الجلسة 23

لمسات برمجية :

يمكن أن نعتبر برنامج الهاتف برنامج جاهز للتوزيع والنشر ولكن بإضافة بعض اللمسات إلى البرنـامج يمكـن أن نحول هذا البرنامج إلى برنامج سهل الاستخدام ومرغوب بشكل أكبر من قبـل المـستخدمين، وأقـصد باللمـسات البرمجية بمجموعة الأعمال التي يقوم بها المبرمج والتي يمكن أن تأخذ وقت طويلاً لتسهيل عمل المستخدم .

1 ـ تمكين المستخدم من التعامل مع البرنامج بدون استخدام الفأرة وبأسرع الطرق الممكنة :

مثلاً عندما يكتب المستخدم اسم للبحث عنه ضمن Edit1 فلا حاجة لأن يعود المستخدم إلى الفأرة ليـضغط علـى الزر لكي يتم تنفيذ البحث بل يكفي أن يضغط المستخدم على الزر إدخال (Enter) من لوحة المفاتيح ليتم البحـث ولتحقيق ذلك برمجياً : اكتب التعليمات التالية في حدث OnKeyPress للعنصر Edit1 :

```
if key = char( VK_Return ) then
btnsearch.Click ;
```

حيث VK_Return ثابت معرف في دلفي ويحوي شيفرة الأسكي للزر Enter، في الحقيقة يجود الكثير من هـذه الثوابت والتي تحوي شيفرة الأزرار الافتراضية (Virtual Key) وهذه الأزرار هي المفاتيح F1..F12 و مفاتيح الأسهم و Tab و Caps lock و Delete و End ... (أي مجموعة أزرار لوحة المفاتيح عدا الأحرف والأرقام) وللتعرف على هذه الثوابت راجع ملفات المـساعدة فـي دلفـي أو اكتـب فـي مكـان فـي محـرر الـشيفرة Windows.VK_ وستكمل دلفي بقية الثوابت.

ملاحظات :

أ ـ يمكن الاستعاضة عن التعليمة السابقة بالتعليمة :

```
If key = #13 then btnsearch.Click ;
```

حيث 13 هي القيمة التي يحويها الثابت VK_Return وهي القيمة الاسكي للزر Enter و # تستخدم قبل شـيفرة الأسكي لتحويل الرقم إلى حرف ، مثلاً إذا كتبنا التعليمة التالية : S:=#89#111#117 فهذا يكافئ التعليمة S := 'You'

ب ـ يمكن الاستعاضة عن التعليمة السابقة بالتعليمة :

```
If key = #13 then  BtnSearchClick (Sender);
```

بدلاً من استخدام المنهج Click للزر BtnSearch نستخدم الإجراء الذي يقوم بهذه العملية مباشـرة (لا حـظ أن الإجراء BtnSearchClick قد تم تعريفة من قبل الدلفي في القسم Type والقـسم Implementation بمجـرد ضغطك على الزر ....

2 ـ من الملاحظ أننا نستخدم العنصر Edit2 لإدخال الأرقام فقط أو على الأقل رمـوز لوحـة مفـاتيح الهـاتف وبالتالي يجب منع المستخدم من إدخال الحروف والرموز غير المرغوبة ولتحقيق ذلك اكتب التعليمة التالية :

```
const s = ['0','1','2','3','4','5','6','7','8','9','p',char(VK_back)];
begin
if not ( key in s )
then key:= char(vk_capital)
```

نعرف مجموعة من الرموز التي يستخدمها الهاتف بالإضافة إلى الزر VK_back و هو زر الحذف ← ، فإذا لم يكن الزر المدخل من هذه المجموعة فحوله إلى الزر Vk_Capital وهو زر Caps Lock والذي لن يؤثر على وضع الزر Caps Lock ولكنه سيصدر صوت ما .

3 ـ في النافذة إدخال على المستخدم أن ينتقل من عنصر تحرير DBEdit إلى آخر بواسطة الضغط على الزر Tab من لوحة المفاتيح على الرغم من أن معظم البرامج الجيدة تستخدم الزر Enter للتنقل بين عناصر الإدخال لذلك اكتب التعليمة التالية في الحدث OnKeyPress للعنصر DBEdit1 :

```
If key = #13 then DBEdit2.SetFocus ;
```

واكتب في حدث OnKeyPress التعليمة التالية :

```
If key =#13 then dbedit3.SetFocus ;
```

وهكذا في بقية العناصر .

المنهج SetFocus يستخدم لتحفيز (تفعيل العنصر) وهو يستخدم مع معظم العناصر في دلفي فمثلاً التعليمة Button1.SetFocus ستظهر إطار شفاف داخل الزر والذي يعني أن هذا هو الزر الفعال والذي يمكن الضغط عليه بواسطة الزر Enter أو الزر Space من لوحة المفاتيح .

4 ـ التحكم بتحميل النوافذ في الذاكرة :

افتح القائمة Project → View Source سيظهر لك ملف المشروع MyPhone.Dpr وستجد فيه تعريف الوحدات في القسم Uses و ستجد التعليمات التالية في القسم Begin

```
Application.Initialize;
Application.CreateForm(TForm1, Form1);
Application.CreateForm(TDataEnter, DataEnter);
Application.CreateForm(TForm2, Form2);
Application.Run;
```

التعليمة Application.CreateForm (TFom1,Form1) تقوم بإنشاء النافذة Form1 والتي من النوع TForm1 والمعرف في القسم Interface للوحدة Main.pas التي يتبع لها النموذج Form1 وكذلك الأمر بالنسبة لتعليمات CreateForm التالية ، ولكن ماذا يعني إنشاء نافذة.... عند استخدام التعليمة Application.CreateForm (TFom1,Form1)

1 ـ سيتم إنشاء هذه النافذة في الذاكرة (أي حجز حيز في الذاكرة يحوي النافذة التي ستظهر) .

2 ـ سيتم إنشاء جميع العناصر الموجودة على هذه النافذة من أزرار و عناصر قواعد البيانات و....

3 ـ سيتم حجز المتحولات العامة وإعطاءها حيز من الذاكرة أيضاً .

4 ـ سيتم تنفيذ التعليمات التي تم وضعها في الحدث OnCreate التابع للنموذج Form1 .

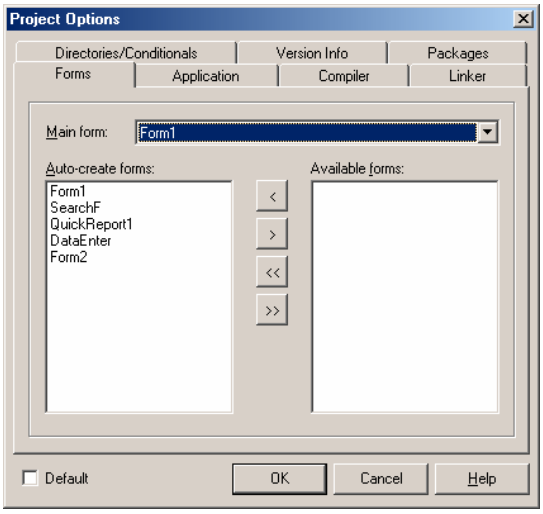
سيتم الآن تنفيذ هذه الخطوات بالنسبة للنوافذ الثلاثة المستخدمة في البرنامج ، لنقم الآن بدراسة بسيطة لحجم استخدام كل نافذة :

1 ـ النافذة الأولى Form1 : سيستخدم أي مستخدم هذه النافذة عند كل دخول إلى البرنامج وذلك لأنها تحوي عناصر البحث والفهرسة بالإضافة إلى عرض البيانات.

2 ـ النافذة DataEnter : تستخدم هذه النافذة عدة مرات في الأسبوع الأول لاستخدام البرنـامج لأن المـستخدم سيقوم بإدخال أرقام هواتفه إلى البرنامج وبعد ذلك لن يدخل المستخدم إلى هذه النافذة إلا مرة في الأسبوع أو الشهر أو حتى بالسنة .

3 ـ النافذة Form2 : هذه النافذة تحوي عناصر الطابعة وإذا فرضنا أن المستخدم يملك طابعة فإنه لن يطبع دفتر هواتفه كل يوم .

هذا يعني أننا نحجز حيز من الذاكرة لنافذتين لن يستخدمها المستخدم إلا مرات قليلة، لهذا زودتنـا دلفـي بطريقـة لإنشاء النموذج عن الحاجة إليه وذلك عن طريق :



أ ـ إزالة النموذج من قائمة التحميل التلقائي :

افتح القائمة Project → Options لتظهر النافذة التالية : حدد منهـا الصفحة Forms والتي فيها القائمة المنسدلة Main Form وهي تحدد النموذج الذي سيظهر عند تشغيل البرنامج.

اللائحة Auto-Create Forms وتظهر فيها أسماء النماذج التي سيتم إنشاءها تلقائياً عند تشغيل البرنامج ،

اللائحة Available Form وتحوي جميع بقية النوافذ فـي البرنـامج والتي لا توجد في لائحة التشغيل التلقائي.

حدد النوافذ التي لا تريد أن تعمل تلقائياً واضغط على السهم < لتنتقل هذه الاسماء إلى اللائحة Avialable Form .

ب ـ إنشاء النموذج عند الطلب :

بما أنه النافذة لن تنشأ تلقائياً فعلينا أن نكتـب بعـض التعليمـات لإنـشائها قبـل اسـتخدام المـنهج Show أو ShowModal أو حتى قبل طلب أي متحول ما منها لذلك لنكتب التعليمات التالية في حدث الضغط علـى الـزر إدخال :

```
Dataenter := TDataEnter.Create(application);
Try
DataEnter.Showmodal;
Finally
DataEnter.Release  ;
end;
```

التعليمة الأولى تقوم بإنشاء المتحول DataEnter من النوع TDataEnter (المعرف في القسم Var التابع للقـسم Interface في الوحدة DataEnterU.pas) ويأخذ المنهج Create لعناصر Delphi وسيطاً وحيداً يـدل علـى العنصر المالك للعنصر المراد إنشائه وتمرر هنا القيمة Application عادة لأن النموذج المنشأ سـيعود لتطبيـق مباشرة .

التعليمة DataEnter.Release هي عملية لإزالة النموذج من الذاكرة بعد انتهاء عملـه والتـي يمكـن اسـتبدالها بالتعليمة DataEnter.Free ولكن المنهج Release يعتبر أفضل لأنه لا يعيد رسالة خطأ إذا لم يكن النموذج منشأً بالفعل .

إذا كنا نستخدم المنهج Show بدلاً من المنهج ShowModal فيجب إضافة التعليمات التالية للتأكـد مـن عمـل البرنامج بشكل صحيح :

في الحدث OnClose للنافذة DataEnter أضف التعليمة التالية :

```
action := Cafree;
DataEnter := nil;
```

واكتب الآن التعليمات التالية في حدث الضغط على الزر إدخال (في النموذج الأول) :

```
If DataEnter = nil then
DataEnter := TDataEnter.Create(application);
DataEnter.Show;
```

الحدث OnClose لنافذة يتم تنفيذه عندما يتم إغلاق النافذة (ضغط المستخدم على زر إغلاق) ويأخذ هذا الحـدث الوسيط Action والذي يمكن أن يأخذ القيم التالية :

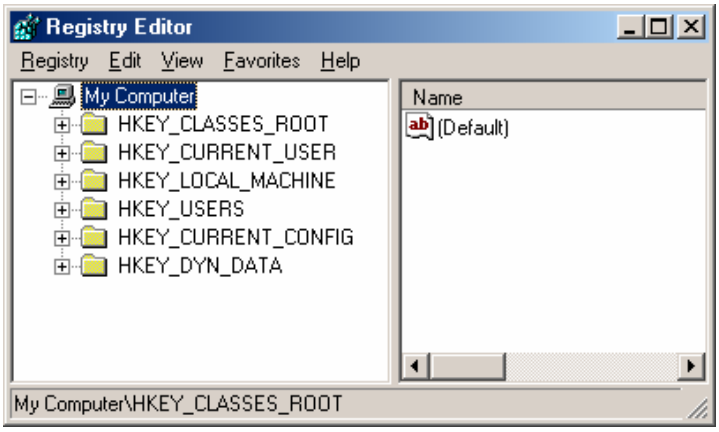
القيمة	المعنى
CaNone	لا يمكن إغلاق النافذة ولن يحدث شيء إذا ضغط المستخدم على الزر إغلاق
CaFree	سيتم إغلاق النافذة وتحرير كامل الذاكرة المستخدمة من قبلها
CaHide	سيتم إخفاء النافذة بدلاً من إغلاقها ويمكن إعادة إظهارها عن طريق المنهج Show وهو الخيـار الافتراضي للنوافذ من النوع FormStyle := fsNormal
CaMinimize	سيتم تصغير النافذة وهو الخيار الافتراضي مع النوافذ من النوع fsMDIChild

التعليمة DataEnter = Nil تقوم بتحميل قيمة اللاشيء (Null) في المتحول DataEnter الأمر الـذي يجعلنـا فيما بعد أن الحدث OnClose قد تم تنفيذه ومن ثم تنفيذ عملية إنشاء النموذج .

ملاحظة : هذه إحدى الطرق المستخدمة لإنشاء النافذة بشكل ديناميكي وهناك طرق أخرى متعددة .

مسجل النظام Windows Registry :

مسجل النظام عبارة عن مجموعة من الملفات يتم تخزين فيه جميع معلومات نظام التشغيل Windows بمـا فيهـا خصائص النظام واسم المستخدم واسم الكمبيوتر بالإضافة إلى أن معظم البرامج تضيف بعض المفاتيح إلى مـسجل النظام للتحكم بطريقة إظهار البرنامج حسب خيارات يحددها المستخدم ، يدعى مسجل النظام بقاعدة بيانات النظـام ويكون مخزن في الملفين System.dat و User.dat الموجودين داخل فهرس الويندوز ،لمشاهدة مـسجل النظـام وتحريره افتح القائمة أبدأ و اضغط على Run واكتب Regedit في النافذة تشغيل ثم اضغط على Ok .



ستظهر عندها النافذة التالية والتني تحـوي فـي جهـة اليمين ستة من المفاتيح الرئيسية تدعى عـادة مفـاتيح الجذر Root key ، بالضغط على إشارة + بجانب هذه المفاتيح تظهر لها مفاتيح فرعية و التي يمكن الوصـول من خلالها إلى مفاتيح فرعية أخرى، تدعى الرموز التي تظهر على القسم اليميني من النافذة بالقيم وهـي علـى ثلاثة أنواع ،

قيم عددية و قيم نصية وقيم منطقية ،

قبل أن تعدل أي قيمة ما في مسجل النظام عليك أن تعرف ما تفعل لأن بعض التعديلات قد تؤدي إلى توقف النظام عن العمل أو تعطيل البرنامج الذي عدلته على الأقل ، يمكنك أن تأخذ نسخة عن مسجل النظام أو عن جـزء منـه ضع حدد بمؤشر الفأرة المفتاح الذي تريد أخذ نسخه منه واختر الأمر Regestry → Export سـتطالب بإدخـال اسم ملف ما وسيتم بعدها حفظ نسخة عن المفتاح ، لاستعادة مسجل النظام اضغط مرتين على ملف الملـف الـذي كنت قد حفظته سابقاً .

سأشرح بشكل موجز ماهي المعلومات المخزنة في كل مفتاح رئيسي :

HKEY_ClASSES_ROOT : يحوي هذا المفتاح معلومات عن كل أنواع الملفات المعرفة في النظام (لا حظ أنه يحوي قائمة بجميع امتدادات الملفات التي يتعامل معها ويندوز) بالإضافة إلى معلومات عن البـرامج القـادرة على التعامل مع هذه الملفات مثلاً jpg. أو Doc. ...

HKEY_CURRENT_USER : يحوي معلومات عن مستخدم النظام والقيم الافتراضية التي أعطاهـا لـبعض برامج الويندوز بالإضافة إلى الخصائص لوحة التحكم والبرامج التي تنصبها المستخدم على الجهاز .....

HKEY_LOCAL_MACHINE : يحوي معلومات عن جهاز الحاسوب بالكامل بما فيهـا أنـواع الأجهـزة المستخدمة في الجهاز بالإضافة إلى مشغلات هذه الأجهزة (Driver) مثلاً تستطيع معرفة نوعية كرت الـشاشة المستخدم ونوعية الشاشة المعرفة على الجهاز ....

HKEY_USERS : يحوي معلومات عن التشكيلات الجانبية للمستخدم (الخصائص التـي يحـددها المـستخدم لألوان سطح المكتب مثلاً ..) ، بالإضافة إلى التشكيلات الجانبية الافتراضية التي ستعطى عندما يـدخل مـستخدم جديد إلى الويندوز .

HKEY_CURRENT_CONFIG : يعتبر جزء مـن المفتـاح HKEY_LOCAL_MACHINE ولكـن بطريقة عرض أخرى .

HKEY_DYN_Data : يحوي معلومات عن أجهزة النظام وطرق إعدادها (غير موجود في Windows NT) .

التعامل مع مسجل النظام يسمح لك بالتحكم بالنظام بشكل كامل ولا بد أنك سمعت عن برامج تتحكم بسرعة قائمـة ابدأ و تغير أسم سلة المحذوفات وتغير الصورة التي تظهـر بجانـب قائمـة ابـدأ (الـصورة المكتـوب فيهـا WINDOWS Me مثلاً) و بالإضافة إلى تغير صورة جهاز كمبيوتر و بالإضافة إلى البرامج التي تضيف نفسها

إلى القائمة التي تظهر عند الضغط بزر اليمين على سطح المكتب أو مستكشف ويندوز ، حتى أن البـرامج التـي تدعي أنها تتحكم بسرعة الاتصال انترنت تتعامل مع مسجل النظام .....)
هذه موجز صغير جداً عن Windows Registry وإذا أردت أن تصبح مبرمجاً محترفاً عليك قراءة أحد الكتـب المختصة بهذا الموضوع فكتاب Windows 2000 Registry بحجم 800 صفحة تقريباً .
التعامل مع Registry في دلفي :
إذا لم تكن تريد أن تتحكم بنظام ويندوز من داخل برنامجك فأنت بحاجة إلى تخزين بعض المعلومات عن برنامجك داخل مسجل النظام الأمر الذي يطرك إلى التعامل مع النظام .
مثال : سنقوم بإنشاء برنامج يقوم بمعرفة اسم المستخدم الذي سجل النظام باسمه ومن ثم تغيره (نـستطيع معرفـة اسم المستخدم عن طريق الضغط بزر الفأرة اليميني على أيقونة جهاز الكمبيوتر واختيار خصائص من القائمة) .
اسم المستخدم يوجد داخل مسجل النظام تحت المفتاح
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion]
ومخزن في القيمة النصية RegisteredOwner (افتح البرنامج Regedit للتأكد من ذلك) .
ابدأ مشروعاً جديداً وأضف زرين إلى النموذج وعنصر Edit1 ،
للتعامل مع مسجل النظام عليك إضافة الوحدة Registry إلى القسم USES
للقراءة من مسجل النظام اكتب التعليمات التالية في حدث الضغط على الزر الأول

```
Var reg:tregistry ;
Begin
reg:= tregistry.Create  ; //إنشاء المتحول وحجز مكان له في الذاكرة
try
reg.RootKey := HKEY_LOCAL_MACHINE ;// تحديد المفتاح الرئيسي
reg.OpenKey('SOFTWARE\Microsoft\Windows\CurrentVersion',false ) ;
edit1.text := reg.ReadString('RegisteredOwner');//قراءة القيمة النصية المخزنة في الوسيط المذكور
reg.CloseKey ;  //إغلاق المفتاح
Finally
reg.Free ; //تحرير الذاكرة بعد استخدام المتحول
end;
```

التعليمة OpenKey تقوم بفتح المفتاح المرر لها كوسيط أول وإذا لم يكن موجوداً فتقوم بإنشاء هذا المفتاح إذا كان وسيطها الثاني True.
للكتابة إلى مسجل النظام اكتب التعليمات التالية في حدث الضغط على الزر الثاني :

```
Var reg:tregistry ;
Begin
reg:= tregistry.Create  ;// إنشاء المتحول وحجز مكان له في الذاكرة
try
reg.RootKey := HKEY_LOCAL_MACHINE ;// تحديد المفتاح الرئيسي
reg.OpenKey('SOFTWARE\Microsoft\Windows\CurrentVersion',false ) ;
```

reg.WriteString('RegisteredOwner',edit1.Text); // كتابة القيمة المدخلة في عنصر التحرير إلى اسـم القيمة المحددة
reg.CloseKey ; //إغلاق المفتاح
Finally
reg.Free ; //تحرير الذاكرة بعد استخدام المتحول
end;

مثال 2 : سنعود إلى برنامج الهاتف :

يقوم برنامجنا بالترتيب حسب الاسم دائماً وعلى فرض أن المستخدم يفضل الترتيب حسب الكنية فإن عليـه عـن يضغط على زر الراديو الكنية كلما دخل إلى البرنامج ، سنضيف الآن مفتاح إلى مسجل النظام يقوم بتخزين الخيار الافتراضي للمستخدم ، وسنقوم بفحص قيمة هذا المفتاح كلما دخلنا إلى البرنامج :

سنخزن قيمة عددية تحت أسم IndexBy في المفتاح :

HKEY_CURRENT_USER\Software\MyPhone

وعند تشغيل البرنامج سنفحص قيمة هذا المتحول فإذا كان هذا المتحول يحوي قيمة 1 فالفهرسة ستتم حسب الكنيـة وإلا فالفهرسة ستتم حسب الاسم .

أضف زر إلى النموذج الأول Form1 في برنامج الهاتف وأعطه العنوان " فهرس دائماً حسب الكنية " .

وفي حدث الضغط على هذا الزر اكتب التعليمات التالية :

```
Var reg:tregistry ;
Begin
reg:= tregistry.Create  ;
try
reg.RootKey := HKEY_CURRENT_USER ;
reg.OpenKey('SOFTWARE\MyPhone',True ) ;
if button4.Caption = 'فهرس دائماً حسب الكنية ' Then
reg.WriteInteger ('IndexBy',1 )
else reg.WriteInteger ('IndexBy',0 )
reg.CloseKey ;
Finally
reg.Free ;
end;
```

تقوم التعليمة OpenKey بإنشاء المفتاح إذا لم يكن موجـوداً وتقـوم التعليمـة WriteInteger بإنـشاء القيمـة IndexBy وتخزن فيها القيمة 1 أو صفر حسب الشرط .

الآن سنفحص هذا المفتاح عند تشغيل البرنامج :

```
RBTLname
reg:= tregistry.Create  ;
try
reg.RootKey := HKEY_CURRENT_USER ;
reg.OpenKey('SOFTWARE\MyPhone',True) ;
if reg.ReadInteger ('IndexBy') = 1 then begin
```

```
 RBTLname.Checked := true;
 RBTLnameClick(Sender);
 button5.Caption  := 'رتب دائماً حسب الاسم' ;
end;
 reg.CloseKey ;
Finally
reg.Free ;
end;
```

تفحص هذه التعليمات القيمة المخزنة IndexBy فإذا كانت تساوي 1 فإنها تقوم بتحفيز زر الراديو الكنيـة وتنفيـذ الحدث الخاص بالضغط على هذا الزر والذي فيه تعليمات تغيير الفهرس بالإضافة إلى تغير عنوان الزر إلى رتب دائماً حسب الاسم .

بهذا نكون قد تعلمنا التعامل مع مسجل النظام .

وظيفة : فكر بطريقة تقوم بها بإضافة كلمة سر إلى برنامج الهاتف ، حيث ستظهر نافذة عنـد تـشغيل البرنـامج تطالب المستخدم بإدخال كلمة السر فإذا أدخل كلمة السر الصحيحة فستظهر النافذة الرئيسية وإلا فسيطالب بإدخـال كلمة السر مرة أخرى وإذا فشل في الدخول ثلاث مرات متتالية فيتم إغلاق البرنامج (خزن كلمة السر في مـسجل النظام) .

توابع وثوابت الويندوز Windows API :

توابع وثوابت الويندوز هي مجموعة هائلة من التوابع والثوابت المعرفة في ويندوز والموجـودة ضـمن مكتبـات ويندوز في ملفات ذات الامتداد DLL. وتستخدم هذه النوافذ للتحكم في ويندوز وفي كل شيء في وينـدوز ابتـداء بلوحة المفاتيح وانتهاء مروراً بالذاكرة والمعالج و الشاشة بما في ذلك التحكم بالبرامج والنوافذ وطرق إظهارهـا (يمكنك على سبيل المثال التحكم بارتفاع الصوت باستخدام توابع Api و معرفة معلومات معينة عن الأجهزة مثـل معرفة حجم السواقة الصلبة Hard Disk يمكنك الحصول على معلومات في برنامج آخر من برنامجك كأن تحصل على معلومات من ماف ورد والمزيد الذي لا يمكن حصره هنا .

سنشرح أمثلة عن التوابع وطرق التعامل معها :

1 ـ إغلاق Windows :

يستخدم التابع ExitWindowsEx( ) لإطفاء الجهاز أو إعادة التشغيل :

ابدأ برنامجاً جديداً واضف زر إلى النموذج واكتب التعليمة التالية :

```
ExitWindowsEx(ewx_ShutDown,0) ;
```

احفظ البرنامج وقم بتنفيذه ولا تضغط على الزر ،أغلق البرنامج وشغله من خارج الدلفي (من المكان الذي حفظته فيه) ، اضغط على الزر الآن وسيتم إيقاف تشغيل ويندوز .

لإعادة إقلاع النظام استخدم التعليمة : ExitWindowsEx(ewx_Reboot,0) .

2 ـ التحكم بلغة البرنامج الافتراضية : إذا كانت اللغة الافتراضية على الجهاز هي اللغة الإنكليزية فإن برنامجـك سيظهر المؤشر في عناصر التحرير بحيث يتم الكتابة باللغة الإنكليزية وعلى المستخدم أن يحول إلى أسلوب الكتابة بالغة العربية كلما أراد الكتابة ، ويمكن حل هذه المشكلة عن طريق التعليمة :

LoadKeyboardLayout('00002801',klf_activate) ;
ابدأ مشروعاً جديداً وضع فيه عنصر تحرير Edit وأضف التعليمة السابقة إلى الحدث OnCreate للنموذج وشغل البرنامج وستلاحظ أن المؤشر أصبح بجهة الكتابة بالغة العربية .
الرقم 00002801 هو الرقم الذي يرمز إلى لوحة المفاتيح العربية السعودية إذا أردت العودة إلى اللغة الإنكليزية الأمريكية بدل هذا الرقم إلى 00000409 .
تطبيق : اذهب إلى برنامج الهاتف وأضف هذه التعليمة للحدث OnCreate للنموذج الأول Form1 .
يوجد لدينا في النافذة إدخال في برنامج الهاتف حقلين يجب كتابتهما باللغة الإنكليزية هما حقل البريد الإلكتروني وحقل موقع الانترنت لذلك يمكنك كتابة التعليمة التالية في الحدث OnEnter لهذين الحقلين :
LoadKeyboardLayout('00000409 ',klf_activate) ;
حيث سيتم التحول إلى اللغة الإنكليزية عند دخول المؤشر إلى هذين الحقلين واكت التعليمة التالية في الحدث OnExit
LoadKeyboardLayout('00002801',klf_activate) ;
إذا أردت أن يعود المؤشر إلى اللغة العربية عن خروج المؤشر من هذين الحقلين .

التابع MessageBox :

تعاملنا مع التابعين ShowMessage وMessageDlg سابقاً ولكن مشكلة هذين التابعين أن عناوين الأزرار تظهر بالغة الإنكليزية دائماً لذلك يمكنك استخدام التابع MEssageBox من ويندوز Api الذي سيظهر عناوين الأزرار تبعاً للغة النظام .

أكتب التعليمة التالية في حدث الضغط على الزر :

MessageBox (0,'هذا نصر الرسالة ' , ' هذا عنوان الرسالة ' , MB_Ok) ;
نفذ البرنامج ولاحظ أن الزر Ok سيظهر بالغة العربية (موافق) إذا كنت تستخدم ويندوز بالغة العربية .
اكتب التعليمة التالية في حدث الضغط على زر آخر :
If MessageBox (0,'هل تريد تلوين النافذة باللون الأحمر' , ' تغير لون النافذة ' ,Mb_YesNo) = IDYES then
Form1.color := clred;
لمزيد من التعليمات ضع المؤشر فوق اسم التابع MessagBox واضغط على F1 .

التعامل مع النوافذ :

لكل نافذة في ويندوز مقبض أو ممسك يدعى Handle وهو من النوع Thandle وهو عبارة عن عدد صحيح طويل يتعامل النظام مع هذا المقبض للوصول إلى النافذة (لا يقصد بالنافذة هنا النموذج Form فقط بل لكل عنصر على النموذج مقبض أيضاً فهناك مقبض للزر وللعنصر Edit و....) ، المقبض ليس له قيمة ثابتة إذ يعطى للنافذة رقم مقبض جديد عند كل إنشاء للنافذة ، عندما تقوم بتحريك نافذة ما على سطح المكتب فإن النظام يستدعي مجموعة من توابع Api لتنفيذ هذه الحركة .

يوجد عدد هائل من التوابع المتعلقة بالنوافذ سنذكر بعض منها في هذه العجالة :

1 ـ برنامج يعيد أسماء جميع النوافذ الموجودة في الذاكرة :

أضف زر و عنصر ListBox إلى النموذج في برنامج جديد واكتب في حدث الضغط على الزر التعليمات التالية :

```
var h:Thandle;
p:pchar;
begin
getmem(p,250);
h:= gettopwindow ( getdesktopwindow) ;
getwindowtext(h,p,250);
listbox1.Items.Add(p);
while h > 0 do begin
h:= getnextwindow( h,gw_hwndnext);
getwindowtext(h,p,250);
listbox1.Items.Add(p)  ;
end;
freemem(p,200);
end;
```

شرح البرنامج :

في البداية تم تعريف المتحول h وهو من النوع Thandle بالإضافة إلى المتحول P من النوع Pchar (ملاحظـة جميع توابع Api تتعامل مع النوع Pchar)

التعليمة GetMem(p,200) تخصص 200 بايت من الذاكرة للمتحول p .

GetDesktopwindow : يعيد مقبض سطح المكتب في ويندوز .

GetTopWindow يعيد النافذة الموجودة في أعلى النوافذ في حيز الذاكرة المخزن فيه مقبض النافذة الذي يمـرر كوسيط لهذا التابع

getwindowtext(h,p,250): يعيد هذا التابع في المتحول p عنوان النافذة التي مقبضها h ويعيد نص بطول 250 حرف

GetNextWindow(h, gw_hwndnext) : يعيد مقبض النافذة التي تلي النافذة h وفي حال وصلنا إلـى آخـر نافذة فيعيد القيمة Nil .

2 ـ برنامج إخفاء وإظهار شريط المهام :

ابدأ مشروعاً جديداً وأضف إليه زرين :

1 ـ عرف المتحول التالي في القسم Private للوحدة :

```
hTaskBar: THandle;
```

2 ـ اكتب التعليمة التالية في الحدث OnCreate للنموذج :

```
hTaskBar := FindWindow('Shell_TrayWnd', nil);
```

تعيد هذه التعليمة النافذة التي نوعها مذكور في الوسيط الأول وعنوانها مذكور في الوسيط الثاني

3 ـ أكتب التعليمة في حدث الضغط على الزر الأول :

```
ShowWindow(hTaskBar, SW_HIDE);
```

تقوم هذه التعليمة بإظهار أو إخفاء النافذة تبعاً للوسيط الثاني .

4 ـ أكتب التعليمة التالية في حدث الضغط على الزر الثاني :

```
ShowWindow(hTaskBar, SW_SHOW);
```

نفذ البرنامج .

ملاحظات :

1 ـ للمزيد من المعلومات عن التوابع المذكورة سابقاً اضغط F1 فوق بعد وضع المؤشر فوق اسم التابع .

2 ـ للحصول على اسماء معظم توابع Windows Api اكتب Windows في أي مكان من محـرر الـشيفرة و ضع النقطة ، فسيتم كتابة جميع الثوابت والتوابع المعرفة في الوحدة Windows.pas وهي مرتبطة مباشـرة مـع Windows Api .

3 ـ للحصول على معلومات وافرة عن Windows Api عليك مراجعة الكتب المختـصة أو الحـصول علـى مجموعة الأقراص الليزرية المسماة MSDN Library والتي تباع مع MicroSoft Visual Studio .

إعداد المهندس : محمد خالد ناصر آغا