	Attribute used outside class.
1129	A function with return type Void may not return a value.
1130	The 'extends' clause must appear before the 'implements' clause.
1131	A type identifier is expected after the ':'.
1132	Interfaces must use the 'extends' keyword, not 'implements'.
1133	A class may not extend more than one class.
1134	An interface may not extend more than one interface.
1135	There is no method with the name '<methodName>'.
1136	This statement is not permitted in an interface definition.
1137	A set function requires exactly one parameter.
1138	A get function requires no parameters.
1139	Classes may only be defined in external ActionScript 2.0 class scripts.
1140	ActionScript 2.0 class scripts may only define class or interface constructs.
1141	The name of this class, '<A.B.C>', conflicts with the name of another class that was loaded, '<A.B>'. (This error occurs when the ActionScript 2.0 compiler cannot compile a class because of the full name of an existing class is part of the conflicting class' name. For example, compiling class `mx.com.util` generates error 1141 if class `mx.com` is a compiled class.)
1142	The class or interface '<Class or Interface Name>' could not be loaded.
1143	Interfaces may only be defined in external ActionScript 2.0 class scripts.
1144	Instance variables cannot be accessed in static functions.
1145	Class and interface definitions cannot be nested.
1146	The property being referenced does not have the static attribute.
1147	This call to super does not match the superconstructor.
1148	Only the public attribute is allowed for interface methods.
1149	The import keyword cannot be used as a directive.
1150	You must export your Flash movie as Flash 7 to use this action.
1151	You must export your Flash movie as Flash 7 to use this expression.
1152	This exception clause is placed improperly.
1153	A class must have only one constructor.
1154	A constructor may not return a value.
1155	A constructor may not specify a return type.

Error number	Message text
1156	A variable may not be of type Void.
1157	A function parameter may not be of type Void.
1158	Static members can only be accessed directly through classes.
1159	Multiple implemented interfaces contain same method with different types.
1160	There is already a class or interface defined with this name.
1161	Classes, interfaces, and built-in types may not be deleted.
1162	There is no class with this name.
1163	The keyword '<keyword>' is reserved for ActionScript 2.0 and cannot be used here.
1164	Custom attribute definition was not terminated.
1165	Only one class or interface can be defined per ActionScript 2.0 .as file.
1166	The class being compiled, '<A.b>', does not match the class that was imported, '<A.B>'. (This error occurs when a class name is spelled with a different case from an imported class. For example, compiling class `mx.com.util` generates error 1166 if the statement `import mx.Com` appears in the util.as file.)
1167	You must enter a class name.
1168	The class name you have entered contains a syntax error.
1169	The interface name you have entered contains a syntax error.
1170	The base class name you have entered contains a syntax error.
1171	The base interface name you have entered contains a syntax error.
1172	You must enter an interface name.
1173	You must enter a class or interface name.
1174	The class or interface name you have entered contains a syntax error.
1175	'variable' is not accessible from this scope.
1176	Multiple occurrences of the 'get/set/private/public/static' attribute were found.
1177	A class attribute was used incorrectly.
1178	Instance variables and functions may not be used to initialize static variables.
1179	Runtime circularities were discovered between the following classes: <list of user-defined classes>. This runtime error indicates that your custom classes are incorrectly referencing each other.
1180	The currently targeted Flash Player does not support debugging.
1181	The currently targeted Flash Player does not support the releaseOutside event.
1182	The currently targeted Flash Player does not support the dragOver event.
1183	The currently targeted Flash Player does not support the dragOut event.
1184	The currently targeted Flash Player does not support dragging actions.
1185	The currently targeted Flash Player does not support the loadMovie action.
1186	The currently targeted Flash Player does not support the getURL action.

Error number	Message text
1187	The currently targeted Flash Player does not support the FSCommand action.
1188	Import statements are not allowed inside class or interface definitions.
1189	The class '<A.B>' cannot be imported because its leaf name is already resolved to the class that is being defined, '<C.B>'. (For example, compiling class `util` generates error 1189 if the statement `import mx.util` appears in the util.as file.)
1190	The class '<A.B>' cannot be imported because its leaf name is already resolved to a previously imported class '<C.B>'. (For example, compiling `import jv.util` generates error 1190 if the statement `import mx.util` also appears in the AS file.)
1191	A class' instance variables may only be initialized to compile-time constant expressions.
1192	Class member functions cannot have the same name as a superclass' constructor function.
1193	The name of this class, '<ClassName>', conflicts with the name of another class that was loaded.
1194	The superconstructor must be called first in the constructor body.
1195	The identifier '<className>' will not resolve to built-in object '<ClassName>' at runtime.
1196	The class '<A.B.ClassName>' needs to be defined in a file whose relative path is <'A.B>'.
1197	The wildcard character '*' is misused in the ClassName '<ClassName>'.
1198	The member function '<classname>' has a different case from the name of the class being defined, '<ClassName>', and will not be treated as the class constructor at runtime.
1199	The only type allowed for a for-in loop iterator is String.
1200	A setter function may not return a value.
1201	The only attributes allowed for constructor functions are public and private.
1202	The file 'toplevel.as', which is required for typechecking ActionScript 2.0, could not be found. Please make sure the directory '$(LocalData)/Classes' is listed in the global classpath of the ActionScript Preferences.
1203	Branch between <spanStart> and <spanEnd>> exceeds 32K span.
1204	There is no class or package with the name '<packageName>' found in package '<PackageName>'.
1205	The currently targeted Flash Player does not support the FSCommand2 action.
1206	Member function '<functionName>' is larger than 32K.
1207	Anonymous function around line <lineNumber> exceeds 32K span.
1208	Code around line <lineNumber> exceeds 32K span.
1210	The package name '<PackageName>' cannot also be used as a method name.
1211	The package name '<PackageName>' cannot also be used as a property name.
1212	The ASO file for the class '<ClassName>' could not be created. Please make sure the fully-qualified class name is short enough so that the ASO filename, '<ClassName.aso>', is less than 255 characters.
1213	This type of quotation mark is not allowed in ActionScript. Please change it to a standard

Error number	Message text
	(straight) double quote.

ملحوظة : هذا الملحق منقول من موقع http://www.c4arab.com

للكاتب : عماد عدلي

الموسوعة العربية للكمبيوتر و الانترنت

ملحق عن لغة الXml

وماذا عن لغة XML؟

إنّ XML هي طريقة لتمثيل أيّ بيانات منظّمة، وذلك بتحويلها لنصّ يعبّر عنها.. وبهذا تصلح لغة XML للتعبير عن أيّ نوع من أنواع البيانات، مثل الجداول والصور وغيرهما.
ورغم أنّ الملفّات النصّيّة أكبر حجما من الملفّات الثنائيّة Binary Files، إلا إنّ الأولى صالحة للتعامل مع أيّ تطبيق بل مع أيّ نظام تشغيل.. لهذا فقد صارت لغة XML في السنوات الأخيرة هي أنسب وسيلة لنقل البيانات عبر الإنترنت، وذلك حتّى تتجاوز مشاكل عدم التوافق بين التطبيقات وأنظمة التشغيل المختلفة.

متطلبات العمل مع XML .

جميع ما تحتاجه إلى استخدام XML موجود في معظم الحواسب .

- برنامج تحرير نصوص عادى مثل المفكرة (Notepad) .

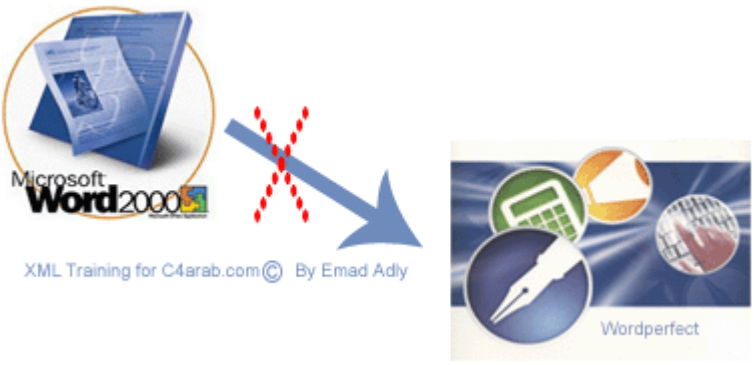
- متصفح يمكنه استعراض مستندات XML مثل Internet Explorer 5.5 أو احدث .

قبل التعرف على تقنيات XML والدافع إلى تعلمها نلقى نظرة على هيئة البيانات والملفات والنصوص ولغات الترميز الأخرى وتاريخها .

فلكي نفهم لغة الترميز يجب علينا فهم كيفية حفظ البيانات وكيفية الوصول لها . فهناك نوعان رئيسان من الملفات التي يفهمها الحاسب .الملفات النصية والملفات الثنائية.

أولا : الملفات الثنائية :-

وهى عبارة عن سلسلة من البتات (١-٠) صفر و واحد ويتم التعرف عليها وفهمها بواسطة التطبيقات التي أنشأتها . ونلاحظ ذلك في عدم القدرة على قراءة ملف ما إلا بواسطة التطبيق الذي صنع لأجله فمثلا لو أرت فتح مستند مكتوب بواسطة برنامج معالج كلمات ما ولنفرض Word Perfect على برنامج Microsoft Word فان الملف لم يفتح أو أنة يفتح بشكل غير مناسب وذلك لأنة لم يعد للعمل على هذا التطبيق بالاختلاف انه صمم لكي يناسب العمل على تطبيق أخر . ولحسن الحظ فان معظم تطبيقات معالجة الكلمات اليوم تحتوى على محولات تستطيع فتح وقراءة الملفات التي تم إنشاءه على معالجات أخرى .



أن الميزة الجيدة للملفات الثنائية هي سهولة فهم الشفرات الثنائية من قبل الحاسب ، بما أن بنية الحاسب هي أصلا بنية ثنائية تعتمد على الوحدات والأصفار فان قراءة الملفات الثنائية ستكون أسرع من قراءة هيئات الملفات الأخرى .

وكما ذكرنا السيئة الوحيدة هي عدم القدرة على فتح تطبيق تم بناءة على تطبيق أخر . وقد يصل الأمر إلى عدم إمكانية فتح وقراءة الملف في نفس التطبيق ولكن ضمن منصة تشغيل Platform مختلفة أو ضمن إصدارة سابقة لنفس التطبيق.

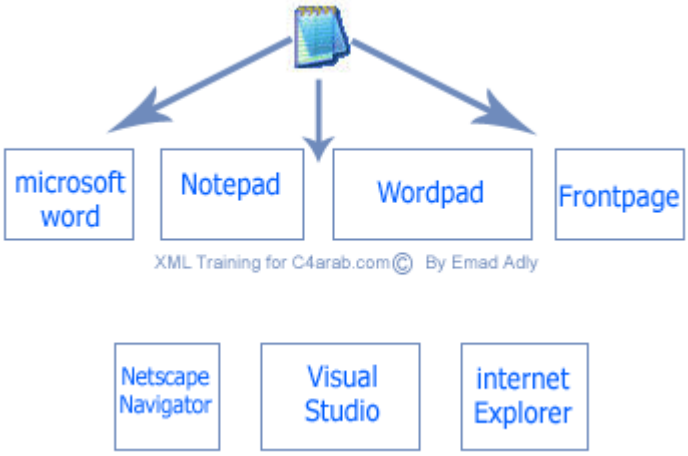
ثانيا : الملفات النصية :-

الملفات النصية تشبه أيضا الملفات الثنائية . فهيا أيضا تمثل سلسلة من البتات (٠ - ١) صفر و واحد ولكنها تعمل مع بعضها بصورة قياسية بحيث تشكل أرقاما على سبيل المثال .

١١٠٠٠٠١

هذه المجموعة من البتات تترجم بحيث تمثل الرقم ٩٧ والذي يتم أيضا بحيث يمثل الحرف (a) .

وينتج عن هذه الشفرات القياسية أن الملفات النصية يمكن قراءتها من قبل معظم التطبيقات هذا بالإضافة إلى إمكانية قراءتها من قبل البشر .



إذا قمنا بكتابة ملف نصي فان إي شخص في العالم يستطيع فهم لغة النص ويمكن قراءته بأي محرر نصوص يفضله . فهو يساعد على مشاركة وتبادل المعلومات مع الآخرين .

السيئة الوحيد لهذا النوع من الملفات هو أننا لا نستطيع تهيئة النص المكتوب بالطريقة التي نرغب بها كما في الملفات الثنائية فلا نستطيع توسيط النص مثلا أو تلوينه بلون معين أو نجعل النص مائلا .

تاريخ لغات الترميز :-

لقد لاحظنا ميزات الملفات الثنائية ، فهي سهلة الفهم بالنسبة للحاسب ولاحظنا أيضا ميزات الملفات النصية فهي قابله للتشارك وتبادل المعلومات بشكل واسع أليست صفقة رابحة لنا أن توجد هناك هيئة ملفات تجمع بين قابلية تبادل المعلومات الواسعة التي توفرها الملفات النصية بالإضافة لفعالية وإمكانيات التخزين القوية التي توفرها الملفات الثنائية .

أن هذه الفكرة ليست جديده . فمنذ أن توفرا الحاسب أما المبرمجين وهم يبحثون عن طرق لتبادل المعلومات بين مختلف الأنظمة والبرامج وكخطوة أولية لذلك ظهرت لغة الترميز المعممة القياسية SGML وهى اختصارا إلى

Standard Generalized Markup Languge كانت تلك عبارة عن لغة نصية تستخدم لترميز البيانات . مثل الملفات الثنائية ولكن بطريقة تشرح نفسها بنفسها وهو ما يعرف بالملفات ذاتية الوصف Self Describing وهذه اللغة احتلت موضعا مرموقا في العديد من أنظمة إدارة البيانات الكبيرة . فهيا تحتاج إلى الكثير من الاعتبارات عن ترميز البيانات الضخمة ونتيجة إلى ذلك أصبحت لغة معقدة جدا ولكن من التعقيد تأتى القوة .

مقدمة إلى تقنية XML وما حولها

لغة الترميز الموسعة eXtensible Markup Language التي يرمز لها بالاختصار XML وهى تستخدم في وصف وتخزين وتنظيم البيانات بخلاف لغة HTML التي تستخدم لكيفية عرض البيانات على المتصفح .

تحدثنا سابقا عن لغة SGML وذكرنا إنها لغة معقدة لا تصلح لتبادل المعلومات عبر الشبكة . وعلى الرغم من أنها لغة لوصف البيانات على المتصفح ولا يمكن استخلاص معلومات معينة حول شخص ما ولسبب قصور لغة HTML في قدرتها على وصف أنواع محددة من المعلومات دعت الحاجة لوجود لغة تحقق ذلك وهى لغة الترميز الموسع أو XML وهى لغة أيضا مشتقة من لغة SGML ومتوافقة بصورة كبيرة مع هذه اللغة. هذا يعنى إن أي مستند يتبع مصطلحات وتعابير لغة XML فأنة يتبع أيضا مصطلحات وتعابير لغةSGML .

وهنا يجب ملاحظة شي هام أن لغة XML ليست لغة في الأصل فا XML تصف مجموعة من التعابير التي تستخدمها لبناء لغاتك الخاصة على سبيل المثال لنفترض أن لدينا بيانات حول اسم شخص ما وانك تريد تبادل هذه البيانات مع الآخرين يمكنك تمثيل هذه البيانات في ملف نصي بالصورة التالية.

Emad Adly

أو بصيغة HTML بهذا الشكل .

```
<html>
<head><title>Name</title></head>
<body>
<p>Emad Adly</p>
</body>
</html>
```

يمكن تمثيل هذه البيانات في XML بالشكل التالي .

```
<name>
   <first>Emad</first>
   <last>Adly</last>
</name>
```

ذكرنا سابقا أن لغة SGML وXML تسميان بالغات ذاتية الوصف لان البيانات يمكنها بسهولة معرفة أن هذه المعلومات تمثل اسم Name لشخص ما وأيضا هناك بيانات تسمى <first>و بيانات أخرى تسمى <last> يجب آن تكون ذات معنى طبعا المعنى يدل على محتوى المعلومة بداخلها .

لو قمنا بحفظ الملف السابق باسم name.xml فيمكنا فتح هذا الملف بواسطة متصفح الإنترنت لديك بشرط أن لا يقل عن ٥,٥ وسوف يظهر بهذا الشكل .

```
- <name>
    <first>Emad</first>
    <last>Adly</last>
  </name>
```

وبالرغم من أن ملف XML السابق لا يحتوى على إي معلومات حول كيفية العرض فان المتصفح قام باستعراض الملف بصورة لطيفة وبتنسيق لونى مختلف وأيضا البنية الشجرية التي فهمها المتصفح وترجمها أيضا وذلك بالنقر على الرمز (-) بجانب البند <name> وهذه الطريقة مفيدة جدا عندما يكون الملف كبير الحجم .

ونلاحظ أننا لم نقم بوصف البيانات لكي تظهر بهذا التنسيق الموجود ولكن هذا ما يقدمه لنا متصفح الإنترنت فلهذا المتصفح ورقة تنسيق Style Sheet افتراضية مبيتة داخلة مما يمكن المتصفح من عرض إي مستند XML وفق ورقة التنسيق هذه .

وبهذا قد عرفنا مقدمة بسيطة إلى هذه اللغة وفى الدرس القادم سوف نتحدث عن الأسباب التي تدفعنا لتعلم هذه اللغة .

والجدير بالذكر أن لغة XML تتطلب منا بعض القوانين المحددة لكتابه وثائق XML قابلة للعرض . مثلها مثل لغة HTML فيها أيضا تتطلب دراية بطريقة كتابه الـ Tags الخاص بها .

معربات لغة XML Parsers :-

إذا اتبعنا القوانين المحددة وفق لغة XML يمكننا الوثوق من سهولة استخلاص المعلومات . يعود ذلك إلى وجود برمجيات تسمى بالمعربات Parsers وظيفتها قراءة عبارات XML واستخلاص المعلومات من تلك العبارات .

وهى تستخدم في برامجنا للتعامل مع مستندات XML .

فلا داعي للقلق حول كيفية استخلاص المعلومات من ملفات XML. فاليوم افضل من الأمس . في الماضي وقبل استخدام هذه المعربات كان يتحتم عليك القيام بالكثير من العمل لبناء قوانين تحكم هذه المعلومات ولكن الآن مع صيغة XML يمكنك فقط إعطاء معرب لغة XML ملفا كما يلي :-

```
<name>
   <first>Emad</first>
   <last>Adly</last>
</name>
```

سيقوم المعرب باستخلاص البيانات من هذه الشفرة وسيخبرنا بأن هناك بندا للبينات باسم وأن المعلومة المرفقة لهذا البند هي Emad ولا يتحتم على معرب XML معرفة أي قوانين حول موضع الاسم من البيانات .

أن الشفرة المكتوبة بلغة XML مهما كان نوعها لا تؤثر في عمل المعرب فان كنت كتبت شفرة XML بالإنجليزية أو بالعربية أو بأية لغة أخرى فجميعها يمكن قراءتها وفهمها بنفس المعرب . وحتى إذا كان الشخص الذي كتبها لا يعرف شي عن هذه اللغة .

لغة XML لغة مرنة بدرجة كبيرة فهي تهدف إلى أن تكون الأساس للغات تبادل المعطيات المختلفة خاصة عبر الإنترنت فهي تجعل من تشارك الملفات والمعلومات على الإنترنت وتبادلها مسالة سهله جدا.

الترميز الموسع Extensible :-

باعتبار أنة يمكننا التحكم بصورة كاملة في تكوين مستندات XML فيمكننا أن نشكل البيانات بالطريقة التي تحلو لنا .

أو قررنا عدم احتياجنا لمرونة اكبر في المثال السابق فنقوم بكتابة الشفرة بالطريقة التالية .

```
<name>Emad Adly</name>
```

أنت حر في طريقة تمثيل البيانات . يتوقف ذلك على الطريقة التي تمكن برامجنا من استخدام هذه البيانات . أما إذا أردت استخدام المرونة المتوافرة xml فيمكنك القيام بذلك . فاختار ما يناسب احتياجاتك .

من هنا جاءت تسمية هذه اللغة بالترميز الموسع أو Extensible فان أي شخص يستطيع ترميز البيانات نفسها بأي طريقة باستخدام هذه اللغة .

ولكن يجب أن نأخذ بالحسبان أنه تكمن الاستفادة الحقيقية من لغة XML عندما يستخدم الأشخاص نفس الهيئة لتمثيل البيانات الشائعة لان ذلك سيسمح بتبادل المعلومات بصورة اكبر واسهل .

هناك فعلا العيد من المشاريع للوصول إلى مفردات قياسية لأنواع البيانات الشائعة الاستخدام . فعلا سبيل المثال لغة الرسوميات الشعاعية القابلة للتوسع Scalable Vector Graphics أو SVG وهى مفردات XML التي تمثل قاعدة أساسية لتمثيل الرسوميات ثنائية البعد .

ولغة MathMLوهى مفردات XML لوصف الرياضيات كقاعدة أساسية لاتصال الآلات ببعضها البعض .

ولغة الترميز الكيميائية Chemical Markup Languageأو CML هي مفردات XML لأداره المعلومات الكيميائية

وأيضا لغة WML وهى لغة ترميز اللاسلكي والمستخدمة في بناء مواقع يمكن تصفحها باستخدام الهواتف الخلوية

وهناك الكثير من المفردات المتخصصة في مجالات أخرى تجعل من كتابة البيانات باستخدام XML خاصة بنا إلا إن استخدام المفردات القياسية يساعدك على زيادة توافقية مستندات XML مع البرمجيات الأخرى.

مما تتكون لغة XML ؟

تعتبر تقنية XML عائلة خاصة بذاتها بعضها ما يزال في قيد التطوير في سنواته الأولى .

ولكل فرد في هذه العائلة موصفاته الخاصة . النسخة XML 1.0 هي القاعدة الأساسية التي تبنى عليها XML فهي تصف التراكيب التي يجب على مستند XML اتباعها وكذلك القوانين التي يجب على معربات XML تطبقها بالإضافة إلى تعاريف أنواع المستندات DTDs.

لغة Xpathوهى أيضا جزء لا يتجزأ من تقنية XML وهى لغة الاستعلام لعنونة الأقسام في مستند XML فهي تمكن التطبيقات من إيجاد معلومات محدده ضمن مستند XML .

تدخل لغة CSSضمن عائلة XML أيضا وذلك في حالات استعراض مستندات XML البسيطة ويمكن العوض عنها باستخدام لغة XSL وذلك في الحالات المعقدة وهى تتضمن تحويلات خاصة تسمى هذه التحويلات XSLT والتي تستخدم لتحويل مستندات XML إلى أنواع مستندات أخرى .بإضافة إلى أسلوب عرض المعلومات .

الأختان XLink و XPointer هما لغتان تستخدما لربط مستندات XML مع بعضهما البعض بصورة مشابها للوصلات التشعبيه في مستندات HTML .

نموذج كائن المستند Document Object Model أو DOM هذا النوع من عائلة XML هو معروف جيداً لمن تعامل من قبل مع لغة DHTML و JavaScript ألم تمر عليك جملة Document.write قبل ذلك .

فهذا الكائن يمكنه ربط مستندات XML مع لغات برمجة أخرى مع إمكانية الإضافة والحذف التعديل داخل مستندات XML بواسطة لغتك المفضلة

لماذا XML وفيما تستخدم؟

تقنية XML هي تقنية حديثة وفى دور التطوير حاليا وهى كما عرضنها سابقا في هذه السلسلة وعرفنا ما هي XML .سوف نوضح هنا ما الذي تقدمه لنا هذه اللغة .

تعرفنا سابقا أن لبناء مستندات XML يجب علينا معرفة أشياء وتعليمات كثيرة . فلماذا نستخدم كل هذه الأمور . أليس من السهل وضع بعض القوانين لمثال مستند الأسماء السابق شرحه في الدرس السابق . ونضع بعض المعايير لذلك فمثلا يمكننا القول أن الاسم الأول يبدأ عند بداية الملف ، والاسم الأخير يأتي بعد الفراغ الأول . بهذه الطريقة يمكن لتطبيقنا أن يقرا بيانات الاسم الأول بصورة منفصلة عن الاسم الأخير .

ولتوضيح ذلك : لنفترض إننا نريد إضافة الاسم الأوسط لمثالنا فيصبح هكذا .

Emad Adly Faik

وأيضا عند إضافة هذه العملية لا يوجد إي مشكلات . يمكننا تعديل القانون السابق بحيث إن كل شي يأتي بعد الفراغ الأول وقبل الفراغ الثاني يمثل الاسم الأوسط ، وما يأتي بعد الفراغ الثاني يمثل الاسم الأخير .

إلى الآن يمكننا تمثيل البيانات باستخدام هذا القانون وتنفيذه برمجيا ، ولكن ماذا لو قابلنا اسم كالتالي .

Emad Adly Faik Gabala

الآن لدينا اسمين أوسطين . هنا يصبح القانون اكثر تعقيدا وعلما إن البشر تستطيع تميز الاسم الأوسط من الاسم الأول من الاسم الأخير بسهولة فانه من الصعب برمجة ذلك بطريقة منطقية بحيث يمكن الحاسب من تميز ذلك بسهولة .

للأسف فان معظم المبرمجين عندما يواجهون مشكله من هذا النوع يتصرفون لوضع قوانين صارمة بدلا من التعامل مع البيانات المعقدة كما هي . فالمشكلة الأسماء كما سبق يمكن لمطوري البرامج أن يقرروا أن للشخص الواحد اسم أوسط واحدا فقط ولا يمكن للتطبيق قبول أكثر من ذلك .

فالمثال السابق ليس صعب التطبيق ولكن هو يسلط الضوء حول الأسباب الجوهرية التي أدت لظهور لغة XML.
فالمبرمجين يستطيعون تنظيم بياناتهم بعدد غير محدد من الطرق . وفى كل طريقة من الطرق نظام معين لاستخلاص المعلومات التي نحتاجها باستخدام لغة XML

إلى هنا قد ذكرنا مثال بسيط جدا لتوضيح ابسط الأشياء التي تدفعنا كمطورين إلى الدخول إلى عالم XML .

وحتى إن كان هذا المثال غير مفهوم للمرة الأولي فلا تقلق سيمكنك التميز والتوسع اكثر عند البدء في تشغيل وكتابة مستندات XML.

كما يمكن للغة XML العمل على إي منصة تشغيل فهيا ليست مخصصه للعمل عن نظام عين مثل في Visual Basic مخصص للعمل على منصات ويندوز . فيمكنك استخدامها على إي نظام تشغيل حالي أو سوف يظهر مستقبلا .

فمن الأشياء الرائعة عند استخدامك لـ XML هو تقليص زمن التحميل على خادمات الشبكة Web Server بالإضافة إلى عمليات إرسال واستقبال كميات ضخمة من المعلومات بين جهاز الزبون والخادم .

وأيضا من الميزات الجميلة في التعامل مع ملفات XML هي إمكانية تحويل ملفاتXML إلى ملفات html بسهوله لعرضها بشكل جميل على المتصفح وذلك بواسطة محولات XSLT . أو أن تعرض مباشرة عبر المتصفح بواسطة ورقة التنسيق css كما في المثال الأول .

استدعاء الإجراءات البعيدة . فهيا تستخدم بروتوكول RPC لذلك فهو يسمح للكائنات الموجودة على جهاز ما باستدعاء الكائنات الموجودة على جهاز أخر للقيام بعمل ما .

أما في مجال التجارة الإلكترونية أحد المواضيع الساخنة حاليا في الإنترنت . التي ستظل ساخنة لفترة طويلة .فمعظم الشركات اليوم تقر بضرورة ربط عملائها معها عبر الإنترنت بد من استخدام الطرق التقليدية .

فكلما احتاجت شركة ما لإرسال بياناتها إلى جهة أخرى عبر الإنترنت فان xml هي التقنية الأفضل لذلك باعتبار أن الهدف من وراءها هو تبادل المعلومات والتشارك عليها .

فهناك الكثير والكثير من الأماكن التي تقدم لنا فيها تقنية XML الفائدة التي نرجوها.

منهجية عمل مستندات XML وتكوينها الهرمي

سنناقش اليوم التركيب الهرمي الذي يكون مستندات XML وكيفية بناء البيانات وكيفية تمثيلها في مستند XML

- عندما نكون أمام كمية معلومات ضخمة أو حتى متوسطة الحجم فمن الأفضل تجميع هذه المعلومات التي تنتمي إلى موضوع واحد مع بعضها البعض بدل من تمثيلها كما هي .

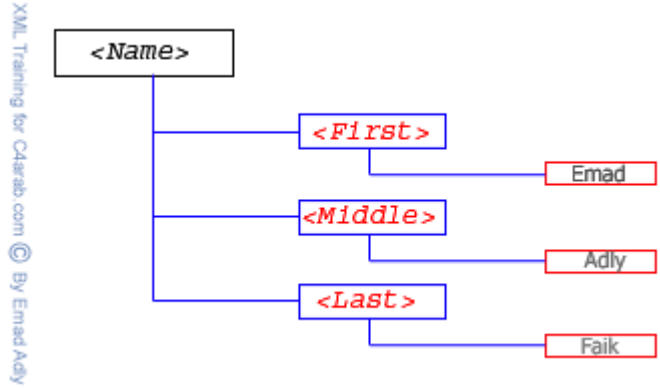
نعطى مثال لكى توضح الصورة .

مثلا هذه المنتدى مجزأ إلى مجموعة من الأقسام والمواضيع الفرعية التي تتبع موضوعات فرعية والتي تتبع بدورها مواضيع رئيسيه بالإضافة إلى تقسيم الموضوع الواحد إلى مجموعة من الفقرات وهى عبارة عن ردود الأعضاء . فذلك يسهل من عملية استخراج المعلومات .

فنجد مطوري البرمجيات تستخدم هذا النموذج منذ سنوات عدة باستخدام بنية بينات تسمى **Object model** أو نموذج الكائن وهى مرتبطة مع بعضها بتسلسل هرمي . أيضا في لغة **XML** تجمع البينات في تسلسل هرمي فالبنود في المستند تتبع بعضها البعض بعلاقات **Parent / Child** أو الأب / الابن .

وهذه البنود تسمى بالعناصر **elements** وهى أجزاء منفردة من المعلومات .

نأخذ مثال الاسم السابق شرحه ونمثله بطريقة هرمية كالشكل التالي.



نلاحظ أن البند **<Name >** هو أب للبند **<First >** والبند **<First >** هو ابن للبند **<Name >** والبنود **<First >** و**<Middle >** و **<Last >** جميعها انساب لبعضها البعض لان جميعهم أبناء للبند **<Name >**

ونلاحظ أيضا أن النص هو ابن للعنصر الذي ينتمي له فالنص **Emad** يمثل ابنا للبند **<First >** .تسمى هذه البنية من البينات بالشجرة **Tree** فكل جزى من الشجرة يحتوى على أبناء تسمى بالفروع **Branches** وجميع الأجزاء التي تحتوى على أبناء تسمى بالأوراق **Leaves**

إذا نقول :

Element Content

العنصر **<Name >** يعتبر **element content** لان هناك عناصر تنتمي له وليس مجرد نص فإنه يعتبر محتوى عنصر .

Simple Content

العنصر <First > و <Middle >و<Last > هي محتوى بسيط Simple Content لأنها تحتوى على نص فقط .

Mixed Content

أيضا يمكن للعناصر أن تحتوى على عناصر أخرى وعلى نصوص في تلك الحالة فان للعناصر تلك محتوى مختلط Mixed Content على سبيل المثال .

```
<doc>
    <parent>Computer<em>4</em>arab</parent>
</doc>
```

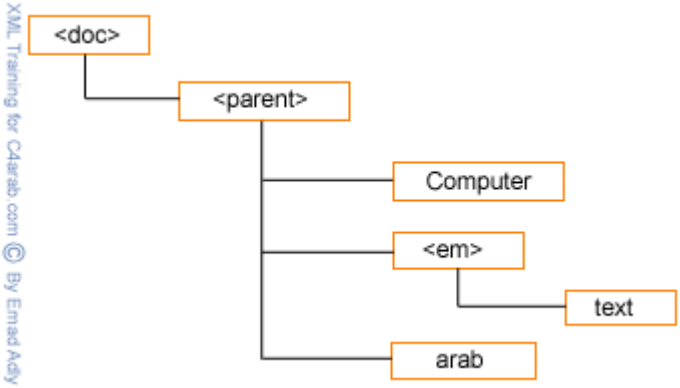
نلاحظ في المثال هذا أن للبند <parent > ثلاث أبناء

نص ،، يحتوى على النص Computer

عنصر ،، وهو عنصر ابن أيضا

نص ،، نص أخر يحتوى على النص arab

فيمكننا ألان تمثيل هذا المثال ببنية شجرية هكذا .



نرجو أن تكونوا تفهمتم كيفية التمثيل الهرمي للبنود والعلامات فيما بينهما فأنك حينما تفهم ذلك ستتمكن من فهم طبيعة لغة XML

منهجية عمل XML وقواعدها :-

١) اللواحق والنصوص والعناصر Tags and Text and Elements

اللاحقة أو ما يطلق عليها البعض والوسم (Tag) هي عبارة عن كلمة أو مصطلح موضوع بين رمزي إحاطة < > يمثل رمزا معرفا لتنسيق ما وذلك في مستندات HTML بينما يمثل اسما لعنصر Elements في مستندات XML

مثل ....

```
<name>
    <first>Adel</first>
    <last>Maher</last>
</name>
```

وكما تلاحظ فان الـ Tag تأخذ الطابع الزوجي فكل لاحقة لها لاحقة مقابله لها تعرف الأولى لاحقة البداية Start Tag وتعرف الثانية بلاحقة النهاية End Tag . الاختلاف بين الاثنين هو أن لاحقة النهاية تحتوى على الرمز "/" .

في XML جميع المعلومات الموجودة بين لاحقة البداية ولاحقة النهاية نسمى بالعناصر Element وبالتالي فان .

<first > هي لاحقة بداية

</first > هي لاحقة نهاية

<firest>Adel< /first > هو عنصر

Element Content
النص الواقع بين لاحقة البداية ولاحقة النهاية يسمى بمحتوى العنصر Element content

PCDATA
المحتوى الواقع بين لاحقتين عبارة عن بيانات ويعرف في هذه الحالة ببيانات الرمز المعرب PCDATA وذلك إذا احتوى هذه العنصر على معلومات نصية مثل العنصر <middle > فهوPCDATA

Root Element
المستند ككل بدء باللاحقة <name > وانتهاء باللاحقة</name > فهو يمثل عنصر يحتوى على مجموعة من العناصر وهنا نطلق عليه عنصر الجذرRoot Element

قوانين العناصر

يجب على مستندات XML الخضوع لهذه القوانين كي تشكل فعليا مستندات XML محكمة الهيئة Well-formed XML Documents

- لكل لاحقة بداية لاحقة نهاية مماثلة لها .
- لا يمكن للواحق أن تتداخل .
- يحتوى مستند XML على عنصر جذر واحد فقط.
- لغة XML حساسة لحالة الحروف Case-Sensitive

- لغة XML لا تتجاهل المساحات الفارغة في مستنداتها .

أسماء العناصر :-

لغة XML توفر لك الحرية في تسمية العناصر فهي لا تحتوى على أسماء محجوزة كما في معظم اللغات ، فهي لديها مرونة كبيرة في اختيار الأسماء . ولكن يوجد مجموعة من القوانين التي يجب مراعاتها :-

- يمكن للأسماء أن تبدأ بأحرف لاتينية أو غير لاتينية أو أن تبدأ بالرمز underscore(_) ولكن لا يمكن أن تبدأ برقم أو بعلامة ترقيم .
- بعد الحرف الأول يمكن للأسماء أن تحتوى على أرقام بالإضافة إلى الرمزين " _ " و " . "
- لا يمكن للأسماء أن تحتوى على فراغات .
- لا يمكن للأسماء أن تحتوى على ":" فهو محجوز في XML
- لا يمكن للأسماء أن تبدأ بالأحرف XML سواء كانت بأحرف صغيرة أو كبيرة .
- لا يمكن أن يكون هناك فراغ بين قوس الإحاطة المفتوح < وبين اسم العنصر

٢) الصفات Attributes

أن مستندات XML يمكن أن تتضمن صفات أو سمات معينة attributes الصفات عبارة عن اسم معين تسند له قيمة معينة بحيث يرتبط ذلك الاسم وتلك القيمة بعنصر معين في مستند XML .

مثل ..

```
<name nickname='mrScript'>
    <first>Emad</first>
    <last>Adly</last>
</name>
```

يجب أن تحتوى الصفات على قيم ويجب أن تكون هذه القيم واقعة بين علامتي اقتباس ولا يشترط أن تكون علامة الاقتباس مفردة أو مزدوجة .

يمكن للصفات أن تقدم بيانات وصفية Metadata والتي يمكن أن لا تكون وثيقة الصلة بمعظم التطبيقات التي تتعامل مع المستندات XML

على سبيل المثال إذا علمنا أن بعض التطبيقات يمكن أن تهتم بالاسم المستعار Nickname ولكن معظم التطبيقات لا تهتم بهذه المعلومات فان استخدام هذه المعلومات كصفة سيكون ذا معنى .

إذا ما الذي تقدمه الصفات ولا يمكن للعناصر أن تقدمه .

مثال ..

```
<Name Nicname='MrScript'></Name>
```

أيضا ،،،

```
<Name>
<Nickname>MrScript<Nickname>
</Name>
```

السبب يرجع إلى أن بعض الأشخاص يجدون أن استخدام الصفات تسهل على سبيل المثال فانك لا تحتاج إلى الاهتمام بدرجة تعشيش العناصر ولا تحتاج للقلق حول المعرفات المتداخلة.

فيما أن البعض الآخر يرى إن عملية تداخل وتعشيش العناصر سهل .

مجموعة أخرى ترى أن الصفات تستهلك مساحة اقل بكثير من العناصر . فمثلا المثال الأول لو قمنا بكتابته بواسطة الصفات سيكون بهذا الشكل .

```
<Name Nickname='MrScript' First='Emad' Last='Adly'></Name>
```

ولكن ذلك قلل من مرونة مستندات XML التي تتسم بالمرونة .

- في النهاية نقول إن عملية أفضلية الصفات عن العناصر أو العناصر عن الصفات تعود على اختيار الشخص وعلى أسلوبه . فاختار منهم ما يشعرك بالارتياح .

٣) التعليقات
Comments

هي عبارة عن إضافة النصوص التي لا تشكل جزءا من المستن و إنما تخص الشخص الذي يقراء شفرة XML نفسها .

إذ صادف لك وتعاملت من قبل مع أي لغة برمجة فانك تعرف تماما ماذا تعنى بالتعليقات .

وهى تستخدم لوضع بها تعليقات تصف بعض الفقرات و الاكواد لكي تساعدك في فهم شفره مستندات XML

ولكنها ليست مهمة بالدرجة مثلما في إي لغة برمجة أخري حيث أن لغة XML لغة تصف البينات إي أنها ذاتية الوصف تساعدك على فهمها .

تبدأ التعليقات بالرمز < !— وتنتهي بالرمز -- > مثل التعليقات في HTML

مثلا ..

```
<site>
<site-name> الموسوعة العربية </site-name>
<!-- URL Address -->
<site-url>www.c4arab.com</site-url>
</site>
```

٤) العناصر الفارغة Empty Elements

قد توجد في بعض مستندات XML عناصر فارغة من البينات

فمثلا ...

```
<Name>
<first>Adel</first>
<last></last>
</Name>
```

في هذه الحالة تقدم لك XML الحل الأفضل والبديل الذي تكون فيه لا تحتاج إلى معرف نهاية End Tag فتكتب بهذا الشكل

< last/>

على أن يكون العنصر ثم الرمز "/" بدون لاحقة النهاية المعتادة وهى الحالة الوحيدة التي لا تستخدم فيها لواحق النهايات .

ولو أجريت تجربة بسيطة داخل معرب XML المدمجة بمتصفح إنترنت اكسبلورر ٥,٥ وكتبت الشفرة التالية

أنها سوف تظهر بالشكل التالي

```
- <Name>
    <first>Adel</first>
    <last />
  </Name>
```

٥) تصريح XML XML Declaration

تعد التصاريح في تعريف مستندات XML بأنها تتبع نوعا معينا بالإضافة إلي إعطاء المعرب التعليمات الأخرى

فمثلا هذا التصريح

```
<?xml version ='1.0'  encoding ='UTF-16' standalone = 'yes'?>
```

وهو يجب أن يكون في بداية مستندات XML وليس معنى عدم وجوده يسبب الخطأ ولكن الأفضل تعريف المستندات .

شرح التصريح السابق

- فبدا تصريح XML بالرمز < ؟xml
و ينتهي بالرمز ؟ >

?>

- إذا أضفت تصريح XML يجب أن تضيف أيضا الصفة version أما بالنسبة للصفتين Encoding و Standalonge بهذا الترتيب .

- حاليا فان الإصدار Version يجب أن يكون يحوى القيمة ١,٠ إذا قمت بوضع رقم غير هذا الرقم فان XML المكتوب للتعامل مع مواصفات XML الإصدار ١,٠ سترفض المستند .

- يجب أن يأتي هذا التصريح في أول سطر في مستند XML كما ذكرنا .

الصفة الثانية فى تصريح XML هى Encoding

صفة التشفير Encoding Attributes ، لو رجعنا إلى درسنا الأول في هذه السلسلة والذي تحدثنا فيه عن البيانات وأنواعها فلا تستغرب إذا عرفت أن النصوص تخزن في الحاسب على شكل أرقام باعتبار أن الأرقام هي اللغة ألام للحاسب فمثلا الحرف a يمثل في جدول الـ ASCII الرقم ٩٧ والحرف A يمثل الرقم ٦٥ .

هناك نوعان من شفرة ASCII الشفرة القياسية والتي تتكون من سبعة بتات والشفرة الموسعة والتي تتكون من ثمانية بتات أي أن شفرة ASCII الموسعة تستخدم بايتا واحدا لكل رمز وبالتالي فان الشفرة لا يمكن استيعابا اكثر من ٢٥٦ رمز مختلف فهو كافي لتمثيل كل الحروف الأبجدية الإنجليزية الكبيرة والصغيرة بالإضافة إلى العلامات والأرقام ولكن غير كافي لتمثيل لغات أخرى مثل العربية أو اليابانية ... الخ

لهذا السبب وجدت الشفرة الموحدة Unicode .

بعد هذه المقدمة نرجع إلى مثالنا السابق جملة تصريح XML عند الصفة

encoding='UTF-16'

هنا تم ضبط التشفير لمستند XML على انه Unicode وذلك بوضع UTF-16

لذا فقد نصت مواصفات XML على استخدام شفره موحدة عالمية لتمثيل البيانات ولكن للأسف فان القليل يستخدم هذه الشفرة والعديد يستخدم شفرات أخرى مثل ISO-88591 وشفرة windows-1252

ملاحظة إذ لم تقم بتحديد شفرة إي انك لم تقم بوضع هذه الصفة أصلا أن معرب XML يقراء المستند بشفرة UTF-8 أو UTF-16

الصفة الأخيرة فى تصريح XML هى صفة Standalone

وهى تعنى أن المستند قائم بذاتية فإذا قمت بإضافة هذه الصفة إلى تصريح XML فإنه لهذه الصفة قيمتين yes وno

تشير القيمةyes إلى أن المستند قائم بذاته ولا يعتمد على أي ملفات أخرى

تشير القيمة No إلى أن المستند يمكن إن يكون معتمد أو مرتبطا بملفات أخرى.

٦) رموز نصوص PCDATA غير مسموح بها .

تعرفنا سابقا بالمقصود بالمصطلح PCDATA وعرفنا انه مصطلح مستخدم من قبل SGML يشير إلى البيانات النصية الموجودة داخل العناصر ، فتوجد بعض القوانين تحكم هذه البيانات النصية عند كتابتها لفادى الأخطاء أثناء عمل معرب XML .

فهناك بعض الرموز المحجوزة التي لا تستطيع تضمينها في بينات PCDATA لان تركيب XML يستخدمها من هذه الرموز < والرمز &

فمثلا ..

```
<!-- This is not well-formed XML -->
<comparison> 6 is < 7 & 7 > 6 </comparison>
```

عند تشغيل هذه الشفرة على المتصفح سوف تعرض لك الخطأ بهذا الشكل ..

The XML page cannot be displayed

Cannot view XML input using XSL style sheet. Please correct the error and then click the Refresh button, or try again later.

```
<comparison> 6 is < 7 & 7 > 6 </comparison>
-------------------^
```

هذا يعنى انه عندما يصل المعرب إلى الرمز < سيتوقع أن يجد اسما لمعرف Tag وبدلا من ذلك فقد وجد فراغ Space وحتى إن تجاوز المعرب هذا الخطأ فإنه سيتوقف عند خطا أخر عندما يصل إلى الرمز " & "

فلا تقلق فهناك طريقتان تمكنك من تضمين هذه الرموز ضمن PCDATA إما أن تستخدم الرموز المرادفة Escaping Characters أو أن تستخدم قسم CDATA سنتعرف على الطريقتان الآن في الجزء الثاني هنـــــــا .

الرموز المرادفة Escaping Characters

يمكنك استخدام الرمز < ضمن بيناتك النصية في XML باستبدال هذا الرمز بمرادفة والذي نصت علية XML وهو

```
&lt;
```

ويمكنك أيضا استخدام الرمز & وذلك باستبداله بمرادفة وهو

```
&amp;
```

إذا نقوم بكتابة المستند السابق بصورة سليمة بالشكل التالي ..

```
<comparison> 6 is &lt; 7 &amp; 8 &gt; 6 </comparison>
```

وعند عرضه على المتصفح سيظهر بالشكل التالي ..

```
<comparison>6 is < 7 & 8 > 6</comparison>
```

واليك هذا الجدول يوضح لك الرموز المحجوزة في XML ومرادفاتها .

الرمز المحجوز	الرمز المرادف
&	& amp;
>	& lt;
<	& gt;
‘	'
"	&qu;

توجد العديد من هذه الرموز وهى تعرف باسم Character Referances .

فهي رموز Unicode تبدأ بـ #& و تنتهي بالرمز ; ويتوسطها رموز وفق التمثيل الستعشرى وعلى سبيل المثال يمكنك تمثيل الرمز © في مستند XML بإضافة الرمز البديل ؛ & #١٦٩ أو المرادف الستعشرى له & #x٩A؛

أقسام CDATA

إذا كان لديك الكثير من الرموز التي يتحتم عليك استخدامها رموز بديله لها سيصبح شكل شفرة مستندك مزعجا فيأتي هنا الحاجة إلى استخدام ما يعرف بأقسام CDATA

عند استخدام أقسام CDATA سيقوم المعرب بعدم إعراب النص و إنما التعامل معه على انه عبارة عن بيانات PCDATA بما يحتويها من رموز محجوزة وتكتب أقسام CDATA بهذا الشكل

```
<comparison><![CDATA[6 is < 7 & 7 > 6]]></comparison>
```

أي أن كل ما يوضع بين الرموز < ![CDATA[والرموز]] > أي أن معرب XML سيمرر هذا النص كما هو وسو يتجاهل الرموز المحجوزة.

تظهر قوة التعامل مع أقسام CDATA عندما تريد كتابة شفرة برمجية داخل مستند Xml فمثلا..

```
<script language='javaScript'><![CDATA[
    function myFunc()
    {
        if(0 < 1 && 1 < 2)
            alert("Hello");
    }
]]></script>
```

وسيظهر بهذا الشكل على المتصفح ..

```
- <script language="javaScript">
  - <![CDATA[
            function myFunc()
        {
            if(0 < 1 && 1 < 2)
            alert("Hello");
        }

    ]]>
  </script>
```

إعراب مستندات XML :

يرجع السبب لوجود هذه القوانين لكتابة مستندات xml محكمة هو لتسهيل عملية استخراج المعلومات من هذه المستندات .

يسمى معالج لغة xml بمعرب XML Parser ولأنه يقوم ببساطة بأعراب شفرة xml ويوفر للتطبيق المعلومات التى يحتاجها من المستند .

هناك الكثير من المعربات المجانية وسوف اذكر لكم مجموعة منها الآن .

Microsoft Internet Explorer Parser :

أول معربات XML ضمنته شركة مايكروسوفت في متصفحها الإصدار الرابع ولك في بدايات XML ومع الإصدار الخامس من المتصفح زودت لغة xml بأدوات بحيث تم تحديثها إلى الإصدار الأول من هذه المواصفات وعرفت باسم xml 1.0

يمكنك تحميل الإصدار الأخير من معرب xml من موقع مايكروسوفت على هذه الوصلة .

http://msdn.microsoft.com/downloads/webtechnology/xml/msxml.asp

James Clark's Expat :

معرب جايمز كلارك يعد Expat معرب xml 1.0 مكتوب بلغة c وهو أحد معربات XML المجانية ويمكنك تحميلة من الموقع التالي . أخر إصدار هو expat v1.2

ftp://ftp.jclark.com/pub/xml/expat1_2.zip

ولمزيد من المعلومات حول هذا المعرب يمكن مراجعة هذه الوصلة

http://www.jclark.com/xml/expat.html

DataChannel XJ Parser

تعد شركة DataChannel شركة برمجيات حلول الأعمال عملت مع شركة مايكروسوفت لأنتاج معرب xml بلغة جافا يمكنك من الحصول على أخر المعلومات حول هذا المعرب أخر إصدار له من الموقع التالي.

http://xdev.datachannel.com/directory/xml-parser.html

IBM XML4j

تقدم شركة IBM عددا من أدوات وتطبيقات xml بالإضافة إلى المعرب xml4j وهو كتب بلغة الجافا وهو متوفر بالمجان على الموقع التالي

http://www.alphaworks.ibm.com/

Apache Xerces

توفر أيضا مؤسسة Apache للبرمجيات مشروعا أوليا لمعرب XML مازال بإصدارته التجريبية Beta وهو أيضا كتب بلغة الجافا ولغة C++ وباستخدام لغة بيرل أيضا وتجده هنا

http://xml.apache.org/

التعامل مع الأخطاء فى XML :

وكما تم تحديد التعامل مع المعلومات داخل مستند XML تم أيضا التعامل مع الأخطاء بواسطة معرب XML فهناك نوعان من الأخطاء وهى الأخطاء Error والأخطاء fatal errors .

فالأخطاء ببساطة هي انتهاك لموصفات وقوانين XML التي تكلمنا عنها هنا حيث يكون الناتج غير معروف .

أما الأخطاء المميتة أو fatal Errors فهي التي تحتم على المعرب عدم الاستمرار في معالجة مستند xml أي إن أي خطأ يجعل مستند xmlغير محكم الهيئة يسمى خطأ مميت .

فهذه الصرامة في كتابة مستندات xml هي التي تزيد من قوتها وليس مجرد تفادى لاخطاء المعربات ولكنها توحيد كتابه مستندات قياسية يمكن التعامل معه من اكثر من متصفح واكثر من منصة تشغيل . بعكس لغة html لا توجد صرامة في كتابة النصوص مما قد تجد متصفحا يقوم بعرض الصفحة بشكل غير لائق في حين متصفح أخر يعرضها كما تريد . وهذا العيب في html وهو سبب من أسباب ظهور لغة XHTML التي تعتبر العوض لقصور html

Html مقابل XML

الاختلاف الرئيسي بين XML وHTML هو أن الـ XML يأخذ وجهة نظر مختلفة عن HTML ، بالرغم من انه ما ذال يستعمل وسم Tags وهو ليس بديل لـ HTML .

XML و HTML صمما كل منهم لهدف مختلف . أن الاختلاف الرئيسي بان XML صمم لوصف تركيب النص وليس ما هو يجب أن يعرض على صفحة المتصفح .باختصار . XML صمم لحمل البيانات ، من الناحية الأخرى HTML صمم لعرض البيانات والتركيز على شكل عرض هذه البيانات . هكذا يمكن أن نقول إن HTML يعمل على عرض المعلومات ، بينما XML يعمل وصف المعلومات .

دعنا نأخذ مثال بسيط :

نقوم بإنشاء ملف file.html ونكتب بداخل هذه الكود

```
<body>

Hello !!
<h1> Welcome To The C4arab.com </h1>
This is normal text. <b> while this is bold text.</b>

</body>
```

وعند عرض هذه الشفرة على المتصفح سوف يظهر لك شي مثل هذا:

Hello !!

Welcome To The C4arab.com

This is normal text. **while this is bold text.**

ويجب أن نلاحظ إن جملة While this is bold text تظهر كنص غامق .

أما إذا قمنا بعرض الملف هذا على انه وثيقة XML بدون تغير في شكل الـ Tags ، فقط قم بتغير امتداد الملف بدل من file.html إلى file.xml

سوف نلاحظ أن متصفحك قام بعرض الملف ولكن بشكل أخرى يشبه هذا الشكل :

```
- <body>
    Hello !!
    <h1>Welcome To The C4arab.com</h1>
    This is normal text.
    <b>while this is bold text.</b>
  </body>
```

نستنتج من المثال السابق أن شفرة HTML التي تحمل الامتداد html تم عرضها بشكل يجب أن يكون معروف أو متوقع عرضها به . وذلك لأنها تحتوى على وصفات Tags تم تعريفها مسبقا لدى المتصفح لكي يمكنه التعرف عليها وعرض البيانات على أساسها . مثل <body> , <h2> , <h1> .. الخ . فأي متصفح يمكنه أيضا إضافة وصفات جديده لعرض البيانات على سبيل المثال متصفح النتسكيب له tags خاص هي <BLINK> ولكن ليس هناك طريقة قياسية لتقديم أنواع العنصر الجديدة . أما هذا الحال مختلف تماما مع وثائق XML

في حالة تغير امتداد الملف ليتم عرضة كوثيقة xml على المتصفح . هنا تظهر القدرة لتعريف العناصر الجديدة وعلاوة على ذلك ، يمكنها العمل على الوثيقة نفسها ، بحيث يكون الوصف عبارة عن وصف ذاتي self-describing .

دعنا نتخيل مثالا الشكل التالي كوثيقة HTML على هيئه جدول دورات تدريبية :

PHP Programming 15-04-2003 Emad Adly jimmy salh adel

Java Programming 8-03-2004 jimmy Adle Devman karim

ASP Programming 12-5-2004 salh Mohammed SeGa mrscript

```
<table >
    <tr>
        <td> <i>PHP Programming</i> </td>
        <td> 15-04-2003</td>
        <td>  Emad Adly </td>
        <td> jimmy <br> salh <br> adle  </td>
    </tr>
```

(وهكذا لكل صف). . .

```
</table>
```

وبالرغم من إننا كبشر أو مبرمجين يمكن لأي شخص تخمين أن الاسم المذكور شفى هذا الجدول يدل عن انه اسم المعلم أو المحاضر للدورة المذكور بجوارها ، وليس من المعقول التعرف على PHP Programming كاسم المعلم ، أو من هم المشاركين في الدورة ، أما في البرمجة وفهم الحاسب لذلك بطريقة منهجية فان XML يثبت هذا في تعديل هذا الجدول هكذا :

```
<course>

    <name> PHP Programming </name>

    <date> 15-04-2003</date>

    <teacher> Emad Adly </teacher>

    <student> Salh  </student>

    <student> Jimmy </student>

    <student> Adle  </student>

</course>
```

في المثال السابق لـ XML هو يعتبر وثيقة XML قياسية يمكنك أن تعطى إي بيانات تعطي المعنى المناسب للوصف الخاص بهذه البيانات والتي تم تعريفة كوصف قياسي ذاتي لوثيقة XML . باختيارنا لـ Tags تعطى وصف واضح وصريح لما تحتوى من بيانات ،

فهكذا اصبح نص قياسي لذا يمكن أن يتحول بسهولة من جهاز إلى جهاز أو نظام إلى نظام ، لذا إي واحد يمكنه أن يفهم أن هذه الوثيقة تحمل شي من التوضيح إنها دورة Course .

والسؤال الذي يدور في ذهن الآن هو كيف لي أن أهيئ هذا ليظهر في شكل وصورة html على المتصفح ؟

نلاحظ أن HTML تحتوى على تعليمات لوصف النص على المتصفح فعلى سبيل المثال PHP Programming يشير إلى انه نص italics ، هنا نجد أن XML ليس عنده عدد ثابت من الـ tags مثلما يعمل HTML ولكنه قابل للامتداد مثل لغة SGML تسمح لمصمم الوثيقة بتعريف البيانات الخاصة ، XML جاءت لتلبيه النقص في هذه اللغات والحاجة لنشر المعلومات التي تتضمن شبكات المعلومات الكبيرة والنشر الورقي التقليدي واستعمالها على أنظمة غير تقليدية ، وكلما توسعه الإنترنت ستصبح اكثر وضوحا.

لم نتطرق في هذه السلسلة إلى كيفية استخراج البيانات وعرضها على المتصفح بطرق غير تقليدية أو بصورة html باستخدام لغة XSLT

وأيضا لم نتطرق إلى الوصول إلى البيانات في مستندات XML برمجيا بواسطة تقنيات DOM و SAX

ملحوظة : هذا الملحق منقول من موقع http://www.c4arab.com

للكاتب : أسماء المنقوش

الموسوعة العربية للكمبيوتر و الانترنت

خالص الشكر و التقدير للأسرة الموقع و كاتب الموضوع و نحتفظ بكافة الحقوق فى هذا الموضوع للكاتب و للموقع .

ملحق : هندسة البرمجيات

مقدمة:

لم يعد خافيا على أي منا أهمية البرمجيات Software في حياتنا اليومية سواء في البيت أو المصنع أو المستشفى أو ... الخ، فنحن نتعامل يوميا مع العديد من الأجهزة والمعدات التي تعتمد في عملها على البرمجيات ومن المهم لنا أن تعمل هذه الأجهزة وبرامجها بالشكل والكفاءة التي نتوقعها منها. لذا فإن هندسة البرمجيات أصبحت اليوم أكثر أهمية من أي وقت مضى.

المرجع:

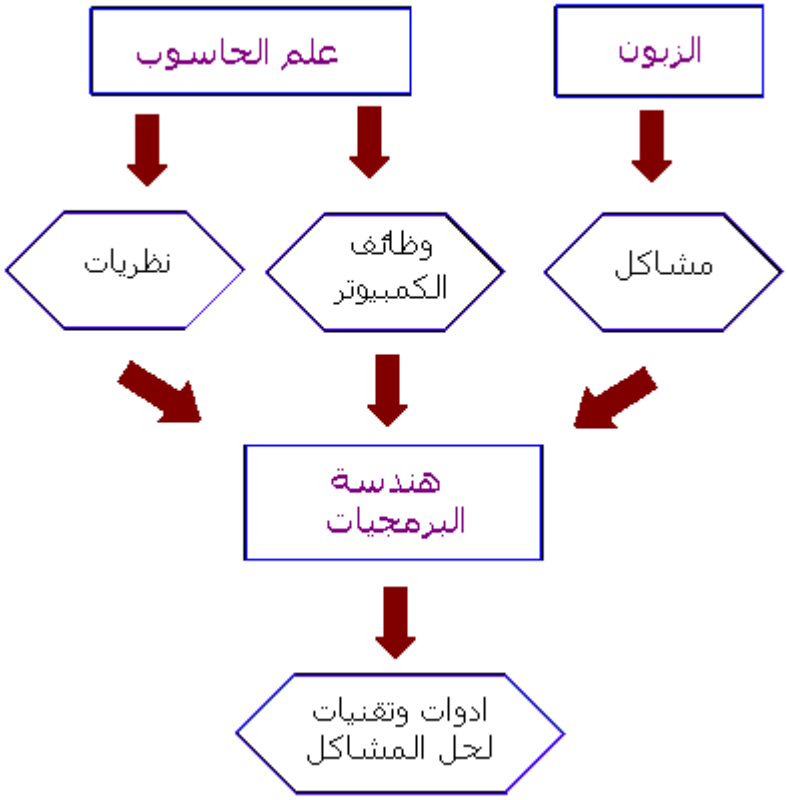
1- Shari Pfleeger, "Software Engineering - Theory and Practice", 2nd Edition

ما هي هندسة البرمجيات؟

لنفهم معا علاقة هندسة البرمجيات بعلوم الكومبيوتر، دعونا نأخذ هذا المثال عن علم الكيمياء واستخدامه في حل المشاكل التي نقابلها في حياتنا اليومية.
يهتم الكيميائي بدراسة المواد الكيميائية (تركيبها، تفاعلاتها، والنظريات التي تحكم سلوكها.)
بينما المهندس الكيميائي يستخدم النتائج التي توصل إليها الكيميائي لحل المشاكل التي يطلب منه إيجاد حل لها.
من وجهه نظر الكيميائي الكمياء هي موضوع الدراسة بحد ذاتها.
ومن وجهه نظر المهندس الكميائي الكيمياء هي أداة tool تستخدم لأيجاد الحلول لمشاكل عامة) وقد لا تكون هذه المشكلة ذات طبيعة كيميائية بحد ذاتها.(

وبنفس الفكرة يمكن النظر إلى علم الحوسبة computer science حيث يكون تركيزنا على الحواسيب ولغات البرمجة لدرستها وتطويرها في حد ذاتها.
أو يمكن النظر إليها والتعامل بها على أنها أدوات نستخدمها عند تصميم وتطوير حل لمشكلة ما تواجهنا أو الآخرين.

مهندس البرمجيات Software Engineer يعتبر أن الكمبيوتر هو أداة لحل المشاكل problem-solving tool.
وعليه أن يستخدم معلوماته حول الحاسوب وعلم الحوسبة للمساعدة في حل المشكلة التي يطلب منه إيجاد حل لها.



شكل (١)

ولكن ومن المهم أن نتذكر أن عملية كتابة البرامج تعد فن Art بقدر ما هي علم، لماذا؟

لأنه يمكن لأي شخص لديه معرفة كافية بأحد لغات برمجة الحاسوب hacker أن يكتب برنامج ليؤدي مهمة محددة، لكن الامر يتطلب مهارة ومعرفة مهندس برمجيات محترف لكتابة برنامج أكثر تناسقا ووضوحا ،وأسهل في الصيانة، ويقوم بالمهمة المطلوبة منه بفعالية ودقة أكبر.

أي أن، هندسة البرمجيات تعنى بتصميم وتطوير برامج ذات جودة عالية.

من يشارك في هذه العملية؟
المشاركون في عملية صناعة البرنامج، عادة ما يندرجون تحت ثلاث مجموعات:

- **الزبون :Customer وهو الشركة (أو الشخص) الممولة لمشر وع تطوير البرنامج المطلوب**
- **المستخدم :User الشخص (أو مجموعة الاشخاص) الذي سوف يقوم فعلا باستعمال البرنامج، والتعامل معه مباشرة.**
- **المطور :Developer وهو الشركة (أو الشخص) الذي سوف يقوم بتطوير البرنامج لصالح الزبون.**

الشكل التالي يظهر العلاقة بين الفئات الثلاثة السابقة

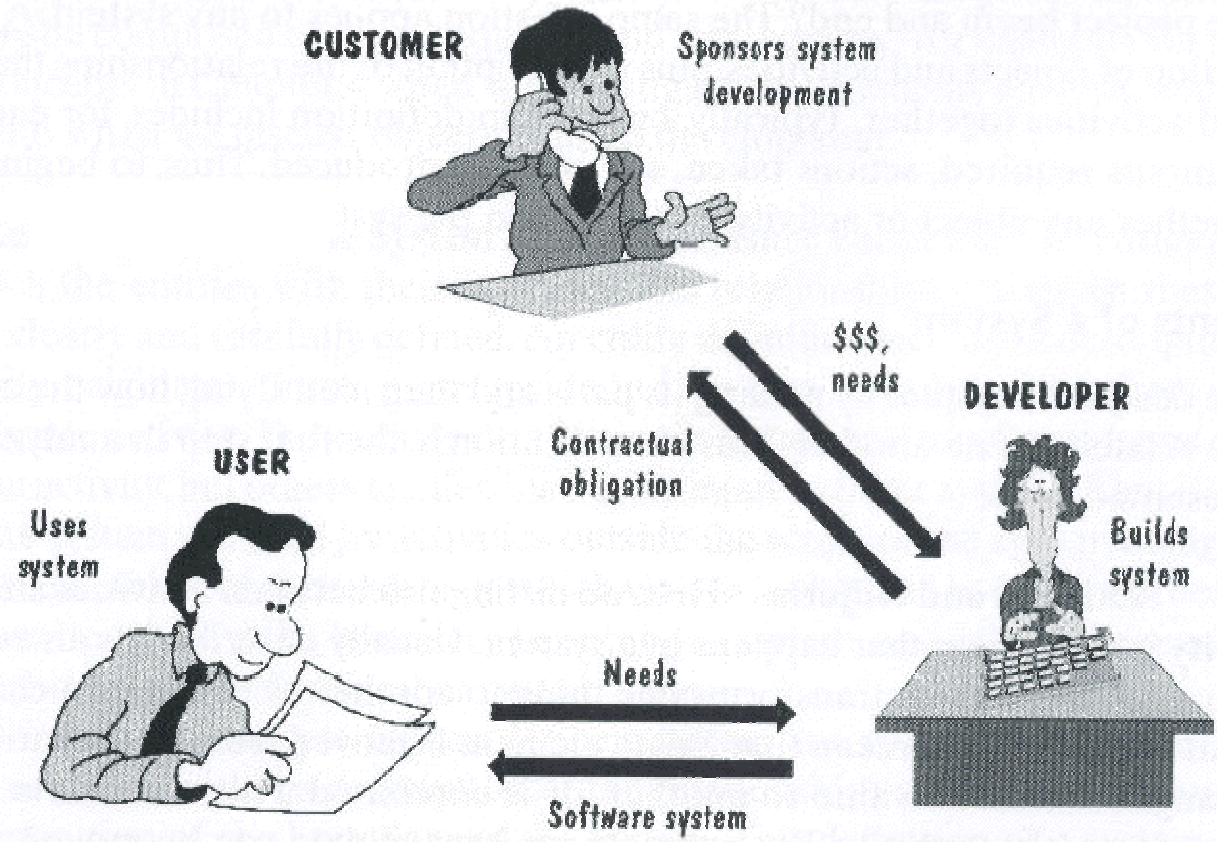


FIGURE 1.7 Participants in software development.

المصدر: المرجع رقم 1

شكل (٢)

مكونات النظام

مشاريعنا التي نطورها لن تعمل في الفراغ، فعليها أن تتفاعل مع مستخدمين، أجهزة ومعدات متنوعة، نظم تشغيل وبرامج وملفات وقواعد بيانات .... إلخ و ربما حتى أنظمة حواسيب آخرى. لهذا يجب تعريف حدود النظام ومكوناته جيدا .أي يجب تعريف ما الذي يشتمل عليه النظام وما الذي لا يشتمل عليه.

أي نظام هو عبارة عن مجموعة من الكائنات objects والنشاطات activities بالإضافة إلى وصف للعلاقات التي تربط تلك الكائنات والنشاطات معا. مع تعريف قائمة المدخلات المطلوبة والخطوات المتبعة والمخرجات الناتجة لكل نشاط.

أول خطوات تحليل المشكلة هو فهم ماهية المشكلة وتعريفها بوضوح، لذا علينا أولا أن نصف النظام بتحديد مكوناته والعلاقات التي تربط بين هذه المكونات.

1.النشاطات والكائنات: النشاط هو عميلة تحدث بالنظام وعادة ما يوصف كحدث يتم من خلال حافز. النشاط يغير شئ ما إلى آخر بتغير خواصه (صفاته(

هذا التغير يمكن أن يعنى تحويل أحد عناصر البيانات من موقع إلي آخر، أو تعديل قيمته إلى قيمة مختلفة.

هذه العناصر تسمى كائنات objects وهي عادة ماتكون مرتبطة ببعضها البعض بشكل أو بأخر. مثلا الكائنات يمكن أن تكون مرتبة في مصفوفة أو سجل) قيد.(

وصف هذه الكائنات نوعها، النشاطات التي يمكن إجرائها عليها ... يجب وضعها بدقة هي ايضا.

2.العلاقات وحدود النظامRelationships and System Boundary

بعد تعريف الكائنات والنشاطات جيدا، يمكن أن نربط بين كل كائن والنشاطات المتعلقة به بدقة. تعريف الكائن يتضمن الموقع الذي سوف ينشأ به(نعض العناصر يمكن أن تكون موجودة بملف سبق انشاءه، والبعض قد يتم انشاءه خلال

حدث ما(، والهدف من انشاءه(بعض الكائنات تستخدم من قبل نشاط واحد فقط والبعض يمكن أن يستعمل من قبل نظم آخرى كمدخلات , (Input لذا يمكن أن نعتبر أن لنظامنا حدود boundary بعض الكائنات بمكن أن تعبر هذه الحدود إلى داخل النظام، والبعض الآخر هي مخرجات من نظامنا ويمكن أن ترحل إلى نظم آخرى.

بهذا يمكن أن نعرف النظام A System على أنه تجمع من:
- مجموعة من الكائنات.entities
- مجموعة من الانشطة.activities
- وصف للعلاقات بين الكائنات والانشطة.Relationship
- تعريف لحدود النظام.boundary

كيف نبني نظام؟

إذا طلب منا عميل تطوير نظام (برنامج) له، لحل مشكلة معينة تواجهه في عمله. فمثلا يحتاج نظام حماية لشركته، أو نظام صرف آلي لبنك، أو ممكن أن يكون صاحب مكتبة أو متجر و يريد تغير نظام البيع و الشراء أو العرض ليتم بشكل آلي. علينا اتباع الخطوات التالية لبناء هذا النظام:

1. عقد اجتماع مع العميل لتحديد متطلباته، هذه المتطلبات تشمل وصف النظام بجميع مكوناته التي شرحنا.
2. وضع تصميم عام للنظام يحقق المتطلبات التي حددها العميل، وعرضه على العميل ليوضح له الشكل الذي سيظهر عليه النظام عند الانتهاء، و ومراجعته معه لأخذ موافقته عليه.
3. بعد موافقة العميل على التصميم يتم العمل على وضع التصاميم التفصيلية لأجزاء المشروع.
4. كتابة البرنامج
5. اختباره، واعادة مراجعة المتطلبات التي وضعها العميل للتأكد من تحققها في البرنامج.
6. تسليم النظام إلى العميل.
7. بعد تسلم العميل للنظام قد تظهر بعض المشاكل أو الاخطاء التي لم تظهر خلال عملية الاختبار، والتي تجب على المطور اصلاحها فيما يعرف بصيانة النظام

عملية بناء أي منتج تمر بعدة مراحل يطلق عليها عادة "دورة الحياةLife Cycle "، ومما تعلمنا في الدرس السابق فإن دروة حياة تطوير أي نظام برمجي Software development life cycle تتضمن المراحل التالية:

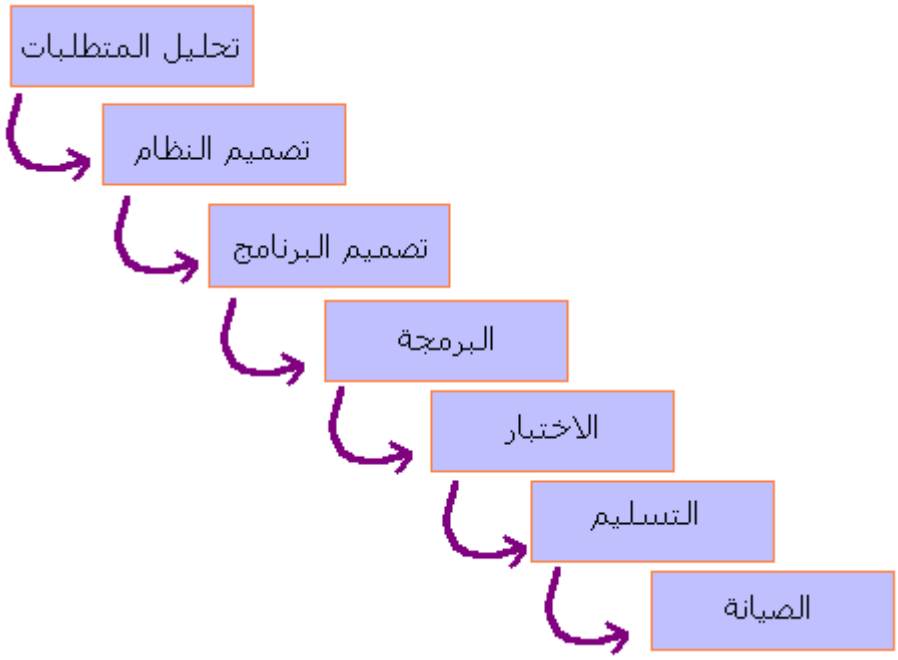
1. تحديد وتعريف المتطلباتRequirements analysis and definition
2. تصميم النظامSystem design
3. تصميم البرنامجProgram design
4. كتابة البرنامج (تطويرهProgram implementation)
5. أختبار وحدات البرنامجUnit testing
6. أختبار النظامsystem testing
7. تسليم النظامsystem delivery
8. الصيانةmaintenance

كل مرحلة من تلك المراحل تتضمن العديد من الخطوات أو النشاطات ولكل منها مدخلاتها ومخرجاتها وتأثرها على جودة المنتج النهائي) البرنامج.(
دورة حياة أي منتج تبدأ بأول خطوة وهي تحديد المتطلبات وتتدرج إلى باقي الخطوات كما هي مرتبة حتى الوصول إلى آخر خطوة وهي تسليم البرنامج وصيانته (إن دعت الحاجة)، إلا أن التجارب العملية تظهر أن هذا ليس ضروريا وأن دورة حياة تطوير البرامج قد تأخذ أشكال (أو أنماط) مختلفة. وفي هذا الدرس سوف نتعرف إلى هذه الأنماط

أنماط دورة الحياة:Lifecycle Models

النموذج الانحداريWaterfall Model

في هذا النموذج تسير دورة الحياة بشكل تدريجي بدأ من الخطوة (١) وحتى الخطوة (٨)، وكما يظهر بالشكل (١) فإن كل مرحلة تبدأ بعد الأنتهاء من المرحلة التي تسبقها مباشرة.



شكل (١)

يتميز النموذج الانحداري بالبساطة، ولذا فإنه يسّهل على المطور توضيح كيفية سير العمل بالمشروع للعميل (الذي عادة لا يعرف الكثير عن صنع البرمجيات) والمراحل المتبقية من العمل. وقد كان هذا النموذج أساس عمل كثير من المؤسسات لفترة طويلة مثل وزارة الدفاع الامريكية، واستنبط منه العديد من النماذج الاكثر تعقيدا.
إلا أن لهذا النموذج العديد من العيوب، أهمها أنه لا يعكس الطريقة التي يعمل بها المطورون في الواقع. فباستثناء المشاريع الصغيرة والبسيطة (أي أنها مفهومة بشكل جيد للمطور) فإن البرمجيات عادة ما تنتج بعد قدر هائل من التكرار والاعادة. في حين أن هذا النموذج يفترض أن يكون الحل واضح ومفهوم وسبق تحليله بالكامل قبل مباشرة مرحلة التصميم وهو أمر يكاد يكون شبه مستحيل مع الانظمة الضخمة. وحتى إن كان ممكن فإنه يأخذ وقت طويل جدا (ربما سنوات!)

باختصار،النموذج الانحداري سهل الفهم و بسيط في إدارته. لكن مميزاته تبدأ في التداعي بمجرد أن يزداد تعقيد المشروع.

التطوير على مراحلPhased Development

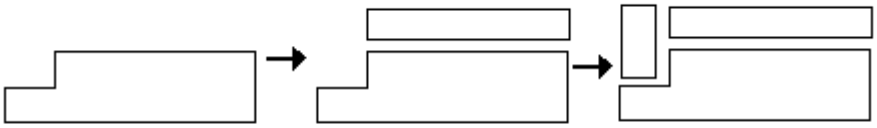
حسب النموذج الانحداري فإنه يجب على المطورين إنهاء مرحلة تحليل المشروع بشكل تام قبل البدأ في التصميم، وكما وضحنا فإن هذه المرحلة قد تتطلب وقت طويل في بعض المشاريع وقد تمر عدة سنوات قبل أن يرى البرنامج النهائي النور، ولكن هل يمكن لسوق العمل الانتظار كل هذا الوقت؟!

الاجابة بالطبع لا.
لذا كان لابد من ايجاد طرق آخرى لتقليل زمن تطوير المشروع .Cycle time أحد هذه الطرق هي التطوير على مراحل Phased Development حيث يتم تطوير النظام على عدة مراحل، بتقديم إصدار من البرنامج به بعض الوظائف للعميل والعمل على تطوير الاصدار الاحق الذي سوف يقدم له بقية الوظائف.

يوجد عدة طرق يمكن بها تنظيم عملية تطوير إصدارات البرنامج، ومن اشهرها:

·النموذج التزايديIncremental model

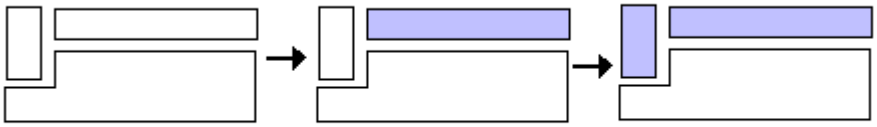
حيث يتم تقسيم النظام المطلوب تطويره إلى عدة اجزاء حسب الوظائف التي يعتين عليه القيام بها، يبدأ أول إصدار بأحد تلك الاجزاء ومع الوقت يتم إضافة المزيد من الاجزاء (الوظائف) حتى يتم الانتهاء من تطوير النظام بشكل تام وحسب متطلبات العميل.



Incremental Development

·النموذج التكراريIterative model

هذه المرة يتم تسليم برنامج بكامل الوظائف من أول مرة، ولكن يتم تعديل وتغيير بعض تلك الوظائف مع كل إصدار من البرنامج.



Iterative Development

من مميزات هذا الأسلوب أنه يمكن المطورين من الحصول على ملاحظات وتقييم الزبون مبكرا و بصورة منتظمة، ورصد الصعوبات المحتملة قبل التمادي بعيدا في عمليات التطوير.كم أنه يمّكن من اكتشاف مدى حجم و تعقيد العمل مبكرا.

النموذج اللولبيSpiral Model

وهو شبيه لدرجة كبيرة إلى النموذج التزايدي والتكراري، ولكن فيه يتم دمج فعاليات التطوير مع إدارة المخاطر riskمن إجل التحكم بها وتقليلها.

يبدأ النموذج اللولبي بمتطلبات العميل مع خطة العمل المبدئية (الميزانية، قيود النظام، والبدائل المتاحة). ثم يتقدم خطوة إلى الامام بتقدير المخاطر وتمثيل البدائل المتاحة قبل تقديم ما يعرف بـ "وثيقة العمليات Concept of " Operationsالتي تصف وبشكل عام (بدون الدخول في التفاصيل) كيف يجب على النظام أن يعمل. بعدها يتم تحديد وتدقيق المتطلبات للتأكد من أنها تامة ودقيقة إلى أقصى حد ممكن.

بذلك تكون وثيقة العمليات هي المنتج من الطور الأول، و المتطلبات هي المنتج الاساسي من الطور الثاني. وفي الطور الثالث تتم عملية التصميم، أما الاختبار فيتم خلال الطور الرابع.

في كل طور أو مرحلة يساعد تحليل المخاطر على تقدير البدائل المختلفة في ضوء متطلبات وقيود النظام، وتساعد النمذجة على التحقق من ملائمة أي بديل قبل أعتماده.

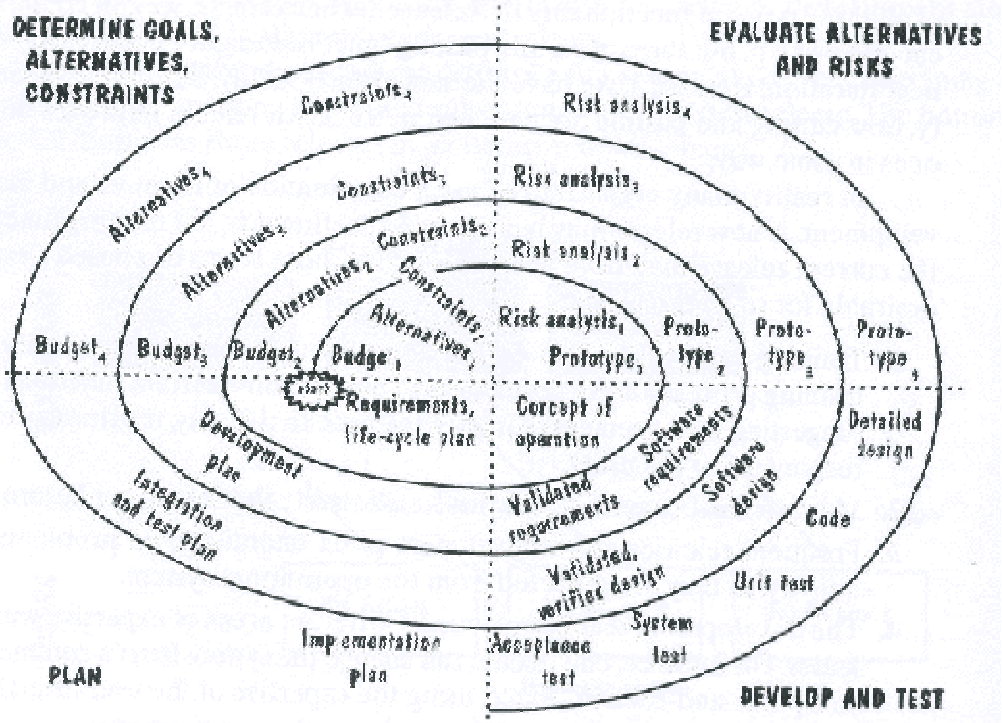


FIGURE 2.10 The spiral model

المصدر: المرجع رقم 1

سوف نبدأ في دراسة أول (ولعلها أهم) خطوة في تطوير البرامج وهي تحديد متطلبات النظام Capturing the requirements.

الهدف من تحديد المتطلبات هو فهم ما يتوقعه العميل والمستخدم من النظام (ما الذي يمكن للنظام أداؤه وما لا يمكنه أداؤه).فقد يكون النظام المطلوب تصميمه بديل لنظام أو لطريقة مستخدمة لأداء مهمة محددة، أو ممكن أن يكون نظام جديد يقدم خدمة جديدة لم يسبق تقديمها من قبل. فلكل نظام برمجي وظيفة معينة، تحدد بما يمكن له أن يقوم به من أجل أداء تلك الوظيفة.

المتطلبات :هي تعريف لشكل النظام أو وصف لما يستطيع هذا النظام أن يقوم به لأداء وظيفته التي سيصمم من أجلها.

خطوات تحديد المتطلبات :

أولا: الاجتماع مع العميل للتعرف على المتطلبات:

وهذه خطوة هامة جدا إذ أن بقية الخطوات التالية تعتمد عليها بشكل أساسي. لذا يجب علينا أن نستخدم كافة التقنيات المتاحة لنكتشف ما الذي يطلبه العميل والمستخدم، نبدأ بفهم وتحليل المشكلة التي تواجه المستخدم بكل أبعادها، نتعرف على العمليات والمصادر التي تتضمنها المشكلة والعلاقات التي تربطها معا و نحدد حدود النظام. وهذا يمكن أن يتم من خلال:

- **طرح الأسئلة على العميل، ومن المفيد أحيانا أن نطرح نفس السؤال ولكن بأسلوب مختلف أكثر من مرة فهذا يساعدنا على التأكد من أننا نفهم ما يقصده العميل بالتحديد.**
- **عرض نظم مشابه للنظام المطلوب سبق تصميمها من قبل.**

- تصميم وعرض نماذج لأجزاء من النظام المطلوب أو للنظام بالكامل.

تقسم المتطلبات إلى عدة عناصر تشمل:

- البيئة المحيطة بالنظامPhysical Environment
- وجهات الاستخدامInterfaces
- المستخدمين وإمكاناتهمUsers and human factors
- وظائف النظامFunctionality
- التوثيقDocumentation
- البياناتData
- المصادرResources
- الأمنSecurity
- ضمان الجودةQuality Assurance

ويجب التأكد من أن نناقش جميع هذه العناصر

ثانيا: تسجيل هذه المتطلبات في وثائق أو قاعدة بيانات، وعرضها على العميل ليوافق عليها باعتبار أنها ما يطلبه بالفعل

المتطلبات لا تصف فقط تدفق البيانات والمعلومات من وإلى النظام، وأما تصف كذلك القيود المفروضة على عمل النظام. وبذلك فإن عملية تحديد المتطلبات تخدم ثلاثة أغراض:

- أولا تمكن المطورين من شرح فهمهم للطريقة التي يود المستخدمين أن يعمل بها النظام.
- ثانيا توضح للمصممين ماهية الوظائف والخصائص التي سيمتاز بها النظام ,
- وثالثا: توضح المتطلبات لفريق الاختبار ما الذي يجب إثباته لإقناع الزبون أن النظام الذي تم تطويره هو ما سبق أن طلبه بالضبط .

لذلك ولضمان أن كلا من المطورين والزبون متفاهمون تماما على ما يجب القيام به، فإن المتطلبات المسجلة حتى هذه الخطوة يجب أن تكون لها الصفات التالية:

1. أن تكون صحيحة Correct وخالية من الأخطاء.
2. أن تكون ثابتة consistent بمعنى أن لا يكون هناك أي تعارض بين متطلب وآخر.
3. أن تكون تامة Complete يجب أن يتم ذكر جميع الحالات المحتملة للنظام، المدخلات، المخرجات المتوقعة منه، ...الخ .
4. أن تكون واقعية realistic بمعنى أن تكون قابلة للتطبيق في الواقع.
5. أن تكون متعلقة بأمور ضرورة للعميل، ويتطلبها النظام.
6. أن يكون من الممكن التحقق منهاverifiable
7. أن تكون قابلة للتتبعtraceable

يطلق على هذه الوثائق "وثائق تعريف المتطلبات Requirement Definition Document "

ثالثا: إعادة تسجيل المتطلبات بشكل رياضي mathematical ليقوم المصممون بتحويل تلك المتطلبات إلى تصميم جيد للنظام في مرحلة التصميم.

لسنوات عديدة كان يتم الاكتفاء بوثيقة تعريف المتطلبات (التي تحدثنا عنها قبل قليل) والتي تكتب باستعمال اللغة الطبيعية) لغة البشر) لوصف وتسجيل متطلبات النظم بحيث يمكن للعميل أن يفهم كل كلمة موجودة بها، إلا أن ذلك يسبب العديد من المشاكل والتي يعود سببها في أغلب الأحيان إلى سوء تفسير بعض التعبيرات للمستخدمين من قبل المصمم أو العكس، فعلى سبيل المثال قد يطلق المستخدم على النظام التعبير (متوقف عن العمل) إذا كان النظام مشغول بعملية تسجيل احتياطي backup باعتبار أن لا يستجيب لأوامر المستخدم في هذه الحالة، بينما يعتبر المصمم أن النظام في هذه الحالة (مستمر في العمل) لأنه يقوم بمهمة أساسية!

لذا فأن الاعتماد على اللغة البشرية بشكل تام قد يؤدي إلى أخطاء كثيرة عند تصميم النظام، وينتج عنها نظام لا يقبله العميل لأنه لا يلبي متطلباته التي حددها من قبل، لذلك يتم كتابة نوع ثاني من الوثائق تسمى "وثائق مواصفات

المتطلبات Requirement specification Document "وهي تكتب باستعمال وسائل وطرق خاصة ابتكرها مهندسو البرمجيات لكتابة المتطلبات باسلوب تقني بحت. منها على سبيل المثال: لغة النمذجة الموحدة UML Unified Modeling Languageو هي لغة نمذجة رسومية تقدم لنا صيغة لوصف العناصر الرئيسية للنظم البرمجية.
الشكل التالي يعرض مثال على استعمالUML

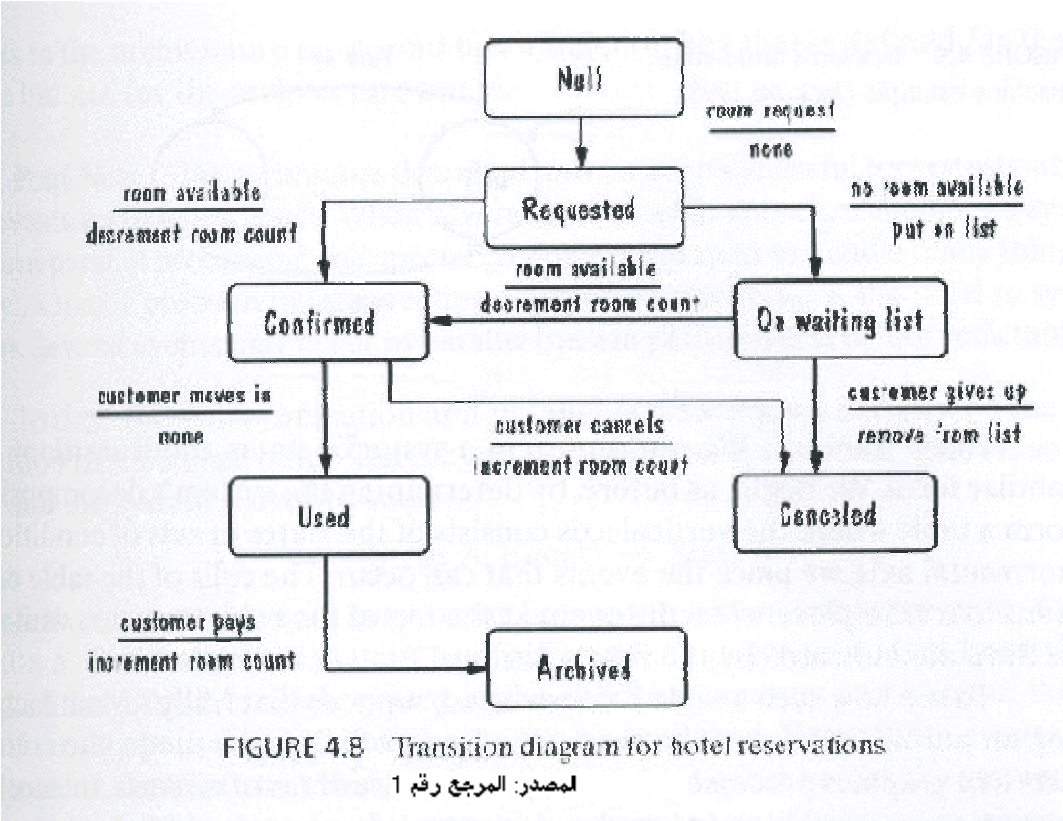


FIGURE 4.8 Transition diagram for hotel reservations
المصدر: المرجع رقم 1

رابعا: التثبت والتحقق من المتطلبات التي تم تسجليها في كلا من وثيقة تعريف المتطلبات (والتي تقدم للعميل) ووثيقة مواصفات المتطلبات (والتي تقدم للمصمم (للتأكد من صحتهما وشموليتهما وأن كلا منهما لا تعارض الثانية في أي نقطة، وإلا فإن النتيجة سوف تكون نظام لا يلبي طلبات العميل.!

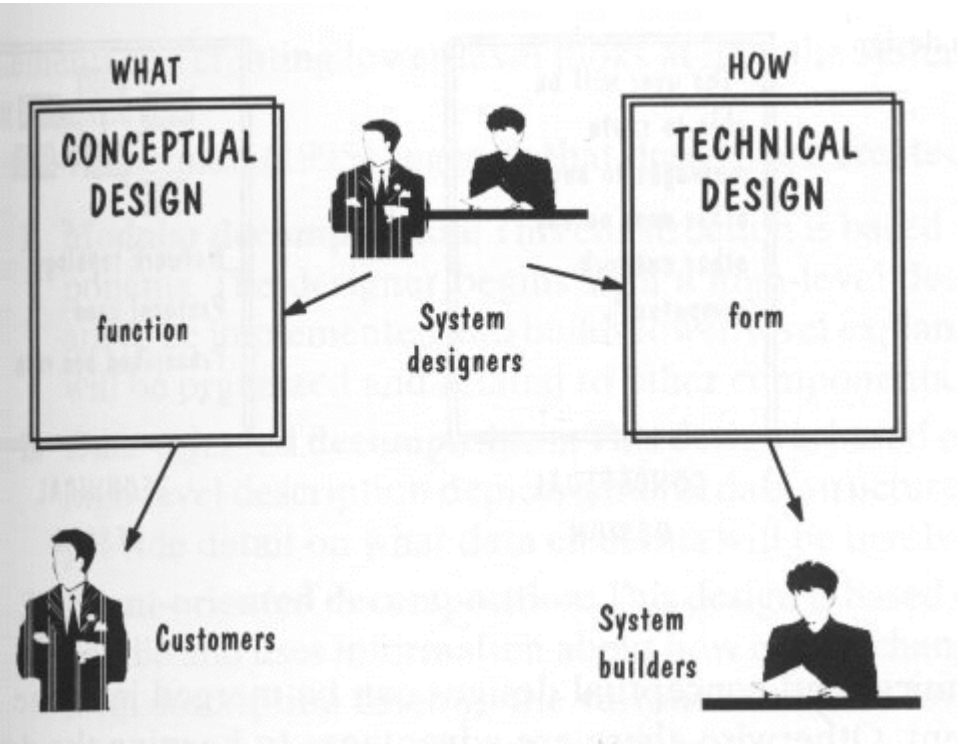
نكمل مع خطوات بناء النظام، وهذه المرة سوف نتحدث عن خطوة "تصميم النظام "

ما هو التصميم؟
التصميم هو عملية إبداعية لإيجاد حل لمشكلة، كما تطلق عادة كلمة تصميم على وصف هذا الحل.
حيث نستفيد من المتطلبات التي حددنها في الخطوة السابقة في التعرف على المشكلة، ثم نبدأ في التفكير في الحل الذي يفي بجميع الشروط والمواصفات التي تحددها المتطلبات، وغالبا ما يمكن إيجاد عدد غير محدود من الحلول يمكن لنا أن نختار أحدها و الذي نجده الأنسب من بينها.

عند الانتهاء من خطوة تحديد المتطلبات، فإننا ننتهي بوثيقتين (كما ذكرنا في الدرس السابق) الأولى هي (وثيقة تعريف المتطلبات) ويتم تقديمها للعميل والثانية (وثيقة مواصفات المتطلبات) ويتم تقديمها للمصمم.

ودور المصمم هو تحويل هذه الوثائق إلى نظام يرضي العميل (يلبي احتياجاته)، وفي نفس الوقت يرضي المطور (يمكن تطبيقه.)
لذا فإن عملية التصميم في عملية تكرارية iterative من خطواتين:

أولا :يتم إنتاج التصميم التصوري conceptual design والذي يوضح للعميل ما الذي سيقوم به النظام



بالتحديد .
وفي حال موافقة العميل على هذا النظام، يتم الانتقال للخطوة التالية.

ثانيا :تحويل التصميم التصوري إلى وثيقة بها تفاصيل أكثر عن التصميم يطلق عليها اسم **التصميم التقني technical design**والذي يجب أن يظهر للمطور ما هي المعدات والبرمجيات اللازمة لبناء النظام.

أحيانا يتطلب الأمر للعودة إلى الخطوة الأولى) التصميم التصوري) والتعديل عليه، لذا فأنها عملية تكرارية حتى الوصول إلى التصميم الذي يرضي العميل ويمكن تطبيقه على أرض الواقع في ظل الإمكانيات المتاحة للمطورين.

التصميم التصوري:conceptual design
يركز هذا التصميم على وظائف النظام **functions** ويكتب بلغة يمكن للعميل أن يفهمها (لغة البشر) ليجيب عن أسئلة العميل حول ماذا (WHAT) يعمل النظام. ويجب أن يكون خالي تماما من أي تفاصيل برمجية أو فنية. والاهم أن يحقق كل المتطلبات التي تم تحديدها سابقا.

التصميم التقنىtechnical design
هذا التصميم سوف يتم تقديمه إلى مطوري النظام ليقوموا هم بتحويله إلى النظام المطلوب، لذا يجب أن يقدم هذا التصميم إجابة شافية لأسئلة المطور عن كيفية (HOW) تطوير النظام. ولمنع إلى تضارب في المفاهيم فإن هذا التصميم عادة ما يكتب باستعمال تعبيرات وأساليب تقنية.

كتابة البرنامج واختباره

هذا الدرس لن يعلمك لغة برمجة لتكتب بها البرامج، ولكن الهدف منه التعرف على:

- القواعد الصحيحة لكتابة البرامج
- خطة الاختبار وأنواع الاختبارات

الجزء الأول :كتابة البرامج:

بعد وضع التصميم للنظام واختيار لغة البرمجة المناسبة، تبدأ الخطوة التي سوف تنقل التصميم المكتوب على الورق إلى واقع. خلال هذا الدرس سوف نناقش أهم القواعد التي على المبرمج إتباعها أثناء كتابة برامجه. ولكن قبل ذلك لنجيب على هذا السؤال الذي لا شك أنه ورد على ذهنك الآن

س: لماذا علينا إتباع هذه القواعد؟

ج :إذا كنت تعمل منفردا في كتابة برامجك، فإن إتباعك لقواعد وأساليب قياسية في البرمجة سوف تساعدك على تنظيم أفكارك لتجنب الوقوع في الأخطاء. كما أنها ستساعدك على اكتشاف أي أخطاء قد تحدث بسرعة وبسهولة.

أم إذا كنت تعمل ضمن فريق برمجي، فإن إتباع القواعد والأساليب القياسية في كتابة أجزاء البرامج التي يطلب منك كتابتها، سوف تساعدك وبقية الفريق من تنسيق أعمالكم وتنظيمها، كما أنها ستقلل من عدد الأخطاء في البرنامج وتساعد على اكتشاف ما يقع منها في اسرع وقت ممكن.

تفرض الكثير من شركات البرمجة على مبرمجيها إتباع قواعد قياسية في كتابة برامجهم، وذلك لضمان التكامل في جميع البرامج، كما أن بعض الشركات تعين فرق لاختبار البرامج، غير الفريق الذي قام بالبرمجة ولذلك يجب أن يكون الكود البرمجي مكتوب بطريقة واضحة لجميع من يقرأه، وليس لمن قام بكتابته فقط.

بعض قواعد البرمجةProgramming Guidelines

- **هياكل التحكمControl Structures**

يقصد بها تلك الهياكل التي تتحكم في مسار عمل البرنامج (مثل Goto ،if- else) ، وأثناء كتابة هذه الهياكل علنا أن نحاول أن نجعلها واضحة وسهلة التتبع، وخالية من القفزات الواسعة قدر الإمكان. انظر لهذا المثال:

```
benefit = minimum;
      if (age < 75) goto A;
     benefit = maximum;
     goto C;
     if (age < 65) goto B;
     if (age < 55) goto C;
A:    if (age < 65) goto B;
     benefit = benefit * 1.5 + bonus;
     goto C;
B:    if (age < 55) goto C;
     benefit = benefit * 1.5;
C:   next statement
```

نفس الكود يمكن كتابته على هذا النحو:

```
if (age < 55)  benefit = minimum;
   else if (age < 65)  benefit = minimum + bonus;
   else if (age < 75)  benefit = minimum * 1.5 +bonus;
   else  benefit = maximum;
```

- **عالم البرمجة هناك قاعدة تقول أن العمومية ميزةgenerality is a virtue ، لذلك حاول دائما أن تجعل شفراتك البرمجة عامة، لتتمكن من إعادة استعمالها في بقية برامجك بأقل قدر ممكن من التعديل، ولكن حاذر من التمادي في ذلك!**
- **لا تستخدم أبدا أسماء لا معنى لها لمتغيرات أو بارمترات برنامجك (ينصح بمراجعة هذا الدرس" التسمية في البرنامج، درس لابد من أن يقرأه كل مبرمج!(""**
- **"أريد برنامجا سريعا" وكلنا نريد ذلك، ولكن ما هو الثمن؟!**

عندما تفكر في جعل برنامجك أسرع ما يمكن، عليك أن تفكر كذلك في الثمن الذي ستدفعه مقابل ذلك:

١. **البرنامج السريع قد يتطلب منك كتابة كود معقد يتطلب منك (ومن فريق العمل) المزيد من الوقت والجهد في كتابته.**
٢. **الوقت الذي تحتاجه عملية اختبار البرنامج المعقد في مختلف حالته.**
٣. **الوقت والجهد الذي تحتاجه لتعديل هذا الكود أو لتطويره.**

زمن تنفيذ البرنامج ما هو إلا جزء من معادلة كبيرة لحساب تكلفة البرنامج، لذلك عليك أن تعادل بين السرعة، والجودة، واحتياجات الزبون. ولا تضحي بالبساطة والوضوح من أجل السرعة.

- **التوثيق: لا تهمل أبدا توثيق برنامجك، ما سُمي الإنسان إنسانا إلا لنسيانه.**

الجزء الثاني :اختبار البرامج:

وصلنا الآن إلى آخر مرحلة في تطوير النظام، وهي اختبار البرنامج للتأكد من أنه يعمل على النحو الذي يتوقعه الزبون.

قبل تسليم النظام النهائي إلى الزبون تجرى عليه الكثير من الاختبارات، بعضها يعتمد على ما الذي يتم اختباره مثلا:

)أحد مكونات البرنامج - مجموعة من المكونات - جزء من النظام - النظام بالكامل(

والبعض الأخر يعتمد على ما الذي نريد معرفته من هذه الاختبارات، مثلا:

- **هل يعمل النظام وفقا لما ورد في المتطلبات؟**
- **هل يعمل النظام وفقا لما ورد في التصميم؟**
- **هل يعمل النظام كما يتوقعه الزبون منه؟**

مراحل الاختبار:

عند العمل على اختبار نظام من الحجم الكبير، فإن عملية الاختبار تتم على عدة مراحل موجزها في ما يلي:

١. **اختبار المكون Module Testing أوcomponent Testing**

أول مراحل اختبار النظم، هي اختبار كل مكون على حدى بمعزل عن بقية مكونات النظام، للتأكد من عمله على النحو المتوقع منه. باختبار المعلومات المتحصل عليها (output) منه بعد إمداده بالبيانات اللازمة له.(input)

٢. **اختبار التكاملIntegration Testing**

بعد اختبار كل مكونات النظام والتأكد من سلامة تصميمها، يجب أن نتأكد من أنها ستعمل معا بشكل صحيح وأنه لا يوجد تضارب بين بعضها البعض بحيث أن المعلومات المنتقلة بين هذه المكونات تصل بالهيئة المتوقعة لها. وهذا هو الهدف من اختبار التكامل.

٣. **اختبار الوظيفةFunction Testing**

ويقصد به اختبار النظام بعد تجميع كل مكوناته للتأكد من أنه يؤدي الوظيفة التي يتعين عليه القيام بها، والموضحة في وثائق متطلبات النظام .عندما يجتاز النظام هذا الاختبار يمكننا اعتبار هذا النظام على أنه نظام عامل Functioning System

٤. **اختبار الأداء Performance Testing**

في هذه الخطوة يتم اختبار أداء البرنامج في بيئة عمل الزبون للتأكد من أن النظام متوافق مع بقية المتطلبات . عند اجتياز النظام لهذا الاختبار يتم التصديق على النظام validated system وبهذا فإننا نعتبر أن النظام أصبح جاهز حسب مفهومنا لما طلبه الزبون.

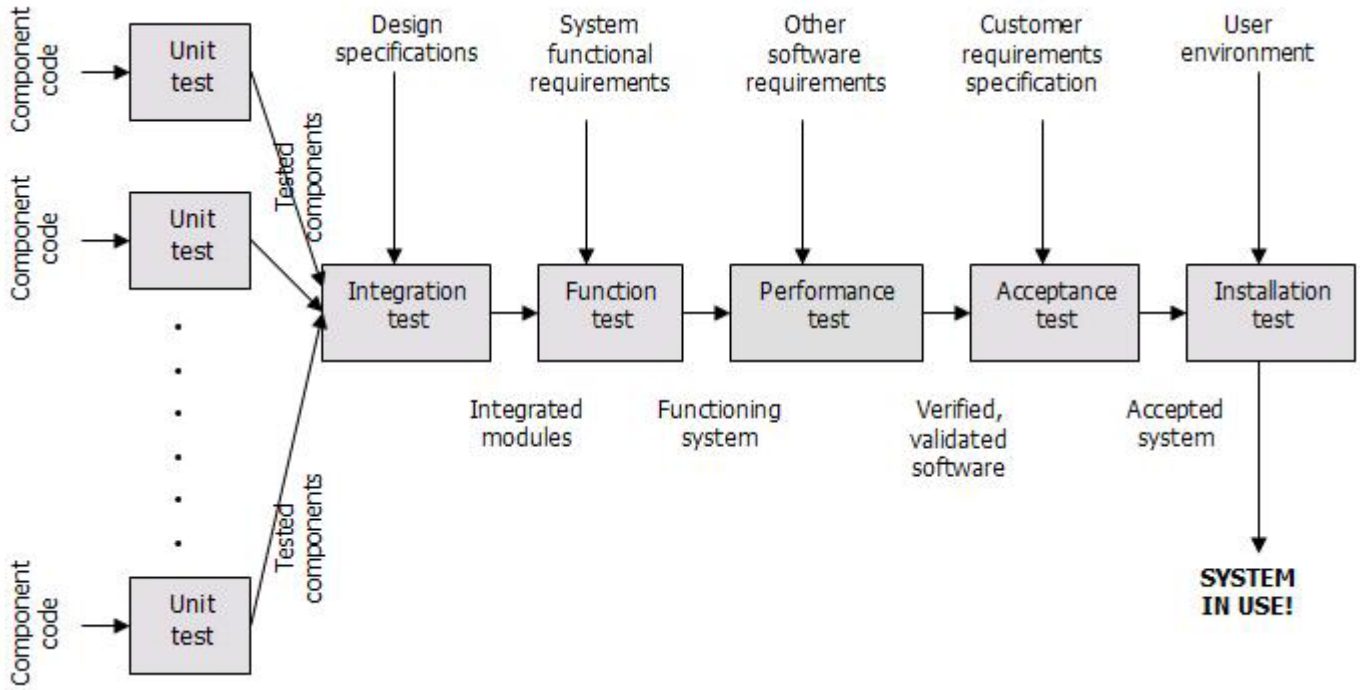
٥. اختبار القبول Acceptance Test

يتم إجراء هذا الاختبار للتأكد من أن النظام المحقق موافق لما توقعه الزبون، وبعدها يعد النظام مقبول عند المستخدم والزبون Accepted system

٦. اختبار التثبيت Installation Test

الاختبار الأخير يتم فيه تثبيت النظام في بيئة العمل الخاصة به والتأكد من أنه يعمل كما هو مطلوب منه.

الشكل التالي يوضح خطوات تطبيق عملية اختبار النظام، والتي يحسن تطبيقها على اي نظام مهما كان حجمه للتأكد من أنه سيؤدي المهمة المطلوبة منه.



المراجع

Sams Teach Yourself Macromedia® Flash™ 8 in 24 Hours
By Phillip Kerman
Essential ActionScript 2.0
By Colin Moock
Object-Oriented Programming with ActionScript 2.0
By Jeff Tapper, James Talbot, Robin Haffner
Macromedia Flash MX 2004 Game Programming
by Craig S. Murray and Justin Everett-Church
Flash™ MX 2004 ActionScript Bible
Robert Reinhardt and Joey Lott
Object-Oriented
ActionScript for Flash 8
Peter Elst and Todd Yard
with Sas Jacobs and William Drol
Macromedia® Flash® 8 ActionScript: Training from the Source
By Jobe Makar, Danny Patterson

Learning ActionScript 2.0 for Macromedia® Flash® 8
Publisher: **Macromedia**
http://www.c4arab.com
http://xml.apache.org/
www.sadaagroup.com

السيرة الذاتية للمؤلف

Michael Nabil Akhnokh

Mobile : 0103546609

E-mail : ramegypt@yahoo.com
Micheal240001@yahoo.com

Education

- Faculty of Commerce, Ain Shams University
BA of Commerce
Major: Accounting
Graduation Year: 2001
- BBSA Nasr City, EG
Basic Business Skills Acquisition Program Sponsored by Future Generation Foundation
During the period from 13/1/2002 To 25/4/2002
- IT Professional Training Program provided by Ministry of communication and information Technology in cooperation with IBM Egypt
Specialty track: Solution Developer and programmer
During the period from 29/6/2003 To 15/1/2004

Qualifications and Skills

Computer Skills:

- **MCP (Microsoft Certified Professional) in:**
  Developing Web Applications with Microsoft Visual C#. NET (Exam 70 – 315), (January 2004)
  - **Operating Systems (Windows 98, Windows 2000 server, Linux)**
  - **Skillfully using all Office Applications (Access, Word, PowerPoint, Excel)**
  - **A background knowledge about Network**

- Programming Languages: (C, C++, C#, Java, Visual Basic 6.0)
-Web developing: (HTML, ASP, ASP.NET, PHP , JavaScript, VBscript)
-Web design: (FrontPage 2003, Photoshop 7, Switch MX, Flash MX2004, Action Script 2 & 3)

- **Courses attended about Database:**
  Database Methodology and Design
  SQL Server 2000 Programming
  MySQL 4.0
  Oracle 9i DBA Fundamentals I

Other Courses
- Project Research, Development and Presentations.
- Business Concepts.

Experience and Projects:

1) Information Project Group(IPG) Company
Position: Working on a variety of projects in the field of distance learning.
Environment: FlashMx2004 and Action Script 2.0
- Please check(www.ipgegypt.com/afdl.htm)

2) **Worked as a web site developer**
(Please check:**www.ramegypt.cjb.net**, **www.faculity4eg.cjb.net** , **www.linuxeg.cjb.net**)

Hobbies

Chess, Computer.

Personal Information

Date of Birth: 24/6/80
Military Status: Deferred
Marital Status: Single