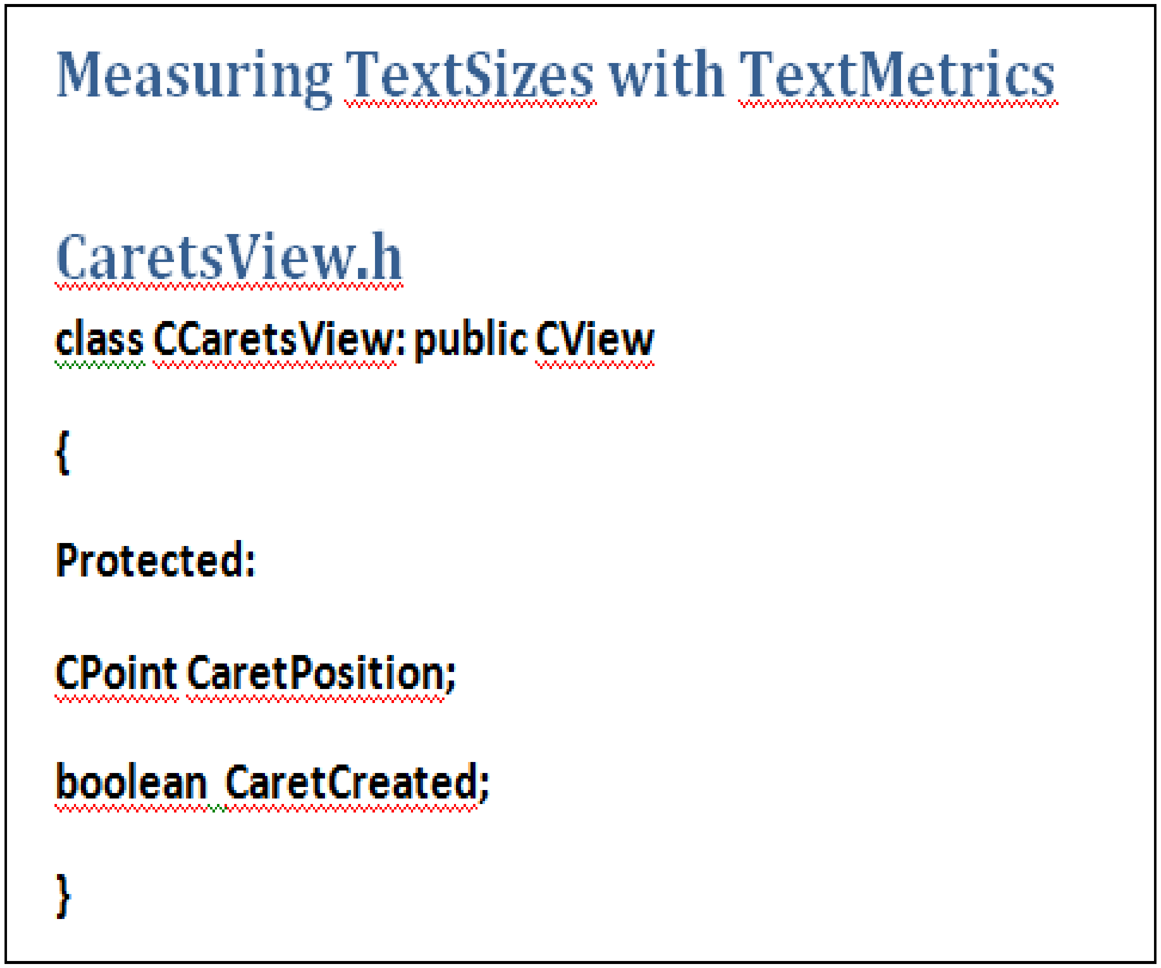
Rapid Application Development

- ## Adding a caret to a window
- The mouse can generate quite a number of events from **WM_LBUTTONDOWN** – i.e. when the user press the left button on the mouse.**WM_MOUSEMOVE** – when the user moves the mouse .When the user clicks a new location it is handled in windows with a caret(called insertion point)
- Use AppWizard to create a SDI program named Carets.
- Write all the same code written in keystroke program.
- We create a new caret and decide the size of the caret. A caret is usually made the same height as the current character and 1/8 of the width of average character.
- To determine the **height** and **width** of characters we use CDC method **GetTextMetrics();**

Creating an Application to make a cursor



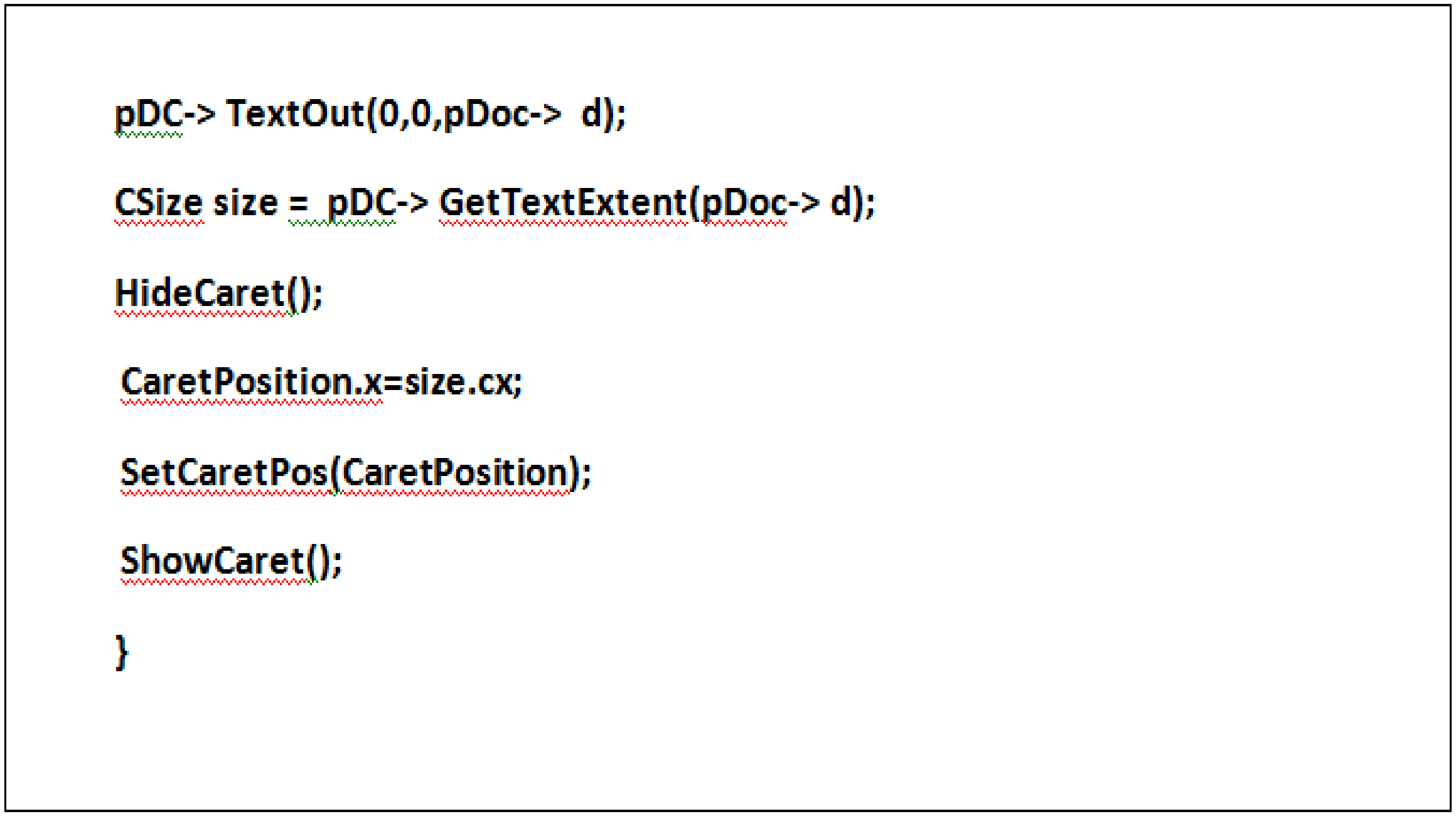
Measuring TextSizes with TextMetrics

CaretsView.h

```
class CCaretsView: public CView

{

Protected:

CPoint CaretPosition;

boolean CaretCreated;

}
```

CaretView.cpp

```
void CCaretsView :: OnDraw( CDC * pDC)

{  CCaretsDoc* pDoc = GetDocument();

ASSERT_VALID(pDoc);

TEXTMETRIC t;

pDC -> GetTextMetric(&t);

CreateSolidCaret(t.tmAveCharWidth/8,t.tmHeight);

CaretPosition.x = CaretPosition.y =0;

SetCaretPos(CaretPosition);

ShowCaret();

CaretCreated = true;
```



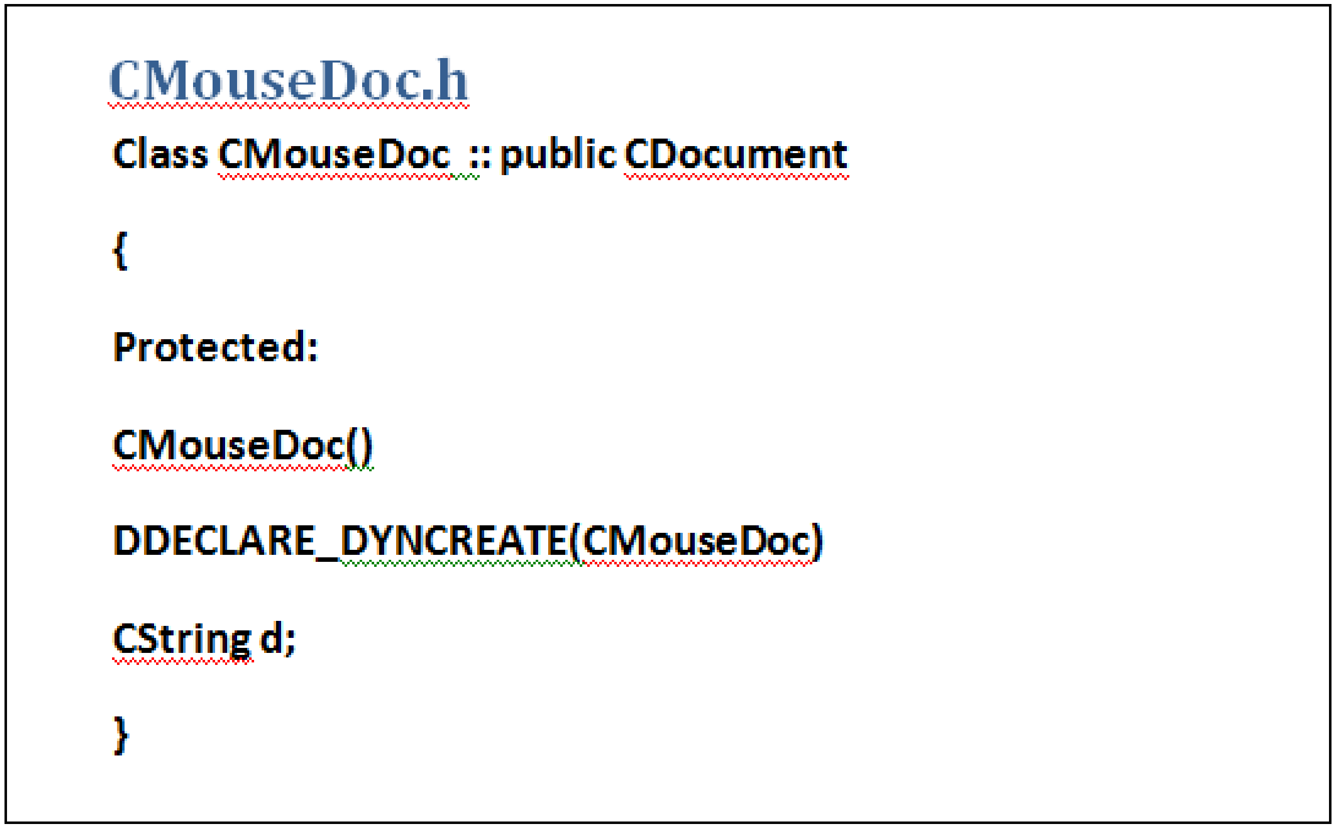
```
pDC-> TextOut(0,0,pDoc->  d);

CSize size =  pDC-> GetTextExtent(pDoc-> d);

HideCaret();

 CaretPosition.x=size.cx;

 SetCaretPos(CaretPosition);

 ShowCaret();

}
```

- The Caret method also include ShowCaret(),SetCaretPos(), and HideCaret().We make the caret the same height as our text using textmetric .tmHeight and 1/8 th of the width of average character.we call **CreateSolidCaret()** to actually create the caret.
- **Setting the Caret's Position**
- We store the caret's position in a new **CPoint** object named **CaretPosition**.CPoint object named CaretPosition .CPoint class has two data data members x and y which will hold the position of the caret.
- **CaretPosition.x = CaretPosition.y =0;**
- Now we select the Caret's position with **SetCaretPos()-** shows the caret's position.**ShowCaret()-** It shows the caret on the screen and set the **CaretCreated** boolean flag to true.

- **SetCaretPos(CaretPosition);**
- **ShowCaret();**
- **CaretCreated = true;**
- The caret appears on the screen as the blinking function
- The next step is to move the caret as the user type text.
- **pDC-> TextOut(0,0,pDoc-> StringData);**
- Now we have to determine the end of string where we can place the caret .we do this by **CSize** object named “size” using **GetTextExtent();**
- **CSize size = pDC-> GetTextExtent(pDoc-> StringData);**
- To display caret at the end of the text string we first hide it using **HideCaret().** Next we set x data member of caret position point at the end of text string.

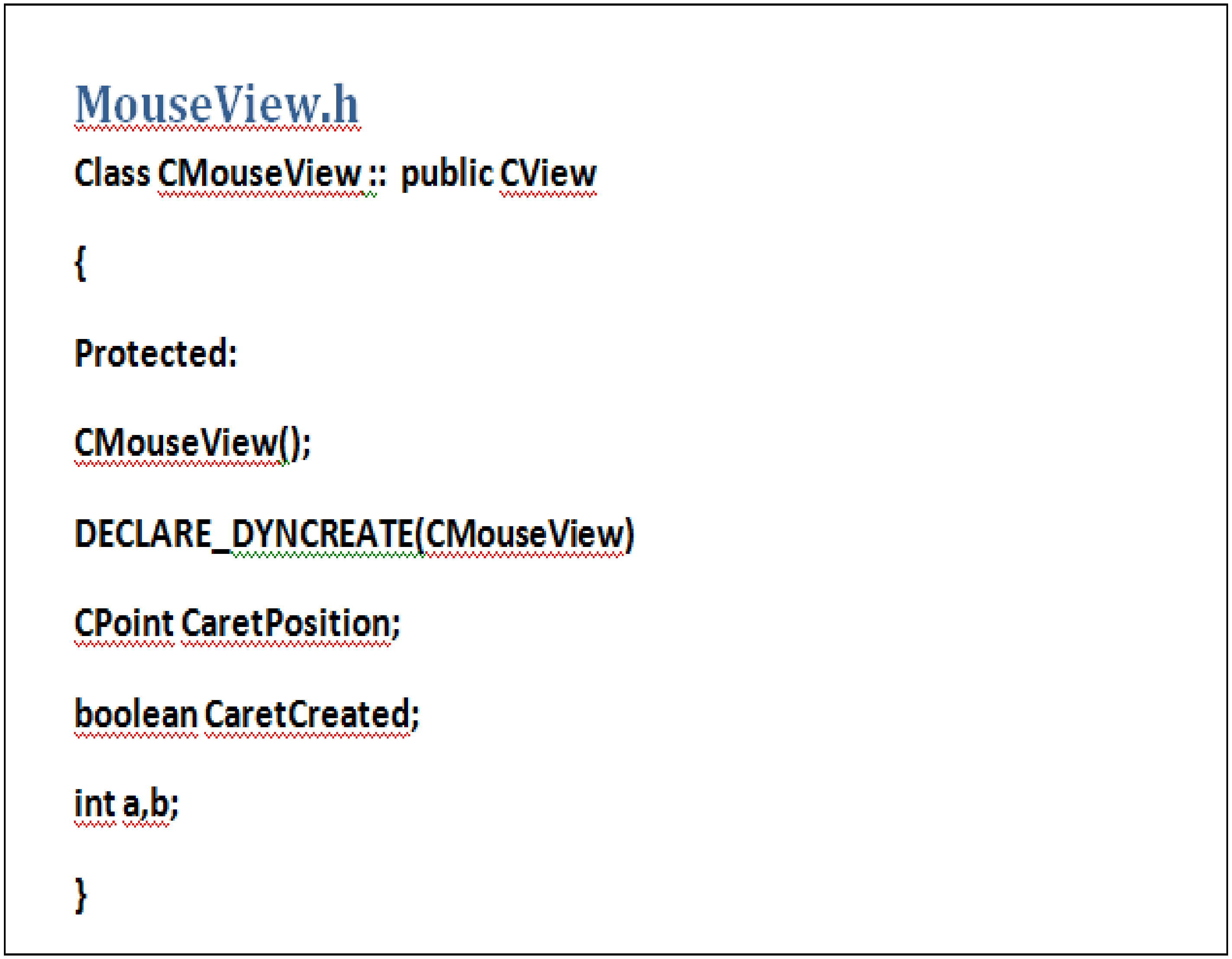
- **CARETPOSITION.X=SIZE.CX;**
- **SETCARETPOS(CARETPOSITION);**
- **SHOWCARET();**

Mouse Handling



CMouseDoc.h

```
Class CMouseDoc  :: public CDocument

{

Protected:

CMouseDoc()

DDECLARE_DYNCREATE(CMouseDoc)

CString d;

}
```

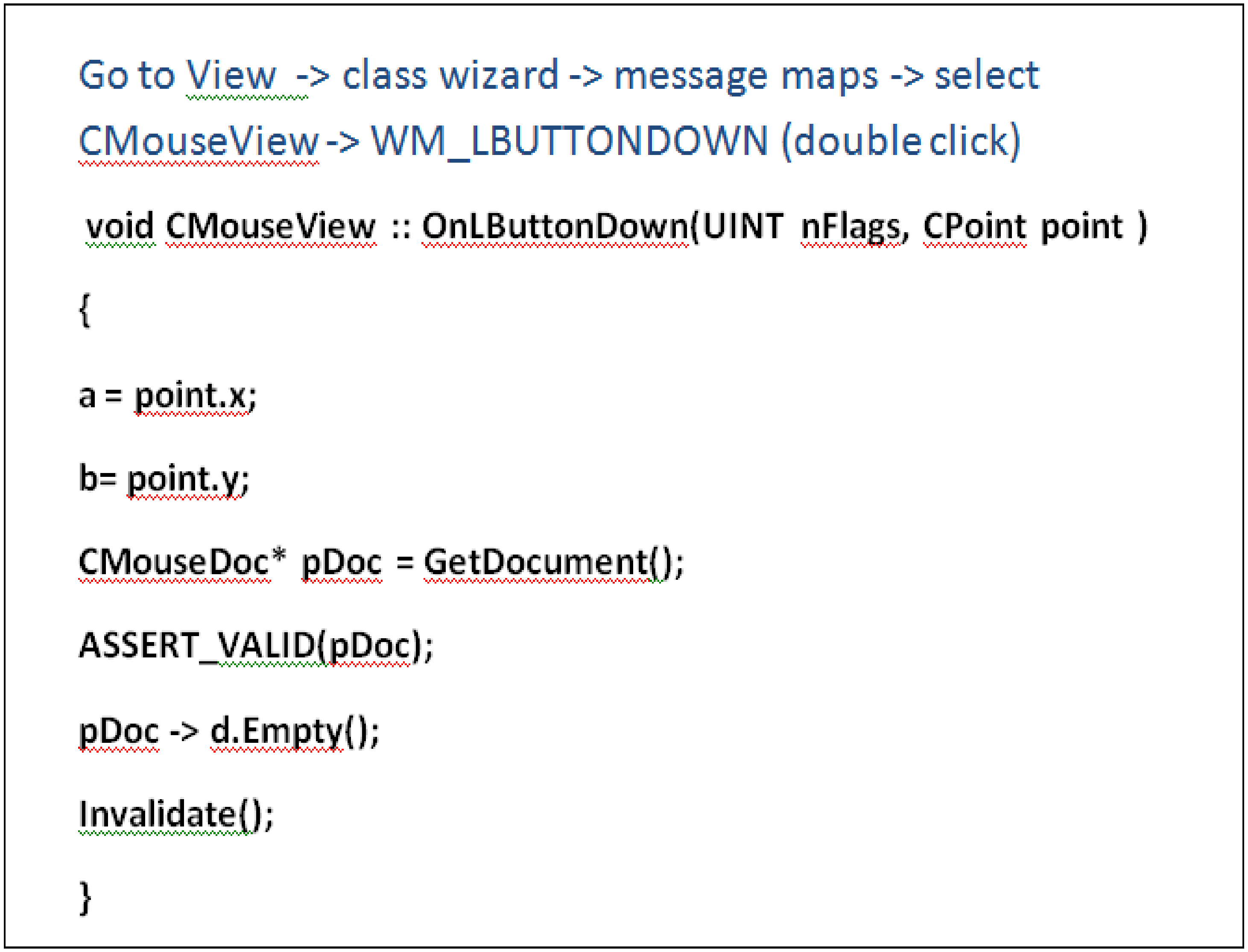


MouseView.h

```
Class CMouseView ::  public CView

{

Protected:

CMouseView();

DECLARE_DYNCREATE(CMouseView)

CPoint CaretPosition;

boolean CaretCreated;

int a,b;

}
```

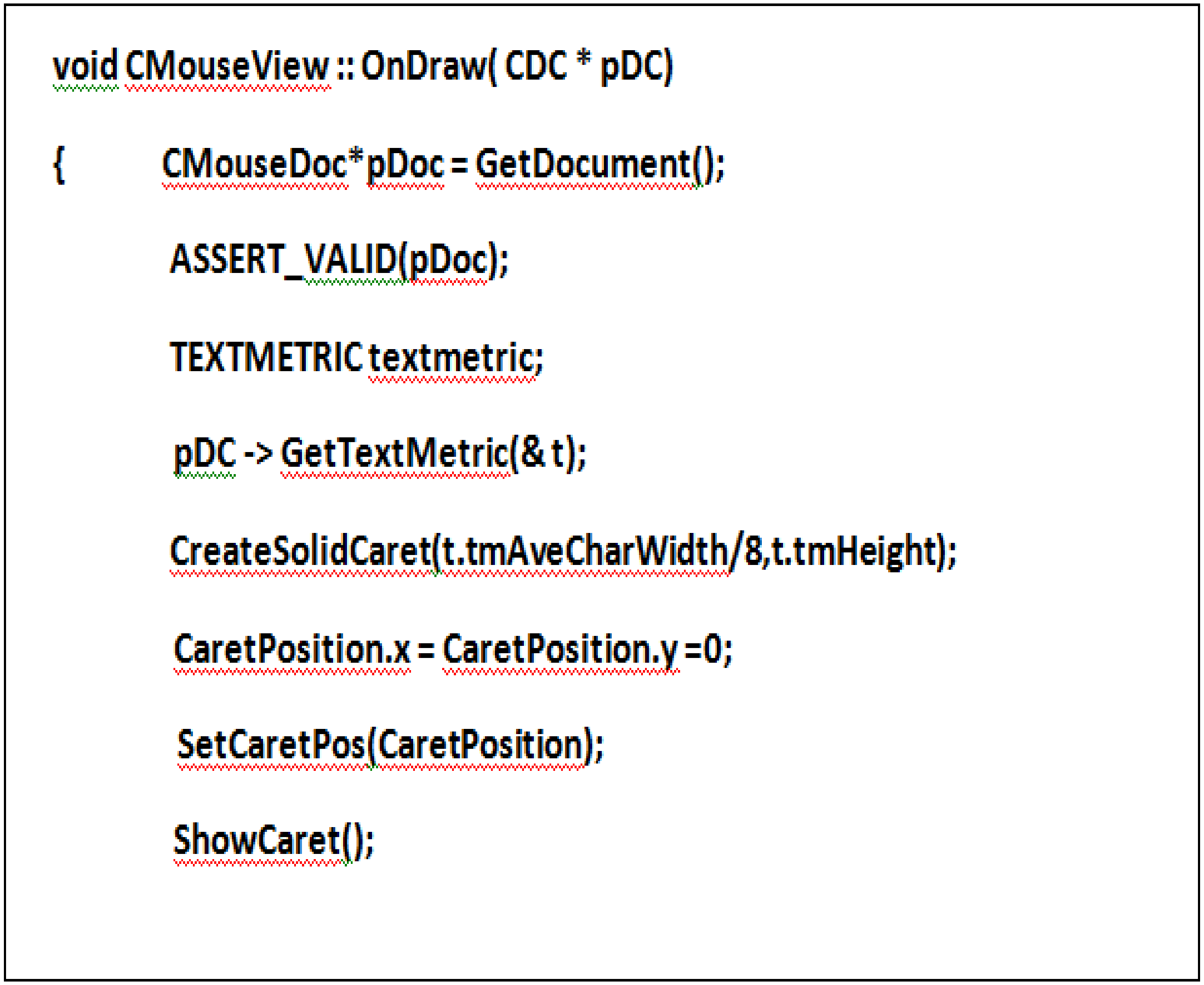
MouseView.cpp

```
Void CMouseView :: OnChar(UINT nChar, UINT nRepCnt ,UINT nFlags)

{

CMouseDoc * pDoc = GetDocument();

ASSERT_VALID(pDoc);

pDoc ->  d+= nChar;

Invalidate();

}
```



Go to View -> class wizard -> message maps -> select CMouseView -> WM_LBUTTONDOWN (double click)

```
 void CMouseView :: OnLButtonDown(UINT nFlags, CPoint point )

{

a = point.x;

b= point.y;

CMouseDoc* pDoc = GetDocument();

ASSERT_VALID(pDoc);

pDoc -> d.Empty();

Invalidate();

}
```



```
void CMouseView :: OnDraw( CDC * pDC)

{        CMouseDoc*pDoc = GetDocument();

          ASSERT_VALID(pDoc);

          TEXTMETRIC textmetric;

          pDC -> GetTextMetric(& t);

          CreateSolidCaret(t.tmAveCharWidth/8,t.tmHeight);

          CaretPosition.x = CaretPosition.y =0;

          SetCaretPos(CaretPosition);

          ShowCaret();
```

```
        CaretCreated = true;

         pDC -> TextOut(x,y,pDoc->d);

        CSize size  = pDC->  GetTextExtent(pDoc->d);

        HideCaret();

        CaretPosition.x = a + size.cx;

        CaretPosition.y = b;

        SetCaretPos(CaretPosition);

        ShowCaret();

    }
```