

### CDesignArkApp

+ m\_nAppLook : UINT

+ m\_bHiColorIcons : BOOL

+ m\_ActiveView : CView\*

-----

+ smartColour\_String : std::vector<std::vector<Cstring>>

+ smartColour\_Colour : std::vector<COLORREF>

+ commentType : CString

+ commentColour : COLORREF

+ numberColour : COLORREF

+ highlightColour : COLORREF

.\_\_\_\_

+ zoom : float

+ sFont : CString

+ indentSize : int

+ CDesignArkApp()

+ ~CDesignArkApp()

.....

+ InitInstance(): BOOL

+ ExitInstance(): int

-----

+ OnSmartColour(): void

+ OnAppAbout(): void

+ OnOpenSite(): void

+ OnUserdocs(): void

\_\_\_\_\_

# PreLoadState(): void

# LoadCustomState() : void

# SaveCustomState() : void

# DECLARE\_MESSAGE\_MAP()

MFC
CMDIChildWndEx

**CChildFrame** 

## # m\_wndSplitter: CSplitterWndEx

+ CChildFrame()

+ ~CChildFrame()

-----

+ DECLARE\_DYNCREATE(CChildFrame)

-----

+ PreCreateWindow(): BOOL

----- #ifdef \_DEBUG

+ AssertValid() const: void

+ Dump(CDumpContext& dc) const: void

-----

# DECLARE\_MESSAGE\_MAP()

### **CTextHandler**

- text: CString

- shft: BOOL

+ CTextHandler()

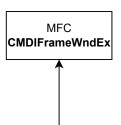
+ ~CTextHandler()

+ RecieveText(): CString

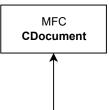
+ DestroyText(): void

+ OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags): BOOL

+ OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags): BOOL

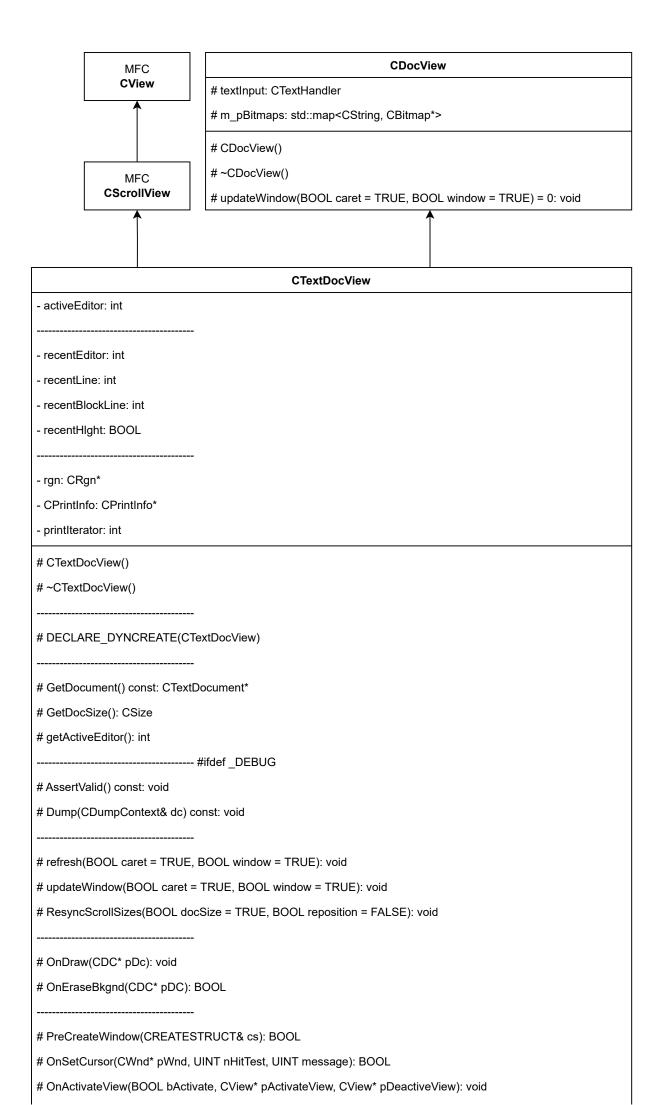


## **CMainFrame** # m\_wndMenuBar: CMFCMenuBar # m\_wndToolBar: CMFCToolBar # m\_wndStatusBar: CMFCStatusBar # m\_UserImages: CMFCToolBarImages + CMainFrame() + ~CMainFrame() + PreCreateWindow(): BOOL + LoadFrame(UINT nIDResource, DWORD dwDefaultStyle = WS\_OVERLAPPEDWINDOW | FWS\_ADDTOTITLE, CWnd\* pParentWnd = nullptr, CCreateContext\* pContext = nullptr): BOOL + DECLARE\_DYNCREATE(CMainFrame) -----#ifdef \_DEBUG + AssertValid() const: void + Dump(CDumpContext& dc) const: void # DECLARE\_MESSAGE\_MAP() # OnCreate(LPCREATESTRUCT lpCreateStruct): int # OnWindowManager(): void # OnViewCustomize(): void # OnToolbarCreateNew(WPARAM wp, LPARAM lp): void # OnApplicationLook(UINT id): void # OnUpdateApplicationLook(CCmdUI\* pCmdUI): void # OnSetActiveView(CView\* pViewNew, BOOL bNotify = TRUE): void - init\_tabs(): int - init\_menubar(): int - init\_toolbar(): int - init\_statusbar(): int - init\_dockingwindows(): int - init\_customisation(): int - init\_userimages(): int - init commands(): int

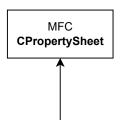


# **CTextDocument** + objects: CArray<CTextEditorObject\*> + defBoxHeight: int + lineOffset: int + filename: CString + CTextDocument() + ~CTextDocument() + DECLARE\_DYNCREATE(CTextDocument) + OnNewDocument(): BOOL + Serialize(CArchive& ar): void + OnFileSave(): void + OnFileOpen(): void -----#ifdef SHARED\_HANDLERS + OnDrawThumbnail(CDC& dc, LPRECT lprcBounds): void + InitializeSearchContent(): void -----#ifdef \_DEBUG + AssertValid() const: void + Dump(CDumpContext& dc): void # DECLARE\_MESSAGE\_MAP() -----#ifdef SHARED\_HANDLERS

# SetSearchContent(const CString& value): void



```
# OnPreparePrinting(CPrintInfo* pInfo): BOOL
# OnBeginPrinting(CDC* pDc, CPrintInfo* pInfo)): void
# OnEndPrinting(CDC* pDc, CPrintInfo* pInfo): void
# OnPrepareDC(CDC* pDC, CPrintInfo* pInfo = (CPrintInfo*)0): void
# OnPrint(CDC* pDC, CPrintInfo* pInfo): void
# OnCreate(LPCREATESTRUCT lpcs): int
# OnInitialUpdate(): void
# OnSize(UINT nType, int cx, int cy): void
# OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar): void
# OnLButtonUp(UINT nFlags, CPoint point): void
# OnLButtonDown(UINT nFlags, CPoint point): void
# OnLButtonDblClk(UINT nFlags, CPoint point): void
# OnMouseMove(UINT nFlags, CPoint point): void
# OnMouseWheel(UINT nFlags, short zDelta, CPoint pt): BOOL
# OnRButtonUp(UINT nFlags, CPoint point): void
# OnRButtonDown(UINT nFlags, CPoint point): void
# OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags): void
# OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags): void
# OnContextMenu(CWnd* pWnd, CPoint point): void
# OnCopy(): void
# OnPaste(): void
# DECLARE_MESSAGE_MAP()
- expandLineBounds(): void
- selectClippingRgn(int nAction): void
- drawHeader(CDC* pDc): void
```



#### **CSmartColour**

# types: CSmartColourTypes

# statements: CSmartColourStatements

# builtin: CSmartColourBuiltin
# other: CSmartColourOther

+ CSmartColour(std::vector<Std::vector<CString>> page\_strings, std::vector<COLORREF> page\_colour,
CString commentType, COLORREF commentColour, COLORREF numberColour, COLORREF highlightColour, CString fonts, CString indents,
LPCTSTR pszCaption = L"", CWnd\* pParentWnd = nullptr, UINT iSelectPage = 0)

+ ~CSmartColour()

-----

# DECLARE\_DYNAMIC(CSmartColour)

# DECLARE MESSAGE MAP()

MFC CMFCPropertyPage

#### **CSmartColourTypes**

-----#ifdef AFX\_DESIGN\_TIME

+ enum { IDD = IDD\_SMARTCOLOUR\_TYPES}

.\_\_\_\_

- + m\_Listbox\_Types\_control: CVSListBox
- + m\_Combo\_Types\_colour\_control: CMFCColorButton
- + m Combo Types colour var: COLORREF
- + m\_listbox\_types\_strings: std::vector<CString>
- + CSmartColourTypes(std::vector<CString> types\_string, COLORREF colour, CWnd\* pParent = nullptr)
- + ~CSmartColourTypes()

-----

- # DoDataExchange(CDataExchange\* pDX): void
- # OnInitDialog(): BOOL
- # OnKillActive(): BOOL

\_\_\_\_\_

# DECLARE\_DYNAMIC(CSmartColourTypes)

# DECLARE\_MESSAGE\_MAP()

### **CSmartColourStatements**

-----#ifdef AFX DESIGN TIME

+ enum { IDD = IDD\_SMARTCOLOUR\_OPS}

+ m Listbox Statements syntax control: CVSListBox

- + m\_Combo\_Statements\_syntax\_control: CMFCColorButton
- + m Listbox Statements statments control: CVSListBox
- + m\_Combo\_Statements\_statments\_control: CMFCColorButton
- + m\_Strings\_Statements\_syntax\_var: std::vector<CString>
- + m\_Colour\_Statements\_syntax\_var: COLORREF
- + m\_Strings\_Statements\_statements\_var: std::vector<CString>
- + m\_Colour\_Statements\_statements\_var: COLORREF
- + CSmartColourStatements(std::vector<CString> syntax\_string, COLORREF syntax\_colour, std::vector<CString> statement\_string, COLORREF statement colour, CWnd\* pParent = nullptr)
- + ~CSmartColourStatements()

-----

- # DoDataExchange(CDataExchange\* pDX): void
- # OnInitDialog(): BOOL
- # OnKillActive(): BOOL

- # DECLARE\_DYNAMIC(CSmartColourStatements)
- # DECLARE\_MESSAGE\_MAP()

#### **CSmartColourOther**

-----#ifdef AFX\_DESIGN\_TIME

+ enum { IDD = IDD\_SMARTCOLOUR\_OTHER}

\_\_\_\_\_

- + m\_pCombo\_comments\_type\_control: CComboBox
- + m\_pCombo\_comments\_colour\_control: CMFCColorButton
- + m pCombo numbers colour control: CMFCColorButton
- + m\_pCombo\_highlight\_colour\_control: CMFCColorButton
- + m\_pCombo\_fonts\_control: CComboBox
- + m pCombo indent control: CEdit
- + m\_pCombo\_comments\_type\_var: CString
- + m\_pCombo\_comments\_colour\_var: COLORREF
- + m pCombo numbers colour var: COLORREF
- + m\_pCombo\_highlight\_colour\_var: COLORREF
- + m\_pCombo\_fonts\_var: CString
- + m pCombo indent var: CString
- + CSmartColourOther(CString comments\_type, COLORREF comments\_colour,COLORREF numbers\_colour, COLORREF highlight\_colour, CString fonts, CString indentSize, CWnd\* pParent = nullptr)
- + ~CSmartColourOther()

-----

- # DoDataExchange(CDataExchange\* pDX): void
- # OnInitDialog(): BOOL
- # OnKillActive(): BOOL

\_\_\_\_\_

- # DECLARE DYNAMIC(CSmartColourOther)
- # DECLARE\_MESSAGE\_MAP()

#### **CSmartColourBuiltin**

-----#ifdef AFX\_DESIGN\_TIME

+ enum { IDD = IDD\_SMARTCOLOUR\_BUILTIN}

-----

- + m\_Listbox\_Builtin\_Classes\_control: CVSListBox
- + m\_Combo\_Builtin\_Classes\_colour\_control: CMFCColorButton
- + m Listbox Builtin Functions control: CVSListBox
- + m\_Combo\_Builtin\_Functions\_colour\_control: CMFCColorButton
- + m\_Listbox\_Builtin\_Classes\_var: std::vector<CString>
- + m Combo Builtin Classes colour var: COLORREF
- + m\_Listbox\_Builtin\_Functions\_var: std::vector<CString>
- + m\_Combo\_Builtin\_Functions\_colour\_var: COLORREF
- + CSmartColourBuiltin(std::vector<CString> classes\_string, COLORREF classes\_colour, std::vector<CString> functions\_string, COLORREF functions\_colour, CWnd\* pParent = nullptr)
- + ~CSmartColourBuiltin()

-----

- # DoDataExchange(CDataExchange\* pDX): void
- # OnInitDialog(): BOOL
- # OnKillActive(): BOOL

-----

- # DECLARE\_DYNAMIC(CSmartColourBuiltin)
- # DECLARE\_MESSAGE\_MAP()

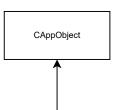
MFC CDialogEx

### **CAboutDlg**

-----#ifdef AFX\_DESIGN\_TIME

- + enum { IDD = IDD\_APP\_ABOUT }
- + CAboutDlg(CWnd\* pParent = nullptr)
- + ~CAboutDlg()
- # DoDataExchange(CDataExchange\* pDX): void
- # DECLARE\_DYNAMIC(CAboutDlg)
- # DECLARE MESSAGE MAP()

## **CAppObject** # ID: CString # bounds: CRect # active: BOOL + CAppObject(CRect bounds, CString ID, BOOL active = FALSE) + ~CAppObject() + draw(CDC\* pDc): void + OnSize(UINT nType, int cx, int cy): void + OnLButtonUp(UINT nFlags, CPoint point): void + OnLButtonDown(UINT nFlags, CPoint point): void + OnLButtonDblClk(UINT nFlags, CPoint point): BOOL + OnRButtonUp(UINT nFlags, CPoint point): void + OnRButtonDown(UINT nFlags, CPoint point): void + OnMouseMove(UINT nFlags, CPoint point): BOOL + OnRecieveText(CString text): int + OnRecieveBackspace(): void + OnRecieveReturn(): void + OnRecieveTab(): void + OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags): void + OnContextMenu(CWnd\* pWnd, CPoint point): BOOL + getID(): CString + setID(CString ID): void + getBounds(): CRect + checkPointInBounds(CPoint point): BOOL + setBounds(CRect bounds): void + setBounds(int x, int y, int cx, int cy): void + setPosition(CPoint point): type + move(int x, int y): void + isActive(): BOOL + setActive(): void



CTextEditorObject
- activeLine: int
- mouseLine: int
- blockLine: int
- printLine: int
- caretPos: int
- defBoxHeight: int
- textExtent: CSize
- hlght: BOOL
- hlghting: BOOL
- hlghtStartP: CPoint
- hlghtEndP: CPoint
- startPos: int
- startLine: int
- recentHlght: type
- recentPos: int
- recentLine: int
- lines: std::vector <ctextlineobject*></ctextlineobject*>
- maxLines: int
- lineOffset: int
- lineNums: BOOL
- hoverSubLine: BOOL
- clickSubLine: BOOL
- sidebar: CRect
- brackets: std::vector <std::vector<int>&gt;</std::vector<int>
- cursor_arrow: int - IMouseDown: BOOL
+ CTextEditorObject(CRect bounds, CString ID, BOOL lineNums, int maxLines = 0, int defBoxHeight = 0, BOOL active = FALSE, std::vector <cstring> text = {L""}, std::vector<int> line = {1}, int lineOffset = 0)</int></cstring>
+ ~CTextEditorObject()
+ draw(CDC* pDc, CSize textExtent, int xScrollPosition, int returnNewLines, BOOL printing, CRect printAreaLength = CRect()): int

```
+ getCaretPos(): int
+ getCaretPoint(CSize caretSize): CPoint
+ getTextExtent(): CSize
+ setTextExtent(CSize size): void
+ getRecentPos(): int
+ pointHighlighted(CPoint point): BOOL
+ hasHighlight(): BOOL
+ hlghtingOff(): void
+ getHighlightClippingRect(): CRect
+ getHighlightExactRgn(int x_offset, int y_offset): CRgn*
+ getStartLine(): int
+ isHlghtMultiline(): BOOL
+ lineHighlight(int line): int
+ iGetLineNum(int a = 0): std::vector <int>
+ sGetLineNum(int a = 0): CString
+ incrementSublines(int subline, int val): void
+ getActiveLine(): int
+ getBlockLine(): int
+ getPrintLine(): int
+ getLineText(int line = 0): CString
+ getLineBounds(int line = 0): CRect
+ getHighlightedText(): CString
+ getLineTextWidth(): int
+ getNumLines(): int
+ getBoxHeight(BOOL default = FALSE): int
+ setActive(BOOL active): void
+ move(int x, int y): void
+ setBounds(CRect bounds): void
+ getPrintBounds(int returnNewLines, int printAreaLength): std::tuple<CRect, int>
+ getCursorArrow(): int
+ initialise(): void
+ OnSize(UINT nType, int cx, int cy): void
+ OnLButtonUp(UINT nFlags, CPoint point): void
+ OnLButtonDown(UINT nFlags, CPoint point): void
+ OnLButtonDblClk(UINT nFlags, CPoint point): BOOL
+ OnRButtonUp(UINT nFlags, CPoint point): void
+ OnRButtonDown(UINT nFlags, CPoint point): void
+ OnMouseMove(UINT nFlags, CPoint point): BOOL
+ OnRecieveText(CString text): int
+ OnRecieveBackspace(): void
+ OnRecieveReturn(): void
+ OnRecieveTab(): void
```

+ OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags): void
+ OnContextMenu(CWnd\* pWnd, CPoint point): BOOL

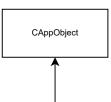
- setSidebar(): void

- setHighlighter(): void

- setBracketsNull(): void

- bracketContains(int value, int& itPos, int type, int side = 1): BOOL

- moveBrackets(int val, int index): void



## **CTextLineObject** - text: CString - highlighter: Gdiplus::Rect - highlight: BOOL - hStart: int - hEnd: int - lineHighlight: BOOL - numSubLines: int - lineNums: std::vector<int> - smartColour: BOOL + CTextLineObject(CRect bounds, CString ID, std::vector<int> lineNums, CString text = L"", BOOL active = FALSE) + ~CTextLineObject() + draw(CDC\* pDc, CSize textExtent, std::vector<std::vector<int>> brackets, int returnNewLines, BOOL printing, int printAreaLength = -1): int + getText(): CString + getLength(): int + setText(CString text): void + concatenateString(CString text, int position): void + backspace(int position): void + isHighlighting(): BOOL + setHighlighting(BOOL val): void + setHighlighter(int pos1, int pos2): void + highlightLine(): void + isLineHighlighted(): BOOL + getHStart(): int + getHEnd(): int + getHighlightedText(): CString + getNumSubLines(): int + iGetLineNums(): std::vector<int> + sGetLineNums(): CString + addSublines(std::vector<int> subs): void + incrementLine(int subline, int val): void + setBounds(CRect bounds): void + move(int x, int y): void

- onlyNums(CString str): BOOL

- brkPosContains(std::vector<std::vector<int>> vector, int value, int side = 1): BOOL

InsertionSort.txt 1

```
1
   2
     // To meet the SQA Advanced Higher Computing Science criteria,
   3 // the program implements a insertion sort in all
    // CSmartColour's property pages when closing them
   4
   5
   6 // The following pseudocode is a design for CSmartColourTypes' sort
     // It will show how it works in practise for the rest of the property
   7
     sheets
   8
   9
    FOR i IN RANGE(1, m Listbox Types control.SizeOf()):
  10
  11
         CREATE variables
 12
 13
         WHILE pos > 0 AND m Listbox Types control[pos-1] > currentValue:
             SET m Listbox Types control[pos] TO m Listbox Types control[po
  14
     s - 1
  15
             pos -= 1
  16
  17
         m Listbox Types control = currentValue
 18
 19
11.1 DECLARE CString currentValue INITIALLY AS m Listbox Types control[i]
11.2 DECLARE INT pos INITIALLY AS i
```