

What is Create Window Function .? Explain different arguments we can pass in Create Window Function.

Ans.

This is the function by which we will create a window. In this function we will pass the following arguments.

Create Window

(LPCTSTR classname,LPCTSTR windowname,DWORD style,int x,int y ,int width,int height,HWND hwndparent,HMENU menu,HINSTANCE hinstance, LPVOID param)

There are 11 arguments we will pass in this function.

- 1) LPCTSTR classname :
In this argument we will provide the class name. the class name can be given in two ways.

- a) Predefine Class
In the case of predefine class we will use the following class.

System class	Meaning
BUTTON	This class will make a small rectangular child window that represents a button the user can click to turn one or off. Button can be created alone or in a group. The button control will contain a text that appear at the top of the button window to indicate the purpose of the button. There are different styles are used for button control
COMBOBOX	This class indicate a control that is combination of Listbox and text box. Listbox is used to perform the selection of items and text box is used to accept and modify the selection. COMBOBOX class contains various styles that can be used in windows designing.
EDIT	This class will make a edit control. That will accept the input from the user. Edit control can be classified into two category. The edit control will accept single line and multiline from the user.
LISTBOX	This class will make a listbox. Listbox will contain various object according to user requirement. If inserting is done ascending order then elements will be appear in ascending order and otherwise descending order will appear.
RichEdit	This will make rich edit control on the window. This control contains various options that is not available on other edit control. This control will ensure different types of formatting with different color options.
SCROLLBAR	This class will make a scrollbar that can be classified into two category one is horizontal and second is vertical scrollbar control.
STATIC	This class will make a static control on the window. Static will be used to show only information to the user.

- b) User Define

When you are defining user define class then we will use a structure WNDCLASS structure that contains different members inside it to make user define class.

There are following member are used in this structure.

WNDCLASS Structure

This structure contains window class attributes that is registered with RegisterClass function

This structure contains the following members inside it.

Syntax

```
WNDCLASS {
    UINT style;
    WNDPROC lpfnWndProc;
    int cbClsExtra;
    int cbWndExtra;
    HINSTANCE hInstance;
    HICON hIcon;
    HCURSOR hCursor;
    HBRUSH hbrBackground;
    LPCTSTR lpszMenuName;
    LPCTSTR lpszClassName;
}
```

UINT style;

In this member we will specify the class style that we will use for window creation.

In the case of predefined class we will use the following values inside this arguments.

Constant/value	Description
CS_BYTEALIGNCLIENT	This style of class will align the window's client area in x direction and this style will change the width of client area horizontally
CS_BYTEALIGNWINDOW	This style of class will change the width of the window in x direction as well as width of the window will be shifted horizontally.
CS_CLASSDC	This style will allow to share one device context by all windows in the class. As we know that window classes are process specific and there is chance to create multiple thread of an application to create a window of the same class then multiple threads will use that device context again and again then I will allow one device context to perform one drawing task at one time
CS_DBLCLKS	This style will send a double click message to the window procedure. When user will click double click on the window then mouse related to window procedure then it will execute the message.

CS_DROPSHADOW	This style of the class will enables the drop show effects on the window.
CS_GLOBALCLASS	Indicates that the window class is an application global class.
CS_HREDRAW	Redraws the entire window if a movement or size adjustment changes the width of the client area.
CS_NOCLOSE	Disables Close on the window menu.
CS_OWNDC	This style of class will provide unique device context to each window in which class is used.
CS_PARENTDC	Sets the clipping rectangle of the child window to that of the parent window so that the child can draw on the parent. A window with the CS_PARENTDC style bit receives a regular device context from the system's cache of device contexts. It does not give the child the parent's device context or device context settings. Specifying CS_PARENTDC enhances an application's performance.
CS_SAVEBITS	<p>Saves, as a bitmap, the portion of the screen image obscured by a window of this class. When the window is removed, the system uses the saved bitmap to restore the screen image, including other windows that were obscured. Therefore, the system does not send wm_paint messages to windows that were obscured if the memory used by the bitmap has not been discarded and if other screen actions have not invalidated the stored image.</p> <p>This style is useful for small windows (for example, menus or dialog boxes) that are displayed briefly and then removed before other screen activity takes place. This style increases the time required to display the window, because the system must first allocate memory to store the bitmap.</p>
CS_VREDRAW	Redraws the entire window if a movement or size adjustment changes the height of the client area.

WNDPROC lpfnWndProc

This member will define the window procedure that will handle all operation of the window. In this member we can define a procedure that will handle all messages and action performed by the window.

int cbClsExtra

This member will define the extra bytes allocated to window class structure. System by default initializes the bytes to zero.

int cbWndExtra;

This member will decide number of extra bytes will be allocated to window instance . The system initializes the bytes by zero . if an application use WNDCLASS to register a dialog box then we required resource.h header file then we have to user provide extra memory space to properly handle the dialog box,

HINSTANCE hInstance;

This member will handle the instance that contains the window procedure for the class.

HICON hIcon;

This member will decide which icon will appear on the window. If this argument is given NULL then by default icon will appear on the window other wise we will user LoadIcon function to load the specific icon on the window.

HCURSOR hCursor;

This member will decide which cursor will appear on the window. If this member value is given NULL then by default cursor will appear on the window otherwise we will provide LoadCursor function to load the specific cursor on the window.

HBRUSH hbrBackground

This Member will decide the back ground of the window.
We can change the back ground of the window in the following ways.

- a) Get StockObject() Function using this function we can change the background of the window. In this function we will provide different values to change the back ground color of the window. In this function we can pass the values from

Value	Meaning
BLACK_BRUSH(0)	Black brush.
DKGRAY_BRUSH(1)	Dark gray brush.
DC_BRUSH(2)	Solid color brush.
GRAY_BRUSH(3)	Gray brush.
HOLLOW_BRUSH(4)	Hollow brush
LTGRAY_BRUSH(5)	Light gray brush.
NULL_BRUSH(6)	Null brush
WHITE_BRUSH(7)	White brush.
BLACK_PEN(8)	Black pen.

In
values according

DC_PEN(9)	Solid pen color.
NULL_PEN(10)	Null pen.
WHITE_PEN(11)	White pen.
ANSI_FIXED_FONT(12)	Windows fixed-pitch (monospace) system font.
ANSI_VAR_FONT(13)	Windows variable-pitch (proportional space) system font.
DEVICE_DEFAULT_FONT(14)	Device-dependent font.
DEFAULT_GUI_FONT(15)	Default font for user interface objects such as menus and dialog boxes.
OEM_FIXED_FONT(16)	Original equipment manufacturer (OEM) dependent fixed-pitch (monospace) font.
SYSTEM_FONT(17)	System font. By default, the system uses the system font to draw menus, dialog box controls, and text. It is not recommended that you use DEFAULT_GUI_FONT or SYSTEM_FONT to obtain the font used by dialogs and windows; for more information, see the remarks section. The default system font is Tahoma.
SYSTEM_FIXED_FONT(18)	Fixed-pitch (monospace) system font. This stock object is provided only for compatibility with 16-bit Windows versions earlier than 3.0.
DEFAULT_PALETTE(19)	Default palette. This palette consists of the static colors in the system palette.

this function we can pass 19
to user requirement .

b) CreateSolidBrush

Using this function we can change the background of the window with a solid color that is generated with CreateSolidBrush function.

c) CreateHatchBrush

Using this function we will create hatch brush that is having different types of values

We can pass to make interactive background of the window.

d) CreatePatternBrush

Using this function we will display picture in the back ground of the window. In this function we will provide the picture

LPCTSTR lpszMenuName

In this member we will specify the menu name that we want to display on the window. We will use one macro MAKEINTRESOURCE function to get the name of the menu. There are different functions available to load and manipulate menu according to user requirement.

LPCTSTR lpszClassName

In this member we will specify the class name that will be used in RegisterClass function. The class name should be according the structure of class and functioning of the class.

b) LPCTSTR windowname

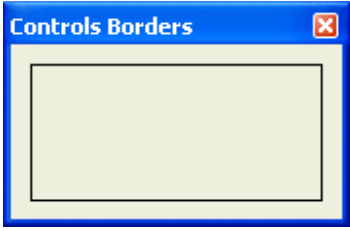
In this arguments we will provide the caption of the window. This window name will appear at the top of the window. If we are using button as a class then window name will appear at top of button.

c) Style

In this option we will specify the style of the window. In this option we can specify the following style of the window.

Window Styles

The following are the window styles. After the window has been created, these styles cannot be modified, except as noted.

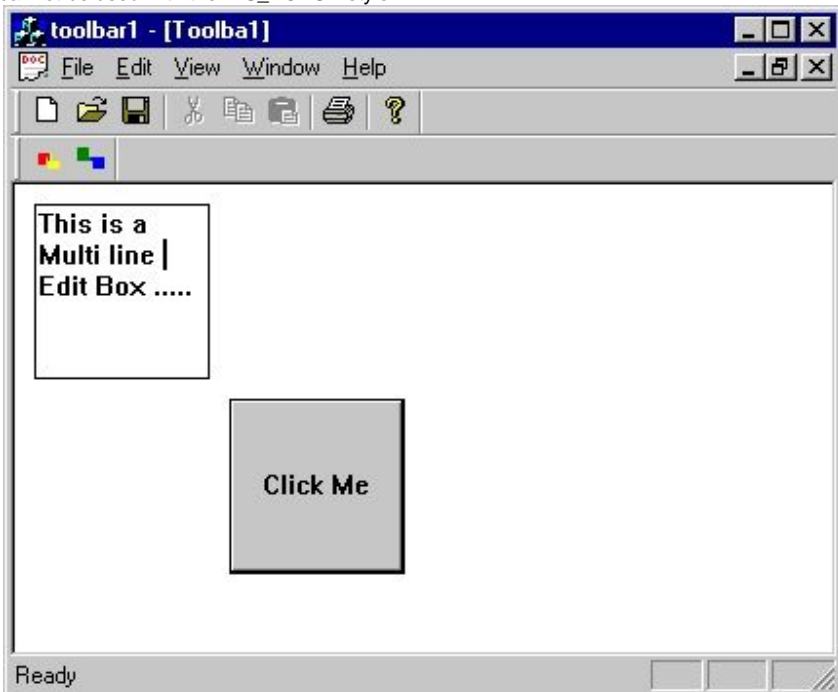
Constant/value	Description
WS_BORDER	<p>The window has a thin-line border.</p>  <p>This style will make a thin line border surrounding the window.</p>
WS_CAPTION	The window has a title bar



This window is having a window name that is Dialog Box

WS_CHILD

The window is a child window. A window with this style cannot have a menu bar. This style cannot be used with the WS_POPUP style.



This window is having two child control. One is edit control and second is command button on the window.

WS_CHILDWINDOW

Same as the WS_CHILD style.

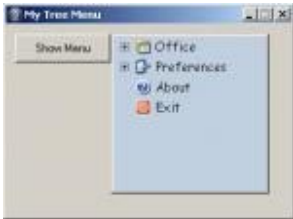
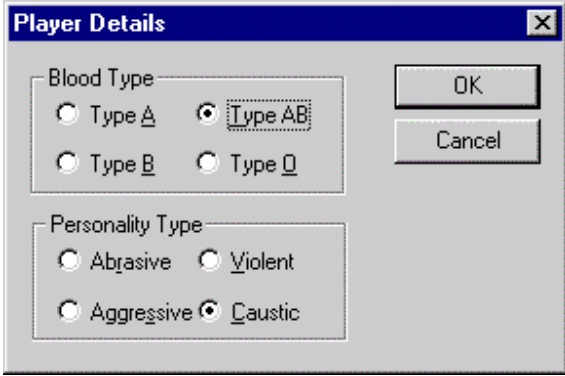

WS_CLIPCHILDREN

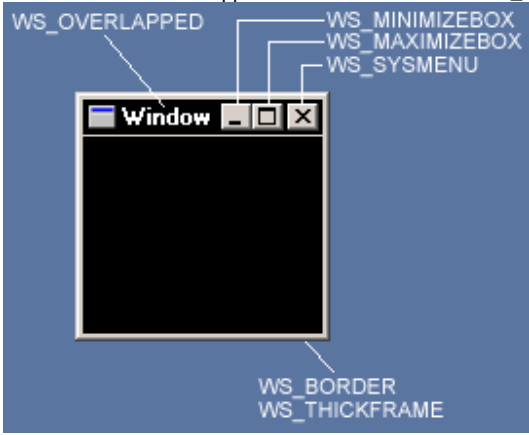
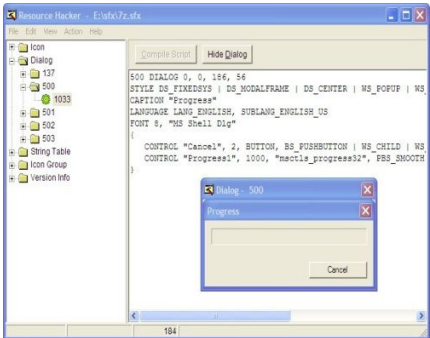

Excludes the area occupied by child windows when drawing occurs within the parent window. This style is used when creating the parent window.

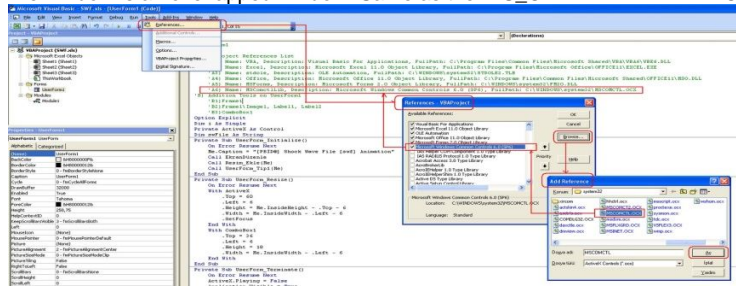


According to this diagram one window is parent window and another is child window.

<p>WS_CLIPSIBLINGS</p>	<p>Clips child windows relative to each other; that is, when a particular child window receives a WM_PAINT message, the WS_CLIPSIBLINGS style clips all other overlapping child windows out of the region of the child window to be updated. If WS_CLIPSIBLINGS is not specified and child windows overlap, it is possible, when drawing within the client area of a child window, to draw within the client area of a neighboring child window.</p> <div data-bbox="608 539 1254 994" data-label="Diagram"> </div> <p>According to this style window has been divided into three parts in other words we can say the window is subdivided into different parts and each part performs a particular task.</p>
<p>WS_DISABLED</p>	<p>The window is initially disabled. A disabled window cannot receive input from the user.</p> <div data-bbox="539 1173 1131 1599" data-label="Image"> </div> <p>According to this diagram the command button 'Start Agent' is in a disabled state.</p>
<p>WS_DLGFAME</p>	<p>The window has a border of a style typically used with dialog boxes. A window with this style cannot have a title bar.</p>

	 <p>According to this diagram one blue portion is coming in blue color that is called dialog frame window style.</p>
WS_GROUP	<p>The window is the first control of a group of controls. The group consists of this first control and all controls defined after it, up to the next control with the WS_GROUP style. The first control in each group usually has the WS_TABSTOP style so that the user can move from group to group. The user can subsequently change the keyboard focus from one control in the group to the next control in the group by using the direction keys.</p>  <p>This window is the example of WS_GROUP because all control is coming in a group.</p>
WS_HSCROLL	<p>The window has a horizontal scrollbar.</p>  <p>This window contains a horizontal scrollbar on the window.</p>
WS_ICONIC	<p>The window is initially minimized. Same as the WS_MINIMIZE style.</p>
WS_MAXIMIZE	<p>The window is initially maximized.</p>
WS_MAXIMIZEBOX	<p>The window has a maximize button. Cannot be combined with the WS_EX_CONTEXTHELP style. The WS_SYSMENU style must also be specified.</p>
WS_MINIMIZE	<p>The window is initially minimized. Same as the WS_ICONIC style.</p>
WS_MINIMIZEBOX	<p>The window has a minimize button. Cannot be combined with the WS_EX_CONTEXTHELP style. The WS_SYSMENU style must also be specified.</p>

<p>WS_OVERLAPPED</p>	<p>The window is an overlapped window. An overlapped window has a title bar and a border. Same as the WS_TILED style.</p>
<p>WS_OVERLAPPEDWINDOW (WS_OVERLAPPED WS_CAPTION WS_SYSMENU WS_THICKFRAME WS_MINIMIZEBOX WS_MAXIMIZEBOX)</p>	<p>The window is an overlapped window. Same as the WS_TILEDWINDOW style.</p> 
<p>WS_POPUP</p>	<p>The windows is a pop-up window. This style cannot be used with the WS_CHILD style.</p>  <p>This diagram indicate that this is Popup Window with the name of Dialog-500</p>
<p>WS_POPUPWINDOW (WS_POPUP WS_BORDER WS_SYSMENU)</p>	<p>The window is a pop-up window. The WS_CAPTION and WS_POPUPWINDOW styles must be combined to make the window menu visible.</p>  <p>This diagram indicate the a popup Window appear on a single window.</p>
<p>WS_SIZEBOX</p>	<p>The window has a sizing border. Same as the WS_THICKFRAME style.</p>
<p>WS_SYSMENU</p>	<p>The window has a window menu on its title bar. The WS_CAPTION style must also be specified.</p>
<p>WS_TABSTOP</p>	<p>The window is a control that can receive the keyboard focus when the user presses the TAB key. Pressing the TAB key changes the keyboard focus to the next control with the WS_TABSTOP style.</p>

	<p>You can turn this style on and off to change dialog box navigation. To change this style after a window has been created, use the SetWindowLong function. For user-created windows and modeless dialogs to work with tab stops, alter the message loop to call the IsDialogMessage function.</p>
WS_THICKFRAME	The window has a sizing border. Same as the WS_SIZEBOX style.
WS_TILED	The window is an overlapped window. An overlapped window has a title bar and a border. Same as the WS_OVERLAPPED style.
WS_TILEDWINDOW (WS_OVERLAPPED WS_CAPTION WS_SYSMENU WS_THICKFRAME WS_MINIMIZEBOX WS_MAXIMIZEBOX)	<p>The window is an overlapped window. Same as the WS_OVERLAPPEDWINDOW style.</p>  <p>This example indicate that window has been sub divided into different other windows.</p>
WS_VISIBLE	The window is initially visible.
WS_VSCROLL	The window has a vertical scroll bar.

int x This argument will decide the horizontal spacing of the window.

int y This argument will decide the vertical spacing of the window.

int width This argument will decide the width of the window.

int height This argument will decide the height of the window.

HWND hwndparent

In this argument we will decide the parent window identification . If we want to create a child window then child window id we have to specify in this arguments other single window contains this arguments null.

HMENU menu

This arguments will contain the name of menu. If argument is passed null then no menu will appear on the window otherwise we have to specify the name of menu with the help of MAKEINTRESOURCE Macro.

HINSTANCE hinstance

This argument will contain the instance of current window. In the case of child window we have to pass the parent window instance.

LPVOID param

In this argument we will pass a pointer to value that is passed to the window with the help of CREATESTRUCT structure.