# TSubclass Component



**Description**

With the Subclass component, you can subclass any window whether that window is a part of your application or another application.  By trapping messages sent by windows to another application's window, you can effectively control that application and / or alter it's behavior.

# Properties

WndHandle
SType

# WndHandle Property

**Description**

The WndHandle property is the window handle of the window to be subclassed. It is not necessary that the window to be subclassed belong to your application.

**Example:**

```
var
        notewnd : THandle;
begin
       notewnd := FindWindow('','notepad');
       If IsWindow(notewnd) then
       begin
           subclass1.WndHandle := notewnd;
           subclass1.Hook; {start subclassing}
      end;
end;
```

The above example subclasses Notepad's main window.

# SType Property

**Description**

The value of the SType Property determines the type of subclassing in effect. The allowable values are stPreDefault and stPostDefault. The default value is stPreDefault. If you want to perform processing before the window's default window procedure is given the message, leave this property stPreDefault. If you want to process after default processing is finished, set this property to stPostDefault.

**Example:**

```
subclass1.SType := stPostDefault;
subclass1.WndHandle := form1.handle;
subclass1.hook;
```

In the above example, the OnSysEvent event handler for the subclass component will be triggered after all default window processing has occurred.

# Methods

Hook
UnHook

# Hook Method

**Description**

The hook method starts the subclassing of the window indicated by the WndHandle property.

**Example:**

```
Subclass1.WndHandle := form1.handle;
Subclass1.hook;
```

# UnHook Method

**Description**

The Unhook method stops subclassing.

**Example:**

```
subclass1.wndhandle := form1.handle;
subclass1.hook; {start}
...
...
...
...
subclass1.unhook;
```

# OnSysEvent Event

**Description**

When a window is being subclassed, all messages sent to it by windows are trapped by the subclass component. For each message sent, the OnSysEvent handler is called. You can put any code you want in the event handler to alter the behavior of the subclassed window.

**Example:**

```
Procedure Subclass1.SysEvent(msg:TMessage;handled:boolean);
begin
      if msg.msg := WM_MENUSELECT then
      begin
        Form1.panel1.caption := 'Menu Item being selected';
        handled := false;
      end;
end;
```

If you set the handled parameter in this event handler to true, the subclass component will NOT pass the message back to the subclassed window's window procedure. The parameter is false when the event handler is fired.

If you want the window's window procedure to be called, leave this parameter false. When the window's window procedure is called depends on the setting of the SType property of the subclass component. Refer to that property for further details.

**See Also:**

SType Property

# stPostDefaul

The subclassed window's window procedure will be called before the OnSysEvent event handler is triggered.

# stPreDefault

The subclassed window's window procedure will be called after any event processing occurs withing the OnSysEvent handler.