API (Application Programming Interface)

API مخفف Application Programming Interface به معنای رابط برنامه نویسی کاربردی است. در واقع API یک کتابخانه سیستمی شامل صدها تابع استاندارد قابل دسترسی است که شرکت Microsoft آنها را داخل یکسری فایل DLL برای برنامه نویسی سیستمی قرار داده است.

دلایل استفاده از توابع API در زبانهای مختلف برنامه‌نویسی می‌تواند این باشد که:

- ✓ توابع API به دلیل آنکه در فایلهای DLL هر سیستم عامل ویندوز می‌باشد و در هر مکان مانند یکدیگر می‌باشند پس نیازی به ارائه آن فایل DLL در نسخه برنامه نمی‌باشد و درنتیجه حجم نسخه کم می‌شود و درضمن سندیت برنامه نیز بیشتر می‌شود و می‌توان گفت که شما از منابع ویندوز به نحو احسن استفاده کرده‌اید.
- ✓ نسخه‌های ویندوز به طور مداوم تغییر می کند ولی به دلیل آنکه سازندگان همیشه حالتی را در نظر می‌گیرند که نسخه‌های قبلی را نیز پشتیبانی کند. درنتیجه اگر شما برنامه‌ای را به کمک توابع API بنویسید با تغییر نسخه ویندوز نیازی به تغییر جدی در توابع API نمی‌باشد.
- ✓ در بعضی از زبانهای برنامه‌نویسی برای آنکه بتوان یک حالت را بوجود آورد و یا کار مشخصی را انجام داد، باید کدهای زیادی بنویسیم و یا در زمان خطاگیری مدت زیادی را صرف کنیم. این موارد هر کدام به نوبه خود می‌توانند از محبوبیت، قدرتمند و خوانابودن برنامه بکاهند. ولی توابع API به دلیل آنکه روتین شده و از قبل نوشته شده می‌باشند پس فقط کافیست تابع را فراخوانی کنیم و به آن ورودی دهیم و خروجی مورد نظر خود را دریافت کنیم.

✓ بیشترتوابع API کارهایی را انجام می‌دهند که زبانهای برنامه‌نویسی قادر به انجام آن نمی‌باشند. به عنوان مثال به تابع SetlateradwindowAttributes مراجعه کنید که باعث می‌شود یک پنجره (فرم و یا کنترلهای داخل آن) و با یک رنگ مشخص در آنها به مقدار دلخواه به حالت شفاف و Transparen تبدیل شوند. و یا توابع دیگر مانند StretchBlt , TransparentBlt , LokworkStation , TimGetTimو SHShutDownDialog و...

توابع API می توانند در فایلهای متعددی تعریف شده باشند ولی مهمترین و پرکاربردترین فایلهای به شرح زیر می باشد:

User32.dll : شامل توابعی می باشد که ابزار و محیط واسط بین سیستم عامل و کاربرد مثل ماوس ، صفحه کلید منوها و پنجره ها را کنترل می کنند.

Kernel32.dll : توابع مربوط به حافظه ، فایل، پارتیشن ، درایو و پوشه در این فایل می باشند.

Gdi32.dll : این فایل که مخفف Graphics Device Interface میباشد، توابع گرافیکی و ترسیمی را در خود دارد.

Netapi32.dll : توابع مربوط به شبکه در این فایل موجود می باشد.

Advapi32.dll : توابع کار با ریجیستری در این موجود می باشد.

Winmm.dll : توابع مربوط به مالتی مدیا دراین فایل تعریف شده اند.

Winspool.drv : توابع مربوط به چاپگر وکار با آن در این فایل می باشند.

Comdlg32.dll : توابع مربوط به حالتهای مختلف پنجره ی common Dialog در این فایل تعریف شده است.

کتابخانه موردنیاز برای فایل های Dll :

```
Using System.Runtime.InteropServices;
```

نحوه ی فراخوانی در سی شارپ :

```
[Dllimport("LibName.dll"),Entity Point = "Name"]

[Public/Private] Static Extern ReturnType AliasName([Parameters]);
```

زمانی که بخواهیم به سگمنتی بغیر از سگمنت خود برنامه اشاره کنیم از Extern استفاده میکنیم.

تابع FindWindow :

کاربرد : این تابع دستگیره(هندل) پنجره مورد نظر را بدست می آورد. معمولا این هندل مورد استفاده ی توابع دیگر قرار میگیرد.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
Public Static Extern IntPtr  FindWindow( String LpClassName , String LpWindowName );
```

اگر تابع کار خود را با موفقیت انجام دهد تابع یک اشاره گر از پنجره ی مورد نظر را برمیگرداند.

پارامترها :

LpClassName : این پارامتر نام کلاس پنجره را بیان می کند.

LpWindowName : این پارامتر عنوان پنجره را بیان می کند.

- ✓ برنامه ای بنویسید که شناسه پنجره برنامه Notepad را نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern IntPtr FindWindow(string LpclassName, string
   LpWindowName);
12.
13.              private void button1_Click(object sender, EventArgs e)
14.              {
15.                  IntPtr p = FindWindow("Notepad", null);
16.                  MessageBox.Show(p.ToString());
17.              }
18.          }
19.      }
```

- ✓ برنامه ای بنویسید که در حال اجرا بودن یا نبودن برنامه Notepad را مشخص کند.

```
1. using System.Runtime.InteropServices;
2.
3. namespace Milad
4. {
```

```
5.     public partial class Form1 : Form
6.     {
7.         public Form1()
8.         {
9.             InitializeComponent();
10.              }
11.
12.              [DllImport("user32")]
13.              public static extern IntPtr FindWindow(string LpclassName, string
   LpWindowName);
14.
15.
16.
17.              private void button1_Click(object sender, EventArgs e)
18.              {
19.                  IntPtr p = FindWindow("Notepad", null);
20.                  if (p.Equals(IntPtr.Zero))
21.                      MessageBox.Show("Not Running");
22.                  else
23.                      MessageBox.Show("Running");
24.              }
25.          }
26.      }
```

تابع GetForegroundWindow :

کاربرد : هندل پنجره ای را که کاربر با آن کار میکند بدست می آورد. به عبارت دیگر پنجره ای را که روی سایر پنجره

های ویندوز قرار دارد، بر میگرداند.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern IntPtr GetForegroundWindow ();

✓ برنامه ای بنویسید که شناسه پنجره ای که روی سایر پنجره های ویندوز قرار دارد را نمایش دهد.

```
1. using System.Runtime.InteropServices;
2.
3. namespace Milad
4. {
5.     public partial class Form1 : Form
6.     {
7.         public Form1()
8.         {
9.             InitializeComponent();
10.              }
11.
12.              [DllImport("user32")]
```

```
13.              public static extern IntPtr GetForegroundWindow();
14.
15.              private void timer1_Tick(object sender, EventArgs e)
16.              {
17.                  IntPtr p = GetForegroundWindow();
18.                  MessageBox.Show(p.ToString());
19.              }
20.
21.          }
22.      }
```

❖ تمرین : برنامه ای بنویسید که در صورتی که پنجره Notepad روی سایر پنجره ها قرار گیرد پیغام مناسب نمایش دهد.(حل شد)

تابع GetActiveWindow :

کاربرد : هندل پنجره فعال را بدست می آورد. پنجره فعال پنجره ای است که روی همه پنجره های ویندوز قرار دارد.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
Public Static Extern IntPtr  GetActiveWindow ();
```

✓ برنامه ای بنویسید که شناسه پنجره فعال را نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern IntPtr GetActiveWindow();
12.              private void timer1_Tick(object sender, EventArgs e)
13.              {
14.                  IntPtr p = GetActiveWindow();
15.                  MessageBox.Show(p.ToString());
16.              }
17.          }
18.      }
```

تابع GetParent :

کاربرد : این تابع هندل والد پنجره ی مورد نظر را بدست می آورد.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern IntPtr GetParent (IntPtr hwnd);

✓ برنامه بنویسید که شامل یک دکمه باشد و با کلیک روی آن Parent آن دکمه نمایش داده شود.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         [DllImport("user32")]
7.         public static extern IntPtr GetParent(IntPtr hwnd);
8.         public Form1()
9.         {
10.                 InitializeComponent();
11.             }
12.             private void button1_Click(object sender, EventArgs e)
13.             {
14.                 IntPtr par = GetParent(button1.Handle);
15.                 MessageBox.Show(par.ToString());
16.             }
17.         }
18.     }
```

تابع FlashWindow :

کاربرد : اگر تابع کار خود را با موفقیت انجام دهد مقدار برگشتی تابع True و در غیراینصورت False خواهد بود.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern bool FlashWindow (IntPtr hwnd , bool blnvert);

پارامترها:

Hwnd : هندل پنجره ای که میخواهیم به صورت چشمک زن شود.

Blnvert : از نوع منطقی بوده و چشمک زن بودن یا نبودن پنجره مورد نظر را مشخص میکند.

نکته : اگر پارامتر دوم را True وارد کنیم ، در ویندوز 7 فقط یکبار چشمک میزند ولی در ویندوز XP یکسره چشمک میزند.

✓ برنامه ای بنویسید که برنامه Notepad یکبار چشمک بزند.

```
1. using System.Runtime.InteropServices;
2.
3. namespace Milad
4. {
5.     public partial class Form1 : Form
6.     {
7.         public Form1()
8.         {
9.             InitializeComponent();
10.              }
11.
12.              [DllImport("user32")]
13.              public static extern IntPtr FindWindow(string LpclassName , string
   LpWindowName);
14.
15.              [DllImport("user32")]
16.              public static extern bool FlashWindow(IntPtr p , bool bln);
17.
18.
19.              private void button1_Click(object sender, EventArgs e)
20.              {
21.                  IntPtr p = FindWindow( "Notepad" , null);
22.                  FlashWindow(p,true);
23.              }
24.
25.
26.          }
27.      }
```

✓ برنامه ای بنویسید که شامل یک دکمه بوده که وقتی روی آن کلیک کردیم برنامه Notepad شروع به چشمک زدن کند و با کلیک دوباره روی آن دیگر چشمک نزند.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }



10.              [DllImport("user32")]
```

```
11.              public static extern IntPtr FindWindow(string LpclassName , string
   LpWindowName);
12.
13.              [DllImport("user32")]
14.              public static extern bool FlashWindow(IntPtr p , bool bln);
15.
16.              private void button1_Click(object sender, EventArgs e)
17.              {
18.                  if (timer1.Enabled == true)
19.                  {
20.                      timer1.Enabled = false;
21.                      button1.Text = "Start";
22.                  }
23.                  else
24.                  {
25.                      timer1.Enabled = true;
26.                      button1.Text = "Stop";
27.                  }
28.              }
29.
30.              private void timer1_Tick_1(object sender, EventArgs e)
31.              {
32.                  IntPtr p = FindWindow("Notepad", null);
33.                  FlashWindow(p, true);
34.              }
35.
36.
37.          }
38.      }
```

نکته : اگر در برنامه بالا از حلقه بینهایت بجای تایمر استفاده کنیم Busy Waiting رخ میدهد.

تابع GetWindowRect :

کاربرد : این تابع مختصات مکان پنجره دلخواه را بدست می آورد.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
Public Static Extern bool GetWindowRect( IntPtr hwnd , out Rectangle lpRect );
```

مقدار بازگشتی : اگر تابع کار خود را با موفقیت انجام دهد مقدار برگشتی تابع True و در غیراینصورت False خواهد بود.

پارامترها:

Hwnd : هندل پنجره ای را که میخواهیم مختصات مکان آن را بدست آوریم.

ipRect : این پارامتر خروجی بوده و از نوع ساختار Rectangle می باشد که مختصات مکان پنجره مورد نظر در این پارامتر قرار میگیرد، در صورت بروز خطا مقادیر صفر در این پارامتر قرار می گیرد.

✓ برنامه ای بنویسید که مختصات پنجره Notepad را نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.    public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern IntPtr FindWindow(string x, string y);
12.              [DllImport("user32")]
13.              public static extern bool GetWindowRect(IntPtr hwnd, out Rectangle
   lpRect);
14.              private void button1_Click(object sender, EventArgs e)
15.              {
16.                  IntPtr hn = FindWindow("Notepad", null);
17.                  Rectangle rec = new Rectangle();
18.                  if (GetWindowRect(hn, out rec))
19.                      MessageBox.Show(rec.ToString());
20.              }
21.          }
22.      }
```

تابع GetWindowText :

کاربرد : عنوان پنجره دلخواه را بدست می آورد.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern bool GetWindowText(IntPtr hwnd , StringBuilder lpString , int nMaxCount);

مقدار برگشتی : اگر تابع کار خود را با موفقیت انجام دهد مقدار برگشتی تابع True و در غیر اینصورت False خواهد بود.

پارامترها:

hWnd : هندل پنجره ای را که می خواهیم عنوان آن را بدست آوریم.

lpString : در واقع این پارامتر خروجی بوده و عنوان پنجره مورد نظر در این پارامتر قرار می گیرد، در صورت بروز خطا مقدار Null در این پارامتر قرار می گیرد.

nMaxCount : طول بافر مورد نظر (lpString) در این پارامتر قرار می گیرد.

✓ برنامه ای بنویسید که عنوان برنامه Notepad را نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.      public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern IntPtr FindWindow(string x, string y);
12.              [DllImport("user32")]
13.              public static extern bool GetWindowText(IntPtr hwnd, StringBuilder
   lpClassName, int nMaxCount);
14.              private void button1_Click(object sender, EventArgs e)
15.              {
16.                  IntPtr hn = FindWindow("Notepad", null);
17.                  StringBuilder sb = new StringBuilder(100);
18.                  if (GetWindowText(hn, sb, sb.Capacity))
19.                      MessageBox.Show(sb.ToString());
20.              }
21.          }
22.      }
```

تابع GetWindowTextLength :

کاربرد : تعداد حروف عنوان یک پنجره دلخواه را بدست می آورد.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern int GetWindowTextLength (IntPtr hwnd);

مقادیر بازگشتی : اگر تابع کار خود را با موفقیت انجام دهد مقدار برگشتی تابع طول عنوان پنجره و در غیر اینصورت خروجی تابع مقدار صفر خواهد بود.

پارامترها:

hWnd : هندل پنجره ای که میخواهیم طول عنوان آن را بدست آوریم.

✓ برنامه ای بنویسید که طول عنوان برنامه Notepad را نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern IntPtr FindWindow(string x, string y);
12.              [DllImport("user32")]
13.              public static extern int GetWindowTextLength(IntPtr hwnd);
14.              private void button1_Click(object sender, EventArgs e)
15.              {
16.                  IntPtr hn = FindWindow("Notepad", null);
17.                  int len = GetWindowTextLength(hn);
18.                  MessageBox.Show(len.ToString());
19.              }
20.          }
21.      }
```

تابع SetForegroundWindow :

کاربرد : این تابع پنجره مورد نظر را بر روی سایر پنجره ها قرار می دهد.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern bool SetForegroundWindow (IntPtr hwnd);

مقادیر بازگشتی : خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها :

Hwnd : هندل پنجره ای که می خواهیم روی سایر پنجره ها قرار گیرد.

✓ برنامه ای بنویسید که برنامه Notepad را روی سایر پنجره ها قرار دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern IntPtr FindWindow(string x, string y);
12.              [DllImport("user32")]
13.              public static extern bool SetForegroundWindow(IntPtr hwnd);
14.              private void button1_Click(object sender, EventArgs e)
15.              {
16.                  IntPtr hn = FindWindow("Notepad", null);
17.                  SetForegroundWindow(hn);
18.              }
19.          }
20.      }
```

تابع EnableWindow :

این تابع پنجره مورد نظر را فعال یا غیر فعال میکند.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern bool EnableWindow (IntPtr hwnd , bool bEnable);

مقادیر بازگشتی :

خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها :

Hwnd : هندل پنجره ای که میخواهیم فعال یا غیرفعال شود.

bEnable : از نوع منطقی بوده و فعال یا غیرفعال بودن پنجره را مشخص میکند.

✓ برنامه ای بنویسید که شامل یک دکمه باشد که با کلیک روی آن برنامه Notepad غیرفعال شده و با کلیک مجدد روی آن برنامه فعال شود.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         static bool b = false;
7.         [DllImport("user32")]
8.         public static extern bool EnableWindow(IntPtr hwnd, bool bEnable);
9.         [DllImport("user32")]
10.             public static extern IntPtr FindWindow(string LpClassName, string
   LpWindowName);
11.             public Form1()
12.             {
13.                 InitializeComponent();
14.             }
15.             private void button1_Click(object sender, EventArgs e)
16.             {
17.                 IntPtr p = FindWindow("Notepad" , null );
18.                 if(!p.Equals(IntPtr.Zero))
19.                 {
20.                     if (!b)
21.                         button1.Text = "Enabled";
22.                     else
23.                         button1.Text = "Disable";
24.                     }
25.                     EnableWindow(p,b);
26.                     b = !b;
27.                     }
28.             }
29.     }
```

تابع SetParent :

کاربرد : این تابع والد یک کنترل خاص را عوض میکند. در هر پنجره ای ممکن است کنترل های زیادی وجود داشته باشد که والد همه آن ها پنجره مورد نظر است، ما می توانیم توسط این تابع آن ها را به برنامه دیگر متصل نماییم.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
Public Static Extern IntPtr SetParent (IntPtr hwndChild , IntPtr hwndNewParent );
```

مقادیر بازگشتی: در صورتی که تابع با موفقیت کار خود را انجام دهد هندل پنجره ای را که قبلا والد این پنجره(کنترل) بوده برمیگرداند و در صورتی که با شکست مواجه شود صفر را بر میگرداند.

پارامترها:

HwndChild : هندل پنجره یا کنترلی که میخواهیم والد آن جابه جا شود.

hwndNewParent : این پارامتر هندل پنجره والد جدید را مشخص می نماید.

- ✓ برنامه ای بنویسید که شامل یک دکمه باشد و با کلیک روی دکمه ، دکمه روی برنامه Notepad انتقال یابد و با کلیک مجدد روی آن روی فرم قرار گیرد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern IntPtr FindWindow(string x, string y);
12.              [DllImport("user32")]
13.              public static extern IntPtr GetParent(IntPtr hwnd);
14.              [DllImport("user32")]
15.              public static extern IntPtr SetParent(IntPtr hwndChild, IntPtr
   hwndNewParent);
16.              private void button1_Click(object sender, EventArgs e)
17.              {
```

```
                IntPtr pb = GetParent(button1.Handle);
                IntPtr hn = FindWindow("Notepad", null);
                if (pb == this.Handle)
                    SetParent(button1.Handle, hn);
                else
                    SetParent(button1.Handle, this.Handle);
            }
        }
    }
```

تابع SetWindowText :

کاربرد: این تابع عنوان یک پنجره خاص را تغییر میدهد.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern bool SetWindowText (IntPtr hwnd , String lpString);

مقادیر بازگشتی : خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در

صورت شکست مقدار False را برمیگرداند.

پارامترها :

Hwnd : هندل پنجره ای که میخواهیم عنوان آن را تغییر دهیم.

lpString : عنوان جدیدی که میخواهیم برای پنجره مورد نظر نشان داده شود.

✓ برنامه ای بنویسید که عنوان پنجره ای که روی تمام پنجره ها قرار دارد را تغییر دهد.

```
using System.Runtime.InteropServices;
namespace Milad
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
          [DllImport("user32")]
          public static extern IntPtr GetForegroundWindow();
          [DllImport("user32")]
          public static extern bool SetWindowText(IntPtr hwnd , string
lpString);
```

```
14.              private void timer1_Tick(object sender, EventArgs e)
15.              {
16.                  IntPtr h = GetForegroundWindow();
17.                  SetWindowText(h ,  textBox1.Text );
18.              }
19.            }
20.      }
```

تابع DestroyWindow :

کابرد: این تابع موجب بسته شدن و از حافظه خارج شدن پنجره مورد نظر می شود.

نحوه ی تعریف :

[Dllimport("user32.dll")]

Public Static Extern bool DestroyWindow (IntPtr hwnd);

مقادیر بازگشتی : خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

Hwnd : هندل پنجره ای که میخواهیم بسته شده و از حافظه خارج شود.

✓ برنامه ای بنویسید که عملکرد تابع DestroyWindow را روی برنامه Notepad نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         [DllImport("user32")]
7.         public static extern IntPtr FindWindow(string x, string y);
8.         [DllImport("user32")]
9.         public static extern bool DestroyWindow(IntPtr hwnd);
10.              public Form1()
11.              {
12.                  InitializeComponent();
13.              }
14.              private void button1_Click(object sender, EventArgs e)
15.              {
16.                  IntPtr p = FindWindow("Notepad", null);
17.                  DestroyWindow(p);
18.              }
19.          }
20.      }
```

تابع CloseWindow :

کاربرد : این تابع باعث Minimize شدن پنجره مورد نظر می شود.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
Public Static Extern bool CloseWindow (IntPtr hwnd );
```

مقادیر بازگشتی : خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

Hwnd : هندل پنجره ای که میخواهیم Minimize شود.

✓ برنامه ای بنویسید که عملکرد تابع CloseWindow را روی برنامه Notepad نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         [DllImport("user32")]
7.         public static extern IntPtr FindWindow(string x, string y);
8.         [DllImport("user32")]
9.         public static extern bool CloseWindow(IntPtr hwnd);
10.              public Form1()
11.              {
12.                  InitializeComponent();
13.              }
14.              private void button1_Click(object sender, EventArgs e)
15.              {
16.                  IntPtr p = FindWindow("Notepad", null);
17.                  CloseWindow(p);
18.              }
19.          }
20.      }
```

تابع OpenIcon :

کاربرد : این تابع پنجره ای را که در حالت Minimize است به حالت نرمال تغییر میدهد.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
Public Static Extern bool OpenIcon (IntPtr hwnd );
```

مقادیر بازگشتی : خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

✓ برنامه ای بنویسید که عملکرد تابع OpenIcon را روی برنامه Notepad نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         [DllImport("user32")]
7.         public static extern IntPtr FindWindow(string x, string y);
8.         [DllImport("user32")]
9.         public static extern bool OpenIcon(IntPtr hwnd);
10.              public Form1()
11.              {
12.                  InitializeComponent();
13.              }
14.              private void button1_Click(object sender, EventArgs e)
15.              {
16.                  IntPtr p = FindWindow("Notepad", null);
17.                  OpenIcon(p);
18.              }
19.          }
20. }
```

تابع MoveWindow :

کاربرد: این تابع موجب جابه جایی و تغییر اندازه پنجره می شود.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
```

Public Static Extern bool MoveWindow(IntPtr hwnd , int x , int y , int nWidth , int nHeight , bool bRepaint);

مقادیر بازگشتی: خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها :

Hwnd : هندل پنجره ای که میخواهیم تغییر اندازه پیدا کرده یا جا به جا شود.

X : این پارامتر مختصات طولی جدید پنجره (سمت چپ پنجره) را مشخص می نماید.

Y : این پارامتر مختصات عرضی جدید پنجره (بالای پنجره) را مشخص می نماید.

nWidth : این پارامتر عرض جدید پنجره را مشخص می کند.

nHeight : این پارامتر ارتفاع جدید پنجره را مشخص می کند.

bRepaint : این پارامتر مشخص می کند که پنجره مجددا ترسیم شود یا خیر. درصورتی که True باشد پنجره از نو ترسیم شده و در صورتی که False وارد کنیم از ترسیم مجدد پنجره خودداری می کند.

✓ برنامه ای بنویسید که طول و عرض برنامه Notepad را نصف حالت فعلی تنظیم کند.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         [DllImport("user32")]
7.         public static extern IntPtr FindWindow(string x, string y);
8.         [DllImport("user32")]
9.         public static extern bool GetWindowRect(IntPtr hwnd, out Rectangle lpRect);
10.             [DllImport("user32")]
11.             public static extern bool MoveWindow(IntPtr hwnd, int x, int y, int
    nWidth, int nHeight, bool bRepaint);
12.             public Form1()
13.             {
14.                 InitializeComponent();
15.             }
16.             private void button1_Click(object sender, EventArgs e)
17.             {
18.                 IntPtr p = FindWindow("Notepad", null);
```

```
19.                 Rectangle rec = new Rectangle();
20.                 GetWindowRect(p, out rec);
21.                MoveWindow(p, rec.X, rec.Y, (rec.Width - rec.X) / 2, (rec.Height -
   rec.Y) / 2, true);
22.             }
23.         }
24. }
```

تابع AnimateWindow :

کاربرد : این تابع حالت انیمیشن و افکت های مختلف برای فرم ، پنجره و یا اجزای داخل فرم را به وجود می آورد.

نحوه ی تعریف :

```
[Dllimport("user32.dll")]
public static extern bool AnimateWindow(IntPtr hwnd , UInt32 Time , UInt32 Flags);
```

مقادیر بازگشتی : در صورتی که تابع با موفقیت کار خود را انجام دهد مقدار True و در غیر اینصورت مقدار False را برمیگرداند.

پارامترها:

Hwnd: هندل پنجره و یا اجزای داخل فرم است که افکت بر روی آن اعمال میشود.

Time : مقدار این پارامتر زمان اجرای افکت به میلی ثانیه می باشد.

Flags : این پارامتر نوع انیمیشن و افکت را مشخص می کند که میتواند یکی از مقادیر ثابت جدول زیر باشد.

عنوان مقدار ثابت	توضیح
AW_HOR_POSITIVE = 0X1;	از چپ به راست باز و یا بسته می شود.
AW_HOR_NEGATIVE = 0X2;	از راست به چپ باز و یا بسته می شود.
AW_VER_POSITIVE = 0X4;	از بالا به پایین باز و یا بسته می شود.
AW_VER_NEGATIVE = 0X8;	از پایین به بالا باز و یا بسته می شود.
AW_CENTER = 0X10;	از وسط آن به اطراف باز می شود.

توضیح	عنوان مقدار ثابت
مخفی می کند.	AW_HIDE = 0X10000;
پنجره را فعال می کند این مقدار را با AW_HIDE به کار نبرید.	AW_ACTIVATE = 0X20000;
حالت انیمیشن به حالت اسلاید.	AW_SLIDE = 0X40000;
حالت انیمیشن به صورت Fade.	AW_BLEND = 0X80000;

✓ برنامه ای بنویسید که شامل دو دکمه و یک جعبه متن باشد که با کلیک روی دکمه اول جعبه متن از چپ به راست در طول پنج ثانیه مخفی شود و با کلیک روی دکمه دوم جعبه متن از راست به چپ در طول پنج ثانیه نمایش داده شود.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern bool AnimateWindow(IntPtr hwnd, UInt32 Time,
   UInt32 Flags);
12.              const uint AW_HOR_NEGATIVE = 0X2;
13.              const uint AW_ACTIVATE = 0X20000;
14.              const uint AW_HOR_POSITIVE = 0X1;
15.              const uint AW_HIDE = 0X10000;
16.              private void button1_Click(object sender, EventArgs e)
17.              {
18.                  AnimateWindow(textBox1.Handle, 5000, AW_HOR_POSITIVE | AW_HIDE);
19.              }
20.             private void button2_Click(object sender, EventArgs e)
21.              {
22.                  AnimateWindow(textBox1.Handle, 5000, AW_HOR_NEGATIVE |
   AW_ACTIVATE);
23.              }
24.          }
25.      }
```

✓ برنامه ای بنویسید که فرم را بصورت محو در طول سه ثانیه ظاهر کند.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern bool AnimateWindow(IntPtr hwnd, UInt32 Time,
   UInt32 Flags);
12.              const uint AW_ACTIVATE = 0X20000;
13.              const uint AW_BLEND = 0X80000;
14.              private void Form1_Load(object sender, EventArgs e)
15.              {
16.                  AnimateWindow(this.Handle, 3000, AW_BLEND | AW_ACTIVATE);
17.              }
18.          }
19.      }
```

✓ برنامه ای بنویسید که دارای دو دکمه باشد که با کلیک روی دکمه اول برنامه Notepad مخفی شده و با کلیک رو دکمه دوم ظاهر شود.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.           [DllImport("user32")]
11.           public static extern bool AnimateWindow(IntPtr hwnd, UInt32 Time,
   UInt32 Flags);
12.           [DllImport("user32")]
13.           public static extern IntPtr FindWindow(String lpClassName, string
   lpWindowName);
14.           const uint AW_HOR_POSITIVE = 0X1;
15.           const uint AW_HIDE = 0X10000;
16.           const uint AW_ACTIVATE = 0X20000;
17.           private void button1_Click(object sender, EventArgs e)
18.           {
19.               IntPtr hwnd = FindWindow("Notepad", null);
```

```
20.                 AnimateWindow(hwnd, 8000, AW_HOR_POSITIVE | AW_HIDE);
21.             }
22.             private void button2_Click(object sender, EventArgs e)
23.             {
24.                 IntPtr hwnd = FindWindow("Notepad", null);
25.                 AnimateWindow(hwnd, 8000, AW_HOR_POSITIVE | AW_ACTIVATE);
26.             }
27.         }
28.     }
```

تابع ActivateKeyboardLayout :

کاربرد: این تابع یکی از زبان های نصب شده در ویندوز را فعال می کند.

نحوه ی تعریف :

```
[DllImport("user32")]
public static extern bool ActivateKeyboardLayout(HKL hkl , uint uFlags );
```

مقادیر بازگشتی : در صورتی که تابع کار خودش را درست انجام دهد خروجی تابع True و در صورتی که با شکست مواجه شود خروجی تابع False خواهد بود.

پارامترها :

Hkl : مقادیر این پارامتر از نوع شمارشی HKL می باشد که در جدول زیر شرح داده شده اند.

uFlags : مقدار این پارامتر را صفر قرار دهید.

توضیح	عنوان مقدار ثابت
زبان بعدی(بعد از زبان جاری) صفحه کلید را بعنوان زبان فعال انتخاب می کند.	HKL_Prev = 0
زبان قبلی(قبل از زبان جاری) صفحه کلید را بعنوان زبان فعال انتخاب می کند.	HKL_Next = 1

نکته : بعضی از توابع یکسری Flag دارند(یک عدد که بعنوان پارامتر به تابع پاس داده می شود و رفتار تابع را مشخص میکند).

✓ برنامه ای بنویسید که زبان جاری سیستم را به زبان بعدی تغییر دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [Flags]
11.              public enum HKL : int
12.              {
13.                  HKL_Prev=0,
14.                  HKL_Next=0,
15.              }
16.              [DllImport("user32")]
17.              public static extern bool ActivateKeyboardLayout(HKL hkl , uint
   uFlags );
18.              private void button1_Click(object sender, EventArgs e)
19.              {
20.                  ActivateKeyboardLayout(HKL.HKL_Next,0);
21.              }
22.          }
23.      }
```

تابع GetKeyState:

کاربرد : این تابع بررسی می کند که آیا کلید مشخص شده(ماوس یا صفحه کلید) در لحظه ی فراخوانی تابع فشار داده شده است یا خیر؟

نحوه ی تعریف :

```
[DllImport("user32")]
public static extern short GetKeyState(Keys nVirtKey);
```

مقادیر بازگشتی :

در صورتی که تابع با شکست مواجه شود مقدار برگشتی تابع صفر خواهد بود، اگر مقدار برگشتی تابع 0x8000 باشد به این معنی است که کلید مشخص شده فشار داده شده است. اگر مقدار خروجی 0x1 باشد به این معنی است که چراغ کلید مشخص شده روشن است(چراغ کلیدهای CapsLock , NumLock , ScrollLock).

پارامترها:

nVirtKey : مقدار این پارامتر کد اسکی کاراکتر می باشد.

✓ برنامه ای بنویسید که مشخص کند کلید CapsLock روشن است یا خیر.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern short GetKeyState(Keys nVirtKey);
12.              private void button1_Click(object sender, EventArgs e)
13.              {
14.                  short t = GetKeyState(Keys.CapsLock);
15.                  if (t == 0)
16.                      MessageBox.Show("CapsLock Off");
17.                  else
18.                      MessageBox.Show("CapsLock On");
19.              }
20.          }
```

تابع ClipCursor:

کاربرد : خروجی این تابع محدوده ی فعالیت اشاره گر ماوس صفحه نمایش می باشد.

نحوه ی تعریف :

```
[DllImport("user32")]
public static extern bool ClipCursor(ref Rectangle lpRect);
```

مقادیر بازگشتی: خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها:

lpRect : این پارامتر به یک ساختار مستطیل اشاره می کند و می توانیم محدوده ی گوشه های بالا و چپ را همراه با گوشه پایین و راست در آن وارد کنیم.

✓ برنامه ای بنویسید که دارای یک دکمه باشد و با کلیک روی دکمه ماوس در محدوده فرم جاری حبس شود.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern bool ClipCursor(ref Rectangle lpRect);
12.              private void button1_Click(object sender, EventArgs e)
13.              {
14.                  Rectangle lp = new Rectangle(this.Left, this.Top, this.Right,
   this.Bottom);
15.                  ClipCursor(ref lp);
16.              }
17.          }
18.      }
```

تابع GetClipCursor:

کاربرد : خروجی این تابع محدوده ی فعالیت اشاره گر ماوس در صفحه نمایش می باشد.

نحوه ی تعریف :

```
[DllImport("user32")]
public static extern bool GetClipCursor(out Rectangle lpRect);
```

مقادیر بازگشتی: خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها :

lpRec :

این پارامتر از نوع ساختار مستطیل می باشد که محدوده گوشه های بالا و چپ را همراه با گوشه پایین و راست فعالیت اشاره گر ماوس مشخص می کند.

✓ برنامه ای بنویسید که عملکرد تابع GetClipCursor را نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern bool GetClipCursor(out Rectangle lpRect);
12.              private void button1_Click(object sender, EventArgs e)
13.              {
14.                  Rectangle lp = new Rectangle();
15.                  if (GetClipCursor(out lp))
16.                      MessageBox.Show(lp.ToString());
17.              }
18.          }
19.      }
```

تابع GetCusorPos:

کاربرد: این تابع مختصات X و Y ماوس را در لحظه فراخوانی مشخص میکند.

نحوه ی تعریف :

```
[DllImport("user32")]
Public static extern bool GetCursorPos(out Point lpPoint);
```

مقادیر بازگشتی: خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها :

lpPoint: این پارامتر از نوع ساختار Point می باشد. این ساختار دارای دو فیلد به نام X و Y می باشد که مختصات جاری اشاره گر ماوس در این دو فیلد ذخیره می شود.

✓ برنامه ای بنویسید که در هر لحظه مختصات X و Y ماوس را روی یک برچسب نمایش دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4. public partial class Form1 : Form
5. {
6.    public Form1()
7.      {
8.          InitializeComponent();
9.      }
10.      [DllImport("user32")]
11.      public static extern bool GetCursorPos(out Point lpPoint);
12.       private void timer1_Tick(object sender, EventArgs e)
13.         {
14.            Point p;
15.            GetCursorPos(out p);
16.            label1.Text = string.Format("Mouse Cursor Position=({0},{1})",p.X,p.Y);
17.            }
18.         }
19.      }
```

تابع SetCursorPos :

کاربرد : این تابع اشاره گر ماوس را به مختصات X و Y داده شده در صفحه انتقال می دهد.

نحوه ی تعریف :

```
[DllImport("user32")]
 Public static extern bool SetCursorPos(int x , int y );
```

مقادیر بازگشتی: خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها:

X : این پارامتر مختصات X را در صفحه مشخص می کند.

Y : این پارامتر مختصات Y را در صفحه مشخص می کند.

✓ برنامه ای بنویسید که اشاره گر ماوس را در مختصات دلخواه قرار دهد.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern bool SetCursorPos(int x , int y );
12.               private void button1_Click(object sender, EventArgs e)
13.              {
14.                  SetCursorPos(500, 500);
15.              }
16.                 }
17.      }
```

تابع ShowCursor :

کاربرد : نشانگر ماوس را مخفی و یا آشکار می کند.

نحوه ی تعریف :

```
[DllImport("user32")]
Public static extern int ShowCursor(bool bShow);
```

مقادیر بازگشتی : در صورتی که تابع کار خود را با موفقیت انجام دهد خروجی تابع عددی مثبت است که نمایانگر مدت زمان نمایش اشاره گر می باشد در صورتی که تابع با خطا مواجه شود خروجی تابع **1-** خواهد بود.

پارامترها:

bShow : مقدار False برای این پارامتر نشانگر ماوس را مخفی می کند و مقدار True نشانگر ماوس را دوباره نشان می دهد.

✓ برنامه ای بنویسید که دارای یک دکمه باشد که با کلیک روی آن اشاره گر ماوس مخفی شده و با کلیک مجدد روی آن اشاره گر ماوس نمایش داده شود.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern int ShowCursor(bool bShow);
12.              static bool CursorHide = false;
13.              private void button1_Click(object sender, EventArgs e)
14.              {
15.                  ShowCursor(CursorHide);
16.                  CursorHide = !CursorHide;
17.              }
18.                 }
19.      }
```

تابع SwapMouseButton :

کاربرد: در تنظیمات پنجره ماوس ، تنظیمی وجود دارد که می تواند کلیک راست و کلیک چپ را با یکدیگر جابه جا کرد این تابع همان کار را انجام می دهد.

نحوه ی تعریف :

```
[DllImport("user32")]
Public static extern bool SwapMouseButton(bool bSwap);
```

مقادیر بازگشتی: خروجی این تابع از نوع منطقی بوده و در صورتی که کار خود را با موفقیت انجام دهد مقدار True و در صورت شکست مقدار False را برمیگرداند.

پارامترها:

fSwap : این پارامتر دو مقدار True و False را می گیرد که مقدار False به این معنی است که کلیک راست و چپ بر سر جای خود برمیگردند(در واقع بصورت همان پیشفرض اولیه ویندوز) و مقدار True به این معنی است که جای کلیک راست و چپ ماوس را تعویض می کند.

✓ برنامه ای بنویسید که جای کلیک چپ و راست ماوس را جابه جا کند.

```
1. using System.Runtime.InteropServices;
2. namespace Milad
3. {
4.     public partial class Form1 : Form
5.     {
6.         public Form1()
7.         {
8.             InitializeComponent();
9.         }
10.              [DllImport("user32")]
11.              public static extern bool SwapMouseButton(bool bSwap);
12.              static bool ButSwap = true;
13.              private void button1_Click(object sender, EventArgs e)
14.              {
15.                  SwapMouseButton(ButSwap);
16.                  ButSwap = !ButSwap;
17.              }
18.                  }
19.      }
```