بسم الله تعالی

موضوع

# آشنایی با توابع Windows API

محمد ارشد طهرانی

خرداد ۱۳۸۷

# چکیده

اگـــر در دایرکتـــوری سیســـتم وینـــدوز نگـــاهی بیانـــدازیم (در وینـــدوزهای برپایـــه ۹۸/۹۵ در آدرس Windows\System\ و در وینـــدوزهای مبتنـــی بـــر NT در آدرس WinNT\System32\) تعـــدادی فایـــل کتابخانه ای(Dynamic Link Library) DLL مشاهده می کنیم. این فایل ها مجموعه توابعی را تشـکیل مـی دهند که برای ایجاد و کنترل اجزای مختلف سیسـتم عامـل، تـامین رابـط کـاربری (User Interface) و محـیط کـاربری اسـتفاده مـی شـوند. ایـن مجموعـه فایلهـا در واقـع (Application Programming Interface) Windows API را تشکیل می دهند.

هدف Windows API این است که به برنامه نویسان اجازه داده شود تا برنامه هایی مبتنی بر سیسـتم عامـل ویندوز و رابط کاربری آن تولید کنند. در واقع به جای اینکه هر کس با توجه به سلیقه های شخصی کدهایی جهـت تولید و کار با اجزاء اصلی ویندوز همچون فرمها، کلیدها، منوها و ... تنظیم کند، همه برنامه نویسان می توانند توابـع اختصاص داده شده مرتبط را که در Windows API وجود دارند صدا کنند و به سیستم عامل اجازه دهند تـا آن اجزاء را ایجاد کند.

برنامه نویسان اصولاً با هدف اجرای عملیاتی، فراتر از آنچه کـه محـیط هـای مختلـف برنامـه نویسـی در حالـت استاندارد خود ارائه می کنند، به استفاده از توابع API روی می آورنـد. بـه عنـوان مثـال فـرض کنـیم زمـانی کـه بخواهیم به محض ایجاد فوکوس بر روی یک Edit Box زبان نوشتاری به فارسی تغییـر کنـد و بـه محـض اینکـه کنترل تغییر کرد زبان کاربری به حالت عادی بازگردد. استفاده از این توابع در مسائل پیچیده گاهاً بسیار مفید مـی باشد.

# فهرست مطالب


| عنوان | صفحه |
|---|---|
| پیشگفتار | ۱ |
| شیوه های کار با تابع | ۲ |
| تفاوت های کتابخانه های ایستا و پیوندی پویا | ۳ |
| Windows API چیست؟ | ۴ |
| علل استفاده از توابع API در برنامه نویسی | ۴ |
| API Wrappers چیست؟ | ۶ |
| فایل های حاوی توابع API | ۶ |
| مشخصات توابع API | ۷ |
| مراجع Windows API | ۸ |
| نحوه ی استفاده از Windows API | ۸ |
| نتیجه گیری | ۹ |
| پیوست | ۱۰ |
| منابع و ماخذ | ۱۴ |
| واژه نامه | ۱۵ |

## فهرست جداول


| شماره جدول | توضیح | صفحه |
|---|---|---|
| ۱ | فایل های حاویی توابع API و کاربرد آن ها | ۷ |


## فهرست اشکال


| شماره شکل | توضیح | صفحه |
|---|---|---|
| ۱ | نمونه ای از استفاده از کتابخانه های ایستا (در زبان C++) | ۲ |
| ۲ | نمونه ای از استفاده از کتابخانه های پویای پیوندی (در زبان VB6) | ۳ |


## فهرست اختصارات

| | |
|---|---|
| API | Application Programming Interface |
| COM | Component Object Model |
| DDL | Dynamic Link Library |
| MSDN | Microsoft Developer Network |
| SDK | Software Development Kit |
| VB6 | Visual Basic 6 |

## پیشگفتار

امروزه با ورود کامپیوتر به عرصه های مختلف، نیاز به نرم افزار در تمامی زمینه ها احساس می شود. و با افزایش روزافزون استفاده از سیستم عامل های ویندوز شرکت مایکروسافت در ایران، اکثر نرم افزارهایی که نوشته شده و یا در حال نوشته شدن است، به گونه ای تهیه می گردند که سازگار و منطبق با استاندارد های سیستم عامل های ویندوز این شرکت باشند.

زبان های شی گرایی برنامه نویسی ابزارهایی را به منظور ایجاد برنامه های مبتنی بر استانداردهای سیستم عامل ویندوز در اختیار برنامه نویسان قرار می دهند، اما این زبان ها قادر نیستند، تمامی امکانات و قابلیت های سیستم عامل ویندوز را در اختیار کاربران قرار دهند، و در چنین مواقعی برنامه نویسان مجبورند، از توابع API کمک بگیرند.

بدلیل عدم وجود منابع فارسی در این زمینه و اهمیت این موضوع برای برنامه نویسان، این بود تا بر آن شدیم، که در این زمینه مطالبی را هر چند کوتاه ولی کاربردی، گردآوری نماییم.

در طول این گزارش ابتدا با مفهوم Windows API آشنا خواهید شد، و سپس به اهمیت و دلایل استفاده از این توابع در برنامه هایمان خواهیم پرداخت و در ادامه برخی از فایل های حاوی این توابع، و نحوه بکارگیری آن ها را مورد بررسی قرار خواهیم داد و در پایان با ذکر چند مثال کاربردی مطالب را به پایان خواهیم رساند، امید است این مطالب مورد استفاده قرار گیرد.

## شیوه های کار با تابع

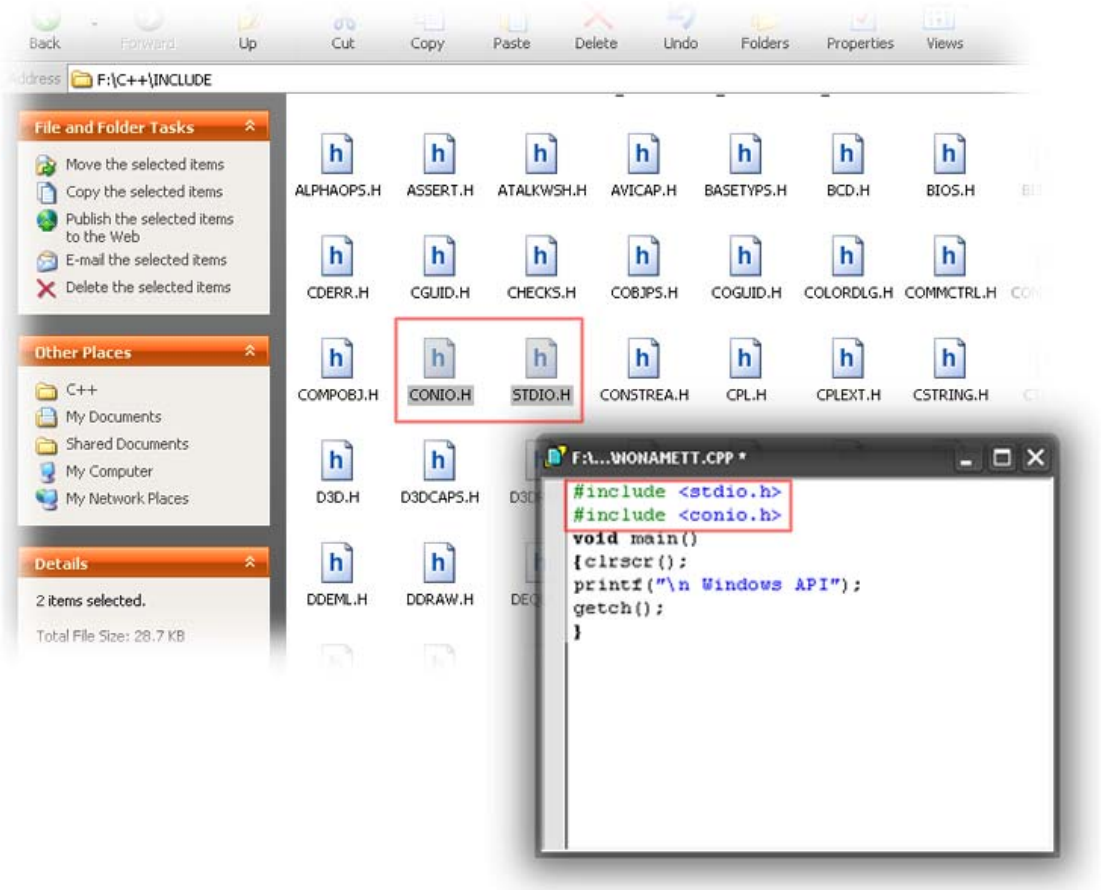
اولین مطلبی که در این زمینه مورد توجه قرار می گیرد، مفهوم کلمه تابع می باشد، تابع برنامه ایست برای حل بخشی از مسئله، که دارای تعدادی پارامتر ورودی و یک مقدار خروجی است. قابلیت هایی که زبان های برنامه نویسی، برای کار با توابع، در اختیار برنامه نویسان قرار می دهند، به شرح زیر می باشد:

- ✓ امکان نوشتن توابع توسط برنامه نویس

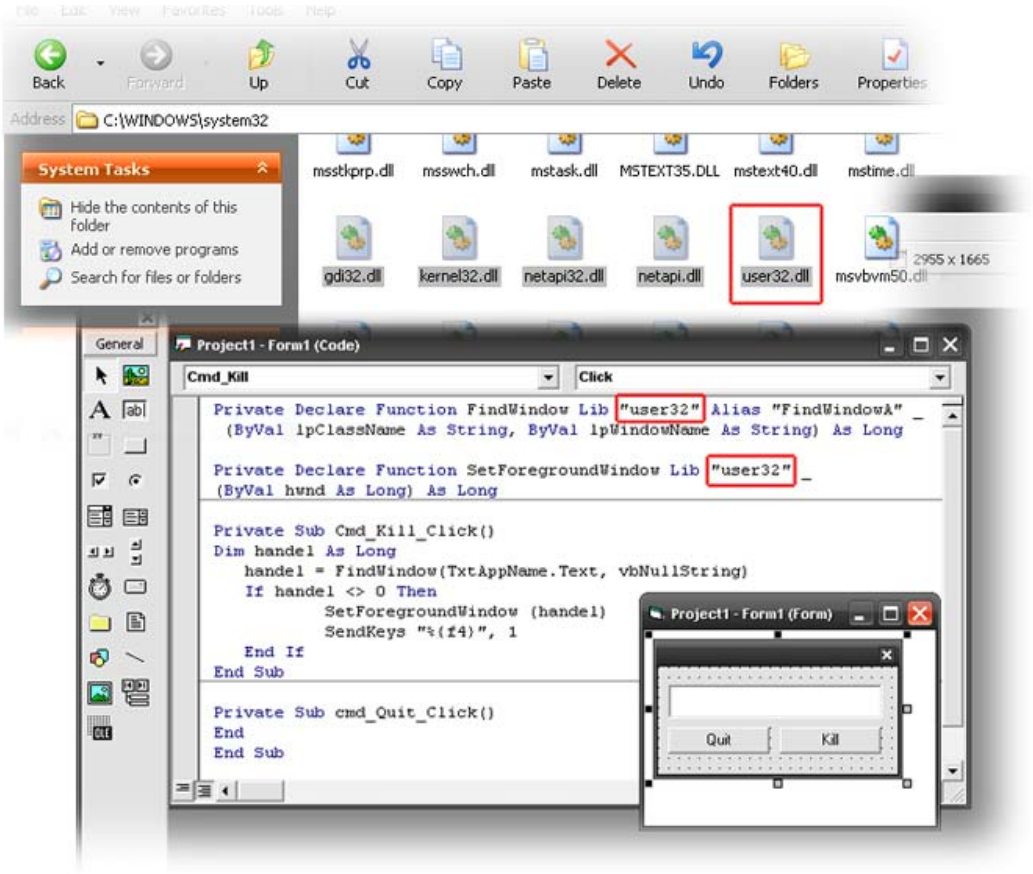
در این روش برنامه نویس، با توجه به هدف خود اقدام به نوشتن تابع مورد نظر می کند.

- ✓ ارایه توابع پرکاربرد و مهم به صورت کتابخانه های ایستا

این گونه توابع به همراه نرم افزارهای برنامه نویسی توسط کمپانی سازنده ارائه می شوند، بدین صورت که توابع پرکاربرد و مهم که برنامه نویسان برای ساخت برنامه به آن ها نیاز دارند را از قبل تهیه و به صورت آماده در اختیار برنامه نویسان قرار می دهند، و برنامه نویس با صدا زدن آن در برنامه خود از امکانات آن استفاده می کند.




شکل ۱ – نمونه ای از استفاده از کتابخانه های ایستا (در زبان C++)




شکل ۲ – نمونه ای از استفاده از کتابخانه های پویای پیوندی (در زبان[۱] VB6)

✓ استفاده از توابع موجود در کتابخانه های پویای پیوندی Dll[۲]

در این روش، توابع درون فایل های dll قرار دارند، و برنامه نویس می تواند این فایل ها را به برنامه خود پیوند زند و از توابع موجود در این فایل ها استفاده نماید، توابع Windows API در این دسته قرار می گیرند، که در طول این گزارش به آن خواهیم پرداخت.

## تفاوت های کتابخانه های ایستا و پیوندی پویا

استفاده از کتابخانه های ایستا ویژگی قابلیت حمل[۳] را تحت تاثیر قرار نمی دهد، یعنی کدهای آن ها با برنامه پیوند ایساتیک دارند و با کامپایل شدن برنامه این کتابخانه ها با کدها برنامه ترکیب شده و یک فایل اجرایی واحد را ایجاد می کند، و در زمان اجرای فایل اجرایی به فایل کتابخانه ای نیازی نیست، در حالی که کتابخانه های پویای

---

۱ .Visual Basic 6

۲ .Dynamic Link Library

۳.Compatibility

پیوندی این ویژگی را ندارند، یعنی کدهای موجود در آن با کدهای برنامه ترکیب نمی شود، بلکه کدهای موجود در آن ها فقط در زمان اجرا در دسترس هستند[۱]. به همین دلیل در زمان اجرا علاوه بر فایل اجرایی، باید فایل کتابخانه ای مورد استفاده نیز در سیستم مقصد وجود داشته باشد، تا برنامه بدون مشکل اجرا شود.

فایل های dll مستقل از زبان هستند و آن ها را می توان در هر زبانی که از COM[۴] پشتیبانی می کند، نوشت[۱]. و این ویژگی به برنامه نویسان این امکان را می دهد تا از یک dll در سایر زبان های برنامه نویسی استفاده نمایند، بدون آنکه نیاز باشد، کد آن را تغییر دهند. در حالی که فایل های کتابخانه ای ایستا فقط قابلیت استفاده در همان زبان برنامه نویسی را دارا می باشند، که با آن ارائه شده اند. (مثلاً از توابع کتابخانه ای ایستا C نمی توان در Delphi یا VB استفاده نمود.)

## Windows API چیست؟

Windows API و یا WinAPI نامیست که مایکروسافت به هسته واسطه‌ای برنامه نویسی کاربردی موجود در سیستم عامل‌های ویندوز داده است[۲]. API مخفف Application Programming Interface می باشد که به معنای رابط برنامه سازی کاربردی می باشد، که از آن به نام های Application Programmer Interface و Application Programmers Interface نیز یاد می شود[۳].

ویندوز برای انجام اعمال مختلف خود از این توابع استفاده می کند. تمام برنامه های کاربردی و برنامه‌های ویندوز به جزء برنامه‌های میز فرمان (console) صرف نظر از زبان تشکیل دهندشان باید با Windows API تعامل داشته باشند[۲]. به عنوان مثال در زبان برنامه نویسی ویژال بیسیک هر خط کد را، که تایپ می کنید و بعد برنامه را اجرا می کنید، توسط توابع API کنترل می شود. هنگامی که دستوری را برای نمایش یک پیغام بر روی صفحه می نویسید، در واقع ویژال بیسیک در پشت صحنه از توابع API، Textout استفاده می کند و یا هنگامی که از توابع رسم بیضی استفاده می کنید، در حقیقت از تابع API، Elliptic استفاده می شود. و همچنین توابع دیگر مثل حرکت ماوس، صفحه کلید، رنگ ها، دکمه ها، منوها و صد ها عملیات دیگر[۳].

این توابع در فایل های کتابخانه ای ویندوز و معمولا در درون فایل های dll قرار دارند.

## علل استفاده از توابع API در برنامه نویسی

دلایل زیادی وجود دارد که نظر یک برنامه نویسان را به سمت استفاده از این توابع معطوف کند، که در اینجا به بیان برخی از مهم ترین آن ها خواهیم پرداخت:

---

۴. Component Object Model

- ✓ به وسیله این توابع برنامه نویسان می توانند برنامه هایی مبتنی بر سیستم عامل ویندوز و رابط کاربری آن تولید کنند، و این امکان را پیدا کنند تا از درون برنامه خود، به اجزای مختلف سیستم عامل ویندوز دسترسی داشته باشند.
- ✓ بیشتر زبان های برنامه نویسی، بخصوص زبان های تحت ویندوز که خود به صورت پنهان از توابع API استفاده می کنند، ممکن است به علت محدودیت هایی نتوانند تمام امکانات این توابع را در اختیار برنامه نویسان قرار دهند. به همین منظور برنامه نویسان می توانند با دسترسی مستقیم به این توابع از حداکثر قابلیت های این توابع استفاده کنند[۳].
- ✓ به دلیل وجود فایل های حاوی توابع API در تمامی نسخه های ویندوز، دیگر نیازی به ارائه این فایل ها به همراه برنامه اصلی نیست، که این موضوع باعث می شود قابلیت حمل[۵] برنامه نیز حفظ شود.
- ✓ استفاده از این توابع باعث کاهش حجم کد برنامه خواهد شد که این خود موجب می شود تا حجم فایل اجرایی برنامه نیز کاهش یابد و در ضمن سندیت برنامه نیز افزایش پیدا کند و می توان گفت که برنامه نویس از یکی از منابع ویندوز به نحو احسن استفاده کرده است.
- ✓ نسخه های ویندوز به طور مداوم تغییر می کند ولی به دلیل آنکه مایکروسافت همیشه حالتی را در نظر می گیرد که نسخه های قبلی را نیز پشتیبانی کند، در نتیجه اگر برنامه ای به کمک توابع API تولید شده باشد با تغییر نسخه ویندوز نیازی به تغییر جدی در توابع API نمی باشد و این موضوع باعث می شود تا سازگاری[۶] برنامه تحت تاثیر قرار نگیرد و برنامه با تمام نسخه های ویندوز سازگار باشد.
- ✓ در بعضی از زبان های برنامه نویسی برای آنکه بتوان یک حالت را بوجود آورد و یا کار مشخصی را انجام داد، برنامه نویس باید تعداد خط های زیادی کد نویسی کند و یا در زمان خطاگیری مدت زیادی را صرف کند . به طور حتم کاربر استفاده کننده از این برنامه نیز باید زمان بیشتری را برای گرفتن جواب صرف کند. این موارد ذکر شده هر کدام به نوبه خود می توانند از محبوبیت، قدرت و خوانا بودن برنامه بکاهند. ولی توابع API به دلیل آنکه روتین شده و از قبل نوشته شده اند، فقط کافیست، برنامه نویس تابع را فراخوانی کند و به آن ورودی دهد و خروجی مورد نظر خود را دریافت کند[۳].
- ✓ استفاده از این توابع، موجب جلوگیری از برنامه نویسی سلیقه ای می گردد ، در واقع به جای اینکه هر کس با توجه به سلایق شخصی کدهایی جهت تولید اجزاء اصلی ویندوز همچون فرم ها، کلیدها، منوها و ... تنظیم کند، همه برنامه نویسان می توانند توابع اختصاص داده شده مرتبط را که در Windows API وجود دارند، را صدا کنند و به سیستم عامل اجازه دهند تا آن اجزاء را ایجاد کند.

و ...

---

۵. Portability

۶. Compatibility

## API Wrappers چیست؟

شاید بیان چنین عنوانی به صورت مستقل چندان منطقی و قابل قبول نباشد اما چون چنین کلماتی در متون مختلف مورد استفاده قرار گرفته و می گیرند، توضیحاتی که به درک این مفاهیم کمک کنند، می توانند مفید باشند.

همانگونه که گفته شد امکان تولید کلیه اجزاء استاندارد سیستم عامل ویندوز از طریق صدا زدن توابع Windows API وجود دارد، علاوه به مزایایی که درباره آن صحبت شد، اما صدا زدن مستقیم توابع API کد برنامه را پیچیده می کند و امکان تولید خطاهای غیرقابل پیش بینی را افزایش می دهد که ایجاد خطاهای سیستمی می کنند و حتی احتمال ضربه زدن به سیستم نیز وجود دارد.

از این رو در نرم افزارهای برنامه نویسی محیط هایی تولید شده اند تا برنامه نویسان را در این امر یاری دهند. کامپایلرهای Delphi و CBuilder و همچنین مترجم Visual Basic و ... همراه خود محیط رابطی دارند که کاربر از طریق آنها به اجزاء استاندارد طراحی دسترسی دارد.

به این صورت برنامه نویس برای تولید اجزاء در برنامه خود نیاز به نوشتن کدهای طولانی ندارد. در این محیط ها توابع مورد نیاز برای ایجاد یک جزء کامل یک جا جمع شده اند و در یک مجموعه قرار گرفته اند و کاربر به راحتی تنها به این مجموعه اشاره می کند، این مجموعه یک API Wrapper است.

اصولا Wrapperها اختصاصا به اجزاء گرافیکی اطلاق نمی گردند. هر مجموعه ای مانند: یک ActiveX، یک Component یا یک تابع ایجاد شده در یک فایل dll و ... می تواند یک API Wrapper باشد. در حالتی که چنین مجموعه هایی در دسترس باشند پیشنهاد شده که از آنها استفاده کنیم.

## فایل های حاوی توابع API

همان طور که گفته شد، توابع API درون فایل هایی قرار دارند، که این فایل ها در ویندوزهای مبتنی برNT در آدرس %WinDir%\SYSTEM32 در دیگر ویندوز ها در آدرس %WinDir%\SYSTEM قرار دارند. که در جدول زیر به مهم ترین و پرکاربرد ترین آن ها اشاره خواهیم کرد:

| نام فایل | توابع موجود در فایل |
|---|---|
| User32.dll | توابع کنترل محیط و واسط گرافیکی ویندوز از قبیل ماوس، صفحه کلید، منوها، پنجره ها و... |
| GDI32.dll | توابع گرافیکی و ترسیمی و توابع کنترل خروجی های ویندوز از قبیل صفحه نمایش، چاپگر و ... |
| Kernel32.dll | توابع کنترلی سخت افزار و نرم افزار از قبیل حافظه، فایل، پارتیشن، درایو، و پوشه و ... |
| Advapi32.dll | توابع کار با Registry |
| Comdlg32.dll | توابعی برای حالت های مختلف Common dialog |
| Shell32.dll | توابعی برای Shell API |
| Lz32.dll | روتین های فشرده سازی |
| Netapi32.dll | توابعی مربوط به شبکه |
| Winmm.dll | توابعی مربوط به Multi Media |
| Winspool.drv | توابعی برای چاپگر و کار با آن |

جدول ۱ – فایل های حاویی توابع API و کاربرد آن ها

## مشخصات توابع API

اگر بخواهیم توابع API را که اکثراً با زبان قدرتمند C نوشته شده اند را بر اساس ساختار سیستم عامل تقسیم بندی کنیم، می توان آن را به دو قسمت ۱۶ بیتی و ۳۲ بیتی تقسیم بندی کرد.

توابع ۱۶ بیتی به تعداد بیش از ۷۳۰ تابع و توابع ۳۲ بیتی بیش از ۶۵۰۰ تایع می رسند. در تابع API مانند هر تابع نوشته شده در زبان های برنامه نویسی، متغییرهای ورودی و خروجی وجود دارند. این متغییرها می توانند از انواع Long, Integer, Int32, String, Byte و ... باشند. در بعضی از توابع API این متغییرها فقط می توانند مقدار ثابت تعریف شده ای را بگیرند. این ثابت ها در نوع ۱۶ بیتی به تعداد بیش از ۱۸۰۰ ثابت و در ۳۲ بیتی به بیش از ۵۵۰۰ ثابت می رسند[۳].

در تعدادی از توابع، متغییر های ورودی و یا خروجی به یک ساختار اشاره دارند که فیلدهای این ساختار اطلاعات ورودی و یا برگشتی تابع را در خود نگهداری می کنند. تعداد این ساختار در نوع ۱۶ بیتس یه بیش از ۵۰ ساختار و در ۳۲ بیتی به بیش از ۹۵۰ ساختار می رسند[۳].

## مراجع Windows API

تا اینجا توابع API را معرفی کردیم، شاید این سوال به ذهن برسد که، ما فایل های حاویی توابع API را شناختیم و حال می دانیم که توابع درون هر فایل مربوط به چه زمینه کاری می باشد، اما چگونه به لیست این توابع، اهداف و نحوه استفاده از آنها پی ببریم؟

اهداف توابع API و نحوه استفاده از آنها در پلت فرم SDK[۷] آمده است و از طریق اسناد مربوطه قابل دسترسی هستند. به عنوان مثال در پکیج MSDN[۸] که همراه با Visual Studio ارائه گردید و یا در ابزارکمک دلفی بخش Windows SDK می توانیم لیست این توابع و نحوه استفاده از آن ها را مشاهده کنیم. علاوه بر این ها امکان دستیابی به این اطلاعات از طریق آدرس زیر نیز وجود دارد:

http://www.microsoft.com/msdownload/platformsdk/setuplauncher.htm

## نحوه ی استفاده از Windows API

نحوه صدا زدن این توابع در زبان های برنامه نویسی مختلف متفاوت است اما اصول کار در تمامی آنها یکسان است. همان طور که در قسمت قبل اشاره کردیم، به همراه کلیه کامپایلرها و مترجم های ارائه شده در بازار که قابلیت استفاده از توابع Windows API را دارند اسنادی در رابطه با نحوه استفاده از این توابع وجود دارد که برنامه نویسان می تواند با مراجعه به آنها با شیوه کار و صدا زدن توابع مورد نظرشان آشنا شود.

نکته مشترک بین تمام محیط های برنامه نویسی این است که برای استفاده از این توابع ابتدا نیاز به بخشی جهت تعریف این توابع می باشد و سپس امکان صدا زدن آنها در بین برنامه ایجاد می گردد. یعنی کار با این توابع دو مرحله دارد:

- تعریف مشخصات تابع مربوطه (Prototype)

- صدا زدن تابع مورد نظر در هر جای برنامه درست همانند دیگر توابع استاندارد (Call)

برای آشنایی بیشتر با نحوه ی استفاده از این توابع به چند مثالی که در پیوست آورده شده است مراجعه کنید.

---

۷. Software Development Kit

۸. Microsoft Developer Network

## نتیجه گیری

به طور خلاصه می توانیم، بیان کنیم که تمام برنامه ها و نرم افزار هایی که در محیط ویندوز اجرا می شوند، در پشت صحنه و به صورت مداوم از توابع API استفاده می کنند، همان طور که در طول این گزارش اشاره شد، بدلیل برخی از محدودیت ها ابزارهای برنامه نویسی نمی توانند تمامی قابلیت های این توابع را در اختیار کاربران خود قرار دهند، اما این امکان را فراهم کرده اند تا برنامه نویسان بتوانند در این محیط ها به صورت مستقیم از این توابع بهره گیرند. پس چه بهتر آن که یک برنامه نویس حرفه ای این توابع را بشناسد، و بتواند در طول برنامه های خود از تمام قابلیت های این توابع استفاده لازم را ببرد.

البته به برنامه نویسان عزیز توصیه می شود که قبل از استفاده از این توابع، حتماً به پکیج های راهنمایی که معرفی شد، مراجعه کنند، استفاده از پکیج های راهنما موجب می شود تا با نحوه کار توابع مورد نظر، پارامترهای مختلف آنها و سایر مطالبی که برای کار با آن ها دانستنش لازم است، آشنایی کامل پیدا کنید.

توجه داشته باشید که، عدم استفاده درست از این توابع در برنامه ها، مشکلات نرم افزاری و سخت افزاری جدی برای کامپیوتر بوجود می آورد.

## پیوست (ارائه چند مثال کاربردی)

همان طور که قبلاً نیز اشاره کردیم بیش از ۶۵۰۰ تابع API ، ۳۲ بیتی داریم و خارج از حوصله و وقت است که بخواهیم به تک تک این توابع بپردازیم ولی، برای آشنایی بیشتر با این توابع، سعی کردیم در چند مثال نحوه ی کار با این توابع را در زبان های دلفی و ویژال بیسیک ۶، توضیح دهیم.

در نظر داشته باشید که در محیط دلفی نیاز به تعریف Prototype این توابع نیست، زیرا این کار از قبل توسط شرکت سازنده (Borland) انجام شده است، و فقط کافیست تابع مورد نظر را صدا (Call) زنیم. برای دسترسی به نام و مشخصات این توابع می توانیم به Windows SDK که همراه با دلفی نصب می شود رجوع کنیم. اما در محیط برنامه نویسی ویژال بیسیک ۶ ابتدا باید Prototype این توابع به صورت کامل و دقیق در قسمت General برنامه تعریف شود، تا بتوانیم در طول برنامه آن ها را صدا زنیم.

بدین منظور یکسری برنامه های جانبی یا Add- Ins برای راحتی کار برنامه نویسان تعبیه شده است. یکی از این برنامه ها برنامه ی API Viewer می باشد که منبع بزرگی از توابع API می باشد. برای استفاده از این برنامه در ویژال بیسیک ۶ از منوی Add-Ins گزینه ی Add-In Manager را انتخاب می کنیم. در این پنجره لیستی از برنامه های جانبی را خواهیم دید. روی گزینه ی VB 6 API Viewer دو بار کلیک می کنیم و گزینه ی Load On Startup را هم در قسمت راست پائین پنجره تیک می زنیم .

حالا دکمه ی Ok را کلیک می کنیم. حال، دوباره به منوی Add-Ins می رویم و گزینه ی API Viewer را انتخاب می کنیم. در این برنامه منوی فایل را باز می کنیم و اولین گزینه یعنی Load Text File را انتخاب می کنیم و از پنجره ی باز شونده فایل متنی WIN32API.txt را باز می کنیم. حالا با جستجوی تابع مورد نظر و دو بار کلیک بر روی هر کدام می توانیم به Syntax آن تابع به راحتی دسترسی داشته باشیم و همچنین می توانیم آن تابع را در قسمت General برنامه در ویژال بیسیک ۶ کپی نماییم.

در این قسمت به توضیح کاربرد چند تابع خواهیم پرداخت.

ایجاد پوشه جدید *(زبان دلفی)*

---

```
Var Result : Boolean
Result := CreateDirectory('C:\Windows API', nil);
```

وظیفه ی تابع CreateDirectory از کتابخانه ی Kernel32.dll ایجاد یک پوشه جدید در یک مسیر مشخص است، که این مسیر به صورت پارامتر ورودی برای آن مشخص می شود.

حذف پوشه *(زبان ویژال بیسیک)*

```
Public Declare Function RemoveDirectory Lib "kernel32" Alias "RemoveDirectoryA" (ByVal lpPathName
As String) As Long
```

```
Dim Result As Long
Result = RemoveDirectory("C:\Windows API");
```

وظیفه ی تابع RemoveDirectory از کتابخانه ی Kernel32.dll حذف یک پوشه در یک مسیر مشخص است، که این مسیر به صورت پارامتر ورودی برای آن مشخص می شود.

تابعی برای تغییر زبان سیستم *(زبان دلفی)*

```
LoadKeyboardLayout ('00000409‘ , KLF_ACTIVATE ); //English keyboard
LoadKeyboardLayout ('00000429‘ , KLF_ACTIVATE ); //Farsi keyboard
```

وظیفه ی تابع LoadKeyboardLayout از کتابخانه ی User32.dll تغییر زبان فعال صفحه کلید می باشد، این تابع بر اساس کد زبانی که بعنوان پارامتر ورودی دریافت می کند (مثلاً '00000429' برای زبان فارسی) زبان فعال صفحه کلید را به آن تغییر می دهد. به عنوان مثال در نظر بگیرید که در برنامه ای، فرمی را برای دریافت اطلاعات از کاربر طراحی کرده اید، و در این فرم کاربر بعضی از گزینه ها را فارسی (مثل نام و نام خانوادگی و ...) و بعضی را نیز باید به زبان انگلیسی (مثل پست الکترونیک) تایپ می کند، برای پر کردن این اطلاعات کاربر باید دائماً به صورت دستی (Alt+Shift) زبان فعال سیستم را تغییر دهد، اما می توانیم با بهره گیری از تابع LoadKeyboardLayout زبان را به صورت خودکار (یعنی با آمدن کنترل[۹] روی textbox به زبان مورد نظر تغییر) دهیم.

اجرای برنامه ها و فایل ها *(زبان دلفی)*

- اجرای برنامه Notepad

```
ShellExecute(Handle, 'open', 'c:\Windows\notepad.exe', nil, nil, SW_SHOWNORMAL) ;
```

- اجرای فایل " test.txt" توسط Notepad

```
ShellExecute(Handle,'open', 'c:\windows\notepad.exe','c:\test.txt', nil,SW_SHOWNORMAL) ;
```

- چاپ محتوای فایل " test.txt"

```
ShellExecute(Handle,'print','c:\test.txt',nil, nil,SW_SHOWNORMAL) ;
```

۹. Focus

▪ نمایش محتوی پوشه "WindowsAPI"

```
ShellExecute(Handle,‘explorer', 'c:\WindowsAPI',nil,nil,SW_SHOWNORMAL) ;
```

▪ باز کردن وب سایت ها توسط جستجوگر اینترنتی پیش فرض

```
ShellExecute(Handle, 'open', 'http://www.Google.com',nil,nil, SW_SHOWMAXIMIZED) ;
```

تابع ShellExecute از کتابخانه ی Shell32.dll توانایی اجرای برنامه های اجرایی (exe) و فایل های مختلف، آدرس سایت، Email و ... نمایش محتوای پوشه ها درون پنجره Windows Explorer و همچنین چاپ فایل های متنی و تصاویر با فرمت های خاص را دارد. برای آشنایی بیشتر با پارامتر های این تابع به Windows SDK مراجعه کنید.

حال می خواهیم به کمک توابع API برنامه ای کاربردی ایجاد کنیم، که مانع از اجرای برنامه notepad شود، برای کد نویسی از محیط ویژال بیسیک ۶ استفاده می کنیم.

در این برنامه از تابع FindWindow برای پیدا کردن نام (Caption) برنامه مورد نظر استفاده می کنیم و از تابع SetForegroundWindow برای بردن کنترل (focus) بر روی برنامه مورد نظر استفاده می کنیم.

در قسمت General کد های زیر را می نویسیم .

```
Public Declare Function FindWindow Lib "user32" Alias "FindWindowA" (ByVal lpClassName As String,
ByVal lpWindowName As String) As Long

Public Declare Function SetForegroundWindow Lib "user32" Alias "SetForegroundWindow" (ByVal hwnd
As Long) As Long
```

و حال یک Timer قرار داده و کد زیر را داخل آن تایپ می کنیم و مقدار Interval آن را برابر ۵۰ قرار می دهیم .

```
Private Sub Timer1_Timer()
Dim handle As Long

handle = FindWindow(vbNullString, " notepad" )

If handle <> 0 Then
SetForegroundWindow handle
SendKeys "%{f4}",1
End If

End Sub
```

پس از اجرا اگر تابع FindWindow نام برنامه مورد نظر را پیدا کرد در متغییر handle مقداری غیر از صفر قرار می دهد و سپس تابع SetForegroundWindow، focus را روی برنامه می برد و سپس توسط تابع SendKeys کلید های Alt+F4 را برای برنامه می فرستیم و برنامه فوراً بسته می شود .

( برای کلید Alt از علامت ٪ ، برای Ctrl از علامت ^ و برای Shift از علامت + استفاده می کنیم ) دقت کنید که حروف کوچک یا بزرگ در نام برنامه مهم است .

برای پی بردن بیشتر به اهمیت این توابع در اینجا فقط به نام برخی از توابعی که بارها در محیط ویندوز کارایی آن ها را مشاهده کرده اید، اشاره خواهیم کرد.

- ساخت منوی و زیر منو (AppendMenu)
- کپی کردن یک فایل با امکان تغییر نام و پسوند (CopyFile)
- تغییر تنظیمات و وضوح صفحه نمایش (ChangeDisplaySettingEx)
- بدست آوردن یک لیست از تمام فونت های موجود در پوشه فونت ویندوز (EnumFonts)
- Logoff, Shutdown, Restart و PowerOn کردن سیستم (ExitWindowsEx)
- ارسال یک متن، عکس و ... به داخل Clipboard (SetClipboardData)
- تغییر کلمه عبور مربوط به کاربرها ScreenSaver و ورودی ویندوز (PwdChangePassword)
- بررسی اینکه آیا آدرس اینترنتی داده شده از لحاظ دستوری درست است یا خیر (PathIsURL)
- اضافه کردن یک آیکون به نوار وظیفه در کنار ساعت به همراه زیر منو (Sell_NotifIcon)

و ...

## منابع و ماخذ

[۱] *علیرضا جباریه*، برنامه سازی۳(دوره آموزش متوسطه فنی و حرفه ای)، چاپ دوم، تهران، شرکت چاپ و نشر کتاب های درسی ایران، ۱۳۸۶

[۲] حسین *صادقی راد*، مرجع توابع API، چاپ و ویراست دوم ، تهران، سازمان چاپ و انتشارات وزارت فرهنگ و ارشاد اسلامی، بهار ۱۳۸۴

[3] http://fa.wikipedia.org/wiki/Windows_API

## واژه نامه

| | |
|---|---|
| Add-Ins | برنامه های اضافه و کمکی |
| API Viewer | ابزاری برای کار با API در ویژال بیسیک ۶ |
| Application Programming Interface | رابط برنامه سازی کاربردی |
| C++ | یکی از قدرتمندترین ابزارهای برنامه نویسی |
| Call | دستوری برای فراخوانی توابع |
| CBuilder | یکی از قدرتمندترین ابزارهای برنامه نویسی |
| Common Dialog | یکی از Component های ویندوز که پنجره های استاندارد مانند جعبه Open و Save و Color و... را در خود دارد. |
| Compatibility | سازگاری |
| Component | مولفه |
| Console | میز فرمان |
| Delphi | یکی از قدرتمندترین ابزارهای برنامه نویسی |
| Dynamic Link Library | کتابخانه های پویای پیوندی |
| Elliptic | بیضی |
| Focus | کنترل |
| General | عمومی |
| Multi Media | چند رسانه ای |
| Portability | قابلیت حمل |
| Prototype | نمونه اولیه |
| Registry | محلی در کامپیوتر برای انجام تنظیمات |
| Syntax | دستور |
| Textout | تابعی برای رسم اشکال بر روی صفحه نمایش |
| Visual Basic 6 | یکی از قدرتمندترین ابزارهای برنامه نویسی |
| Windows | یکی از قویترین سیستم عامل های ساخت شرکت مایکروسافت |