**BOOL AdjustTokenPrivileges(**

**HANDLE**TokenHandle**,**//包含特权的句柄

**BOOL**DisableAllPrivileges**,**//禁用所有权限标志

**PTOKEN\_PRIVILEGES**NewState**,**//新特权信息的指针(结构体)

**DWORD**BufferLength**,**//缓冲数据大小,以字节为单位的PreviousState的缓存区(sizeof)

**PTOKEN\_PRIVILEGES**PreviousState**,**//接收被改变特权当前状态的Buffer

**PDWORD**ReturnLength //接收PreviousState缓存区要求的大小

)

**参数**

**BufferLength**

标志参数PreviousState指针以字节大小缓存区(sizeof).

如果参数PreviousState是NULL,这个参数可以为NULL.

**PreviousState**

这个函数填充一个TOKEN\_PRIVILEGES[结构体](http://baike.haosou.com/doc/6031580-6244581.html" \t "_blank)【[指针](http://baike.haosou.com/doc/1043844-1104112.html" \t "_blank)】,它包括该函数修改之前任何特权状态.这个参数可以为NULL.

如果指定的缓冲区太小，无法收到完整的修改权限列表，这个函数失败并不会修改任何特权.

这个函数设置了一个 拥有修改权限完成列表【 参数ReturnLength 】的字节数 的[指针变量](http://baike.haosou.com/doc/6086286-6299391.html" \t "_blank).[结果的Buffer]

**ReturnLength**

接收 参数PreviousState的缓存区指针的 字节大小 的 变量指针(长度指针).

如果PreviousState为NULL,这个参数可以为NULL.

》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》

typedef struct \_TOKEN\_PRIVILEGES {   
DWORD PrivilegeCount;   
LUID\_AND\_ATTRIBUTES Privileges[];   
} TOKEN\_PRIVILEGES, \*PTOKEN\_PRIVILEGES;

PrivilegeCount指的数组原素的个数

LUID\_AND\_ATTRIBUTES类型的数组声明如下：

typedef struct \_LUID\_AND\_ATTRIBUTES {   
LUID Luid;   
DWORD Attributes;   
} LUID\_AND\_ATTRIBUTES, \*PLUID\_AND\_ATTRIBUTES

第二个参数就指明了我们要进行的操作类型，有三个可选项： SE\_PRIVILEGE\_ENABLED、SE\_PRIVILEGE\_ENABLED\_BY\_DEFAULT、SE\_PRIVILEGE\_USED\_FOR\_ACCESS。要使能一个权限就指定Attributes为SE\_PRIVILEGE\_ENABLED。

第一个参数就是指权限的类型，是一个LUID的值， GUID大家是比较熟悉的，和GUID的要求保证全局唯一不同，LUID只要保证局部唯一，就是指在系统的每一次运行期间保证是唯一的就可以了。

》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》》

例 TOKEN\_PRIVILEGES NewState;

//填充LUID\_AND\_ATTRIBUTES

LUID\_AND\_ATTRIBUTES qxluid[5];

qxluid[0].Luid = \*quanxianluid.lpLuid1;

qxluid[1].Luid = \*quanxianluid.lpLuid2;

qxluid[2].Luid = \*quanxianluid.lpLuid3;

qxluid[3].Luid = \*quanxianluid.lpLuid4;

qxluid[4].Luid = \*quanxianluid.lpLuid5;

qxluid[0].Attributes = SE\_PRIVILEGE\_ENABLED;

qxluid[1].Attributes = SE\_PRIVILEGE\_ENABLED;

qxluid[2].Attributes = SE\_PRIVILEGE\_ENABLED;

qxluid[3].Attributes = SE\_PRIVILEGE\_ENABLED;

qxluid[4].Attributes = SE\_PRIVILEGE\_ENABLED;

//填充TOKEN\_PRIVILEGES

NewState.PrivilegeCount = 5;

NewState.Privileges[0] = qxluid[0];

NewState.Privileges[1] = qxluid[1];

NewState.Privileges[2] = qxluid[2];

NewState.Privileges[3] = qxluid[3];

NewState.Privileges[4] = qxluid[4];

AdjustTokenPrivileges(

TokenHandle, //包含特权的句柄

NULL, //禁用所有权限标志

&NewState, //新特权信息的指针(结构体)

NULL, //缓冲数据大小,以字节为单位的PreviousState的缓存区(sizeof)

NULL, //接收被改变特权当前状态的Buffer

NULL //接收PreviousState缓存区要求的大小

);

**返回值**

如果这个函数成功,返回非0.为了确定这个函数是否修改了所有指定的特权,可以调用**GetLastError**函数,当这个函数返回下面的值之一时就代表函数成功: