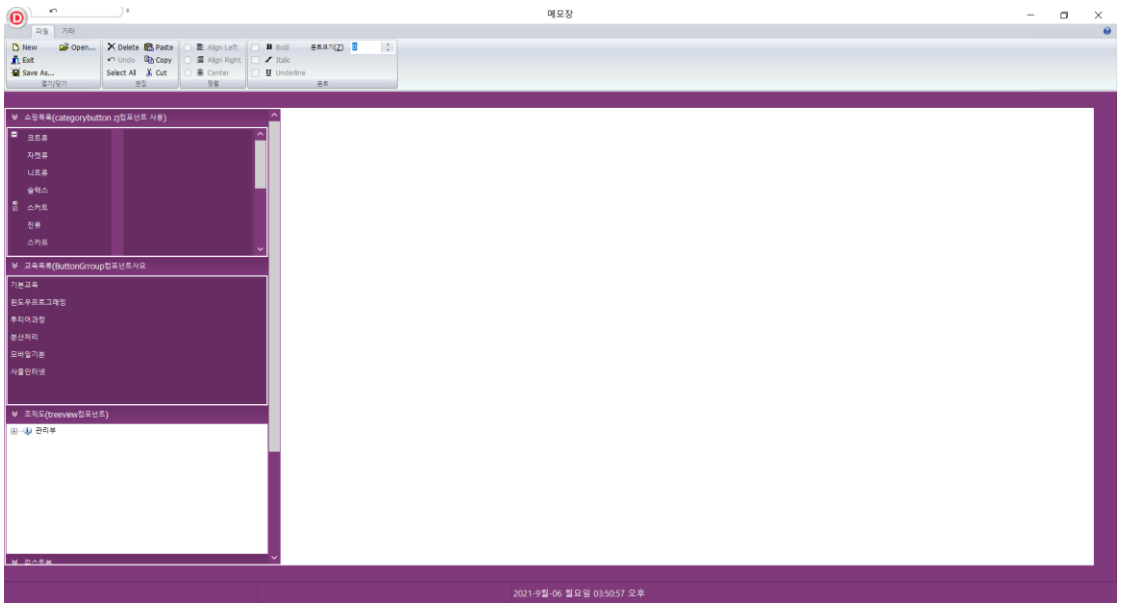
워드패드(메모장) 만들기

파일을 편집(복사, 붙이기, 정렬하기 등등)하여 열기/저장한다.

[완성된 화면]



실습목적

- 메뉴 작성 컴포넌트들을 사용하여 UI를 작성해 본다.
- 파일을 열고 편집하여 저장해 본다.
- 스프래쉬화면, AboutBox 대화상자등 다른 폼들을 표시한다.
- 다양한 컴포넌트들을 사용한다.
- 화면을 띄우기 위해서 필요한 이벤트등(초기처리/마무리처리)등을 구현해본다.

사용 컴포넌트

- **TActionManager**
- **TImageList**
- **Tribbon**
- **TRibbonSpinEdit**

- **TStatusbar**
- **TTimer**
- **TGridPanel**
- **TRichEdit**
- **TGestureManager**
- **TPopupMenu**
- **TCategoryPanelGroup**
- **TButtonGroup**
- **TTreeView**
- **TListView**
- **TTrayIcons**
- **TJumpList**
- **TTaskBar**

[따라하기]

1. **File < New < Vcl Forms Application** 메뉴를 선택하여 새로운 프로젝트를 시작한다.

2. 프로젝트를 **Save Project As**..를 이용하여 저장한다(PMemo.dpr/Umemo.pas).

3. 폼의 속성을 아래와 같이 설정합니다.

속성	속성값
Name	MainForm
WindowState	wsMaximized

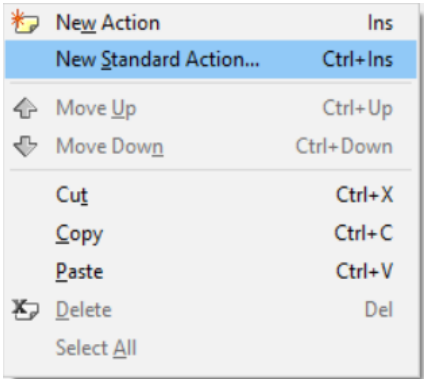
4. 툴 팔레트에서 TImageList를 선택하여 폼 위에 ImageList1 컴포넌트를 올린다.

5. 툴 팔레트에서 TActionManager 컴포넌트를 올려 놓고 Images속성을 ImageList1으로 지정한다.

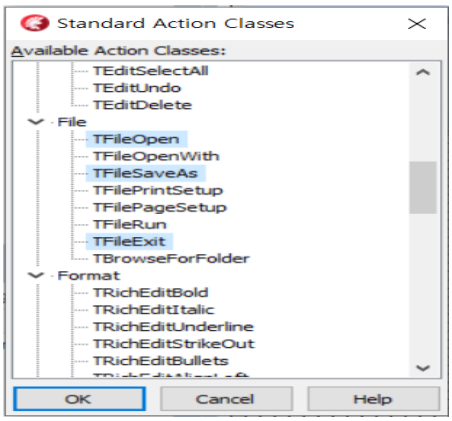
6. TActionManager 컴포넌트를 더블 클릭하거나 오른쪽 마우스 버튼을 클릭하여 팝업메뉴

에서 “Customize”메뉴를 클릭한다.

7. Actions 항목에서 마우스 오른쪽 버튼을 클릭한 후 팝업메뉴에서 ‘New Standard Action’ 메뉴를 클릭합니다.

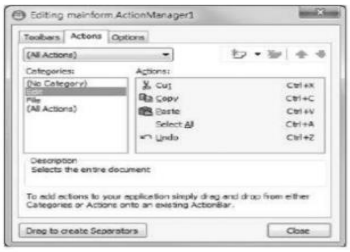


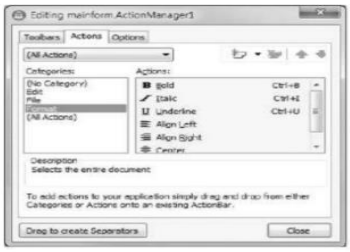
8. 표준 클래스 목록에서 File 항목의 TFileOpen과 TFileSaveAs, TFileExit를 선택하여 추가한다.



아래의 표를 참고하여 메뉴들을 추가합니다.

Action	Category	값
Standard	Edit	TEditCut
		TEditCopy
		TEditPaste
		TEditSelectAll
		TEditUndo
	Format	TRichEditBold
		TRichEditItalic
		TRichEditUnderline
		TRichEditLeft
		TRichEditRight
		TRichEditCenter





9. File을 선택하고, 마우스 오른쪽 버튼을 이용하여 New Action을 선택합니다. 아래와 같이 속성을 변경한다.

컴포넌트	Property	값
Action1	Caption	New
	Category	File
	Hint	새로시작…
	Name	new_Action

10. 임의로 생성한 액션의 경우, 다른 기존 액션들과는 달리 아이콘이 생성되어 있지 않아 ImageList를 이용하여 아이콘을 추가할 수 있습니다.

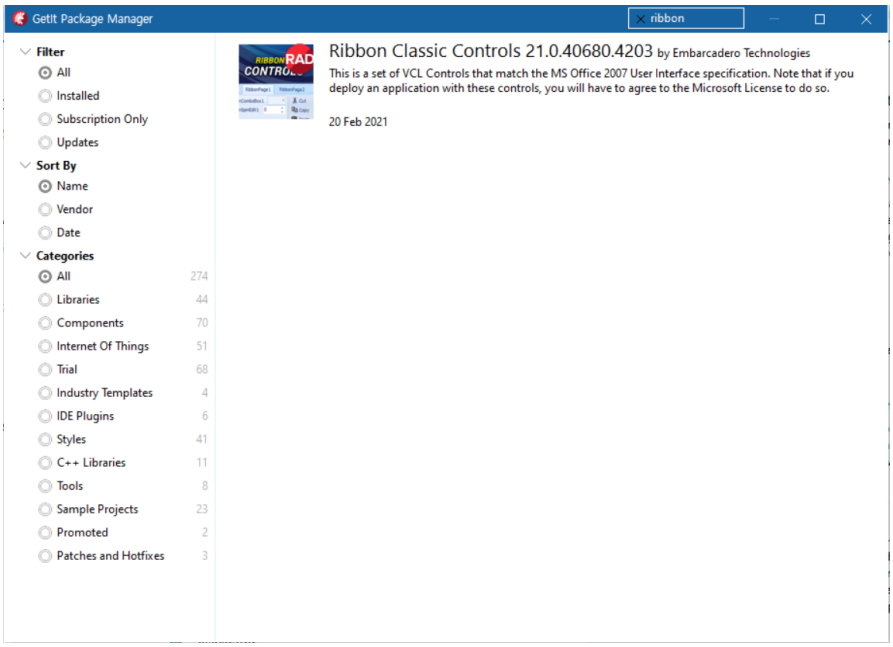
C:₩Program Files₩Common Files₩CodeGear Shared₩Images₩Buttons에서 아이콘 이미지를 확인할 수 있습니다.(새로운 버전에는 이 폴더가 없음) New의 아이콘으로 FILENEW를 선택한다.

New Action으로 아래와 같이 더 추가한다.

Action	속성	값
Action1	Name	Top_Action
	CateGory	기타
	Hint	화면을 맨위로..
	Caption	항상 최상위

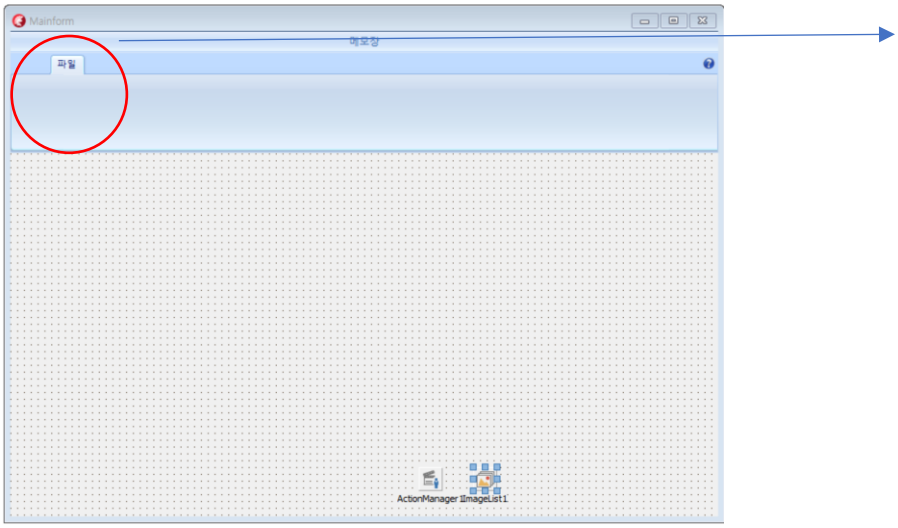
	ImageIndex	적정한 이미지를 추가하고 연결한다.
Action2	Name	About_Action
	CateGory	기타
	Hint	대화상자표시
	Caption	About
	ImageIndex	적정한 이미지를 추가하고 연결한다.

11. 겟잇 패키치 매니저(Getit Package Manager)에서 'Ribbon'을 검색하여 설치한 후 툴 팔레트에 컴포넌트가 올라오는지 확인한다.



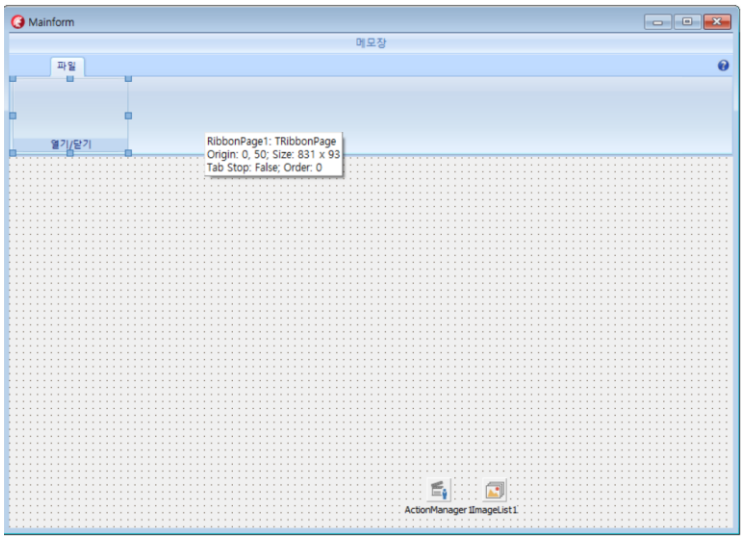
12. 툴 팔레트에서 TRibbon 컴포넌트를 선택하여, 폼 위에 올려 놓고 Caption속성을 '메모장'으로 바꾼다.

13. 리본 컴포넌트 위에서 마우스 오른쪽 버튼을 클릭하여 'Add Tab'을 선택하여 탭을 추가한다.

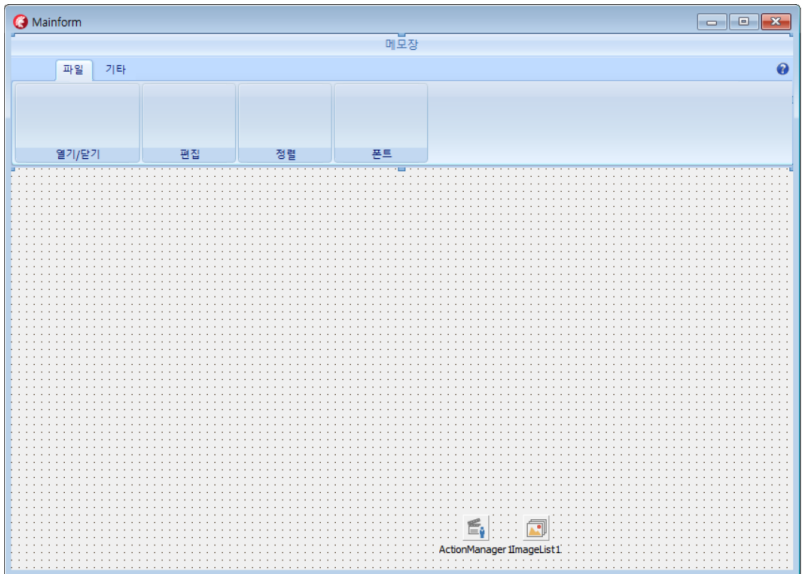


14. 동일한 방법으로 편집 탭에 TRibbonGroup을 한 개 더 생성한다. (Caption 속성을 '기타'로 설정)

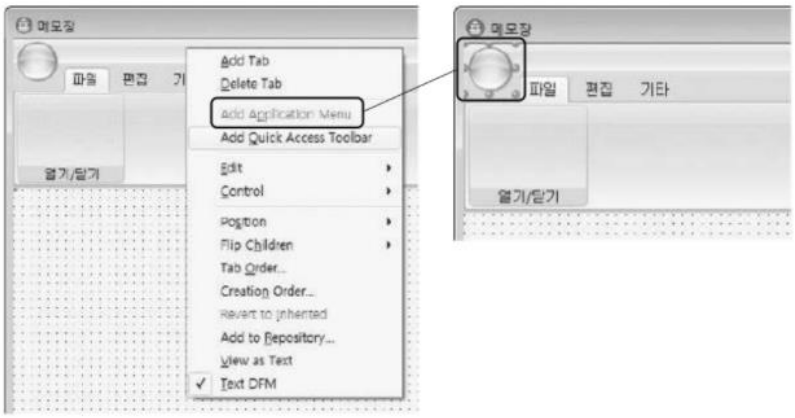
15. 생성한 파일 탭을 선택한 후 마우스 오른쪽 버튼을 눌러 Add Group을 선택하여 그룹을 추가한다. RibbonGroup1 Caption 속성을'열기/닫기'로 지정한다.



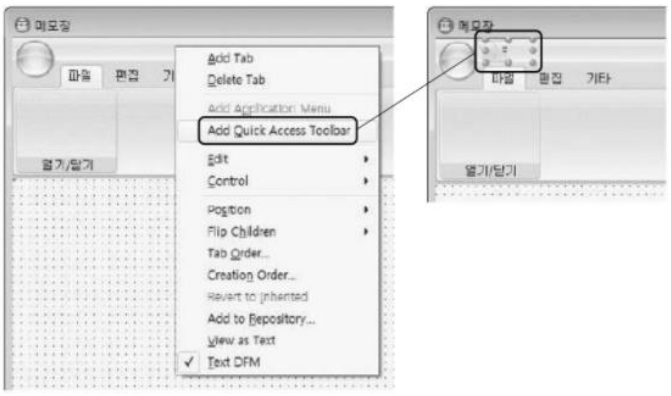
16. 동일한 방법으로 편집 탭에 TRibbonGroup을 아래와 같이 생성한다.



17. Ribbon 위에서 마우스 오른쪽 버튼을 클릭하여 Add Application Menu를 선택한다.

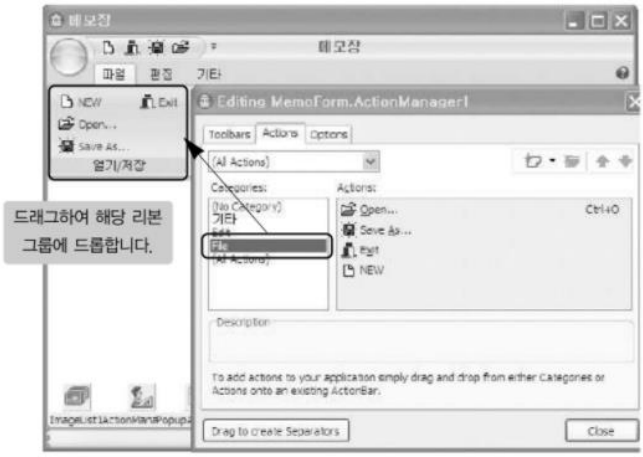


18. Ribbon 위에서 마우스 오른쪽 버튼을 클릭하여 Add Quick Access Toolbar를 추가한다.

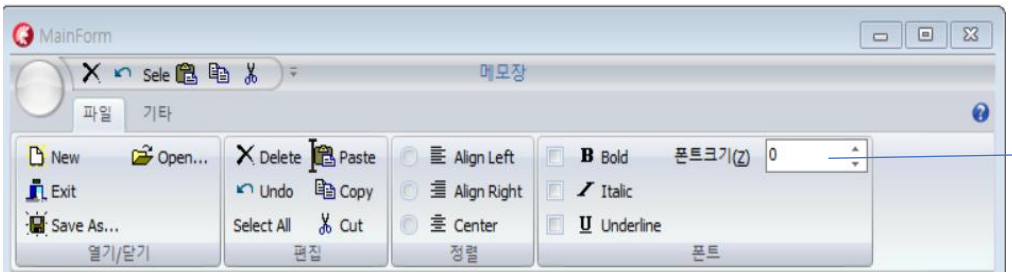


.

19. ActionManager 화면을 열고 카테고리의 Action을 Ribbon 메뉴 위에 끌어 다 놓으면 된다.

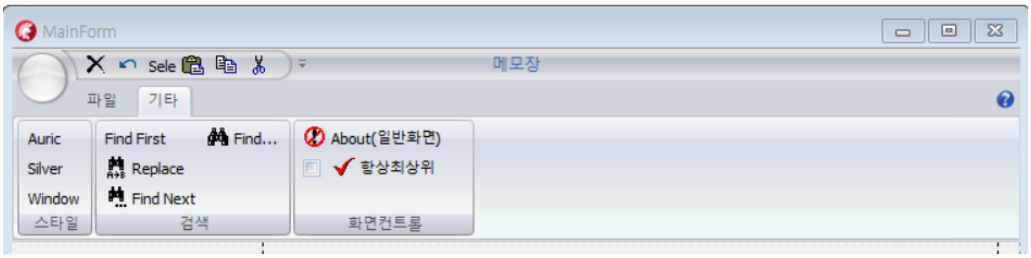


20. 편집 메뉴도 액션들을 드래그 &드롭하여 아래와 같이 배치한다.

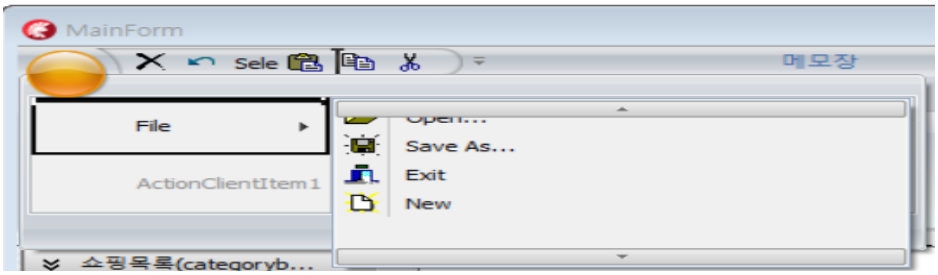


- '정렬'에 있는 액션들의 CommandStlye 속성을 csRadioButton 으로 지정한다.
- 폰트에 있는 액션들의 CommandStlye 속성을 csCheckBox 로 지정한다.
- 폰트에 RibbonSpinEdit 컴포넌트를 추가한다.

21. 기타 메뉴도 액션들을 드래그 &드롭하여 아래와 같이 배치한다.

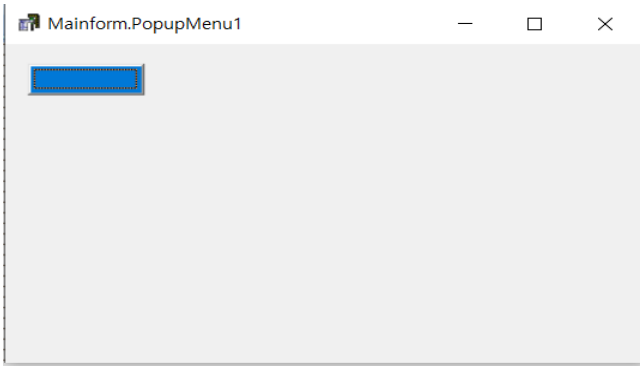


22. ActionManager1에서 File 카테고리를 드래그& 드롭하여 애플리케이션 메뉴에 추가한다.

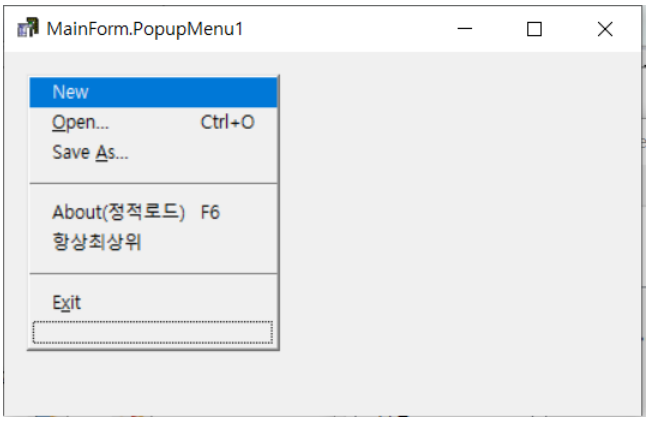


23. ActionManager1에서 원하는 액션들을 드래그&드롭하여 Ribbon Quick AccessToolbar에 추가한다.

24. PopupMenu 컴포넌트를 폼 위의 내려놓고 컴포넌트를 더블-클릭하면, 아래와 같은 메뉴 에디터 창이 표시된다.



25. Action 속성에 원하는 Action 이름을 지정하면 아래와 같이 팝업 메뉴를 디자인한다.



26. 데이터를 입력 할 수 있는 TRichEdit 컴포넌트를 툴 팔레트에서 선택하여, 폼 위에 올려 놓고 속성값을 다음과 같이 설정한다

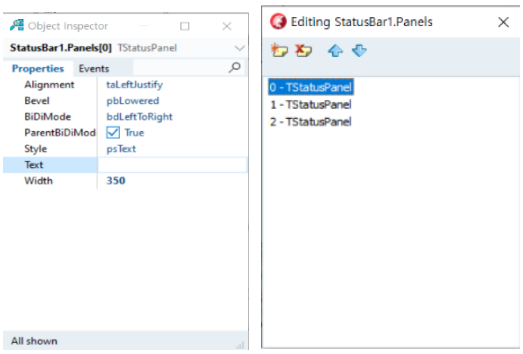
컴포넌트	Property	값
TRichEdit	Align	alClient
	Lines	(TStrings)
	PopupMenu	PopupMenu1
	ScrollBars	ssBoth

.

27. 아래와 같이 현재 시간이나, 메뉴의 힌트값을 표시하기 위하여 TStatusBar 컴포넌트를 폼 위에 올려 놓는다. Align 속성은 기본으로 alBottom으로 설정되어 있다.

28. StatusBar를 나누어 각 부분에 원하는 내용을 표시할 수 있다. 이를 위해 오브젝트 인스펙터에서 Panels 속성을 이용한다

29. Panels 속성 우측의 버튼을 클릭하거나 StatusBar 컴포넌트의 팝업메뉴에서 패널 에디터 창을 표시한다. Add New 버튼 혹은 단축키 Insert를 이용하여 TStatusPanel 값을 추가한다. Add new 아이콘을 눌러, 패널을 3게 생성한다. 각 Panel의 Width 속성을 각각 400, 400 등으로 지정한다.



30. 툴 팔레트를 이용하여 TTimer를 폼 위에 올려놓고 OnTimer 이벤트 핸들러를 작성하여 TStatusBar의 세 번째 패널에 현재시간을 1초 마다 표시하도록 코딩 한다.

```
procedure TMainForm.Timer1Timer(Sender: TObject);
begin
    StatusBar1.Panels[2].Text :=
    FormatDateTime('yyyy-mmmm-dd dddd hh:nn:ss ampm',now);
// FormatFlaot('#,##0.00', i);
// Format('현재수 =%d , 총수 =%d',[i,j])
end;
```

31. 각 액션들의 OnExecute 이벤트 핸들러를 작성하도록 하겠다. Standard 액션의 경우는 이미 코딩이 되어 있다.

32. FileOpen1 액션은 다이어로그 대화상자가 표시되기 전의 BeforeExecute와 '확인'버튼 클릭 시 발생하는 OnAccept 이벤트 핸들러를 작성한다. BeforeExecute 이벤트는 디폴트 디렉토리나 파일 형식들을 지정하기 위해서 작성한다.

```
procedure TMainForm.FileOpen1BeforeExecute(Sender: TObject);
begin
    FileOpen1.Dialog.InitialDir := FilePath;
    FileOpen1.Dialog.Filter :=
    '프로젝트파일|*.dpr|유니트파일|*.pas|텍스트파일|*.txt';
end;

procedure TMainForm.FileOpen1Accept(Sender: TObject);
begin
  RichEdit1.Lines.LoadFromFile(FileOpen1.Dialog.FileName);
  Ribbon1.AddRecentItem(FileOpen1.Dialog.FileName);
end;
```

33. FileSaveAs1 액션은 다이어로그 대화상자가 표시되기전의 BeforeExecute와 '확인'버튼 클릭 시 발생하는 OnAccept 이벤트 핸들러를 작성한다.

```
procedure TMainForm.FileSaveAs1BeforeExecute(Sender: TObject);
begin
  FileSaveAs1.Dialog.InitialDir := FilePath;
end;

procedure TMainForm.FileSaveAs1Accept(Sender: TObject);
```

```
begin
  RichEdit1.Lines.SaveToFile(FileSaveAs1.Dialog.FileName);
end;
```

34. 'New'메뉴에 대한 이벤트 핸들러를 구현한다.

```
procedure TMainForm.New_ActionExecute(Sender: TObject);
begin
   RichEdit1.Clear;
end;
```

35. '항상 최상위'메뉴에 대한 이벤트 핸들러를 구현한다.

```
procedure TMainForm.Top_ActionExecute(Sender: TObject);
begin
   Top_Action.Checked := not Top_Action.Checked;
   if Top_Action.Checked then
      FormStyle := fsStayOnTop
   else
      FormStyle := fsNormal;
end;
```

36. 현재 작업중인 라인수와 컬럼을 표시하기 위해 RichEdit 컴포넌트에 대한 OnChange 이벤트 핸들러를 구현한다.

```
procedure TMainForm.RichEdit1 Change(Sender: TObject);
begin
  statusbar1.Panels[1].Text :=
  Format('현재컬럼:%d 현재라인수:%d', [GetcurrPos(RichEdit1)+1,
  GetCurrLine(Richedit1)+1]);
end;
```

커서가 위치한 곳의 정보를 알려주기 위한 function을 선언한다.

```
Private
function GetCurrPos(RichEdit:TRichEdit):integer;
function GetCurrLine(RichEdit:TRichEdit):integer;
```

function 선언 후 Ctrl + Shift + C을 이용하여 function에 대한 코드를 구현한다.

```
function TMainForm.GetCurrLine(RichEdit: TRichEdit): integer;
begin
  result := RichEdit.Perform(EM_LINEFROMCHAR,richEdit.SelStart,0);
end;

function TMainForm.GetCurrPos(RichEdit: TRichEdit): integer;
begin
  result := RichEdit.SelStart - RichEdit.Perform(EM_LINEINDEX,GetcurrLine(RichEdit),0);
end;
```

37. 메뉴에 마우스가 움직일 때 마다 StatusBar에 힌트 값을 표시하기 위해 다음과 같이 프로시저를 선언하고 구현한다.

```
public
    procedure ShowHint(Sender:Tobject);

Implemetation

procedure TMainForm.ShowHint(Sender: Tobject);
begin
  StatusBar1.Panels[0].Text := Application.Hint;
//Application.hintpause
end;
```

38. 폼의 Create 이벤트 핸들러에서 다음과 같이 구현한다.

```
procedure TMainForm.FormCreate(Sender: TObject);
begin
  Application.onhint := ShowHint;
  RibbonSpinEdit1.Value := RichEdit1.font.Size;
end;
```

39. RibbonSpinEdit의 폰트 크기가 조정되면 RichEdit의 폰트 크기를 조정한다.

```
procedure TMainForm.RibbonSpinEdit1Change(Sender: TObject);
begin
  RichEdit1.Font.Size := RibbonSpinEdit1.value;
end;
```

40. 폼의 OnCloseQuery 이벤트 핸들러에서 메모장의 내용이 있는 경우 프로그램이 종료되지 않도록 코드를 구현한다.

```
procedure TMainForm.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
  if RichEdit1.Lines.Text <> '' then
  begin
    Showmessage('메모장 지우시고 종료하십시오');
    CanClose := false;
  end;
end;
```