Użycie kontrolki TreeView wymaga wykonania kilku czynności konfiguracyjnych przed i w czasie implementacji. Po pierwsze należy tak skonfigurować kompilator aby przy linkowaniu projektu dołączył bibliotekę **comctl32.lib** (common controls). W samym kodzie źródłowym należy dołączyć plik nagłówkowy **commctrl.h**. Poza tym, aby zapewnić załadowanie odpowiednich bibliotek dynamicznych związanych z dołączanymi kontrolkami, przed ich użyciem należy wywołać funkcję **InitCommonControls()**. Od tego momentu kontrolka TreeView jest dostępna.

Poniższy kod tworzy okno kontrolki TreeView :

```
hTree = CreateWindowEx(
      0,                        //style rozszerzone
      WC_TREEVIEW,              //klasa okna
      "Widok - Drzewka",        //nazwa okna
                          //style okna
      WS_BORDER|TVS_HASLINES|TVS_LINESATROOT|TVS_HASBUTTONS|WS_CHILD|WS_VISIBLE,
      3,3,rozmiar.right - 160,rozmiar.bottom - 5,  //położenie i rozmiary
      hwnd,NULL,hInstance,0);  //rodzic, menu, instancja aplikacji, dodatkowe
```

Jedyną różnicą w stosunku do stworzenia zwykłego okna jest zastosowanie identyfikatora WC_TREEVIEW w nazwie klasy, oraz użycie styli specyficznych dla TreeView.

Opis użytych styli (specyficznych dla TreeView):

TVS_HASLINES : rysuj linie w drzewku
TVS_LINESATROOT : rysuj linie w głównej gałęzi
TVS_HASBUTTONS : pokaż przyciski rozwijające poddrzewa

Tak stworzona kontrolka jest już w pełni funkcjonalna. Można do niej dodawać ciągi i tworzyć strukturę drzewa. Kontrolka TreeView posiada możliwość przyporządkowania każdemu przypisanemu jej ciągowi (elementowi) dwóch obrazów. Jednego widocznego gdy element jest wyświetlany zwyczajnie w drzewie, a drugiego gdy element zostanie wybrany (wskazany).
Obrazy widoczne są po lewej stronie ciągów tekstowych, a przypisuje się je elementom w momencie wstawiania ich do kontrolki lub przypisując je istniejącym już elementom drzewa.

Użycie obrazów w kontrolce wymaga utworzenia listy obrazów ImageList, następnie przypisania jej żądanych obrazów oraz skojarzenie tak stworzonej listy z kontrolką TreeView.

Poniższa funkcja przedstawia wszystkie operacje przyporządkowania obrazów kontrolce TreeView :

```
// Funkcja jako parametr otrzymuje uchwyt kontrolki TreeView
// - użyte w funkcji globalne zmienne typu int g_nParent, g_nParentChild i g_nChild
//   po zakończeniu działania funkcji będą zawierały deskryptory wczytanych obrazów

BOOL InitTreeViewImageLists(HWND hwndTV)
{
    HIMAGELIST himl;  // uchwyt listy obrazków
    HBITMAP hbmp;     // uchwyt bitmapy

 // Stworzenie listy obrazów
    if ((himl = ImageList_Create(16, 16, TRUE, 3, 3)) == NULL)
        return FALSE;

       // ładowanie bitmap z zasobów i przypisywanie ich liście obrazów :

    hbmp = LoadBitmap(hInstance, MAKEINTRESOURCE(IDB_PARENT));
    g_nParent = ImageList_Add(himl, hbmp, (HBITMAP) NULL);
    DeleteObject(hbmp);

    hbmp = LoadBitmap(hInstance,MAKEINTRESOURCE(IDB_PARENTCHILD));
    g_nParentChild = ImageList_Add(himl, hbmp, (HBITMAP) NULL);
    DeleteObject(hbmp);
```

```
    hbmp = LoadBitmap(hInstance, MAKEINTRESOURCE(IDB_CHILD));
    g_nChild = ImageList_Add(himl, hbmp, (HBITMAP) NULL);
    DeleteObject(hbmp);

// Zakończ niepowodzeniem jeśli nie wszystkie obrazki załadowane
    if (ImageList_GetImageCount(himl) < 3)
        return FALSE;

// Przypisz listę obrazków do TreeView gdy stworzenie listy powiodło się
    TreeView_SetImageList(hwndTV, himl, TVSIL_NORMAL);

    return TRUE;
}
```

Ostatnim punktem opisu kontrolki TreeView pozostało dodanie do niej elementów.

Aby dodać element do kontrolki należy wypełnić pola dwóch struktur TV_ITEM oraz TV_INSERTITEM (szczegółowy opis pól nie został umieszczony). Poniższy przykład zawiera całą procedurę wstawienia elementu i przypisania mu obrazów :

```
    TV_INSERTSTRUCT tvistruct;    // struktura wsadowa elementu TreeView
    TV_ITEM tvitem;               // struktura opisująca element TreeView
    HTREEITEM hElement;           // uchwyt elementu w TreeView
    char tekst[50];               // tekst elementu wsadowego

    wsprintf( tekst, "Jakiś tekst"); //skomponowanie tekstu

  // ustalenie parametrów struktur wsadowych elementu TreeView :

    tvitem.mask = TVIF_IMAGE | TVIF_SELECTEDIMAGE | TVIF_TEXT;
    //Kolejne parametry maski oznaczają wpisywanie do kontrolki :
    // TVIF_IMAGE - obrazka dla elementu
    // TVIF_SELECTEDIMAGE - obrazka gdy element jest zaznaczony
    // TVIF_TEXT - tekstu elementu

  // przypisanie parametrów zgodnie z ustaloną maską

    tvitem.pszText = tekst;             //przypisanie tekstu elementowi
    tvitem.cchTextMax = lstrlen(tekst);//przypisanie długości tekstu

    //przypisanie obrazków (normalnego i gdy element jest wybrany przez użytkownika)
    tvitem.iImage = g_nParent;
    tvitem.iSelectedImage = g_nParent;
              // g_nParent jest globalnym wskaźnikiem obrazka ustalonym przy
              // tworzeniu listy obrazków (patrz wyżej w opisie)

    tvistruct.hParent = TVI_ROOT;       //ustalenie rodzica wkładanego elementu
    tvistruct.hInsertAfter = TVI_LAST; //włożenie na koniec
    tvistruct.item = tvitem;            //przypisanie struktury opisującej element

      //włożenie elementu do kontrolki wraz z zapamiętaniem jego pozycji
    hElement = (HTREEITEM) SendMessage(hTree,TVM_INSERTITEM,0,
        (LPARAM)(LPTV_INSERTSTRUCT)&tvistruct);
```

Zmianę stanu istniejącego elementu w kontrolce można zrealizować wypełniając pola struktury TV_ITEM określając w niej dodatkowo jakiego elementu tyczą się zmiany (pole **tvitem.hItem**) i wysyłając komunikat:

```
    SendMessage(hTree,TVM_SETITEM,0,(LPARAM)(const TV_ITEM FAR*)&tvitem);
```