

مرجع کامل

Multimedia Builder 4.9.7

The Complete Reference



Unleash Your Creativity

● محیط کاملا مبتنی بر اشیا

● برنامه نویسی سریع، آسان و درعین حال قدرتمند

● سازگاری کامل با فرمت‌های صوتی و تصویری متداول

● امکان تولید محصول در عرض چند دقیقه

● قابلیت توسعه توسط Plug-In

● قابلیت ساخت برنامه های بزرگ چند رسانه ای

● قابلیت ارتباط و استفاده از برنامه های دیگر نظیر Flash, Photoshop و...

● تولید برنامه های چندرسانه ای در عرض چند دقیقه

تالیف : حامد تکمیل

MIMB مرجع کامل

تمام حقوق محفوظ است. استفاده از مطالب با ذکر منبع بلا مانع است.

تالیف : حامد تکمیل

info@mmbstation.com

www.mmbstation.com

فهرست مطالب

۷	مقدمات
۱۷	شرح رابط کاربر
۴۱	اشیا
۷۹	جلوه های MMB
۹۰	اسکرپت نویسی
۱۱۰	ثابت های سیستمی
۱۲۲	ماکروهای مسیر
۱۳۲	عبارت شرطی
۱۳۷	حلقه های For...next
۱۴۳	آرایه
۱۴۸	جادوگر اسکرپت
۲۰۰	توابع رشته ها
۲۰۷	Plug-In
۲۵۴	تولید نهایی
۲۶۱	ضمیمه

مقدمه :

در طی ۶ سالی که با نرم افزار MMB کار می کنم، احساس می کنم که جزئی از زندگی من شده. چرا که تقریباً هر روز در حال استفاده از MMB هستم. این باعث شده که من رابطه ی بسیار خوبی با آن برقرار کنم و بتوانم مهارت ها و قابلیت های خودم را توسعه دهم. به همین منظور خود را وام دار و مدیون جامعه MMB می دانم و همین احساس مرا بر آن داشت تا اقدام به تهیه این کتاب نمایم

من برای نوشتن این کتاب در حدود ۱۱ ماه وقت صرف کردم و در این زمان از کمک بسیاری از دوستان خوبم بهره بردم که جا دارد از آنها تشکر کنم. در ابتدا من به قصد چاپ این اثر اقدام به نوشتن مطالب کردم، اما به لحاظ مشکلات بسیار و فراز و نشیب های بسیاری که پشت سر گذاشتم، تصمیم گرفتم تا آن را به صورت کتاب الکترونیکی تقدیم علاقه مندان نمایم. هم اکنون ۲ عنوان کتاب برای MMB موجود است که بنده سعی نموده ام که در این اثر مطالب بیشتری به نسبت آنها ارائه کنم تا غالب نیازهای علاقه مندان برطرف شود.

در این مرجع سعی شده است تا نیاز غالب مخاطبان برآورده شود و با ارائه مطالب ساده و پیشرفته در کنار هم به نیاز مخاطبان پاسخ گفته شده است. نوشته ای را که پیش روی خود دارید به عنوان مرجعی کامل برای برنامه مولتی مدیا بیلدر محسوب می شود. کمتر کسی است که در عرصه مولتی مدیا فعالیت داشته باشد و نام این برنامه را نشنیده باشد و یا با آن کارنکرده باشد، لذا با توجه به این مطلب و نیاز روز افزون مخاطبان (که میتوان پرسش های مکرر آنان در اینترنت را گواهی بر این مطلب دانست) بر آن شدم تا مرجعی کامل به زبان فارسی برای این برنامه مفید تهیه کنم. در این نوشته مطالب پیشرفته از قبیل برنامه نویسی به گونه ای کاملاً ساده بیان شده است و پس از پایان مطالعه مطلب از جانب مخاطبان می توان از آنان انتظار تولید هر گونه محصول چند رسانه ای را داشت.

این مرجع در ابتدا با معرفی برنامه MMB آغاز می شود و در ادامه برای آشنایی بیشتر خوانندگان خود آموزی قرار داده شده است. در ادامه به تدریج رابط کاربر برنامه مورد بررسی قرار گرفته است. پس از آن در فصلی مفصل مفاهیم برنامه نویسی و پیشرفته تر این برنامه بیان شده است. در این بخش مطالب به همراه مثال های متعددی ارائه شده است. در بخش بعدی برای پاسخگویی به نیاز کاربران حرفه ای طریقه استفاده از Plug-In های این برنامه به همراه آموزش چند Plug-In قدرتمند این برنامه آمده است. به جرات می توان گفت که تا کنون هیچ مطلبی در این رابطه به زبان فارسی برای مخاطبان ارائه نشده است. در نهایت بخش تولید نهایی و ضمیمه ارائه شده است که حاوی نکات تکمیلی و مطالبی جهت تولید بهینه محصولات می باشد. در این بخش برای تثبیت مطالب مشروح در فصل های قبلی خود آموزی جامع جهت تولید یک محصول ارائه شده است. برای کامل شدن مرجع یک مجموعه کم یاب از مثال ها و Plug-in های این برنامه بر روی یک سی دی که حاصل ۶ سال جستجو می باشد جمع آوری شده است که به مخاطبان عرضه می شود. جهت رفع نیاز های احتمالی مخاطبان پایگاهی اینترنتی به آدرس www.mmbstation.com به دو زبان فارسی و انگلیسی بر روی شبکه جهانی قرار گرفته است.

مطالب ارائه شده در این مرجع حاصل ترجمه مطالب متعدد ارائه شده و در بخش هایی تالیفی می باشند. که درصد مطالب تالیفی بیشتر می باشد. از آنجاییکه شخصا در تهیه راهنمای اصلی (زبان انگلیسی) برنامه مشارکت داشته و جز تیم بتای توسعه این برنامه می باشم روز آمد بودن و کارآمدی این مرجع را تایید نموده و آن را تقدیم مخاطبان می کنم.

از دیگر مزیت های این نرم افزار می توان به سایر نرم افزار های تکمیلی که برای این نرم افزار تولید شده اند نام برد که از جمله آنان نرم افزار RealDraw می باشد که نرم افزاری برداری جهت تولید عناصر گرافیکی می باشد و به شهادت کاربران بسیار ساده و در عین حال قدرتمند می باشد. این برنامه با ارائه خروجی مختص برنامه MMB فرآیند تولید برنامه های چند رسانه ای را بسیار آسان کرده است. همچنین این برنامه کاملا با نرم افزار Photoshop سازگار بوده که می تواند قدرتش را دو چندان کند. با توجه به این مطالب یک طراح برنامه های مولتی مدیا غالب نیاز های خود را در این مجموعه یافته و جلب آن نیز می شود.

در پایان از اینکه این مرجع را برگزیدید از شما تشکر می کنم. بی صبرانه منتظر نظرات، پیشنهادات و انتقادات شما هستم.

تمام حقوق محفوظ است. استفاده از مطالب با ذکر منبع بلا مانع است.

حامد تکمیل

زمستان ۱۳۸۴

تقدیم به :

فہمیہ راہی

مڑگان کرمی

محسن نجفی

محمد نجفی

محمد زارع

ہادی تکمیل

و ہمہ علاقہ مندان بہ برنامہ MMB

مقدمات

Multimedia Builder

◀ معرفی نرم افزار Multimedia Builder

◀ کار با MMB را تجربه کنید



در این بخش سعی شده است تا تصویری کلی از برنامه **Multimedia Builder** ارائه شود. به همین منظور ابتدا ویژگی‌های شاخص این برنامه مورد بررسی قرار می‌گیرد و در ادامه با بررسی خودآموزی روند آشنایی با برنامه تکمیل می‌شود.

معرفی نرم افزار Multimedia Builder :

MMB (مولتی میدیا بیلدر، به اختصار MMB) نرم افزاریست جهت طراحی و تولید برنامه های چند رسانه ای^۱ که مبتنی بر سیستم عامل ویندوز می باشد. اگر تا به حال توسط زبان های برنامه نویسی دیگری اقدام به تولید نرم افزارهای چند رسانه ای کرده باشید، حتما می دانید که مراحل نسبتا پیچیده ای را باید بپیمایید تا به نتیجه مطلوب برسید. و حتما در حین کار آرزو کرده اید که کاش می شد راه حل تازه تر و آسان تری را یافت. اما باید بگویم که راه حل هم اکنون پیش روی شماست و آن چیزی نیست جز MMB. نرم افزاری کوچک با قابلیت های بزرگ که کلیه ابزارهای مورد نیاز تولید برنامه های چند رسانه ای را در اختیار شما قرار می دهد. با گذشت زمان و کار با این برنامه خواهید فهمید که به آن علاقه مند شدید و علل آن هم در سادگی کار با MMB و قدرتمندی آن نهفته است. حال جهت آشنایی بیشتر به توضیح اجمالی توانایی ها و خصوصیت های MMB می پردازیم:

- یادگیری آسان و سریع:

شما به عنوان یک طراح برنامه های چند رسانه ای با استفاده از MMB و بدون صرف هزینه و وقت جهت دوره های آموزشی این توانایی را خواهید یافت که دست به تولید و توسعه برنامه های چند رسانه ای مورد نیاز خود بزنید.

- برای خود CD های AutoRun (خود اجرا) بسازید

- چه نوع برنامه های چند رسانه ای را می توان با استفاده از MMB تولید نمود؟

- CD های AutoRun
 - خودآموزهای کامپیوتری چند رسانه ای
 - کارت ها، راهنما ها و بروشورهای محصولات
 - CD های صوتی به همراه برنامه های اجرای آنها
 - برنامه های چند رسانه ای مورد نیاز شخصی یا اداری
 - برنامه های کوچک کاربردی تحت ویندوز
- در فرآیند تولید یک محصول چند رسانه ای چه امکاناتی در اختیار طراح است؟
- محیط WYSIWYG (What You See Is What You Get)
 - محیط طراحی مبتنی بر اشیا
 - سازگاری در بارگذاری و استفاده از فرمت های متداول گرافیکی از قبیل GIF, SWF و ...
 - جلوه های ویژه برای تغییر تصاویر
 - امکان اسکرپت نویسی برای کنترل جزئیات بیشتر

۱- به برنامه هایی گویند که از طریق پخش صوت و تصویر با کاربر ارتباط برقرار می کنند.

- تولید پنجره نهایی پروژه به شکل دلخواه در زمان طراحی و اجرا
- طراحی اشیا در لایه های مختلف
- Plug-In های متنوع و قدرتمند جهت تولید برنامه های قویتر

در نهایت با استفاده از MMB می توانید تولیدات خود را به صورت یک فایل اجرای مستقل (*.exe) در آورده و مورد استفاده قرار دهید.

چه کسانی باید این کتاب را بخوانند؟

کتاب حاضر برای کسانی است که هیچ تجربه ای در ساختن نرم افزارهای چند رسانه ای ندارند. البته کاربران حرفه ای نیز می توانند به جمع خوانندگان این کتاب بپیوندند، چرا که راهی جدید را برای تولید برنامه های مورد نظر خود خواهند یافت. این کتاب به عنوان مرجعی برای برنامه MMB به حساب می آید. این کتاب ابتدا بر روی اصول اولیه متمرکز می شود و سپس به سراغ مباحث پیشرفته از جمله Plug-In ها می رود. حداقل معلوماتی که برای شروع کار با MMB مورد نیاز است، آشنایی با سیستم عامل ویندوز می باشد.

کار با MMB را تجربه کنید!

مطمئناً هنوز MMB برای شما ناشناخته است، به همین منظور خود آموز تولید یک برنامه ساده را در ادامه قرار داده ایم تا به کنکاش و کسب تجربه بپردازید، مسلماً قضاوت شما پس از انجام این خود آموز درباره MMB نظر نهایی شما در باره MMB خواهد بود. خوب آماده شوید تا عملاً کار با MMB را تجربه کنید. در این خود آموز فرا می گیرید که چگونه برنامه Slideshow را که برای مشاهده عکس ها می باشد، بنویسید.

- قبل از هر چیز لازم است تا برنامه MMB را نصب نمایید. چنانچه اینطور نیست می توانید برای نصب آن به CD همراه کتاب و یا سایت www.mediachance.com مراجعه نمایید.

- در آغاز برنامه MMB را اجرا نمایید.

۱. ابتدا توسط گزینه Add Page از منوی Page دو صفحه ایجاد نمایید.

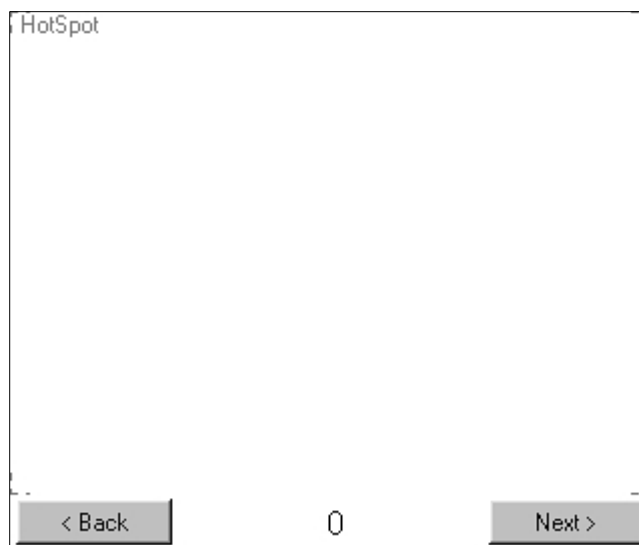
۲. از منوی Project گزینه Project Settings را برگزینید. سپس از قسمت Window Size برای تعیین عرض و ارتفاع پنجره به ترتیب برای عرض (Width) عدد ۳۲۰ و برای ارتفاع (Height) عدد ۲۷۰ وارد نمایید. در قسمت Window Title نیز عبارت My first project in MMB را وارد نمایید و نهایتاً دکمه OK را جهت تایید تنظیمات فشار دهید. برای ذخیره پروژه از منوی File گزینه Save را انتخاب نمایید و پروژه را با نام MyProject ذخیره نمایید.

۳. در اولین صفحه توسط منوی Object->Create گزینه Rectangular HotSpot را برگزینید و توسط موس محدوده این شی را در صفحه تعیین نمایید. در حالی که شی HotSpot هنوز در حالت انتخاب قرار دارد (با کلیک بر روی شی، شی به حالت انتخاب در می آید)، برای تعیین موقعیت قرار گیری و اندازه شی از منوی View گزینه Dimensions را انتخاب کنید. در کادری که نمایان می شود به ترتیب از بالا به پایین اعداد ۰،۰،۳۲۰ و ۲۴۰ را وارد نمایید و Enter را فشار دهید. سپس کادر Dimensions را ببندید.

۴. از منوی Object->Create توسط گزینه Text Button دو دکمه در صفحه ایجاد کنید. به ترتیب با ۲ بار کلیک بر روی هر دکمه به قسمت خصوصیات آنها رفته و در قسمت Text هر کدام جهت تعیین عنوان دکمه عبارت های < Back و > Next را وارد کنید.

۵. از منوی Object شی Text را برگزینید و آن را در صفحه ایجاد نمایید. با ۲ بار کلیک بر روی شی متن (Text) به قسمت خصوصیات آن بروید و در قسمت ویرایشگر آن عدد صفر را تایپ نمایید. و سپس با زدن کلید OK از کادر خصوصیات خارج شوید.

۶. اکنون همانند تصویر ، موقعیت دو دکمه و سایر اشیا را تنظیم کنید. لازم به ذکر است که اشیا در MMB در هنگام طراحی هم توسط صفحه کلید و هم توسط موس قابل جا به جایی هستند.



شکل ۱-۱

۷. پس از اینکه موقعیت اشیا را تنظیم نمودید از منوی Edit گزینه ی Select All را انتخاب کنید تا تمام اشیا موجود در صفحه به حالت انتخاب در بیایند. در حالی که تمام اشیا در حال انتخاب هستند کلید های Ctrl+C را جهت کپی برداری از اشیا فشار دهید. سپس به صفحه دومی که ایجاد کرده اید بروید (از پنل Pages در پایین رابط کاربر برنامه MMB می توانید برای نقل مکان بین صفحات استفاده نمایید). در حالی که در صفحه دوم قرار دارید کلید های Ctrl+V را جهت چسباندن اشیا به درون صفحه فشار دهید. با این عمل ظاهر صفحه دوم نیز همانند صفحه اول می شود.

۸. در پنل Page دو بار بر روی آیکن صفحه اول کلیک نمایید تا به قسمت خصوصیات شی منتقل شوید. در کادر خصوصیات صفحه در قسمت Page Transition از لیست موجود گزینه Alpha Blending را انتخاب کنید. توسط قسمت Page Transition می توان جلوه های بصری جالبی را به هنگام تعویض صفحات در موقع اجرای برنامه ایجاد نمود.

۹. در پنجره خصوصیت صفحه از قسمت اسکریپت (Script) دکمه Action On Page Start را فشار دهید تا ویرایشگر اسکریپت ظاهر شود. پس از آن در قسمت ورود متن ویرایشگر، کد زیر را تایپ نمایید:

```
If (i$>'') Then
  LoadText ("Text", "i$")
```

```

End
If (i$='') Then
    i=i+1
    i$=CHAR(i)
    name$='<SrcDir>\images\\' + i$ + '.jpg'
    ReplaceImage("HotSpot", "name$")
    LoadText("Text", "i$")
End

```

سپس با زدن OK از ویرایشگر خارج شوید و در ادامه پنجره خصوصیت را با زدن کلید OK ببندید. از اینکه مفهوم کدهای نوشته شده در این خود آموز را نمی فهمید نگران نباشید. در قسمت های بعدی به تفصیل راجع به آنها بحث می شود.

۱۰. عملیات های قسمت های ۸ و ۹ را برای صفحه دوم نیز انجام دهید با این تفاوت که در قسمت ورود متن ویرایشگر اسکرپت صفحه دوم کد زیر را بایستی تایپ کنید:

```

If (i$>'') Then
    LoadText("Text", "i$")
End

```

۱۱. به صفحه ی اول باز گردید و با ۲ بار کلیک بر روی دکمه با عنوان <Back> به قسمت خصوصیات آن بروید. در کادر ظاهر شده از قسمت Action گزینه More Action (اولین دکمه از سمت راست) را برگزینید تا ویرایشگر اسکرپت نمایان شود. از زیر پنجره ویرایشگر گزینه Mouse Up را انتخاب کنید تا ویرایشگر مربوط به رویداد Mouse Up ظاهر شود، نهایتاً کد زیر را در قسمت ورود داده های رویداد Mouse Up وارد نمایید:

```

i=i-1
If (i<1) Then
    i=3
End
i$=CHAR(i)
name$='<SrcDir>\images\\' + i$ + '.jpg'
ReplaceImage("Page 2::HotSpot", "name$")
DisplayValue("Text", "i")
NextPage()

```

در ادامه با فشردن کلید OK از پنجره ویرایشگر Script خارج شوید و بار دیگر با زدن OK از پنجره خصوصیت دکمه خارج شوید.

۱۲. در صفحه اول به قسمت Actions دکمه با عنوان > Next بروید و در پنجره رویداد Mouse Up کد زیر را تایپ کنید:

```

i=i+1
If (i>3) Then
    i=1
End
i$=CHAR(i)
name$='<SrcDir>\images\' + i$ + '.jpg'
ReplaceImage("Page 2::HotSpot", "name$")
DisplayValue("Text", "i")
NextPage()

```

و در آخر با ۲ بار فشردن کلید OK به ترتیب از پنجره های ویرایشگر و خصوصیت دکمه خارج شوید.

۱۳. عملیات های قسمت های ۱۱ و ۱۲ را نیز برای دکمه های Back و Next صفحه دوم نیز انجام دهید با این تفاوت که کدهای :

```
ReplaceImage("Page 2::HotSpot", "name$")
```

به

```
ReplaceImage("Page 1::HotSpot", "name$")
```

و

```
NextPage()
```

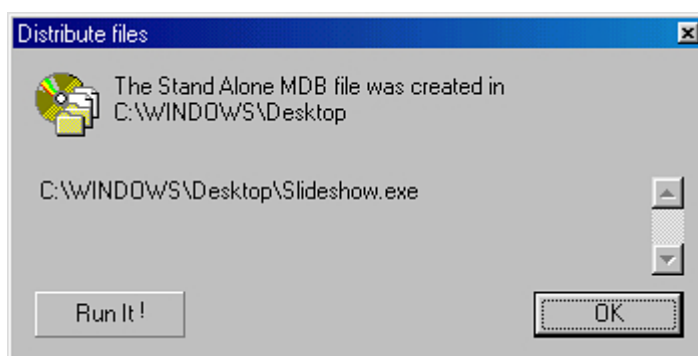
به

```
PrevPage()
```

تغییر می کنند.

۱۴. حال بر روی دیسک سخت خود یک پوشه به نام images ایجاد کنید و چند عکس را با پسوند JPG که نام آنها از عدد یک الی آخر می باشد را در آن پوشه کپی نمایید. مثلاً چنانچه ۳ عکس داشته باشیم نام آنها به ترتیب 1.jpg، 2.jpg و 3.jpg می شود.

۱۵. از منوی File گزینه Compile را جهت تولید نهایی و کامپایل برگزینید. از کادر بعدی که ظاهر می شود در قسمت Distribute to Destination توسط کلید Locate مسیر را برای ذخیره فایل نهایی مشخص نمایید و در ادامه دکمه OK را فشار دهید. هنگامیکه فایل توسط MMB کامپایل شد کادری همانند شکل ظاهر می شود که بایستی دکمه OK را از آن فشار داد.



۱۶. اکنون کافیست تا پوشه images را که قبلا ایجاد کردید به کنار فایل اجرایی که در مرحله قبل تولید شد ، منتقل شود تا برنامه به درستی عمل کند. به محل پوشه images بروید و آن را به کنار فایل اجرایی که در مرحله ۱۵ تولید شده است منتقل کنید.

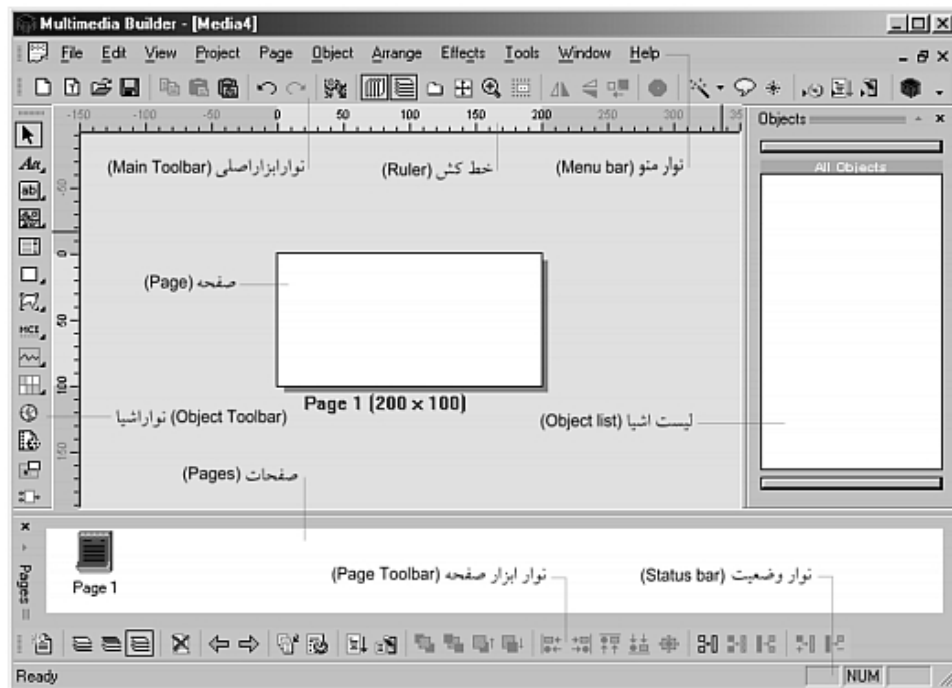
۱۷. اکنون برنامه شما آماده است تا مورد استفاده قرار گیرد. به همین منظور فایل تولید شده را اجرا نمایید و از حاصل کار لذت ببرید.

این بخش اختصاص به بررسی رابط کاربر (User Interface) برنامه Multimedia Builder دارد. یادگیری این قسمت به مثابه ی زیر بنای روند یادگیری تلقی می شود.

هنگامیکه تصمیم به تولید یک برنامه ی چند رسانه ای می گیرید اولین قدم در تولید آن نرم افزار ایجاد ظاهر آن می باشد. در اینجاست که رابط کاربر MMB به کمک ما می آید تا بتوانیم با بهره گیری از عناصر ارائه شده به تکمیل پروژه بپردازیم. البته باید یاد آور شد که این رابط تنها برای ساخت ظاهر برنامه سود مند نمی باشد بلکه روند کد نویسی برنامه نیز توسط همین رابط صورت می گیرد. به طور کلی مهمترین ویژگی ای که رابط کاربر به ارمغان می آورد سهولت دسترسی به ابزارها و استفاده از آنها می باشد. در این کتاب روند بررسی رابط کاربر را ابتدا با بررسی منوهای برنامه شروع می کنیم، سپس به سراغ ابزارهای کنترل و ویرایش اشیا می رویم. در ادامه تنظیمات پروژه مورد بررسی قرار می گیرد تا به بررسی صفحه ها در MMB برسیم. در نهایت اشیا ارائه شده در MMB معرفی می شوند.

شرح رابط کاربر MMB (User Interface)

اولین گام در یادگیری هر نرم افزاری، یادگیری چگونگی عملکرد رابط کاربر آن نرم افزار می باشد. به همین منظور در این قسمت می خواهیم به شرح رابط کاربر MMB بپردازیم. رابط MMB نسخه 4.9.7 به اندازه کافی قدرتمند می باشد و فرآیند کاری بسیار ساده و موثری را فراهم می کند. به کمک این رابطه پیشرفته کار کردن با MMB را آسان می یابید. شکل زیر رابط MMB و نواحی عمده را نشان می دهد.



شکل ۱-۲

ابتدا با بررسی منوهای رابط کاربر MMB شروع می کنیم.

«منوی File»

New

برای تولید یک سند جدید استفاده میشود.

New From Template

با استفاده از این گزینه می توانیم براساس قالب های ارائه شده قالبی را انتخاب کرده و آن را تا حدودی ویرایش نموده و از آن استفاده نماییم. مزیت این قسمت در داشتن wizard جهت تکمیل مرحله به مرحله قالب میباشد.

Open

برای بازگشایی یک سند با فرمت MBD میباشد.

Close

این گزینه سبب بسته شدن سند جاری میشود.

Save

جهت ذخیره سند جاری بر روی دیسک میباشد.

Save As

جهت ذخیره یک کپی دیگر از سند جاری بر روی دیسک میباشد.

Compress & Export

گاهی اوقات پیش می‌آید که شما مایلید تنها به نشر فایل منبع یعنی فایل *.mbd* بپردازید. برای مثال آن را بر روی شبکه برای دسترسی دیگران قرار دهید. با استفاده از این قابلیت میتوانید فشردگی فایل و حجم فایل خروجی را تغییر دهید.

Compile

جهت کامپایل^۱ نهایی پروژه میباشد.

Reduce size

این گزینه حجم ها و اقلام اضافی پروژه را حذف می نماید.

Import object

ممکن است در حین طراحی یک پروژه تمایل داشته باشید از یک شیء که قبلاً درجایی دیگر و در زمانی دیگر طراحی گردیده است استفاده نمایید، به همین منظور میتوانید از این گزینه استفاده نمایید. البته پسوند شیء ها obm میباشد.

Export object

برای صدور یک قسمت از پروژه مان که مایلیم آن را ذخیره نماییم و در دفعات بعد از آن استفاده کنیم مناسب می باشد؛ در دفعات بعد توسط گزینه Import object می توان آنها را به درون پروژه وارد کرد.

MEF import

از این گزینه برای بارگذاری فایل های MEF که توسط برنامه Real-DRAW تولید می شود استفاده میشود. MEF مخفف (Meta Export Format) می باشد

۱- فرآیندی است که طی آن برنامه نوشته شده به زبان ماشین ترجمه می شود .

Webcam Image

همانطور که از نام این گزینه پیداست می توان تصاویری را که توسط web cam گرفته شده را بارگذاری نمود.

Import page

برای وارد کردن یک صفحه که از قبل صادر شده است استفاده می شود.

Export page

چنانچه بخواهیم میتوانیم صفحه پروژه را با پسوند MMP صادر نماییم تا در دفعات بعد بتوانیم مکرراً از آن استفاده کنیم.

Print

این قسمت جهت چاپ صفحه جاری پروژه میباشد (در زمان طراحی)

Print preview

برای بازبینی قبل از چاپ میباشد

Print setup

این گزینه جهت تنظیم پارامترهای چاپ می باشد.

Exit

سبب خروج از برنامه MMB می شود.

«منوی Edit»

Undo

سبب لغو آخرین تغییر اعمال شده می شود.

Redo

سبب تکرار آخرین تغییری که به واسطه فرمان Undo ایجاد شده می شود.

Copy

جهت تهیه یک کپی از شیء یا اشیاء و ارسال آنها به Clipboard^۱ می باشد.

۱-ابزاری است که عملیات کپی و چسباندن فایل ها و پوشه ها را در ویندوز مدیریت می کند.

Copy Incremental

این گزینه جهت تهیه چندین کپی از یک شیء می‌باشد. بدین صورت که با انتخاب یک شیء و برگزیدن این ویژگی تنها با کلیک کردن بر روی شیء و کشیدن، کپی جدیدی از شیء اصلی ایجاد می‌شود.

Paste

جهت چسباندن شیء یا اشیاء از clip board به پروژه MMB می‌باشد.

Paste bitmap

چنانچه عکسی در clip board باشد توسط این گزینه می‌توان آن را در صفحه پروژه چسباند.

Delete

جهت پاک کردن شیء یا اشیاء در حال انتخاب می‌باشد.

Hide / Show

جهت مخفی کردن / نمایان ساختن شیء یا اشیاء درون صفحه در هنگام طراحی می‌باشد.

Clone graphics

این قسمت برای تهیه یک کپی از شیء یا اشیاء گرافیکی (عکس‌ها) می‌باشد. تفاوت بین این قسمت با قسمت Copy در این است که چنانچه شیء مادر (اصلی) را تغییر دهیم بلافاصله همان تغییر در اشیاء کپی برداری شده (فرزند) اعمال می‌شود.

Select all

جهت انتخاب تمام اشیاء موجود در صفحه می‌باشد.

Snap to grid

جهت پرش اشیاء بر اساس شبکه‌بندی از پیش تنظیم شده می‌باشد.

Edit grid

با این گزینه می‌توان شبکه‌بندی را تغییر داد.

Snap to Guides

Guides (راهنما) خطی است که با کلیک موس بر روی خط‌کش MMB و کشیدن آن به پایین و یا به سمت راست تولید می‌شود. از این خط جهت بهبود عمل ترازبندی استفاده می‌شود. حال با برگزیدن گزینه مذکور اشیاء به این خط راهنما پرش می‌کنند.

Edit Guides

جهت ویرایش و تنظیم خط‌های راهنما می‌باشد

«منوی View»

Object list

این گزینه لیست اشیاء موجود را در محیط طراحی نشان می‌دهد و یا حذف می‌کند.

Page list

این گزینه لیست صفحات را که در پایین محیط طراحی قرار دارد مخفی و یا نمایان می‌سازد.

Toolbars

از این گزینه برای تنظیم نمایش مولفه‌های نوار ابزار استفاده می‌شود.

Status bar

از این گزینه جهت نمایان ساختن یا مخفی کردن نوار وضعیت استفاده می‌شود.

Master page

مشخص می‌کند که آیا صفحه جاری صفحه اصلی باشد یا خیر.

Quick object

این گزینه اشیاء موجود در دایوها را که قابلیت افزوده شدن به پروژه را دارند نمایان می‌سازد، که در نهایت با انتخاب شیء و کشیدن آن به محیط طراحی توسط موس، آن شیء به محیط طراحی افزوده می‌شود.

Dimensions

از این ابزار جهت تغییر اندازه و موقعیت اشیاء استفاده میشود.

Zoom

برای بزرگ نمایی محیط کار و طراحی جهت تسلط بیشتر و در عین حال دیدن جزئیات صفحه از این ابزار استفاده می‌شود.

«منوی Project»

Project Settings

از این قسمت جهت تنظیم پاره‌ای از ویژگی‌های پروژه نظیر اندازه صفحات، نوار عنوان، شکل صفحات، عکس پس زمینه و... استفاده می‌شود.

Path replace

یک ابزار کارآمد دیگر جهت اعمال درست مسیرها در هنگام تولید و کامپایل نهایی پروژه می‌باشد.

Text replace

از این گزینه برای ویرایش و تنظیم و یا تصحیح کلی متون به کار رفته در پروژه استفاده می‌شود.

Check & distribute

از این گزینه جهت کامپایل نهایی پروژه استفاده می‌شود.

Embedded¹ files

از این قسمت برای به انضمام در آوردن سایر فایل‌ها استفاده می‌شود.

Comments

چنانچه بخواهیم فایل منبعی را منتشر کنیم می‌توانیم توضیحاتی را به آن اضافه کنیم تا کاربران دیگر در هنگام بارگذاری فایل منبع از آن توضیحات مطلع شود.

Debug

از این گزینه جهت رفع عیب و بررسی جزئیات پروژه در هنگام اجرا استفاده می‌شود.

Embedded sound

برای به انضمام در آوردن فایل‌های صوتی استفاده می‌شود.

Run

از این گزینه جهت اجرای پروژه در زمان طراحی استفاده می‌شود.

«منوی Page»

Add page

صفحه‌ی جدیدی را به پروژه اضافه می‌کند.

۱- عملی است که طی آن فایل یا فایل‌هایی به بدنه اصلی پروژه الحاق می‌یابد.

Insert

این گزینه صفحه‌ای جدید را به صفحات پروژه اضافه می‌کند با این تفاوت که می‌توان صفحه‌ی جدید را به قبل و یا بعد از آخرین صفحه موجود انتقال داد.

Delete page

صفحه منتخب را از پروژه حذف می‌کند.

Page manager

این قسمت کادری را جهت مدیریت قرارگیری صفحات ارائه می‌دهد.

Next page

در زمان طراحی به صفحه بعد می‌رود.

Previous page

در زمان طراحی به صفحه قبل می‌رود.

Master page

این گزینه سبب رفتن به صفحه‌ی اصلی می‌شود.

Master top layer

این گزینه سبب رفتن به صفحه اصلی از بالاترین لایه می‌شود.

Background

از این گزینه جهت تنظیم عکس پس زمینه صفحات استفاده می‌شود.

Debug

از این قسمت جهت رفع عیب و کنترل جزئیات صفحه منتخب جاری در زمان اجرا استفاده می‌شود.

Test current page

این قسمت برای اجرا و امتحان کردن صفحه جاری می‌باشد.

Check for lost object

این گزینه بر اساس تنظیمات جاری بررسی می‌کند که آیا شی‌ای در صفحه جاری گم شده است یا خیر.

Properties

این گزینه کادر خصوصیت هر صفحه را نشان می‌دهد.

«منوی Object»

Create

با کلیک بروی این گزینه لیست تمامی اشیاء موجود ظاهر می شود که می توان بر اساس نیاز یکی را برگزید.

Convert to bitmap

این گزینه منتهای منتخب را به عکس تبدیل می کند. استفاده از این گزینه هنگامی مفید است که شک داشته باشیم نوع قلم (Font) مورد نظر ما در کامپیوتر کاربر نهایی موجود است.

Clone bitmap object

از این گزینه جهت تهیه یک نسخه از عکس منتخب جاری استفاده می شود با این تفاوت که اعمال هر تغییر بر روی نسخه اصلی سبب تغییر نسخه های کپی شده هم میشود.

Actions

این قسمت جهت دسترسی به بخشهایی می باشد که آن قسمت ها حاوی فعالیتی هستند. قسمت هایی از قبیل Script, Sound, Interactions, Command & page

Properties

جهت دسترسی به کادر خصوصیت شیء منتخب می توان از این گزینه استفاده کرد.

«منوی Arrange»

Nudge

با استفاده از زیرشاخه های این گزینه می توان سبب تغییر مکان شیء شد.

Order

چنانچه بخواهیم یک شیء را بین لایه ها حرکت دهیم این گزینه به کمک ما می آید.

Align

برای تراز کردن موقعیت قرارگیری اشیاء از این قسمت استفاده می شود.

Combine

این گزینه اشیاء را با پس زمینه ادغام می کند و یک مجموعه واحد به عنوان پس زمینه تشکیل می دهد.

Group

برای گروه بندی اشیاء است.

Ungroup

جهت ختنی کردن گروه بندی است.

Add to group

می توان با استفاده از این گزینه اشیاء جدیدی را به گروه اضافه کرد.

Remove from group

سبب می شود تا شیء یا اشیاء منتخب از گروه خارج شود.

Group manger

این گزینه کادر مدیریت گروه ها را نمایان می سازد.

«منوی Effects»

Flip

جهت چرخاندن اشیاء در راستاهای افقی، عمودی به مقدار 180° می باشد.

Blur

برای ایجاد جلوه بلور در عکسها می باشد. ابزار blur لبه ها و حاشیه های سخت و تیز تصویر را نرم و صاف می کند.

Sharpen

این گزینه جلوه Sharpen را به عکس ها اعمال میکند. از جلوه Sharpen برای سخت و تمیز کردن (واضح کردن) لبه ها و حاشیه های صاف و نرم موجود در تصویر استفاده میشود.

Convert to gray scale

به عکس جلوه سیاه و سفید با ته مایه خاکستری می بخشد.

Tile

سبب ایجاد جلوه ای همانند کاشی ها می شود.

Crop

با استفاده از این ابزار میتوان قسمتی از عکس منتخب را برش داد.

MMB Effects

این قسمت برای اعمال جلوه هایی نظیر سایه گذاری برجسته سازی براق سازی و ... می باشد.

Special Effects

این گزینه قسمتی از جلوه‌هایی را که می‌توان بر عکسها اعمال کرد ارائه می‌دهد. افرادی که نحوه کار با photoshop را می‌دانند با این گزینه آشنا هستند.

Inetractive Effects

این گزینه هم برای اعمال چند جلوه ویژه دیگر به عکس ها می‌باشد. این جلوه ها قابلیت پویایی و حرکت را دارند.

Color Tweak Effects

جهت تغییر رنگ اشیایی که با کادر خصوصیت آنها نمی‌توان رنگ آنها را تنظیم کرد می‌توان از این گزینه استفاده کرد. یادآور می‌شویم که این گزینه فراتر از تنها تغییر رنگ عمل می‌کند.

Restore original

چنانچه عکسی را تغییر داده باشیم با اعمال این گزینه می‌توانیم تغییرات اعمال شده را حذف کنیم.

Make new original

چنانچه به یک عکس تغییراتی را اعمال کنیم و مایل باشیم یک نسخه از همان عکس تغییر یافته به عنوان نسخه اصلی تهیه کنیم از این گزینه استفاده می‌کنیم.

«منوی Tools»

Designers Settings

این گزینه انتخابهایی را جهت تنظیم برخی از خصوصیات صفحه طراحی از قبیل اجرا کننده، تنظیم صدا و... ارائه می‌دهد.

Grid setting

برای تنظیم شبکه‌بندی می‌باشد.

Skin

MMB به همراه خود 4 پوسته (Skin) جالب برای رابط کاربر خود ارائه می‌دهد که توسط این گزینه قابل تعویض می‌باشند.

Smart Expanding Menus

کاربران ویندوز XP ویا ۲۰۰۰ با این گزینه آشنا هستند. این گزینه سبب می‌شود تا منوها به صورت هوشمند عمل نمایند. بدین ترتیب که مولفه‌هایی را که مکرراً استفاده می‌شود نشان می‌دهد و بقیه مولفه‌ها را به صورت مجتمع در می‌آورد.

«منوی Window»

New window

سبب تولید یک پنجره جدید در محیط طراحی می‌شود.

Cascade

این گزینه پنجره‌ها را به صورت آبشاری مرتب می‌کند.

Tile

این قسمت سبب قرارگیری پنجره‌ها به صورت کاشی گونه می‌باشد.

«منوی Help»

Help topic

این گزینه راهنمای MMB را نمایان می‌سازد.

About MMBuilder

این گزینه کادر درباره برنامه را نشان می‌دهد که حاوی اطلاعاتی از قبیل اطلاعات ثبت برنامه می‌باشد. چنانچه اطلاعات ثبت برنامه MMB از قبیل نام و کد رمز را در اختیار داشته باشید، می‌توانید از این قسمت نسبت به ثبت برنامه اقدام نمایید.

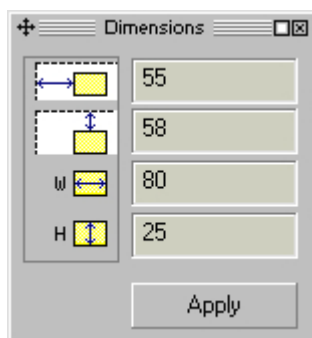
Tip of the day

انتخاب این گزینه سبب میشود تا هر بار که MMB را اجرا می‌کنیم کادری حاوی نکته‌ای نمایان شود.

ابزارهای کنترل ویرایش

بدیهی است که در زمان طراحی نیاز مند ابزارهایی جهت تغییر موقعیت و اندازه ویا مدیریت قرار گیری اشیا می باشیم. بنابراین قبل از فراگیری سایر قسمت ها لازم است تا این ابزارها را بشناسیم.

ابزار Dimensions



شکل ۲-۲

این ابزار برای تغییر موقعیت و تغییر اندازه شی می باشد.

ابزار Zoom tool



شکل ۲-۳

از این ابزار جهت بزرگ نمایی محیط کار و طراحی به خاطر داشتن تسلط بیشتر و در عین حال برای دیدن جزئیات استفاده می شود. چنانچه بر روی پنجره این ابزار راست کلیک کنیم با منویی مواجه خواهیم شد که قسمت های منو به شرح ذیل است.

Zoom factor –

این قسمت درصد بزرگ نمایی را مشخص می نماید.

Refresh period –

این گزینه دوره تازه سازی تصویر را مشخص می کند که این دوره در واحد میلی ثانیه می باشد.

Floating window –

چنانچه این گزینه را انتخاب کنیم پنجره ابزار Zoom به دنبال نشانه گر موس حرکت میکند.

Fixed window –

این قسمت جهت ثابت نگه داشتن پنجره ابزار Zoom میباشد.

Lock / Unlock view –

با استفاده از این ویژگی می توان ابزار Zoom را بر روی محدوده ای خاص ثابت نگه داشت و با تغییر مکان نما ابزار Zoom دیگر قسمتها و محدوده های دیگر را بزرگ نمایی نمی کند.

Show / Hide grid –

این گزینه شبکه بندی را بر روی ابزار Zoom نشان می دهد و یا محو می کند.

Grid color –

با این ویژگی می توان رنگ شبکه بندی را تعیین کرد.

ابزار Align

در تولید پروژه در MMB به دفعات پیش می آید که بخواهیم اشیاء حاضر در صحنه تراز بندی شوند یا در صف خاصی در راستایی مرتب شوند. برای این منظور ابزار Align به کمک ما می آید. ویژگی Align متشکل از قسمتهای زیر است:



Left –

اشیاء را نسبت به سمت چپ آنها تراز بندی می کند. مبنای این تراز اولین شیء انتخاب شده است.



Right –

اشیاء را نسبت به سمت راست آنها تراز بندی می کند. مبنای این تراز اولین شیء انتخاب شده است.



Top -

اشیاء را نسبت به سمت بالای آنها ترازبندی می کند. مبنای این تراز اولین شیء انتخاب شده است.



Bottom -

اشیاء را نسبت به سمت پایین آنها ترازبندی می کند. مبنای این تراز اولین شیء انتخاب شده است.



Center -

این قسمت یک مرکز مجازی برای اشیاء انتخاب شده در نظر می گیرد سپس همگی آنها را به آن مرکز منتقل می کند.

ابزار گروه بندی (Grouping)

گروه بندی کردن یک ترفند کارآمد جهت ایجاد سهولت در امر طراحی و پیش برد پروژه می باشد به همین منظور این ویژگی نیز در MMB گنجانده شده است.



Group -

اشیاء منتخب را به شکل یک گروه در می آید.



Ungroup -

گروه بندی را خنثی می کند.



Remove from group -

این قسمت عضو انتخاب شده گروه را از گروه خارج می کند.



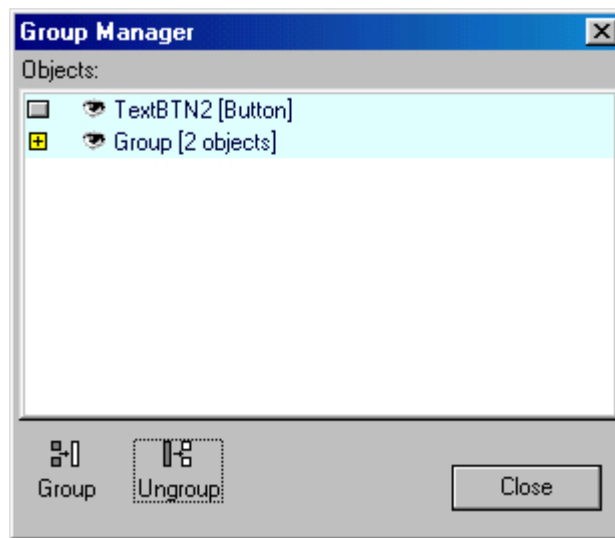
Add to group -

شیء یا اشیاء در حال انتخاب را به گروه منتخب جاری اضافه می کند.



Group manger -

این ویژگی لیستی از گروه های موجود را ارائه می دهد. که می توان آنها را خنثی و یا ادغام کرد.



شکل ۲-۳

ابزار Order

بعضی اوقات در هنگام طراحی ممکن است چند شیء بر روی شیء دیگری قرار گیرند یا برعکس زیر یک شیء. به همین سبب برای تغییر دادن قرارگیری اشیاء بر روی هم و تغییر لایه های قرارگیری اشیاء این ویژگی پیش بینی شده است.



Bring to front -

این ویژگی شیء منتخب را به بالاترین سطح انتقال می دهد یعنی بر روی تمام اشیاء.



Send to Back -

این ویژگی شیء منتخب را به پایین ترین سطح انتقال می دهد یعنی به زیر تمام اشیاء موجود.



Bring Forward -

این قسمت شیء منتخب را به یک لایه بالاتر انتقال می دهد.

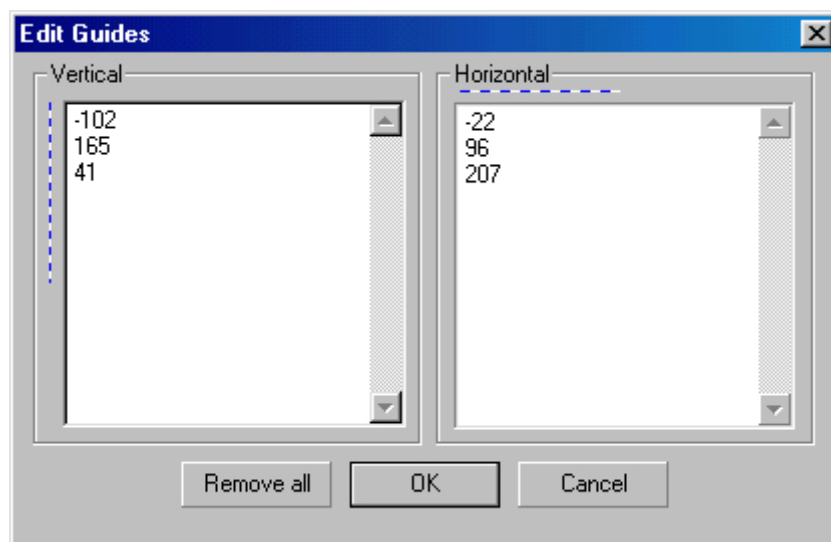


Bring Backward -

این قسمت شیء منتخب را به یک لایه پایین تر انتقال می دهد.

ابزار Guide

Guide (راهنما) خطی است که با کلیک موس بر روی خطکش MMB و کشیدن آن به پایین و یا به سمت راست تولید می شود. از این خط جهت بهبود عمل ترازبندی استفاده می شود. پس از آن با نزدیک کردن اشیاء به این خطوط اشیاء به این خطها پرش می کند. در MMB هیچ محدودیتی برای تعداد این خطوط وجود ندارد.



شکل ۴-۲

Edit Guide –

جهت ویرایش و تنظیم خطهای راهنما می باشد.

Snap to Guide –

چنانچه این گزینه فعال باشد خطوط راهنما عملاً قابل استفاده هستند. در غیر این صورت پرش به خطوط صورت نمی گیرد.

ابزار Grid

این ابزار کارآمد یک شبکه بندی خاص را بر روی صفحه پروژه در هنگام طراحی ایجاد می کند و هدف از آن باز ایجاد سهولت در امر جابه جایی اشیاء و ترازبندی آنها بر اساس این شبکه ها می باشد.

Snap to Grid –

جهت پرش اشیاء بر اساس شبکه بندی به منظور ترازبندی اشیاء می باشد.

Edit Grid –

با این گزینه می‌توان شبکه‌بندی را تغییر داد.

ابزار Flip

این گزینه جهت چرخاندن عکس‌ها در راستای افقی، عمودی، به مقدار 180° می‌باشد.



Vertically –

شکل‌ها را در راستای عمودی به میزان 180° درجه می‌چرخاند.

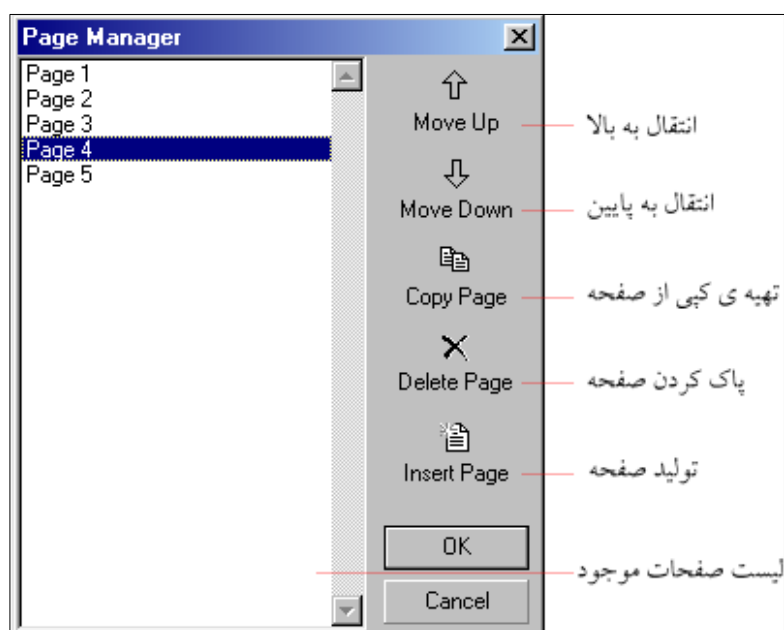


Horizontally –

شکل‌ها را در راستای افقی به میزان 180° درجه می‌چرخاند.

ابزار Page managers

این ابزار برای مدیریت ترازبندی و تولید صفحات می‌باشد. این ابزار هنگامیکه پروژه ما از صفحات متعددی تشکیل شده است بسیار کارآمد و مفید می‌باشد.



شکل ۲-۵

تنظیمات پروژه «Project Setting»

تولید هر پروژه‌ای نیاز به یک سری تنظیمات کلی و اولیه دارد. به همین منظور قسمت Project Setting برای انجام این تنظیمات کلی در اختیار ماست. لازم به تذکر است که تنظیمات موجود بر کل صفحات پروژه اعمال می‌شوند. به عنوان مثال ۲ صفحه در یک پروژه نمی‌توانند دو اندازه‌ی مختلف داشته باشند. برای ایجاد تمایز در برخی از خصوصیات صفحه می‌توانیم از اسکریپت نویسی و فرامین موجود بهره‌بریم. در ادامه به شرح جزئیات این قسمت می‌پردازیم.

Windows size

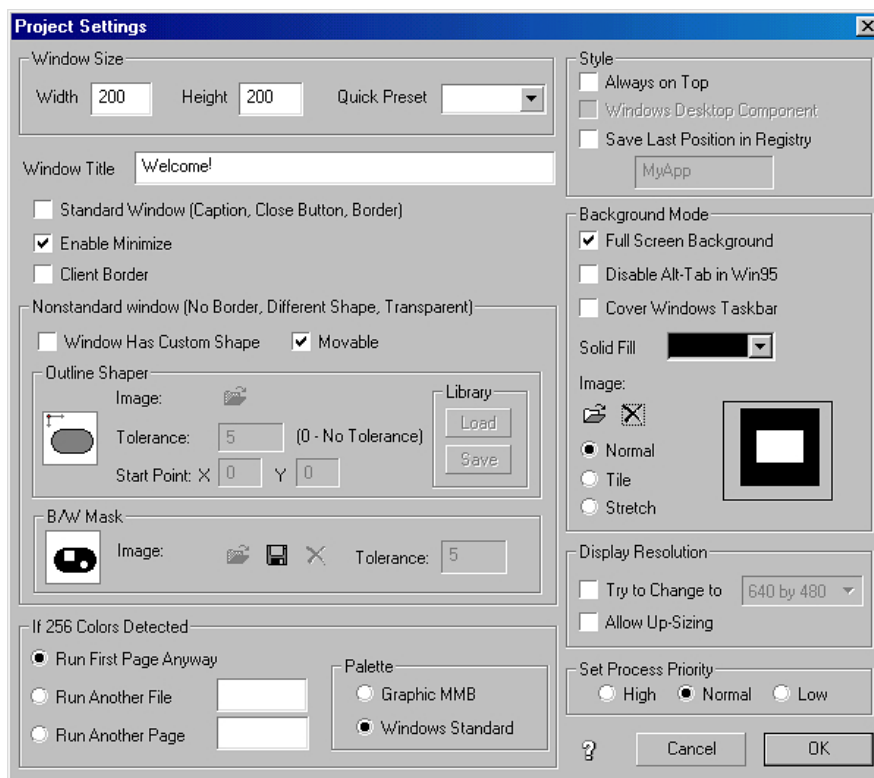
این قسمت اندازه پنجره برنامه را تعیین می‌کند. Width برای عرض و Height برای ارتفاع است. همچنین در قسمت Quick Preset چند اندازه که معمولاً استفاده می‌شود موجود است.

Windows Title

برای تعیین عنوان پنجره برنامه می‌باشد.

Standard Windows (Caption, Close Button, Border)

شکل ۶-۲



در صورت فعال بودن این گزینه پنجره برنامه به صورت یک پنجره‌ی معمولی ویندوز ظاهر می‌شود. این پنجره همانند سایر پنجره‌ها می‌تواند دارای عنوان (Caption) دکمه بستن (Close Button) و حاشیه (Border) باشد.

Enable Minimize

این گزینه قابلیت Minimize شدن را فعال می‌کند.

Client Border

این گزینه حاشیه‌ای مشکی و باریک را به پنجره برنامه اضافه می‌کند.

Nonstandard Window

همان طور که می‌دانید در حالت عادی پنجره برنامه حالت استاندارد یک پنجره‌ی ویندوز را دارد. اما توسط این قسمت می‌توانیم شکل پنجره را به شکل دلخواه خود تغییر دهیم، برای مثال به دایره. این گزینه در صورت غیر فعال بودن گزینه Standard Windows فعال می‌شود. اما برای تغییر ظاهر برنامه ابتدا باید عکس را که مایلیم پنجره به همان صورت در آید بارگذاری کنیم سپس می‌توانیم مختصات قرارگیری این عکس را نسبت به منتهی علیه سمت چپ و بالای پنجره برنامه جهت نتیجه بهتر تغییر دهیم. لازم به ذکر است که بهترین نتیجه هنگامی حاصل می‌شود که عکس ما حاشیه‌هایی واضح و پس زمینه‌ای یکدست داشته باشد. قسمت Library هم برای ذخیره پنجره‌ی جدیدی که تولید شده است می‌باشد تا در دفعات بعد هم بتوانیم از این پنجره استفاده کنیم. در صورت تمایل می‌توانید با استفاده از Tweak PlugIn که در قسمت Plug-In شرح داده شده است شکل پنجره را در هنگام اجرا مکرراً تغییر دهید.

B/W Mask

توسط این قسمت می‌توانیم تصویری را بارگذاری کنیم و از آن به عنوان ماسکی بر روی صفحه استفاده کنیم. به عنوان مثال فرض کنید که تصویری داریم که در آن دایره‌ای قرار دارد و ضمناً این عکس زمینه‌ای یکدست دارد. چنانچه این تصویر را بر روی پنجره برنامه قرار دهیم تنها قسمتی از پروژه که زیر دایره و در محدوده‌ی دایره است قابل دید است. باید گفت که با استفاده ماهرانه از این قابلیت می‌توان پروژه‌هایی زیبا تولید نمود.

If 256 Color Detected

چنانچه پروژه ما با مد ۲۵۶ رنگ ناسازگار باشد می‌توانیم به هنگام مواجهه با مد ۲۵۶ رنگ فایل دیگری (Run another file) ، صفحه‌ای دیگر از پروژه (Run another page) و یا اجرای خود برنامه تحت هر شرایطی (Run first page anyway) را برگزینیم تا پروژه به اجرای خود ادامه دهد.

Palette

در این بخش می‌توانیم مشخص کنیم که برنامه ما از پلت رنگ ویندوز (Windows standard) استفاده کند و یا از پلت رنگ MMB (Graphic MMB).

Style

در این قسمت می‌توان نوع قرارگیری پنجره برنامه را اعم از:

Always on Top –

همیشه بر بالا .

Windows Desktop Component -

این گزینه پنجره برنامه را همیشه بر روی دسکتاپ ویندوز اجرا می‌کند .

Save Last Position in Registry -

آخرین موقعیت قرارگیری پنجره در صفحه را در رجستری^۱ تحت نامی که برمی‌گزینیم ذخیره می‌کند.

Background Mode

در صورت تمایل می‌توانیم برای پروژه پس زمینه‌ای را در نظر بگیریم. البته این پس زمینه با پس زمینه هر صفحه متمایز است. در این حالت عکس پس زمینه در زیر پنجره برنامه و در کل پروژه قابل رؤیت است. در حالی که هر صفحه می‌تواند پس زمینه منحصر به خودش را داشته باشد.

Fullscreen Background

امکان داشتن پس زمینه تمام صفحه را ارائه می‌دهد. برای این عمل کافیست از قسمت Image دکمه Load Image (تصویر) را فشار دهیم.

Disable Alt – Tab in win 95

همان طور که می‌دانید با فشردن همزمان کلیدهای Alt و Tab در ویندوز پنجره‌ای نمایان می‌شود که حاوی لیستی از برنامه‌های در حال اجرا می‌باشد که توسط این لیست می‌توانیم بین برنامه‌ها نقل مکان نماییم. حال انتخاب این گزینه این قابلیت را غیر فعال می‌سازد.

Cover Windows Taskbar

انتخاب این گزینه در هنگام اجرای برنامه نوار وظیفه را مخفی نگه می‌دارد.

Solid Fill

می‌توانیم به جای عکس برای پس زمینه کل پروژه، از رنگ‌ها استفاده کنیم.

Display Resolution

توسط این قسمت می‌توانیم بر اساس نیاز معین کنیم که وضوح تصویر (Resolution) در هنگام اجرای برنامه چقدر باشد.

۱- محلی است که اطلاعات مورد نیاز سیستم در آنجا ذخیره و نگهداری می‌شود. هرگونه تغییر در این بخش منجر به تغییر قسمت مربوط در ویندوز می‌شود.

Allow Up-Sizing

این انتخاب برای سازگاری برنامه در هنگام تغییر وضوح تصویر با سخت افزارهای قدیمی است.

Set Process Priority

این بخش مقدار استفاده برنامه از CPU را تعیین می‌کند، گزینه Low (پایین) سبب می‌شود که برنامه به مقدار کم از CPU استفاده نماید یا Normal (معمولی) باعث استفاده به صورت معمولی می‌شود و High (بالا) استفاده از CPU را تا حد امکان افزایش می‌دهد.

۱- توصیه می‌شود تنها در صورت نیاز از این بخش استفاده نمایید.

صفحه (Page)

محلی است که حاوی گروهی از اشیا به همراه توابع آنها می باشد تا در نهایت عملیات مورد نظر صورت گیرد. بنابراین صفحات برای سازماندهی و تقسیم نیازهای پروژه می باشد.

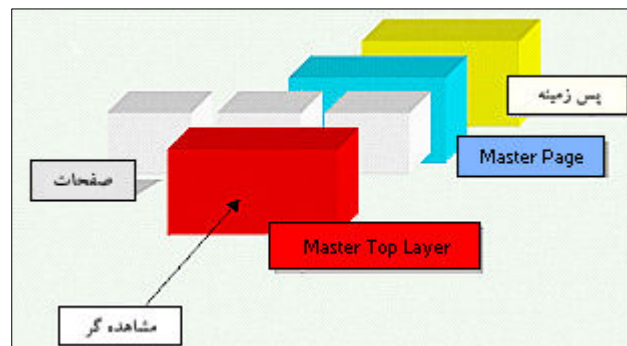
ممکن است پروژه ها تنها متشکل از یک صفحه باشند، اما استفاده از چندین صفحه به سهولت طراحی و کارایی بیشتر پروژه کمک می کند. یکی از قابلیت های کارآمدی که صفحات به ارمغان می آورند اینست که به راحتی می توان در زمانهای بعد از تولید پروژه، اقدام به به روز رسانی صفحات کرد. اما باید توجه داشت که استفاده زیاد از صفحات سبب افزایش حجم برنامه می شود که این افزایش حجم در همه جا خوشایند نیست، مثلاً ممکن است در نظر داشته باشید پروژه را در اینترنت انتشار دهید و چنانچه حجم پروژه بالا باشد سبب بروز مشکلاتی می شود. برنامه MMB همچنین ۲ صفحه ی ویژه را برای افزایش قابلیت ها و سهولت در امر طراحی ارائه می دهد.

Master Page –

تمامی اشیا در این صفحه در کل صفحات پروژه قابل دید هستند. این صفحه در زیرترین لایه بعد از پس زمینه قرار دارد.

Master Top Layer –

تمامی اشیا در این صفحه نیز در کل صفحات پروژه قابل دید هستند. این صفحه در بالا ترین لایه قرار دارد.



شکل ۷-۲

۱ و ۲- این دو صفحه از نوار ابزار صفحه (Page Toolbar) قابل دسترسی می باشند.

رویدادهای صفحه

رویداد یعنی اینکه حالتی به وقوع بپیوندد. به عنوان مثال عملکرد پدال گاز در وسیله نقلیه را در نظر بگیرید. هنگامیکه پدال گاز توسط راننده فشار داده می شود بر سرعت وسیله نقلیه افزوده می شود. فشردن پدال گاز که منجر به انجام عملی می شود در رویداد داین مثال گویند. صفحات در MMB دارای ۲ رویداد هستند:

- Page Start :

زمانی است که صفحه اجرا می شود.

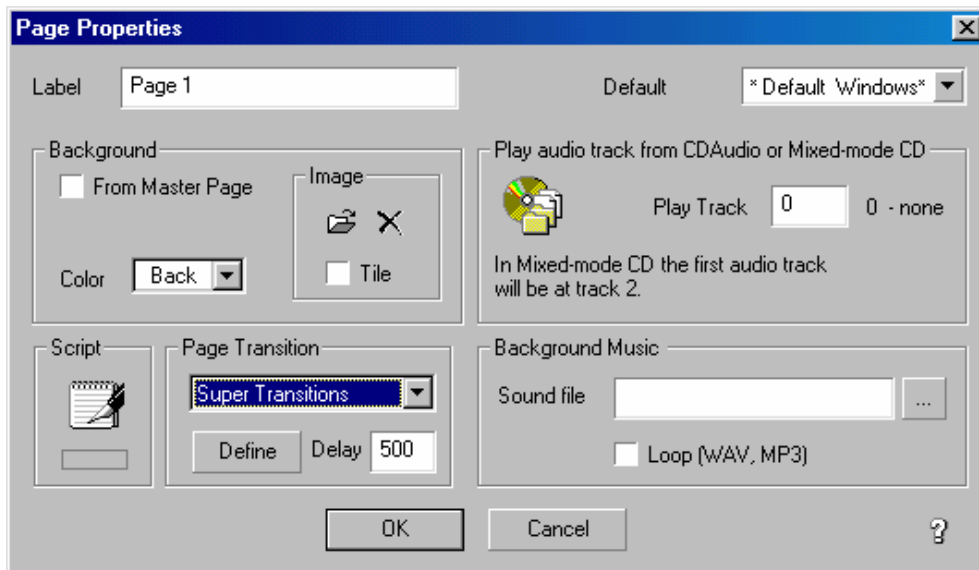
- Page Exit :

هنگامی است که از صفحه خارج می شویم.

بنابراین می توان با استفاده از این رویدادها کد نویسی را انجام داد، بدین صورت که می توانیم به هنگام به وقوع پیوستن رویدادها به برنامه انجام عملی را نسبت دهیم.

خصوصیات صفحه

خصوصیت های هر صفحه از قسمت Page Properties و یا با ۲ بار کلیک بر روی آیکن هر صفحه در پنل صفحه قابل دسترسی است.



شکل ۸-۲

Label

نام صفحه را مشخص می کند.

Background

این قسمت برای مشخص کردن پس زمینه صفحه می باشد. چنانچه گزینه **From Master Page** فعال باشد عکس پس زمینه همانند پس زمینه **Master Page** می شود. قسمت **Image** برای برگزیدن عکس می باشد و **Color** نیز برای تعیین رنگ می باشد.

Script

جهت نوشتن اسکریپت می باشد.

Page Transition

برای معین کردن **Transition** (واسط) می باشد. **Transition** جلوه ای بصری است که به هنگام تغییر صفحات نمایان می شود.

Default

نمایشگر موس (**Cursor**) صفحه را مشخص می کند.

Play Audio track from CD Audio or mixed mode CD

این قسمت یکی از آهنگ های **CD** صوتی را به عنوان آهنگ زمینه انتخاب می کند.

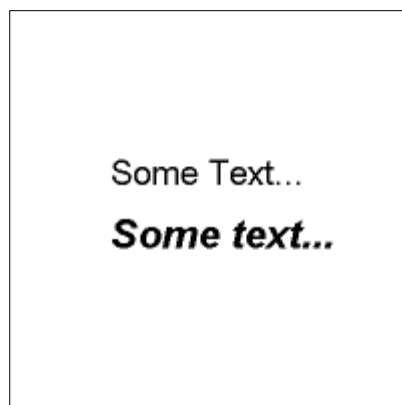
Background music

برای معین کردن آهنگ زمینه می باشد. گزینه **LOOP** جهت ایجاد تکرار می باشد.

اشیا (Objects)

اشیا در MMB عناصر تشکیل دهنده ی پروژه هستند، به همین منظور MMB طیف وسیعی از اشیا و خصوصیات آنها را ارائه می دهد. در این قسمت قصد داریم تا به بررسی اشیا موجود در MMB به همراه خصوصیات آنها پردازیم. لازم به تذکر است که در پایان این قسمت خصوصیات را که در همه اشیا مشترک می باشند توضیح داده ایم تا از تکرار پرهیز کنیم. در MMB برای ایجاد اشیا کافیست که آن شی را از لیست اشیا و یا نوار ابزار (Toolbar) برگزینیم و در نهایت با عمل کشیدن و رها کردن موس شی مربوطه را ایجاد کنیم. برای دسترسی به خصوصیات اشیا نیز دو راه موجود است: ۱- با دو بار کلیک بر روی شی ۲- گزینه Properties از منوی Object هنگامیکه شی در حالت انتخاب قرار دارد. در ادامه به بررسی اشیا ارائه شده به همراه ویژگی های آنها می پردازیم.

• Text (متن)

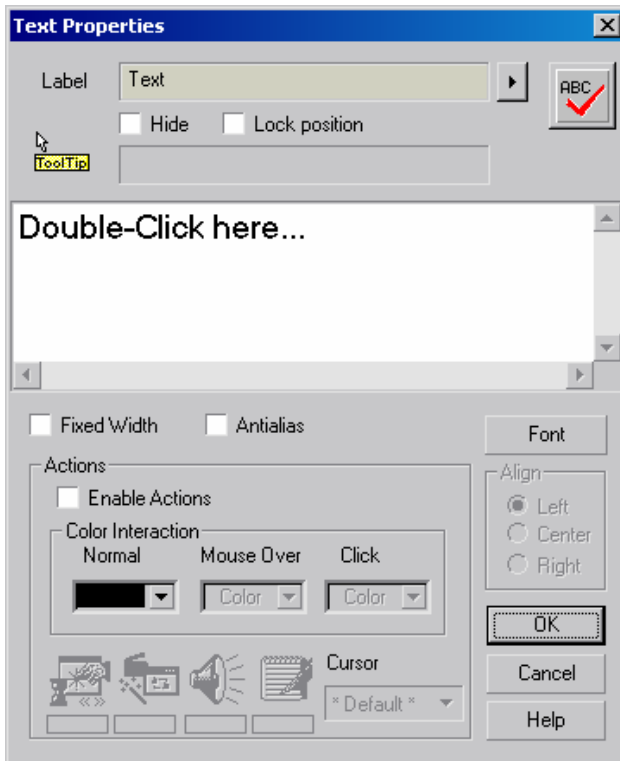


شکل ۹-۲

این شی برای نمایش دادن متون، کلمات، عبارات و ... مورد نظر شما می باشد. در ضمن این شی می تواند به صورت فعال باشد.

نکته: منظور از فعال اینست که شی می تواند حاوی اسکریپتی جهت انجام عملی باشد.

خصوصیت‌های شی Text



شکل ۱۰-۲

Color Interaction (تعامل رنگ)

توسط این ویژگی می‌توانیم مشخص کنیم که بر اساس وضعیت موس نسبت به شی متن، متن چه رنگی شود.

در این قسمت ۳ حالت:

۱- **Normal** (معمولی) به هنگامی که موس بر روی متن قرار ندارد، یا کلیک نکند.

۲- **MouseOver** (موس بر روی متن) هنگامی است که نشانه گر موس بر روی شی متن قرار می‌گیرد.

۳- **Click** (کلیک) هنگامی که موس بر روی متن کلیک کند.

پیش‌بینی شده است.

Align (ترازبندی)

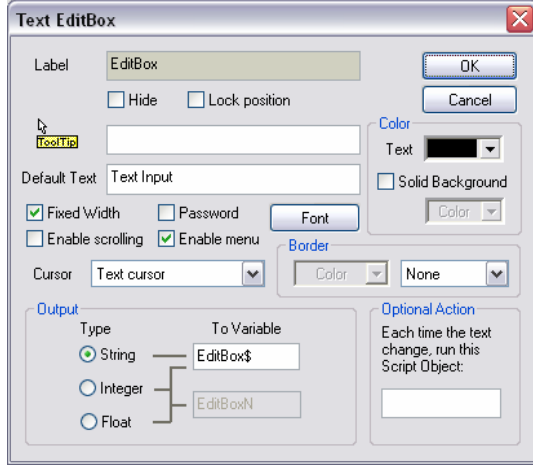
این ویژگی برای تراز کردن متن است.

• EditBox (کادر ورود داده)

این شی این امکان را فراهم می‌کند که اطلاعاتی را از کار گرفته و از آن در برنامه‌ی خود استفاده کنید. البته شی ارائه شده در MMB نسخه 4.9.7 برخی از ضعف‌های نسخه‌های قبلی را از میان برده است.

- خصوصیت‌های شی (EditBox)

- Default Text (متن پیش فرض)



توسط این کادر می‌توانید متن پیش فرض را برای شیء EditText در هنگامی که پروژه اجرا می‌شود تنظیم نمایید.

Color (رنگ)

با استفاده از این قسمت رنگ زمینه (Solid Background) و رنگ متن (Text) را تعیین می‌کنیم.

Out put (خروجی)

برای تعیین متغیری است که بایستی اطلاعات وارده شده به آن متغیر نسبت داده شود. بر اساس نیاز می‌توان نوع متغیر را به یکی از حالات String (رشته‌ای) Integer (صحیح) و یا نوع Float که شامل هر دو نوع متغیر است تغییر داد. متغیرها در قسمت بعد مفصلاً شرح داده خواهند شد.

تذکر: برای نمایش اعداد با تعداد ارقام بیشتر از ۶ سعی کنید از نوع Float استفاده نمایید.

Fixed Width

این گزینه سبب می‌شود که طول شیء Editbox در همه حال ثابت باشد.

Password

توسط این گزینه حروف به صورت رمزی و یکسان نمایان می‌شوند.

Enable Menu

جهت فعال سازی / غیر فعال کردن منوی Right Click به هنگام اجرا می‌باشد.

Optional Action (فعالیت دلخواه)

چنانچه بخواهیم به هنگام تغییر متن کادر ورود داده‌ها، فعالیت‌های خاصی صورت گیرد و یا اسکریپتی اجرا شود می‌توانیم از این قسمت استفاده کنیم. تنها کافیست که در کادر زیر این قسمت نام شیء اسکریپت را معین کنیم. چنانچه در هنگام مطالعه اشیا توضیحات مربوط به برخی از قسمت‌های آن‌ها را نیافتید می‌توانید در ادامه و در بخش خصوصیت مشترک بین اشیا درباره آنها مطالعه نمایید.

• Paragraph Text (پاراگراف)

برای نمایش دادن متن های بلند خود می توانید از این شی استفاده کنید. با انتخاب شی و کشیدن کادر مورد نظر در صفحه و رفتن به قسمت خصوصیات شی می توانیم متن خود را وارد کنیم.

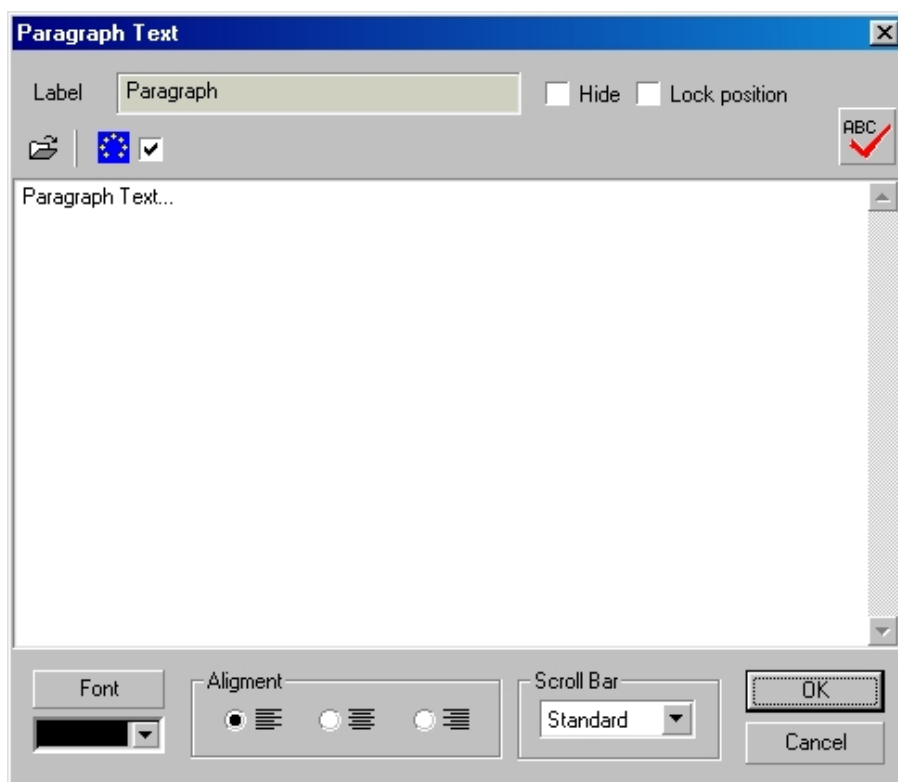
خصوصیت های شی Paragraph

International Character Editor (ویرایشگر کاراکتر بین المللی)

این گزینه نوع ویرایشگر متن را تغییر می دهد.

Spell Checking (بررسی واژه ها)

چنانچه بخواهیم متن خودمان را از نظر غلط های املایی (تنها زبان انگلیسی) بررسی کنیم می توانیم از این ویژگی استفاده کنیم.



شکل ۱۲-۲

Alignment (ترازبندی)

جهت تراز کردن متن می باشد.

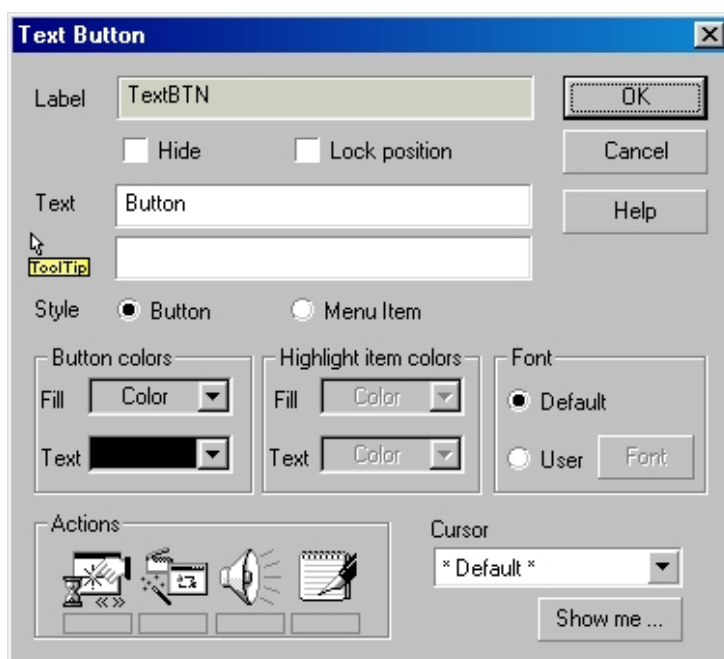
Scroll Bar (نوار نمایش)

اگر متن مورد نظر ما تماماً در کادر مورد نظر ننگنجد به طور خود کار نوارهای پیمایش در کنار کادر ظاهر می شود. برای شی Paragraph، نوع نوار پیش نمایش معمولی و تخت (Flat) موجود می باشد.

• Text Button (دکمه‌ی متنی)

این شی برای ساختن دکمه‌های استاندارد ویندوزی می باشد. ضمناً می توان رنگ و اندازه‌های این دکمه‌ها را تغییر داد، یا آنها را به گونه‌ای تنظیم کرد که همانند منوها عمل کنند.

- خصوصیت‌های شی Text Button



شکل ۱۳-۲

Style (سبک)

توسط این قسمت می توان نوع دکمه را تعیین کرد. ۲ نوع موجود است:
۱- Button (دکمه استاندارد) ۲- Menu Item (همانند منو)

Button Colors (رنگ دکمه)

این قسمت برای تعیین رنگ دکمه (Fill) و رنگ متن (Text) می باشد. این قسمت در صورتی فعال می باشد که نوع دکمه همان حالت استاندارد باشد.

Highlight Item Colors

چنانچه نوع دکمه Menu Item باشد. می توان رنگ دکمه (Fill) و متن (Text) را تنظیم نمود.

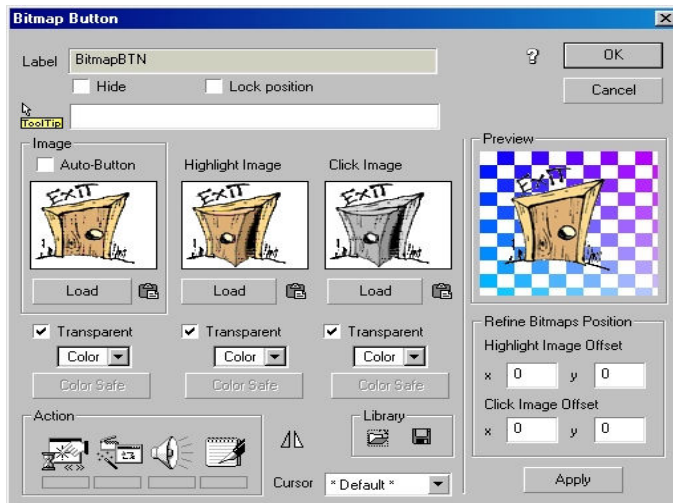
• Bitmap Button (دکمه تصویری)

این گزینه به شما این امکان را می دهد که یک دکمه متشکل از ۳ عکس داشته باشید. (عکسی که در حالت عادی می بینید، در زمانی که موس بر روی آن قرار می گیرد و زمانی که بر روی آن کلیک می کنید) و یا اینکه شما می توانید با انتخاب گزینه Auto Button یک دکمه بسازید که فقط از یک عکس تشکیل شده باشد. مهمترین مزیت این شی همان داشتن فرم ۳ حالتی است.

- خصوصیت های شی Bitmap Button

Image (تصویر)

این قسمت برای تعیین ۳ عکس دکمه می باشد. قاب سمت چپ برای عکس پیش فرض، قاب وسط برای حالتی که موس بر روی دکمه واقع می شود (Highlight) و قاب سوم برای حالتی که با موس بر روی شی کلیک می کنیم (Click).



Preview

(بازبینی)

این قسمت جهت بازبینی دکمه می باشد.

Transparent

با استفاده از این گزینه می توان رنگ پس زمینه ی عکسها را حذف کرد. مبنای شناسایی رنگ زمینه، پیکسل پایینی و منتهی علیه سمت چپ می باشد.

شکل ۱۴-۲

Auto Button

چنانچه بخواهیم دکمه ما تنها متشکل از یک عکس باشد، می توان از این گزینه استفاده کرد.

Color Safe

جهت رعایت موارد رنگ از نظر رنگ‌هایی که قابلیت نمایان شدن در اینترنت و... را دارند می‌توان از این گزینه استفاده کرد.

Refine Bitmap Position

اگر جهت تشکیل دکمه تصویری، عکس یا عکس‌هایی مناسب برای ۳ قسمت دکمه در اختیار نداشته باشیم و یا نخواهیم از ۳ تصویر استفاده کنیم می‌توانیم در حالت‌های (Highlight) و (Click) به گونه‌ای عمل کنیم که تصویر آن قسمت‌ها تغییر مکان جزئی و محسوس بدهد. قسمت **Highlight Image offset** برای حالت **Highlight** و قسمت **Click Image offset** برای هنگام کلیک کردن است. در استفاده از این ویژگی تنها کافیست تا مقدار تغییر مکان را در کادرهای مربوطه وارد نمائیم.

Library

گاهی اوقات پیش می‌آید که می‌خواهیم دکمه‌ای را که ایجاد کرده‌ایم برای استفاده‌های آتی ذخیره کنیم. برای این امر می‌توانیم از **Library** استفاده کنیم. حال چنانچه بخواهیم دکمه‌های از پیش ذخیره شده را بارگذاری کنیم بایستی بر روی دکمه‌ی **Load Button** (شکل) کلیک کنیم.

Flip Horizontal

با این گزینه می‌توان تصاویر را در راستای افقی 180° درجه چرخاند.

• Alpha Button (دکمه آلفا)

یک نوع مخصوص از دکمه است که به وسیله برنامه‌ای به نام **Real-DRAW** محصول شرکت **Mediachance®** تولید می‌شود. مزیت این دکمه‌ها در کیفیت و سادگی آنهاست. در این دکمه‌ها هم می‌توان ۳ حالتی را که شی **Bitmap Button** دارا بود را در اختیار داشت. داشتن ویژگی **Transparency** از دیگر مزیت‌های این نوع دکمه است.

– خصوصیت‌های شی Alpha Button

Load *.3sb button

برای بارگذاری دکمه‌های **Alpha** می‌توان از این دکمه استفاده کرد.

• Bitmap (بیت مپ)

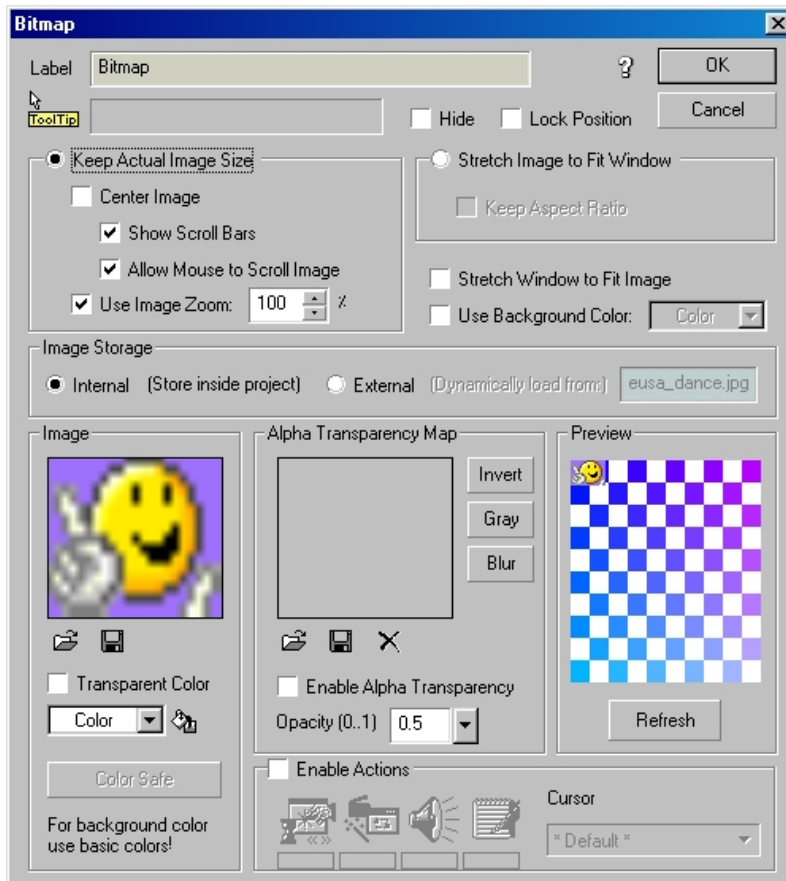
این شی برای نشان دادن تصاویر گرافیکی در پروژه می باشد. این شی می تواند به صورت فعال و غیر فعال باشد. Bitmap از ۲ قسمت تشکیل شده است: ۱- Bitmap-۲ Alpha Transparency map

این شی می تواند از فرمت های TIFF, PNG, PCX, GIF, JPG, BMP حمایت کند.

خصوصیت های شی Bitmap

Keep Actual Image size

این قسمت اندازه واقعی تصویر را حفظ می کند و دارای مولفه های زیر است:



:Center Image

تصویر را در وسط کادر مورد نظر Bitmap نمایش می دهد.

Show Scroll bars

چنانچه تصویر از محدوده ای که برایش در نظر گرفته شده بزرگتر باشد نوارهای پیمایش ظاهر خواهند شد.

Allow Mouse to scroll Image

در صورت فعال بودن این گزینه عمل پیمایش را می توان توسط موس انجام داد.

Use Image Zoom

می‌توان تصویر را بزرگ‌نمایی یا کوچک‌نمایی کرد.

Stretch Image to fit windows

این قسمت برای ایجاد کشیدگی در تصویر جهت تطبیق در محدوده مشخص شده برای عکس می‌باشد.

Image storage

می‌توانیم مشخص کنیم که آیا تصویر به همراه فایل اجرایی الحاق شود یا در کنار فایل اجرایی در پوشه‌ای مخصوص قرار گیرد. گزینه‌ی Internal برای الحاق فایل‌هاست و External جهت ذخیره در خارج از فایل اجرایی می‌باشد.

Stretch windows to fit Image

اگر تصویر ما از محدوده‌ای مشخص شده برای تصویر بزرگتر باشد با استفاده از این گزینه می‌توان به MMB فهماند که پنجره را با عکس تطبیق دهد.

Use Background Color

جهت برگزیدن رنگ زمینه برای محدوده‌ی تصویر می‌باشد.

Image

این بخش تصویر بارگذاری شده را نشان می‌دهد. در این قسمت نیز ویژگی Transparency وجود دارد.

Alpha Transparency map

در صورت فعال بودن این قسمت تصویر مربوطه بر اساس مقدار غلظت (opacity) حالت شفافیتی به خود می‌گیرد. حتی می‌توان از تصویری دیگر برای اعمال حالت شفافیت استفاده کرد. دکمه‌های Blur / Gray / Invert به ترتیب برای معکوس کردن حالت / اعمال یک پوشش خاکستری / جلوه بلور به کار می‌روند.

• Animated GIF (متحرک)

با انتخاب این گزینه می‌توانید فایل GIF متحرک را وارد پروژه خود کنید. فایل GIF معمولاً در قسمت برنامه‌نویسی کنترل می‌شود. در قسمت تنظیمات Animated GIF می‌توانید توسط گزینه‌های Transparency)

شفافیت)، Loop (تکرار)، Speed (سرعت) و... کنترل اجرای GIF را بهبود بخشید. این شی به پویایی پروژه شما کمک شایانی می‌کند.

فایل های GIF متحرک در اینترنت بسیار رایج و مشهور هستند. به راحتی می‌توانید با جستجو در اینترنت اقدام به تهیه آنها نمایید یا اینکه یک Clipart CD را تهیه نمایید. نحوه کارکردن GIF هم همانند سایر اشیاء می‌باشد. برای مثال می‌توان آنها را گروه بندی نمود، حرکت داد و...

چنانچه توسط اسکریپت GIF پنهان شده را نمایان سازید نمایش فایل GIF از فریم اول آن آغاز می‌شود. توسط اسکریپت نویسی کنترل بیشتری بر روی GIF خواهید داشت

نکته: فایل های GIF متحرک احتیاج به استفاده ی بیشتری از CPU نسبت به GIF های ثابت دارند. چنانچه یک GIF بر روی یک Bit map دارای Alpha Transparency قرار گیرد اجرای GIF کندتر صورت می‌گیرد. بنابراین از قرار دادن مقادیر زیادی فایل GIF متحرک در پروژه تا حد امکان خودداری کنید. فایل های GIF متحرک برای بارز ساختن قسمتی ویژه مناسب می‌باشد!

-خصوصیات شی Animated GIF

Info (اطلاعات)

این قسمت حاوی اطلاعاتی درباره فایل GIF می‌باشد.

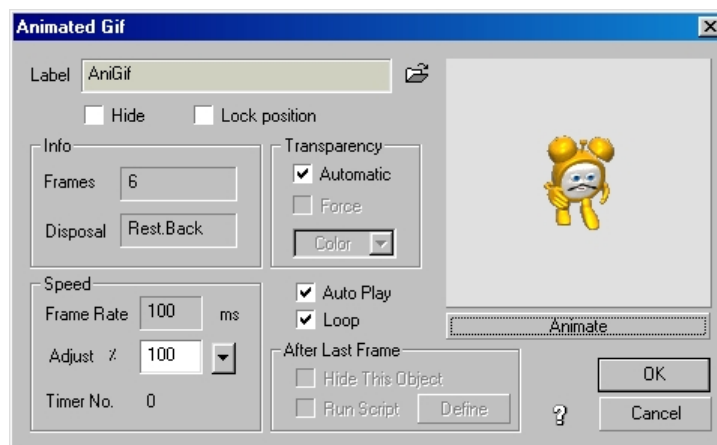
Transparency

برای حذف رنگ زمینه تصویر متحرک استفاده می‌شود این عمل را می‌توان هم به صورت خودکار (Automatic) و هم با اعمال نیرو (Force) به صورت دستی انجام داد.

Speed

تنظیم سرعت نمایش فایل GIF در این مکان صورت گیرد.

Auto play



با اعمال این گزینه در هنگام اجرای پروژه فایل GIF خودکار اجرا می شود.

Loop

سبب تکرار اجرای تصویر GIF می شود.

After Last Frame

در این مکان می توان مشخص کرد که پس از پایان اجرای GIF آیا شی پنهان شود (Hide the Object) و یا اسکریپتی (Run Script) اجرا شود.

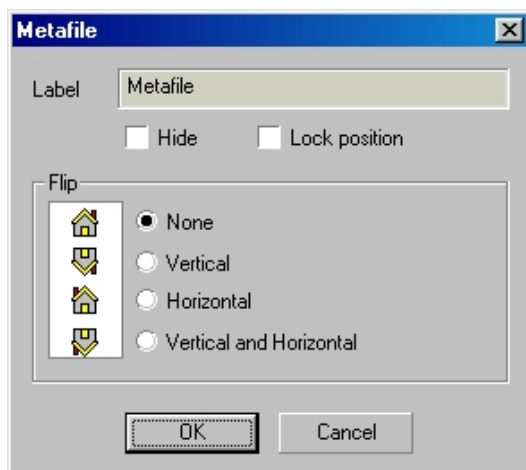
Animate

جهت بازبینی فایل GIF می باشد.

Meta File •

Windows Meta file (WMF) یک فرمت مشهور برداری می باشد. شما می توانید با استفاده از ابزارهای طراحی (از قبیل Corel draw) فایل هایی برداری تولید نمایید. فایل های با فرمت wmf به راحتی قابلیت تغییر مقیاس بدون تغییر کیفیت را در اختیار کاربر قرار می دهند. MMB از این فرمت پشتیبانی می نماید با این توضیح که فعلاً در نسخه ی جاری این شی غیر فعال است و حتی خصوصیات منحصر به فردی هم همانند سایر اشیاء ندارد. جهت قرار دادن wmf در صحنه ابتدا، بایست از منوی object گزینه Meta file را برگزیند. سپس با کشیدن موس محدوده wmf را مشخص کرده در ادامه کادری جهت بارگذاری فایل wmf آشکار می شود که می توان فایل دلخواه را بارگذاری نمود.

– خصوصیات شی MetaFile

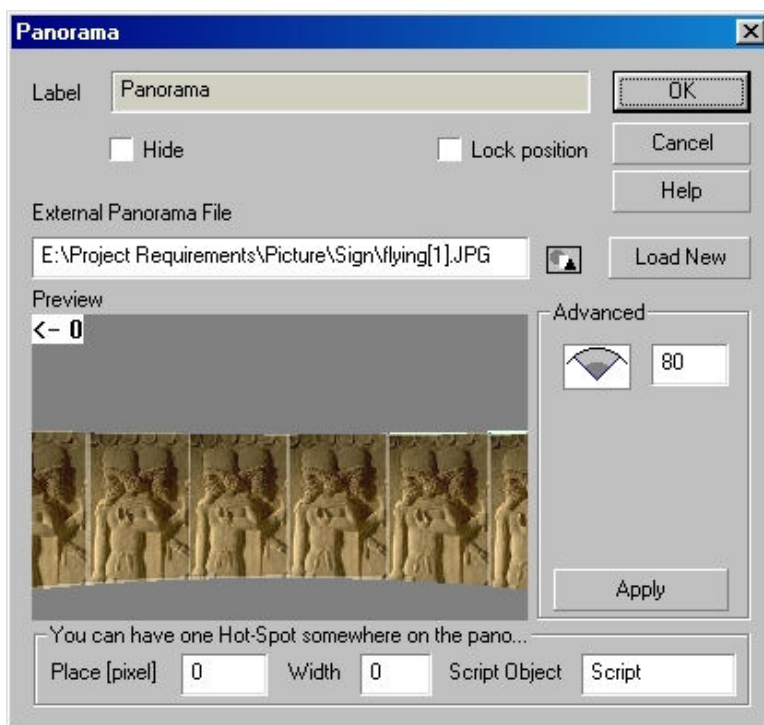


برای چرخاندن تصویر در راستاهای عمودی (Vertical) افقی (Horizontal) افقی و عمودی (Vertical / Horizontal) و یا عدم چرخاندن (None) می باشد.

• Panorama

این شی سبب ایجاد جلوه‌ای بر روی عکس می شود که بیننده تصور می کند آن عکس در فضایی ۳ بعدی و قابل چرخش قرار دارد. این ویژگی برای تزئین پروژه بر اساس عکسها مفید است. در مواقعی هم که محور فعالیت شما عکس می باشد Panorama می تواند انتخاب خوبی باشد.

– خصوصیت های شی Panorama



Load New

با این گزینه تصویر جدیدی را انتخاب می‌کنیم.

Embedded File

می‌توان توسط این قسمت تصویر شی Panorama را به برنامه الحاق نمود.

Preview

برای باز بینی تصویر شی Panorama می‌باشد.

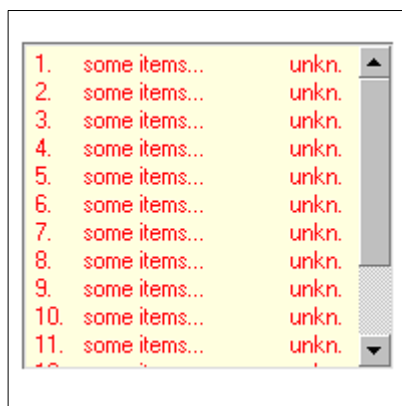
Advanced

برای تعیین زاویه دید می‌باشد.

You can have one hotspot somewhere on the pano...

همان طور که واضح است می توان بر روی شی Panorama محدوده ای مشخص کرد که همانند شی Hotspot عمل نماید و به هنگام کلیک بر روی آن قسمت، اسکرپتی که در قسمت شی Script باید معرفی شود اجرا می شود.

• List Box (کادر لیست)



شی List Box قابلیت نشان دادن و لیست کردن تعدادی از قسمت‌ها و انتخاب‌ها را فراهم می نماید که کاربر بر حسب نیاز می تواند یکی یا بیشتر از یک قسمت از انتخاب‌ها را برگزیند. هدف اولیه MMB از نهادن شی List Box برای نشان دادن لیست اجرایی (Play List) فایل‌های صوتی است. با این حال این شی می تواند به عنوان یک List Box معمولی با قابلیت چندین انتخابی عمل نماید. می توانید List Box را توسط متغیرهای رشته‌ای، متون، آرایه‌ها و لیست اجرایی صوتی پر نمایید. به دنبال اضافه شدن این شی به برنامه انجام کارهای مختلف و کارآمدی امکان پذیر شده است. نکته حائز اهمیت در این قسمت اینست که برای تعامل با این شی توسط فرامین مربوطه حتما لازم است تا شی قابل رؤیت باشد، بدین معنا که در هنگام طراحی نباید آنرا مخفی ساخت. اما چنانچه شرایطی پیش بیاید که مجبور باشیم این شی را مخفی کنیم می توانیم آن را به خارج از محدوده ی صفحه در زمان طراحی انتقال دهیم تا مجازا این شی را پنهان کرده باشیم. یاد آوری می شو دکه با استفاده ماهرانه از این شی به همراه Edit Box می توان شی Combo Box را ایجاد نمود.

— خصوصیت های شی List Box



Style

این قسمت چگونگی شمایل ListBox را معین می‌کند.

- Scroll bar (نوار پیمایش)

- Multi selection (چند انتخابی)

- Client Edge (خط حاشیه)

- Static Edge (لبه‌های ایستا)

- Modal Frame (لبه‌های برجسته)

- Border (حاشیه)

Properties

این قسمت یکسری دیگر از پارامترها را تنظیم می‌کند.

۱- شی ای است که علاوه بر خصوصیات شی ListBox قابلیت انتخاب مولفه هایش بر اساس عمل تایپ کردن مهیا می باشد.

Drag & Drop-

این قسمت قابلیت کشیدن و رها کردن را برای List Box فعال می‌کند.

Hide Numbers-

عددها را در کنار مؤلفه‌ها نمایان نمی‌سازد.

Search for ID tags-

این گزینه به جستجوی tag فایل‌های OGG و MP3 می‌پردازد.

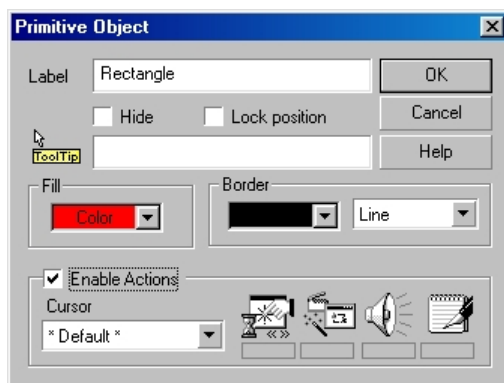
Hide time-

این گزینه زمان فایل‌های صوتی را به هنگام قرارگیری در لیست نشان نمی‌دهد.

Color-

در اینجا می‌توان رنگ زمینه (Background) و متن (Text) شی ListBox را تعیین کرد.

• Rectangle (مستطیل)



این شی برای کشیدن مربع و یا مستطیل می‌باشد. این شی می‌تواند فعال یا غیر فعال باشد.

- خصوصیت‌های شی Rectangle

Fill

رنگ شی را تعیین می کند.

Border

۲ قسمت موجود است، قسمت سمت چپ برای رنگ حاشیه شی و قسمت سمت راست برای چگونگی حاشیه شی اعم از: Line(خط)، None(هیچ)، Sunken(فرورفتگی)، Window(برجستگی) است.

• Circle (دایره)

توسط این قسمت به آسانی می توانید یک دایره و یا بیضی را ترسیم کنید. این شی هم می تواند فعال و یا غیر فعال باشد.

- خصوصیت های شی Circle

خصوصیت های این شی نیز کاملاً همانند شی مستطیل است.

• Polygon (چند ضلعی)

برای ترسیم چند ضلعی از این ابزار استفاده می شود.

- خصوصیت های شی Polygon

خصوصیات این شی هم مشابه مستطیل است.

• Line(خط)

این شی برای ترسیم خطوط مایل، شکسته و مستقیم می باشد.

- خصوصیت های شی Line

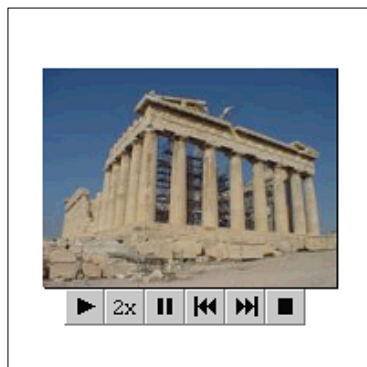
Position

برای تعیین موقعیت خط استفاده می شود. $x1$ و $x2$ به ترتیب طول نقطه های ابتدایی و انتهایی خط را مشخص می کند. $y1$ و $y2$ هم عرض نقاط ابتدایی و انتهایی را تعیین می کنند.

Line Color

رنگ خط را مشخص می کند.

Video •



برای وارد کردن فایل های ویدئویی مورد نظرتان به پروژه می توانید از این شی استفاده کنید. از فرمت های حمایت شده برای شی Video می باشند. شی Video یکی از اشیای غیر قابل اجتناب در اکثر پروژه های چند رسانه ای می باشد دکمه توسط اسکریپت نویسی و استفاده خلافتانه می تواند پروژه ای حرفه ای و کارا را به ارمغان بیاورند. در صورتی که نوع دیگری از فرمت های ویدئویی موجود باشد که در اینجا ذکر نشده باشد، می توان با نصب Codec¹ مربوطه آن اجرای آن فرمت خاص را در MMB میسر ساخت.

– خصوصیت های شی Video

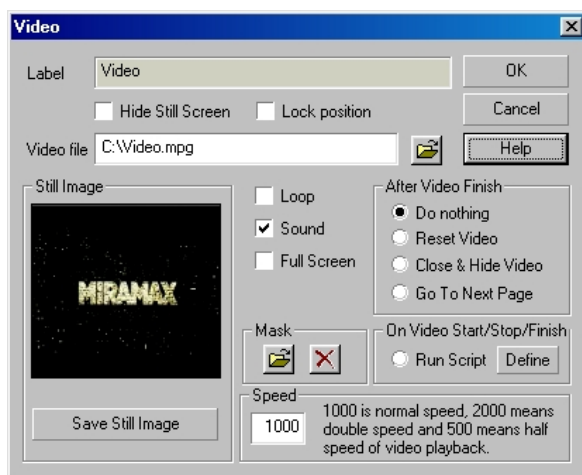
این کادر جهت وارد کردن فایل های ویدئویی می باشد. این کادر به شما اجازه می دهد تا هنگامی که فایل ویدئویی در حال اجرا نمی باشد عکسی را بر روی شی ویدئویی به عنوان عکس جاری نمایش دهید. هنگامی که توسط این کادر فایل ویدئویی را بارگذاری کردید، کارد دیگری جهت مشاهده فایل نمایان می شود که از آنجا می توانید عکس جاری را به وسیله عقب و جلو بردن فایل ویدئویی تنظیم کنید. چنانچه فایل ویدئویی شما Mpg باشد و بخواهید تصویری را به عنوان عکس جاری برگزینید ممکن است موفق به انجام آن نشوید و عکس جاری تنها یک محدوده به رنگ مشکی باشد. این رفتار به خاطر MCI driver می باشد که مانع برگزیدن عکس جاری می شود. راه حل این مشکل این است که عکس مناسب با اندازه فایل ویدئویی بر روی آن قرار دهید. هنگامی که فایل اجرا است بالاتر از همه اشیاء حاضر در صفحه قرار می گیرد.

۱-ابزاری است که از آن برای فشرده سازی فایل های ویدئویی استفاده می شود.

البته در نسخه های جدید MMB اصلاحاتی در این زمینه صورت گرفته است. در نسخه 4.9.7 می توانید اشیا را بر روی شی ویدئو قرار دهید. در این حالت شی ویدئو در بالاترین سطح قرار می گیرد و هنگام اجرا قسمتی که در زیر شی دیگر قرار گرفته است قابل رویت نمی باشد.

Video File

مسیر فایل ویدئویی را مشخص می کند.



Still Image

تصویری را که در اولین فریم موجود است نشان می دهد.

Save Still Image

تصویر اولین فریم را ذخیره می کند.

Loop

سبب ایجاد تکرار در اجرای فایل ویدئویی می شود.

Sound

صدای فایل ویدئویی را پخش می کند.

Full Screen

فایل ویدئویی را به صورت تمام صفحه نشان می دهد.

Mask

این قسمت فایل تصویری را به عنوان ماسک بر روی فایل ویدئویی قرار می‌دهد. از این قسمت می‌توان برای ایجاد دریچه‌های دید جالبی استفاده کرد.

Speed

این قسمت سرعت اجرای فایل ویدئویی را تعیین می‌کند. سرعت پیش فرض 1000 است. 2000 به معنی ۲ برابر سرعت معمولی و 500 به معنای نصف سرعت معمولی است.

After Video Finish

این قسمت معین می‌کند که بلافاصله پس از اتمام اجرای فایل ویدئویی برنامه چه کاری انجام دهد.

Do nothing-

هیچ عملی صورت نمی‌گیرد.

Reset Video-

فایل ویدئویی به ابتدا باز گردانده می‌شود.

Close & Hide Video-

این گزینه پنجره ویدئو را بسته و محو می‌کند.

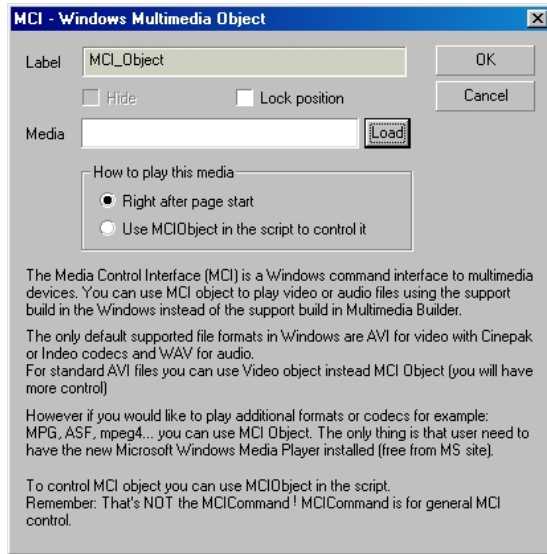
Go to next Page-

پس از اتمام اجرای فایل ویدئویی به صفحه‌ی بعد می‌رود.

On Video Start / Stop / Finish-

در این قسمت برنامه می‌تواند بر اساس رویدادهای شروع / توقف / اتمام فایل ویدئویی اسکریپتی را اجرا کند.

• MCI Object



راه دیگری جهت اجرای فایل‌های ویدئویی (هم چنین صوتی) استفاده از شی MCI می‌باشد و به تبع آن می‌توان شی MCI را توسط فرامین MCI کنترل نمود. لازم به ذکر است که توسط طیف گسترده فرامین ارائه شده توسط MMB در نسخه های جدید نیاز به استفاده از این شی رو به کاهش گذاشته است.

• خصوصیت‌های شی MCI

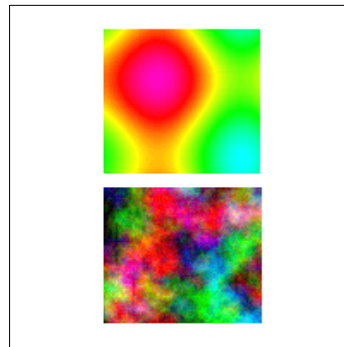
Media

برای بارگذاری فایل‌های صوتی و تصویری می‌باشد.

How/To play this Media

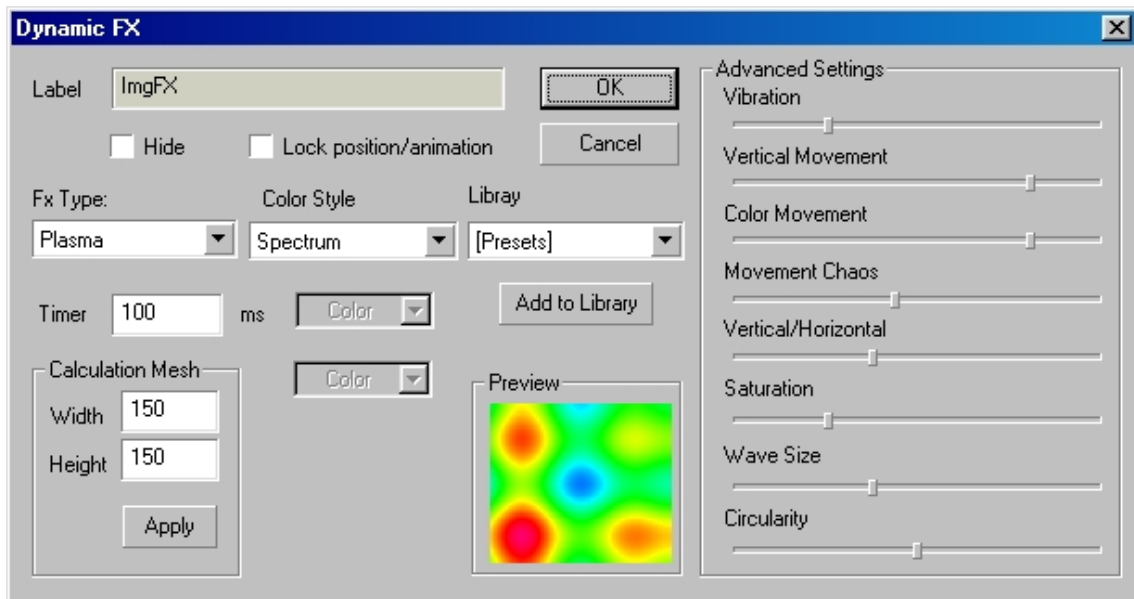
این قسمت معین می‌کند که فایل صوتی و یا تصویری چگونه اجرا شود. یعنی بلافاصله پس از اجرای برنامه page (Right after start) اجرا شود یا توسط اسکریپت (Use MCI Object in Script to Control it) کنترل شود.

• (Dynamic fx)



این شی می‌تواند جلوه‌های گوناگون و جالبی را در صفحه و یا بر روی عکس‌ها ایجاد کند. جلوه‌هایی از قبیل آتش، دود و... Dynamic fx ها جای زیادی اشغال نمی‌کنند ولی فضای زیادی از CPU را به خود اختصاص می‌دهد.

خصوصیت های شی Dynamic fx



fx Type

این قسمت نوع fx را تعیین می‌کند. در این قسمت نوع‌های Plasma (پلازما)، Smoke (دود)، Fire (آتش)، 3D Dots / Fire (نقطه‌های ۳ بعدی / آتش)، Bumbing Light (نور برجسته) و Roto Zoom (بزرگ‌نمایی چرخنده) موجود است.

Color Style

این قسمت سبک رنگ را مشخص می‌کند. این قسمت شامل سبک‌های Spectrum (طیف)، Spectrum Red، Lumi Blue (آبی روشن)، Lumi Green (سبز روشن)، Color Blend (آمیزه رنگ)، Acid (اسید) می‌باشد.

Add to Library

چنانچه در هنگام تنظیم سبک رنگ fx به ترکیب جالب و قشنگی دست یابیم و مایل باشیم که آن را ذخیره کنیم می‌توانیم از دکمه Add to Library استفاده کنیم.

Library

این قسمت شامل لیستی از ترکیباتی است که قبلاً به کتابخانه اضافه شده‌اند.

Timer

این قسمت سرعت حرکت D-fx را معین می‌کند.

Calculation Mesh

این قسمت میزان درهم رفتگی مؤلفه‌های D-fx را تعیین می‌کند.

Preview

پیش‌نمایشی را از D-fx نشان می‌دهد.

Advanced Setting

در این قسمت یک سری تنظیمات پیشرفته را می‌توانیم بر روی D-fx اعمال کنیم.

Horizontal Movement

این قسمت سرعت حرکت افقی مؤلفه‌ها را تنظیم می‌کند.

Vertical Movement

برای تعیین سرعت حرکت عمودی مؤلفه‌ها است.

Interference

این گزینه مقدار تداخل مؤلفه‌ها را تنظیم می‌کند.

Chunks

غلظت مؤلفه‌ها را تعیین می‌کند.

Vibration

میزان لرزش مؤلفه‌ها و مقدار بی‌نظمی را تعیین می‌کند.

Color Movement

این گزینه سرعت حرکت و تغییر رنگ را تعیین می‌کند.

Saturation

این گزینه میزان اشباع رنگ را تعیین می‌کند.

Wave size

این قسمت اندازه موج را معین می‌کند.

Circularity

جهت تعیین میزان دایره‌وار بودن مولفه‌ها است.

Cut off

این گزینه محدوده‌ی بین ۲ مولفه‌ی fx را تعیین می‌کند.

Horizontal wave

این گزینه حالت موج افقی را مشخص می‌کند.

Vertical wave

این گزینه حالت موج عمودی را مشخص می‌کند.

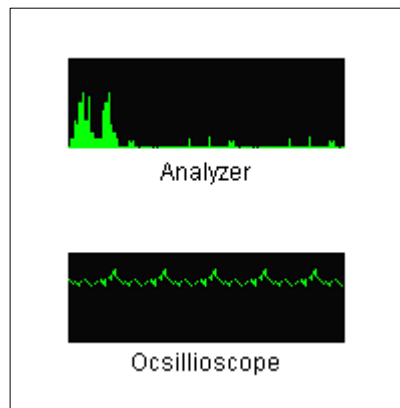
X,Y,Z Rotation

این گزینه‌ها به ترتیب چرخش در راستای X, Y, Z را مشخص می‌کند.

Burning Speed

سرعت سوختن را مشخص می‌کند.

• Audio VISUALIZATION (رقص نور)



توسط این قابلیت منحصر به فرد MMB می‌توانید احساسی بصری به آهنگ‌های در حال اجرا ببخشید. بدین معنا که یک رقص نور جهت القای این حس در پروژه خود قرار دهید. از این رقص نور می‌توان برای اجرای فایل‌های صوتی S3M, XM, OGG, Wav, MP3 استفاده نمود. در حقیقت این شی‌آنچمنان سودمند نیست، جز اینکه کاربران به هنگام گوش کردن به یک آهنگ تمایل به دیدن رقص نور را دارند. برای کنترل بهتر رقص نور می‌توانید نوع آن را نیز تغییر دهید. ۲ نوع رقص نور موجود است:

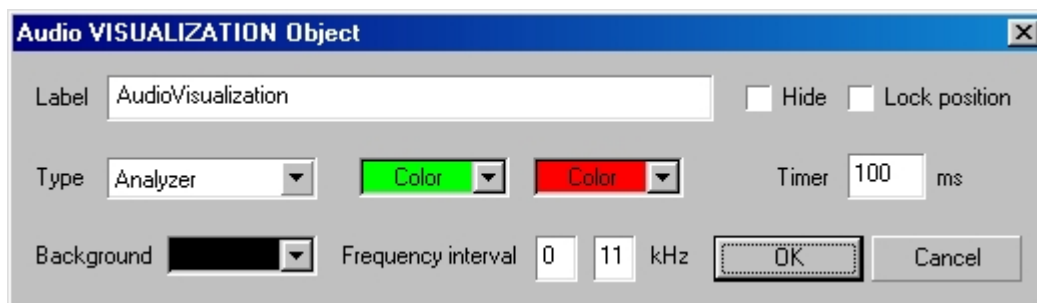
۱- Oscilloscope (نوسانی) ۲- Analyzer (تحلیلی)

خصوصیت‌های شی AV

Timer

سرعت به‌روز رسانی را بر حسب زمان تعیین می‌کند.

Background

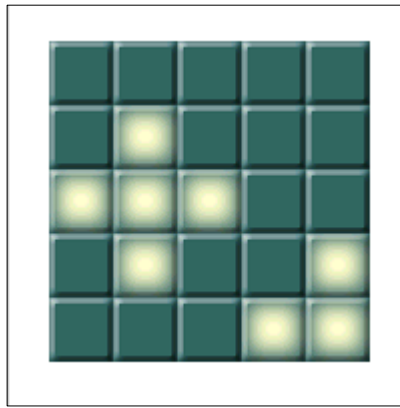


این قسمت رنگ زمینه رقص نور را تعیین می‌کند.

Frequency Interval

محدوده‌ی فرکانس‌ها را جهت پردازش معین می‌کند.

• Image Matrix (ماتریس)



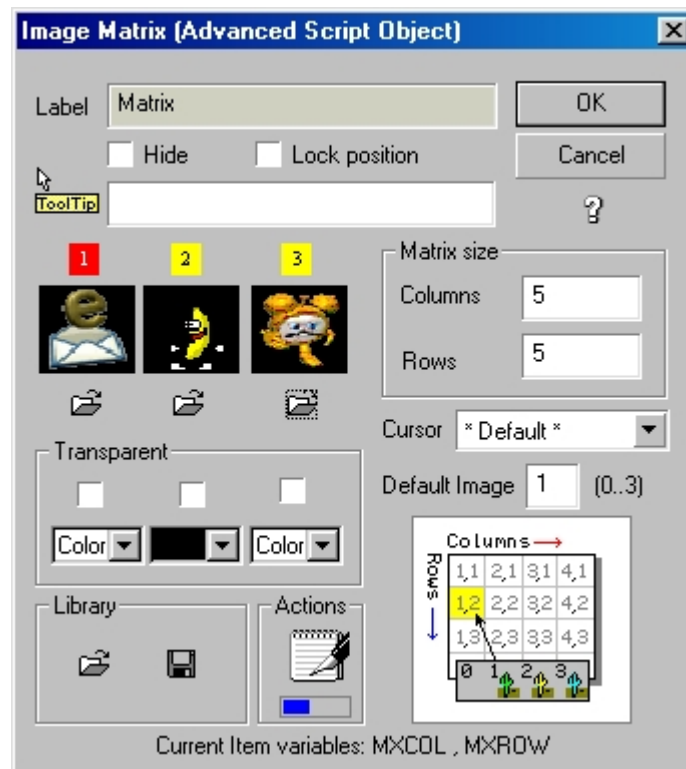
تصور کنید که قصد ساختن صفحه یک بازی را دارید که دارای ۵×۵ قسمت یکسان است و در هر قسمت صفحه می‌توانید ۳ شکل بر روی هم داشته باشید. چنانچه قوانین بازی حکم نماید که این عکس‌ها ترتیب‌شان تغییر نماید شما باید از فرامین Hide / show برای انجام این عمل استفاده کنید. اما MMB با استفاده از شی Matrix راه حل مناسبی را پیشنهاد می‌کند. به طور کلی، با این شی می‌توان اشیایی را تولید کرد که وضعیت آنها بایستی با کلیک موس تغییر کند. به عنوان مثال توسط این شی می‌توان اقدام به تولید Radio Button, Check Box (دکمه رادیویی) و... نمود. تعداد سطرها و ستونهای شی ماتریس اندازه شی ماتریس را مشخص می‌کنند. ضمناً جهت استحصال نتیجه مطلوب باید هر ۳ شکل قرار داده شده هم اندازه باشند. برای مثال از این شی می‌توان به عنوان نشان دهنده وضعیت استفاده کرد. بدین ترتیب که برای وضعیت های مد نظر خود تصویری خاص را برگزینیم تا در هنگام برقراری شرایط نمایان شود. برای تعداد سطر و ستون ها محدودیتی وجود ندارد اما بهتر است که حتی الامکان از تعداد کمتری استفاده کنیم.

سطر
→

1.1	2.1	3.1
1.2	2.2	3.2
1.3	2.3	3.3

↓

- خصوصیت های شی Image Matrix



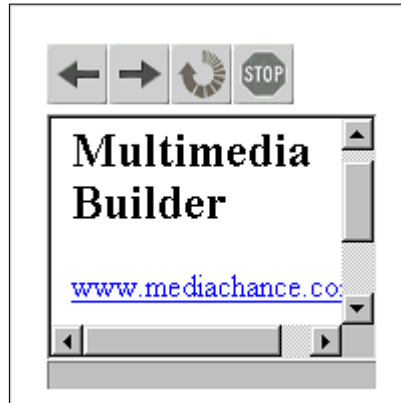
1,2,3-

این قسمت ها محل بارگذاری تصاویر می باشد که تنها مجاز می باشید برای هر خانه از ماتریس ۳ تصویر مجزا وارد نمایید. هرچند با استفاده ماهرانه می توان تعداد آنها را بیشتر نمود.

Matrix Size

این گزینه اندازه ماتریس را معین می کند. Columns (ستونها) و Rows (سطرها).

HTML Browser •

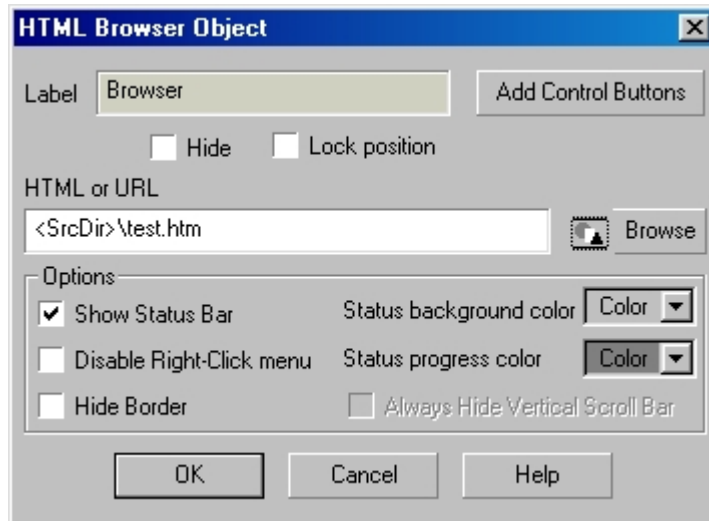


شی HTML یک جستجوگر کامل در MMB می‌باشد. این شی از کنترل‌های IE استفاده می‌نماید و این بدان معناست که این شی به Internet Explorer وابستگی دارد. ولی در عین حال این وابستگی مزیت‌هایی را به ارمغان می‌آورد:

- قادر به چاپ محتوی شی HTML هستید.
 - می‌توانید از اسکریپت‌های Java ، اپلت‌ها، فرم و تمام چیزهایی که با IE سازگار است سود جوئید.
 - امکان اتصال به اینترنت
 - امکان استفاده از FTP
 - می‌توانید از شی HTML برای اجرای ActiveX و فایل‌های Pdf استفاده نمایید
 - سازگاری با IE نسخه 4.x به بالا
 - قابل کنترل توسط اسکریپت نویسی
 - سفارشی سازی ظاهر HTML
- همان طور که بیان شد با استفاده از این شی می‌توانید محتویاتی را که برنامه Internet Explorer قادر به نمایش آنهاست را در MMB نمایش دهید. به عنوان مثال می‌توانید با پایگاه داده‌ها ارتباط برقرار کنید و یا تصاویر با فرمت‌های گوناگون را بار گذاری نمایید.

- خصوصیت‌های شی HTML

HTML or URL



این قسمت تعیین می‌کند که در هنگام اجرای برنامه شی HTML حاوی چه چیزی باشد. این محتوی هم می‌تواند یک فایل HTML الحاق شده باشد و هم آدرس سایتی خاص.

Options

این قسمت یکسری انتخاب‌ها را در اختیار ما قرار می‌دهد.

• Show Status Bar

در صورت فعال بودن نوار وضعیت را نمایان می‌سازد.

• Disable Right Click Menu

در صورت فعال بودن منویی را که به هنگام راست کلیک کردن نمایان می‌شود را غیر فعال می‌کند.

• Hide Boarder

حاشیه شی HTML را مخفی می‌کند.

• Status Background Color

رنگ پس زمینه نوار وضعیت را تعیین می‌کند.

• Status Progress Color

این گزینه رنگ نوار پیشرفت نوار وضعیت را تعیین می‌کند.

• Always Hide Vertical Scroll Bar

نوار پیمایش عمودی را مخفی می‌کند.

• Add Control Button

این قسمت چهار دکمه ی Back, Forward, Refresh, Stop را جهت تعامل با شی HTML ارائه می‌دهد.

• Flash

شی فلش (Flash) یک شی کارآمد و پرکاربرد است. از این شی جهت نمایش دادن کلیپ های ساخته شده توسط نرم افزار فلش استفاده می شود. جالب است بدانید که در MMB از شی فلش تنها برای اجرای کلیپ (Clip) های فلش استفاده نمی شود بلکه توسط Action Script برنامه Macromedia Flash می توانید فرامین MMB فراخوانی کنید و پروژه را کنترل کنید. بنابراین با استفاده از شی فلش می توانید بر قدرت و توانایی های تولید خود بیفزایید. به عنوان مثال می توانید از توابع ریاضی و سایر فرامین فلش استفاده نمایید و نتیجه را به MMB ارسال کنید. البته این گونه از استفاده مستلزم دانستن Action Script فلش می باشد. شما می توانید فایل فلش مورد نظر خود را به برنامه الحاق نمایید و یا به صورت خارجی از آن استفاده کنید. حتی این قابلیت نیز فراهم است که فایل فلش مورد نظر را از روی اینترنت بارگذاری نمایید.

- خصوصیت های شی Flash

Load Movie

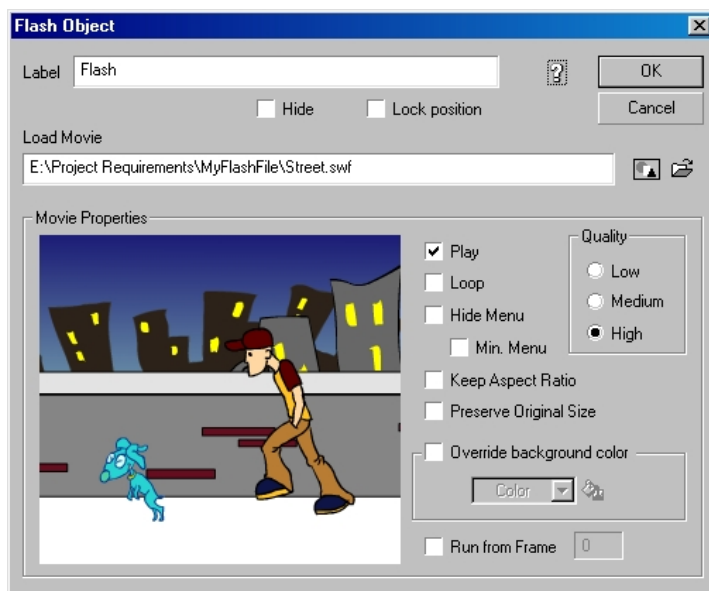
برای بارگذاری فایل های SWF می باشد.

Movie Properties

این قسمت جهت تنظیم پارامترهای فایل SWF بارگذاری شده است.

• Play

سبب اجرا شدن فایل SWF



بلافاصله پس از اجرا شدن برنامه می شود.

Loop •

سبب اجرای مکرر فایل SWF می شود.

Hide Menu •

این گزینه منویی را که به هنگام راست کلیک بر روی فایل SWF ظاهر می شود را مخفی می کند.

Min.Menu •

این گزینه به هنگام راست کلیک بر روی شی فلش منوی کوچک شده را نشان می دهد.

Keep Aspect Ratio •

این گزینه مانع از به هم ریختن نسبت وجوه می شود. این گزینه به هنگام تغییر اندازه شی کارآمد است.

Preserve Original Size •

انتخاب این گزینه سبب نگه داشتن اندازه اصلی تحت هر شرایطی می شود.

Override Background Color •

این گزینه رنگ پس زمینه فایل SWF را مشخص می کند. البته این کار همیشه امکان پذیر نیست.

Run from frame •

این قسمت شروع اجرای فایل SWF را بر اساس فریم داده شده آغاز می کند.

Quality •

این قسمت کیفیت فایل SWF را تعیین می کند. Low (کیفیت پایین) Medium (کیفیت معمولی) و Hight (کیفیت بالا).

Binding object •

شی Binder راهی ساده است برای نهادن و یا برقراری ارتباط با یک فایل اجرایی (Exe) خارجی. در صورتی که فایل اجرایی مورد نظر به برنامه الحاق شود جزئی از اجرا کننده MMB محسوب می شود. به عنوان مثال یک انیمیشن فلش به همراه یک اجرا کننده (Flash Player) ، Notepad با یک متن و ...

نکته: توجه کنید که تمامی فایل های اجرایی (exe) نمی توانند در پروژه جا سازی شوند چرا که بعضی از آنها به خاطر عدم سازگاری سبب توقف فایل اجرایی و در نهایت تولید نقص در اجرای پروژه می شود. توسط MMB می توانید منوهای فایل اجرایی اتصال یافته را کنترل نمایید. حتی در صورت تمایل می توانید بیشتر از یک شی Binder را در پروژه خود قرار دهید. مثالی از این نوع برنامه اجرایی EditBox است که جهت رفع محدودیت های EditBox برنامه MMB ساخته شده است.



- خصوصیت های شی Binder

Exe to Bind

این قسمت فایل اجرایی (Exe) را جهت اتصال مشخص می کند.

Parameters

در این قسمت بر اساس فایل اجرایی می توان پارامترهایی را وارد کرد تا برنامه اتصال یافته عملی را انجام دهد. استفاده از این قسمت به هنگام به کار گیری برنامه های CommandLine سودمند می باشد.

Options

Hide Menu •

این گزینه منوهای فایل اتصال یافته را مخفی می کند.

Hide Sizing Grip •

این گزینه قسمتی از برنامه را که برای تغییر اندازه پنجره است را مخفی می کند.

Larger Grip •

محدوده‌ای را که در سمت راست و پایین برخی از برنامه جهت تغییر اندازه پنجره برنامه می‌باشد را بزرگتر می‌کند.

- **Border**

نوع حاشیه را تعیین می‌کند. None (هیچ)، Raised (برجسته)، Sunken (فرو رفته)

- **How to Un-bind the application**

این قسمت مشخص می‌کند که برنامه اتصال یافته چگونه از حالت اتصال خارج شود.

- **Terminate Process-** (پایان بخشیدن به پردازش)

به صورت قاطع و بدون توجه به وضعیت اجرایی برنامه از آن خارج می‌شود.

- **Send close and wait-**

روشی ملایمتر جهت بستن فایل اجرایی اتصال یافته است. بدین معنا که به برنامه اجرایی فرصت بسته شدن و احیاناً پرسشی مبنی بر ذخیره فایل وابسته به برنامه جاسازی شده را ارائه می‌دهد. از این گزینه در صورت ناسازگار بودن روش قبل می‌توان استفاده کرد.

- **Un Bind On Page Exit-**

به محض انتقال از صفحه حاوی شی Binder اجرای فایل اجرایی اتصال یافته متوقف می‌شود. همچنین در صورت بازگشت به صفحه حاوی شی Binder مجدداً فایل جاسازی شده اجرا می‌شود.

- **Advanced Properties**

برخی از برنامه‌های پیچیده ممکن است دارای چندین پنجره وابسته باشند و یا برخی از قسمتهای آنها ایجاد مزاحمت نماید. به همین سبب MMB ممکن است که پنجره‌ای نادرست را جهت باز کردن و اتصال انتخاب کند. در این مواقع است که خصوصیات پیشرفته به کمک MMB می‌آید. برای حل این مشکل می‌توانید به MMB بگویید که عنوان پنجره مورد نظر حاوی چه رشته‌ای است و یا بر عکس حاوی چه رشته‌ای نیست.

نکته: در برنامه‌های ساده از قبیل Notepad, FlashPlayer احتیاجی نیست که به MMB بگویید پنجره مورد نظر حاوی چه نوشته‌ای است. این ویژگی زمانی به کار می‌آید که MMB در تشخیص پنجره‌ها اشتباه کند.

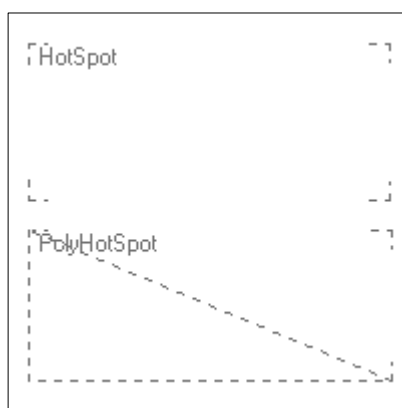
- **Wait Before Binding-**

برخی از برنامه‌ها ممکن است زمان بیشتری برای بارگذاری نیاز داشته باشند چون MMB بلافاصله سعی در اجرای برنامه دارد ممکن است برنامه در زمان شروع سریعاً اجرا نشود، به همین منظور بایستی از این ویژگی سود جست، چرا که این ویژگی سبب توقف MMB تا زمان بارگذاری کامل برنامه اتصال یافته می‌شود.

Additional Cut from Top or bottun-

از دیگر مزیت‌های Binder این است که شما به آسانی قادرید محدوده‌ای مشخص از بالا و یا پایین برنامه اتصال یافته را پنهان نمایید. از این ویژگی بیشتر زمانی استفاده می‌شود که بخواهید نوار ابزار یا نوار وضعیت یک برنامه را پنهان کنید.

• Hot-Spot



یک شی فعال می‌باشد. Hot-Spot یک شی و یا به عبارتی محدوده‌ای غیر قابل دید است که در پروژه تعریف شده و کاربر نهایی می‌تواند با کلیک کردن روی آن و یا انتقال موس بر روی آن سبب اجرای اسکریپتی شود. Hot-Spot در جلوی دیدگان کاربر نهایی غیر قابل دید است و تنها شما در زمان طراحی می‌توانید محدوده‌ی آن را که توسط خط چین مشخص شده است را مشاهده نمایید. اما باید گفت که ۲ نوع Hot-Spot موجود است:

Hot-Spot-۱ (چهار ضلعی)

Polygonal Hot-Spot-۲ (چند ضلعی)

که هر کدام بسته به نیاز به کار می‌آید. از شی Hotspot استفاده‌های زیادی می‌توان کرد همانند قرار دادن آن به عنوان چهارچوب بارگذاری عکس‌ها و یا برای غیر فعال سازی برخی اشیا در زمان اجرای برنامه.

• Script

شی Script یک شی غیر فعال است که در هنگام اجرای برنامه قابل دید نیست. شی Script می‌تواند توسط میانبرها فراخوانی و یا از جایی دیگر اجرا شود. این شی به شما این امکان را می‌دهد که کد برنامه‌ی خود را در صورت نیاز بدون استفاده از موس اجرا کنید. این شی در هنگام اجرای برنامه نامرئی می‌باشد و تنها اثر آن محسوس است.

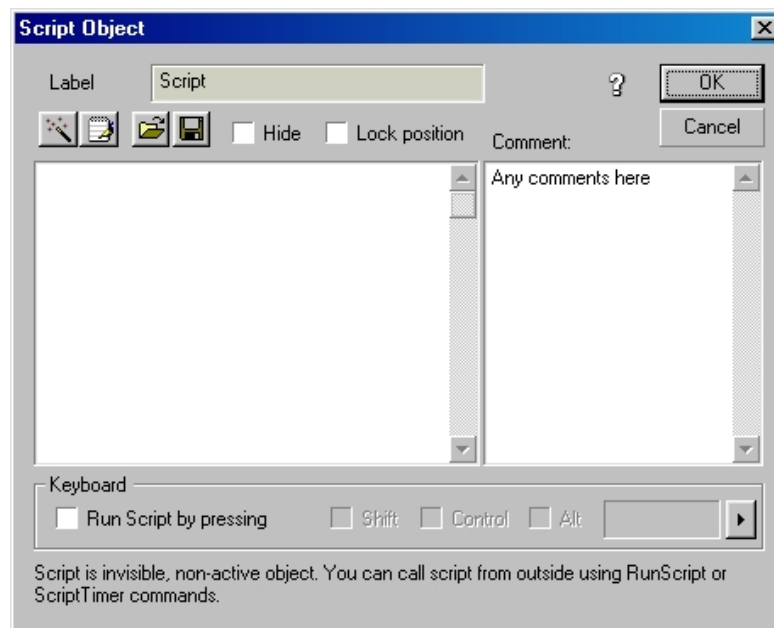
- خصوصیت‌های شی Script

Comment

جهت اضافه کردن توضیحاتی به کد برنامه می‌باشد. توضیحات می‌توانند سبب خوانایی بیشتر کد ها شوند.

Shortcuts

برای تعیین کلیدهای میانبر می‌باشد. بدین ترتیب چنانچه میانبری برای شی اسکریپت تعریف کرده باشیم ، اگر در زمان اجرای پروژه کلید های مشخص شده را بفشاریم اسکریپت مربوط هاجرا خواهد شد.



خصوصیت های مشترک بین اشیاء

همانطور که می دانید تمام اشیاء در MMB دارای خصوصیتی می باشند که برخی از این خصوصیات تقریباً در تمامی اشیاء مشترک می باشد و ما هم به جهت جلوگیری از تکرار آنها قصد داریم تا آنها را در این قسمت توضیح دهیم. شایان ذکر است که به ۲ صورت می توانیم به کادر خصوصیت یک شیء در MMB دسترسی پیدا کنیم اول اینکه با ۲ بار کلیک بر روی شیء مورد نظر و راه دوم توسط گزینه properties از منوی Object است. بقیه خصوصیات منحصر به فرد هر شیء در قسمتهای قبلی بررسی شد.

- Label (نام شیء)

در این کادر میتوانیم بر چسبی را به شیء اختصاص دهیم. توصیه می کنیم که حتی الامکان از اسامی معنادار استفاده کنید چرا که بعداً در اسکرپت نویسی راحت تر می توانید با اشیاء تعامل برقرار کنید.

- Hide (پنهان سازی)

این گزینه شیء را هم در زمان طراحی و هم در زمان اجرا مخفی نگه می دارد تا زمانی که توسط دستوری شیء از نو نمایان شود، حالت مخفی بودن ادامه خواهد داشت. ضمناً هنگامی که شیء ای در زمان طراحی مخفی باشد کنار نام آن شیء در پنجره object ضربدر قرمزی نمایان می شود.

- Lock position (قفل کردن موقعیت)

با استفاده از این گزینه میتوان در هنگام طراحی، از حرکت شیء، ممانعت کرد. این ویژگی هنگامی بیشتر کارآمد است که صفحه حاوی اشیاء ریز و درشت بسیاری باشد و ما بخواهیم از تغییر مکان ناخواسته اشیاء جلوگیری کنیم.

- Font (نوع قلم)

این ویژگی جهت تغییر نوع قلم متن های شیء است.



Actions (فعالیت ها)

توسط این قسمتها فعالیت‌های پروژه مشخص می‌شود. قسمت Action شامل چهار بخش:

External command & Page actions (فرمان خارجی و فعالیت‌های صفحه)

با اعمال این قابلیت میتوان فعالیت‌هایی نظیر نصب یک برنامه، اجرای یک پروژه خارجی، جستجو و گردش در اینترنت، ارسال نامه الکترونیکی، باز کردن یک سند، نمایش فایل راهنما، اجرای فایل MMB دیگر، جستجو در دیسک سخت، بارگذاری و اجرای فایل‌های صوتی، حرکت بین صفحات و خروج از صفحات را انتظار داشت.

teractions with other object video (تعامل با اشیاء دیگر و ویدئو)

بسته به نوع رفتار موس (قرار گرفتن روی شیء یا کلیک بر روی آن) شیء می‌تواند رفتاری از خود بروز دهد. برای مثال پنهان شدن و نمایش داده شدن، اجرای فایل ویدئویی و ... با استفاده از این قسمت میتوان یک پروژه‌ی حرفه‌ای داشته باشید.

Sound Action (فعالیت صدا)

باز هم بسته به نوع قرارگیری موس (روی شیء یا کلیک بر روی شیء) MMB میتواند صدای مربوطه را اجرا نماید. لازم به تذکر است که MMB از چندین کانال حمایت میکند این بدین معناست که MMB میتواند همزمان چندین صدا را اجرا نماید و نگرانی از بابت بخش آهنگ زمینه به خاطر ایجاد وقفه وجود ندارد. البته در این حالت ویژگی DirectSound باید فعال باشد.

More actions (فعالیت‌های بیشتر)

با توجه به اینکه شما میتوانید یک سری فعالیت‌های اصلی و ساده را به اشیاء اعمال کنید، برای کنترل بهتر پروژه‌ها و سودمند کردن آنها میتوان از این قسمت جهت اسکریپت نویسی استفاده کرد و در نهایت پروژه تعاملی تری را تولید نمود.

Cursors (مکان نما)

با استفاده از این قسمت می‌توان مکان نمایی را که به هنگام قرارگیری بر روی شیء بایستی ظاهر شود مشخص کرد.

Tooltip (توضیح مختصر)

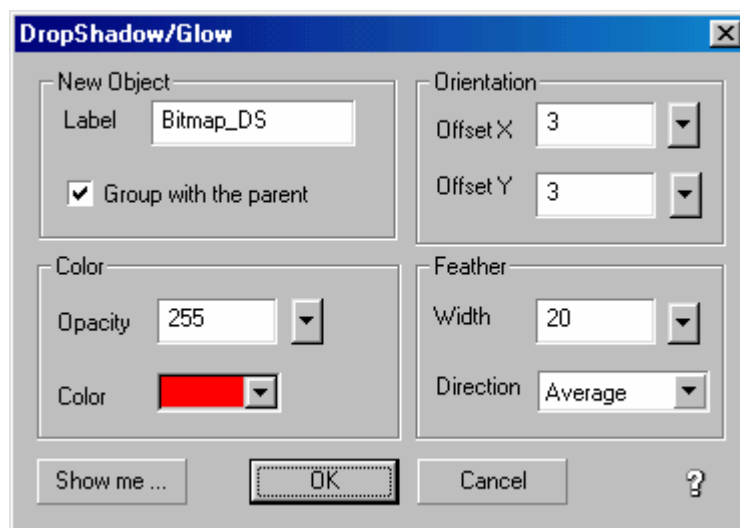
حتماً تا به حال پس از قرارگیری موس بر روی دکمه‌ای یا قسمتی در برنامه‌های دیگر مشاهده کرده‌اید که متن یا عبارت مختصری موقتاً جهت راهنمایی ظاهر میشود و پس از حرکت مجدد موس دوباره محو میشود این قابلیت همان tooltip می‌باشد که ما میتوانیم برحسب نیاز به اشیاء در MMB آن را اضافه کنیم.

جلوه های MMB (Effects)

همان طور که قبلا در بررسی منوهای MMB مشاهده کردید، می توانید جلوه هایی را به اشیا اعمال نمایید، جلوه هایی نظیر (برجسته سازی) Bevel، (سایه) Shadow، (آتش) Fire، (درخشش) CutOut.Glow، در ادامه قصد داریم تا پارامترهای این جلوه ها را بررسی نماییم.

Drop Shadow

این جلوه برای ایجاد سایه می باشد. اندازه سایه و رنگ سایه نیز قابل تنظیم است.



New Object

سایه ای را که MMB ایجاد می کند در نهایت به صورت یک شی تصویری در می آید که در این قسمت می توان برچسبی را برای این شی در نظر گرفت. گزینه Group with the parent سایه و شی اصلی را در یک گروه قرار می دهد.

Color

همان طور که واضح است این قسمت برای تعیین رنگ سایه می باشد. گزینه Opacity برای تعیین غلظت می باشد.

Orientation

برای تعیین موقعیت سایه می باشد.

Feather

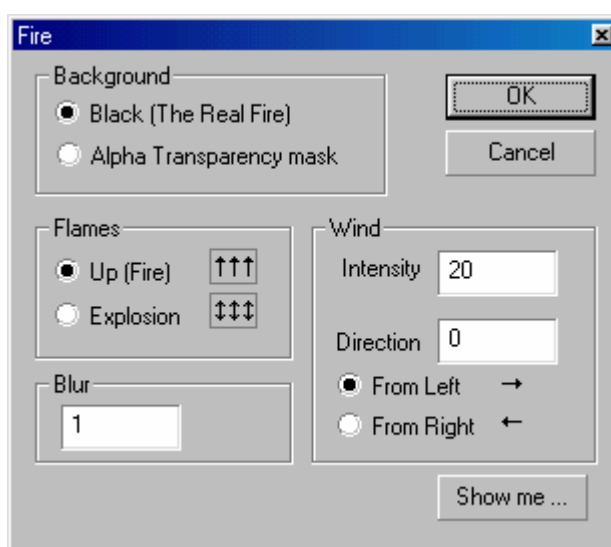
این قسمت برای ایجاد هاله ای اطراف سایه است. Width عرض این هاله را معین می کند و Direction نیز جهت هاله را مشخص می کند.

Glow

این قسمت جلوه درخشندگی را اعمال می کند. پارامترهای این جلوه نیز کاملاً شبیه جلوه سایه است.

Fire

این قسمت جلوه آتش را بر حاشیه اشیا اعمال می کند.



Background

برای رنگ زمینه جلوه آتش می باشد. Black زمینه مشکی را به آتش اضافه می کند و Alpha Transparency mask زمینه آتش را به صورت شفاف در می آورد.

Flame

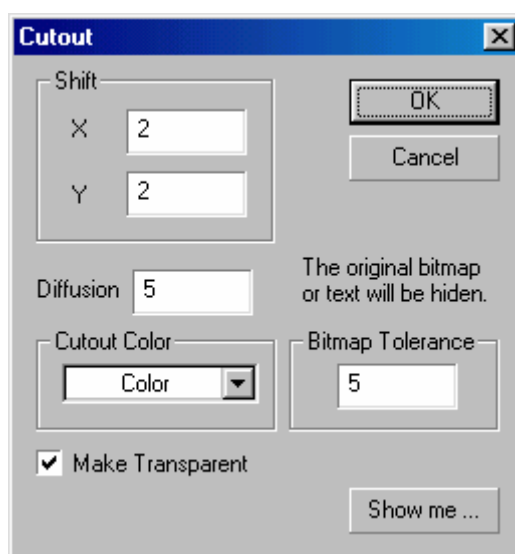
این قسمت برای مشخص کردن جهت شعله های آتش است.

Wind

توسط این ویژگی می توان اثر باد را بر آتش اعمال کرد. Intensity شدت باد را تعیین می کند. Direction برای جهت می باشد.

CutOut

این قسمت حالت فرورفتگی را به تصاویر اعمال می کند. البته این جلوه از کیفیت بالایی برخوردار نمی باشد.



Shift

موقعیت را تعیین می کند.

Diffusion

برای مشخص نمودن حالت انتشاری و گستردگی نور است.

Cutout Color

رنگ این جلوه را معین می کند

Bitmap Tolerance

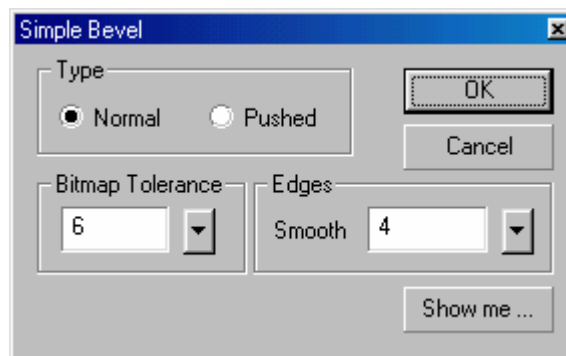
شدت تحمل Bitmap را مشخص می کند.

Make Transparent

خصوصیت Transparency را به این جلوه اضافه می کند.

Bevel

این جلوه برای برجسته سازی حاشیه تصاویر است.



Type

گونه ی برجسته شدن را مشخص می کند.

Bitmap Tolerance

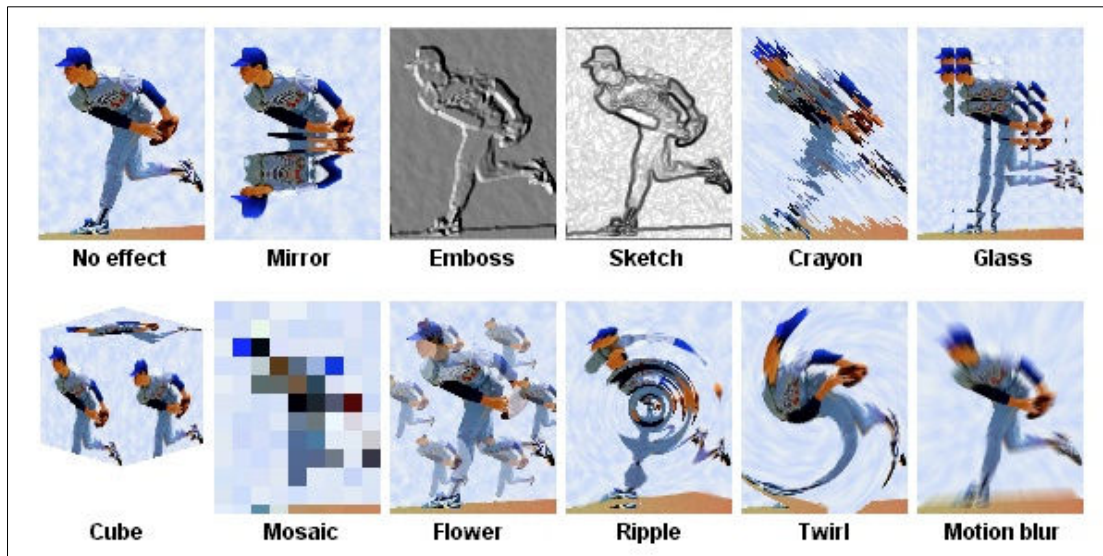
شدت تحمل Bitmap را مشخص می کند.

Edge

مقدار هموار بودن لبه ها را مشخص می کند.

Special Effects

در این قسمت یکسری از فیلترهایی را که بر تصاویر اعمال می شود را ارائه می دهد. کسانی که با برنامه Photoshop کار کرده باشند با این فیلترها آشنا می دارند. تصویر زیر اثر هر یک از این فیلترها را نشان می دهد:

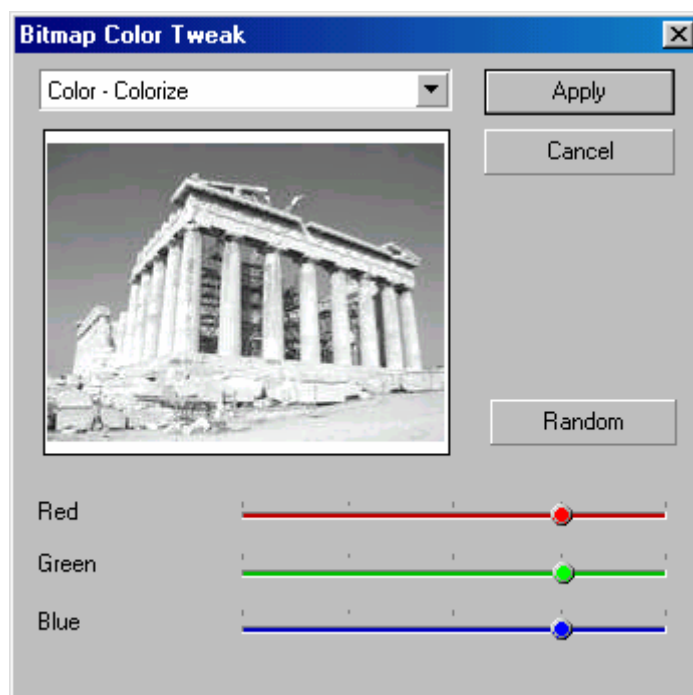


Interactive Effect

در این قسمت نیز سه جلوه تصویری دیگر برای عکسها ارائه شده است. جلوه Wrap حالت خمیدگی و مچالگی در عکسها ایجاد می کند، Impressionist برای مخدوش کردن عکسها است و Metal Light برای اعمال رنگی با ته مایه های رنگ فلزات است. لازم به ذکر است که این جلوه ها در کادر خصوصیت آنها و توسط نشانه گر موس اعمال می شود.

Color Tweak

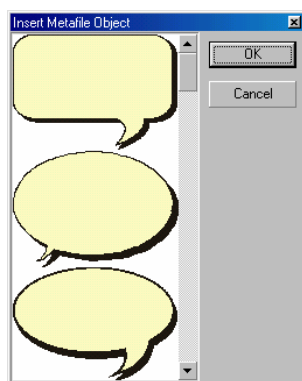
از این ویژگی کارآمد برای تغییر رنگ، کنتراست (تمایز رنگ)، کیفیت، مقدار اشباع رنگ و... عکس ها استفاده می شود.



این قسمت گزینه های متعددی را در رابطه با تغییر رنگ و کیفیت عکس ها ارائه می دهد. هنگامی که یک گزینه را انتخاب نماییم بسته به نوع آن پارامترهای آن در زیر کادر Color Tweak ظاهر می شود.

Bubble

برای ایجاد شکل حباب است. این ابزار از منوی ابزار قابل دسترسی است. کاربرد این اشکال به هنگامی است که خواهیم مطلبی را به طور خاص بازگو کنیم، البته استفاده از این ابزار بستگی به سلیقه طراح دارد.



Insert Sparkle

این قسمت نیز اشیایی براق را به پروژه اضافه می کند. رنگ این اشیا نیز قابل تغییر است. این ابزار از منوی ابزار قابل دسترسی است.



Crop

این ابزار جهت برش قسمتی از تصویر به صورت برشهای مستطیلی می باشد. این ابزار هنگام برش هیچ تاثیری بر درجه تفکیک (Res) و یا ابعاد تصویر نمی گذارد.

جهت برش تصویر به صورت زیر عمل نمایید:

۱- شئی تصویر را انتخاب کنید.

۲- از منوی Effect قسمت Crop را برگزینید.

۳- یک مستطیل مجازی که محدوده برش را مشخص می کند بر روی تصویر بکشید.

چنانچه از نتیجه کار راضی نبودید و مایل به بازسازی تصویر اصلی می باشید، می توانید از منوی Effect گزینه Restore original را بگزینید و چنانچه از قسمت برش یافته راضی می باشید می توانید از منوی Effect گزینه

Make new original را برگزینید، این کار قسمت برش یافته را همانند یک تصویر اصلی می کند که بعداً باز می توانید به ویرایش آن بپردازید.

Make mew original

اگر قصد تغییر اندازه و یا ویرایش قسمت برش یافته از تصویر را داشته باشید بدون اعمال ویژگی Make mew original به همراه تغییر اندازه قسمت برش یافته، کل تصویر اصلی نیز تغییر می کند. حال برای جلوگیری از این حالت می توان از منوی Effect گزینه Make mew original را برگزید.

اسکرپت نویسی

Multimedia Builder

اسکرپت نویسی

ویرایشگر اسکرپت

نابت ها

متغیر

آرایه

عبارت شرطی

حلقه ها

جادوگر اسکرپت



به قسمت برنامه نویسی MMB خوش آمدید. همیشه با شنیدن برنامه نویسی تا حدودی تردید و ترسی در رابطه با یادگیری آن ایجاد می شود. اما شیوه ی برنامه نویسی که MMB ارائه می دهد این شک و تردید را بی مورد می کند. به طور کلی قواعد و فرامینی که در این برنامه تعبیه شده است در مقایسه با سایر زبانهای برنامه نویسی بسیار ساده تر و قابل فهم تر است.

در این بخش ابتدا بر اصول برنامه نویسی تمرکز می شود، اصولی از قبیل مفاهیم ثابت ها، متغیرها و... سپس عبارت های شرطی و حلقه ها معرفی می شوند و در پایان نیز فرامین ارائه شده توسط MMB مورد بحث و بررسی قرار می گیرد. توصیه می شود که مطالب این قسمت را به دقت بررسی و مطالعه نمایید چرا که هر چه مهارتتان در برنامه نویسی بیشتر و بهتر شود به همین نسبت می توانید نرم افزارهای قویتری تولید نمایید.

اسکرپت نویسی (Scripting)

یکی از سوالاتی که همیشه به ذهن کاربران MMB خطور می کند اینست که آیا آنها برنامه نویسی محسوب می شوند یا خیر؟ در جواب باید گفت بله برنامه نویسی محسوب می شوند حتی اگر یک خط برنامه رانوشته باشند. این مطلب اصلاً غیرطبیعی نیست چرا که هنگامیکه شما دستگاه پخش و ضبط ویدئویی خانگی خود را برای ضبط خودکار یک برنامه از تلویزیون تنظیم می نمایید عمل برنامه ریزی صورت پذیرفته است. نوشتن نمایشنامه خوب یک نوع برنامه ریزی و برنامه نویسی است، مهم نیست که چقدر بی ارتباط با مقوله ی کامپیوتر می باشد. به هر حال شما با انجام فعالیتهای مذکور برنامه نویسی محسوب می شوید.

درحقیقت، پس از قرنهای برنامه ریزی به صورت های گفته شده اخیراً پا در عرصه برنامه نویسی ماشینی نهاده ایم، ماشینها به واسطه برنامه ها کارها را ساده تر نموده اند. بله هر کاری که انجام کارهای ما را ساده تر نماید برای ما خوشایند است و قریب به یقین همه آرزو داریم تا بتوانیم به نحوی از این قدرت ساده سازی بهره مند شویم. اما آرزوی شما برآورده شده است و آن چیزی نیست جز MMB که شما را در محقق شدن آرزوهایتان یاری می کند.

اصول برنامه نویسی در MMB

در کار با MMB همانند سایر برنامه ها قواعد و قوانین خاصی وجود دارد که باید از آنها پیروی کنیم. در مقایسه با زبان های بشری و سایر زبانهای برنامه نویسی MMB یکی از ساده ترین و سریعترین زبانها را جهت فراگیری برنامه نویسی ارائه می دهد.

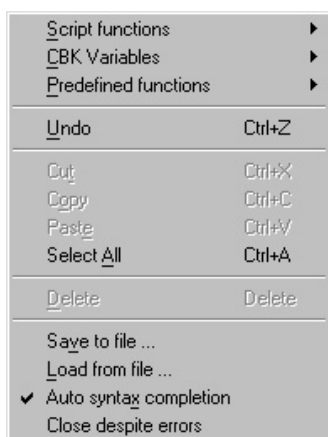
دانستنیهای اصلی

۱- برنامه نویسی توسط خطوط صورت می گیرد. همانند اینکه شما اقدام به تایپ یک داستان می نمایید و خط به خط آن را توسط ماشین تایپ می کنید. عمل نوشتن در MMB توسط صفحه کلید صورت می گیرد تا کدهای برنامه نوشته شود (یا همان داستان نوشته شود). و همان طور که ممکن است حدس زده باشید یک جمله در هر خط.

۲- جمله ها در کد برنامه هایی که توسط MMB تولید می شوند خط کد (codeline) نامیده می شوند. هر خط به طور معمول نمایانگر یک دستور می باشد البته در این باره استثناهایی هم وجود دارد.

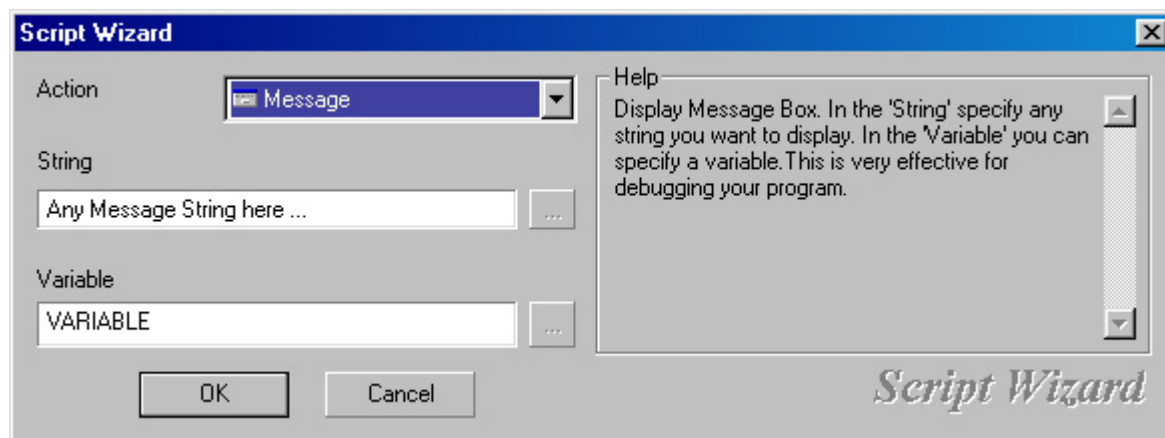
۳- احتیاجی به حفظ کردن نام فرامین MMB نیست. MMB به شما به چند طریق زیر در نوشتن کد برنامه کمک می نماید.

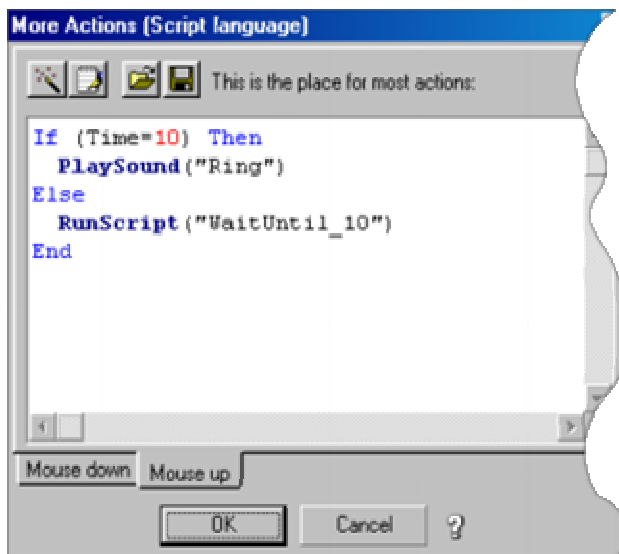
۴- کامل کردن قواعد (Syntax completion) : MMB در هنگام تایپ کد برنامه توسط شما به طور دائم نظارت می‌کند که چه چیزی را می‌نویسید و چنانچه براساس متون نوشته شده فرمانی را از لیست فرامین خود مناسب دید باقی کد برنامه را به صورت خودکار کامل می‌نماید، حال تنها چیزی که باید توسط شما تایپ شود یکسری پارامترهای مخصوص و متغیرهای مدنظر شماست.



۵- منوی فرامین با استفاده از راست کلیک (Rightclick menu): یک لیست سریع فرامین توسط عمل راست کلیک در اختیار شماست تا بتوانید با انتخاب سریع فرمان مورد نظر عمل کدنویسی را تسریع بخشید.

۶- جادوگر اسکریپت (Script wizard): توسط این ویژگی همواره یک لیست درونی از فرامین در اختیار شماست که با انتخاب هر فرمان شرح مختصری از آن به همراه محل ورود پارامترهای مورد نظر جلوی روی شما ظاهر خواهد شد و همان طور که در ابتدای این بحث بیان شد نیازی به حفظ کردن اسامی فرمان ها نمی‌باشد.





۷- راهنمای MMB هر وقت که احتیاج به اطلاعات و جزئیات بیشتری داشتید به راهنمای MMB رجوع نمایید تا مشکلاتان را رفع نمایید.

۸- متمایزسازی و پیکربندی کدهای نوشته شده MMB به طور خودکار متنهای مختلف را رنگی می کند تا در نهایت سبب آسانی فهم و افزایش خوانایی کد شود. به علاوه MMB به طور خودکار نحوه قرارگیری فرامین را تنظیم می کند. در شکل، شما پیکربندی کدهای نوشته شده را ملاحظه می نمایید هر خط کد بیان کننده شرایط و فرامین است که برنامه ما را شکل می دهند در مثال بالا برنامه به قرار زیر است:

- چنانچه ساعت ۱۰ باشد ساعت زنگ می زند.
- اگر ساعت ۱۰ نباشد برنامه تا ساعت ۱۰ صبر می کند.

اما چه جزئیاتی را می توان از مثال بالا دریافت نمود؟

- اسامی (ساعت) با رنگ مشکی نشان داده می شود.
- اعداد به صورت قرمز نمایان می شوند.
- شرایط و فرامین به صورت آبی نمایان می شوند (اگر ساعت ۱۰ باشد / اگر ساعت ۱۰ نباشد).
- یکسری فعالیتهای تعیین شده به صورت مشکی درمی آید.
- نحوه قرارگیری خطوط سبب خوانایی بیشتر کد می شود.

قواعد برنامه نویسی (Scripting Syntax)

فرامین از مهمترین قسمت های یک کد می باشند و توسط آنها به MMB دستور می دهید که چه کاری را انجام دهد. به عنوان مثال تصور کنید که شما فرمانده هستید (برنامه نویس) و یک ارتش الکترونیکی در اختیار دارید (MMB) هر چه که شما فرمان دهید صورت می پذیرد اما نه هر فرمانی! فرامینی که MMB قادر به شناسایی و درک آنهاست را باید صادر کنیم.

در MMB چندین گونه فرمان موجود است، اما نترسید آنها تقریباً به یک صورت هستند با این تفاوت که نامهای متفاوتی دارند.

Command("Parameter1", "Parameter(s) 2")

تمامی فرامین در MMB حداقل از ظاهری شبیه فرمان بالا برخوردار هستند اما به مثال زیر توجه فرمایید.

همانند اینکه شما می‌توانید گلدان را بدون گل داشته باشید در MMB هم می‌توانید فرمان با پارامتر و یا بدون



گلدان بدون گل همانند
فرمان بدون پارامتر



گلدان دارای گل
فرمان با پارامتر

پارامتر داشته باشید در جدول زیر گونه‌های مختلف فرامین توصیف شده است:

گونه فرمان	مثال
بدون پارامتر	Commandname ()
با یک پارامتر	Commandname ("parameter")
با چند پارامتر	Commandname ("parameter1", "parameter2")
با چند متغیر	Commandname ("variable1", "variable2")

توصیفات ذکر شده تنها صورت کلی از انواع دستورها هستند. جهت اطلاعات بیشتر با ما همراه باشید.

پارامترها (Parameters)

یکسری از داده‌ها را ارائه می‌دهند که توسط فرامین استفاده می‌شوند چندین گونه از این پارامترها موجود می‌باشد.

مثال فرمان	گونه پارامتر
Commandname("hello, MMBers")	متن (text)
Commandname("4329")	عدد (text)
Commandname("my text object")	نام اشیا (object lable)
Commandname("60")	زمان / درصد (time/ percentage)
Commandname("c:\myvoice. Mp3")	نام فایل / مسیر (file name/ path) پارامترهای ثابت fixed paramerters

نگران نباشید در ادامه به طور کامل درباره‌ی پارامترها مطالب تکمیلی را خواهید آموخت. اما توجه نمایید که چگونه پارامترها از قسمت فرمان متمایز شده‌اند، بلکه آنها درون یک جفت (" ") کوتیشن (علامت نقل قول) محصور شده‌اند و چنانچه بیشتر از یک پارامتر موجود باشد بین آنها کاما قرار می‌گیرد.

Commandname("param 1")

Commandname("param 1", "param 2")

گذشته از همه‌ی مطالب گفته شده باید به این حقیقت توجه کنیم که اسکریپت‌های MMB به صورت سطری و یکی پس از دیگری پشت سر هم نوشته می‌شود. در MMB هیچ محدودیتی جهت تعداد خط‌های یک برنامه وجود ندارد. و به خاطر بسپاریم که در MMB فرامین نوشته شده حساس به حروف (case sensitive) نیستند، بدین معنی که COMMAND NAME با command name هیچ فرقی ندارد اما باید توجه داشت که متغیرها و پارامترها حساس به حروف هستند یعنی کوچکی و بزرگی حروف در آنها حائز اهمیت است.

توضیح اسکریپت (Remark)

```

This is the place for most actions:
** Error Text='Error'
توضیحی که خود برنامه به واسطه مشاهده خطا اضافه کرده
**Below is if statement;
توضیحی که توسط برنامه نویس اضافه شده است
If (a=1) Then
  Message ("", "a")
Else
  Message ("", "b")
End

```

توضیحات یکی از قسمتهای مهم در کدنویسی هستند چرا که سبب خوانایی کد نوشته شده می شوند. شما می توانید آنها را میان خطوط کد قرار دهید و یا در ابتدای کد پروژه بنویسید. قاعده ی نوشتن توضیح تنها به این صورت است که ابتدا باید دو ستاره (* *) تایپ کنیم و هر آنچه را که خواستیم در ادامه بنویسیم.

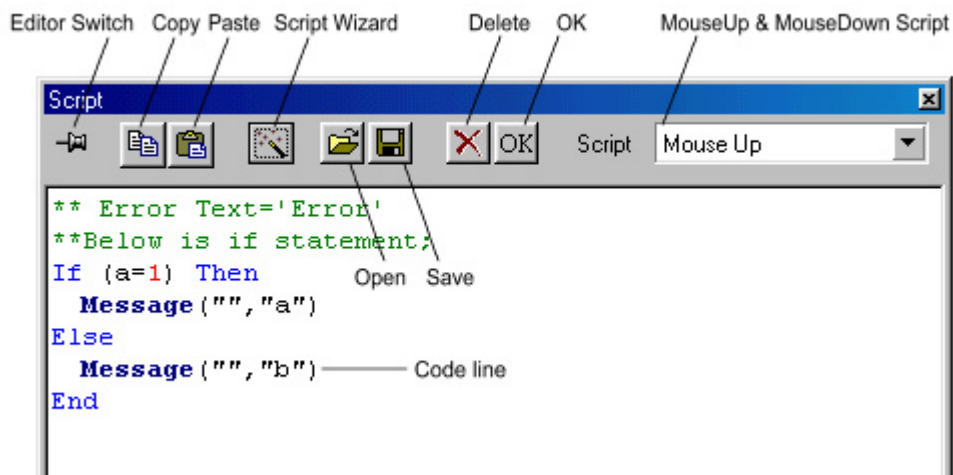
اما شرایطی هست که MMB خود توضیحی را به کد برنامه اضافه می کند و آن در صورتی است که MMB حین بستن پنجره اسکریپت نویسی که کد برنامه را از جهت صحت بررسی می کند چنانچه خطایی را مشاهده کند آن را با اضافه کردن ۲ ستاره و کلمه Error به توضیح تبدیل می کند. می توانیم جهت رفع خطا با برداشتن ۲ ستاره و کلمه Error و اعمال درست کد از ایجاد این نوع توضیح پرهیز کنیم.

ویرایشگر اسکریپت (Script Editor)

اما اجازه دهید قبل از فراگیری نحوه عملکرد و کاربرد ویرایشگر به معرفی اشیا یی از MMB که دارای پنجره اسکریپت نویسی هستند بپردازیم:

- Text (متن)
- Button (دکمه)
- List Box (کادرلیست)
- Script object (شی اسکریپت)
- Hot-spot
- Image matrix (شبکه تصویر)
- اشیا گرافیکی

اما اکنون نوبت به خود ویرایشگر می رسد:



جدول زیر در رابطه با قسمت‌های مختلف پنجره ویرایشگر توضیح می‌دهد.

قسمت	توضیح
Editor switch	این دکمه سبب تعویض پنجره اسکریپت نویسی از حالت کوچک به پیشرفته می‌شود.
Copy/Paste	قسمت کپی برداری و چسباندن متون از کلیپ برد (Clipboard).
Script wizard	این دکمه wizard فرامین را بازگشایی می‌کند که در انتخاب هر فرمان توضیح مختصر آن و کادر ورود پارامترها هم ظاهر می‌شود.
Open/Save	MMB را قادر می‌سازد که اسکریپت‌ها را از فایل‌های متنی rtf, text بازگشایی و بارگذاری نموده و یا در صورت لزوم با همان فرمت‌های مذکور ذخیره نماید.
MouseUp/Mouse Down	با توجه به این انتخاب پنجره‌ای جدید جهت نوشتن کد ظاهر می‌شود. Mouse up بدین معناست که هنگامی که نشانه‌گر موس بر روی شیء حاوی اسکریپت از حالت کلیک خلاص می‌شود کد اجرا شود. Mouse down زمانیست که با موس کلیک کرده و هنوز دکمه موس را رها نکرده‌ایم.

مزیتی دیگر که پنجره ویرایشگر دارد این است که با ۲ بار کلیک بر روی یک فرمان، Wizard برنامه MMB ظاهر می‌شود تا سریعاً بتوانیم در صورت لزوم تغییرات را اعمال کنیم.

تا کنون متوجه شدید که اسکریپت‌های MMB کجا باید نوشته شود، پس به جاست که از آن درست استفاده کنیم و سعی در مراقبت از کدهایمان داشته باشیم چرا که ساعت‌های زیادی را صرف نوشتن آنها خواهیم کرد.

بررسی عملی ویژگی های ذکر شده

پس از بررسی تئوریک قسمت های پیشین اکنون نوبت آن است تا دست به کار شوید و ویژگی های برشمرده شده را عملاً بیازمایید، به همین منظور در این بخش راهنمای ساخت برنامه ای که در آن عمل کشیدن و رها کردن شی توسط موس انجام می شود را ارائه کرده ایم. توصیه می شود در حین ساخت برنامه ویژگی های ذکر شده را بررسی نمایید. در این خود آموز فرا می گیرید که چگونه می توان یک شی را توسط موس به مکانی دیگر منقل نمود یا به بیانی دیگر عمل Drag & Drop.

۱- یک شی (مثلاً یک عکس یا یک مستطیل) را در صفحه قرار دهید. ما در این تمرین از یک مستطیل استفاده کرده ایم.

۲- کد زیر را به درون رویدادMouseDown شی منتقل نمایید:

```
i=1
X=ObjectX(Rectangle)
Y=ObjectY(Rectangle)
Xm=MouseX()
Ym=MouseY()
StartX=Xm - X
StartY=Ym - Y
RunScript("Script")
```

۳- سپس یک شی اسکرپت ایجاد نمایید و کد زیر را به درون آن منتقل نمایید:

```
If (i=1) Then
Xm=MouseX()
Ym=MouseY()
NewX=Xm - StartX
NewY=Ym - StartY
MoveObject("Rectangle1", "NewX, NewY")
Refresh()
ScriptTimer("Script", "0")
End
```

۴- نهایتاً عبارت $i=0$ را به قسمت رویداد MouseUp شی مستطیل اضافه کنید تا برنامه کامل شود.

۵- کلید F5 را برای اجرای پروژه فشار دهید.

متغیر Variable

تغییرات، تغییرات، باز هم تغییرات. این روزها تغییرات بسیار شده و در هر جنبه زندگی بسیار است. مثلاً هنگامی که شما برای خرید به فروشگاه مراجعه می کنید و تمام پول خود را در آن جا صرف می کنید متغیر در این جا کیف پول شماست چرا که از حالت پر به خالی تبدیل شده است.

برنامه شما بستگی به متغیرها دارند و مرتباً داده هایی که به متغیرها اختصاص داده شده اند در حال تغییر هستند از متغیرها به عنوان ابزاری جهت نگه داری داده ها و تغییر آنها به صورت موقت استفاده می شود. در MMB دو نوع متغیر موجود است:

- **متغیر عددی (Integer):** که برای ذخیره اعداد استفاده می شود. این نوع متغیر برای انجام اعمال ریاضی حائز اهمیت است.

- **متغیر رشته ای (String):** متغیر رشته ای جهت ذخیره انواع کاراکترها، کلمات، جملات و پاراگرافهاست.

اما متغیرها شامل ۲ قسمت هستند:

- **برچسب متغیر:** نامی است که به متغیر اختصاص داده می شود و متغیر بعداً توسط همین نام فراخوانی می شود این نام می تواند تنها متشکل از حروف و یا ترکیبی از حروف و عدد باشد مثل My Var, Test 12 و ...

فراموش نکنیم که شروع نام متغیرها نباید عدد باشد. به عنوان مثال نام 3th_Year به عنوان نامی غیر معتبر در نظر گرفته می شود.



- **محتوی متغیر:** که برای متغیرهای عددی همان عدد است و برای متغیر رشته ای همان حروف.

توجه نمایید که در MMB لازم نیست که نوع متغیر از پیش تعیین شود و این امر به صورت خودکار به وسیله برنامه صورت می گیرد.

متغیرهای عددی (Numerical Variable)

برخی از مواقع ما احتیاج به انجام عملیات های ریاضی داریم و اینجاست که پای متغیرهای عددی به میان می آید. در MMB ۲ نوع عدد موجود است که ما از آنها جهت ذخیره در متغیرها استفاده می کنیم:

- اعداد حقیقی (اعم از اعشاری) مثل 5.4 , 21.900 و $\frac{2}{3}$

- اعداد صحیح (گرد شده) مثل 4، 12، 1000

توصیه می شود که حتی الامکان از نامهای دارای مفهوم که نه خیلی کوتاه و نه خیلی بلند هستند استفاده شود. اما مهمترین قسمت در زمینه متغیرها انتساب آنهاست که برای این کار بایستی ابتدا:

الف) نام متغیر را می نویسیم:

Glass

ب) سپس به دنبال آن علامت = را قرار می دهیم.

Glass =

پ) و در نهایت مقدار متغیر را پس از علامت مساوی قرار می دهیم:

Glass =1555

اما این نکته مهم است که ما بتوانیم مقدار یک متغیر را به متغیر دیگری اختصاص دهیم، برای انجام این عمل کفایت که به ترتیب زیر عمل نماییم:

الف) نوشتن متغیر مقصد: (منظور همان است که در نهایت مقدار به آن داده می شود)

face2

ب) علامت = را در ادامه قرار می دهیم:

face2 =

پ) سپس نوشتن متغیر اصلی و منبع را بعد از گذاشتن علامت مساوی انجام می دهیم:

$$\text{face2} = \text{face1}$$

و چنانچه مقدار متغیر face1 ، 255 باشد همین مقدار متعاقباً به face2 انتقال داده می شود. از تکنیک متغیر به متغیر (Variable to Variable) اکثراً در مواقعی که احتیاج به انجام اعمال ریاضی است استفاده می شود.

حال که صحبت از اعمال ریاضی است باید گفت که می توان از چهار عمل اصلی ریاضی در اختصاص دهی و تعریف متغیرها سود جست. برای توضیح بیشتر به جدول زیر توجه کنید:

عملگر	شرح	مثال
+	این عملگر، عمل جمع را بین ۲ یا چند عدد و همچنین ۲ یا چند متغیر عددی انجام می دهد و نتیجه را به متغیر مربوطه اختصاص می دهد.	$a = 12 + 43$ $a = b + c$ Weekend = Sat + Sun Days of week = 1 + 2 + 3 + 4 + 5 + 6 + 7
-	این عملگر، عمل تفریق را بین ۲ یا چند عدد و همچنین ۲ یا چند متغیر عددی صورت می دهد و نتیجه را به متغیر مربوطه اختصاص می دهد.	$a = 43 - 20$ $b = c - a$ Workday = week - weekwnd Tax = 100 - 11 - 65
×	این عملگر، عمل ضرب را بین ۲ یا چند عدد و همچنین بین ۲ یا چند متغیر عددی صورت می دهد و نتیجه را به متغیر مربوطه اختصاص می دهد.	$a = 43 * 12$ $a = c * b$ Month = week * 4 Sales = 24.99 * buyers
/	این عملگر، عمل تقسیم را بین ۲ یا چند عدد و همچنین بین ۲ یا چند متغیر عددی صورت می دهد و نتیجه را به متغیر مربوطه اختصاص می دهد.	$a = 43 / 12$ $a = c / b$ Month = Year / 12 SpecWeight = Spec Quantily / 30

اما با استفاده از همین چهار عمل اصلی ترکیباتی حاصل می شود که به قرار زیر است:

- عملیات ریاضی با ۲ شماره و یا بیشتر

$$a = 432 / 21$$

MyNum=4+1+5

-عملیات ریاضی با دو متغیر یا بیشتر:

```
a = b*c
Price = Quantity * Item price - Taxes
```

-عملیات ریاضی با ترکیبی از اعداد و متغیرهای عددی

```
a= b + 42* Number of clicks
Price = 2032 * Quantity + 83/ Number of Members
```

هنگامیکه در سمت راست این انتساب ها مقدار محاسبه می شود بلافاصله به متغیر نسبت داده می شود و پس از آن در هر کجا می توانیم از آن استفاده نماییم. به عنوان مثال:

```
Price = 2032 * Quantity + 83/Number of Members
Message("Price of your order is: ", "Price")
```

در این اسکرینپت ابتدا مقدار محاسبه می شود و سپس به متغیر Price نسبت داده می شود سپس توسط فرمان Message به نمایش در می آید.

متغیر رشته ای (String Variable)

متغیرهای رشته ای در همه جا هستند، حروف، کلمات، جملات، پاراگرافها و...از جمله متغیرهای رشته ای هستند. مهمترین قسمت یک متغیر رشته ای برچسب (Label) آن می باشد. برچسب متغیر رشته ای از یک اسم و کاراکتر " \$ " تشکیل شده است. MyName\$,Title\$ از جمله برچسب های معتبر می باشند. باید همیشه به خاطر داشته باشیم که پس از برچسب متغیر رشته ای حتماً علامت \$ را باید درج کنیم چرا که توسط این علامت MMB می فهمد که این متغیر رشته ای می باشد. ضمناً توصیه می شود که حتی الامکان از اسامی دارای مفهوم که نه خیلی زیاد کوچک و نه خیلی زیاد بزرگ باشد برای نامگذاری متغیرها استفاده نمایید چرا که این امر سبب ایجاد سهولت و خوانایی برنامه شما می شود.

اما عمل انتساب در MMB برای متغیرهای رشته ای چگونه صورت می گیرد؟

الف) ابتدا برچسب متغیر را می نویسیم:

Sentence

ب) پسوند \$ را اضافه می کنیم:

Sentence\$

پ) در ادامه علامت مساوی را قرار می دهیم:

Sentence\$=

ت) و در نهایت محتوی متغیر را که بایستی بین یک جفت () محصور شود را می نویسیم:

Sentence\$ = ' This is an example '

همچنین برای افزودن Backslash (\) به صورت زیر ۲ بار بایستی Backslash را تایپ نماییم:

نتیجه می دهد

Path\$ = ' C:\Windows\ ' → C:\Windows\

اما برای افزودن کاراکتر (') به محتوی رشته می توانیم از کاراکتر (\) قبل از علامت (') استفاده کنیم، همانند مثال زیر:

String\$ = ' Ali\'s Book ' → Ali's Book

همانند متغیرهای عددی در اینجا هم می توانیم محتوی یک متغیر را به یک متغیر دیگر مستقیماً نسبت دهیم. برای این منظور:

الف) ابتدا بر چسب متغیر مقصد را می نویسیم:

MySecondString\$

ب) علامت = را در ادامه می نویسیم:

MySecondString\$ =

پ) سپس نام متغیر منبع را می نویسیم:

MySecondString\$ = MyFirstString\$

چنانچه محتوی متغیر منبع (This is an example) باشد همین عبارت عیناً به متغیر مقصد نسبت داده می شود. جهت ادغام رشته ها و انتساب آنها به یک متغیر دیگر از عمل جمع استفاده می شود. استفاده از عمل جمع هیچ محدودیتی ندارد. به عنوان مثال اگر متغیر اول به صورت زیر باشد:

FirstPart\$ = ' to be - or not to be ... '

و متغیر دوم به صورت زیر باشد:

```
SecondPart$ = ' The question is now '
```

برای ادغام دو متغیر و انتساب آن همانند زیر عمل می کنیم.

```
CompleteSentence$= FirstPart$ + SecondPart$
```

که نتیجه آن به صورت زیر می شود:

to be – or to be not... The question is now

یادآوری می کنیم که همین نتیجه را می توانستیم از ترکیب ۲ محتوا و سپس انتساب آنها بگیریم.

اما جدول زیر اغلب ترکیبات را در انتساب متغیرهای رشته ای بیان می کند.

شرح	مثال
انتساب متن به متغیر: این متن انتساب یافته ثابت است .	Name\$= ' Alex Green'
انتساب متغیر به متغیر: در این حالت مقداریکی از متغیرهای رشته ای به متغیر دیگری داده می شود.	User password\$=Edit Box\$
انتساب متغیر + متن (یا بر عکس) به متغیر: در این حالت ابتدا توسط عمل ادغام محتوی متغیر تشکیل می شود و در نهایت به متغیر دیگر تخصیص داده می شود.	A\$=User Name\$+ ' 19 Years old'

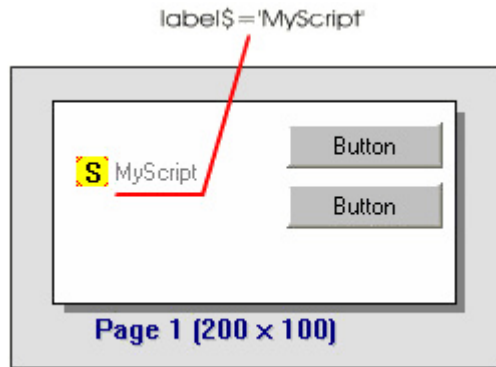
در زیر مثالی از گونه ای که در آن از ادغام استفاده شده آمده است:

```
Player1$= 'Hamed'  
Player2$= 'Hadi'  
MatchPlayers$ = Player1$ + 'Vs' + Player2$  
Message("And now, match: ", "MatchPlayers$")
```

که در نهایت عبارت And now, match: Hamed Vs Hadi در کادر پیغام نمایان می شود.

متغیرهای رشته ای به عنوان برچسب اشیاء

انعطاف پذیری MMB این امکان را فراهم می کند که بتوانید از یک متغیر رشته ای به عنوان منبعی برای برچسب اشیاء استفاده کنید.



با این عمل در هر فرمانی که باید در آن برچسب شی را قرار داد می توان از متغیر رشته ای به جای نام ثابت شی استفاده کرد. به عنوان مثال در فرمان `RunScript` که یک پارامتری می باشد می توان به جای اسم ثابت شی `Script` از متغیر رشته ای استفاده کرد همانند مثال زیر:

```
Label$ = 'MyScriptScript'  
RunScript ("Label$")
```

استفاده از این ویژگی در زمانی مفید است که اشیاء تقریباً نامی متحدالشکل داشته باشند مثل `Text3`, `Text2`, `Text1` و ... و ما بخواهیم در حلقه ها از نام اشیاء استفاده کنیم.

ثابت ها (Constants)

چیزهای زیادی وجود دارند که ما می‌توانیم آنها را تغییر دهیم. اما با این حال تعداد زیادی از چیزها هم هستند که قادر به تغییر دادن آنها نیستیم. اجازه بدهید تا در مورد حالت دوم بحث نماییم. زمین یکبار در ۲۴ ساعت به دور خودش می‌چرخد و ما قادر نیستیم که آن را تغییر دهیم، سنگی را به بالا پرتاب می‌کنیم اندکی بعد به زمین می‌خورد و هیچ راه دیگری نیست. اما رایانه‌ها، همیشه از ثابت‌ها استفاده می‌کنند و بدون آنها ادامه کار میسر نیست. به عنوان مثال رایانه‌ها احتیاج به ولتاژ ثابت دارند و نبود این ثابت سبب ایجاد اختلال می‌شود. ثابت‌ها در دیدگاه نرم‌افزاری با یک نام یا برچسب شناسانده می‌شوند.

موارد استفاده از ثابت‌ها در MMB :

- برای اطلاعات سیستمی و چند رسانه‌ای گوناگون مثل اندازه صفحات، زمان فایل‌های صوتی و تصویری، مسیرهای پروژه در ویندوز، عناوین فایل‌ها و ...

- به عنوان پارامترهای یک فرمان که به MMB بگوید دقیقاً چه کاری را انجام دهد، مثلاً این پنجره را باز کن، آن را تغییر بده و ...

اما همان‌طور که ممکن است حدس زده باشید، بعضی از ثابت‌ها می‌توانند تغییر کنند البته غیرمستقیم. به عنوان مثال اگر شما زمان و عنوان یک فایل صوتی را تغییر دهید محتوی ثابت تغییر می‌کند و MMB خود این تغییرها را اعمال می‌نماید.

یکی از ویژگی‌های مهم MMB اینست که می‌تواند مقدار ثابت‌ها را دریافت و در متغیرهای رشته‌ای (String) یا عددی (Integer) ذخیره نماید و به همین خاطر دامنه‌ی فعالیت ما بر اساس اطلاعات دریافتی بیشتر می‌شود.

جهت انتقال اطلاعات از یک ثابت به یک متغیر همانند زیر عمل می‌کنیم. (ابتدا متغیر عددی)

الف) برچسب متغیر عددی را می‌نویسیم:

Current year

ب) یک علامت مساوی در ادامه قرار می‌دهیم:

Current year =

ج) و در نهایت ثابتی را که می‌خواهیم از آن اطلاعات بگیریم می‌نویسیم:

Current year = CBK- year

اما جهت انتقال اطلاعات از ثابت به متغیر رشته‌ای به صورت زیر عمل می‌کنیم.

الف) برچسب متغیر رشته‌ای را می‌نویسیم:

windows_ver\$

ب) در ادامه یک علامت مساوی قرار می‌دهیم:

windows_ver\$ =

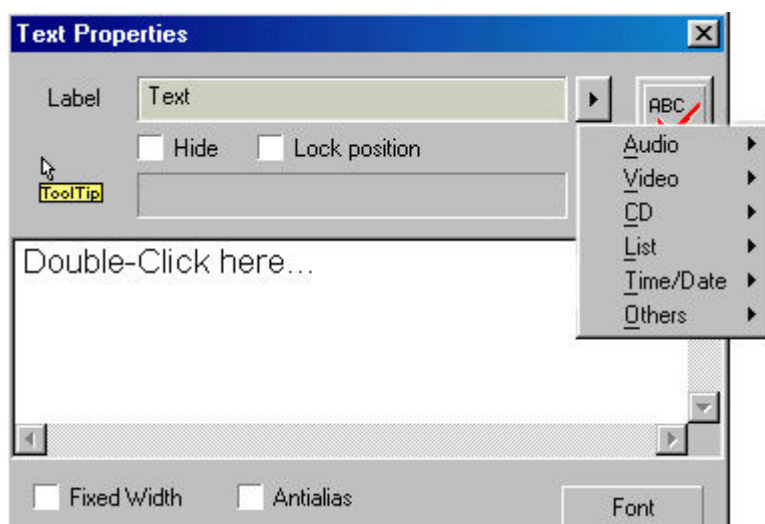
ج) و پس از آن ثابتی را که می‌خواهیم اطلاعات از آن دریافت کنیم می‌نویسیم

windows_ver\$ = WinVer()

همان طور که در ۲ حالت قبل مشاهده کردید بعضی از ثابت‌ها اطلاعات را به صورت عددی باز می‌گردانند و برخی بصورت رشته‌ای (بستگی به محتوای ثابت‌ها دارد).

ثابت‌های CBK

کمی عجیب به نظر می‌رسند، اما آنها چیزی جز ثابت‌های سیستمی، چند رسانه‌ای و... نیستند. ثابت‌های CBK جزئی از خصوصیات و امکانات متن محسوب می‌شود. بنابراین به سراغ آنها می‌رویم تا ببینیم که چه چیزهایی می‌توانیم پیدا کنیم، تغییر دهیم و یاد بگیریم.



با توجه به شکل پس از کادر label شی متن پیکان کوچکی است که با فشردن آن منوی حاوی ثابت‌ها که به ۶ دسته تقسیم شده‌اند ظاهر می‌شود. پس از اینکه ثابت مورد نظر را برگزیدید، اگر برنامه را اجرا نمایید مشاهده می‌کنید که شی متن مقدار ثابت را باز می‌گرداند.

در جدول زیر شرحی از ثابت‌ها و مثالهایی برای آنها آمده است.

نام CBK (Audio CBK)	شرح CBK	مثال
CBK- Total	این ثابت طول مدت فایل صوتی بارگذاری شده را نمایش می‌دهد. فرمت زمان نشان داده شده به صورت MM: SS (Seconds) ثانیه (Minutes) دقیقه) است.	21: 12
CBK- TotalSec	این ثابت طول مدت زمان فایل بارگذاری شده را در واحد ثانیه نمایش می‌دهد.	360
CBK- Time	این ثابت زمان جاری فایل صوتی بارگذاری شده را نمایش می‌دهد.	11: 08
CBK-TimeSec	این ثابت زمان جاری فایل را در واحد ثانیه نمایش می‌دهد.	110
CBK-MP3Type	نوع (Type) فایل mp3 بارگذاری شده را نمایش می‌دهد.	MPEG1 layer 3
CBK- MP3Bit	این ثابت Bitrate فایل صوتی بارگذاری شده را بازمی‌گرداند. فرمت این گونه است Number rbit/ s.	128 rbps
CBK-MP3Freq	این ثابت بسامد نمونه سازی شده فایل صوتی را بازمی‌گرداند. بسامد در واحد Hz است.	44. 1 KHz
CBK-MP3Name	این ثابت نام فایل صوتی بارگذاری شده را بازمی‌گرداند، بدون پسوند و مسیر فایل.	INFINITY
CBK-ID3Song	این ثابت عنوان فایل صوتی mp3 را که از ID3 tag گرفته شده را نشان می‌دهد.	Beauty
CBK-Channels	این ثابت کانال فایل صوتی را که یکی از ۲ گونه Stereo و Mono است باز می‌گرداند.	Stereo
CBK-ID3Artist	این ثابت نام هنرمند و تولید کننده فایل MP3 بارگذاری شده را بر اساس ID3tag باز می‌گرداند.	James- Robins
CBK-ID3Album	این ثابت نام آلبوم حاوی فایل MP3 را باز می‌گرداند. این نام بر اساس ID3tag گرفته می‌شود.	The Ultimate collection
CBK-ID3Year	این ثابت بر اساس ID3tag سال تولید فایل صوتی MP3 بارگذاری شده را باز می‌گرداند.	2002
CBK-ID3Genre	این ثابت بر اساس ID3tag طبقه (ژانر) فایل صوتی MP3 بارگذاری شده را نمایش می‌دهد	Soul
CBK-Numtracks	این ثابت تعداد Track های یک CD صوتی را باز می‌گرداند.	14

Volume	این ثابت حجم صدای اصلی (Master Volume) را در واحد درصد بیان می کند (0 تا 100).	70
CBK-Curltemlist	این ثابت انتخاب جاری در SongList را نمایش می دهد.	After All
CBK-NuminList	این ثابت تعداد قسمتهای موجود در SongList را نمایش می دهد.	31
CBK-TotalList	این ثابت مجموع کل زمان فایل های صوتی موجود در Song List را نشان می دهد. فرمت به گونه MM:SS است.	135:28
CBK-TotalListSec	این ثابت مجموع کل زمان فایل های صوتی موجود در Song List را در واحد ثانیه نشان می دهد.	1130
(VideoCBK) CBK-Vname	این ثابت نام فایل ویدئویی در حال اجرا را بدون پسوند و مسیر فایل باز می گرداند.	The Matrix
CBK- Vtotal	این ثابت طول مدت فایل جاری ویدئویی را با فرمت MM:SS باز می گرداند.	180:19
CBK-VTotalSec	این ثابت طول مدت فایل جاری ویدئویی را در واحد ثانیه باز می گرداند.	10700
CBK-VTime	این ثابت زمان جاری فایل در حال اجرا را با فرمت MM:SS باز می گرداند.	94:32
CBK-VtimeSec	این ثابت زمان جاری فایل ویدئویی را در واحد ثانیه باز می گرداند.	48000
CBK-Vtotal Frames	این ثابت تعداد فریمهای فایل ویدئویی را باز می گرداند.	394025
CBK-VFrame	این ثابت فریم جاری فایل ویدئویی را باز می گرداند.	249253
(Data &Time CBK) CBK – Year	سال جاری را نشان می دهد.	2004
CBK-Month	این ثابت ماه جاری از سال جاری را نشان می دهد.	July
CBK-MonthNum	این ثابت ماه جاری را بر اساس عدد نشان می دهد. (بین 1 تا 12).	7
CBK-Day	این ثابت روز جاری را نشان می دهد.	Tuesday
CBK- DayNum	این ثابت روز جاری را بر اساس عدد نشان می دهد. Wednesday=4 , Sunday=1 , Monday=2 , Tuesday=3 , Saturday=7 Thursday=5 , Friday=6 ,	3
CBK- DateNum	این ثابت روز و ماه جاری را بر اساس عدد نشان می دهد.	22
CBK- DateShort	این ثابت بر اساس فرمت DD/MM/YY تاریخ را نشان می دهد.	21/03/2004
CBK- DataLong	این ثابت بر اساس فرمت Month Day , Year تاریخ جاری را باز می گرداند.	July 23,2004
CBK-TimeHMS	این ثابت زمان جاری سیستم را بر اساس AM/PM و H:M:S (ثانیه:دقیقه:ساعت) نشان می دهد.	01:06:12 PM

CBK-Time24	این ثابت زمان جاری سیستم را بر اساس HH:MM:SS و ۲۴ ساعت نشان می دهد.	13:07:12
CBK-Hour	ساعت جاری را که بین 0 تا 23 است را باز می گرداند.	13
CBK-Minute	دقیقه جاری که بین 0 تا 59 است را باز می گرداند.	54
CBK-Second	ثانیه جاری که بین 0 تا 59 است را باز می گرداند.	12
CBK-PageName	این ثابت نام صفحه باز جاری را باز می گرداند.	Intro Page
CBK-Error	این ثابت خطاهایی را که توسط موتور صوتی FMOD یا شیء ویدئویی ارسال شده را باز می گرداند. (لیست خطاها در ضمیمه)	
CBK-URLPath	این ثابت URL جاری را از شی HTML (در صورت موجود بودن) باز می گرداند.	www.rasane.net
CBK-OpenFile	پس از استفاده از کادر محاوره ای بازگشایی فایل این ثابت نام فایل انتخاب شده را بدون مسیر آن باز می گرداند.	Test. Txt
CBK-OpenDir	پس از استفاده از کادر محاوره ای بازگشایی فایل این ثابت مسیر فایل انتخاب شده را باز می گرداند.	C:\Mydoc\
CBK-ReturnVal	فرمان Run یک مقدار بازگشتی را از برنامه اجرا شده ارسال می کند(در صورتی که کد بازگشت موجود باشد).	1
CBK-Selcolor	این ثابت مقدار RGB را از کادر محاوره ای انتخاب رنگ MMB باز می گرداند. کادر محاوره ای رنگ از سیستم RGB (R قرمز، G سبز، B آبی) استفاده می کند هر قسمت رنگ می تواند دارای سطحی بین 1 تا 255 باشد هر چه مقدار عددی بیشتر باشد بر شدت رنگ افزوده می شود. مقدار بازگشتی یک آرایه رشته ای است که قسمتها توسط کاما از هم جدا می شوند (اول:قرمز، دوم:سبز، سوم:آبی). جهت دریافت مقدار رنگ کافیست به آسانی مقدار CBK-Selcolor را به یک متغیر رشته ای نسبت دهیم . CBK- Sel color=Color\$	R,G,B 128,5,64
CBK-MP3EOF	چنانچه بر چسب یک شی اسکرپت را به CBK-MP3EOF تغییر دهیم و آن را یا در صفحه جاری و یا در Master Toplayer قرار دهیم بلافاصله پس از اتمام اجرا فایل صوتی MP3 آن اسکرپت اجرا خواهد شد. MMB ابتدا در صفحه جاری به دنبال اسکرپت می گردد و در صورت فقدان آن در Master TopLayer به دنبال آن می گردد.	

CBK-Menu	<p>چنانچه بر حسب گروهی از اشیاء جاری CBK-Menu باشد این گروه هر وقت که در خارج از محدوده آن کلیک شود ناپدید خواهد شد. این ویژگی به ما در ساختن منوهای جهنده (Pop-Up) کمک خواهد نمود. هنگامی که کاربر در صفحه کلیک نماید تمام گروههایی که با این ثابت نامگذاری شده اند محو خواهند شد به جز گروهی که زیر نشانه گر موس است.</p>	
CBK-Exit	<p>اگر نام و بر حسب یک شی حاوی اسکریپت را به CBK-Exit تغییر دهید و آن را یا در صفحه جاری و یا در Master Toplayer قرار دهید با فشردن کلید Esc بر نامه طبق معمول بسته نمی شود. و به جای آن شی که با CBK-Exit نامگذاری شده است فراخوانی می شود. از این ویژگی می توان برای تولید کادرهایی که به کاربر پیغام آیا مطمئن هستید را می دهند، استفاده کرد.</p>	

ثابت های سیستمی System Constans

اطلاعات سیستمی نقش مهمی را در تولید و تنظیم پروژه شما ایفا می کند. در MMB ثابتهای سیستمی اطلاعاتی راجع به صفحه (Screen)، پنجره (Window)، پردازشگر (Processor) و حافظه (Memory) را در اختیار ما قرار می دهد. باید توجه داشت که این ثوابت به عنوان پارامتر در سایر فرامین قابل استفاده هستند.

```
ScriptCommand("System Constant","param 2")
```

ثابت های سیستمی به شرح زیر هستند:

ScreenWidth()

این ثابت عرض صفحه را در واحد پیکسل (Pixel) و به صورت عدد صحیح نشان می دهد.

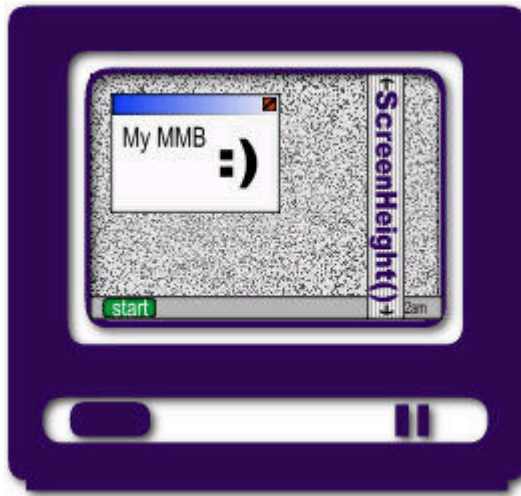


مثال:

```
Message("Display width is:", "ScreenWidth()")
```

ScreenHeight()

این ثابت ارتفاع صفحه را در واحد پیکسل و به صورت عدد صحیح نشان می دهد.



مثال:

```
Message("Display height is:", "ScreenHeight()")
```

WorkAreaWidth()

این ثابت عرض ناحیه کار را در واحد پیکسل باز می گرداند.



مثال:

```
Message("Display work area width is:", "WorkAreaWidth()")
```

WorkAreaHeight()

این ثابت ارتفاع ناحیه کار را به صورت عدد صحیح و در واحد پیکسل باز می گرداند.



مثال:

```
Message("Display work area height is", "WorkAreaHeight()")
```

MouseX()

این ثابت مقدار طول محل نشانگر موس را نسبت به سمت چپ صفحه نمایش در واحد پیکسل باز می گرداند.

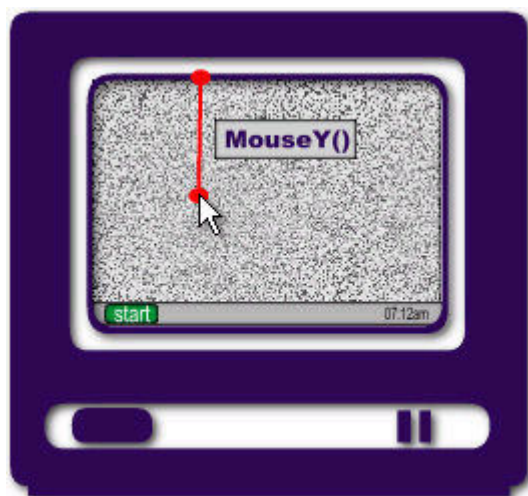


مثال:

```
Message("Current Mouse X Position :", "MouseX()")
```


MouseY()

این ثابت مقدار عرض محل نشانگر موس را نسبت به سمت بالای صفحه نمایش در واحد پیکسل باز می گرداند.



مثال:

```
Message("Current Mouse Y Position :", "MouseY()")
```

ProcType()

این ثابت نام تولید کننده، نوع و سرعت CPU را باز می گرداند.

مثال:

```
Var$ = ProcType()  
Message("Cpu in this Machine is: ", "Var$")
```

ProcFreq()

این ثابت سرعت CPU را در واحد MHz (مگاهرتز) بیان می کند.

مثال:

```
Message("CPU Freq in MHz is:", "ProcFreq()")
```

GetMemory()

این ثابت مقدار کل حافظه آزاد را باز می گذارد. این مقدار بازگشتی همانند 512/231 است و جهت جداسازی آن می توان از فرامین MMB برای رشته ها استفاده کرد.

مثال:

```
Message("Machine Memory status:", "GetMemory()")
```

UsingWinNT()

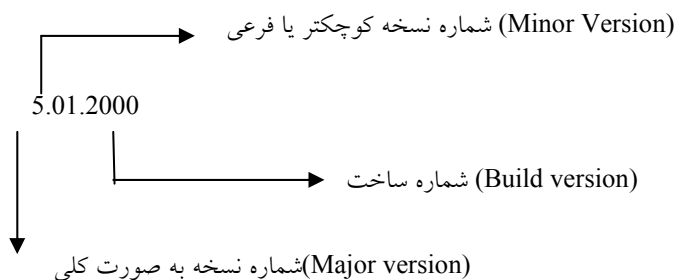
این ثابت ۲ مقدار 0 یا 1 را باز می گرداند. که اگر:

1- باشد سیستم دارای ویندوز NT است.

0 - باشد سیستم دارای ویندوز NT نیست.

WinVer()

این ثابت نسخه سیستم عامل ویندوز را به صورت رشته باز می گرداند. این مقدار بازگشتی متشکل از چند رقم است که با حرف کاما از یکدیگر جدا شده اند چنانچه می خواهید تنها قسمتی از این مقدار را نمایان سازید می توانید از فرامین MMB برای رشته ها استفاده کنید.



مثال:

```
Var$ = WinVer()  
Message("Windows Version is:", "Var $")
```

جدول استاندارد نسخه های ویندوز به صورت زیر است:

Windows 95	4.00.950
Windows 95 SP1	4.00.(>950)/4.00.(≤1080)
Windows 95 OSR2	4.(<10).(≥1080)
Windows 98	4.10.1998
Windows 98 SP 1	4.10.(≥1998)/4.10.(≤2183)
Windows 98 SE	4.10.(≥2183)
Windows Me	4.90.3000
Windows NT 3.51	3.51.1057
Windows NT 4.0	4.00.1381
Windows 2000	5.00.2195
Windows XP/SP1	5.01.2600

ScreenColors()

این ثابت مقدار رنگهای جاری مورد استفاده در سیستم را باز می گرداند. این مقدار بازگشتی عددی صحیح است.

مثال:

```
Message("Current graphic color Mode:", "ScreenColors()")
```

PubWidth()

این ثابت عرض پنجره برنامه را بدون احتساب اندازه حاشیه و عنوان، با یک عدد صحیح بیان می کند.



مثال:

```
Message("Project window width is:", "PubWidth()")
```

PubHeight()

این ثابت ارتفاع پنجره برنامه را بدون احتساب اندازه حاشیه و عنوان پنجره با یک عدد صحیح بیان می کند.



مثال:

```
Message("Project window height is:", "PubHeight()")
```

ClientWidth()

این ثابت عرض پنجره برنامه را به همراه حاشیه آن باز می گرداند می کند.



مثال:

```
Message("Project window width+broder is:", "WinWidth()")
```

ClientHeight()

این ثابت ارتفاع پنجره برنامه را به همراه حاشیه آن و عنوان پنجره بیان می کند.



مثال:

```
Message("project window Height+boarder title is:", "WinHeight()")
```

PubX()

این ثابت مختصات طول پنجره برنامه را نسبت به سمت چپ صفحه در واحد پیکسل باز می گرداند

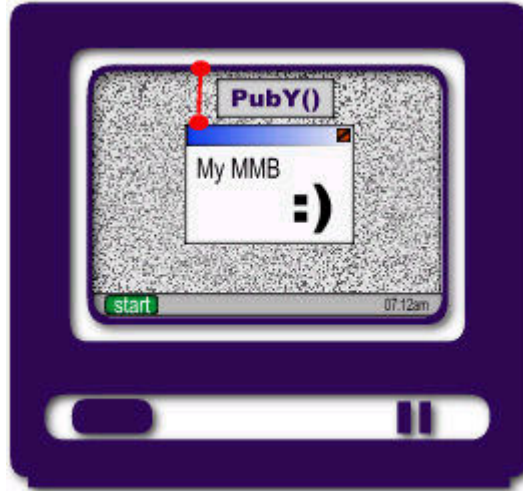


مثال:

```
Message("Project window X position :", "PubX()")
```

PubY()

این ثابت مختصات عرض (ارتفاع) پنجره را نسبت به بالای صفحه در واحد پیکسل باز می گرداند.

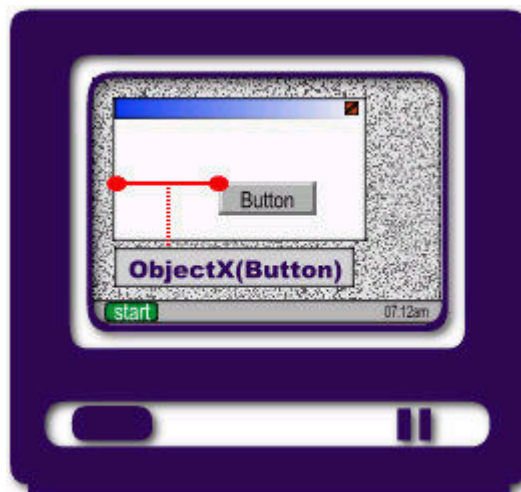


مثال:

```
Message("Project window Y position:", "PubY()")
```

ObjectX(Object Lable)

این ثابت مختصات طول شی را نسبت به منتهی علیه سمت چپ پنجره برنامه باز می گرداند. جهت مشخص کردن شی کفایت نام شی را بین ۲ پرانتز قرار دهیم.



مثال:

```
X = ObjcetX(Circle)
```

Message("X Value is:", "X")

ObjectY(Object Lable)

این ثابت مختصات عرض شی را نسبت به منتهی علیه سمت بالای پنجره برنامه باز می گرداند. جهت مشخص کردن شی کفایت نام شی را بین ۲ پرانتز قرار دهیم.



مثال:

```
Y= ObjectY(Circle)
Message("Y value is:", "Y")
```

ObjectWidth(Object Lable)

این ثابت عرض شی مربوطه را باز می گرداند نام شی را بایستی بین ۲ پرانتز قرار دهیم.



مثال:

```
ObjectWidth = ObjectWidth(Rectangle)  
Message("Object width is:", "Object Width()")
```

ObjectHeight(Object Lable)

این ثابت ارتفاع (طول) شی مربوطه را باز می گرداند. نام شی را بایستی بین ۲ پرانتز قرار دهیم.



مثال:

```
ObjectHeight = ObjectHeight(Rectangle)  
Message("Object Height is:", "Object Height")
```

IsVisible (Object Lable)

این ثابت بر اساس حالات نمایان بودن شی اعم از قابل مشاهده بودن، غیرقابل دید و یا عدم وجود شی به ترتیب مقادیر 1,0,-1 را باز می گرداند. تنها کافیسیت برای بررسی حالت شی نام شی را میان پرانتزها بنویسیم.

IsMinimized()

این ثابت بررسی می کند که آیا پنجره برنامه شما کوچک شده (Minimize) و به نوار وظیفه (Task Bar) منتقل شده است یا خیر و بر اساس آن یکی از ۲ مقدار زیر را باز می گرداند:

- 1- در این حالت پنجره برنامه Minimize شده است.
- 0- در این حالت پنجره برنامه Minimize شده نیست.

مثال:


```
Message("Project window status:", "IsMinimized()")
```

MouseButton()

این ثابت وضعیت دکمه سمت چپ موس را باز می گرداند. این ماکرو چنانچه دکمه سمت چپ در حال فشردن باشد، عدد ۱ و در غیر اینصورت عدد صفر را باز می گرداند.

MouseButton()

این ثابت وضعیت دکمه سمت راست موس را باز می گرداند. این ماکرو چنانچه دکمه سمت راست در حال فشردن باشد، عدد ۱ و در غیر اینصورت عدد صفر را باز می گرداند.

MouseButton()

این ثابت وضعیت دکمه وسط موس را باز می گرداند. این ماکرو چنانچه دکمه وسط در حال فشردن باشد، عدد ۱ و در غیر اینصورت عدد صفر را باز می گرداند.

مثال:

```
Mstate = MouseButton()  
If ( Mstate = 1) Then  
Run("<Embedded>\App.exe")  
End
```

CurrentObject()

این ثابت نام شی را که از درون آن فراخوانی می شود باز می گرداند.

مثال:

```
obj%=CurrentObject()
```

ماکروهای مسیر (Path Macros)

در MMB دو گونه مسیر یکی ثابت (Fixed) و دیگری پویا (Dynamic) موجود است.

- مسیر ثابت (Fixed Path): که به ترتیب از چپ به راست به حالت زیر است:

مسیر فایل (نام فایل) که ثابت است + پوشه، پوشه ها + نام درایو

Drive:\ Folder(s) \ *.*

به عنوان مثال:

C:\MyDoc\App.exe

به خاطر اینکه مسیرها در این نوع ثابت هستند، کاربر نمی تواند به دلخواه خود مسیر را تغییر دهد و این روزها این روش تقریباً قدیمی و غیر حرفه ای شده است.

- مسیر پویا (Dynamic Path): در این حالت ماکروهای MMB به کمک ما می آیند تا بتوانیم مسیر دهی را به صورت پویا انجام دهیم. ماکروهای مسیر در MMB مقادارهای رشته ای هستند که ما از آنها به عنوان پارامترهای فرمان استفاده می کنیم.

مثال:

```
ScriptCommand("Param 1", "<Project Folder>\MyApp.exe")
```

در فرمان بالا از ماکروی <Project Folder> جهت دریافت پوشه ای که برنامه در آن واقع است استفاده شده است. پس در ماکرو مسیر نام کامل فایل را می نویسیم (در این حالت My App.exe) گذشته از ماکروهای MMB تعدادی ماکرو در برخی از Plug-in های MMB هم وجود دارند که جهت فراگیری آنها بایستی به راهنمای آن Plug-In مراجعه کرد.

اما حالا اجازه بدهید تا به بررسی ماکروهای مسیر خود MMB بپردازیم:

<SrcDir>

تقریباً پر استفاده ترین ماکرو مسیراست که پوشه برنامه (که در حال اجراست) را نشان می دهد در حالیکه معمولاً سایر فایل های مورد نیاز پروژه ما در این پوشه است ماکرو <SrcDir> بیشتر مورد استفاده قرار می گیرد.

اگر مسیر کامل برنامه ما به صورت زیر باشد:

C:\Program File\My Project Folder\MyApp.exe

سپس با استفاده از ماکروی مسیر به صورت زیر در می آید:

<SrcDir>\MyApp.exe

همان طور که واضح است این گونه مسیرهی ساده تر و کاراتر است و به جای مسیرهی ثابت می توانیم با استفاده از ماکروها مسیرهی پویا را جهت افزایش کارایی به خدمت بگیریم. اما باید توجه کنیم که هنگامی که MMB ماکروها را ترجمه کرد در نهایت مسیر به صورت ثابت در می آید. به عنوان مثال در مسیر زیر:

که به صورت مقابل ترجمه می شود

Run (“<SrcDir>\MyApp.exe”) → C:\My App\MyApp.exe

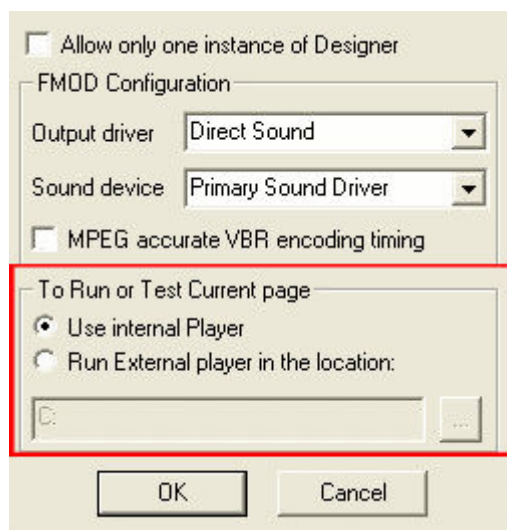
و بر اساس آن برنامه مورد نظر اجرا می شود.

توجه: MMB فایلهای پروژه را به دو گونه مختلف اجرا می کند:

Design Mode: این انتخاب در زمان طراحی پروژه صورت می گیرد و چنانچه با استفاده از گزینه Run برنامه را اجرا نماییم ماکرو (<SrcDir>) پوشه نصب برنامه MMB می شود چرا که اجرا کننده (Player) در این پوشه است. این گونه هنگامی که گزینه Use Internal Player از منوی Designer setting → Tools انتخاب شده باشد فعال است.

RunTime Mode: این گزینه برنامه کامپایل شده را به عنوان یک فایل اجرایی مستقل هنگامی که گزینه Run و <SrcDir> همان پوشه فایل اجرایی می شود.

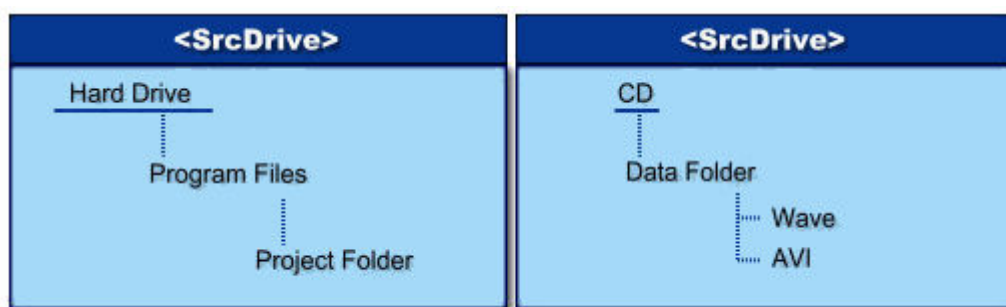
و اما اجرا کننده خارجی (External player) چیست؟



فایل اجرایی (exe) است که در پوشه ای قرار داده شده است و به عنوان یک اجرا کننده مستقل عمل می کند. هنگامیکه شما پس از انتخاب گزینه Run External player in the location در زیر این گزینه مسیر را معین نمایید ماکرو <SrcDir> در این صورت همان مسیری است که شما تعیین کرده اید. و در نهایت باید سایر فایل های پروژه در این پوشه قرار گیرند.

<SrcDrive>

این ماکرو درایو منبع فایل اجرایی برنامه را نشان می دهد.



به عنوان مثال اگر برنامه در درایو C باشد این ماکرو C:\ را باز می گرداند.

<CD>

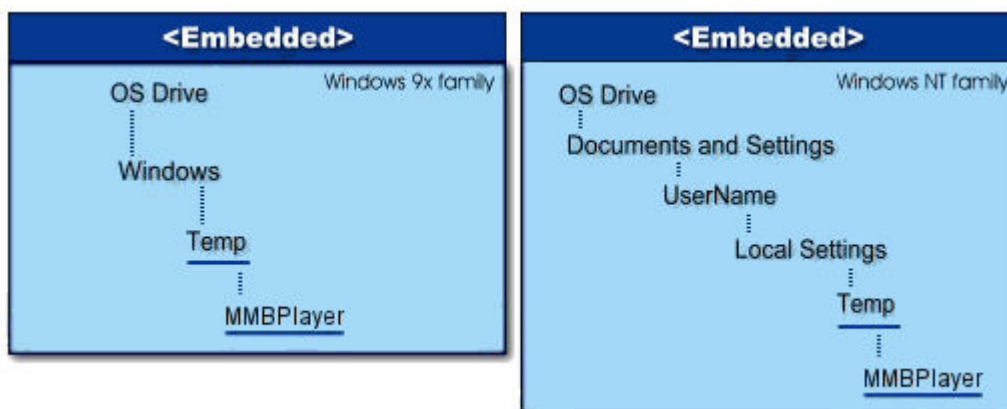
این ماکرو اولین درایو CD را باز می گرداند. به عنوان مثال اگر نام اولین درایو CD، E: باشد، ماکرو <CD>،



E:\ را باز می گرداند. استفاده از این ماکرو در مواقعی کارآمد است که بخواهیم برنامه هایمان را از روی درایو CD یا DVD اجرا کنیم اما باید توجه داشت که این ماکرو تعداد درایوهای CD را حساب نمی کند تنها نام اولین درایو CD موجود را باز می گرداند. برای بدست آوردن نام و برچسب تمام درایوها می توان از PlugIn های ارائه شده استفاده نمود.

<Embedded>

این ماکرو پوشه ای را که برنامه شما جهت فایل های جاسازی شده (Embedded) استفاده می کند باز می گرداند.



به عنوان مثال اگر فایل صوتی زیر را به صورت جاسازی شده داشته باشیم.

Music.Wav

مسیر آن با استفاده از ماکرو به صورت زیر است:

< Embedded >\Music.wav

و هنگامی که شما برنامه را اجرا نمایید این ماکرو به صورت:

(در ویندوزهای XP، ۲۰۰۰ و NT)

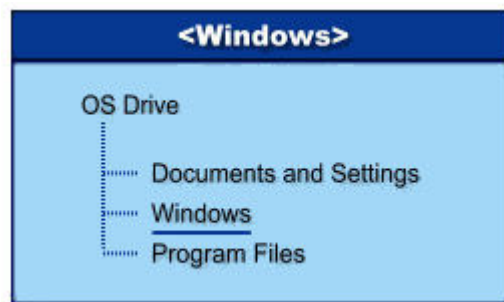
C:\Documant And setting\User Name\Local Setting\Temp\Music.wav

(در ویندوز های 9x)

C:\Windows\Temp\Music.wav

در می آید.

<Windows>



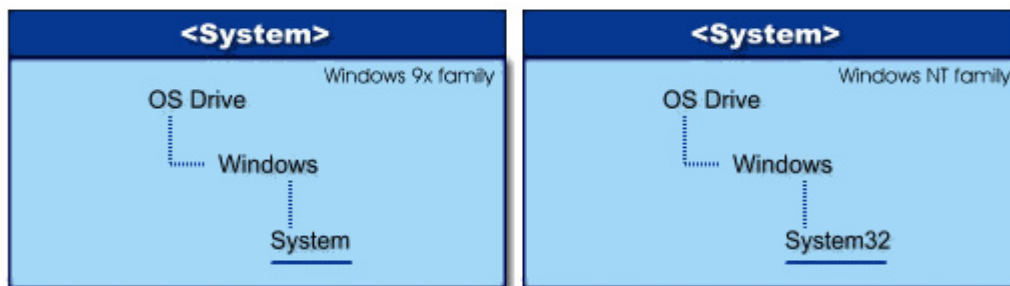
این ماکرو پوشه ویندوز را باز می گرداند. به عنوان مثال چنانچه Windows در درایو C نصب شده باشد و بخواهیم فایل Win.ini را پیدا کنیم به صورت زیر عمل می کنیم:

<Windows>\Win.ini

و در زمان اجرا MMB مسیر بالا را به صورت C:\Windows\Win.ini ترجمه خواهد کرد. معمولاً از این ماکروها جهت فراخوانی فایل های INI و سیستمی استفاده می شود.

<System>

این ماکرو پوشه سیستم ویندوز را باز می گرداند.

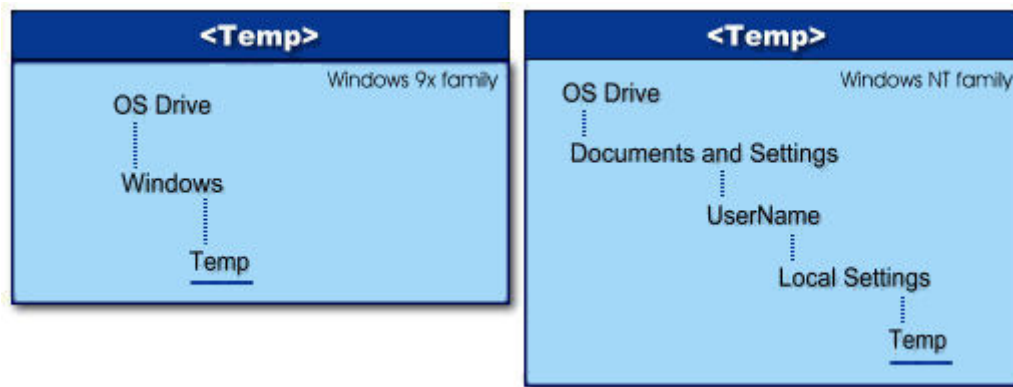


به عنوان مثال مایلم که برنامه dxdiag.exe واقع در پوشه System در درایو C: را اجرا کنیم. مسیر دهی را به صورت زیر انجام می دهیم:

<System>\dxdiag.exe

کـــه در نهایـــت MMB آن را بـــه C:\windows\system\dxdiag.exe در وینـــدوزهای 9x و C:\windows\system32\dxdiag.exe در ویندوزهای NT ترجمه می کند.

<Temp>



این ماکرو پوشه ای از ویندوز را که برای قرار دادن فایل های موقتی است را باز می گرداند. به عنوان مثال مسیر دهی مقابل <Temp>\TempList.txt در نهایت به صورتهای زیر

در ویندوز 9x :

C:\Windows\Temp\TempList.txt

و در ویندوز NT :

C:\Documents and Settings\User\Local Settings\Temp\TempList.txt

در می آیند.

<File>

این ماکرو نام و مسیر فایل منتخب را که توسط کادر انتخاب فایل برگزیده شده را باز می گرداند. چنانچه فایل بازگشایی شده برای مثال C:\test.txt باشد توسط ماکرو <File> می توان همین مسیر را دریافت کرد. لازم به تذکر است در هر بازگشایی فایل محتوی این ماکرو متناسب با آن فایل تغییر می کند. پس برای ذخیره مسیرهها بایستی از چندین متغیر رشته ای یا از آرایه های رشته ای سود جست.

<List>

این ماکرو جهت نگهداری محتوی Internal SongList و شی ListBox می باشد. بنابراین به آسانی می توان محتوی این دو شی را به یکدیگرویا به شی دیگری انتقال داد.

<This>

این ماکرو، یک ماکروی مخصوص فرامین^۱ MCI است که به Device اعلام می کند پنجره MMB پنجره پدر(اصلی) می باشد.

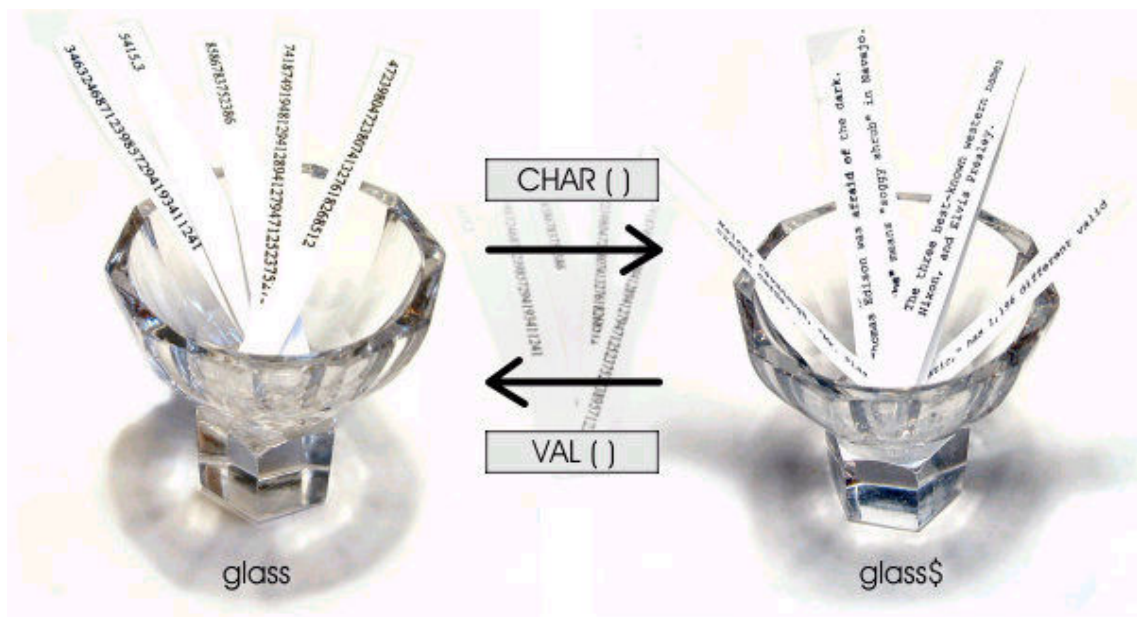
مثال:

```
MCICommand("open <SrcDir>\sample.mpg alias MPEG style child parent <This>")
MCICommand("put MPEG window at 100 50 200 200")
MCICommand("window MPEG state hide")
MCICommand("play MPEG")
```

۱- فرامینی هستند که عملیات های مربوط به صوت و تصویر توسط آنها صورت می گیرد.

« توابع وابسته به متغیرها »

همان طور که ما در زندگی روزمره اقدام به ترجمه یک متن از زبانی به زبان دیگر می کنیم در MMB متغیرها از حالتی به حالت دیگر ترجمه می شوند. استفاده از توابع وابسته به متغیرها سبب تغییر گونه آنها می شود.



جهت رسیدن به این هدف (تبدیل و ترجمه متغیرها) به عمل ترجمه نیازمندیم. در MMB ما عمل ترجمه را بین متغیرهای عددی و رشته ای انجام می دهیم.

تابع CHAR

این تابع متغیر عددی را به متغیر رشته ای تبدیل می نماید جهت این عمل بایستی نام متغیر عددی را بین ۲ پرانتز قرار داد.

CHAR(Numeric Variable Label)

مثال:

```
a=12  
b$=CHAR(a)
```

تابع VAL

برای تبدیل محتوی متغیر رشته ای به عددی از این تابع استفاده می کنیم. این تابع هنگامی مفید است که نیاز باشد بر روی محتوی متغیرهای رشته ای اعمال ریاضی صورت گیرد.

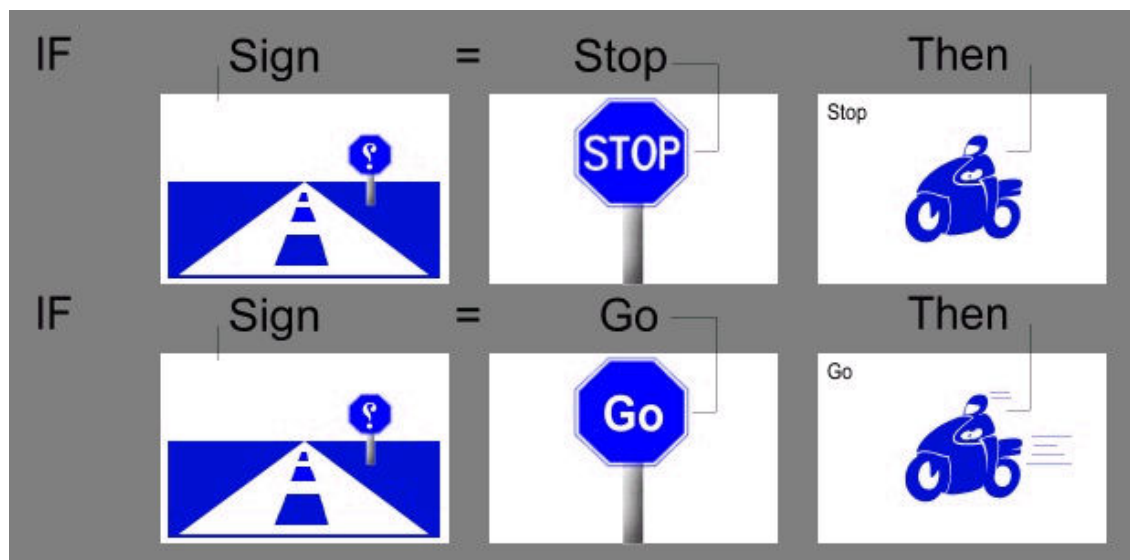
```
VAL(String Variable$)
```

مثال:

```
a$='152'  
b=VAL(a$)
```

عبارت شرطی (If statement)

تصمیم گرفتن مطلبی است که عبارت شرطی If راجع به آن صحبت می کند. توسط علائم راهنمایی و رانندگی شما توقف می کنید، آهسته می رانید و یا دور می زنید. همیشه به هنگام اجرای برنامه شرایطی پیش می آید که باید بر اساس آن شرایط تصمیمی گرفته شود. عمل تصمیم گیری در MMB توسط عبارت شرطی If صورت می گیرد.



بر اساس تابلوها در شکل بالا موتور سوار یا حرکت می کند و یا می ایستد .

عبارت شرطی If شامل قسمت های زیر است:

الف) If

ب) پرانتزی که حاوی شرط است.

پ) علامت مساوی داخل پرانتز.

ت) کلمه Then

ث) فرمان بعدی که بر اساس شرط اجرا می شود.

ج) کلمه End جهت بستن عبارت شرطی

مثال:

```
If (EditBox$='2')Then
  Message("Number is: ", "EditBox$")
End
```

چنانچه در مثال بالا شرط برقرار باشد MMB فرمان Message را اجرا نموده و چنانچه شرط برقرار نباشد MMB از اجرای دستور شرط صرف نظر کرده و به سراغ بقیه قسمتها می رود.

اما همان طور که مشاهده کردید در حالت بالا شرط برقرار بود اما حال اگر شرط برقرار نبود چه کنیم؟ پاسخ ساده است، می توانیم حالت مخالف شرط اصلی را به این قسمت توسط کلمه Else اضافه کنیم. به مثال زیر توجه کنید:

```
If (a$='2') Then
  Message("Number is: ", "a$")
Else
  a$='3'
  Message("Number is: ", "a$")
End
```

در این حالت چنانچه شرط اصلی که همان $a\$ = '2'$ است فراهم نباشد MMB از انجام دستور اولی صرف نظر کرده و قسمت پس از Else را اجرا می نماید. البته براساس گفته های بالا اگر احتیاج به استفاده از Else ندارید هیچ ضرورتی ندارد که آن را بنویسید.

دستورات if چندتایی

با استفاده از چند دستور If می توان تعداد شروط بیشتری تولید و در نتیجه تصمیمات بیشتری را اتخاذ کرد.

مثال:

```
If (a=2) Then
  If (b=3) Then
    Message("You guesed right!", "")
  End
Else
  a=4
  Message("That is wrong, ")
End
```

استفاده از محدوده ی متغیرها در عبارت شرطی If

استفاده تنها از علامت مساوی در همه جا کارآمد نمی باشد چرا که گاهی اوقات لازم است تا اعمال تصمیم بر اساس محدوده ی از مقادیر صورت گیرد. به عنوان مثال می خواهیم شرطی داشته باشیم که هر کس سنش زیر ۲۰ سال است بتواند وارد شود. خوب در این حالت منطقی و به صرفه نیست که اقدام به نوشتن ۲۰ عبارت شرطی کنیم.

بلکه به جای آن می توانیم تنها با یک دستور و با مشخص کردن محدوده، منظورمان را به MMB بفهمانیم. به خاطر این مثال و هزاران مثال مشابه است که حالت‌های دیگری پدید می آید که همگی در جدول زیر آمده است:

عملگر	نام
=	مساوی
<>	مخالف
<	کوچکتر از
>	بزرگتر از
<=	کوچکتر یا مساوی
>=	بزرگتر یا مساوی

به عنوان مثال:

```
If (Age<=20) Then
    Message("You can enter","")
End
```

که چنانچه سن کوچکتر یا مساوی ۲۰ باشد پیغامی مبنی بر اجازه ورود ظاهر می شود.

```
If (Age > 20) Then
    Message("You are not allowed")
End
```

که در این حالت پیغامی مبنی بر عدم اجازه ورود ظاهر می شود.

چند حالت در یک عبارت شرطی If

یکی از قابلیت های کارآمد عبارت شرطی If امکان استفاده از چند وضعیت در یک خط می باشد بر همین اساس ۲ عملگر "&" (و)، "|" (یا) موجود است. شیوه ی استفاده بدین صورت است که ابتدا عبارت If را نوشته پرانتز را باز می کنیم سپس شرط اول را قرار می دهیم با توجه به نیازمان یا عملگر "&" و یا "|" را قرار می دهیم سپس پرانتز را بسته و کلمه then را می آوریم و در ادامه با توجه به شرط می توانیم فرمان مورد نظر را قرار دهیم. حال به مثال زیر توجه نمایید.

```
If (Age=10 | Age=20) Then
  Message("You can go")
End
```

```
If (Age>20 & Job='Student') Then
  Message("You can enter!")
End
```

بر اساس مثال بالا چنانچه سن کاربر یا ۱۰ باشد و یا ۲۰ کاربر مجاز به رفتن می شود. و در مثال دوم چنانچه هر دو شرط برقرار باشد پیغام خطا ظاهر می شود. لازم به ذکر است که در این حالت می توانیم بیش از دو متغیر را مورد آزمایش قرار دهیم.

```
If (Age>20 & Job="Student") Then
  Message("You can enter")
End
```

متغیر در مقابل متغیر

در MMB این امکان فراهم است تا شرطی را داشته باشیم که MMB در آن مستقیماً به امتحان و بررسی متغیرها بپردازد. همانند مثال زیر:

```
If (User1$ = User2$) then
  Message("User are identical !" , "")
End
```

بر اساس شرط بالا چنانچه محتوی ۲ متغیر یکسان باشد پیغام ظاهر می شود. همین حالت را می توانیم برای متغیرهای عددی داشته باشیم.

عبارت های شرطی if با متغیرهای آمیخته شده

همان طور که ممکن است حدس زده باشید این امکان فراهم است که در قسمت شرط از ۲ نوع متغیر استفاده کنیم برای درک بهتر به مثال زیر دقت نمایید.

```
If (User Name $= 'Hadi' & User Height > 170 ) then
  Message("Hadi" is higher than 170 cm", "")
End
```

در کد بالا ابتدا نام شرط بررسی می شود، سپس شرط قد کاربر، چنانچه هر دو شرط موجود باشد پیغام ظاهر می شود.

استفاده از عملگر < >

گاهی اوقات پیش می آید که نمی خواهیم محدوده ای را بررسی نماییم و فقط یک مقدار مورد نظر ماست که تنها بر اساس آن مقدار شرط درست پیدا می کند به همین خاطر می توانیم از علامت "< >" استفاده کنیم به مثال زیر دقت کنید:

```
If (Number < > 1382 ) then
    Message("Entered number is wrong !", "")
End
```

که چنانچه مقدار عددی 1382 نباشد برنامه پیغام خطا را نمایش می دهد.

استفاده از if تو در تو

گاهی اوقات امکان دارد که بخواهیم شرط، بر اساس شرطی دیگر بررسی شود. در این حالت است که از ساختار تودرتو بهره می جوئیم. البته باید توجه داشت که بیشتر اوقات برای پرهیز از این حالت می توان از حالت ترکیبی که قبلاً اشاره شد استفاده کرد.

مثال:

```
If (UserName $= 'Hadi' )then
    If (UserHeight > 170 ) then
        Message ("Hadi is Higher than 170 cm ", "")
    End
End
```

همچنین در این حالت می توان قبل از شرط های تودرتوی بعدی فرمانی را قرار داد. از جمله مواردی که احتیاج به نوشتن دارد کلمه End برای هر دستور If است. باید دقت داشت که همان تعداد که دستور If داریم باید به همان تعداد هم End داشته باشیم. البته محل نوشتن این End ها هم مهم است چرا که نوشتن اشتباه آن ممکن است ۲ شرط را با هم ادغام کند.

```
If (UserName$='Hadi') Then
    Message("Hello Hadi !", "")
    If (UserHeight>170) Then
```



```

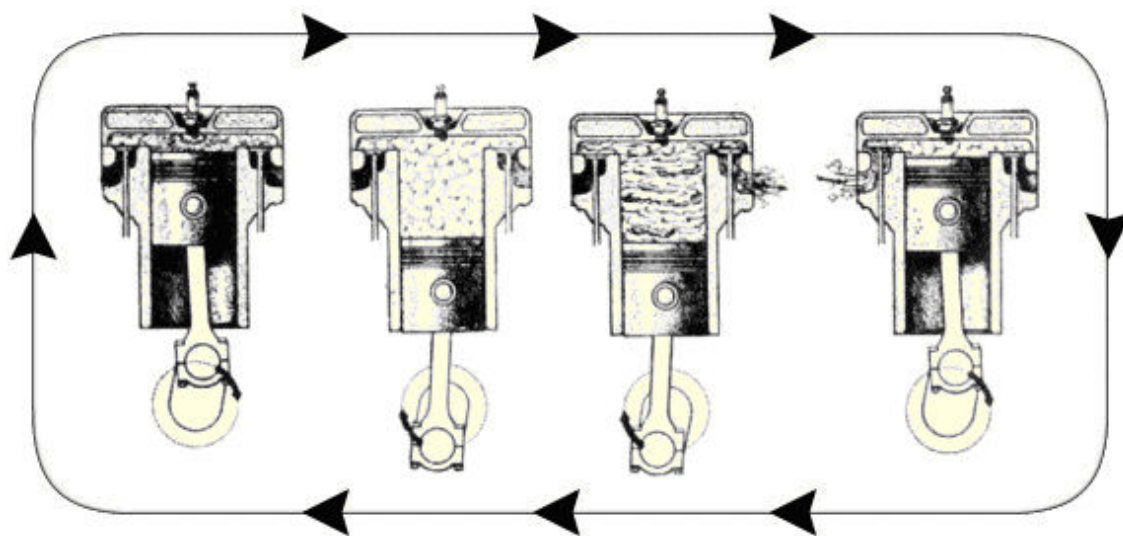
Message("Hadi is higher than 170 cm","")
If (UserWeight>70) Then
    Message("Hadi weighs more than 70 kilos","")
    Return()
End
Message("Hadi weighs less than/or 70 kilos","")
End
End

```

همان طور که در بالا مشاهده می کنید این کد از ۳ دستور If تشکیل شده است که هر کدام با تورفتگی مشخص شده اند.

حلقه های For...next

راه رفتن، چرخش به دور میدان، نفس کشیدن و... این ها همه نشانی از وجود حلقه ها در زندگی روزمره می باشد. که همگی باید توسط ما انجام شود که برخی خودآگاه است و برخی ناخودآگاه به واسطه همین تکرارها بود که ماشین وارد زندگی انسان شد چرا که آنها احساس ندارند و از انجام کارهای تکراری خسته نمی شوند. کمی پیشرفته تر به کامپیوترها می رسیم که روزانه کارهای متعددی را که همگی آنها دارای تکرار هستند انجام می دهند. در MMB همانند سایر محسوط های برنامه نویسی ابزار حلقه برای انجام کارهای تکراری ارائه شده است.



عملیات تکرار در MMB توسط عبارت For...Next انجام می شود که در ادامه به شرح آن می پردازیم.

اما قاعده استفاده از حلقه For...next به صورت زیر است:

الف) ابتدا کلمه For را می نویسیم:

For

ب) سپس متغیر عددی را پس از کلمه For می نویسیم. از این متغیر به عنوان شمارشگر (Counter) استفاده می شود. پس از آن علامت مساوی را پس از متغیر می آوریم:

For Counter =

پ) در ادامه عددی را که می خواهیم حلقه با آن شروع شود می نویسیم، مثلاً 1

ت) پس از آن کلمه to را نوشته و در نهایت محدوده حلقه را با نوشتن عدد مورد نظر مشخص می کنیم.

For Counter = 1 to 10

ث) سپس فرامینی را که می خواهیم مکرراً اجرا شوند می نویسیم.

ج) و در آخر هم کلمه Next را به همراه نام متغیر عددی را که قبلاً به عنوان شمارشگر مشخص کردیم می نویسیم.

Next Counter

حال به مثال زیر توجه کنید:

```
For Counter = 1 to 100  
    Commands  
Next Counter
```

در حلقه مثال بالا فرمان ها به تعداد 100 بار اجرا می شود که می توان این محدوده 100 بار را با تغییر حد بالا و پایین تغییر داد. هنگامی که برای مثال بار 50 ام از 100 بار است مقدار متغیر ما (شمارشگر) همان 50 می باشد که نهایتاً تا 100 ادامه می یابد. از این حالت می توان به صورت پویا در فرامین استفاده کرد. بدین معنی که از مقدار جاری متغیر شمارشگری توان در برنامه و در فرامین استفاده کرد. نباید فراموش کنیم که می توانیم محدوده را طوری تعیین کنیم که سیر نزولی داشته باشد، یعنی بر خلاف مثال بالا که از 1 تا 100 ادامه دارد از 100 تا 1 باشد. پس از اینکه حلقه به اتمام رسید MMB بلافاصله فرمانهای پس از حلقه را اجرا می کند. ما در مثال بالا حد

پایین و بالا را عدد 1 قرار دادیم که الزاماً این طور نیست یعنی می توانستیم حد پایین را از 1 یا از 100 شروع کنیم، تعیین این مقدار بستگی به نیازمان در هنگام کدنویسی دارد. در زیر مثالی برای حالتی که می توانیم از محتوی متغیر شمارشگر استفاده کنیم آمده است:

```
For Runer = 1 to 10
  Var$ = 'You are runner no. ' + CHAR(Runer)
  Message("", "Var$")
Next Runer
```

در مثال بیان شده با استفاده از مقدار جاری متغیر شمارشگر شماره دونه را حساب کردیم.

ابزارهای حلقه

به طور قطع مواقعی پیش می آید که بخواهیم حلقه محدوده مشخصی نداشته باشد یعنی تا بینهایت ادامه یابد، اینجاست که صحبت از حلقه ی بینهایت (Infinity Loop) به میان می آید.

```
For Check = 1 to Infinity
  Check = Check + 1
  DisplayValue("Text ", "Check")
Next Check
```

بلوک فرمان بالا تا هنگامی که برنامه ما در حال اجراست و یا کامپیوتر خاموش نشده باشد به روند خود ادامه می دهد. آیا باید از بابت اجرا نشدن بقیه کدها به هنگام اجرای حلقه ی بینهایت نگران بود؟ جواب خیر است. فرامینی پیش بینی شده اند که اجرای سایر قسمتها را امکان پذیر می سازند.

Refresh () -

این فرمان زمانی را در اختیار سایر قسمتها می گذارد تا توسط برنامه به کار گرفته شوند. به محض اینکه ما در حلقه اسکریپت خود از فرمان (Refresh) استفاده کنیم وارد یک پروسه نیمه موازی شده ایم، این بدین معنا است که هنگامی که حلقه در حال اجرا است و ادامه دارد، می توانیم با شی های دیگر تعامل داشته باشیم و یا اسکریپت های کوچک دیگری را به اجرا درآوریم.

مثال (حلقه معمولی):

```
For n=0 to 10000
  a=n+1
Next n
```

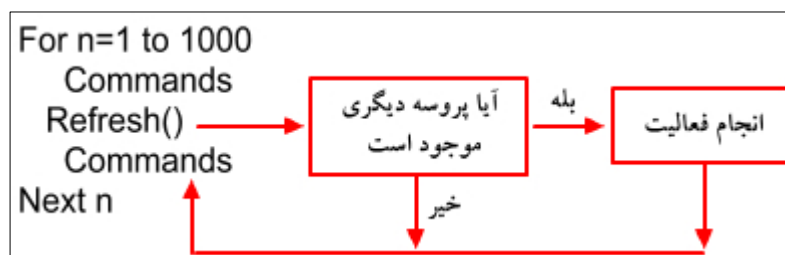
حلقه معمولی باعث توقف MMB در حین انجام عملیات حلقه می شود. لذا شما قادر به کلیک کردن بر روی هیچ دکمه ای در حین عملیات حلقه نخواهید بود.

مثال(حلقه موازی):

```
for n=0 to 10000
  a=n+1
  Refresh ( )
Next n
```

فرمان Refresh () داخل حلقه سبب ایجاد تعامل های موازی در حین عملیات حلقه می شود. این بدان معنا است که هنگامی که حلقه هنوز در حال اجرا است قادر خواهیم بود بر روی هر دکمه ای جهت اجرای اسکریپتی دیگر کلیک نماییم. باید به خاطر داشته باشید که هر حلقه دیگر اعم از موازی یا غیر موازی سبب توقف حلقه اول تا زمان اتمام خودش (حلقه ثانویه) خواهد شد.

به نمودار زیر توجه کنید:



تنها در صورتی که مطمئن هستید که چه کاری توسط حلقه موازی انجام می شود اقدام به استفاده از آن نمایید.

مثالی جهت استفاده از حلقه بینهایت:

با استفاده از حلقه موازی می توانیم حلقه بینهایت تولید کنیم (اما باید به خاطر داشت که راه خروجی را باید تعبیه کنیم). مثال زیر سبب چسبیدن دایره ای به نشانه گر موس تا زمانی که هیچ شی دیگری سبب مقدار دهی متغیر stop به 1 شود، می شود.

```
Stop = 0
For n=0 to Infinity
MoveObject(" Circle", "MouseX( ), MouseY( )")
If (Stop = 1) then
  Return ( )
End
Refresh ( )
Next n
```

Pause ("Time") -

این فرمان سبب ایجاد وقفه در بلوک کد برنامه بر اساس مقدار ورودی می شود واحد این زمان وقفه میلی ثانیه (۰/۰۰۱ ثانیه) است. در زیر مثالی برای Refresh() آمده است. این مثال جهت مقایسه می باشد:

```
For Check = 1 to Infinity
  Check = Check +1
  DisplayValue("Text ", "Check" )
  Refresh( )
Next Check
```

هنگامی که برنامه در حلقه به Refresh() می رسد به قسمتهای دیگر قبل از شروع مجدد حلقه اجازه فعالیت می دهد. حال به مثالی برای Puase (" ") توجه کنید:

```
For Check = 1 to Infinity
  Check = Check +1
  DisplayValue("Text ", "Check" )
  Puase("500")
Next Check
```

در این حالت برنامه به هنگام رسیدن به فرمان ۵۰۰ میلی ثانیه یا نیم ثانیه توقف می کند و سپس روند تکرار حلقه از سر گرفته می شود.

Return (-)-

یک ابزار کارآمد دیگر (Return) است و هنگامی مورد استفاده قرار می گیرد که در حلقه بینهایت شرطی به وقوع بپیوندد، در اینجاست که حلقه باید متوقف شود. برای فهم بیشتر این مسئله به مثال زیر توجه کنید:

```
For Check = 1 to infinity
  Check = Check +1
  DisplayValue ("Text ", "Check" )
  If (User Input $ = 'Go' ) Then
    Return( )
  End
  Puase(" 500")
Next Check
```

مثال بالا آنقدر هوشمند می باشد که به محض برقراری شرایط حلقه را متوقف کند. چنانچه بخواهیم فرمان یا فرمانهایی پیش از محقق شدن شرایط اجرا شود می توانیم در بلوک شرط و قبل از Return فرامین را قرار دهیم. توجه نمایید که MMB به محض رسیدن به Return اجرای اسکریپت را متوقف کرده و از اجرای باقی فرامین ژس از Return() صرف نظر می کند و به ابتدای کد باز می گردد.

Break()-

ممکن است حالت پیش بیاید که تنها مایل باشیم اجرای حلقه متوقف شود و سایر فرامین ژس از حلقه متوالیا اجرا شوند. در این حالت از فرمان Break() استفاده می کنیم. MMB به محض رسیدن به Break() اجرای حلقه را متوقف می کند و بلافاصله اولین خط پس از کلمه کلیدی Next را اجرا می کند.

مثال:

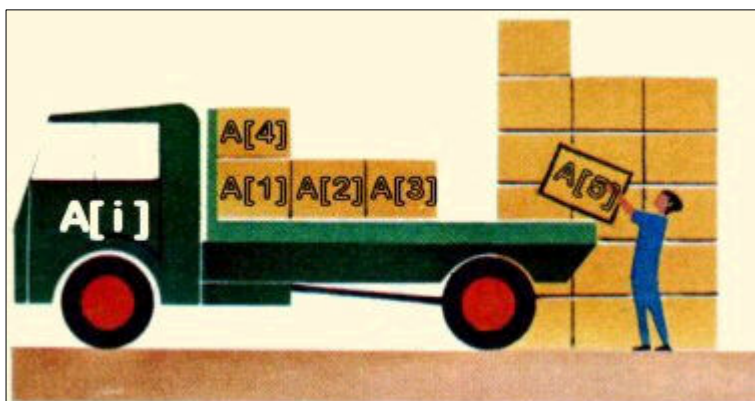
```
maxloop=5
For i=1 To maxloop
  Message("loop 1","i")
  For n=1 To 10
    If (n=3) Then
      Break()
    End
  Message("loop 2","n")
Next n
Next i
```

قطعه کد بالا عملاً بی استفاده است. اما در هنگام اجرا ۵ پیغام از حلقه I اجرا می شود و ۲ پیغام از حلقه

. n

آرایه Array

حتما تا به حال به قفسه بندی ها توجه کرده اید، هدف از ایجاد قفسه ها نظم دادن به قرار گیری اشیا همسان می باشد. برای مثال در یک فروشگاه تمام صابونها در یک قفسه و در کنار هم هستند، یا همانند شکل بسته های چیده شده در یک کامیون.



همان طور که واضح است انتقال اجناس همسان توسط یک کامیون منطقی تر از آن است که اجناس را تک تک به مقصد انتقال دهیم. در MMB نیز می توانیم از آرایه ها استفاده کنیم. البته آرایه های تک بعدی (1-Dimensional) دلیل این نوع نا مگذاری اینست که محتوی این آرایه در یک خط مرتب می شوند. آرایه ها در MMB گروهی از متغیرها تحت یک نام منفرد هستند. در MMB از آرایه ها همانند متغیرها استفاده می شود. جدول زیر مقایسه ای بین آرایه ها و متغیرها را نشان می دهد.

Variable	Array
A\$	A\$[]
Name\$	Name\$[n]
Clicks\$	Clicks[n]
User_address\$	User_address\$[n]
User_Counter	User_counter[n]

در جدول بالا ستون سمت چپ متغیرها هستند و ستون سمت راست آرایه ها هستند. همان طور که مشاهده می کنید آرایه ها نوع توسعه یافته متغیرها هستند. آرایه ی [User_Counter] را در نظر بگیرید می خواهیم ببینیم این آرایه از چه قسمتهایی تشکیل شده است. User_Counter برچسب آرایه است، [] براکت ها جهت مشخص کردن شمارنده و n به عنوان شمارنده (Index) محسوب می شود.

چنانچه قسمت متغیرها را مطالعه کرده باشید متوجه شده اید که متغیرها تقریبا در همه جا می باشند. یکی از نقش های متغیرها اینست که مؤلفه های آرایه ها را مشخص نمایند. البته در این میان از متغیرهای عددی استفاده می شود.

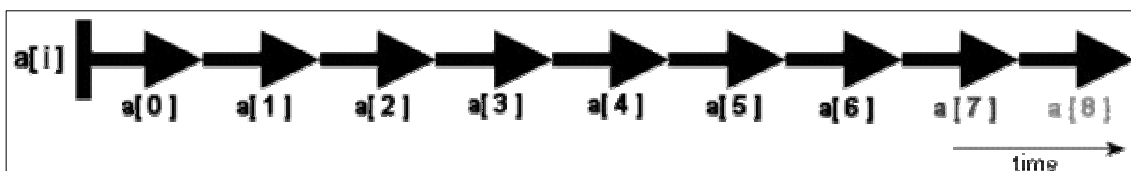
مثالی از آرایه هایی که شمارشگر آنها عددهای معمولی هستند:

```
a[1]
twain$[72]
my_finger[534]
tool$[102]
```

مثالی از آرایه هایی که شمارشگر آنها متغیرهای عددی هستند:

```
a[i]
twain$[chapter_no]
my_finger[102]
```

از آرایه ها معمولا بیشتر در حلقه های For...next استفاده می شود.



هنگامیکه مؤلفه های آرایه ها پر می شوند، جدیدترین مؤلفه بالاترین مقدار را به خود اختصاص می دهد.

همچنین برای مشخص کردن مؤلفه ها در آرایه ها می توانیم از اعمال ریاضی استفاده می کنیم:

```
a[I+1]
twain$[chapterNo_4]
```

اما چطور می توانیم آرایه ها را پر کنیم؟ این امر بستگی به نوع آرایه ها دارد.

«آرایه های عددی»

همانند متغیرهای عددی، آرایه های عددی هم برای نگهداری اعداد استفاده می شوند. از آرایه های عددی نیز برای اعمال ریاضی استفاده می کنیم. اجازه دهید تا آرایه ای را مقدار دهی کنیم:

```
a[I]=c  
myfinger[I]=15
```

در مثال بالا در آرایه ی [a] مقدار c به مؤلفه ۱ آن نسبت داده شده است و در قسمت دوم مقدار ۱۵ به آرایه نسبت داده شده است.

خواندن محتوی آرایه ها نیز به صورت زیر است:

```
c=a[1]  
finger=myfinger[I]
```

در مثال بالا نیز مؤلفه شماره ۱ آرایه [a] به متغیر c اختصاص داده می شود. در قسمت دوم مقدار مؤلفه آرایه به متغیر منتقل می شود.

«آرایه های رشته»

همانند متغیرهای رشته ای، آرایه های رشته ای نیز برای ذخیره کلمات، عبارات متنی و کلا رشته ها مورد استفاده قرار می گیرند. آرایه های رشته ای همانند زیر هستند:

```
a[I]
```

هبرچسب آرایه است، \$ به MMB می گوید که نوع آرایه رشته ای است، [] براکت ها و مؤلفه ها که هم می توانند عدد باشند و هم متغیر عددی. مقدار دهی آرایه های رشته ای همانند زیر است:

```
a[I]=c$  
finger_name[I]=name$
```

در مثال بالا مؤلفه اول آرایه [a] مقدار متغیر c\$ را می گیرد و در قسمت دوم آرایه [finger_name] محتوی متغیر name\$ را دریافت می کند.

مثالی از استفاده آرایه ها در حلقه ها:

```

For I=1 to 50
  a[I]=I
Next I

```

مثال بالا ۵۰ مؤلفه برای آرایه ایجاد می کند که هر کدام مقدار متناظرش را دریافت می کند، اما به مثال زیر جهت خواندن محتوی آرایه ها توجه کنید:

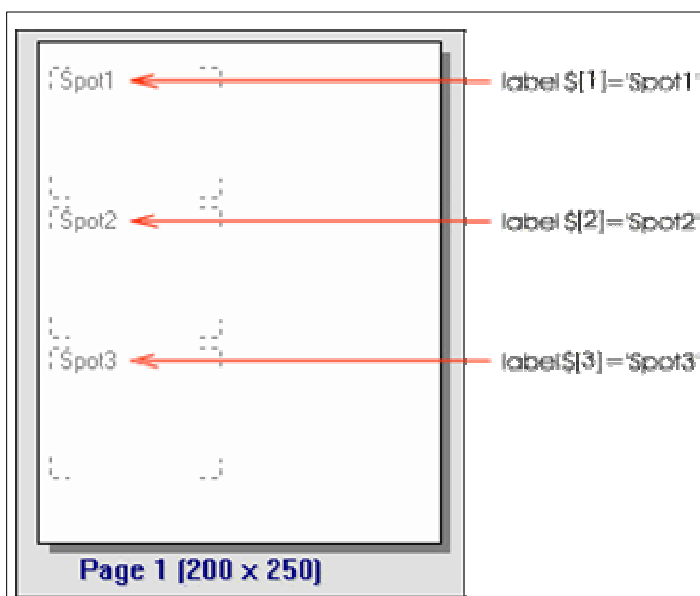
```

For I=1 to 50
  a=a[I]
  Message("Array item content is :","a")
Next I

```

آرایه های رشته ای به عنوان برچسب اشیا

انعطاف پذیری MMB به شما اجازه می دهد تا از محتوی آرایه های رشته ای به عنوان منبعی برای برچسب اشیا استفاده کنید.



این ویژگی زمانی مفید است که اشیا از اسامی متحدالشکل برخوردار باشند.

مثال:

```

For I=1 to 20
  text$='Hello No. '+CHAR(I)
  Label$I=text$ + CHAR(I)
  LoadText("Label$[I]",text$)
Next I

```

- GetArrayItem(Array\$,SeparatorChar\$,Position)

این تابع بر اساس موقعیت داده شده مؤلفه ای از آرایه را باز می گرداند. SeparatorChar\$ کاراکتری را مشخص می کند که آن کاراکتر مرز بین مؤلفه ها را مشخص می کند به این کاراکتر جدا ساز نیز اطلاق می شود. کاراکتر جدا ساز هر کاراکتری می تواند باشد، اما توصیه می شود که از کاراکترهای (#)،(*) و (|) استفاده نمایید. پارامتر Position برای تعیین موقعیت مؤلفه ی مورد نظر می باشد.

مثال:

```
Items$ = 'item1#item2#item3#item4#item5#'
For i = 1 To 5
    ArrayItem$ = GetArrayItem(Items$,#,i)
    Message("Obtained array item: ", "ArrayItem$")
Next i
```

- GetArrayNum(Array\$,SepratorChar\$)

این تابع تعداد مؤلفه های درون آرایه را باز می گرداند. SepratorChar\$ کاراکتر جدا ساز را مشخص می کند.

مثال:

```
Items$ = 'item1#item2#item3#item4#item5#'
NumOfItems=GetArrayItem(Items$,#)
For i = 1 To NumOfItems
    ArrayItem$ = GetArrayItem(Items$,#,i)
    Message("Obtained array item: ", "ArrayItem$")
Next i
```

جادوگر اسکریپت (MMB Script Wizard)

همان طور که پیش تر گفته شد این قسمت برای ساده کردن و در عین حال سریع کردن برنامه نویسی می باشد. در این جادوگر لیستی از فرامین MMB موجود می باشد که با انتخاب هر فرمان توضیحی مختصر به همراه محل ورود پارامترهای فرمان ظاهر می شود، البته برخی از فرامین که هیچ پارامتری را دریافت نمی کنند مسلماً کادر ورود پارامتر برای آنها ظاهر نخواهد شد. بهتر دیدیم که شرح فرامین MMB را براساس این جادوگر انجام دهیم.

« توابع اصلی »

قسمت بنیادی اسکریپت نویسی همان فرامین می باشند. برخی از این فرامین تابع نیز هستند چرا که مقدار خروجی تولید می کنند. صورت کلی فرامین و توابع به صورت زیر است:

Function("Parameter 1" , " Parameter 2")

در این صورت به جای کلمه Function نام فرمان مورد نظر قرار می گیرد. Parameter 1 معمولاً محل نوشتن نام شی می باشد اما می تواند چیزهایی دیگر از قبیل مسیر یک فایل، زمان و... نیز باشد. Parameter 2 نیز محل قرار دادن سایر پارامترهای مورد نیاز هر دستور است. ممکن است گاهی اوقات قسمت Parameter 2 چند قسمت شود که هر قسمت با علامت کاما از هم جدا می شود. به طور کلی پارامترهای هر دستور بر اساس خود دستور معین می شود. در ادامه می خواهیم به شرح و بررسی فرامین و توابع Wizard (جادوگر) MMB بپردازیم.

فرمان « Hide »

شی یا گروهی از اشیاء را پنهان می کند. در قسمت پارامتر این فرمان نام شی را می نویسیم:

Hide("Object Name")

فرمان « Show »

این فرمان شی یا گروهی از اشیاء را نمایان می سازد. اگر شی یک GIF متحرک باشد به هنگام نمایان شدن شروعش از فریم نخست می باشد. در قسمت پارامتر این فرمان بایستی نام شی مورد نظر را بنویسیم.

Show("Object Name")

فرمان « Invert »

برای نمایان ساختن و یا پنهان کردن شی‌ای به کار می‌رود. چنانچه شی قابل دیدن باشد آنرا پنهان و چنانچه پنهان باشد آن را آشکار می‌سازد. نام شی مورد نظر در قسمت پارامتر این فرمان می‌آید.

```
Invert("Object Name")
```

مثال:

```
Hide("Rectangle 1")  
a$='Text'  
Show("a$ ")  
Invert("Circle")
```

نکته: در MMB این قابلیت فراهم است که خارج از صفحه حاوی شی و یا از محلی دیگر همانند Master page حالت نمایش اشیاء را تغییر دهیم. برای این کار بایستی همانند مثال زیر عمل کنیم:

```
Hide("Page 10::Circle")
```

مثال بالا شی Circle را در صفحه ۱۰ پنهان می‌کند. توجه کنید که ابتدا محل شی (صفحه) را می‌نویسیم (همانند (Page 10

سپس :: را می‌نویسیم و نهایتاً نام شی مورد نظر.

مثال:

```
Hide("Master Page :: Rect1")  
Hide("Page 10::Circle")  
Show("Page 1:: Text")
```

فرمان « RunMBD »

این فرمان باعث اجرای فایل دیگری با پسوند M BD می‌شود. همچنین می‌توانید تعیین کنید که این فایل در صفحه جاری اجرا شود و یا در صفحه‌ای دیگر. استفاده از این دستور برای تولید محصولات با حجم بالا بسیار کارا و موثر می‌باشد، با این عمل شما بدنه محصول خود را به چند قسمت مجزا تقسیم می‌کنید که به جای حرکت بین صفحات بین این فایل‌ها حرکت می‌کنید. با استفاده از این فرمان دیگر نگرانی از بابت مشکل با حافظه وجود نخواهد داشت. چرا که به محض بارگذاری فایل جدید، فایل قبلی از حافظه پاک می‌شود. یکی از استفاده‌های این

فرمان در جایی است که مایل هستیم هر صفحه خصوصیت‌های منحصر به فردی داشته باشد. البته بایستی در استفاده از این فرمان دقت داشت چرا که ممکن است مسیرها به درستی ذکر نشود.

```
RunMBD("Path to MBD" , "Page Label")
```

پارامتر اول مسیر فایل MBD را مشخص می‌کند و پارامتر دوم نام صفحه را. می‌توانیم برای اجرای این فایل MBD صفحه‌ای مجازی در نظر بگیریم.

فرمان « Run »

از این فرمان برای اجرای فایل‌های اجرایی خارجی استفاده می‌شود. برای تغییر مکان پنجره فایل اجرایی، به این معنا که پنجره فایل مورد نظر روی پنجره‌های فایل‌های دیگر اجرا شود و یا خیر (بیننده این پنجره را بالای بقیه ببیند)، ۲ پارامتر موجود است که این کار را انجام دهد:

2- Top Most 1- Top

حال بینیم که تفاوت این ۲ پارامتر در چیست؟

Top-

این پارامتر این اطمینان را می‌دهد که پنجره فایل ما همیشه بر روی سایر پنجره‌ها که از قبل باز شده‌اند اجرا می‌شود. اما پس از کلیک کردن روی پنجره‌ای دیگر این پنجره به زیر سایر پنجره‌ها می‌رود.

Top Most-

این پارامتر باعث حفظ وضعیت قرارگیری پنجره از نظر بالاتر بودن نسبت به سایر پنجره‌ها می‌شود. (تا حد امکان بالاتر از سایر پنجره‌ها)

مثال:

برای اجرای برنامه Notepad ویندوز که بالاتر از بقیه باشد و در ضمن سند مورد نظر ما در بالاترین سطح قرار گیرد، از کد زیر استفاده می‌کنیم:

```
Run("C:\Windows\notepad.exe" , "Top Most C:\My doc.txt")  
Run(" Path to executable file" , " Parameters")
```

در این فرمان پارامتر اول مسیر فایل اجرایی را مشخص می‌کند و پارامتر دوم نحوه عملکرد پنجره و خود برنامه را مشخص می‌کند.

علاوه بر آن چنانچه از پارامترهای `Top \ Top Most` استفاده شود، MMB تا زمان اجرا و آغاز برنامه خارجی صبر خواهد کرد. این وقفه گاهی اوقات می‌تواند خطرناک و مشکل آفرین باشد. همانند مواقعی که برنامه کاربردی اتصال یافته اجرا نمی‌شود و همین عدم اجرای صحیح باعث گیرکردن و از کار افتادن اجرا کننده MMB می‌شود. برای رفع این مشکل راه حلی هست و آن اینکه می‌توان با اختصاص دادن زمانی معین در اجرای برنامه اصلی (که توسط شما تولید شده است) وقفه ای ایجاد نمود و پس از سپری شدن زمان مقرر شده فرمان اجرای آن فایل را داد. با استفاده از این ترفند نگرانی از بابت عدم اجرای فایل اجرایی و از کار افتادن اجرا کننده MMB از بین می‌رود. ضمناً فراموش نشود که واحد این زمان بر حسب میلی ثانیه است.

مثال:

```
Run ("C:\Windows\notepad.exe", "Top Most (2000) C:\Mydoc.txt")
```

اما باید توجه کنیم که در استفاده از فرمان Run محدودیت‌هایی وجود دارد، محدودیت‌هایی از قبیل:

- `Top \ Top Most` از پنجره‌های فرزند حمایت نمی‌کنند و تنها برای اولین پنجره باز شده از برنامه به کار می‌روند.

- برخی از برنامه‌های غیر استاندارد که فاقد PID (Process ID) هستند ممکن است که به درستی عمل نمایند.

- چنانچه شما برنامه منبع خود را (که قبلاً در MMB توسط خودتان تولید شده و فایل خارجی را فراخوانی می‌کند) ببندید به هیچ وجه برنامه فراخوانی شده به طور خودکار بسته نخواهد شد.

- چنانچه می‌خواهید برنامه‌های سیستمی از قبیل `Regedit.exe, Notepad.exe` و ... را به همراه پارامترهای `Top \ Top Most` اجرا نماید باید مسیر کامل این فایل‌ها را وارد نمایید. بر خلاف حالتی که این فراخوانی فاقد ۲ پارامتر `Top \ Top Most` می‌باشد که نیازی به نوشتن مسیر کامل نیست و تنها در صورت نوشتن نام فایل اجرایی، برنامه‌های سیستمی اجرا می‌شوند.

فرامین « [PrevPage](#) / [NextPage](#) / [LastPage](#) / [FirstPage](#) »

این فرامین به ترتیب از راست به چپ برای رفتن به اولین صفحه، رفتن به آخرین صفحه، رفتن به صفحه بعد و رفتن به صفحه قبل می‌باشند. این فرامین هیچ پارامتری را دریافت نمی‌کنند و در موقع استفاده از آنها بایستی ۲ پرانتز را به صورت () در انتهای آنها بیاوریم.

مثال:

First Page ()
Next Page ()

فرمان « Page »

توسط این فرمان می‌توان به صفحه‌ی مورد نظر انتقال یافت. برای این کار تنها کافیست تا نام صفحه را در قسمت پارامتر این فرمان بنویسیم. شایان ذکر است که در استفاده از فرمان Page می‌توانید با نوشتن LAST PAGE در قسمت پارامتر این فرمان به آخرین پنجره‌ای که قبلاً باز شده باز گردید.

Page("Page Label")

مثال:

Page ("Page 5")

فرمان « Exit »

سبب خروج از برنامه می‌شود.

Exit ()

فرمان « Minimize »

این فرمان باعث انتقال برنامه به نوار وظیفه می‌شود.

Minimize ()

فرمان « Exit Timer »

این فرمان باعث خروج از برنامه پس از گذشت چند میلی ثانیه می‌شود. زمان دلخواه را می‌توان در قسمت پارامتر این فرمان وارد کرد. توجه نمایید که استفاده از فرمان Page Timer می‌تواند باعث لغو این فرمان شود. همچنین کلیک نمودن بر روی هر شی‌ی فعالی باعث لغو این فرمان حین به کارگیری این فرمان خواهد شد. پس می‌توانید با

گذاشتن دکمه‌های مناسب در برنامه از کلیک‌های ناخواسته بر جاهای دیگر جلوگیری نماید. این قابلیت برای مواقعی که می‌خواهیم وقایعی صورت گیرد و سپس از برنامه خارج شویم مناسب است.

ExitTimer("MiliSec")

مثال:

ExitTimer("1000")

فرمان « Page Timer »

باعث انتقال به صفحه‌ای دیگر پس از گذشت مقدار معینی زمان بر حسب میلی ثانیه می‌شود. چنانچه صفحه‌ی خاصی را بیان نمایید برنامه به صفحه‌ی بعدی می‌رود.

PageTimer("MiliSec", "Page")

در ضمن می‌توانیم به جای پارامتر Page از پارامترهای زیر نیز استفاده کنیم:

- THIS_SCRIPT: باعث اجرای اسکریپت در همان صفحه می‌شود.

- THIS_PAGE: باعث اجرای مجدد همان صفحه جاری می‌شود.

- IF_IDLE: به صفحه دیگر منتقل می‌شود چنانچه کاربر عمل خاصی را جز صبر کردن انجام ندهد.

نکته: کلیک بر روی هر شی فعال باعث لغو این فرمان می‌شود. استفاده از فرمان PageTimer یا ExitTimer دیگر باعث لغو PageTimer جاری می‌شود.

مثال:

PageTimer("10000", " Page 5")

فرمان « DisplayValue »

باعث نمایان شدن جزء صحیح متغیری در شی متنی می‌شود این فرمان برای شمارشگرها مناسب می‌باشد.

DisplayValue("Text Object", "Variable")

پارامتر اول شی متنی را که نمایش دهنده مقدار است تعیین می‌کند و پارامتر دوم متغیر مورد نظر را معین می‌کند.

مثال:

```
DisplayValue("Text 1", "Counter")
```

فرمان « Message »

این فرمان برای نمایان ساختن کادرپیغام (Message Box) می‌باشد. البته باید یادآور شد که این کادر پیغام ارائه شده توسط MMB همانند سایر کادرهای پیغام از انعطاف پذیری بالایی برخوردار نیست.

```
Message("Any String", "Variable")
```

پارامتر اول محتوی پیغام را مشخص می‌کند و پارامتر دوم هم برای نشان دادن محتوی متغیر چه عددی و چه رشته‌ای است. چنانچه بخواهیم پیغام در چند خط نمایان شود بایستی از عبارت (\n) در بین محتوی پیغام استفاده کنیم.

مثال:

```
age=20  
Message ("Your age is : " , "age")
```

فرمان « Return »

در قسمت‌های قبل بررسی شد.

فرمان‌های «AGif Reset, AGif Stop, AGif Play»

این فرامین به ترتیب از راست به چپ باعث اجرای GIF شده، باعث توقف GIF و بازگشت به اولین فریم GIF می‌شود و اجرا را از سر می‌گیرد. بایستی توجه کرد که اجرای یک فایل به مشخصات آن هم بستگی دارد.

```
AGIFPlay("Ani GIF Object")
```

```
AGIFStop("Ani GIF Object")
```

```
AGIFReset("Ani GIF Object")
```

در این فرامین در قسمت پارامتر بایستی نام شی GIF را وارد نماییم.

مثال:

```
AGIFPlay("Start")
```

فرمان « RunScript »

از این دستور برای اجرای شی اسکریپتی استفاده می شود. همچنین از این فرمان در پاره‌ای از اوقات می‌توان برای تولید حلقه استفاده نمود.

```
RunScript("Script object")
```

در قسمت پارامتر باید نام شی اسکریپت را قرار دهیم.

مثال:

```
RunScript("Script 1")
```

فرمان « ScriptTimer »

این فرمان شی اسکریپتی را پس از گذشت چند میلی ثانیه اجرا می‌کند.

```
ScriptTimer("Script Object", "MiliSec")
```

پارامتر اول نام شی اسکریپتی را مشخص می‌کند و پارامتر دوم مقدار زمان بر حسب میلی ثانیه.

فرمان « OpenFile »

این فرمان یک کادر بازگشایی فایل را نمایان می‌سازد و سپس می‌توانیم توسط ثابت‌های CBK-OpenDir, CBK-OpenFile, <File> و متغیر OpenFile\$ مقدارهای بازگشتی را دریافت کنیم.

```
OpenFile("Filter ", "Default Extension")
```

در قسمت پارامتر این فرمان بایستی فیلتر را معین کنیم که تنها کادر بازگشایی چه نوع فایل و یا فایل‌هایی را شناسایی کند. در زیر مثالی برای فیلتر آمده است:

```
MPEG Files (*.mpg)|*.mpg|All Files*|*.*||
```

این فیلتر در نهایت سبب ایجاد ۲ انتخاب برای کادر بازگشایی می‌شود توجه نمایید که کاراکتر " | " برای جدا کردن قسمت‌ها (پسوندها) و کاراکترهای " || " برای پایان عبارت به کار می‌روند. پارامتر دوم این دستور پسوند پیش فرض را تعیین می‌کند.

مثال:

```
OpenFile(" Text File (*.txt )| .txt| All File |*.*| | , *.txt")
Message("The file you selected is: " , "Open file$" )
LoadText("paragraph" , " < File > ")
```

فرمان «SaveFile»

فرمان مذکور کادر محاوره ای ذخیره فایل را نمایان می‌سازد.

SaveFile("Filter","Default Extension") -

اولین پارامتر این فرمان فیلتر را مشخص می‌کنید و دومین پارامتر برای تعیین پسوند پیش فرض می‌باشد. در زیر مثالی برای فیلتر آمده است:

Text File (*.txt)| .txt| All File |*.*| | , *.txt

نحوه مشخص کردن فیلتر همانند دستور OpenFile می‌باشد. در این دستور نیز ۳ ثابت CBK_OpenDir و CBK_OpenFile.<File> مقدار دهی می‌شوند.

مثال:

```
SaveFile("TXT Files (*.txt)|*.txt|All Files (*.*)|*.*|","*.txt")
Message("Selected file in OpenFile$ variable: ","OpenFile$")
file$=<File>
Message("Selected file in <File> constant: ","file$")
file$=CBK_OpenFile
Message("Selected file name: ","file$")
folder$=CBK_OpenDir
Message("Folder path of selected file: ","folder$")
```

مثال:

در این مثال ابتدا کد منبع یک فایل HTML به درون یک ListBox بار گذاری می شود و سپس توسط فرمان SaveFile می توان فایل HTML را در جایی دیگر ذخیره کرد.

```
OpenFile("HTML Files (*.htm;*.html)|*.htm;*.html|All Files
|*.*||", "*.htm;*.html")
If (OpenFile$<>'') Then
ListBoxDeleteItem("ListBox", "-1")
ListBoxAddItem("ListBox", "OpenFile$")
OpenFile$=''
End
Filename$='CBK_OpenFile
SaveFile("HTML Files (*.htm;*.html)|*.htm;*.html|All Files|*.*||", "filename$")
If (OpenFile$>'') Then
SongListSave("ListBox", "OpenFile$")
OpenFile$=''
End
```

فرمان « FileString »

این فرمان به جستجوی زیر رشته های مشخص شده در مقداری که توسط < File > دریافت می شود می پردازد. به عنوان مثال چنانچه < File > دارای زیر رشته wav باشد مقدار متغیر بازگشتی این فرمان l و در غیر این صورت صفر می شود.

FileString ("SubString ", "Variable")

پارامتر اول زیر رشته را مشخص می کند و پارامتر دوم متغیری را برای دریافت وضعیت معین می کند. این متغیر بایستی از نوع عددی باشد.

مثال:

```
FileString(".Swf ", " Status")
```

فرمان «InstallFont»

از این فرمان هنگامی استفاده می شود که نسخه ای از فونت به کار رفته در پروژه شما در سیستم کاربر نهایی موجود نباشد. توسط این فرمان می توانید موقتاً این فونت را نصب نمایید. توجه نمایید که به هنگام خروج از برنامه فونت هم پاک خواهد شد. معمولاً این فرمان در هنگام شروع صفحات مورد استفاده قرار می گیرد.

InstallFont ("Path to font")

در قسمت پارامتر مسیر فونتی که مایلیم نصب شود را باید مشخص کنیم.

مثال:

```
InstallFont("< Scr Dir>\funny.ttf")
```

ضمناً شما می‌توانید از فونت‌های الحاق شده (Embedded) استفاده نمایید.

« Clipboard » فرمان

این فرمان جهت ارسال و یا دریافت محتوی متغیر از Clip Board ویندوز می‌باشد.

```
Clipboard ("SEND/GET ", "String Variable" )
```

چنانچه بخواهیم متغیر را ارسال کنیم از کلمه SEND (فرستادن) در پارامتر استفاده می‌کنیم و اگر بخواهیم متغیر را دریافت کنیم از کلمه GET (گرفتن) در پارامتر اول استفاده می‌کنیم. پارامتر دوم متغیری را مشخص می‌کند اطلاعات در آن قرار می‌گیرد و یا از آن گرفته و به Clip Board ارسال می‌شود. لازم به ذکر است که این فرمان تنها بر اساس متن و عدد کار می‌کند و قادر به ارسال شی‌هایی از قبیل عکس به کلیپ برد نمی‌باشد.

مثال:

```
a$ = 'Text to Clip board'  
Clipboard ("SEND ", "a$" )
```

« FileExist » فرمان

این فرمان به امتحان اینکه فایل مشخص شده موجود است یا خیر می‌پردازد. در صورت وجود مقدار ۱ را به متغیر اختصاص می‌دهد و در صورت عدم وجود مقدار صفر را به متغیر اختصاص می‌دهد.

```
FileExist("Path to file", "Variable" )
```

پارامتر اول مسیر فایل مورد نظر را تعیین می‌کند. و پارامتر دوم متغیر را مشخص می‌کند.

مثال:

```
FileExist("C:\My file.exe", "Isexist" )
```

فرامین «WaveStop, WavePlay»

این فرامین به ترتیب از راست برای اجرا و یا توقف اجرای فایل صوتی خارجی و یا داخلی با فرمت Wave استفاده می‌شود. در قسمت Command این فرمان شما می‌توانید با قرار دادن کلمه Loop باعث اجرای مکرر فایل صوتی شوید. لازم به تذکر است که بهترین نتیجه به هنگام استفاده از Loop موقعی بدست می‌آید که Direct Sound موجود باشد. نکته بسیار مهم دیگر اینست که چنانچه فایل صوتی الحاق شده باشد لازم نیست در قسمت مسیر فایل Wave Play مسیر کامل را بنویسیم تنها کافیست که نام فایل صوتی بدون پسوند نوشته شود.

WavePlay ("Path to file ", "Loop")

پارامتر اول مسیر فایل صوتی Wav را مشخص می‌کند و پارامتر دوم جهت ایجاد تکرار استفاده می‌شود.

WaveStop ()

سبب توقف اجرا می‌شود.

مثال:

WavePlay("D:\Blue.Wav ", " Loop")

فرمان‌های «Audio Rewind, Audio Pause, Audio Stop, Audio Play, Audio Open»

فرامین گفته شده به ترتیب از راست به چپ برای بازگشایی فایل صوتی، اجرای فایل صوتی، توقف اجرای فایل صوتی، وقفه موقت در اجرا و جلو و عقب بردن می‌باشند. این فرامین برای کنترل فایل‌های ASF, WMA, OGG, MP3 می‌باشند. در ضمن این فایل‌ها بایستی خارجی باشند نه الحاق یافته.

AudioOpen(" Path to file")

در قسمت پارامتر مسیر فایل صوتی مشخص می‌شود. اگر در قسمت پارامتر از رشته خالی استفاده کنیم یعنی چیزی ننویسیم این فرمان قادر باز کردن فایل را نمایش خواهد داد. همانند سایر کادرهای بازگشایی این کادر بازگشایی نیز قابلیت انتخاب چندین فایل همزمان را برای ما فراهم می‌کند و اسامی و مسیر و تمام فایل‌های انتخاب شده درون یک لیست اجرایی درونی و در ماکروی <List> ذخیره می‌شود.

AudioPlay()

AudioStop()

AudioPause()

AudioRewind("seconds", "Command")

پارامتر اول زمانی را بر حسب ثانیه معین می‌کند که بایستی بر اساس آن به عقب بازگشت. برای اینکه عمل جلو بردن را انجام دهیم می‌توانیم در پارامتر اول در کنار زمان علامت منفی را قرار دهیم. در پارامتر دوم نیز می‌توانیم کلمه RELATIVE را قرار دهیم تا عمل‌های عقب و جلو بردن نسبی صورت گیرد.

مثال:

```
AudioOpen ("F:\Music\Bird.mp3")  
AudioRewind ("-3000", "")
```

فرامین «CDPlay, CDStop, CDPause, CDTrack, CDForward,

»CDBackward, CDSkipForward, CDSkipBackward, CDPlayerPause, Which CD Track

همان طور که واضح است فرامین گفته شده به ترتیب از چپ به راست، برای اجرای CD صوتی، توقف CD صوتی، ایجاد وقفه موقت در اجرای صوتی، مشخص کننده Track آغازین، جلو بردن CD صوتی، عقب بردن CD صوتی، باعث جلو رفتن به مدت ۱۰ ثانیه می‌شود، باعث عقب رفتن به مدت ۱۰ ثانیه می‌شود، باعث توقف CD می‌شود و فرمان آخر شماره فایل صوتی در حال اجرا را باز می‌گرداند. لازم به تذکر است که چنانچه قصد اجرای CD های Mixed-Mode را دارید شماره اولین Track صوتی به جای یک از ۲ شروع می‌شود زیرا اولین Track داده‌های دیگر می‌باشند.

CDPlay()

CDStop()

CDPause()

CDTrack("Track # ")

در این فرمان در قسمت پارامتر شماره Track (شروع از یک قرار) می‌گیرد.

CDForward()

CDBackward()

CDSkipForward()

CDSkipBackward()

CDPlayerPause()

WhichCDTrack("Track Number")

قسمت پارامتر متغیری را مشخص می‌کند که شماره Track در حال اجرا به آن اختصاص داده می‌شود.

مثال:

```
WhichCDTrack("Track No" )
```

« Midi Stop, Midi Play » فرامین

این فرامین به ترتیب برای اجرای فایل‌های با پسوند MIDI و توقف اجرای آنها می‌باشند.

```
MidiPlay("Path to MIDI " , "LOOP ")
```

پارامتر اول مسیر فایل MIDI را مشخص می‌کند و پارامتر دوم سبب ایجاد تکرار در اجرای فایل می‌شود. نوشتن پارامتر دوم اختیاری است.

```
MidiStop()
```

مثال:

```
MidiPlay("C:\MyMusic\Fall.MIDI")
```

« MOD Stop, MOD Play » فرامین

به ترتیب از راست برای اجرا و توقف فایل‌های MOD می‌باشند.

```
MODPlay("Path to MOD")
```

در قسمت پارامتر مسیر فایل صوتی قرار می‌گیرد.

MODStop()

مثال:

```
MODPlay("C:\MyMusic\Fall.MOD")
```

فرامین «StopSound , PlaySound»

فرمان سمت راست باعث اجرای کلیه فایل‌های صوتی پیش‌بینی شده توسط MMB می‌شود. ضمناً از این فرمان می‌توانیم پس از فرمان File Open استفاده کنیم و در قسمت پارامتر PlaySound می‌توانیم به جای مسیر از ماکروی <File> استفاده کنیم. فرمان دوم هم کلیه فایل‌های صوتی در حال اجرا از جمله آهنگ زمینه را متوقف می‌کند.

PlaySound("Path")

در قسمت پارامتر بایستی فایل صوتی را مشخص نمود.

StopSound()

مثال:

```
PlaySound("C:\MyMusic\Fall.Mp3")
```

فرامین « VolumeDown , VolumeUp »

به ترتیب از راست به چپ، باعث افزایش حجم صدا بر اساس مقدار ورودی می‌شود و فرمان بعدی حجم صدا را هر با به میزان ۰.۵٪ کاهش می‌دهد.

VolumeUp("Volume")

در قسمت پارامتر می‌توان درصدی را که می‌خواهیم بر اساس آن صدا بلندتر شود را وارد کرد (مقادیر صحیح بین 100 تا -100). با استفاده از این فرمان نیز می‌توان حجم صدا را کاهش داد. برای این منظور بایستی در کنار درصد علامت منفی را نیز قرار دهیم.

VolumeDown()

مثال:

```
VolumeUp("50")
```

فرامین «PrintRect, PrintPage, PrintText»

فرمان اول از سمت راست سبب چاپ محتوی شی متنی می‌شود، فرمان دوم باعث چاپ کل صفحه می‌شود و فرمان سوم سبب چاپ محدوده‌ی محصور بین کادر شی فرضی مستطیل شکل خواهد شد.

```
PrintText("Text Object","FONT – FRM – OBJECT")
```

در این فرمان پارامتر اول شی متنی را مشخص می‌کند و پارامتر دوم سبب چاپ متن با همان فونت اصلی خودش می‌شود، اما چنانچه این پارامتر حذف شود MMB متن را بر اساس تنظیمات قسمت چاپ، چاپ خواهد نمود.

```
PrintPage("fit")
```

```
PrintRect("Rectangle", "fit")
```

پارامتر اول شی‌ای را که به منزله‌ی شی فرضی مستطیلی عمل می‌کند را مشخص می‌کند و پارامتر دوم هم درصد اشغال صفحه را مشخص می‌کند:

مثال:

```
PrintText("Paragraph","FONT – FRM – OBJECT")
```

فرامین **Resume TTS, Pause TTS, Stop TTS, Speak Text, Say, Init TTS, Install TTS, Speed TTS, Pitch TTS**

Text – to- Speech چیست؟

به یک فرآیند گویند که طی آن یک متن به صوت دیجیتالی مبدل می‌شود و سپس بازگو می‌شود. فرامین مذکور جهت انجام همین فرآیند می‌باشند. TTS مخفف عبارت Text – to- Speech است.

Install TTS باعث نصب MS Text – to- Speech از یک مسیر ویژه می‌شود. ابتدا کامپیوتر کاربر از نظر داشتن این ویژگی بررسی می‌شود و چنانچه این ویژگی از قبل موجود نباشد MMB به سرعت اقدام به نصب آن می‌نماید. تنها کافیست فایل اجرایی نصب TTS را بر روی CD مربوطه قرار دهید. تا MMB عملیات نصب را به صورت خودکار انجام دهد. نام این فایل اجرایی Ms tts 221.exe است.

اگر در طراحی پروژه خود قصد استفاده از TTS را داشتید حتماً لازم است که دستور Install TTS را در ابتدای برنامه بگنجانید. چنانچه کاربر نهایی از قبل این ویژگی را نصب کرده باشد برنامه بدون توقف به اجرای باقی فرامین

می‌پردازد. `Init TTS` باعث بنیان نهادن ویژگی `TTS` می‌شود. این فرمان ممکن است برای بنیان نهادن `TTS` چند ثانیه را صرف کند و اغلب بر روی سیستم‌های با سرعت پایین بیشتر طول می‌کشد. لذا بهترین مکان برای قرار دادن این فرمان بلافاصله پس از فرمان `Install TTS` می‌باشد. `MMB` در هنگام بنیان نهادن ویژگی `TTS` توسط این دستور کادر سیاه رنگی مبنی بر این عمل را نشان می‌دهد. البته این ویژگی در ویندوزهای 9x ممکن است به درستی عمل نکند. زیرا راه انداز کارت صوتی ممکن است که با این ویژگی سازگار نباشد. بهترین کار در این صورت بازدید از سایت تولید کننده کارت صوتی به منظور دریافت اطلاعات بیشتر می‌باشد. `Say`، باعث گفتن متنی می‌شود البته بایستی `TTS` از قبل نصب شده باشد. `SpeakText` سبب گفتن متن هر شی که حاوی متن است می‌شود. در این حالت نیز باید `TTS` از قبل نصب شده باشد. `Stop TTS` گفتن را متوقف می‌کند. `Pause TTS` سبب ایجاد مکث در گفتن متن می‌شود و `Resume TTS` باعث از سرگیری مجدد گفتن می‌شود. `Pitch TTS`، این فرمان برای تنظیم زیر و بم صدا استفاده می‌شود. `Speed TTS` سرعت گفتن را بر اساس واحد کلمه در ثانیه تنظیم می‌کند.

`InstallTTS()`

`InitTTS()`

`Say("Text")`

متنی را که می‌خواهیم قرائت شود در بین پرانتز در قسمت پارامتر قرار می‌گیرد.

`SpeakText("Text object")`

در قسمت پارامتر شی متنی مشخص می‌شود.

`StopTTS()`

`PauseTTS()`

`ResumeTTS()`

`PitchTTS("Value")`

در قسمت پارامتر مقدارهای بین 50 تا 200 هرتز را می‌توان وارد کرد. مقدار پیش فرض 100 هرتز است.

`SpeedTTS("Value")`

مقدار سرعت در قسمت پارامتر فرمان می‌آید، مقادیر می‌تواند بین 30 تا 450 باشد. مقدار پیش فرض 150 است.

فرامین (MP3 List)

در این قسمت یک لیست اجرایی (Play List) درونی موجود می‌باشد که شما می‌توانید فایل‌های پشتیبانی شده توسط MMB را به این قسمت اضافه نمایید. یا در صورت تمایل لیست اجرایی خارجی را (M3L\M3u) وارد نمایید.

`SongListReset()`

این فرمان باعث پاک کردن یک آهنگ قسمت از لیست اجرا می‌شود.

`SongListAdd("< Src Dir > \ Path ")`

باعث افزودن یک لیست اجرایی دیگر به لیست اجرایی درونی می‌شود.

`SongListDel("number")`

باعث پاک کردن آهنگ مشخص شده از لیست می‌شود.

`SongListPlay("number")`

این فرمان باعث اجرای آهنگ‌های درون لیست، شروع از یک می‌شود.

`SongListNext()`

باعث اجرای آهنگ بعد آهنگ جاری از لیست اجرایی می‌شود.

`SongListPrev()`

باعث اجرای آهنگ قبلی آهنگ جاری می‌شود.

`SongListLoad("Path")`

این فرمان باعث وارد کردن یک لیست خارجی به درون لیست درونی می‌شود و یا در حالت دیگر می‌توانید مستقیماً اقدام به وارد کردن فایل‌های صوتی نمایید.

مثال:

```
OpenFile("PlayLists (*.m3l;*.m3u;*.pls)|*.m3l;*.m3u;*.pls|AllFiles|*.*|", "*.m3l;*.m3u;*.pls")  
SongListLoad("OpenFile$")
```

SongListRND()

باعث ایجاد لیست درونی تصادفی می شود.

SongListEdit()

این فرمان باعث بازگشایی یک کادر ویژه می شود که کاربر می تواند بر حسب نیاز به ویرایش لیست اجرایی پردازد و آن را ذخیره و یا بارگذاری نماید. همچنین در این کادر ویژه می توان با درگ کردن فایلها از Explorer به درون این کادر فایلها را به لیست اضافه نمود. ضمناً این قابلیت فراهم است تا به توسط فرامین SongListDel ، ListLoad و SongListSave به بارگذاری، ذخیره و پاک کردن این لیست پرداخت اما این شرایط برای فایلهایی که از Explorer درگ شده اند، صادق نیست.

SearchForSong("Path")

این فرمان بر اساس مسیر داده شده به جستجوی فایل های پشتیبانی شده توسط MMB می گردد و پس از یافتن فایلها لیست اجرایی را پر می کند.

مثال:

```
SearchForSong("<SrcDir>\MyMP3\")
```

یا

```
OpenFile("Play Lists(*.mp3;*.ogg;*.asf;*.wma) | *.mp3;*.ogg;*.asf;*.wma|All  
Files|*.*||", "*.mp3;*.ogg;*.asf;*.wma")  
SelDir$=CBK_OpenDir  
SearchForSongs("SelDir$")
```

SongListSave("ListBoxObject","Path to save")

این فرمان باعث ذخیره کردن محتوی لیست اجرایی درونی در یک فایل می شود.

فرامین تصویر (External Image Commands)

فرمان «ViewJPG»

VeiwJPG(" Path to EXTERNAL JPG " , " CENTER ")

از این فرمان برای مشاهده فایل های تصویری با پسوند BMP و JPG استفاده می شود. پارامتر اول این دستور مسیر فایل عکس را مشخص می نماید و پارامتر دوم سبب نمایش دادن عکس در وسط صفحه می شود. این فرمان برای CD های حاوی عکس مناسب می باشد. MMB در هنگام نمایش دادن عکس مربوطه آن را به درون پنجره ای قرار می دهد که چنانچه کاربر بر روی آن کلیک نماید باعث بستن پنجره می شود.

ReplaceImage("Bitmap Object" , "PATH TO EXTERNAL JPG File")

این فرمان سبب جایگزینی شی عکس مشخص شد. با عکس جدیدی می شود. به عنوان مثال فرض کنید که شی عکسی در صفحه دارید، حال بر حسب نیاز احتیاج به تغییر آن دارید، می توانید با این فرمان این کار را صورت دهید.

جای مناسب دیگری که می توان از این فرمان سود جست در کاتالوگ ها و برنامه هایی است که بر اساس مقداری عکس معنا دار می باشند است و بدین وسیله هم سرعت اجرای برنامه را بالا برده و هم می توانید محصول بزرگتری را تولید نمایید. در این فرمان پارامتر اول، شی عکسی را مشخص نموده و پارامتر دوم برای مسیر دهی عکس دوم که جایگزین خواهد شد است. از دیگر ویژگی های این فرمان اینست که می توانید به جای مشخص کردن شی عکسی از Hot-Spot استفاده نمایید و از آن به عنوان محل نمایش عکس جدید استفاده نمایید. بدیهی است که فرامین اختصاص داده شده به شی Hot-Spot به قوت خود باقی می ماند. ضمناً برای حذف عکس جاری از محل بارگذاری در صفحه می توان پارامتر دوم فرمان را خالی گذاشت.

یکی از راه های بهینه سازی پروژه استفاده مناسب از این فرمان می باشد. فرض کنید که در هنگام شروع برنامه عکسی بارگذاری می شود حال می توانید با استفاده از دستور Replace Image و پارامتر دوم که خالی است در هنگام بستن صفحات عکس دار مقداری از حافظه را که توسط آنها اشغال شده است را به سیستم باز گردانید. دستور Replace Image فرمت دیگری هم دارد که برای بارگذاری یا حذف عکس در صفحات استفاده می شود. برای مثال می توانید از صفحه جاری عکسی را در صفحه ای دیگر بارگذاری نمایید.

مثال:

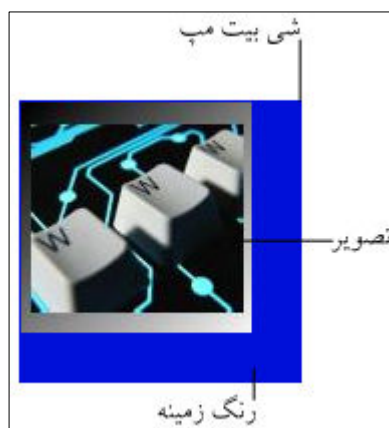
```
ReplaceImage("Page 1::Bitmap2")
```

فرمان « UseImageBkgColor »

این فرمان رنگی را به عنوان زمینه در محدوده ی شی عکس نمایان می سازد. این فرمان ۲ پارامتر دارد که یکی برای مشخص کردن نام شی عکس است و دومی برای فعال یا غیر فعال کردن رنگ زمینه.

اما جهت پارامتر دوم می تون به ترتیب از اعداد 0 (برای غیر فعال سازی) و 1 (برای فعال سازی) استفاده کرد.

UseImageBkgColor("Image","Switch")



فرمان « SetImageBkgColor »

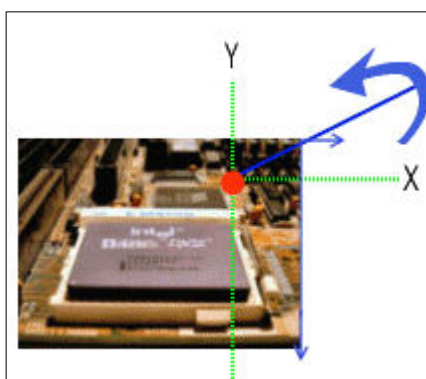
این فرمان رنگ زمینه را تعیین می کند.

SetImageBkgColor("Image ", "RGB Value")

این فرمان ۲ پارامتر دارد که یکی برای مشخص کردن نام شی عکس است و دیگری برای مقدار رنگ که در کانال RGB معرفی می شود است.

فرمان « SetImageOrigin »

از این فرمان جهت تنظیم نقطه شروع شی عکس استفاده می شود (به عبارتی تکیه گاه شی عکس) تنظیم این نقطه بر اساس منتهی علیه بالا سمت چپ شی عکس صورت می گیرد. کاربرد این فرمان برای ۲ حالت تغییر اندازه شی و چرخش شی می باشد و چرخش شی و تغییر اندازه عکس حول این نقطه صورت می گیرد.



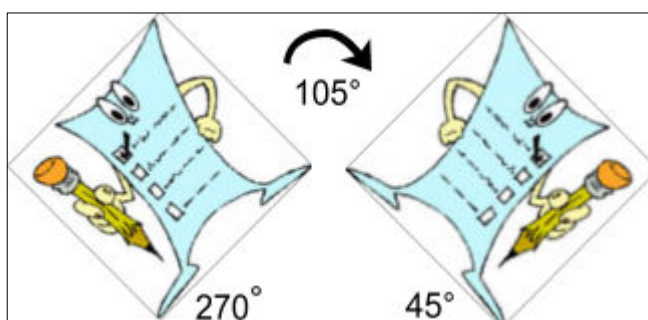
`SetImageOrigin("Image","X,Y")`

این فرمان هم متشکل از ۲ پارامتر است که اولین آن برای نام شی و دومی برای مشخص کردن نقطه می باشد. قسمت دوم خود مستقل از دو قسمت X جهت فاصله افقی و Y جهت فاصله عمودی است.

مثال:

`SetImageOrigin("Image" , "80,30")`

فرمان « `RotateImageRel` »



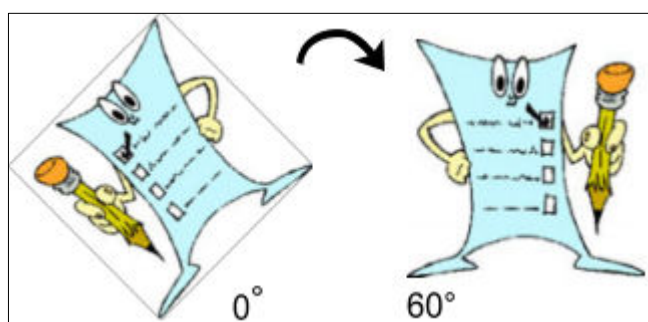
این فرمان سبب چرخش عکس به نسبت زاویه جاری عکس می شود.

RotateImageRel("Image","Degree")

جهت استفاده از این فرمان باید ۲ پارامتر را وارد کرد: اولی نام شی تصویر می باشد و دومی مقدار زاویه چرخش.

« RotateImageTo » فرمان

این فرمان سبب چرخش مطلق شی می شود. بدین معنی که اگر زاویه جاری بین تصویر و افق 45° درجه باشد و ما بخواهیم تصویر چند درجه ای بچرخد، زاویه جاری صفر در نظر گرفته می شود و تصویر به مقدار زاویه داده شده می چرخد.



RotateImageTo("Image","Degree")

همانند قبل این فرمان هم ۲ پارامتر دارد که یکی جهت معین کردن نام شی تصویر و دیگری جهت مشخص کردن مقدار زاویه.

« ScrollImageView » فرمان

گاهی اوقات است که تصویر بارگذاری شده در محدوده ی شی تصویر تماماً قابل دید نیست و چاره ی کار استفاده از دستور پیمایش است. این دستور ۲ پارامتر دارد اولی برای مشخص کردن نام شی و دومی برای مشخص کردن مقدار پیمایش. در پارامتر دوم مقدار X برای طول Y برای عرض است. همچنین می توان از مقادیر منفی استفاده کرد.

`ScrollImageView("Image","X,Y")`

فرمان « **ZoomImageView** »

از این فرمان برای بزرگ نمایی و یا کوچک نمایی تصویر استفاده می شود. برای این دستور ۲ پارامتر موجود است یکی جهت مشخص کردن نام شی و دیگری برای مقدار تغییر از نظر کوچکی و بزرگی، مقدار پیش فرض ۱۰۰ است.

`ZoomImageView("Image","Value")`

فرمان « **RestoreImage** »

چنانچه بخواهیم تغییرات اعمال شده به شی تصویر از قبیل چرخش و تغییر اندازه را خنثی کنیم از این فرمان استفاده می کنیم. این فرمان تنها یک پارامتر دارد و آن هم تنها نام شی می باشد.

`RestoreImage("Image")`

فرمان "**ImageOpacity**"

`ImageOpacity("OpacityLevel")`

توسط این فرمان می توان شفافیت تصویر را بین محدوده ۱ تا ۱۰۰ تغییر داد.

مثال :

`ImageOpacity("50")`

توابع ماتریس «**Matrix function**»

فرمان «MatrixSet»

Matrix Set ("Matrix Object [Column , Row] , "Image index (0...3) ")

این فرمان باعث تنظیم یک تصویر برای خانه ای از ماتریس می شود. در قسمت اول پارامتر اول نام شی ماتریس را قرار می دهیم، در قسمت دوم همین پارامتر به ترتیب از چپ شماره ستون و سطر را می نویسیم. در ادامه در پارامتر دوم شماره تصویر مورد نظر را می آوریم.

به عنوان مثال تمایل داریم که در موقعیت ۳ و ۲ (سطر و ستون) ماتریس تصویر شماره ۲ را نمایش دهد:

```
MatrixSet ("My Matrix [ 2,3] " . " 2" )
```

چنانچه بخواهیم سطرها و یا ستون ها را به یک تصویر از ماتریس تغییر دهیم می توانیم به جای شماره سطر یا ستون از عدد صفر استفاده نماییم.

```
MatrixSet ("My Matrix [ 5,0] " . " 2" )
```

این گونه از فرمان باعث تنظیم تصویر شماره ۲ برای تمام سطور می شود.

```
MatrixSet ("My Matrix [ 5,0] " . " 2" )
```

این گونه از فرمان باعث تنظیم تمام ستونها برای نشان دادن تصویر شماره ۲ می شود.

```
MatrixSet ("My Matrix [ 0,0] " . " 2" )
```

این گونه از فرمان باعث تنظیم تمام سطرها و ستون ها و سطور برای نشان دادن تصویر ۲ می شود.

فرمان «MatrixGet»

MatrixGet ("Matrix Object [Column, Row]" . "Variable")

این فرمان شماره عکس جاری در موقعیت مشخص شده را به یک متغیر اختصاص می دهد.

مثال:

```
MatrixGet ("MyMatrix[2.3]","Cur")  
If ( Cur = 1 ) then  
    MatrixSet ("MyMatrix[2.3]"," 2")
```

End

برخی اوقات لازم است که بدانیم کاربر بر روی کدام درایه و موقعیت ماتریس کلیک کرده است برای این کار می توانیم از ثابت های $MxRow$, $MxCOL$ و نسبت دهی آنها استفاده نماییم. دو فرمان زیر را در نظر بگیرید.

```
MatrixSet("MyMatrix [ 0,0] " . " 1" )
```

```
MatrixSet("MyMatrix [ Mx C ol, Mx Row] " . " 2" )
```

اولین فرمان باعث تنظیم کلیه قسمتها برای نشان دادن شکل ۱ می شود و دومین فرمان قسمت کلیک شده را با شکل شماره ۲ تعویض می کند.

فرمان «SysCommand»

```
SysCommand("Command","Parameters" )
```

این فرمان برای کنترل و تغییر خصوصیات و ویژگی های پیشرفته پنجره ها استفاده می شود. البته یک استثناء و آن هم برای Copy File وجود دارد. پارامتر اول نوع دستور است و پارامتر دوم بر حسب نوع دستور معین می شود.

فرامین:

```
SysCommand("ResizeWindows", "Width,Height")
```

از این فرمان برای تغییر اندازه پنجره استفاده می شود. در قسمت width (عرض) و height (ارتفاع) می توان مقدار مورد نیاز را قرار داد. این مقادارها می توانند ثابت و یا متغیر باشند.

مثال:

```
SysCommand("ResizeWindows", "0,400")
```

این فرمان در مثال بالا باعث تغییر اندازه پنجره به ترتیب به ارتفاع ۴۰۰ پیکسل و عرض بدون تغییر می شود.

```
a=200
```

```
b=100
```

```
SysCommand("ResizeWindows", "a,b")
```

این فرمان اندازه را به مقدار 100×200 تغییر می دهد.

```
SysCommand("CenterWindow","")
```

این فرمان باعث تغییر مکان پنجره به وسط صفحه نمایش می شود.

SysCommand("MoveWindows", "X,Y")

این فرمان باعث تغییر مکان پنجره پروژه بر اساس پارامترهای X و Y می شود.

نکته: MMB از قبل ۲ متغیر از پیش تعیین شده ScreenWidth و ScreenHeight را در اختیار شما جهت دریافت اندازه پنجره جاری قرار می دهد.

SysCommand("AlwaysOnTop", "")

این فرمان پنجره پروژه را به بالای تمامی پنجره ها و بر روی همه ی پنجره های دیگر موجود منتقل می نماید.

SysCommand("NotAlwaysOnTop", "")

این فرمان باعث می شود که موقعیت قرارگیری پنجره پروژه از نظر قرار گیری بر روی سایر پنجره ها به موقعیت اولیه و معمولی اش باز می گردد.

SysCommand("CopyFile", "Source\$,Destination\$")

این فرمان باعث کپی شدن یک فایل از منبع به مقصد می شود. چنانچه پوشه مقصد موجود نباشد این دستور آن را ایجاد می نماید در این دستور پارامتر Source\$ منبع را مشخص و Destination\$ مسیر را مشخص می نماید.

مثال:

```
destdir$ = 'C:\Program Files\Copy Test'  
Dest$ = destdir$ + '\TestFile.txt'  
Source$ = '< Embedded >\Text.txt'  
SysCommand("CopyFile" , "Source$ , Dest$" )
```

فرامین متحرک سازی «Animation Commands»

MoveObject("Object", "X,Y,W, H")

از این دستور برای تغییر مکان یک شی (گروه) بر اساس پارامترهای X,Y استفاده می شود. همچنین با مقدار دادن پارامترهای W (عرض) و H (ارتفاع) می توان همزمان اندازه شی را تغییر داد. البته شی های متنی از این قاعده مستثنی است و این شی ها قابلیت تغییر اندازه توسط این دستور را ندارند.

مثال:

```
For i=0 to 100
MoveObject ("Bitmap", "i,20")
Refresh ()
Pause ("30")
Next i
```

ترفند: ترکیب ثوابت () MouseX() و MouseY() با MoveObject و حلقه next ... for سبب حرکت بلادرنگ شی بر اساس حرکت موس می شود.

MoveTo("Object", "X,Y,Steps,Type")

این فرمان ساده باعث حرکت دادن شی از مکان جاری به مکان مشخص شده بر اساس پارامترهای X,Y می شود. پارامتر Steps مقدار جهش حرکت را بیان می کند و پارامتر Type نوع حرکت طولی را تعیین می کند که آیا شتاب دار باشد یا خیر؟ برای پارامتر Type می توان از عبارت های EASY TO و

EASYFORM یا هیچ استفاده کرد.

EASY TO: باعث شروع با سرعت معمولی شده و هر چه به انتهای مسیر نزدیک می شود سرعت کم تر می شود.

EASYFORM: حرکت از حالت آهسته شروع شده و به مراتب سرعت بیشتر می شود.

«فرامین ListBox»

-ListBoxAddItem("Object", "Parameter")

این فرمان سبب اضافه شدن یک گزینه به ListBox می شود. پارامتر اول شی ListBox را معین می کند.

در قسمت پارامتر دوم این فرمان می توانید از عبارت های ذیل استفاده نمایید:

- متن (this is test)

- متغیر رشته ای (text\$)

- آرایه ی رشته ای (text[]\$)

- فایل متنی (C:\My Documents\ testfile.txt)

- لیست اجرایی صوتی

(<SrcDir>\ mix.m3u) ⏪ (*. Pls,m3u *.m3l)

- Reset جهت پاک کردن محتویات شی ListBox

- RANDOMIZE جهت تصادفی پر کردن ListBox

مثال:

```
OpenFile("Play Lists (*.m3l;*.m3u;*.pls) | *.m3l;*.m3u;*.pls | All  
Files|*.*||", "*.m3l;*.m3u;*.pls")  
ListBoxAddItem("ListBox", "RESET")  
ListBoxAddItem("ListBox", "OpenFile$")  
ListBoxDeleteItem("Object", "Number")
```

این فرمان بر اساس شماره داده شده در قسمت Number گزینه ای را پاک می نماید چنانچه بخواهیم تمامی محتویات پاک شود از عدد ۱- استفاده می کنیم.

نکته: شماره گذاری گزینه های ListBox همیشه از ۱ یک شروع می شود.

این مثال پیشرفته گزینه های انتخاب شده را توسط دکمه Delete از صفحه کلید پاک می کند.

تمرین:

- یک شی Script ایجاد نمایید و میانبر آن را به کلید Delete نسبت دهید.

- اسکریپت های ذیل را در شی اسکریپتی کپی نمایید.

- یک ListBox در صفحه ایجاد نمایید.

```
ListBoxGetSelectedItems("ListBox", "SelItemsArray$, NumSelItemsArray$, #, NumOfSel  
Items")  
for i=NumItems to 1
```



```
numitem$ = GETARRAYITEM(NumSelItemsArray$, #, i)
numitem = VAL(numitem$)
ListBoxDeleteItem("SongList", "numitem")
next i
```

-ListBoxSelectItem("Object" , "Number")

این فرمان سبب انتخاب گزینه ای بر اساس پارامتر Number می شود. چنانچه بخواهید تمامی آیتم ها انتخاب شود باید در قسمت پارامتر این فرمان از عدد (۱-) استفاده نمایید.

مثال:

```
ListBoxSelectItem("ListBox", "3")
```

-ListBoxDeselectItem("Object" , "Number")

این فرمان بر خلاف فرمان قبل عمل می نماید بدین معنی که بر اساس پارامتر Number گزینه یا گزینه هایی را از حالت انتخاب خارج می نماید جهت درآوردن تمامی گزینه ها از حالت انتخاب باید از عدد ۱- در قسمت Number استفاده کرد.

مثال:

```
ListBoxDeselectItem("List Box", "-1")
```

-ListBoxGetItem("Object" , "Param1.Param2,Param3,Param4")

از این فرمان برای دریافت تمام گزینه های یک List Box بهره برده می شود. اولین پارامتر باید یک متغیر رشته ای جهت نگهداری رشته گزینه ها باشد که این رشته ها توسط یک کاراکتر محدودگر از هم متمایز می شود. همچنین پارامتر دوم هم باید یک متغیر رشته ای باشد که بتواند تقدم گزینه ها را ذخیره نماید. پارامتر سوم کاراکتر محدودگر را مشخص می نماید. و پارامتر چهارم که باید متغیر عددی باشد برای دریافت تعداد گزینه ها است.

مثال:

```
ListBoxGetItems("ListBox", "ItemsArray$, NumItemsArray$, #, NumOfAllItems")
for i=1 to NumOfAllItems
GetItem$ = GETARRAYITEM(ItemsArray$, #, i)
Message("Selected Item...", "GetItem$")
next i
```

-ListBoxGetSelectedItems("Object", "Param1,Param2,Param3,Param4")

این فرمان محتوی گزینه ی انتخاب شده را دریافت می دارد. اولین پارامتر این دستور باید یک متغیر رشته ای باشد که جهت نگهداری محتوی گزینه ها است. همچنین پارامتر دوم باید یک متغیر رشته ای باشد که تقدم گزینه ها را دریافت دارد. پارامتر سوم کاراکتر محدودگر جهت سوا کردن رشته ها از هم می باشد و پارامتر چهارم بایستی یک متغیر عددی باشد تا تعداد گزینه ها را دریافت کند.

مثال:

```
ListBoxGetSelectedItems("ListBox", "SelItemsArray$, NumSelItemsArray$, #, NumOfSelItems")
for i=1 to NumOfSelItems
SelItem$ = GETARRAYITEM(SelItemsArray$, #, i)
Message("Selected Item...", "SelItem$")
next i
```

-ListBoxSortItems("Object", "Type")

این فرمان مولفه های شی ListBox را بر اساس نام (NAME)، زمان (TIME) و یا برعکس (REVERSE) یا تصادفی (RANDOM) مرتب می کند. اولین پارامتر این دستور برای مشخص کردن نام شی ListBox است و در قسمت پارامتر دوم می توان از کلمات گفته شده بالا برحسب نیاز استفاده کرد.

مثال:

```
ListBoxSortItems("ListBox", "NAME")
```

-ListBoxMoveItem("Object", "Number")

این فرمان حالت انتخاب را به جایی که توسط پارامتر دوم معرفی می شود انتقال می دهد. پارامتر اول این فرمان برای تعیین نام شی `ListBox` است و پارامتر دوم موقعیت حرکت را مشخص می کند. پارامتر دوم هم می تواند عدد باشد و هم متغیر عددی.

توجه: شماره مؤلفه های `ListBox` همیشه از یک شروع می شود.

-ListBoxParam("Object","Switch")

هنگامی که بخواهیم یکسری از خصوصیات شی `List Box` را که در ادامه بحث می شود تنظیم کنیم از این فرمان استفاده می کنیم. به طور معمول پارامتر اول برای نام شی است و در قسمت پارامتر دوم کلمات زیر جای می گیرد که به ترتیب با انتساب آنها با `ON` و `OFF` می توان آن خصوصیت را فعال یا غیر فعال کرد.

DRAG & DROP: این پارامتر جهت ایجاد و حالت کشیدن و انداختن می باشد که به وسیله آن می توان آیتمی را با موس جا به جا کرد.

NUMBERS: از این پارامتر جهت نشان دادن / حذف اعداد کنار آیتم های لیست استفاده می شود.

TIMES: از این پارامتر جهت نشان دادن / حذف زمان کنار آیتم های لیست استفاده می شود.

BACKGROUND: از این پارامتر برای تعیین رنگ زمینه بر اساس کانال `RGB` استفاده می شود. با انتساب عدد مورد نظر رنگ تعیین می شود.

TEXT: این پارامتر برای تعیین رنگ متن ها بر اساس کانال `RGB` می باشد.

IDTAGS: این پارامتر هم در صورت فعال بودن تگ های `ID` فایل های صوتی را نشان می دهد. جهت فعال سازی و غیر فعال سازی این پارامتر می توان از `OFF/ON` استفاده کرد.

مثال:

```
ListBoxParam("ListBox","NUMBERS = OFF")  
ListBoxParam("ListBox","TIMES = ON")
```

فرامین «Audio Visualization»

-AudioVisualizationType("AV Object" , "AV Type")

این فرمان نوع رقص نور را مشخص می نماید. پارامتر AV TYPE می تواند دو عبارت زیر باشد:

- ANALYZER

- OSCILLOSCOPE

-AudioVisualizationColor("AV Object" , "Color")

این فرمان رنگ جدیدی را برای شی رقص نور تعیین می نماید. پارامتر Color می تواند به حالت های زیر باشد:

BACKGROUND = RGB

جهت تعویض رنگ زمینه

R,G,B

OSCILLOSCOPE جهت تعویض رنگ منحنی

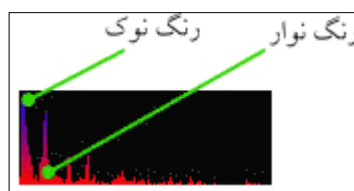
مثال:

```
AudioVisualizationColor("AudioVis", "BACKGROUND=150,79,205")
```

این مثال رنگ زمینه شی AV را تغییر می دهد..

```
AudioVisualizationColor("AudioVis", "RGB(255,0,0),RGB(0,0,255)")
```

این مثال رنگ نوارها و نوک آنها را در نوع ANALYZER تغییر می دهد



فرمان «LoadText»

```
LoadText("Text object" , " Variable $ ")
LoadText("Text object" , " Path ")
LoadText("Text object" , "< List > ")
LoadText("Text object" , "< List > Number")
LoadText("Variable $ " , " Path ")
LoadText("Variable $ " , "< List > Number ")
```

یک فرمان مهم برای متغیرهای رشته ای می باشد. این فرمان سبب بارگذاری یک رشته، یک فایل از یک مسیر، یک لیست یا یک مولفه منفرد به درون یک شی متنی می شود. این فرمان قادر است یک فایل متنی را از مسیری به درون شی متنی بارگذاری نماید، یا به یک متغیر رشته ای اختصاص دهد.

همچنین این فرمان می تواند محتوی یک متغیر رشته ای را به درون یک شی متنی بارگذاری نماید.

```
C$='<SrcDir>\My File.txt'
LoadText("Object" , "C$")
```

این مثال یک فایل متنی را به درون یک شی متنی بارگذاری می نماید.

```
C$='what ever text'
LoadText("Object" , "C$")
```

این مثال متن مشخص شده را درون شی متنی نمایش می دهد.

چنانچه مسیر مشخص شده یک فایل متنی اشتباه و یا ناقص باشد در نهایت در شی متنی همان مسیر اشتباه و ناقص نمایان خواهد شد.

« فرمت مستقیم Direct Format »

فرمت قبلی به MMB می گفت که چه کند. اما چنانچه در استفاده از این فرمت (قبلی) شک داشته باشیم بهتر است از فرمت مستقیم بهره جوییم.

FILE : Variable

در این حالت، محتوی فایل متنی در شی متنی نمایان می شود..

STRING : Variable

در این حالت تنها مسیر فایل در شی متن نمایش داده می شود.

مثال:

```
Path$='<SrcDir>\MyFile.txt'  
LoadText("text","FILE:Path$")
```

این مثال سبب بارگذاری محتوی فایل مشخص شده به درون شی متنی می شود.

```
LoadText("text", "STRING : Path$")
```

این گونه از فرمان عیناً رشته اختصاص یافته به متغیر Path\$ را در شی متنی نمایان می سازد.

نکته: چنانچه بخواهید یک رشته طولانی را به خطهای کوچکتری تقسیم نمایید باید از پارامتر n در رشته استفاده نمایید.

مثال:

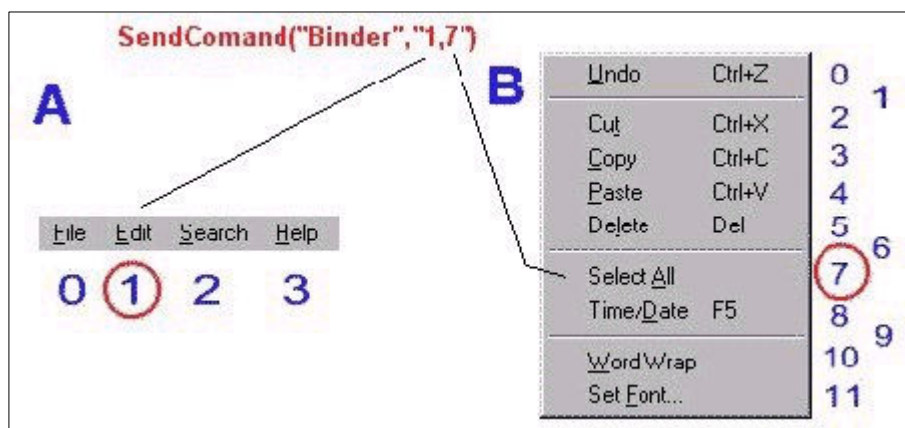
```
LongStr$='This is a very long string \n splitted into two rows'  
LoadText("text","LongStr$")
```

فرامین کنترل «Binder»

MMB قادر به فراخوانی مولفه های منو های برنامه های اتصال یافته می باشد. قاعده کار به این صورت است که بایستی شماره منو و مولفه را توسط فرمان SendCommand در اختیار MMB گذاشت.

«فرمان SendCommand»

```
SendCommand("Object", "N1,N2")
```



پارامتر اول شی Binder را مشخص می کند و پارامتر دوم برای شماره منو و مولفه مورد نظر است. همانند شکل می توانیم برای بدست آوردن شماره منو و مولفه عمل نماییم.

« ارسال رشته به شی جاسازی شده »

جهت ارسال رشته ای به شی جاسازی شده (اینجا Note pad) می توانیم از فرمان های MMB سود جوییم. ابتدا رشته را به Clipboard می فرستیم و سپس با انتخاب تمام رشته ها و عمل چسباندن، رشته را منتقل می کنیم.

مثال:

```
C$ = 'Hello form MMB'  
Clipboard("SEND", "C$")  
SendCommand("Binder", "1,7")  
SendCommand("Binder", "1,4")
```

با ایده ای مشابه می توان رشته ای را از شی اتصال یافته دریافت نمود، ابتدا تمام رشته را انتخاب می کنیم. سپس آنها را توسط کپی کردن به Clipboard ارسال نموده و توسط فرمان

Clipboard("Get", "Str\$") آن را دریافت می کنیم.

مثال: (در Notepad)

```
SendCommand("Binder", "1,7")  
SendCommand("Binder", "1,3")  
Clipboard("Get", "Str$")  
LoadText("TextDisp", "C$")
```

فرامین کنترل شی « HTML »

Browser("HTML Object", "Command")

پارامتر اول این فرمان نام شی HTML را مشخص می کند و پارامتر دوم می تواند یکی از کلمه های زیر باشد: (چنانچه با Microsoft IE کار کرده باشید، با عملکرد فرامین زیر به خوبی آشنا می باشید)

Back - فرمان به عقب را به شی HTML صادر می کند.

- Forward: فرمان به جلو را به شی HTML می دهد.

- Stop: فرمان توقف را صادر می کند.

- Refresh: فرمان تازه سازی را صادر می کند.

- OpenFile: کادر بازگشایی فایل را نمایان می سازد(تنها فایل های سازگار با IE قابل انتخاب می باشند).

- Print: فرمان چاپ سند جاری را صادر می کند.

- URL: مسیر سند مورد نظر را مشخص می کند. این مسیر می تواند بر روی اینترنت هم باشد.

می توانید از اتصالاتی خاص در شی HTML استفاده کنید که به MMB بگویند به صفحه ای دیگر برود یا اسکریپتی را اجرا نماید.

قاعده به صورت زیر است:

< a href = "page pag 2 "> Go to page 2 < \ a >

این اسکریپت باعث رفتن به صفحه ی ۲ پروژه می شود.

< a href = " script : Script " > Run Script < \ a >

این دستور برای اجرای اسکریپتی خاص استفاده می شود

« فرامین کنترل شی Flash »

Flash("Object", "Event")

با این فرمان می توانید مشخص کنید که فایل فلش چگونه اجرا شود. پارامتر اول این دستور، شی فلش را معین می کند و پارامتر دوم هم برای مشخص کردن رویداد می باشد. اما رویدادهای ممکن به قرار زیر است:

- PLAY: فایل فلش را اجرا می کند.
- STOP: فایل فلش را که در حال اجرا است متوقف می نماید.
- LOOP: سبب ایجاد تکرار در اجرای فایل فلش می شود.

- SHOW MENU: منوی Right - click را نشان می دهد.
- HIDE MENU: منوی Right - click را پنهان می کند.
- BACK: به فریم قبلی نسبت به فریم جاری می رود.
- FORWARD: به فریم بعدی نسبت به فریم جاری می رود.
- REWIND: سبب به عقب برگشتن فایل فلش می شود.

هر چیز غیر از رویدادهای بالا از قبیل رشته ها به منزله ی مسیر فایل تلقی می شود.

مثال:

```
OpenFile("Flash Files (*.swf)|*.swf|All Files|*.*||", "*.swf")
if (OpenFile$ <> '') then
Flash("Flash", "OpenFile$")
Flash("Flash", "PLAY")
end
```

FlashGetVar("Object", "FLASHVARname, MMBVARname")

فرمان بالا متغیر را از کلیپ فلش دریافت داشته و در متغیر برنامه MMB ذخیره می نماید.

توجه: متغیر مورد فراخوانی باید از قبل در فلش موجود باشد. چون که ترتیب اجرای اسکریپت های فلش ترتیب خاصی ندارد ممکن است فراخوانی یک متغیر بی نتیجه باشد. بنابراین چنانچه مطمئن نیستید که متغیر مورد نظر از قبل در شی فلش موجود است یا اینکه به درستی مقداردهی شده است، توسط فرمان (Pause) در برنامه MMB اجرای پروژه MMB را تا زمان مقداردهی متغیر در فلش متوقف سازید. راه حل دیگر کنترل MMB از طریق شی فلش و اسکریپت نویسی شی در فلش می باشد که لازمه آن داشتن فایل منبع (*.fla) کلیپ فلش می باشد. پارامتر اول این فرمان شی فلش را معین می کند و پارامتر دوم به ترتیب از چپ نام متغیر درون کلیپ فلش و نام متغیر MMB را معین می کند.

مثال:

```
FlashStr$ = 'MmbText.mmbstr'+ 'GetDateTime'
FlashSetVar ("Flash", "FlashStr$")
```

سبب تنظیم GetDateTime به متغیر MmbText.mmbstr واقع در کلیپ فلش می شود.

Pause (" 100 ")

به شی فلش مهلت انجام فعالیت های مورد نیاز را میدهد.

متغیرهای Date \$ و Time \$ توسط فرامین زیر مستقیماً از شی فلش مقداردهی شده اند.

```
**FsCommand ("mmb ", " Date $ = "+ MMB Cur Date Full )
```

```
**FsCommand ("mmb ", " Time $ = "+ MMB Cur Time Full )
```

```
Load text (" Date Str" , "Date $" )
```

```
Load text ("Time Str" , " Time $" )
```

اما متغیر Day\$ مستقیماً توسط فرمان FlashGetVar از MMB خوانده می شود.

```
FlashGetVar("Flash" , "MmbText.MMB cur day , Day $" )
```

```
LoadText(" DayStr" , "Day$")
```

FlashSetVar ("Object", "Flash VAR name , MMB VAR name ")

این فرمان سبب تنظیم مقدار جدید به متغیر کلیپ فلش می شود.

FlashSetFrame("Object" , "FRAMENUM")

این فرمان بر اساس پارامتر FRAMENUM شی فلش رابه فریمی که مد نظر است منتقل می کند.

FlashGetFrame("Object" , "VARIABLE")

شماره را فریم جاری را گرفته و در متغیر مشخص شده توسط VARIABLE ذخیره می کند.

FlashGetProp (" OBJECT" , " , PROPERTY , VARIABLE ")

با این دستور می توان چگونگی خصوصیات تعیین شده برای شی فلش را دریافت کرد.

پارامتر PROPERTY (خصوصیت) می تواند یکی از عبارت های زیر باشد:

- SCALE مقیاس کلیپ را باز می گرداند.

- BGCOLOR رنگ زمینه کلیپ فلش را به صورت hexadecimal در کانال RGB باز می گرداند.

مثال: (رنگ قرمز = F F0000)

- QUALITY کیفیت کلیپ فلش را باز می گرداند.

- PLAYING در حال اجرا بودن یا نبودن کلیپ فلش را می سنجد.

- MOVIE نام کلیپ فلش را باز می گرداند.

- TOTALFRAMES تعداد فریم های کلیپ فلش را باز می گرداند.

قسمت دوم از پارامتر دوم این دستور برای معین کردن متغیری می باشد که مقدارهای بازگشتی خصوصیات در آن قرار می گیرد. نوع متغیر با توجه به خصوصیت تعیین می شود.

مثال:

فرمانی جهت گرفتن رنگ زمینه:

FlashGetPorp("Flash" ,"BGCOLOR, BGColor\$ ")

فرمانی جهت تشخیص در حال اجرا بودن یا نبودن کلیپ فلش:

FlashGetPorp("Flash" ,"PLAYING,Play\$")

فراخوانی فرامین MMB مستقیماً از Action Script فلش:

جهت برقراری از تباط بین کلیپ فلش و MMB از فرمان fs Command فلش استفاده می شود.

این فرمان ۲ پارامتر دارد: Command و Arguments . چنانچه بخواهید توسط این فرمان، فرمانی را به MMB ارسال کنید پارامتر Command باید همیشه رشته "MMB" باشد و پارامتر دوم Command باید حاوی یک اسکریپت MMB می باشد.

مثال:

fscommand("mmb", "Time \$, "+ MMBCurTimeFull)

این فرمان سبب مقدار دهی متغیر \$ Time توسط فلش می شود.

`fscommand ("mmb","LoadText(\"DirStr\",\""+ DirString +"\")")`

این فرمان سبب فراخوانی فرمان Load Text از MMB می شود و سپس متن DirStr را در MMB پر می کند.

جهت فرار دادن علامت نقل قول در رشته بایستی آن را با کاراکتر Back flash همراه کرد. به این عملیات escaping یک کاراکتر می گویند کاراکترهای متعددی هستند که نمی توان در Action script به تنهایی از آنها استفاده کرد و نشان داد. جدول زیر آنها را نشان می دهد:

Escape sequence	Character
\ b	Backspace کاراکتر (ASCII 8)
\ f	Form – feed کاراکتر (ASCII 12)
\ n	Line- feed کاراکتر (ASCII 10)
\ r	Carrige Return کاراکتر (ASCII 13)
\ t	Tab کاراکتر (ASCII 19)
\ "	کاراکتر "
\ '	کاراکتر '
\\	Back slash
\ 000-\377	یک بایت مشخص شده در مبنای هشت
ff×00 - \ ×\	یک بایت مشخص شده در مبنای 16
\ u 0000 –	یک کاراکتر بین المللی (Uni Code) مشخص شده در مبنای 16
\ uff-ff	

جهت اطلاعات بیشتر پیرامون fs Command از راهنمای برنامه فلش استفاده نمایید

فرمان «ColorPicker»

ColorPicker()

این فرمان کادر محاوره ای انتخاب رنگ را نشان می دهد. این دستور هیچ پارامتر ورودی را قبول نمی کند. به محض اینکه کاربر رنگ را برگزیند و دکمه O k را فشار دهد ثابت CBK-SelColor مقداردهی می شود. لازم به

ذکر است رنگ بر مبنای حالت RGB می باشد. مقدار بازگشتی یک آرایه است که توسط کاما از هم جدا شده اند مثل 100,210,250 که جهت خرد کردن آنها می توان از فرامین رشته استفاده کرد.

فرمان "FontPicker"

این فرمان کادر انتخاب فونت را نمایان می سازد. هنگامی که کاربر فونت مورد نظر خود را برگزیند ثابت CBK_SelFont مقدار دهی می شود. این مقدار یک رشته ساده است که همانند زیر می باشد :

FONTNAME|FONTSTYLE|FONTSIZE|FONTSCRIPT|FONTEFFECT|

FONTNAME : نام فونت

FONTSTYLE : گونه فونت (پر رنگ، ایتالیک و...)

FONTSIZE : اندازه فونت

FONTSCRIPT : شیوه کدگذاری فونت برای زبان

FONTEFFECT : جلوه اعمال شده از قبیل زیر خط دار و...

هر کدام از این مقادیر را می توانید توسط تابع **GetArrayItem** دریافت کنید.

مثال :

```
FontPicker()  
FontParams$=CBK_Font  
If (FontParams$<>'') Then  
    item$[1]=GetArrayItem(FontParams$,1,1)  
    item$[2]=GetArrayItem(FontParams$,1,2)  
    item$[3]=GetArrayItem(FontParams$,1,3)  
    item$[4]=GetArrayItem(FontParams$,1,4)  
    item$[5]=GetArrayItem(FontParams$,1,5)  
    msg$= 'FONTNAME = ' + item$[1] + CHR(13) + CHR(10) + 'FONTSTYLE = ' +  
item$[2] + CHR(13) + CHR(10) + 'FONTSIZE = ' + item$[3] + CHR(13) + CHR(10) +  
'FONTSCRIPT = ' + item$[4] + CHR(13) + CHR(10) + 'FONTEFFECT = ' + item$[5] +  
CHR(13) + CHR(10)  
    Message("Font parameters:", "msg$")  
End
```

در این مثال کادر انتخاب فونت نمایان شده و پس از انتخاب فونت ، مقادیر به صورت پیغام نمایش داده می شود.

فرمان «SaveVariable»

SaveVariable("Name","Variable")

با استفاده از این فرمان می توان محتوی یک متغیر را در رجیستری کامپیوتر ذخیره کرد. رجیستری یک کامپیوتر، یک پایگاه داده ای است که حاوی اطلاعاتی راجع به پیکربندی سیستم می باشد. با استفاده از MMB می توانید یک متغیر را در رجیستری ذخیره نمایید و یا آن را متعاقباً بارگذاری نمایید. مسیر ذخیره یک متغیر با استفاده از Save Variable به صورت زیر است:

HKEY – CURRENT – USER\Software\MMBplayer\Project EXE\RegName

که قسمت Project EXE نام فایل EXE شماست و قسمت RegName نام همان کلید است که می خواهید متغیر را در آن ذخیره نمایید، می توان این نام را در قسمت ProjectSetting -> Style تنظیم نمود. حال که دریافتید که متغیرها در کجا ذخیره می شوند بهتر است راجع به چگونگی ذخیره سازی هم صحبت کنیم. این فرمان ۲ پارامتر دارد که اولی جهت تعیین نام متغیر به هنگام ذخیره در رجیستری می باشد و پارامتر دوم نیز که محتوی متغیر را مشخص می کند و هم می تواند رشته ای باشد و هم عددی. به عنوان مثال می خواهیم متغیر Pass\$ را ذخیره کنیم:

```
Pass$='My Password'  
SaveVariable("Password","Pass$")
```

به همین ترتیب می توانیم برای متغیرهای عددی عمل کنیم:

```
HiScore = 5500  
SaveVariable("User Hi Score" , "HiScore")
```

فرمان « LoadVariable»

LoadVariable("Name","Variable")

این فرمان متغیری را که توسط SaveVariable از قبل ذخیره شده است را بارگذاری می نماید این فرمان هم دارای ۲ پارامتر است که اولی نام متغیر ذخیره شده در رجیستری است و دومی محتوی متغیر را مشخص می کند و هم می تواند رشته ای باشد و هم عددی.

در زیر مثالی برای بارگذاری یک متغیر رشته ای از رجیستری آمده است:

```
LoadVariable("Password","Pass$")
```

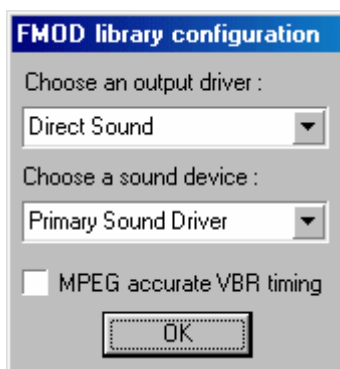
و یک مثال برای متغیر عددی:

```
LoadVariable("User Hi Score","HiScore")
```

فرمان «FMODConfig»

از این فرمان برای نشان دادن کادر پیکربندی کتابخانه FMOD استفاده می شود. این فرمان این قابلیت را فراهم می کند تا کاربر بتواند راه انداز صدای خروجی و ... را تنظیم نماید. همچنین این قابلیت فراهم است که بتوان این تنظیمات را از طریق اسکریپت کنترل کرد. فرمت دستور به شکل مقابل است:

```
FMODConfig("Param1","Param2")
```



چنانچه ما از این دستور بدون هیچ پارامتری استفاده کنیم، کادر تبادلی زیر ظاهر می شود.

اما این فرمان ۲ پارامتر دارد که به وسیله کاما از هم متمایز می شوند، عبارتهای زیر برای قسمت اول پارامتر استفاده می شوند و نوشتن عدد کنار این قسمتها به منزله همان اسامی می باشد.:

1-DirectX Sound(DirectX صدای)

2-Windows Media WaveOut

3-No Sound(بدون صدا)

و برای قسمت دوم پارامتر عبارت های زیر موجودند:

0- MPEG Accurate VBR (Variable Bit Rate) timing off

1- MPEG Accurate VBR (Variable Bit Rate) timing on

برای مثال با استفاده از قسمتهای بالا فرمان به صورت زیر در می آید:

```
FMODConfig("1,0")
```

« BackgroundPlay » فرمان

```
BackgroundPlay("Path","LOOP")
```

این فرمان یک فایل صوتی را بازگشایی کرده و به عنوان آهنگ زمینه اجرا می کند. این فرمان ۲ پارامتر دارد که اولی برای مسیر فایل صوتی است. برای مسیر دهی هم می توان از ماکروها استفاده کرد و هم از مسیر های ثابت. پارامتر دوم مربوط به تنظیم تکرار آهنگ زمینه در حین اجرای برنامه می باشد.

```
BackgroundPlay("<ScrDir >\Moon.MP3","LOOP")
```

فرمان بالا فایل صوتی Moon.MP3 را به عنوان آهنگ زمینه مکرراً اجرا می کند.

« BackgroundPause » فرمان

```
BackgroundPause()
```

این فرمان سبب ایجاد وقفه در اجرای آهنگ زمینه می شود و این فرمان هیچ پارامتری ندارد.

« BackgroundStop » فرمان

BackgroundStop()

این فرمان اجرای آهنگ زمینه را به کل لغو می کند، این فرمان هیچ پارامتری ندارد.

فرمان « VideoLoad »

VideoLoad("Object", "Path")

از این فرمان برای بارگذاری یک فایل ویدئویی استفاده می شود. این فرمان ۲ پارامتر دارد اولی جهت معین کردن نام شی ویدئویی و دومی جهت مسیر فایل ویدئویی که مسیر می تواند هم ثابت و هم پویا باشد. چنانچه در این فرمان پارامتر دوم یعنی مسیر فایل مشخص نباشد، به هنگام استفاده از این دستور کادر بازگشایی فایل ویدئویی نمایان می شود.

فرمان « VideoPlay »

VideoPlay("Object")

این فرمان فایل ویدئویی بار گذاری شده را اجرا می کند. در قسمت پارامتر این فرمان بایستی نام شی ویدئویی را مشخص نماییم.

فرمان « VideoPause »

VideoPause("Object")

از این فرمان جهت ایجاد وقفه در اجرای فایل ویدئویی استفاده می شود. چنانچه فایل ویدئویی در حال توقف باشد این فرمان سبب اجرای مجدد فایل ویدئویی می شود. این فرمان تنها یک پارامتر دارد و آن هم تنها نام شی ویدئویی می باشد.

فرمان « VideoRewind »

VideoRewind("Object","Time/Frame,RELATIVE")

این فرمان سبب عقب برگرداندن فایل ویدئویی می شود. می توانیم مقدار بازگشت را تعیین نماییم، اما این مقدار بستگی به تنظیمات فایل ویدئویی دارد که این تنظیمات توسط فرمان Video Param تعیین شده و بر نوع میلی ثانیه و فریم است. این فرمان ۲ پارامتر دارد یکی جهت مشخص کردن نام شی و دیگری جهت تنظیم مقدار بازگشت. چنانچه مایل باشیم که عقب گرد نسبی صورت گیرد می توانیم کلمه RELATIVE را پس از مقدار بازگشت بیاوریم، همانند مثال زیر:

VideoRewind("Video",-30,RELATIVE")

فرمان « VideoSpeed »

VideoSpeed("Object","Time/Frame,RELATIVE")

این فرمان سرعت اجرای فایل را که توسط فرمان VideoLoad بارگذاری شده است را تنظیم می کند. با استفاده از اولین پارامتر این فرمان می توان نام شی ویدئویی مورد نظر را مشخص کرد. دومین پارامتر سرعت اجرا را معین می کند که این پارامتر هم می تواند عدد ثابت باشد و هم متغیر عددی. البته باید توجه داشت که سرعت می تواند بین محدودی 2000 تا 0 باشد.

مقدار پیش فرض سرعت 1000 است و این بدان معناست که سرعت بالای 1000 سبب افزایش سرعت نسبت به حالت معمول و سرعت پایین 1000 سبب کاهش سرعت اجرا نسبت به حالت معمول می شود. جهت استفاده نسبی از تنظیمات جاری می توانیم از عبارت RELATIVE بهره ببریم. شایان ذکر است که می توان با استفاده از شی Matrix سرعت جاری را به صورت گرافیکی در مقیاس درصدی یا هر واحد دیگر بیان کرد.

مثال:

VideoSpeed("Video", "2000,RELATIVE")

فرمان « VideoParam »

VideoParam("Object","Switch")

هنگامی که بخواهیم یکسری پارامتر اضافه تر از قبیل تکرار، قطع صدا و ... را تنظیم نماییم از این فرمان استفاده می کنیم. همانند سایرین اولین پارامتر این فرمان برای معین کردن نام شی ویدئویی است. و در قسمت پارامتر دوم از کلمات زیر استفاده می شود:

FULLSCREEN: این کلمه حالت تمام صفحه را (و بر عکس آن را) فراهم می کند.

LOOP: این فرمان جهت ایجاد تکرار (یا برعکس) می باشد که خود ۲ حالت دارد:

۱- LOOP=ON که سبب ایجاد تکرار می شود.

۲- LOOP=OFF که سبب لغو تکرار فایل ویدئویی می شود.

MUTE: این کلمه سبب قطع صدای فایل ویدئویی (یا برعکس) می شود ک خود دارای ۲ قسمت است:

۱-MUTE=ON که سبب قطع صدا می شود.

۲-MUTE=OFF که سبب لغو قطعی صدا می شود.

MODE: این حالت تنظیم می کند که آیا ویدئو بر اساس زمان اجرا شود یا بر اساس فریم، با این تفاسیر ۲ حالت وجود دارد:

۱- MODE=TIME بر اساس زمان

۲- MODE=FRAME بر اساس فریم

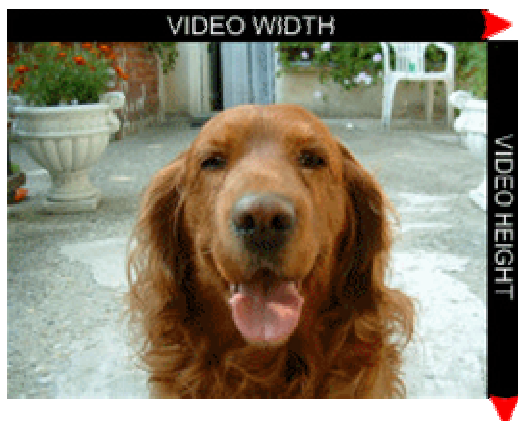
نکته: ماکرو CBK-VTime تنها در حالتی کاراست که MODE به MODE=TIME تنظیم شده باشد. -CBK-Vframe هم تنها در صورت وجود عبارت MODE=FRAME فعال می شود.

باید گفت که MODE پیش فرض مد زمان (TIME) است. چنانچه بلافاصله پس از بارگذاری فایل ویدئویی CBK-Vframe عدد 0 را نشان داد این بدان معناست که فایل ویدئویی ازمد فریم پشتیبانی نمی کند و در نتیجه باید گفت که نمی توان از فرامین جلو بردن / عقب بردن فایل ویدئویی استفاده کرد.

فرمان « VideoScale »

VideoScale("Object","X,Y")

این فرمان اندازه پنجره ویدئو را تعیین می کند. اولین پارامتر این دستور برای تعیین نام شی ویدئویی است. دومین پارامتر جهت تعیین عرض و ارتفاع می باشد.



مثال:

```
VideoScale("Video", "50,50")
```

«فرامین MCI»

فرامین MCI برای کنترل اجرای فایل های صوتی و تصویری می باشند. یادگیری این قسمت تا حدودی مشکل است، چرا که مقوله MCI مبحثی با جزئیات زیاد است و استفاده از آن تنها برای کاربران حرفه ای توصیه می شود. البته باید گفت که امکانات موجود در MMB برای کنترل فایل های صوتی و تصویری در حد مطلوبی است و تقریباً نیازی به استفاده از این فرامین ندارید. اما به هر ترتیب در این قسمت قصد داریم تنها به ذکر چند فرمان مهم و پر کاربرد MCI بپردازیم.

```
MCICommand("status cdaudio number of tracks")
```

فرمان بالا تعداد شیار (Track) های یک CD صوتی را باز می گرداند، بله درست حدس زده اید، این دستور یک تابع نیز هست بدین معنی که یک مقدار خروجی تولید می کند به طور کلی فرامین MCI خروجی خود را به متغیر MCIResult نسبت می دهند.

مثال:

```
MCICommand("status cdaudio number of tracks")
```

```
DisplayValue("Text", "MCIResult")
```

```
MCICommand("play cdaudio from # to #")
```

این فرمان بر اساس شماره های وارد شده Track یا Track های مربوطه از CD صوتی را اجرا می کند.

مثال:

```
MCICommand("play cdaudio from 3 to 4")
```

```
MCICommand("status cdaudio length track #")
```

تابع مذکور طول Track مشخص شده را از CD صوتی باز می گرداند.

مثال:

```
MCICommand("status cdaudio length track 2")  
DisplayValue("Text", "MCIResult")
```

```
MCICommand("play cdaudio")
```

CD صوتی را اجرا می کند.

```
MCICommand("stop cdaudio")
```

اجرای CD صوتی را متوقف می کند.

```
MCICommand("status cdaudio mode")
```

تابع بالا وضعیت اجرایی CD صوتی را بررسی می کند. چنانچه CD صوتی در حال اجرا باشد تابع مقدار 2- را باز می گرداند. در غیر اینصورت مقدار 1- باز گردانده می شود.

مثال:

```
MCICommand("status cdaudio mode")
If (MCIResult== -2) Then
  Message("CD is now playing", "")
Else
  MCIResult== -1
  Message("CD is not playing")
End
```

MCICommand("set cdaudio door open")

فرمان بالا در درایو CD را باز می کند.

MCICommand("set cddaudio door closed")

این فرمان در درایو CD را می بندد.

برای بررسی سایر فرامین MCI به راهنمای MCI و مثالهای درون CD همراه کتاب مراجعه نمایید.

توابع رشته ها^۱

به هنگام کار با رشته ها نیاز است تا اعمال خاصی از قبیل تبدیل آنها به حروف بزرگ و بالعکس ، اندازه گیری آنها و ... صورت گیرد که خوشبختانه MMB توابعی سودمند جهت کار با رشته ها در اختیار برنامه نویس قرار می دهد.

CHR(Value)

این تابع براساس عدد وارد شده معادل اسکمی آن را باز می گرداند.

مثال:

```
string$=CHR(169) + 'Rasane 2003'  
LoadText("Text","string$")
```

در نهایت محتوی متغیر string\$ صورت زیر خواهد شد:

© Rasane 2003

ORD(character\$)

این تابع عدد معادل کاراکتر مورد نظر را بر اساس جدول اسکمی باز می گرداند.

مثال:

```
character$='#'  
RetVal=ORD(character$)
```

نتیجه کد بالا عدد ۳۵ خواهد بود.

LEN(string\$)

این تابع طول رشته مورد نظر را باز می گرداند.

۱- چنانچه به توابع بیشتری در زمینه کار با رشته ها احتیاج دارید ، می توانید از Plugy Plug-In استفاده نمایید.

مثال:

```
LoadText("string$","c:\test.txt")
RetVal=LEN(string$)
```

نتیجه کد بالا عدد ۱۱ خواهد بود.

LOW(string\$)

این تابع حروف رشته را به حروف کوچک تبدیل می کند.

مثال:

```
string$='ThIs TeXt WiLL bE CoNvErTED to LoWeRcAsE'
RetString$=LOW(string$)
```

نهایتاً عبارت به صورت this text will be converted to lowercase در می آید.

UPP(string\$)

این تابع حروف رشته را به حروف بزرگ تبدیل می کند.

مثال:

```
string$='ThIs TeXt WiLL bE CoNvErTED to UppErCasE'
RetString$=UPP(string$)
```

نهایتاً عبارت به صورت THIS TEXT WILL BE CONVERTED TO UPPERCASE در می آید.

POS(Substring\$,String\$)

این تابع به جستجوی زیر رشته (Substring) در رشته (String) داده شده می پردازد. چنانچه زیر رشته در رشته موجود باشد این تابع موقعیت اولین حرف زیررشته را باز می گرداند. اگر زیر رشته در رشته داده شده موجود نباشد تابع مقدار صفر را باز می گرداند.

مثال:

```
string$='Have a nice day!'
substring$='nice'
RetVal=POS(substring$,string$)
```

کد بالا عدد ۸ را باز می گرداند.

توجه نمایید که تابع بالا حساس به حروف می باشد و قبل از استفاده از این تابع بایستی حروف رشته مورد نظر را به حروف کوچک تبدیل کنیم.

NOL(FileName\$)

تابع بالا تعداد خطوط فایل متنی بارگذاری شده را باز می گرداند. چنانچه هیچ فایلی برگزیده نشده باشد مقدار بازگشتی این تابع صفر خواهد بود.

مثال:

```
OpenFile("txt Files (*.txt)|*.txt|All Files|*.*||", "*.txt")
If (OpenFile$<>'') Then
    RetVal=NOL(OpenFile$)
End
```

کد بالا تعداد خطوط فایل متنی بارگذاری شده را باز می گرداند.

StrCopy(string\$,index,count)

این تابع براساس مقادیر ورودی، رشته ای را از رشته اصلی باز می گرداند. **index** شروع را مشخص می کند و **count** طول رشته را معین می کند. اگر عددی که شروع را مشخص می کند از طول رشته بیشتر باشد این تابع هیچ رشته ای را باز نمی گرداند. همچنین اگر عددی که طول رشته را معین می کند از طول رشته بیشتر باشد، تابع تنها رشته شروع از **index** تا آخر رشته اصلی را باز می گرداند.

مثال:

```
string$='Have a nice day!'
ReturnExt$=StrCopy(string$,1,4)
```

کد بالا کلمه **Have** را باز می گرداند.

StrDel(string\$,index,count)

تابع بالا براساس مقادیر داده شده مقداری از رشته اصلی را کسر می کند و باقی مانده رشته اصلی را باز می گرداند. index شروع را مشخص می کند و count طول را مشخص می کند. چنانچه index از طول رشته بزرگتر باشد این تابع هیچ کاراکتری را از رشته اصلی حذف نمی کند.

مثال:

```
string$='Have a nice day!'  
ReturnExt$=StrDel(string$,1,5)
```

کد بالا عبارت **a nice day!** را باز می گرداند.

StrIns(SourceStr\$, DestStr\$, Index)

تابع گفته شده رشته مقصد را با رشته اصلی در موقعیت بیان شده ادغام می کند. SourceStr\$ رشته اصلی را بیان می کند و DestStr\$ رشته مقصد را معین می کند. index نیز برای تعیین موقعیت می باشد.

مثال:

```
sourcestr$='Have a nice day!'  
deststr$=' too'  
count=LEN(sourcestr$)  
ReturnStr$=StrIns(sourcestr$,deststr$,count)
```

کد مثال بالا عبارت **Have a nice day too!** را باز می گرداند.

StrGet(string\$,Int)

این تابع بر اساس مقدار (Int) داده شده کاراکتر متناظرش را باز می گرداند.

مثال:

```
string$='Have a nice day!'  
count=LEN(string$)  
ReturnStr$=StrGet(string$,count)
```

کد مثال بالا کاراکتر **!** را باز می گرداند.

StrSet(String\$, Int,C\$)

این تابع بر اساس مقادیر داده شده رشته ای جدید را در رشته اصلی جایگزین می کند. String\$ رشته اصلی را معین می کند، Int، موقعیت را بیان می کند و C\$ رشته ای را که بایستی جایگزین شود مشخص می کند.

مثال:

```
string$='Have a nice day!'
count=LEN(string$)
c$='!!!'
ReturnStr$=StrSet(string$,count,c$)
```

کد بالا در نهایت عبارت Have a nice day!!! را باز می گرداند.

StrOfChar(C\$, Int)

این تابع بر اساس مقدار (Int) معین شده همان تعداد از کاراکتر مشخص شده را باز می گرداند.

مثال:

```
c$='#'
ReturnStr$=StrOfChar(c$,5)
```

مثال بالا ##### را باز می گرداند.

StrChange(String\$, FromStr\$, ToStr\$)

این تابع بر اساس مقادیر داده شده عبارتی را جایگزین عبارتی مشخص شده از رشته اصلی می کند. String\$ رشته اصلی را مشخص می کند، FromStr\$ عبارتی را که باید جایگزین شود را معین می کند و ToStr\$ عبارت جدید را مشخص می کند.

مثال:

```
string$='My name is ali'
fromstr$='ali'
tostr$='reza'
ReturnStr$=StrChange(string$,fromstr$,tostr$)
```

مثال بالا در نهایت جمله My name is reza را باز می گرداند.

StrToFile(FileName\$, String\$, Append, LineFeed)

تابع بالا رشته مشخص شده را در فایلی و یا در انتهای فایلی ذخیره می کند. `FileName$` فایل مقصد را معین می کند، `String$` رشته ای که قرار است ذخیره شود را معین می کند، `Append` شرایط ذخیره کردن رشته در فایل را مشخص میکند که چنانچه به جای آن `TRUE` را قرار دهیم رشته مورد نظر در انتهای فایل مشخص شده و پس از آخرین کاراکتر ذخیره می شود. اما اگر فایل مشخص شده از قبل موجود نباشد این فرمان فایل را ایجاد می کند. اگر `LineFeed` و `Append` هر دو `TRUE` باشند تابع، رشته مورد نظر را در خط بعد از آخرین خط فایل مورد نظر ذخیره می کند. اما چنانچه هر دو عبارت `Append` و `LineFeed`، `FALSE` باشد تابع رشته مورد نظر را در فایلی جدید و یا بر روی فایل موجود ذخیره می کند.

مثال:

```
file$='c:\temp\test.txt'  
string$= 'this string will be append to text file'  
ReturnVal=StrToFile(file$,string$,TRUE,FALSE)
```

کد بالا رشته مورد نظر را در فایل `test.txt` ذخیره می کند.

StrFromFile(FileName\$, FromLine, NumOfLines)

این تابع رشته یا رشته ها از فایل مورد نظر بارگذاری می کند. `FileName$` برای مشخص کردن فایل متنی می باشد، `FromLine` شماره خطی را که بایستی عمل بارگذاری از آنجا صورت گیرد مشخص می کند و `NumOfLine` تعداد خطوطی را که باید بارگذاری شوند معین می کند. اگر به جای `NumOfLine` از عدد `-1` استفاده کنیم تمام خطوط فایل متنی بارگذاری می شود. چنانچه `NumOfLine = -1` و `FromLine` بزرگتر از صفر باشد تابع تمام خطوط پس از خط معین شده را باز می گرداند.

مثال:

```
file$='c:\temp\test.txt'  
fromline= 1  
numoflines=10  
ReturnStr$=StrFromFile(file$,fromline,numoflines)
```

کد مثال بالا ۱۰ خط ابتدایی فایل `test.txt` را باز می گرداند.

ExtractExt(FileName\$)

تابع بالا پسوند فایل مشخص شده را باز می گرداند.

ExtractDir(FileName\$)

این تابع پوشه فایل مورد نظر را باز می گرداند.

ExtractName(FileName\$)

این تابع نام فایل مشخص شده را باز می گرداند.

ExtractDrive(FileName\$)

تابع بالا نام درایو فایل مورد نظر را باز می گرداند.

مثال:

```
Path$='c:\MyFiles\text.txt'  
ReturnExt$=ExtractExt(Path$)  
ReturnDir$=ExtractDir(Path$)  
ReturnName$=ExtractName(Path$)  
ReturnDrive$=ExtractDrive(Path$
```

مثال:

کد زیر عبارتی را دریافت می کند و ترتیب قرار گیری حروف آنرا تصادفا تغییر می دهد.

```
StrNew$=''  
String$='House'  
StrLen=Len(String$)  
For i=StrLen To 1  
  RndNum=RND(StrLen-1)+1  
  GetLetter$=StrCopy(String$, RndNum, 1)  
  StrNew$=StrDel(String$, RndNum, 1)  
  StrLen=Len(String$)  
Next i  
LoadText("Text", "StrNew$")
```

البته برای مشاهده نهایی بایستی یک شی Text در صفحه قرار دهید.

PlugIn

Multimedia Builder

PlugIn چیست؟ ←

کنترل رویداد Event Handling ←

Tweak PlugIn ←

Freeplay PlugIn ←

Slideshow PlugIn ←

Eval PlugIn ←

MMBDLL PlugIn ←



تا به حال عملاً شما قسمت های عمده MMB را فرا گرفته اید و تقریباً آماده ی تولید پروژه های معمولی می باشید، اما ممکن است در ابتدای کار هیچ محدودیتی را به جهت تولید فرامین و قابلیت های ارائه شده احساس نکنید. اما هر چه که پیش بروید و تصمیم به تولید پروژه های بزرگتر بگیرید احتمالاً به محدودیتهایی بر خواهید خورد. اما نگران نباشید چاره این کار از قبل اندیشیده شده است و آن چیزی نیست جز PlugIn، قابلیتی که ویژگی هایی اضافی بر سازمان را به بدنه اصلی MMB اضافه می نماید. هم اکنون به مدد PlugIn راه پیش روی خود را برای تولید پروژه مورد نظر هموارتر احساس می کنید. در این بخش قصد داریم به بررسی نحوه عملکرد PlugIn های MMB بپردازیم. در ادامه چند PlugIn معروف این برنامه را مورد کنکاش قرار خواهیم داد.

Plug-In چیست؟

اکثر برنامه‌های امروزی این قابلیت را فراهم می‌کنند تا کاربر بتواند از Plug-In ها استفاده نماید. برای مدت‌های مدیدی این امکان فراهم نبود تا یک کاربر بتواند تعدادی توابع اضافه بر سازمان برنامه در اختیار داشته باشد و می‌بایست تا نسخه بعدی برنامه صبر می‌کرد. اما پس از چندی اشخاصی این قاعده را شکستند و این آرزوی کاربران را برآورده ساختند. پس از آن با تدوین استانداردها برنامه‌نویسان و توسعه دهندگان اجازه یافتند تا براساس ضوابطی توابع دلخواه خود را در صورت امکان به بدنه برنامه اصلی اضافه نمایند. به همین دلیل این امکانات و توابع اضافه نام plug-in را به خود گرفت. امروزه می‌توانیم ردپای plug-In را در اکثر برنامه‌ها از قبیل برنامه‌های طراحی و ویرایشی و ... ببینیم. اما حال ببینیم plug-In ها دقیقاً چه هستند و چگونه کار می‌کند؟

مطمئناً همه‌ی شما ایستگاههای فضایی را دیده‌اید که حول زمین در حال چرخش هستید. حالا تصور کنید که MMB ساختار مرکزی این ایستگاه فضایی است.



البته شاید شکل بالا شبیه یک ایستگاه فضایی کامل نباشد چرا که سلولهای خورشیدی ندارد. اما به هر حال با تصویری که از شکل بالا پیدا کردید فرض کنید که می‌توانید در این ایستگاه فضایی به ادامه حیات و فعالیت بپردازید.

اما پس از مدتی احساس خواهید کرد که به امکاناتی اضافه احتیاج دارید امکاناتی از قبیل اسباب راحتی و... اما نگران نباشید این احساس نیاز از قبل پیش‌بینی شده و تنها کافیست که وسایل را تهیه و با اتصال آنها به ایستگاه فضیایمان را گسترش دهیم. حالا با استفاده از قابلیت توسعه توانستید فضایتان و قدرتان را بهبود بخشید اما اگر بخواهیم به واقعیت نزدیکتر شویم و قدرتمان را در MMB افزایش دهیم دقیقاً باید چه کنیم؟

لازم نیست نگران باشید تنها کافیست برنامه MMB و plug-In های مورد نظر را در اختیار داشته باشید. اما حتماً از خود می‌پرسید plug-In های MMB چگونه هستند؟ بله plug-In های MMB تنها فایل‌های ساده با پسوند DLL هستند که امکان دارد در حالت عادی پنهان شده (Hidden) باشند. (می‌توانید از قسمت Folder option ویندوز آنها را در صورت مخفی بودن نمایان سازید. همچنین می‌توانید در حین اجرای یک برنامه حاوی plug-In، plug-In آن را از شاخه temp ویندوز بیابید). برای مثال فایل‌های misc.dll, tweak.dll, posxy.dll یکسری از plug-In های موجود هستند. اما این plug-In ها را از کجا باید تهیه کنیم؟ پاسخ ساده است شما می‌توانید با استفاده از سایت‌های زیر اقدام به تهیه آنها نمایید:

۱- <http://www.mmbstation.com> (سایت شخصی مؤلف)

۲- <http://www.bokzy.com>

۳- <http://www.Advanced-Microtechnologies.com>

۴- <http://www.Yomoweb.com>

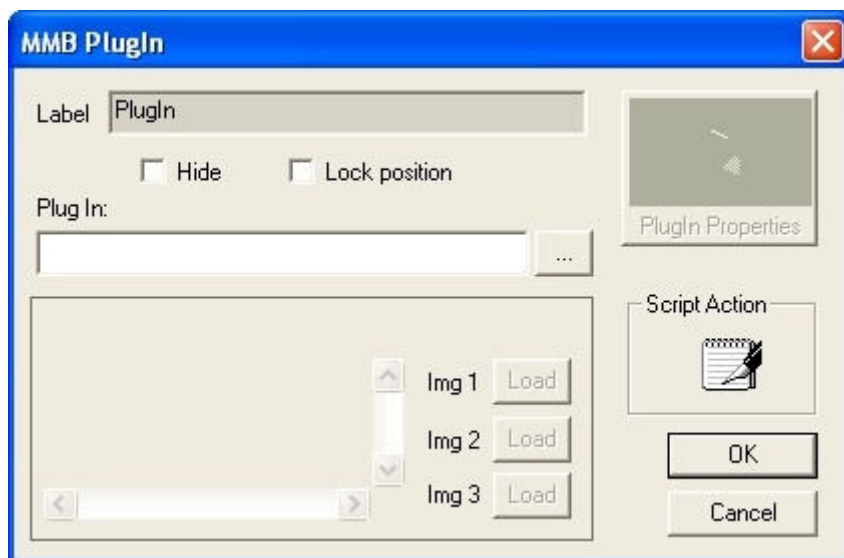
۵- <http://www.B8dae9.Hestultra.com>


۶- <http://www.squashproduction.com>

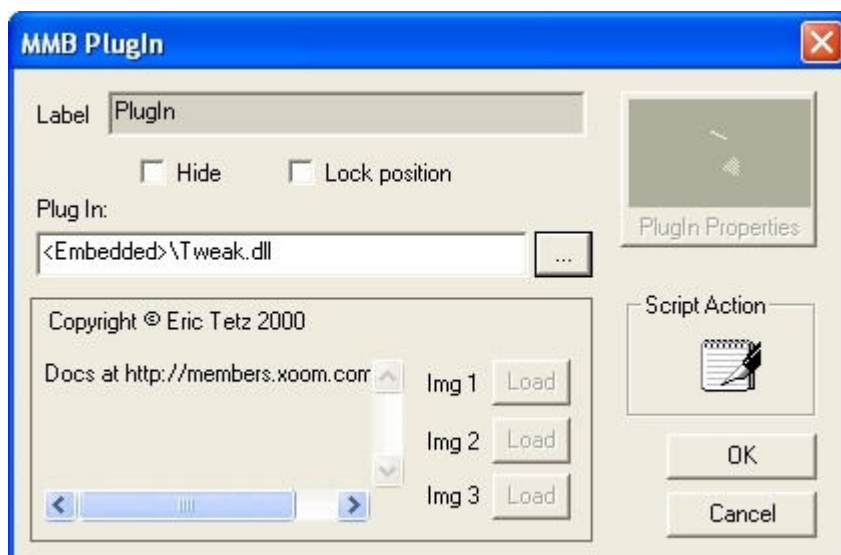
توجه: ممکن است به مرور زمان و به خاطر مشکلاتی لینک‌های بالا کار نکند بنابراین می‌توانید از موتورهای جستجو استفاده نمایید.

بسیار خب پس از اینکه plug-In های مورد نیاز را جمع‌آوری کردید وقت استفاده از آنهاست. ابتدا قبل از هرکاری بایستی یک پنجره در صفحه پروژه برای plug-in بکشیم تا در ادامه این پنجره به عنوان نگهدارنده فایل plug-In ما باشد. از منوی Object با انتخاب قسمت plug-In پنجره مورد نظر را می‌کشیم.

برخی از plug-In ها هستند که به صورت بصری عمل می‌کنند. به عنوان مثال plug-In هایی که جلوه‌هایی نظیر بارش برف و باران و حرکت ستارگان را ایجاد می‌کنند. بنابراین این plug-In ها نیاز به پنجره‌ای مناسب و قابل دید دارند، اما در کنار همین plug-In های بصری تعداد دیگری plug-In هم هستند که بصری نیستند و نیاز به پنجره‌ای بزرگ و قابل دید ندارند بلکه فقط وجود پنجره‌ای کوچک در هر جایی برای آنها کافیست. پس از اینکه پنجره را ایجاد کردید بر روی آن ۲ بار کلیک نمایید تا پنجره‌ای همانند شکل ظاهر شود.

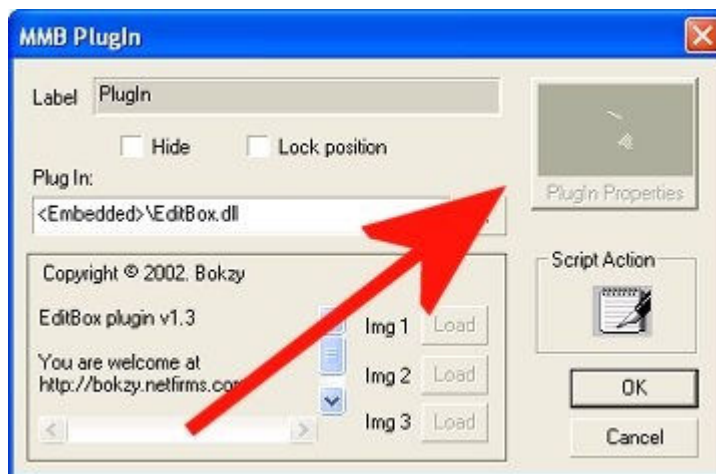


همانند سایر پنجره‌های خصوصیت برای اشیاء دیگر گزینه‌های **Hide** و **Lock position** عمل مشابهی را انجام می‌دهند و **label** (برجسب) هم نام شیء **Plug-In** را مشخص می‌نماید که باید این نام را به خاطر بسپارید. در وسط این پنجره کادر ورودی هست که کنار آن دکمه‌ای قرار گرفته که توسط  این دکمه کادری تبدیلی جهت مسیره‌ی فایل **plug-In** ظاهر می‌شود. خوب پس چرا منتظرید؟! دکمه **...** را فشار دهید... اگر هیچ **plug-In** ای را جهت مسیره‌ی دهی ندارید به زیرشاخه **plug-In** از پوشه نصب برنامه **MMB** بروید و در آنجا یک **Plug-In** بصری به نام **Tenblobs.dll** را که همراه برنامه **MMB** ارائه شده است را خواهید یافت. بلافاصله پس از اینکه **Plug-In** را برگزیدید، پیغامی مبنی بر اینکه آیا میخواهید **Plug-In** را الحاق نمایید؟ ظاهر میشود. به آن پاسخ مثبت بدهید. دیگر احتیاجی نیست که نگران این فایل پس از اتمام پروژه‌تان باشید زیرا **MMB** تمام فایل‌های الحاق شده را به همراه فایل نهایی بسته‌بندی می‌کند. پس از الحاق یک **Plug-In** پنجره خصوصیت به صورت شکل درمی‌آید.



باتوجه به شکل مسیر منتهی به فایل Plug-In دارای ماکروی <Embedded> است که پیش‌تر درباره این ماکرو بحث کردیم. در زیر این کادرمسیر، قسمتی است که حاوی اطلاعاتی از قبیل نام مؤلف Plug-In، پست الکترونیکی، سایت و ... مؤلف میباشد.

اکنون دکمه‌ی ok را فشار دهید تا عمل نصب Plug-In تمام شود. در اینجا است که پس از یکبار اجرای برنامه احساس می‌کنید که چیزی کم است، بله درست فهمیدید بر اساس سایر Plug-In های برنامه‌های دیگر شما به این



تصور رسیده‌اید. اما اگر Plug-In شما بصری باشد در کادر خصوصیت Plug-In قسمتی در بالا سمت راست که در شکل نشان داده شده فعال می‌شود که میتوانید به آنجا بروید و تنظیمات موجود را تغییر دهید.

اما خصوصیات و کنترل Plug-In های غیر بصری را چگونه می توان در دست گرفت؟ در اینجا است که فرامین به کمک ما می آیند. فرامینی که در دل اسکریپت های MMB جا خوش کرده اند. اما اجازه دهید که ببینیم در پشت پرده چه خبرست؟ همان طور که گفتیم فایل های Plug-In برنامه MMB، DLL هستند که DLL مخفف عبارت Dynamic link library (کتابخانه اتصال پویا) است به این معنا که DLLها مجموعه ای از توابع و فرامین هستند. پس با این تفاسیر ارتباطی پویا بین MMB و Plug-In ها برقرار است. این ارتباط می تواند به ۳ صورت زیر باشد:

۱- تنظیم متغیر در MMB و دریافت متغیر عددی توسط Plug-In

۲- تنظیم متغیر در MMB و دریافت متغیر رشته ای توسط Plug-In

۳- اجرای فرمانی خاص که ممکن است این فرمان خروجی به همراه داشته باشد.

برای استفاده از Plug-In ها بایستی با مفاهیم متغیرها و نحوه استفاده از آنها آشنا باشید. چنانچه این گونه نیست از ادامه خواندن این قسمت تا زمانیکه مفاهیم متغیرها را فراگیرید پرهیز نمایید. خب گذشته از اینها صورت پیشرفته تر ۳ حالت قبل به صورت زیر در می آید:

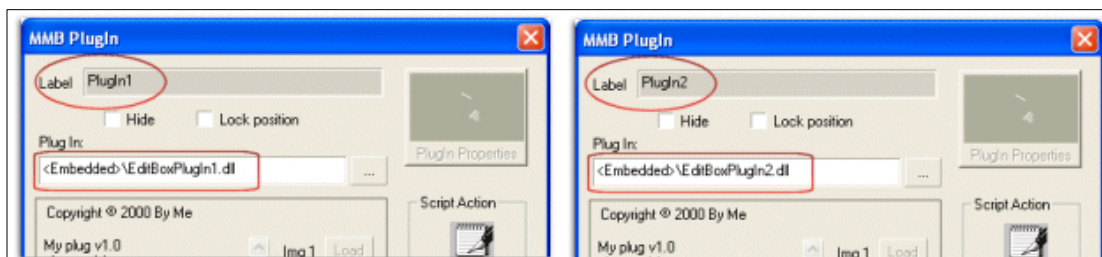
۱- تنظیم پارامترها بر اساس متغیرها (همیشه لازم نیست)

۲- اجرای فرامین Plug-In بر اساس پارامترها

۳- دریافت متغیرهای خروجی پس از اجرای فرامین (همیشه اینطور نیست)

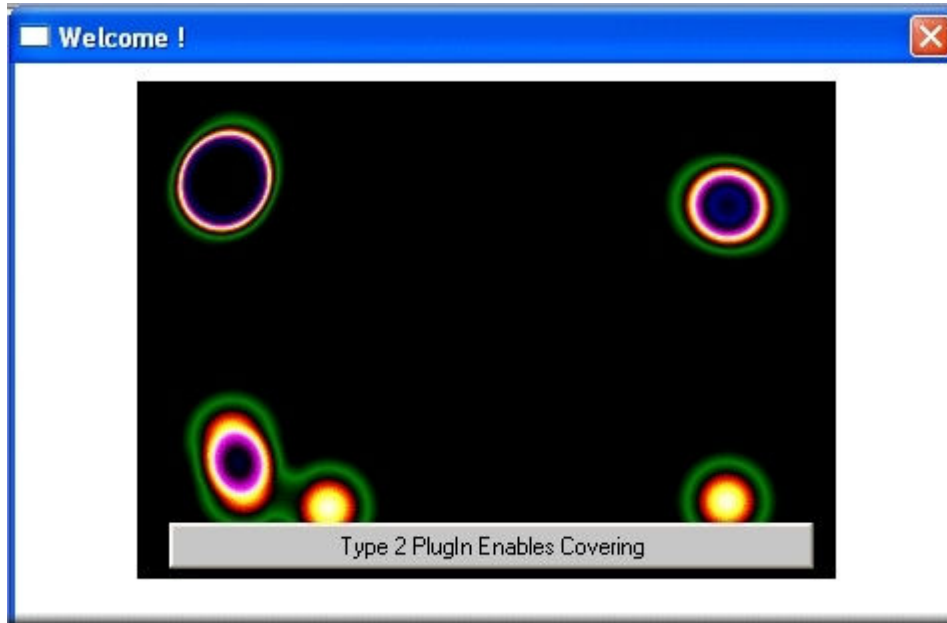
۳ حالت گفته شده زیربنای درک عملکرد Plug-In ها می باشند. همان طور که قبلاً گفته شد Plug-In های MMB بر ۲ نوع هستند: ۱- نوع غیربصری و قابل کنترل توسط فرامین ۲- نوع بصری

غالب Plug-In های MMB (نزدیک به ۹۸٪) از نوع اول هستند که محدوده فعالیت آنها در قسمت پنجره تعبیه شده است. باقی فعالیتها توسط فرامین داخل Plug-In صورت می گیرد که یا منجر به تغییر حالت Plug-In یا ارسال خروجی به متغیرها می شود. این نکته حائز اهمیت است که Plug-In های از نوع اول تقریباً مستقل از MMB عمل می کند و بر سایر اشیاء حاضر در پروژه هیچ تاثیر مستقیمی اعمال نمی کنند. عدم تعامل با سایر اشیاء ضعفی برای MMB شناخته می شود که شاید در نسخه های بعدی برطرف گردد. نکته ی دیگری که در خصوص نوع اول باید گفت اینست که نمی توان از یک Plug-In در ۲ پنجره استفاده کرد در مفهومی عام تر نمی توان



آنها را به اشتراک گذاشت. راه حل این نقیصه استفاده از ۲ Plug-In مشابه اما با نامهای مختلف است.

همانطور که شکل بالا نشان می دهد دو Plug-In مشابه با نامهای مختلف داریم که این نامها در جای خود مهم است، چرا که در ادامه توسط همین نامها و به مدد از فرامین از Plug-In ها استفاده می کنیم. اما نوع دوم بصری هستند که قبلاً راجع به آنها صحبت کردیم. شکل زیر یک نمونه از Plug-In های بصری است.



بگذارید پس از شرح قسمتهای مهم قبل به جمع بندی مطالبی که تا کنون آموختیم بپردازیم:

- Plug-In ها همه جا هستند.

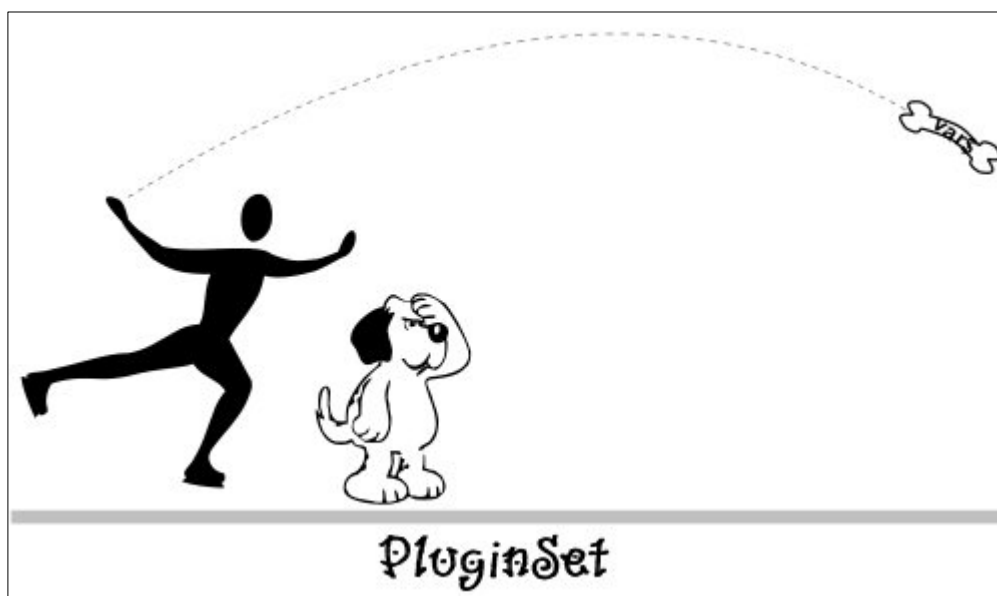
- قسمت هایی اضافه بر سازمان هستند.

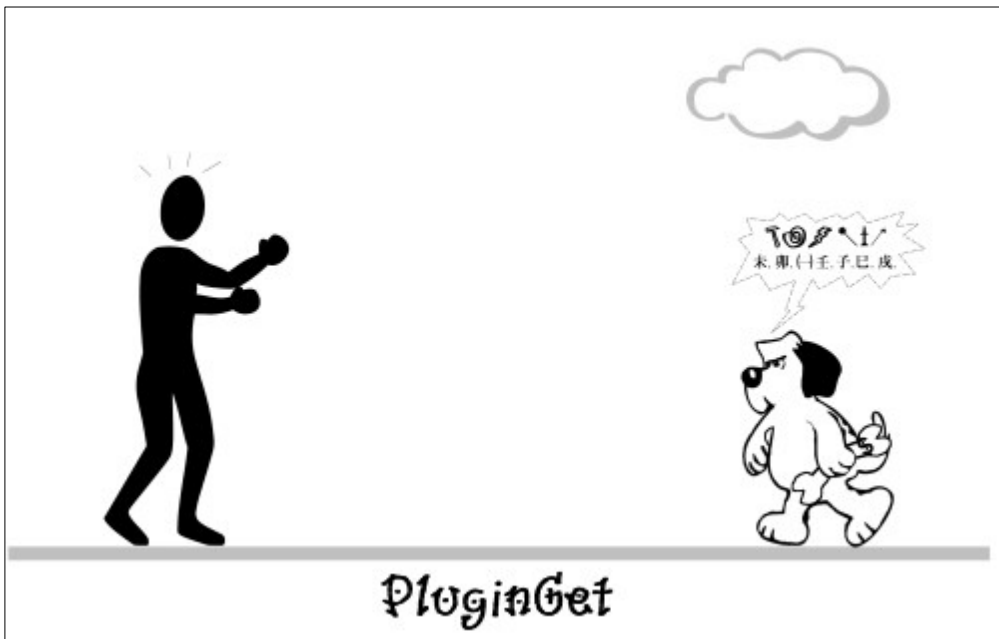
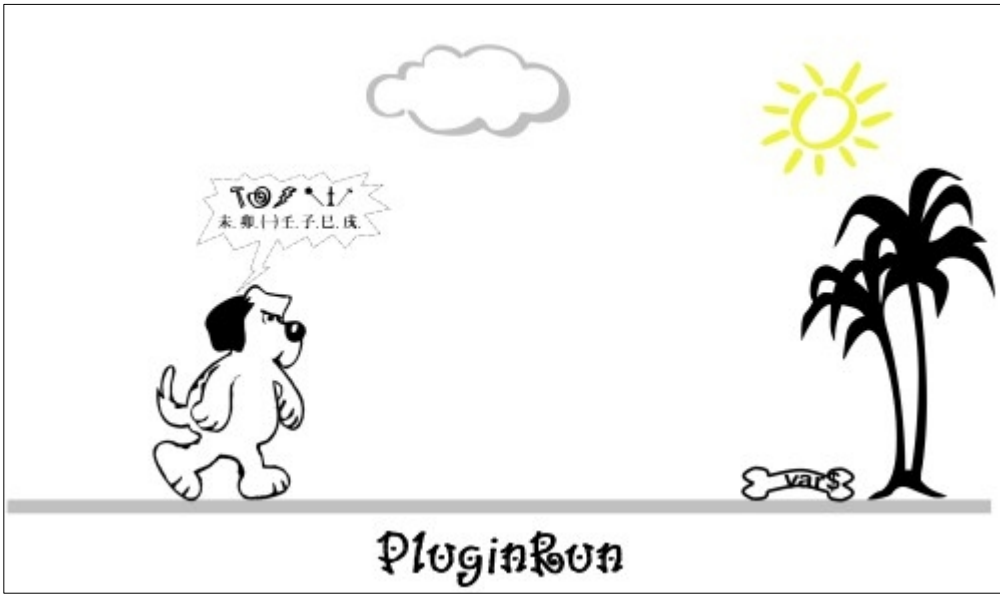
- هدف Plug-In ها افزایش قابلیت برنامه هاست.

خب دیگر تئوری بس است. به جاست که تمرین را شروع کنیم. گفتیم که تنظیم خصوصیات Plug-In های نوع ۱ توسط اسکریپت نویسی انجام می شود. در زبان اسکریپت نویسی MMB، ۳ فرمان مرتبط با Plug-In ها وجود دارد که شما از آنها جهت ایجاد ارتباط بین MMB و Plug-In استفاده می کنید. این فرامین به ترتیب در جدول زیر بحث شده اند.

PlugInSet	این فرمان جهت تنظیم پارامتر برای فرامین Plug-In می‌باشد. این فرمان پارامترها را که همان متغیرهای عددی و رشته‌ای هستند را به فایل Plug-In ارسال می‌کند و Plug-In آنها را ذخیره و بررسی میکند. هر بار استفاده از این دستور با یک متغیر که محتوایش تغییر کرده سبب جایگزینی محتوی جدید در فایل Plug-In می‌شود.
PlugInRun	این فرمان توابع و فرامین تعبیه شده در فایل Plug-In را براساس پارامترهای تنظیم شده اجرا میکند. چنانچه بخواهید نام فرامین موجود در Plug-In را بدانید به راهنمای همراه Plug-In مراجعه کنید. البته همیشه Plug-In ها احتیاج به پارامتر ورودی ندارند و یا در حالاتی هیچ مقدار خروجی تولید نمی‌کنند و تنها سبب تغییر حالت Plug-In و انجام سایر فعالیت‌ها می‌شود.
PlugInGet	این فرمان وظیفه دریافت مقدار خروجی که با استفاده از فرمان PlugInRun تولید شده است را برعهده دارد. پس طبیعتاً این فرمان پس از دستور PlugInRun استفاده می‌شود.

حال به تصاویر زیر دقت کنید که تفهیم مطلب را ساده‌تر می‌نماید.





همانند سایر فرامین MMB این ۳ فرمان هم دارای چند قسمت هستند که در ادامه بحث می‌شوند.

PlugInSet	
در عمل	در تئوری
PlugInSet ("Plug-In" , "Variable")	
شرح	
<p>در رابطه با خود PlugInSet که قبلاً صحبت شد، پارامتر اول همان نام شیء Plug-In را مشخص می‌کند که از کادر خصوصیت Plug-In قسمت label قابل تغییر است و به واسطه همین نام MMB می‌فهمد که با چه Plug-In ای رابطه برقرار کند. در قسمت Variable می‌توانید از متغیری استفاده کنید و آن را به Plug-In ارسال کنید این متغیر هم می‌تواند عددی باشد هم رشته‌ای، نوع متغیر بسته به نوع فرمان است و می‌توانید از روی راهنمای Plug-In آن را تشخیص دهید.</p>	
PlugInRun	
در عمل	در تئوری
PlugInRun("Plug-In" , "Command")	
<p>در رابطه با فرمان PlugInRun که بحث شد، پارامتر اول نام شیء Plug-In را مشخص می‌کند و پارامتر دوم فرمانی از فرامین Plug-In را اجرا می‌کند. جهت اطلاع از فرامین یک Plug-In بایستی به راهنمای ارائه شده همراه Plug-In رجوع کرد. ممکن است فرمان اجرا شده Plug-In خروجی به همراه داشته باشد که در ادامه می‌توان توسط فرمان PlugInGet آن را دریافت کرد.</p>	
PlugInGet	
در عمل	در تئوری
PlugInGet("Plug-In" , "Variable")	
شرح	
<p>همان‌طور که می‌دانید از این فرمان جهت دریافت خروجی یک فرمان پیشتر اجرا شده استفاده می‌شود. قسمت اول این فرمان نام شیء Plug-In را مشخص می‌کند. قسمت دوم نام متغیری را مشخص می‌کند که حاوی مقدار خروجی است این متغیر هم می‌تواند عددی باشد و هم رشته‌ای. نوع متغیر را می‌توان با توجه به راهنمای همراه Plug-In مشخص ساخت.</p>	

مثال

اکنون ۳ فرمان گفته شده را در کنار هم می‌آوریم تا بتوانیم از Plug-In استفاده کنیم.

```
var$ = "text"
PlugInSet ("Plug-In" , " var$")
PlugInRun ("Plug-In" , "uppercase")
PlugInGet ("Plug-In" , "var$")
Message ("out put is" , "var$")
```

در بلوک کد بالا ابتدا محتوی متغیر `var$` توسط فرمان `PlugInSet` به `Plug-In` ارسال می‌شود. سپس دستور `PlugInRun` بر اساس متغییر درستی دسـتور مربوطه (uppercase) را اجرا می‌کند و یک خروجی تولید می‌نماید. در ادامه دستور `PlugInGet` مقدار خروجی را دریافت کرده و به `MMB` انتقال می‌دهد. که در آخر میتوان از این مقدار با استفاده از سایر فرامین سود جست. حالت گفته شده گونه‌ای بود که `Plug-In` خروجی تولید کرد اما حالتی پدید می‌آید که هیچ خروجی تولید نمی‌شود، یعنی اینکه احتیاجی به استفاده از فرمان `PlugInGet` نیست. این حالت برای مثال ممکن است برای مواقعی که می‌خواهیم اندازه یک فونت را تغییر دهیم پدید آید. چیزهایی که نیاز است، یکی اندازه نهایی فونت و فرمانی جهت اجرای آن است و هیچ نیازی به خروجی نمی‌باشد. این گونه از ارتباط همانند مثال زیر می‌باشد.

مثال:

```
fontSize = 16
PlugInSet("Plug-In" , " fontSize")
PlugInRun("Plug-In" ,"Set fonts size")
```

**** توجه کنید که فعلاً تمام این Plug-In ها فرضی و مجازی هستند.**

ممکن است گونه‌های مختلفی در استفاده از فرامین `MMB` برای ارتباط با `Plug-In` ها پدید آید و این بستگی به نوع فرمان `Plug-In` دارد که می‌توان از طریق راهنمای `Plug-In` آن را مشخص ساخت.

بگذارید چون صحبت درباره راهنمای `Plug-In` شد، قدری راجع به آن صحبت کنیم. همانند سایر راهنماها، راهنمای `Plug-In` ها هم بسته به نیاز حاوی تعدادی مثال هستند که این مثالها هدف اصلی هر دستور را نشان می‌دهد و همیشه نمی‌توان با کپی و چسبیدن آنها از راهنما به ویرایشگر اسکریپت انتظار عملکرد صحیح را از آنها داشت. گاهی اوقات لازم است که با توجه به خطوط قبلی کد برنامه و براساس نیازمان کد این مثالها را تغییر دهیم و نهایتاً از آنها استفاده کنیم. نکته‌ی دیگری را که باید یادآور شد این است که در غالب راهنماها بر چسب `Plug-In` ، `In` در نظر گرفته می‌شود که چنانچه بخواهیم نام دیگری داشته باشیم باید در فرامین مرتبط با `Plug-In` این برچسبها را تغییر دهیم.

اما در رابطه با حالت‌های به کارگیری ۳ دستور قبل صحبت می‌کردیم. حالت‌های دیگری هم که ممکن است پدید آید به صورت زیر است:

الف) استفاده از یک دستور بدون هیچ ورودی و خروجی

ب) استفاده از یک دستور بدون مقدار ورودی

حالت الف ممکن است مواقعی پیش بیاید که فرمان ما تنها عملی را بایستی انجام دهد برای مثال ویندوز را Shutdown کند.

حالت ب ممکن است مواقعی پیش بیاید که فرمان Plug-In مستقلاً عمل کرده و مقدار خروجی تولید کند. مثلاً هنگامی که بخواهیم ساعت و تاریخ را دریافت کنیم، اگر پا را فراتر از این حالات بگذاریم حالت‌های دیگری به وجود می‌آید که خود از ترکیب چند حالت قبل پدید می‌آید، همانند مثال زیر:

بلوک ۱

```
var$ = "My computer is P4"  
PlugInSet("Plug-In" , "var$")  
PlugInRun("Plug-In" , "upper case")  
PlugInGet("Plug-In" , "text$")
```

بلوک ۲

```
PlugInSet("Plug-In" , "text$")  
PlugInRun("Plug-In" , "Show message box")
```

در مثال قبل ابتدا رشته در بلوک ۱ به حالت بزرگ تبدیل شده و به بلوک ۲ ارسال می‌شود تا در کادر پیغام نشان داده شود.

رابطه‌ای Plug-In و متغیر تنها به یک انتساب ساده از قبیل var\$=myname منتهی نمی‌شود، بلکه گاهی اوقات پیش می‌آید که این رشته‌ها توسط کاراکتری از هم جدا می‌شوند مثلاً برای تعیین مقدار رنگ:

```
color$ = "220,150,105"
```

این گونه‌ها بر اساس تعاریفی که از پیش در Plug-In به وسیله‌ی مؤلف آن آمده است پردازش و تجزیه می‌شود. حالت‌های دیگری از این قبیل انتساب‌ها هم وجود دارند. مثال:

```
Regkey $ = 'software\program\section\keyname'
```

```
Flag$ = 'con|yes no cancel|caption'
```

کنترل رویداد (Event Handling)

وقایع (Event): اتفاقات و توسعه‌هایی هستند که در Plug-In رخ می‌دهند و به برنامه شما اطلاع داده می‌شوند.

کنترل کننده رویداد: شیء‌های اسکریپت MMB هستند که همانند نوعی شبکه وقایع را دریافت می‌کند.

همانطور که قبلاً خواندید بعضی از فرامین مقادیر خروجی تولید می‌کنند اما برخی از Plug-In ها بلافاصله پس از دستور PlugInRun مقدار خروجی تولید نمی‌کنند. وظایفی از قبیل پردازش داده‌ها، وقفه‌های طولانی، رویدادهای دوره‌ای و ... سبب بروز چنین حالتی می‌شوند. خوب در این صورت سؤال اینجاست که Plug-In برنامه MMB چگونه در این حالت با MMB به تعامل می‌پردازد؟ جواب اینجاست که خود Plug-In فشردن کلید یا کلیدهایی را جهت ارسال اطلاعات به MMB شبیه‌سازی می‌کند. به شکل‌های زیر توجه کنید:

شکل ۱: فرستاده شدن رویداد از Plug-In را بیان می‌کند.

شکل ۲: شبیه‌سازی کلیدها را تداعی می‌کند.

شکل ۳: نحوه در جریان قرار گرفتن برنامه ما را نشان می‌دهد.

شکل ۴: دریافت واقعه را توسط شیء اسکریپتی که برای این واقعه تنظیم شده است را نشان می‌دهد.

می‌بینیم که شیء اسکریپت برای حالت فشردن کلیدهای Alt + Control + Shift + D تنظیم شده است و چنانچه ما از پیش برای Plug-In این ترکیب را تعریف کرده باشیم هنگام بروز یک رویداد برای حالت‌های خاص گفته شده شیء اسکریپت فراخوانی می‌شود و فرامین داخل شیء اسکریپت اجرا می‌شود. این نکته را باید دانست که کلیدهای ترکیبی نایستی کلیدهایی باشند که از پیش برای اعمال مختلف در سیستم عامل در نظر گرفته شده‌اند باشند.



همچنین بایستی به بزرگی و کوچکی حروف و روشن بودن کلید Caps Lock توجه داشت. اما برای استفاده از Event Handling بایستی استفاده از راهنمای Plug-In را فراموش نکرد.

توجه: چنانچه در هنگام ارسال کلیدها برنامه شما در حالت فعال نباشند ممکن است که این شبیه سازی فشردن کلیدها به برنامه ای دیگر فرستاده شود. به عنوان مثال هنگامی که برنامه یک کار تبدالی را احضار می کند دیگر کانون (Focus) روی برنامه اصلی نیست بلکه بر روی کادر تبدالی است، راه حل این مشکل غیر فعال سازی عمل Event Handling در حین نمایان بودن کادر تبدالی است. البته اگر ویژگی غیر فعال سازی برای Plug-In تعریف شده باشد. همچنین استفاده مکرر از کنترل رویداد ممکن است تداخل ایجاد کند که می تواند سبب بروز اشکال در اجرای برنامه شود. بر همین اساس گفته می شود که در نسخه های بعدی MMB این مشکل حل خواهد شد.

خلاصه‌ای از چگونگی Plug-In های MMB:

- Plug-In های MMB همانند Plug-In های برنامه‌های مختلف دیگر نیستند.
 - یک پنجره ۲ بعدی است که گاهی بصری است و گاهی نیست.
 - برای نوع اول در صورت نیاز به چندین پنجره و چندین نسخه نیاز داریم.
 - برای نوع اول تمام تنظیمات و پردازش‌ها از طریق اسکریپت‌های MMB صورت می‌گیرد.
 - حالت‌های مختلفی جهت استفاده از ۳ دستور PlugInSet, PlugInRun, PlugInGet وجود دارد که برحسب نیاز ممکن است یکی یا دو تا از این فرامین حذف شوند.
 - عمل Event Handling هنگامی که به کار می‌آید که یکسری عملیات‌های تأخیری رخ دهد.
 - فایل‌های Plug-In برنامه MMB از سایتهای مربوطه یا سایت مؤلف قابل دریافت هستند.
- حال که به انتهای این بحث رسیدید دنیایی باور نکردنی و جالب را در پیش روی خود خواهید دید، دنیایی که روز به روز در حال توسعه و تکامل است و آن چیزی نیست جز دنیای کاربران MMB.

Tweak Plug-In

این PlugIn یکی از قدیمی ترین و در عین حال کاراترین PlugIn برای MMB می باشد که در ادامه می خواهیم تنها به شرح ویژگی ها و فرامین آن بپردازیم.

• Change video mode

فرامین ارائه شده در این قسمت برای تغییر پارامترهای « ویدئوی » هستند.

VideoModeChange-

جهت تعویض مد « ویدئوی » می باشد که این مد را می توانیم توسط یک متغیر به PlugIn ارائه کنیم. فرمت محتوی متغیر باید به گونه ی زیر باشد.

' WIDTH | HEIGHT | BPP '

(بیت برای هر پیکسل ، ارتفاع ، عرض)

مثال:

```
Vidmode$ = ' 640 | 480 | 8 '  
PlugInSet( 'Tweak' , 'Vidmode$' )  
PlugInRun( 'Tweak' , 'VideoModeChange' )
```

مثال بالا سبب تغییر مد به حالتی می شود که عرض 640، ارتفاع 480 و BPP 6 می شود.

VideoModeRestore-

این فرمان سبب بازگشت مد ویدئویی به تنظیمات پیش فرض می شود.

مثال:

```
PlugInRun( 'Tweak' , 'VideoModeRestore' )
```

Dialoge Box –

این قسمت برای نمایان ساختن کادرهای تبدلی می باشد.

FDSetsDefault –

این فرمان پوشه پیش فرش را برای کادر بازگشایی معین می کند.

FDSetsFilter –

این فرمان فیلتر را برای کادر بازگشایی معین می کند.

FDOpenFileDialoge -

این فرمان کادر بازگشایی را نشان می دهد.

FDSaveFileDialoge-

این فرمان کادر ذخیره فایل را نشان می دهد.

FDGetFileName-

این فرمان مسیر کامل فایل انتخاب شده را نشان می دهد.

FDGetFilePath-

این فرمان فقط نام فایل انتخاب شده بدون احتساب مسیر را نشان می دهد.

مثال:

```
Filter$='Text Files (*.txt)|*.All Files|*.*|'|  
PlugInSet("Tweak","Filter$")  
PlugInRun("Tweak","FDSetsFilter")  
DefaultFolder$='C:\My ocuments\'
```



```

PlugInSet ("Tweak", "DefaultFolder$")
PlugInRun ("Tweak", "FDOpenFileDialog")
PlugInRun ("Tweak", "FDGetFileName")
PlugInGet ("Tweak", "Path$")
Message ("Selected File is: ", "Path$")

```

Message Box –

این ویژگی این امکان را فراهم می‌کند تا بتوانیم کادر پیغام داشته باشیم. البته MMB خود کادر پیغام دارد، اما شما نمی‌توانید کنترل کاملی بر آن داشته باشید. این PlugIn کادر پیغام را با جزئیات کامل که تحت کنترل شماست را ارائه می‌دهد.

MsgBox-

ابتدا بایستی مقدار ورودی را با متغیری جهت ارائه به PlugIn تعیین کرد. برای این کار متغیری رشته‌ای را انتخاب و عمل انتساب را به صورت زیر انجام می‌دهیم:

Mes\$ = ' Message text | Title bar text | Flags '

Message text: متن پیغام را مشخص می‌کند. چنانچه بخواهید متن را در چند خط بگنجانید می‌توانید با استفاده از عبارت \n خط جدیدی تولید و ادامه متن را پس از این عبارت بیاورید. اما عبارت \t هم فاصله‌ای به اندازه ۸ کاراکتر قبل از کلمه بعد از خود ایجاد می‌کند.

Title bar text: این قسمت برای معین ساختن عنوان کادر پیغام می‌باشد.

Flags: این قسمت جهت مشخص ساختن دکمه‌های کادر پیغام و نوع آنها است.

نکته: توجه کنید که چنانچه در متون خود در هنگام استفاده از این فرمان نیاز به استفاده از کاراکتر " \ " را دارید بایستی از " \ \ " استفاده کنید برای مثال 'Path is C:\\ test' به 'Path is C:\ test' ترجمه خواهد شد. اما جدول زیر Flag های موجود را ارائه می‌دهد:

B Flag سبب ایجاد دکمه می‌شود:	
Bo	دکمه‌ی Ok (به صورت پیش فرض)
Boc	دکمه‌های Cancel,Ok

Byn	دکمه‌های No, Yes
Bync	دکمه‌های Cancel, No, Yes
Brc	دکمه‌های Cancel, Retry
Bari	دکمه‌های Ignore, Retry, Abort
I Flag سبب قرار گیری آیکن بر روی کادر می‌شود	
Ie	سبب ظاهر شدن آیکن تعجب می‌شود
Ii	یک آیکن حاوی حرف I انگلیسی در کادر ظاهر می‌شود.
Iq	آیکن پرسش در کادر نمایان می‌شود
is	آیکن ایست را در کادر نمایان می‌سازد.
D Flag کنترل می‌کند در هنگام ظهور کادر کدام دکمه به صورت پیش فرض فعال باشد.	
D1	کانون (Focus) به دکمه اول منتقل می‌شود. (حالت پیش فرض)
D2	کانون به دکمه دوم منتقل می‌شود.
D3	کانون به دکمه سوم منتقل می‌شود.
M Flag رفتار کادر پیغام را مشخص می‌کند.	
ma	این حالت کادر پیغام را نمایان می‌سازد و کاربر بایستی به آن پاسخ دهد اما کاربر می‌تواند در حین فعال بودن کادر به سراغ سایر پنجره‌ها هم برود.
Ms	این حالت کادر پیغام را نمایان می‌سازد که کاربر باید به آن واکنش نشان دهد و در حین نمایان بودن این کادر تمامی پنجره‌های دیگر غیر فعال و غیر قابل دسترس است. از این کادر هنگامی باید استفاده کنیم که بخواهیم به کاربر پیغامی جدی و حیاتی را منتقل کنیم.

پس از نمایش کادر پیغام می‌توانیم با استفاده از دستور PlugInGet بفهمیم که کاربر چه کلمه‌ای را فشار داده است. در جدول زیر مقدارهای بازگشتی به همراه مفاهیم آنها آمده است.

مفهوم	مقدار بازگشتی
Ok	O
Cancel	C
Yes	Y
No	N
Abort	A
Retry	R
Ignore	I

مثال:

```
Message$ = ' Would you like to register? | My App | byn + ig '
PlugInSet('Tweak' , 'Message$')
PlugInRun('Tweak' , 'MsgBox')
PlugInGet('Tweak' , 'Selected$')
If ( Selected$= 'y' ) then
Message('You pressed yes ' , ' ' )
Else
Selected$ = ' n '
Message ( 'You pressed no ' , ' ' ' )
End
```

مثال بالا یک کادر پیغام با ۲ دکمه Yes و No را نشان می‌دهد و در ادامه دکمه انتخاب شده را باز می‌گرداند.

Registry I/O

این قسمت قابلیت ایجاد، خواندن و پاک کردن کلیدهای رجستری را فراهم می‌کند.

RegistrySetRootKey –

این فرمان جهت مشخص کردن ریشه به کار می‌رود. ابتدا، بایستی یکی از عبارتهای زیر را بر حسب نیاز به متغیری رشته‌ای جهت تعیین ریشه نسبت دهیم:

- HKEY – CLASSES – ROOT
- HKEY – CURRENT – CONFIG
- HKEY – CURRENT – USER
- HKEY – LOCAL – MACHINE
- HKEY – USERS

سپس توسط فرمان PlugInSet عبارت مورد نظر را به PlugIn ارسال می‌کنیم به صورت زیر:

```
rootkey$ = ' HKEY - LOCAL - MACHINE '
PlugInSet('Tweak' , 'rootkey$')
PlugInRun('Tweak' , 'RegistrySetRootKey')
```

نکته: ریشه پیش فرض در هنگام بارگذاری فایل PlugIn ، HKEY- CURRENT-USER است.

RegistryCreateKey–

این فرمان برای ایجاد کلید در رجستری مورد استفاده قرار می‌گیرد. و این قابلیت را دارد تا همزمان چند کلید تولید نماید. به عنوان مثال چنانچه محتوی متغیر ورودی به گونه‌ی زیر باشد:

Software\MyMMBApp\Setting

PlugIn همزمان دو کلید Setting و MyMMBApp را تولید می‌نماید. جهت استفاده از این فرمان ابتدا مسیر و نام کلید را با استفاده از PlugInSet به PlugIn ارسال و توسط فرمان PlugInRun کلید را تولید می‌کنیم. حال جهت درک بیشتر به مثال زیر توجه کنید:

```
Newkey$= 'Software\MyMMBApp\MySetting
PlugInSet ("Tweak" , "Newkeys$")
PlugInRun ("Tweak" , "RegistryCreateKey")
```

مثال بالای تنها یک کلید تولید می‌نماید. باید توجه داشت که قبل از به کارگیری دستور RegistryCreateKey بایستی ریشه را توسط فرمان RegistrySetRootKey معین کرد.

RegistryDeleteKey-

برای پاک کردن کلید از رجستری از این فرمان استفاده می‌شود. البته در استفاده از این فرمان باید احتیاط کرد، چرا که استفاده نا به جا و غلط ممکن است به رجستری آسیب رساند و سبب خرابی و ایجاد اشکال جدی در ویندوز شود. برای استفاده ابتدا باید مسیر را توسط متغیری با فرمان PlugInSet به PlugIn ارسال کنیم. در ادامه با دستور PlugInRun کلید را حذف می‌کنیم.

مثال:

```
delkey$= 'Software\MyAppkey\MySetting'
PlugInSet ("Tweak" , "delkey$")
PlugInRun ("Tweak" , "RegistryDeleteKey")
```

مثال بالا کلید MySetting را از مسیر مربوطه حذف می‌کند. البته در این قسمت هم بایستی از دستور RegistrySetRootKey استفاده کرد.

RegistrySetValue-

این فرمان مقدار یک کلید را تنظیم می‌کند چنانچه کلید داده شده موجود نباشد این فرمان آن کلید را ایجاد می‌کند. ابتدا بایستی با استفاده از حالت زیر محتوی متغیر را مشخص کرد:

' Full path to the registry key | Value name | Value to set '

قسمت اول سمت چپ همان مسیر کلید در رجستری است. قسمت دوم نام متغیر در رجستری است و قسمت سوم مقدار جهت تنظیم در رجستری است. چنانچه بخواهیم مقدار متغیر پیش فرض (Default) را در رجستری تنظیم کنیم تنها کافیست تا قسمت دوم را خالی بگذاریم و از نوشتن آن صرف نظر کنیم.

```
rootkey $ = 'HKEY - CURRENT - USER '  
PlugInSet ("Tweak" , "rootkey$")  
PlugInRun ("Tweak", "RegistrySetRootKey")  
SetValCmd$='ControlPanel\Desktop\wallpaper|C:\Windows\Bubbles.BMP'  
PlugInSet ("Tweak" , "SetValCmd$")  
PlugInRun ("Tweak", "RegistrySetVal")
```

مثال بالا عکس پس زمینه دسکتاپ را به عکس مورد نظر تغییر می‌دهد توجه کنید که در اینجا نام متغیر wallpaper است که با کاراکتر "|" جدا شده است و قسمت بعدی محتوی متغیر در رجستری است.

RegistryGetValue-

این فرمان محتوی یک کلید رجستری یا یک متغیر رجستری را دریافت می‌کند. جهت استفاده از این فرمان ابتدا بایستی با استفاده از قالب زیر مسیر کلید را به PlugIn ارسال داشت.

' Full path to the registry key | Value name '

قسمت اول مسیر کامل کلید رجستری را معین می‌کند و قسمت دوم نام متغیر رجستری را. چنانچه بخواهید مقدار متغیر پیش فرض یک کلید را بگیرید کافیست از نوشتن قسمت دوم پرهیز کنید.

مثال:

```
rootkey$ = ' HEKY - CURRENT - USER '  
PlugInSet ("Tweak" , "rootkey$")  
PlugInRun ("Tweak", "RegistrySetRootKey")  
SetValCmd$ = 'ControlPanel\Desktop|wallpaper'  
PlugInSet ( "Tweak" , "SetValCmd$")  
PlugInRun ("Tweak", "RegistryGetValue")  
PlugInGet ("Tweak", "wallpaper$")  
Message ("", "wallpaper$")
```

مثال بالا مسیر عکس پس زمینه ویندوز را که در یک متغیر در رجستری ویندوز است را باز می‌گرداند.

Tray Icon –

این قسمت این امکان را فراهم می‌کند تا برنامه در حال اجرا به SystemTray منتقل شود. همچنین می‌توان آیکنی را برای برنامه به هنگام رفتن به SysTray تعیین کرد و یا آن را به‌روزرسانی کرد. البته شایان ذکر است که این فرمان در نسخه 4.9.5 برنامه MMB به درستی عمل نمی‌کند.

MinimizeToTrayIcon-

این دستور آیکن برنامه هنگامی که به SysTray می‌رود را تعیین می‌کند.

مثال:

```
IconPath$ = ' < Embedded > \Globe.Ico'  
PlugInSet("Tweak", "IconPath$")  
PlugInRun("Tweak", "MinimizeToTrayIcon")
```

UpdateTrayIconTooltip-

این فرمان Tooltip برنامه را هنگامی که به SysTray منتقل شده است را مشخص می‌کند.

مثال:

```
Tooltip$=' My MMB App'  
PlugInSet("Tweak", "Tooltip$")  
PlugInRun("Tweak", "UpdateTrayIconTooltip")
```

UpdateTrayIcon-

برای به‌روزرسانی آیکن برنامه در حالت Minimize در SysTray بایستی از این دستور استفاده کرد.

مثال:

```
IconPath$ = ' < Embedded > \NewIcon.ico'  
PlugInSet("Tweak", "IconPath$")  
PlugInRun("Tweak", "UpdateTrayIcon")
```

TrayIconIsMinimize-

این دستور کنترل می‌کند که آیا برنامه به حالت Minimize شده در Systray است یا خیر.

مثال:

```
PlugInRun("Tweak", "TrayIconIsMinimize")
PlugInGet("Tweak", "IsMinimize")
```

مثال بالا ۲ مقدار ۱ به معنی Minimize بودن و صفر را به معنی عدم Minimize بودن باز می‌گرداند.

Chang Shape -

یکی از قسمتهای جالب Tweak همین ویژگی است که به ما امکان می‌دهد شکل صفحه و پروژه را در زمان اجرا تغییر دهیم. البته از قبل بایستی با برنامه‌ای به نام CreateRegion که در CD همراه کتاب است، شکل را ایجاد و در فایل مخصوص ذخیره نماییم و سپس با دادن مسیر آن فایل به PlugIn می‌توانیم شمایل پنجره پروژه را تغییر دهیم.

SetRegion-

فرمانی است که عمل تغییر شکل را صورت می‌دهد.

ResetRegion-

شکل پنجره را به حالت اول باز می‌گرداند.

مثال:

```
Region$ = ' < Embedded >\MyRegion.rgn'
PlugInSet("Tweak" , "Region$")
PlugInRun("Tweak", "SetRegion")
```

Detect App Instances -

چنانچه بخواهیم بدانیم که چند نسخه از برنامه ما در حال اجراست بایستی از این فرمان استفاده کرد. تنها کافیست که مسیر فایل اجرایی برنامه مورد نظر را به PlugIn ارسال کنیم.

GetNumInstances-

تعداد نسخه‌های در حال اجرا را باز می‌گرداند.

مثال:

```
appPath$ = '< SrcDir>\AutoRun.exe'  
PlugInSet("Tweak" , "appPath$")  
PlugInRun("Tweak", "GetNumInstances")  
PlugInGet("Tweak" , "Instances")  
Message("" , "Instances")
```

نکته: یکی از مواقعی که ممکن است بخواهیم از این تابع استفاده کنیم هنگامی است که تنها می‌خواهیم یک نسخه از برنامه ما در آن واحد اجرا شود.

Detect CPU Vender and Speed

این قسمت نام کارخانه سازنده پردازشگر و سرعت آن را باز می‌گرداند.

CPUGetName-

این تابع نام سازنده پردازشگر را باز می‌گرداند.

CPUGetSpeed-

این تابع سرعت CPU را باز می‌گرداند.

مثال:

```
PlugInRun("Tweak", "CPUGetName")  
PlugInGet("Tweak" , "CPUName$")  
PlugInRun("Tweak", "CPUGetSpeed")  
PlugInGet("Tweak" , "CPUSpeed$")  
Info$='Your CPU is: '+CPUName$ + 'Running at '+ CPUSpeed$  
Message("" , "Info$")
```


Manipulate INI Files

این فرمان هم یکی از قسمت‌های پرکاربرد Tweak است که برای خواندن و نوشتن در فایل‌های INI به کار می‌رود.

IniSetFile-

این فرمان فایل ini را که باید خوانده و یا در آن نوشته شود را مشخص می‌کند.

IniRead-

این فرمان اطلاعات را از فایل ini می‌خواند.

IniWrite-

این فرمان اطلاعات را در فایل ini می‌نویسد.

IniDelete-

این فرمان یک قسمت از اطلاعات فایل ini را حذف می‌کند.

توجه کنید که فرمت ini یک پسوند استاندارد ویندوز است که نحوه ذخیره‌سازی اطلاعات در آن به صورت زیر است:

[Section Name] → نام قسمت

Name = MMB → اطلاعات که مقادیر آنها به

Ver=4.9.5 نسبت داده شده است.

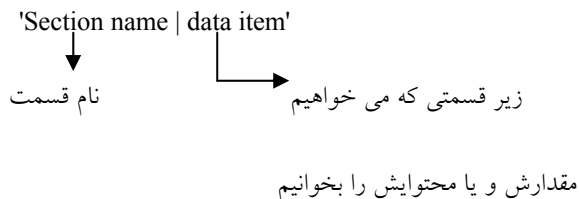
توجه: مزیت فایل‌های INI در این است که عملیات‌های خواندن، نوشتن و حذف تماماً در حافظه صورت می‌گیرد. و بدین ترتیب سبب بهبودی و توسعه عملیات‌های خواندن و نوشتن و حذف داده‌ها می‌شود و مادامی که برنامه شما در حال اجرا است اطلاعات جدید در فایل ذخیره نمی‌شود و به محض خروج از برنامه و بسته شدن فایل ini اطلاعات در فایل ذخیره می‌شود. خوبی دیگر این نوع فایل‌ها این است که به سادگی امکان ویرایش آنها توسط برنامه Notepad ویندوز فراهم است.

مثال:

```
iniFilePath$ = '< Embedded >\MyApp.ini'  
PlugInSet("Tweak", "iniFilePath$")  
PlugInRun("Tweak", "IniSetFile")
```

باید توجه داشت که چنانچه فایل مسیر دهی شده وجود خارجی نداشته باشد این دستور آن را ایجاد می‌کند.

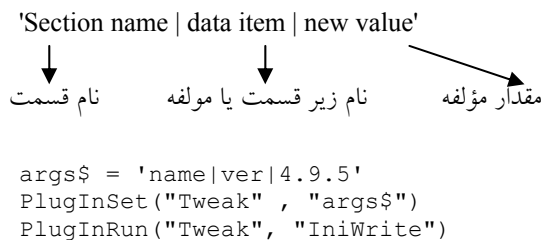
ابتدا بایستی بر طبق فرمت زیر قسمتی را که مایلیم اطلاعات از آن خوانده شود را مشخص کنیم.



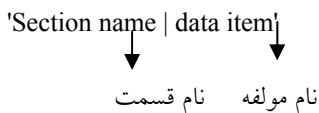
مثال:

```
args$ = 'name|ver'  
PlugInSet("Tweak", "args$")  
PlugInRun("Tweak", "IniRead")  
PlugInGet("Tweak", "data$")  
Message("", "data$")
```

اما برای نوشتن در فایل ini ابتدا بایستی همانند فرمت زیر مقدار و نام مؤلفه و نام قسمت را به PlugIn ارسال کنیم.



مجدداً برای حذف یک قسمت فایل ini هم بایستی به گونه‌ی زیر عمل کرد:



و اگر بخواهیم تنها یک زیر قسمت را حذف کنیم تنها کافیت تا نام آن را به `PlugIn` ارسال کنیم.

```
args$ = 'name |ver'  
PlugInSet("Tweak" , "args$")  
PlugInRun("Tweak", "IniDelete")
```

و برای حذف یک قسمت (Section):

```
args$ = 'name '  
PlugInSet("Tweak" , "args$")  
PlugInRun ("Tweak", "IniDelete")
```

Pop-up Menu -

این قسمت برای تولید منوهای جهنده (Pop-up) می‌باشد. باید گفت که هر برنامه تنها می‌تواند یک منوی جهنده داشته باشد. ممکن است این منوهای جهنده در هنگام طراحی به درستی عمل نکنند اما خوشبختانه در فایل کامپایل شده نهایی به درستی عمل می‌کنند.

MenuReset-

این دستور تمام مؤلفه‌های منوی جهنده جاری را از بین می‌برد. این فرمان هنگامی مفید است که بخواهیم منوی جهنده قبلی را از بین ببریم و یک منوی جهنده جدید تولید کنیم.

مثال:

```
PlugInRun ("Tweak" , "MenuReset")
```

MenuAddItem-

برای افزودن مؤلفه‌ها به یک منوی جهنده از این فرمان استفاده می‌کنیم و مکرراً برای افزودن هر مؤلفه باید از این دستور استفاده کرد. همچنین علاوه بر مؤلفه‌ها می‌توان جدا کننده‌ها (Separators) را با استفاده از کاراکتر "-" به منوها اضافه کرد.

مثال:

```
MenuItem$ = 'This is Item # 1'
PlugInSet("Tweak" , "MenuItem$")
PlugInRun("Tweak", "MenuAddItem")
MenuItem$ = 'This is Item # 2'
PlugInSet("Tweak" , "MenuItem$")
PlugInRun("Tweak", "MenuAddItem ")
MenuItem$ = ' - '
PlugInSet("Tweak" , "MenuItem$")
PlugInRun("Tweak", "MenuAddItem")
MenuItem$ = 'Exit'
PlugInSet( "Tweak" , "MenuItem$")
PlugInRun("Tweak", "MenuAddItem")
```


مثال بالا ابتدا ۲ مؤلفه ایجاد می‌کند، سپس یک جدا کننده تولید و نهایتاً گزینه Exit را به منو اضافه می‌کند.

MenuShow-

وقتی که مؤلفه‌های یک منو را اضافه می‌کنیم بایستی با استفاده از این دستور جهت نشان دادن منو عمل کنیم. البته می‌توان به PlugIn مختصات نقطه‌ای را که می‌خواهیم منو از آنجا ظاهر شود را بدهیم. جهت اینکه دریابیم کدام مؤلفه‌ی منو کلیک شده می‌توان با PlugInGet("Tweak", "Selected") شماره مؤلفه را دریافت کرد. اگر شماره دریافتی صفر باشد به معنای عدم کلیک است و باقی اعداد بین ۱ تا شماره آخرین مؤلفه به معنای مؤلفه‌ی مربوطه است. باید توجه کنیم که جداکننده‌ها در محدوده شماره قرار نمی‌گیرند چرا که قابل کلیک کردن نیستند.

مثال:

Pos\$ = '20|20'



```
PlugInSet( "Tweak" , "Pos$")
PlugInRun("Tweak", "MenuShow")
```

مثال بالا باعث نمایان شدن منو در مختصات (20,20) می شود.

مثال:

```
PlugInRun ("Tweak" , "MenuShow$")  
PlugInGet ("Tweak", "Selected")
```

مثال بیان شده شماره قسمت کلیک شده را باز می گرداند.

MenuSetHandler-

اما حالتی است که می خواهیم منوی ما با عمل کلیک نمایان شود و یا به هنگام انتخاب مؤلفه ای شی اسکریپت مربوطه فراخوانی شود. ابتدا منو را ایجاد کنید و سپس کنترل کننده آن را تنظیم کنید. سپس کاربر با انتخاب یک مؤلفه از منوی جهنده سبب فراخوانی شی اسکریپت می شود. البته باید به PlugIn گفت که چه کلیدهایی سبب فعال شدن شی اسکریپت می شود.

```
Msg$ = 'CONTROL + SHIFT + ALT + A'  
PlugInSet ("Tweak" , "Msg$")  
PlugInRun ("Tweak", "MenuSetHandler")
```

یادآوری می کنم که بایستی همان کلید یا کلیدهایی که برای PlugIn تعریف می شود برای شی اسکریپت هم تعریف شود.

نکته: جهت اضافه کردن یک منوی جهنده به آیکن برنامه هنگامی که در Systray قرار گرفته بایستی قبل از Minimize شدن از دستور MenuSetHandler استفاده کرد.

MenuClearHandler-

این دستور برای پاک کردن کنترل منوی جهنده است و با اعمال این دستور منوی جهنده با عمل راست کلیک و یا فشردن کلیدهای معین شده به عنوان کنترل کننده نمایان نخواهد شد.

مثال:

```
PlugInRun("Tweak", "MenuClearHandler")
```

Client Area Drag

حتماً توجه کرده‌اید که برخی از برنامه‌ها برای عمل درگ کردن تنها از نوار عنوان خود استفاده می‌کنند. با استفاده از این فرمان می‌توان کل سطح پنجره را قابل درگ کرد.

EnableClientAreaDrag-

مثال:

```
PlugInRun("Tweak", "EnableClientAreaDrag")
```

Change the wallpaper-

برای تعویض عکس پس زمینه ویندوز، بایستی از این قسمت استفاده کرد. علاوه بر تغییر عکس زمینه این ویژگی Tweak این قابلیت را فراهم می‌کند تا حالت عکس اعم از کاشی وار (Tile)، کشیده (Stretch) و مرکزی (Center) را تنظیم کنید.

SetWallpaper-

همانند برخی دیگر از فرامین ابتدا باید توسط قالب زیر اطلاعات را به PlugIn ارسال کنیم.

'Bitmap for wallpaper | Bitmap display style | Commit'

Bitmap for wallpaper: مسیر عکس جدید

Bitmap display style: حالت عکس (مرکز = 0 کاشی وار = 1 کشیده = 2)

Commit: دائمی بودن یا موقتی بودن زمینه را تنظیم می‌کند. عدد 1 برای دائمی و 0 برای موقتی بودن

اگر از نوشتن Commit صرف نظر کنیم پس زمینه به طور دائمی نمایان می‌شود.

مثال:

```

BitmapPath$ = ' C:\Windows\Bubbles.bmp'
BitmapStyle$ = '|1'
BitmapPermanet$ = '|1'
WallpaperStuff$=BitmapPath$+BitmapStyle$+BitmapPermanet$
PlugInSet ("Tweak", "WallpaperStuff$")
PlugInRun ("Tweak", "SetWallpaper")

```

Snap an MMB application to the desktop's edge's

با به کارگیری این ویژگی برنامه‌ها به هنگام نزدیک شدن به لبه‌های Desktop به سمت لبه‌ها جذب می‌شوند(پرش می‌کنند).

EnableEdgeSnap-

با فراخوانی این فرمان، عمل جهش صورت می‌گیرد.

DisableEdgeSnap-

با فراخوانی این فرمان عمل جهش غیر فعال می‌شود.

SetEdgeSnapDistance-

به طور پیش فرض هنگامی که لبه‌های برنامه از لبه‌های دسکتاپ فاصله داشته باشد عمل جذب رخ می‌دهد. حال با استفاده از فرمان بالا می‌توان این فاصله را تنظیم کرد. البته این دستور اختیاری است. فاصله در واحد پیکسل تعریف می‌شود.

مثال:

```

Distance=30
PlugInSet ("Tweak" , "Distance")
PlugInRun ("Tweak", "SetEdgeSnapDistance")
PlugInRun ("Tweak", "SetEnableEdgeSnap")

```

GetEdgeSnapDistance-

این دستور مقدار فاصله تنظیم شده برای عمل جهش را باز می‌گرداند.

مثال:

```
PlugInRun ("Tweak", "GetEdgeSnapDistance")  
PlugInGet ("Tweak", "Distance")  
Message ("", "Distance")
```

Memory Stats –

جهت اطلاع از وضعیت حافظه دستگاه می‌توان از این ویژگی استفاده کرد.

GetMemoryLoad-

این فرمان درصدی از حافظه را که در حال استفاده است را باز می‌گرداند.

GetTotalPhys-

این فرمان تعداد بایت‌های حافظه فیزیکی را باز می‌گرداند. (همان RAM)

GetAvailPhys-

مقدار کل حافظه فیزیکی (RAM) را باز می‌گرداند.

GetTotalPageFile-

مقدار کل PageFile را باز می‌گرداند.

GetAvailPageFile-

مقدار در دسترس PageFile را باز می‌گرداند.

GetTotalVirtual-

مقدار کل حافظه مجازی را باز می‌گرداند.

GetAvailVirtual-

مقدار در دسترس حافظه‌های مجازی را باز می‌گرداند.

مثال:

```
PlugInRun("Tweak","GetTotalPhys")  
PlugInGet("Tweak","T-Phys")  
Message("Your system physical memory is: ","T-Phys")
```

نکته: جهت تبدیل بایت به مگابایت کافیست بایت را ۲ بار بر 1024 تقسیم کنیم.

FreePlay PlugIn

این PlugIn که اخیرا به مجموعه ی PlugIn های MMB اضافه شده است قابلیت های منحصر به فرد در عین حال بر کاربردی را به بدنه برنامه MMB اضافه می کند. همانند قسمت Tweak قصد داریم فرامین این PlugIn را فهرست وار شرح دهیم. همانطور که اطلاع دارید حق استفاده از تکنولوژی های Mp3 اخیرا از حالت رایگان به صورت اشتراکی تغییر کرده و محدودیت هایی را ایجاد نموده است. در همین راستا استفاده از این PlugIn می تواند راه حلی جهت این محدودیتها باشد.

SetItemHandler

توسط این فرمان PlugIn فشردن کلید هایی را شبیه سازی و آنها را به MMB ارسال می کند. بنا-براین از قبل باید اینکده را که میانبرهایش عینا همانند کلید هایی باشد که PlugIn شبیه سازی می کند.

مثال:

```
Var$='CTRL,SHIFT,ALT,U'  
PlugInSet ("PlugIn","var$")  
PlugInRun ("PlugIn","SetItemHandler")
```

فرمان بالا از ترکیب ۳ کلید shift و ctrl و alt به همراه یک کلید دیگر از صفحه کلید استفاده می -کنند. چنانچه قصد نداشته باشید از کلید های سیستمی گفته شده استفاده کنید تنها کافیسیت تا نام آن کلید را خالی بگذارید، همانند مثال زیر:

مثال:

```
Var$='CTRL,,ALT,R'  
PlugInSet ("PlugIn","var$")  
PlugInRun ("PlugIn","SetItemHandler")
```

SendEvents

این فرمان ارسال رویداد از PlugIn را فعال یا غیر فعال می سازد. در حالت پیش فرض این ویژگی فعال می باشد. توصیه می شود که این ویژگی قبل از نمایان شدن کادرهای محاوره ای غیر فعال شود چرا که از انتخاب اشتباه کانون توسط برنامه ها جلوگیری می شود.

مثال:

```
Var=1  
PlugInSet ("PlugIn","var$")  
PlugInRun ("PlugIn","SendEvents")
```

Load

از این فرمان برای بارگذاری فایل های صوتی پشتیبانی شده استفاده می شود. فرمت های پشتیبانی شده WMA, mt3 و war می باشند.

مثال:

```
Var$='<srcDir>\Music\Back ground,wma'  
PlugInSet ("PlugIn","var$")  
PlugInRun ("PlugIn","Load")
```

Play

این فرمان فایل یا فایل های بارگذاری شده را اجرا می کند. چنانچه ویژگی **Send Event** فعال باشد **PlugIn** موقعیت اجرای فایل صوتی در حال اجراء را باز می گرداند.

مثال:

```
PlugInRun("PlugIn","Play")
```

Pause

سبب ایجاد وقفه موقت در اجرای فایل صوتی می شود. فشردن مجدد این فرمان سبب لغو وقفه می شود.

مثال:

```
PlugInRun("PlugIn","Pause")
```

Stop

اجرای فایل صوتی را به طور کامل متوقف می کند.

مثال:

```
PlugInRun("PlugIn","Stop")
```

SetPosition

موقعیت اجرای فایل صوتی را بیان می کند. مقدار دریافتی برحسب درصد می باشد.

مثال:

```
Var=40  
PlugInSet("PlugIn","var")  
PlugInRun("PlugIn","SetPosition")
```

GetPosition

فرمان بالا موقعیت اجرای فایل صوتی را بر حسب ۲ مقدار درصد و دقیقه باز می گرداند.

مثال:

```
PlugIn Run("PlugIn","Get Position")  
PlugInGet("PlugIn","var")  
PlugInGet("PlugIn","var$")
```

GetDuration

این فرمان زمان کل فایل صوتی را باز می گرداند. به همین ترتیب این فرمان ۲ مقدار را یکی برحسب ثانیه و دیگری برحسب دقیقه باز می گرداند.

مثال:

```
PlugIn Run("PlugIn","GetDuration")  
PlugIn Get("PlugIn","var")  
PlugInGet("PlugIn","var$")
```

SetVolume

این فرمان سبب تنظیم حجم صدا می شود. البته تغییر حجم صدا توسط این فرمان هیچ تاثیری بر حجم صدای اصلی ویندوز نمی گذارد. در این فرمان مقدار ورودی برحسب درصد بیان می شود.

مثال:

```
Var=70  
PlugInSet("PlugIn","Var")  
PlugInGet("PlugIn","Setvolume")
```

Getvolume

این فرمان حجم صدا را باز می گرداند. البته این مقدار همان مقدار تنظیم شده توسط فرمان `setvolume` می باشد.

مثال:

```
PlugInRun("PlugIn","Getvolume")
PlugInGet("PlugIn","var")
```

UseEqualizer

این فرمان ویژگی `Equalizer` را فعال یا غیر فعال می سازد. در حالت پیش فرض این ویژگی غیر فعال می باشد. پس از فعال کردن این ویژگی بایستی از فرمان `SetEQvalue` جهت دریافت مقدار باند های `Equalizer` استفاده کنیم. جهت فعال کردن ویژگی از پارامتر 0 و غیر فعال کردن از پارامتر 0 استفاده می کنیم.

مثال:

```
Var=1
PlugIn Set(PlugIn","var")
PlugIn Run(PlugIn","UseEqualizer")
```

SetEQvalue

از این فرمان جهت تعیین مقدار باند های `Equalizer` استفاده می شود. ابتدا توسط یک متغیر عددی شماره باند را مشخص می کنیم و سپس توسط یک متغیر رشته ای مقدار رابه `PlugIn` ارسال می کنیم. ترتیب باندها به صورت زیر است:

محدوده ی متغیر عددی بین 1 تا 10 می باشد و محدوده ی متغیر رشته ای بین 0 تا 30 است، مقدار پیش فرض

1-80H2	9-12000H2
2-170H2	10-14000H2
3-310H2	
4-600H2	
5-1000H2	
6-3000H2	
7-6000H2	
8-10000H2	

مثال:

```
VAR=1
VAR$' 30'
PlugInSet("PlugIn","var")
PlugInSet("PlugIn","var$")
PlugInRun("PlugIn","SetEQvalue")
```

SetEchoEffect

از این فرمان برای فعال یا غیر فعال کردن جلوه `Echo` استفاده می شود. جهت فعال کردن این ویژگی از پارامتر 1 و برای غیر فعال کردن آن از پارامتر 0 استفاده می شود. به طور پیش فرض این جلوه غیر فعال می باشد.

مثال:

```
Var=1
PlugInSet(PlugIn","var")
PlugInRun(PlugIn","SetEchoEffect")
```

Setflanger Effect

فرمان مذکور جلوه flanger را فعال یا غیر فعال می سازد. به طور پیش فرض این ویژگی غیر فعال است. توسط پارامتر 1 و 0 می توان این ویژگی را فعال یا غیر فعال کرد.

مثال:

```
Var=1  
PlugInSet("PlugIn","var")  
PlugInRun("PlugIn","SetflangerEffect")
```

SetReverbEffect

این دستور جلوه Reverb را فعال یا غیر فعال می سازد. در حالت پیش فرض این جلوه غیر فعال است. توسط مقادیر 1 و 0 به ترتیب می توانیم این ویژگی را فعال یا غیر فعال سازیم.

مثال:

```
Var=1  
PlugInSet("PlugIn","var")  
PlugInRun("PlugIn","SetReverbEffect")
```

ShowAnalogMeter

این فرمان صوت سنچ آنالوگ را فعال می سازد. منظور از صوت سنچ قسمتی است که به صورت بصری مقدار پیشرفت آهنگ را نمایان می کند. در حالت پیش فرض این قابلیت مخفی می باشد.

مثال:

```
PlugInRun("PlugIn","ShowAnalogMeter")
```

HideAnalogMeter

این فرمان صوت سنچ آنالوگ را مخفی می کند.

مثال:

```
PlugInRun("PlugIn","HideAnalogMeter")
```

MeterBackImage

فرمان بالا برای تعیین زمینه صوت سنچ آنالوگ می باشد. تنها فرمت پشتیبانی شده Bmp می باشد. توصیه می شود که حتی الامکان از تصویری استفاده کنید که اندازه اش با اندازه پنجره PlugIn مطابقت داشته باشد.

مثال:

```
Var$='My-back-image.bmp'  
PlugInSet("PlugIn","var$")  
PlugInRun("PlugIn","MeterBackImage")
```

ShowAnalogKnob

فرمان مذکور دستگیره ی گردی را جهت تغییر حجم صدا نمایان می سازد. این دستگیره همانند پیچ تنظیم صدا در دستگاه های پخش صوت می باشد. در حالت پیش فرض این ویژگی غیر فعال است.

مثال:

```
PlugInRun("PlugIn","ShowAnalogKnob")
```

HideAnalogKnob

این فرمان عکس فرمان قبل عمل می کند.

مثال:

```
PlugInRun("PlugIn"," HideAnalogKnob")
```

Knobcolor

از این فرمان جهت تعیین رنگ دستگیره گرد استفاده می شود. ابتدا بایستی همانند مثال مقدار رنگ را در کانال RGB به PlugIn ارسال کنیم.

مثال:

```
Var$='200,80,100'  
PlugInSet("PlugIn","var$")  
PlugInRun("PlugIn","Knobcolor")
```

KnobTickcolor

جهت تعیین شاخص دستگیره گرد می باشد. در این فرمان نیز بایستی مقدار رنگ را در کانال RGB به PlugIn ارسال کنیم.

مثال:

```
Var$='10,80,200'  
PlugInSet("PlugIn","var$")  
plugInRun("PlugIn","KnobTickcolor")
```

GetKnobposition

فرمان بالا موقعیت دستگیره گرد را در واحد درصد باز می گرداند.

مثال:

```
PlugInRun("PlugIn","GeTKnobPosition")  
PlugInGet("PlugIn","var")
```

SetKnobPosition

این فرمان بر اساس مقدار ورودی که در واحد درصد می باشد موقعیت دستگیره گرد را مشخص می کند.

مثال:

```
Var=40  
PlugInSet("PlugIn","var")  
PlugInRun("PlugIn","SetKnobPosition")
```

MMBDLL

فراخوانی فایل های DLL دیگر فرایندی است که تقریباً تمام نرم افزارها از آن بهره می برند. به واسطه این PlugIn ارزشمند، MMB نیز از این پس قادر است با فایل های DLL ارتباط برقرار کند. برای استفاده از DLL های دیگر تنها کافیست به راهنمای همراه آن DLL مراجعه کنیم و نحوه فراخوانی توابع از آن را استخراج کنیم. بدین ترتیب برخی از فرامینی را که ممکن است هر کاربر MMB به هنگام طراحی برنامه خود آرزوی داشتن آن رداشته باشد از این پس در دسترس شماست. همان طور که در ابتدای مبحث PlugIn شرح داده شد DLL (Dynamic Link Library) مجموعه ای از توابع است که مورد فراخوانی قرار می گیرد. هر فایل DLL رویه ای خاص خود دارد و این معین می کند چگونه بایستی از این فایل DLL استفاده شود. امر فراخوانی و برقراری ارتباط با DLL ها تحت عنوان APL (Application Programming Interface) شناخته می شود. فراخوانی فایل های DLL و استفاده از توابع ارائه شده توسط DLL عمده مطلب API می باشد و این قسمت در زبان های برنامه نویسی از قسمت های اساسی برنامه نویسی تحت ویندوز می باشد. بنابراین می توانیم توسط محیط های برنامه نویسی از قبیل VB و Delphi و هم اکنون MMB به توابع فایل DLL دسترسی داشته باشیم. اصلی ترین توابع API در سه فایل زیر که درون هر سیستم مبتنی بر ویندوز می باشد موجود است:

۱- User32.dll برای کنترل جزئیات بصری و رابط کاربر می باشد.

۲- Kernel32.dll برای کنترل فایل ها و مدیریت حافظه می باشد.

۳- Gdi32.dll فرامین گرافیکی را ارائه می دهد.

اما در عین حال تعداد بیشتری DLL درون سیستم عامل ویندوز نهفته است که می توانیم با دانستن نحوه استفاده به فراخوانی آنها بپردازیم. یک مزیت دیگر استفاده از DLL اینست که این فایل ها قبلاً توسط زبانهایی نظیر ++C و C تولید شده اند و تا حد امکان بهینه سازی شده می باشند. بنابراین این قبیل DLL ها سریعتر و قابل اطمینان تر از سایر DLL ها می باشند. با فراخوانی DLL ها تقریباً می توان تمام اعمالی را که ویندوز انجام می دهد را انجام داد، از خاموش کردن کامپیوتر گرفته تا نصب یک چاپگر. به خاطر طبیعت ویندوز همزمان چندین برنامه می تواند فایل های DLL ویندوز را فراخوانی نمایند. جهت بررسی بیشتر WinAPI می توانید به پایگاههای زیر مراجعه کنید:

- <http://msdn.microsoft.com/library/>
- <http://www.mentalis.org/apilist/apilist.php/>
- <http://www.mentalis.org/>

فرامین

تنها چهار فرمان توسط MMBDLL ارائه می شود که می توانید از آنها برای فراخوانی DLL ها استفاده کنید.

• Path

این فرمان مشخص می کند که فایل DLL مورد نظر در کجا واقع شده است. چنانچه قصد فراخوانی DLLهای ویندوز را داشته باشید ذکر تنها نام فایل کافی می باشد.

مثال:

```
Path$='user32.dll'  
Plugln("plugln","path$")  
Plugln("plugln","path")
```

• Functions

این فرمان مشخص می کند که چه تابعی از DLL مد نظر ماست.

مثال:

```
Function$='MyDLLFunction'  
PluglnSet("Plugln","Function$")  
PluglnRun("Plugln","Function")
```

• Parameters

جهت ارسال پارامترهای مورد نیاز DLL از MMB می باشد. پارامترها می توانند از نوع متغیرهای عددی باشند و هم به صورت رشته ای. چنانچه احتیاج به ارسال بیشتر از یک پارامتر باشد بین پارامتر از علامت "|" استفاده می کنیم. در هنگام عدم نیاز به یک پارامتر می توان جای آن را خالی گذاشت.

مثال:

```
Parameters$='0|Test message|TestTitle'  
PluglnSet("plugln","Parameters$")  
PluglnRun("Plugln","parameters")
```

• Registration

از آنجاییکه این Plugln یک برنامه اشتراکی می باشد از این دستور برای ثبت برنامه استفاده می شود.

مثال:

```
Reg$='RegGde'  
PluglnSet("Plugln","Reg$")  
PluglnRun("plugln","Registration")
```

ممکن است فراخوانی تابعی از DLL خروجی به همراه داشته باشد که برای دریافت خروجی می توانیم از دستور PluglnGet استفاده کنیم. مقدار خروجی همیشه به صورت متغیر رشته ای می باشد.

مثال:

```
PluglnGet("plugln","Result$")  
Message("The DLL Sent these results: ","Result$")
```


SlideShowPlugIn

این PlugIn نیز یکی دیگر از PlugIn های پر کاربرد MMB است. هدف از استفاده از این PlugIn نشان دادن خودکار تعدادی عکس با وقفه های زمانی معین است. این PlugIn از تمامی ابزارهایی که ممکن است در تولید یک SlideShow مورد نیاز واقع شود برخوردار است. بر همین اساس در ادامه قسمت شرح PlugIn های MMB قصد داریم تا به شرح توابع این PlugIn قدرتمند بپردازیم.

SetPictureDirectory

این فرمان پوشه های حاوی عکس ها را مشخص می کند. در استفاده از این فرمان می توان زیر شاخه ها را نیز معین کرد، برای اینکار بایستی شماره ی زیر شاخه را توسط متغیری به PlugIn ارسال نماییم. این PlugIn تنها از فرمت های Ico، Bmp، Jpg، Wmf، حمایت می کند. چنانچه شماره ی زیر شاخه را صفر قرار دهیم هیچ زیر شاخه ای جستجو نمی شود.

SelectPictureDirectory

فرمان گفته شده یک کادر باز گشایی فایل را جهت انتخاب پوشه عکسهای مورد نظر نمایان می سازد. در این قسمت نیز می توانیم زیر شاخه ها را نیز مشمول جستجو کنیم.

LoadPictureList

این تابع لیست عکس ها را از یک فایل متنی خارجی بار گذاری می کند و به درون یک لیست درونی انتقال می دهد. فایل خارجی می تواند هر پسوندی داشته باشد. چنانچه عکس مشخص شده در لیست موجود نباشد، PlugIn از بار گذاری آن به درون لیست درونی صرف نظر می کند. این تابع ۲ مقدار 1 به معنی پیدا کردن لیست و 0 به معنی عدم یافتن لیست را باز می کند.

SetFullScreenDirectory

این فرمان محتوی پوشه عکس را به صورت تمام صفحه (FullScreen) نشان می دهد. شایان ذکر است که در بزرگنمایی نسبت وجوه رعایت می شود.

StartSlideShow

توسط این فرمان، بر اساس تنظیم های انجام شده نمایش عکس ها شروع می شود.

StopSlideShow

سبب توقف نمایش عکس ها می شود.

EnableToolTip

این فرمان ویژگی ToolTip را برای پنجره PlugIn فعال می سازد.

DisableToolTip

فرمان مذکور ویژگی **ToolTip** را غیر فعال می کند.

SetToolTipText

این فرمان متن **ToolTip** را مشخص می کند.

EnableMouseControl

توسط این فرمان می توان از دکمه های موس جهت مشاهده عکسها استفاده کرد. به طور پیش فرض این ویژگی فعال است، اما از آنجاییکه فرمان **DisableMouseControl** موجود می باشد ضرورت وجود این فرمان آشکار می شود.

DisableMouseControl

قابلیت کنترل **SlideShow** توسط موس را غیر فعال می کند.

ExitProject

به خاطر برخی از مشکلات موجود در **MMB** برای کنترل **PlugIn** ها، ممکن است برخی اوقات فرمان **Exit** به درستی عمل نکند، بر همین اساس این فرمان به لیست فرامین این **PlugIn** اضافه شده است.

SetBackgroundColor

از این دستود برای تعیین رنگ زمینه **PlugIn** استفاده می شود. ابتدا بایستی توسط متغیری مقدار رنگ را در کانال **RGB** به **PlugIn** ارسال کنیم.

SetBackgroundPicture

از این دستور جهت مشخص کردن عکس زمینه **PlugIn** استفاده می شود.

EraseBackgroundPicture

فرمان بالا عکس زمینه پنجره **PlugIn** را پاک می کند.

SetTransparentColor

از این فرمان جهت برداشتن رنگ پس زمینه عکس زمینه استفاده می شود. توسط این فرمان می توانید شمایل پنجره **PlugIn** را تغییر دهید. توصیه می شود که حتی الامکان از تصاویر با فرمت **Bmp** استفاده نمایید چرا که تشخیص رنگ زمینه توسط **PlugIn** در فرمت **Bmp** بهتر و راحتتر صورت می گیرد. در فرمت **Jpg** به خاطر فشردگی ممکن است نتیجه مطلوب حاصل نشود.

SetStrechBackground

این فرمان عکس پس زمینه را به فرمی در می آورد که تمام پنجره **PlugIn** را بپوشاند.

SetTileBackground

چنانچه بخواهیم عکس پس زمینه PlugIn را به حالت کاشی وار درآوریم از این دستور استفاده می کنیم.

SetCenterBackground

این فرمان عکس پس زمینه را در وسط پنجره PlugIn قرار می دهد. چنانچه اندازه عکس مربوطه از اندازه پنجره PlugIn بزرگتر باشد تنها قسمتی که مطابق پنجره PlugIn است نمایان می شود.

SetScalePicture

از این فرمان جهت تغییر مقیاس عکس ها به هنگام نمایش در پنجره PlugIn استفاده می شود. در این حالت نیز نسبت وجوه رعایت می شود.

SetOriginalPicture

اگر بخواهیم عکس ها در اندازه اصلی خود در پنجره PlugIn نمایان شوند از این دستور استفاده می کنیم.

SetFillPicture

توسط این فرمان می توان طوری عمل کرد که عکس پس زمینه کاملاً پنجره PlugIn را بپوشاند. البته در این حالت نسبت وجوه رعایت نمی شود.

InitSlideShow

این فرمان برای بنیان نهادن SlideShow می باشد. چنانچه در حین اجرای SlideShow از این فرمان استفاده کنیم نمایش عکسها مجدداً از اولین عکس از سر گرفته می شود.

HidePlugIn

پنجره PlugIn را مخفی می کند.

ShowPlugIn

پنجره PlugIn را نمایان می سازد.

ShowNextPicture

برای نشان دادن عکس بعدی نسبت به عکس جاری از این دستور استفاده می شود.

ShowPreviousPicture

برای نشان دادن عکس قبلی نسبت به عکس جاری از این فرمان استفاده می شود.

ShowPicture

همان طور که گفته شد عکسهای مورد نظر جهت نمایش در **SlideShow** در یک لیست درونی و به ترتیب قرار می گیرند. حال چنانچه بخواهیم یک عکس خاص را از این لیست نمایش دهیم از فرمان بالا استفاده می کنیم.

ShowFullScreenPicture

این فرمان عکس جاری را به صورت تمام صفحه نشان می دهد. ضمناً در این حالت نسبت وجوه رعایت می شود. همچنین کلیک کردن بر روی عکس در حالت تمام صفحه سبب لغو حالت تمام صفحه می شود.

GetCurrentPicture

این فرمان مسیر عکس جاری را باز می گرداند. همچنین این فرمان شماره عکس جاری را باز می گرداند.

GetAllPicture

این فرمان مسیر تمامی عکسها به همراه تعداد عکسها باز می گرداند.

GetPictureDiscription

این فرمان خصوصیت یک عکس را از یک فایل متنی باز می گرداند. چنانچه یک فایل متنی هم نام با عکس مورد نظر در پوشه عکسها باشد این فرمان محتویات این فایل متنی را نمایان می کند. چنانچه هیچ فایل متنی برای نگهداری خصوصیت عکس موجود نباشد این دستور عدد صفر را باز می گرداند.

PrintCurrentPicture

این فرمان عکس جاری را چاپ می کند. به هنگام استفاده از این فرمان **PlugIn** کادر چاپ را نمایان می سازد.

SetScriptKey-PicDone

این فرمان همان **EventHandle** است. توسط این فرمان می توانیم کلید هایی را مشخص کنیم تا به هنگام بارگذاری هر عکس توسط فشرده شدن مجازی این کلیدها شی اسکریپتی از برنامه فرا خوانی می شود.

تولید نهایی

Multimedia Builder

Path Replace ◀

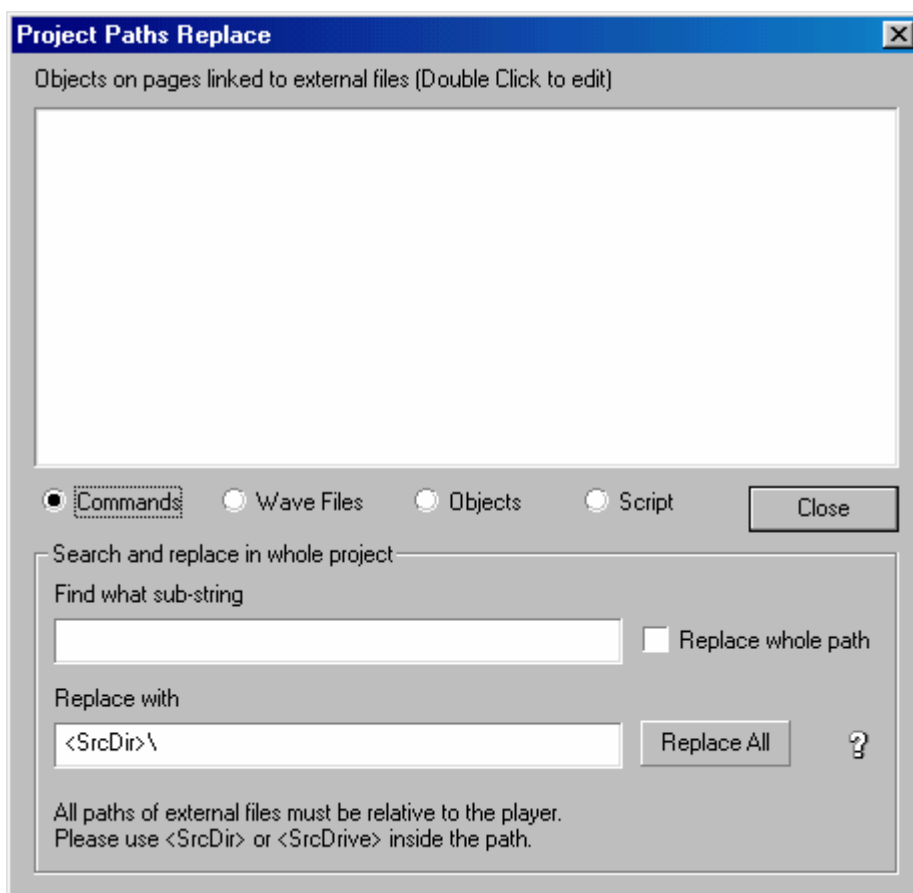
Text Replace ◀

Compile ◀



تولید نهایی

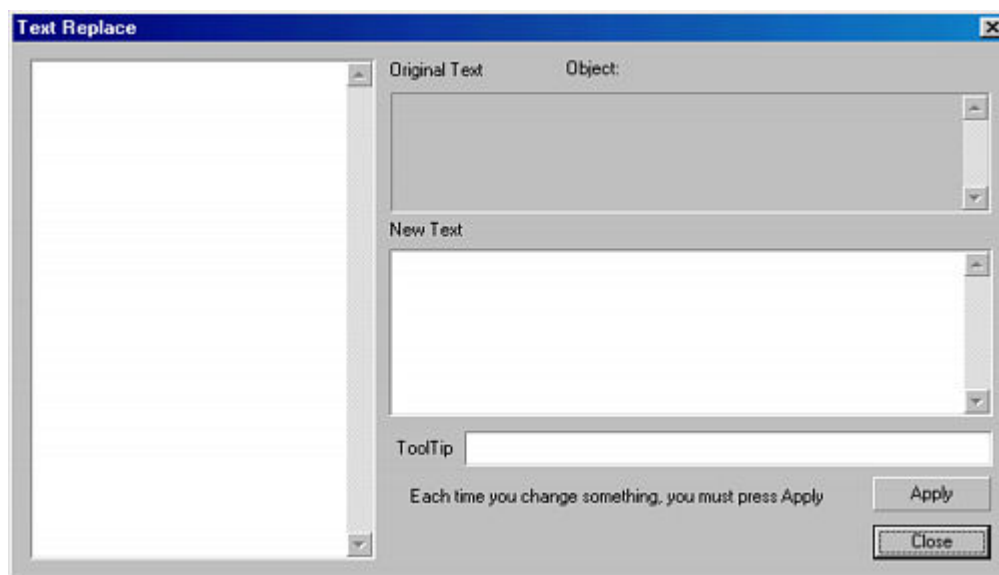
هنگام تولید نهایی محصول فرا رسیده، پس از مدت‌ها تلاش و برنامه‌نویسی منتظرید تا صورت نهایی پروژه‌تان را ببینید. اما قبل از اینکه دکمه **comple** را فشار دهید لازم است تا برای آخرین بار به بررسی برخی از قسمت‌ها و احتمالاً تنظیم صحیح آنها پردازید. اولین قسمتی که بایستی به سراغ آن برویم **path replace** است. فایل‌های ویدئویی، موزیک‌ها و... که فایل‌هایی خارجی محسوب می‌شوند از بیرون با پروژه ما در ارتباط هستند و طبیعتاً جزء فایل **MBD** نمی‌باشند چرا که باعث بالا بردن حجم فایل و در نتیجه کندی اجرای پروژه می‌شوند. با توجه به آن چه گفتیم این بسیار مهم است که مسیر این فایل‌های خارجی به گونه‌ای تنظیم شود که کاربران نهایی در هنگام استفاده از پروژه ما با هیچ مشکلی روبرو نشوند، به همین منظور برای تنظیم این مسیرها ابزار **Path replace** طراحی شده است. در ادامه به شرح قسمتهای مختلف این ابزار می‌پردازیم. شکل زیر پنجره ابزار **path replace** را نشان می‌دهد.



دربالای این پنجره کادری موجود است که اقلام خارجی را نشان می‌دهد. در وسط این کادر چهار گزینه Commands (فرامین)، Object، Wave files (اشیاء) و Script موجود است. که هر کدام بسته به نوع خود فایل‌های خارجی اتصال یافته از همان نوع را نشان می‌دهد. حال جهت تنظیم درست مسیر بایستی بر روی هر مؤلفه کلیک کنیم بلافاصله پس از این عمل قسمتی از مسیر که بایستی تصحیح شود در کادر ورودی که عنوانش Find what sub- string است ظاهر می‌شود. در زیر همین کادر هم کادر ورودی

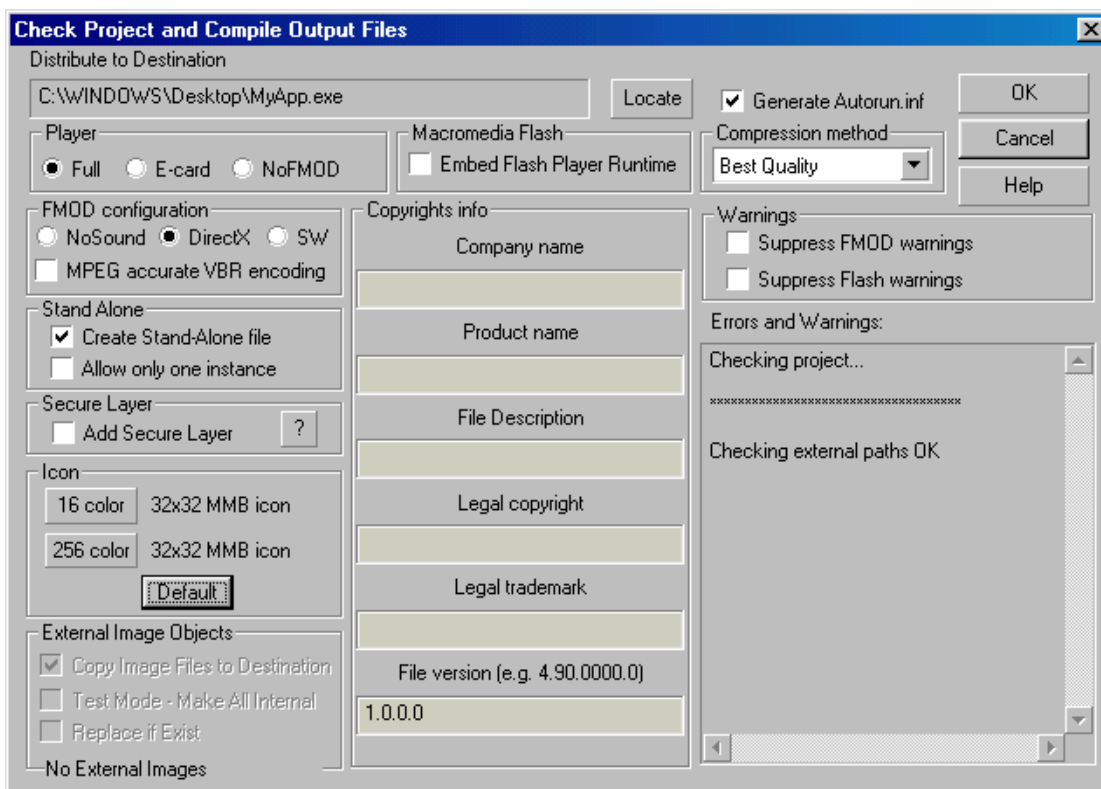
دیگری موجود است که عنوان آن Replace with می‌باشد و کاربرد آن این است که می‌توانیم اصلاح مسیر را در این کادر انجام دهیم. با توجه به توضیحاتی که در بخش‌های قبل راجع به ماکروهای مسیر گفته شد، اکنون باید بدانید که بر حسب نیازتان مسیر را چگونه تصحیح کنید. هنگامیکه تغییرات لازم را صورت دادید تنها کافیست تا دکمه Replace All را برای اعمال تغییرات فشار دهیم، اما یک گزینه دیگر هم مانده که بایستی راجع به آن توضیحاتی گفته شود. در کادر ورودی اول یک گزینه با عنوان Replace whole path قرار دارد و کار آن اینست که تصحیحی را که ما در کادر دوم صورت دادیم بر همه‌ی مسیرها اعمال کند، نه فقط یک مسیر تنها که ما بر روی آن کلیک کرده‌ایم. البته بایستی در استفاده از این گزینه دقت کنیم چرا که یک سهل‌انگاری ممکن است ساعت‌ها وقتمان را هدر دهد. برای مثال چنانچه پروژه ما متشکل از ۳۰۰ فایل خارجی باشد و نوع فایل‌ها مختلف باشد و همگی در پوشه‌های مختلف نهاده شده باشند ممکن است نتوانیم مسیرها را یکجا تصحیح کنیم. همین‌جا اگر با یک اشتباه کوچک تمام مسیرها را یکسان تغییر دهیم دیگر راه بازگشت سریعی نمی‌یابیم و بایستی تک تک مجدداً مسیرها را تصحیح کنیم و این خود مستلزم صرف وقت زیادی می‌باشد.

پس از اینکه خیالمان از بابت مسیرها راحت شد باید به سراغ قسمت Text replace برویم که به کمک آن می‌توانیم متون به کار رفته در پروژه را در صورت لزوم تصحیح کنیم.



در قسمت سمت چپ پنجره این ابزار لیستی از اشیاء حاوی متن ارائه شده که با کلیک بر روی آنها می توان آنها را تغییر داد. درست راست و بالا کادری قرار دارد که محتوی اصلی را نشان می دهد و در صورت تغییر نادرست متن ها می توانیم از این قسمت برای بازگشتن به حالت اصلی استفاده کنیم. در زیر این کادر پنجره ای جهت انجام تغییرات تعبیه شده که متون جدید را بایستی در آن با متون قدیمی جایگزین کنیم. ضمناً در صورت لزوم می توانیم هنگام تغییر متون tooltip آنها را هم تنظیم نماییم. در نهایت با فشردن دکمه ی Apply تغییرات صورت گرفته به پروژه اعمال می شود، اما توجه کنید که برای هر تغییر مجزا باید دکمه ی Apply را فشار دهیم و نمی توانیم اول همه ی متن ها را تغییر دهیم و این کلید را فشار دهیم.

هنگامی که از درستی مسیرها و متون پروژه اطمینان حاصل کردیم باید سری به فایل های خارجی بزنیم و از سالم بودن آنها و همچنین محل قرارگیری آنها بر اساس مسیرهای تنظیم شده اطمینان حاصل کنیم. خسته شدید؟! نگران نباشید دیگر نوبت قسمت Compile است. به هنگام کامپایل MMB بر اساس تنظیمات انجام شده فایل exe نهایی را تولید می کند. بر اساس شواهد موجود MMB فایل اجرایی نهایی را با استفاده از برنامه upx که یک فشرده ساز می باشد فشرده می کند که پس از تشریح قسمت کامپایل در رابطه به آن مطالبی را خواهیم گفت. اما با فشردن دکمه ی Compile کادری همانند شکل زیر نمایان می شود.



اما قسمت‌های این کادر به چه صورت است:

Distribute to destination –

این قسمت برای مشخص کردن مسیر فایل اجرایی نهایی می‌باشد با فشردن دکمه Locate کادر محاوره‌ای جهت تعیین کردن نام فایل و مسیر جهت ذخیره آن ظاهر می‌شود.

Generate AutoRun.inf –

این گزینه باعث می‌شود تا فایل inf که لازمه CD های AutoRun (خود اجرا) می‌باشد به همراه فایل نهایی تولید شود. در قسمت ضمیمه می‌توانید توضیحات مفصلی در رابطه با فایل AutoRun.inf بیابید.

Player –

این قسمت نوع Player (اجراکننده) را مشخص می‌کند که براساس نیاز می‌توانیم آن را تعیین کنیم.

• Full

یک اجرا کننده کامل می‌باشد و تمامی کتابخانه‌ها و Resource ها در آن موجود است. شما در اکثر مواقع از این قسمت استفاده می‌نمایید.

• E- card

یک فایل اجرایی با حجم کمتر تولید می‌کند که برای ارسال توسط پست الکترونیکی مناسب می‌باشد اما باید توجه داشت که این نوع محدودیت‌هایی را دربردارد. محدودیت‌هایی از قبیل:

۱- عدم حمایت از MP3

۲- عدم داشتن زمینه تمام صفحه و آهنگ زمینه

۳- و دیگر محدودیت‌های جزئی

• No FMOD

این گزینه از به الحاق در آوردن موتور FMOD به همراه فایل اجرایی جلوگیری می‌کند. این گزینه زمانی کار آمده است که صدا و صوت نقشی در پروژه ما نداشته باشد. FMOD یک موتور اجرا کننده صوتی بسیار قوی با کیفیت صوتی بسیار بالا می‌باشد.

Maromedia flash –

اعمال این گزینه اجرا کننده فلش (Flash player) را به همراه فایل اجرایی صادر می‌کند.

FMOD Configuration–

این قسمت جهت تنظیم موتور FMOD می‌باشد در این قسمت می‌توان مشخص کرد موتور FMOD چگونه فایل‌های صوتی را اجرا کند.

Copyrights Info –

در این قسمت مؤلف می‌تواند اطلاعاتی از قبیل: company name (نام شرکت)، product name (نام محصول)، file discription (توصیف فایل)، Legal CopyRight (کپی رایست قانونی)، Legal trade mark (مارک تجاری قانونی)، File version (شماره نسخه فایل) را در کادرهای مربوطه حداکثر به طول ۲۵ حرف برای هر قسمت وارد نماید.

Stand Alone –

خروجی استاندارد MMB یک اجرا کننده (Player) و فایل MBD مربوطه می‌باشد که به اجرا کننده اتصال یافته است. با انتخاب گزینه Create stand alone file کامپایلر تنها یک فایل اجرایی (*. Exe) حاوی اجرا کننده و فایل *.mbd را تولید خواهد کرد. گزینه Allow only one instance باعث می‌شود تا تنها یک نسخه از فایل اجرایی در آن واحد اجرا شود.

Secure layer –

با فعال نمودن این قابلیت می‌توان از مشاهده کردن اسکریپت‌ها و متون به کار رفته در برنامه توسط ویرایشگر hex جلوگیری نمود. باید توجه کرد که با اعمال این گزینه از سرعت فایل اجرایی تا حدودی کاسته می‌شود. پس تاکید می‌کنیم تنها در صورت لزوم از این ویژگی استفاده کنید.

Icon –

در این قسمت می‌توانیم Icon فایل اجرایی را مشخص کنیم. برحسب نیاز می‌توانیم یا از Icon های color 16 (۱۶ رنگ) استفاده کنیم یا از color 256 (۲۵۶ رنگ).

External Image objects –

چنانچه عکس‌های موجود در پروژه ما به صورت خارجی باشند این قسمت فعال می‌شود. Copy Image file to destination تصاویر را با توجه به مسیر فایل اجرایی به مقصد کپی می‌کند. گزینه

Test mode- Make all internal برای حالتی است که بخواهیم جهت امتحان پروژه فایل اجرایی نهایی را تولید کنیم. و نهایتاً Replace if exist برای جایگزینی عکسها در صورت موجود بودن آنها در پوشه مقصد است.

Compression method –

برای نشر فایل‌ها در اینترنت و یا داشتن مسئله در رابطه با حجم فایل نهایی می‌توان از گزینه‌های ارائه شده در این قسمت که هرکدام سطحی خاص از فشردگی را ارائه می‌کنند استفاده کرد. کیفیت فایل تولید شده بر اساس این گزینه‌ها از گزینه پایین به بالا بهتر می‌شود.

Warnings –

ممکن است موتور FMOD و همچنین Flash player در هنگام اجرای پروژه اخطارهایی را متذکر شوند که توسط این قسمت می توان از نمایان شدن آنها جلوگیری کرد.

Errors and warning –

این قسمت اشکالات و خطاهایی را که به خاطر عدم توجه ما هنوز در پروژه موجود است را بازگو می کند. اشکالاتی نظیر مسیرهای اشتباه و ... ، چنانچه مشکلی نباشد در این کادر عبارت `checking external path ok` ظاهر میشود.

ضمیمه

Multimedia Builder

نکته ها

پسوندهای معروف

خروجی MMB چگونه فایلی است؟

مقدمه ای بر AutoRun.Inf

RunDLL32.exe

میانبرهای پیش فرض MMB

واژه نامه



نکته ها:

در این قسمت قصد داریم تا چند نکته ی کلیدی را جهت تولید یک پروژه ی هر بهتر ارائه کنیم.

- رفع عیب (Debug)

احتمالا تولید هیچ پروژه ای خالی از اشتباه نیست و همین مطلب اشکال زدایی (Debug) پروژه را اجتناب ناپذیر ساخته است. عدم اشکال زدایی ممکن است منجر به هدر رفتن سرمایه و ساعت ها کار شود. اولین اشکال زدایی توسط خود مؤلف صورت می گیرد و چنانچه مؤلف در آزمایشات خود اشکالی را نیابد نسخه آزمایشی (بتا) را در اختیار چند نفر جهت افزایش دامنه آزمایشات قرار می دهد تا احوانا اشکالات دیگر احتمالی بروز یابند. چنانچه هیچ خطایی یافت نشود مؤلف برنامه نسخه اصلی برنامه (آلفا) را ارائه می دهد. MMB نیز از این قاعده مستثنا نمی باشد و ابزار رفع عیب را در اختیار مؤلف قرار می دهد. مد رفع عیب از منوی Debug -> Setting قابل دسترس است. در هنگام رفع عیب MMB لیستی از متغیرها و محتوی آنها را جهت بررسی بهتر ارائه می دهد. همچنین جزئیات دیگر برنامه در لیستی دیگر در پایین مد رفع عیب ثبت می شود. البته باید گفت که سرعت اجرای برنامه در حالت رفع عیب تا حدودی کم می شود.

- توزیع برنامه (Distribution)

هنگامیکه مراحل رفع عیب برنامه به پایان رسید هنگام عرضه محصول است بدین منظور باید روشی را برای انتقال برنامه به سیستم کاربران نهایی بر گزینیم. به ۲ طریق می توان برنامه را در اختیار کاربر قرار داد:

۱. قراردادن آن در اینترنت برای دریافت:

این روش بیشتر برای فایل های کم حجم استفاده می شود. این روش از امنیت و کارایی بیشتری برخوردار است.

۲. قرار دادن آن بر روی CD:

این روش برای فایل های با حجم بالا و عرضه محصول در فروشگاه ها مناسب می باشد.

اما گذشته از روش انتقال بایستی ملزومات را جهت اجرای برنامه مشخص نمود و سپس آنها را به اطلاع کاربر رسانید. منظور از ملزومات همان مقدار حافظه مورد نیاز، فضای دیسک سخت مورد نیاز، پردازشگر مورد نیاز و سازگارکننده گرافیکی مورد نیاز به همراه سایر ملزومات نرم افزاری از قبیل DirectX

و ... است.

- افزایش سرعت

سرعت اجرای بالا یکی از دغدغه های اصلی تولید کنندگان و کاربران می باشد، به همین منظور باید تا حد امکان طوری عمل کرد که سرعت اجرای برنامه افزایش یابد. در زیر چند توصیه برای بهبود سرعت آمده است:

۱- حتی الامکان سعی کنید از فایلها به خصوص تصاویر به صورت خارجی استفاده نمایید .

۲- در طراحی پروژه های بزرگ سعی نمایید که پروژه را به قسمت های ریز تری تقسیم نمایید و طوری برنامه ریزی کنید که به هنگام اجرای برنامه تنها قسمت های مورد نیاز فراخوانی شوند.

۳- به هنگام کامپایل پروژه در صورت امکان از قسمت فشرده سازی استفاده نکنید چرا که فشردگی کاهش سرعت را به همراه دارد. اگر قصد انتشار محصول خود بر روی CD را دارید در صورت امکان از فشرده سازی فایل اجرایی نهایی صرف نظر کنید تا سرعت اجرای برنامه بالاتر رود.

۴- استفاده صحیح از حلقه ها و وقفه ها سبب بهبود و بهینه سازی سرعت می شود.

۵- از الحاق کردن (Embedding) فایل ها تا حد امکان پرهیز کنید.

۶- در طراحی پروژه از تکلف و زیاده روی در استفاده از تصاویر و قسمتهای گرافیکی پرهیز نمایید. زیرا کارایی پروژه مهمتر از ظاهر آن است.

۷- چنانچه در پروژه از صوت و موسیقی استفاده نمی کنید از به انضمام در آوردن موتور صوتی FMOD خود داری کنید.

۸- حق تقدم (Priority) استفاده از CPU را متناسب با عملکرد پروژه تنظیم نمایید.

- هنگام شروع برنامه (Initialization)

برخی از Plug-In های MMB و فرامین کنترل اشیا را نمی توان در هنگام شروع برنامه مورد استفاده قرار داد. به عنوان مثال در Tweak فرامین INI در هنگام شروع به درستی عمل نمی کنند.

- استفاده از برنامه RealDRAW

توصیه می شود که برای طراحی قالب ها و دکمه مورد نیاز پروژه از این برنامه استفاده نمایید، چرا که این برنامه کاملاً با محیط MMB سازگار می باشد. همچنین این برنامه در عین سادگی از قدرت مناسبی برخوردار است.

پسوندهای معروف

ANI

این پسوند معرف فایل **Animated Cursor** است که برای اولین بار در ویندوز ۹۵ معرفی شد. پس از آن در نسخه‌های بعدی ویندوز مورد استفاده قرار گرفت.

PSD

فایل طرح مورد استفاده در برنامه **Photoshop** با پسوند **PSD**

نشان داده می‌شود.

DLL

این پسوند معرف فایل‌هایی است که توانایی برقراری ارتباط با کتابخانه‌های اینترکتیو (تعاملی) را دارند. این فایل‌ها معمولاً در سیستم عامل‌های ویندوز و **OS/2** وجود دارند و تنها وقتی که برنامه‌های مربوط به آنها اجرا می‌گردند در حافظه بارگذاری می‌شوند.

HTML , HTM

این پسوند معرف فایل **HyperText Mark up Language** است. **HTML** زبانی است که از آن برای طراحی محتوی وب، استفاده می‌شود. بنابراین فایل‌های **HTML** اساساً با صفحات وب در ارتباط هستند.

INI

پسوند فوق معرف فایل راه اندازی است که در برنامه‌های کاربردی مختلف وجود دارد. این نوع فایل وقتی اجرا می‌شود که کامپیوتر راه اندازی شده یا برنامه فعال گردد این فایل برای ذخیره پیکربندی استفاده می‌شود.

JPG

این پسوند معرف فشرده سازی **Joint Photographic Experts Group** می‌باشد. از این فرمت برای تصاویر طرح بیتی و رنگی استفاده می‌شود این فرمت بعد از تعیین استاندارد فشرده سازی پدید آمده و نامگذاری شده است. از این پسوند گرافیکی در اکثر برنامه‌های کاربردی و گرافیکی استفاده می‌شود.

EXE

این پسوند نشان دهنده‌ی فایل اجرایی است که همراه با برنامه‌های کاربردی برای کامپیوترهای مبتنی بر Dos اجرا می‌شود. یک فایل اجرایی فرمتی است که کامپیوتر می‌تواند بدون کمک کاربر اجرا نماید.

GIF

این پسوند معرف فایل **Graphic Interchang Format** می‌باشد. پسوند مذکور اولین بار توسط شرکت **Compuserve** ارائه شده است. در حقیقت **GIF** یک فرمت فشرده‌سازی است که اطلاعات دیجیتالی را فشرده نموده و به تصاویر گرافیکی ثابت و یا متحرک تبدیل می‌کند.

AVI

این پسوند نشان دهنده‌ی یکی از فرمت‌های فایل تصویری به نام **Audio Video Inter Leave** می‌باشد. در ویندوز از این فرمت همراه با برنامه‌ی **Microsoft Video** استفاده می‌شود.

AVI یا MPEG

فرمت **AVI** در تمام نسخه‌های سیستم عامل ویندوز شناسایی و حمایت می‌شود. اما فرمت **MPEG** مستقیماً توسط سیستم عامل حمایت نمی‌شود و لازم است تا اجرا کننده و راه انداز آن در سیستم عامل نصب شود. اما با این وجود تمام نسخه‌های ویندوز ۹۸ به بعد از فرمت **MPEG** پشتیبانی می‌کنند و این امر به واسطه برنامه **Windows Media Player** صورت می‌گیرد. بنابراین این بهترین راه برای اجرای فرمت **MPEG** بر روی سیستم‌های قدیمی نصب **Media Player** جدید و سازگار با آن نسخه از ویندوز می‌باشد. در مقایسه بین این دو فرمت، **MPEG** از کیفیت بالاتری برخوردار است. اما با این حال احتیاج به استفاده بیشتری از **CPU** نسبت به فرمت **AVI** دارد. اگر به بررسی بیشتر فرمت **AVI** پردازیم متوجه خواهیم شد که این فرمت نیز گونه‌های مختلفی دارد و مسبب این حالات **Codec** ها می‌باشند.

Codec چیست؟

در حالت طبیعی می‌توانیم از فایل‌های **AVI** بدون فشرده‌سازی استفاده کنیم، اما با یک مشکل روبرو خواهیم شد و آن چیزی جز مشکل کمبود فضا نیست. تنها چند دقیقه کوتاه از یک فیلم با فرمت **AVI** در این حالت به چندین مگابایت فضا نیاز دارد. بنابراین پای نرم افزارهای فشرده‌سازی تصویری به میان می‌آید که با نام کلی **Codec** شناخته می‌شوند. موارد استفاده از **Codec** تنها برای فشرده‌سازی فایل تصویری نیست، بلکه به هنگام اجرای فایل تصویری فشرده‌سازی شده **Codec** مورد نیاز بایستی از قبل بر روی سیستم نصب شده باشد. همانند سایر شاخه‌های نرم افزار هم اکنون **Codec** های متعددی در دسترس می‌باشند که بسته به نوع هر کدام سطح خاصی از فشرده‌سازی امکان پذیر می‌شود. چنانچه قصد استفاده از فشرده‌سازی را دارید در صورتی که تردید داشته باشید که آیا آن فشرده‌سازی بر روی سیستم کاربر نهایی

موجود است یا خیر بایستی آن Codec را به همراه بسته نرم افزاری خود ارائه دهید. در حالت پیش فرض چند Codec در تمام ویندوز های 9x,2000 و XP از قبل نصب می شود. Codec هایی از نظیر Indeo, Cinepack ، Microsoft Video از این دست هستند. در ادامه به شرح مختصر Codec های موجود می پردازیم.

Microsoft Video1

Codec اصلی که به همراه نسخه های پیشین ویندوز ارائه شده است. از نظر کیفیت این Codec کیفیت بدی را ارائه می دهد، چرا که این Codec در زمانی که وضوح 256 رنگ حالت استاندارد بوده است تولید شده است. بهتر است از این نوع فشرده ساز صرف نظر کنیم.

Indeo 4, 5

این دو نسخه نیز محصول شرکت ایتل هستند اما تفاوتشان با نسخه قبلی در تکنولوژی به کار رفته در هر کدام می باشد. این دو Codec کیفیت بهتری را نسبت به موارد قبلی ارائه می دهند. اما در عین حال به سخت افزارهای قوی تری نیاز دارند. تقریباً تمام ویندوزهای از ۹۵ به بالا دارای این Codec می باشند.

MPEG-1

این فرمت کیفیت مطلوبی را به ارمغان می آورد. این فرمت بسیار مشهور است و فرمتی است که برای سی دی های تصویری (VideoCD) استفاده می شود، MPEG-1 است.

MPEG-2

کیفیتی که این فرمت ارائه می دهد بسیار عالی است. از این فرمت در DVD ها استفاده می شود. برای اجرای این فرمت به سخت افزارهای نسبتاً سریع و پیشرفته ای نیاز است.

MPEG-4

اگر سیر صعودی کیفیت فایل های ویدویی را بررسی کنید هنگامی که به این فرمت برسید در می یابید که این فرمت کیفیت بسیار عالی را فراهم می آورد. شرکت مایکروسافت نیز بر اساس همین فرمت، فرمت ASF را ارائه داده است.

حال که به این قسمت رسیده اید، بدون شک از خود سوال می کنید کدام فرمت مناسب است؟ پاسخ این سوال بستگی به خودتان دارد. ابتدا بایستی بررسی کنید که ملزومات پروژه شما چیست. آیا نیاز است که کیفیت بالای ویدویی استفاده شود یا خیر؟ آیا می خواهید که پروژه تان در تمام سیستم ها اجرا شود؟ اگر کیفیت مهم است بایستی از فرمت MPEG استفاده شود. اگر اجرای صحیح برایتان اهمیت دارد AVI می تواند انتخاب خوبی باشد. البته توصیه می شود که Codec های مربوطه را به لحاظ اطمینان بیشتر به همراه بسته نرم افزاری خود به کاربران ارائه دهید.

« خروجی MMB چگونه فایلی است ؟ »

قبلاً خواندیم که MMB با استفاده از Upx فایل اجرایی نهایی را فشرده و سپس تولید می‌کند، البته در صورتی که به هنگام تولید نهایی از فشرده سازی استفاده کنیم.

Upx برنامه‌ایست که برای فشرده سازی فایل‌هایی از قبیل exe, dll, ocx و... به کار می‌رود. مزیت این فشرده‌سازی علاوه بر ایجاد فشردگی مطلوب عدم نیاز به استخراج فایلها از بسته فشرده شده به هنگام اجرای فایل فشرده می باشد. به عنوان مثال همان طور که می‌بینید فایل‌های اجرایی که توسط MMB تولید شده‌اند هیچ پیغامی را در هنگام اجرا مبنی بر استخراج فایلها ارائه نمی‌دهند. اکثر توسعه‌دهندگان Plug-In هم از این برنامه جهت فشرده سازی فایل‌های dll تولید شده استفاده می‌کنند. لازم به ذکر است که Upx یک برنامه به صورت Command Line می‌باشد. جهت بررسی این برنامه می‌توانید آن را از روی سی دی ضمیمه این کتاب پیدا کنید.

اما اگر پا را فراتر بگذاریم می‌توانیم از همین Upx برای ایجاد تغییراتی جزئی در فایل اجرایی نهایی ایجاد کنیم. همه‌ی برنامه‌های اجرایی دارای منابعی (Resource) هستند که شمایل برنامه‌ها را تشکیل می‌دهند منظور ما از منابع همان تصاویر، آیکن‌ها، متون، مکان نماها و.. به کار رفته در فایل اجرایی است. در همین زمینه برنامه‌هایی از قبیل Resource Hacker موجودند که کارشان تغییر همین منابع است. بنابراین جهت ایجاد تغییرات در فایل اجرایی نهایی می‌توانیم از Upx برای بازگرداندن فایل اجرایی تولید شد از حالت فشرده به حالت عادی استفاده کنیم و در نهایت با استفاده از برنامه‌های تغییر منابع تغییرات دلخواه را صورت دهیم. البته نکته مهمی را که در اینجا باید متذکر شد اینست که عمل تغییر منابع را تنها برای فایل‌هایی می‌توانیم انجام دهیم که مجاز به تغییر آنها می‌باشیم.

مقدمه‌ای بر فایل AUTORUN. INF

در محیط Windows 32-BIT تکنولوژی بنام AUTOPLAY ظهور کرد، که بوسیله آن سیستم عامل می‌تواند وجود دیسک را در درایو تشخیص دهد. در محیط windows ، هنگامی که دیسک در درایو قرار می‌گیرد، سیستم عامل به دنبال فایل AUTORUN. INF می‌گردد، این فایل که یک فایل متنی است، توانایی‌هایی را به شرح زیر برای سیستم عامل فراهم می‌کند.

بوسیله این فایل میتوان:

- مسیر و نام برنامه اجرایی را برای سیستم عامل مشخص نمود تا هنگام قرار دادن دیسک در درایو، این برنامه اجرایی به صورت خودکار به اجرا در آید.

- آیکون مورد نظر خود را جایگزین آیکون پیش فرض درایو در My Computer نمود.

- متن مورد نظر خود را جایگزین متن پیش فرض (Label) درایو در windows explorer نمود.

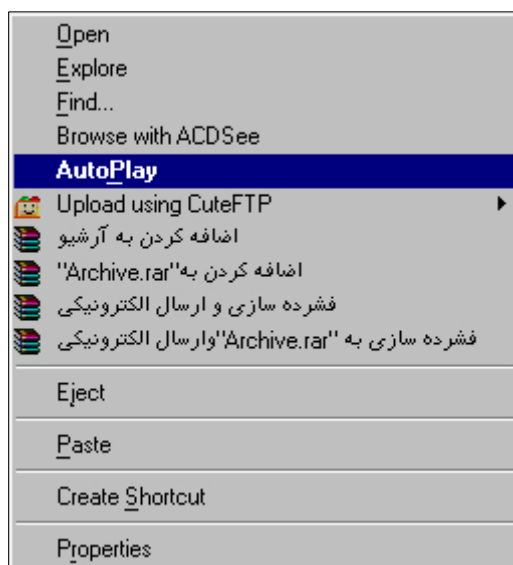
در ابتدا توجه شما را به چند نکته در مورد ایجاد فایل AUTORUN. INF جلب می‌کنیم.

۱- برای ایجاد این فایل می‌توانید از برنامه Notepad کمک بگیرید و سپس پسوند فایل ایجاد شده را از TXT به INF تغییر دهید.

۲- همانطور که در مثال زیر مشاهده می‌کنید، در خط اول از فایل، نوشتن "[Autorun]" الزامی است.

۳- هر دستور را در یک خط مجزا بنویسید.

۴- برای اطلاع از این که دیسکی که در درایو قرار داده‌اید محتوی فایل AUTORUN . INF است یا خیر، می‌توانید بر روی آیکون درایو مذکور راست کلیک کنید سپس منویی که به آن منوی میان‌بر گفته می‌شود مانند شکل زیر باز می‌گردد.



در شکل بالا اگر گزینه Open را انتخاب کنید می‌توانید بدون اجرای AutoRun محتویات درون درایو را مشاهده کنید، اما اگر گزینه Autoplay را انتخاب کنید سیستم عامل با استفاده از برنامه AUTORUN . INF ، فایل AUTORUN . INF را می‌خواند و محتویات آن را همانطور که در آن تعریف شده اجرا می‌کند.

دستورات فایل AUTORUN . INF

این مجموعه، دایره‌المعارفی برای رجوع به دستوراتی است که در فایل AUTORUN . INF بکار می‌رود و شامل shell/ verb open, label, icon می‌باشد.

Icon

دستور Icon آیکون درایوی را مشخص میکند که قابلیت Autoplay را در windows دارد. (منظور درایوی است که Disk در آن قرار می گیرد). و این دستور همانطور که در بالا گفته شد آیکون این درایو را تعیین می کند شکل کلی دستور به صورت زیر است:

icon= iconfilename [, index]

جزئیات دستور

Iconfilename : نام یک فایل با پسوند bmp, exe, dll یا ico می باشد که شامل اطلاعات آیکون مورد نظر باشد. اگر فایل بیش از یک آیکون داشته باشد میتواند با استفاده از پارامتر index (از 0 تا N) آیکون مورد نظر را مشخص کنید.

توضیحات بیشتر

دستور Icon باعث می شود که در Windows Explorer به جای آیکونی که به صورت پیش فرض برای دستگاه دیسک خوان انتخاب شده از آیکون مورد نظر شما استفاده شود. مثال زیر، آیکون دوم را از فایل Myprog. Exe مشخص می کند:

icon= Myprog. Exe 1

Label

دستور Label متن Label دستگاه دیسک خوانی را مشخص می کند که قابلیت Autoplay دارد. شکل کلی دستور به صورت زیر است:

label = Level Text

جزئیات دستور

Label Text: متنی است که برای جایگزینی بجای Label پیش فرض درایو مورد نظر در سیستم عامل انتخاب میشود و میتواند فواصل را نیز شامل شود.

توضیحات بیشتر

Label در زیر؛ آیکون درایوی که قابلیت Autoplay دارد به نمایش در می‌آید و در واقع متن، زیر آیکون می‌باشد. مثال زیر عبارت "MydriveLabel" را بجای Label پیش فرض سیستم عامل قرار می‌دهد.

Label= My Drive Label

Open

دستور Open، مسیر و نام فایل برنامه‌ای را که باید هنگام قراردادن دیسک درون درایو توسط کاربر به صورت خودکار اجرا شود. مشخص می‌کند. شکل کلی دستور به صورت زیر است:

open= [exepath/] exefile [param1 [param2]...]

جزئیات دستور

Exefile: مسیر کامل فایل اجرایی است که قرار است هنگام قرار گرفتن دیسک در درایو، به صورت خودکار اجرا شود. اگر فایل اجرایی در شاخه اصلی درایو باشد می‌توانید فقط نام آن را بنویسید، در غیر این صورت نوشتن مسیر کامل فایل الزامی است.

Shell / Verb

Shell/ Verb: یک دستور به منوی میان‌بر اضافه می‌نماید که فرم کلی آن به صورت زیر است:

Shell/Verb/command= filename. Exe

Shell/Verb= MenuText

جزئیات دستور

Verb: به صورت پیش فرض، متنی است که در منوی میان‌بر نمایش داده می‌شود. دستور Verb, Shell /Verb/Command را با یک فایل اجرایی پیوند می‌زند.

Filename . exe

مسیر و نام فایل برنامه‌ای است که با Verb پیوند می‌خورد.

Menutext

متنی را مشخص می‌کند که در منوی میان‌بر نمایش داده می‌شود که اگر حذف گردد یا نوشته نشود، بجای آن Verb نمایش داده می‌شود. بین کلمه‌ها در Meun Text میتوان فضای خالی داشت (در مورد Verb چنین نیست) همچنین با قراردادن کلمه "&" در ابتدای کلمه مورد نظر در Menu Text میتوانی برای آن کلید shortcut درست نمایید.

توضیحات بیشتر

هنگامی که کاربر بر روی آیکون درایو با قابلیت Autoplay راست کلیک می‌کند منوی میان بر ظاهر می‌گردد. افزودن دستورات Shell/Verb به فایل Autorun.inf به شما اجازه می‌دهد که گزینه‌هایی را به این منو اضافه کنید.

این دستور دو قسمت دارد که باید در دو خط جداگانه باشند. قسمت اول shell/ Verb/ Command می‌باشد که وجود آن الزامی است این دستور یک رشته متنی را که Verb نام دارد به برنامه‌ای که هنگام اجرای دستور باید اجرا شود پیوند می‌زند. قسمت دوم، دستور Shell/Verb می‌باشد که انتخابی است و شامل متنی است که در منوی میان بر نمایش داده میشود.

این قطعه از فایل Readit, AutoRun, inf را (که در واقع همان Verb است) با دستور "Notepad ABC/ Readm. Txt" پیوند می‌زند. متن منوی میانبر (Menu Text) , "Read Me" می‌باشد و "M" به عنوان کلید Shortcut انتخاب شده است. وقتی کاربر گزینه را از منوی میان بر انتخاب می‌کنند فایل ABC/ Readme.Txt بوسیله برنامه Notepad باز میشود.

```
Shell/readit/ command= notpad abc/readme. Txt
```

```
Shell/readit= Read&Me
```

چگونه فایل‌های AutoRun.inf را امتحان کنیم.

همانطور که گفته شد در محیط windows هنگامی که دیسک در درایو قرار می‌گیرد سیستم عامل به دنبال فایل‌های سیستم و فایل AutoRun. Inf میگردد. حال اگر بخواهید این فایل را قبل از نصب بر روی CD امتحان کنید لازم است در Windows Registry کامپیوتر خود تغییرات زیر را اعمال کنید تا قابلیت Autoplay به ابزارهای دیگر نظیر Floppydisk یا Network Drives منتقل شود.

برنامه Regedit را اجرا کرده و تغییرات لازم را در کلید زیر اعمال کنید:

HKEY- CURRENT- USER\

Software\

Microsoft\

Windows\

CurrentVersion\

Policies\

Explorer\

“NoDrive Type AutoRun”

این کلید از نوع Reg- Binary می باشد و از ۴ بایت تشکیل شده است. به بایت اول آن Bitmask گفته می شود Bitmask تعیین می کند که کدام درایو یا ابزار باید به صورت خودکار اجرا شود. ۳ بایت دیگر باید مقدارشان (0) باشد. هر بیت در Bitmask بیانگر توضیحی برای Registry می باشد که شرح آنها در زیر آمده است.

توضیحات	بیت	نوع
-	0	Drive - Unknown
-	1	Drive - No-Root-Dir
Floppy Disk	2	Drive - Removable
-	3	Drive - Fixed
Network Drive	4	Drive - Remote
-	5	Drive - CD ROM
-	6	Drive - Ramdisk

به صورت پیش فرض ارزش این کلید Registry، 0x95 می باشد به عنوان مثال برای این که بتوانید قابلیت Autoplay را برای Flopydisk تنظیم کنید ارزش Bitmask را به 0x91 تغییر (Modify) دهید.

پیغام خطای: Windows cannot find autorun. Exe

هنگامی که به لیست درایوها در Windows explorer نگاه می‌کنید ممکن است یک یا چند درایو، آیکون اشتباه و نامربوط به خود را داشته باشند، هنگامی که برای مشاهده محتویات آنها بر روی آیکون درایو دابل کلیک می‌کنید، ممکن است پیغام خطای زیر نشان داده شود.

Program Not Found Windows Cannot Find Autorun.exe

This Program is needed for opening files of type "file"

سپس از شما آدرس و مسیر فایل Autorun.exe خواسته می‌شود.

توضیحات بیشتر

فایل Autorun. inf همراه با درایوی که قابلیت Autoplay را دارد (مانند CD- Drive) برای اجرای خودکار یک برنامه مشخص به کار گرفته می‌شود. هنگامی که یک درایو که قابلیت Autoplay ندارد محتوی فایل Autorun. inf باشد بعد از دابل کلیک کردن کاربر بر روی آیکون آن درایو، برنامه Autoplay سعی میکند فایل Autorun.exe را (که در آن درایو وجود ندارد) برای خواندن فایل Autorun. inf اجرا کند، که باعث روی دادن خطا در سیستم عامل می‌شود.

راه حل

برای رفع این مشکل بر روی درایو فوق‌الذکر راست کلیک کنید، منوی میانبر باز میشود (مشاهده میکنید که گزینه Autoplay. پیش فرض می‌باشد). گزینه open را انتخاب نمایید و پس از باز شدن درایو، فایل Autorun. inf را که در شاخه اصلی درایو قرار دارد پاک نمایید و سپس کامپیوتر را Restart کنید.

چگونه میتوان به CD- ROM اجازه اجرای Auto Run داد یا از آن سلب اجازه نمود.

اطلاعات این قسمت فقط در سیستم عامل های زیر صدق می کند:

Microsoft windows NT server*

Microsoft windows NT workstation*

اگر یک تعویض کننده CD- ROM (CD- ROM Changer) به کامپیوتر شما متصل باشد، ممکن است بخواهید امکان AutoRun را از آن سلب کنید. اگر تعویض کننده CD- ROM ، امکان AutoRun داشته باشد، هر زمان که شما CD را در یکی از طبقات آن قرار می دهید، سیستم عامل تمام طبقات تعویض کننده CD را مرور میکند.

توضیحات بیشتر

گزینه ای در Control panel برای سلب اختیار نمودن سیستم عامل از اجرای AutoRun وجود ندارد، برای اینکار شما باید Registry را ویرایش کنید. برای انجام این کار مراحل زیر را دنبال کنید.

اخطار: اگر از برنامه Registry Editor درست استفاده نکنید، میتواند سبب بوجود آمدن مشکلات جدی برای سیستم عامل کامپیوتر شما شود، که شما را مجبور به دوباره نصب کردن windows میکند. Registry را با مسئولیت خود تغییر دهید.

۱- مقدار AutoRun را در کلید Registry زیر به شرحی که آمده است تغییر دهید.

HKEY- LOCAL-

Machine\ System\ Current Control Set\ Services \ CDRom

اگر مقدار AutoRun را به 0 تغییر دهید از تعویض کننده CD- ROM برای اجرای AutoRun سلب اختیار میشود و اگر ارزش آن را به 1 تغییر دهید، دوباره اختیار AutoRun به تعویض کننده CD- ROM داده میشود.

۲- کامپیوتر را Restart کنید.

نکته: روش بالا به طور کامل از انجام پروسه AutoRun سلب اختیار میکند. اگر میخواهید بر روی هر دیسک به طور مجزا برای اجرای AutoRun نظارت کنید بعد از قراردادن دیسک در درایو، کلید shift را پایین نگه دارید.

میانبرهای پیش فرض MMB

New	Ctrl+N
Open	Ctrl+O
Print	Ctrl+P
Save	Ctrl+S
Copy	Ctrl+C
Delete	Del
Hide/Show	Ctrl+T
Paste	Ctrl+V
Paste Bitmap	Ctrl+B
Redo	Ctrl+Y
Select All	Ctrl+A
Undo	Ctrl+Z
Test CurrentPage	F5
10 Pixel Down	Ctrl+Down
10 Pixel Left	Ctrl+Left
10 Pixel Right	Ctrl+Right
10 Pixel Up	Ctrl+Up
Bring Forward	Ctrl+PageUp
Bring to Front	Shift+PageUp
Group	Ctrl+G
Pixel Down	Down
Pixel Left	Left
Pixel Right	Right
Pixel Up	Up
Send Backward	Ctrl+Page down
Send to Back	Shift+Page down
Ungroup	Ctrl+U

توجه: با راست کلیک بر روی رابط کاربر MMB منویی ظاهر می شود که می توانید از آن جهت سفارشی سازی رابط کاربر MMB استفاده کنید.

منابع :

- راهنمای برنامه MMB
- سایت www.mmbforums.com
- ماهنامه کامپیوتر جوان

واژه نامه

((A))

Alpha Button = دکمه آلفا

Animated GIF = GIF متحرک

Audio visualization = بصری ساز صوتی، رقص نور

Align = تراز کردن

Arrange = مرتب کردن

Antialias = هموار سازی

Apply = فعال کردن

Aspect = وجه

Actual = حقیقی

Adjust = تنظیم کردن

Animate = متحرک سازی

Advanced = پیشرفته

Additional = اضافی

Array = آرایه

Accurate = دقیق، صحیح

Artist = هنرمند

Absolute = مطلق

((B))

Bitmap Button = دکمه نقشه بیتی

Bitmap = نقشه بیتی

Binder = موظف کننده، در اینجا برقرار کننده اتصال

Background = پس زمینه

Backward = به عقب

Blur = بلور

Bevel = برجسته سازی، اریب کردن، تراشیدن

Button = دکمه

Border = حاشیه، لبه

Bumpy = دارای برآمدگی، نا هموار

Bubble = حباب

Buffer = حافظه میانجی

Blend=مخلوطی، آمیخته

Boost=بالا بردن، ترقی دادن

Balance=تعادل

((C))

Circle=دایره

Customize=سفارشی سازی

Comment=توضیح

Convert=تبدیل کردن

Clone=زاییدن

Combine=ترکیب کردن

Crop=چیدن

Crayon=مداد رنگی مومی

Cube=مکعب

Cascade=آبشاری

Calculation=محاسبه

Circularity=مدور بودن

Chunks=تکه بزرگ و کلفت

Column=ستون

Caption=عنوان

Capture=ربودن، گرفتن

Constant=ثابت، دائم

Clipboard=تخته برش

Curl=پیچاندن، حلقه کردن

Compression=فشرده‌گی

Configuration=پیکربندی

Current=جاری، کنونی

Content=مندرجات، محتوی

Counter=شمارشگر

((D))

Dynamic fx=پویا fx

Dimension=بعد
Debug=اشکال زدایی
Distribute=نشر دادن
Designer=طراح
Default=پیش فرض
Disposal=در معرض گذاری
Define=معین کردن، تعریف کردن
Drag=کشیدن
Drop=انداختن
Disable=غیر فعال کردن
Display=نمایش دادن، نمایان ساختن
Decimal=دهدهی، اعشاری
Delay=به تاخیر انداختن
Destination=مسیر، مقصد
Description=توصیف، توضیح

((E))

Embedded=جاسازی شده
Effect=جلوه
Emboss=برجسته کردن، اندودن
Expand=گسترش دادن
Export=صادر کردن
Enable=فعال سازی
External=خارجی
Edge=لبه، یال
Extension=بسط، پسوند نام فایل
Engine=موتور
Else=دیگر، جز این
Editor=ویرایشگر

((F))

Flash=روشنایی آنی و مختصر

Forward= به جلو
Flip= چرخاندن، تلنگرزدن
Fixed= ثابت و ماندنی
Fade out= محو شدن
Float= معلق
Flat= تخت
Fill= پرکننده
Fit= مناسب، درخور
Frame= فریم، قاب
Factor= فاکتور، عامل
Flare= روشنایی خیره کننده و نامنظم
Function= تابع

((G))

Grid= شبکه، چهارخانه بندی
Glow= درخشش
Glass= شیشه
Grip= چنگ زدن، گرفتن
Gradient= شیب
Generate= تولید کردن

((H))

HTML= زبان نشانه گذاری ابر متن
Horizontally= افقی
Height= ارتفاع
Highlight= منور کردن، پر رنگ کردن
HotSpot= منطقه داغ
Handle= دسته

((I))

Input Text= متن ورودی
Image Matrix= ماتریس تصویر

Interaction= تعامل

Impressionist=

Import= وارد کردن

Invert= معکوس کردن

Integer= عدد صحیح

Internal= درونی

ID= عبارت شناسایی

Interference= تداخل

Interval= فاصله، خلال

Idle= بیکار

If= اگر

Initialize= آغاز کردن

((J))

((K))

Keep= نگهداری کردن

((L))

ListBox= کادر لیست

Line= خط

Label= برچسب

Library= کتابخانه

Loop= حلقه تکرار

Lumi= روشن

Lens= عدسی، به شکل عدسی

Layer= لایه

Legal= قانونی

((M))

Mosaic= موزاییک، تکه تکه بهم پیوستن

Metal= آهن

Menu= منو

Modal=مقید

Mask=ماسک

Media=رسانه

Movement=جنبش، تکان

Master=قسمت اصلی، صاحب، مدیر

Mixed=آمیخته شده

Message=پیغام

Milliseconds=میلی ثانیه

Mute=صامت، بی صدا

Magnify=بزرگ کردن

MMB=مولتی میدیا بیلدر

Macromedia Flash=برنامه فلش شرکت ماکرو مدیا

Method=روش

Macro=ماکرو

((N))

Nudge=اشاره کردن، در اینجا تکان دادن مختصر

Numeric=عددی

Null=پوچ

((O))

Opacity=غلظت

Origin=مبدأ، منشأ

((P))

Paragraph Text=پاراگراف متنی

Panorama=چشم انداز

Polygon=چند ضلعی

Polygonal Hotspot=منطقه داغ چند ضلعی

Plug-In=در اینجا یعنی قسمتهایی که سبب افزایش قابلیت های برنامه می شوند

Property=خصوصیت، مشخصه

Previous=قبلی

Preview= بازبینی
Parameter= پارامتر
Pixel= پیکسل
Primitive= اصلی، بدوی
Plasma= پلاسما
Preset= از پیش نشاندهنده
Progress= پیشرفت
Preserve= حفظ کردن
Priority= حق تقدم
Period= دوره
Picker= چینه‌ده، برداشت کننده
Pitch= زیر و بمب صدا
Predefined= از پیش تعریف شده
Particle= ذره
Poor= نا مرغوب، ناچیز
Path= مسیر
Primary= نخستین
Page= صفحه

((Q))

Quality= کیفیت
Quantize= کمیت گذاری

((R))

Rectangle= مستطیل
Replace= جایگزین کردن
Ripple= موج دار شدن
Restore= بازسازی
Ratio= نسبت
Refresh= تازه کردن
Row= طبقه، ردیف
Raised= برافراشته شده، برجسته شده

Return= بازگشت
Resume= از سر گیری
Rotate= چرخاندن
Relative= وابسته، نسبی
Randomize= تصادفی کردن
Random= تصادفی، رندم
Reverse= معکوس
Redo= دوباره انجام دادن
Reflectivity= بازتاب پذیری
Real-DRAW pro= برنامه ریل درا برای طراحی ۲ بعدی
(S)
Script= اسکریپت
Snap= پرش
Sound= صدا، صوت
Shadow= سایه
Sketch= طرح
Setting= تنظیمات
Skin= پوسته
String= رشته
Solid= یکدست، جامد
Style= سبک
Safe= امن
Stage= مرحله
Scroll= پیمایش
Storage= انبارش، ذخیره سازی
Stretch= کشیدن، امتداد دادن
Static= ایستا، ساکن
Sunken= فرورفته
Still= هنوز، خاموش، بی حرکت
Spectrum= طیف
Saturation= اشباع
Smoke= دود

Status= وضعیت
Shape= شکل، قیافه
Sparkle= تاللوداشتن
Separator= جدا کننده
Speech= سخنرانی
Syntax= نحو
Scale= مقیاس گذاری
Shortcut= میانبر
Smart= باهوش
Suppress= مانع شدن
Solarize= در معرض آفتاب قرار دادن
Sample= نمونه، مثال
Source= منبع، منشأ
Statement= اظهار

((T))

Text= متن
Text Button= دکمه متنی
Template= قالب
Tile= کاشی
Twirl= چرخش، گردش
Transparent= پشت نما، شفاف
Type= سنخ، نوع
Terminate= پایان دادن
Transition= واسط، مرحله تغییر
Title= عنوان
Tolerance= تحمل
Then= سپس
Total= مجموع
Trademark= علامت تجاری
Tweak= نیشگون گرفتن، پیچاندن

((U))

URL= آدرس اینترنتی

Undo= خنثی کردن

UI= رابط کاربر

Urban= شهری

((V))

Video= ویدئو

Vertically= عمودی

Variable= متغیر

Vibration= ارتعاش

Visualization= تجسم

Volume= حجم

Visible= قابل دید

((W))

Wrap= پیچیدن

Width= عرض

Wizard= جادوگر

((Z))

Zero= صفر

منتظر ارائه راهنمای بعدی در این زمینه باشید...