


## 공개소프트웨어 강좌

### – Getting Started Guide for Boa Constructor 4 –

작성일 : 2006. 08. 21.

	<b>프로그램명</b>	<b>라이선스</b>
	BOA Constructor	GPL
	<b>리뷰버전</b>	<b>지원언어</b>
	boa-constructor-0.4.4.win32.exe	English
	<b>운영체제</b>	<b>제작(자)사</b>
	MS Windows (95/98/NT/2000/XP), All POSIX(UNIX-like OSes, Linux)	Riaan Booyesen
	<b>권장사항</b>	

**Homepage** <http://boa-constructor.sourceforge.net/>

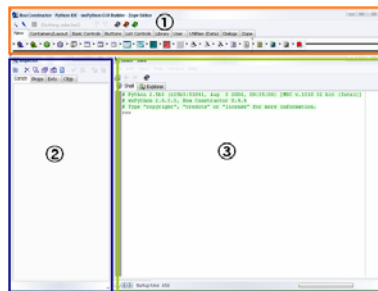
**작성자** 리눅스포털(www.superuser.co.kr) 수퍼유저코리아 서버관리팀

## 소 개

wxPython의 IDE Tool중 공개적으로 사용할 수 있는 Boa Constructor(이하 Boa)를 이용하여 간단한 Application 프로그램 제작 방법을 간략히 알아본다.

※ Boa는 버전 0.2.3 이번버전에서는 아직 다른 상용 IDE Tool에 비하여 완벽한 모습을 보이고 있지는 않지만, wxPython을 이용하여 제작되었고 낮은 버전에 비해 많은 기능이 탑재되어 있어서 앞으로 발전할 가능성이 많아 보인다.

앞 강좌 중 Boa 윈도우 구성을 다시 한번 살펴보겠다.



① **Palette Window** : 여러 Window들을 가져오거나 각종 Helper있는 Window로 첫 번째부터 Inspector, Editor, Class Explore Window를 열 수 있다. Designer에서 사용될 여러 가지 **컴포넌트**를 각 속성별로 분류

② **Inspector Window** : 컴포넌트의 속성변경과 Application에서 Event별로 Event Handler의 설정을 하는 Window

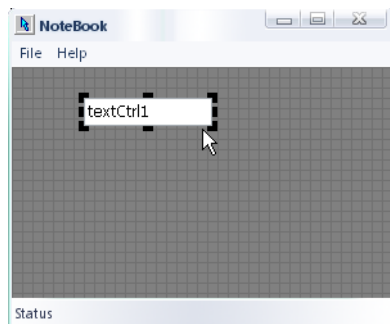
③ **Editor Window** : 애플리케이션의 코드와 프로젝트를 Edit를 할 수 있도록 지원하는 Window으로 Python Shell, File Explore, Editor등을 Tab형태로 지원한다.

## 사용방법

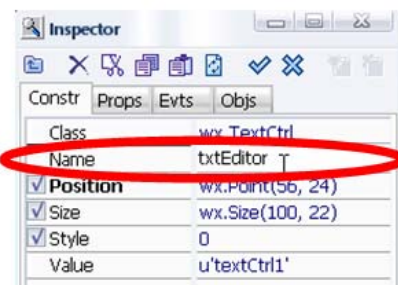
### 3. Text Control 달기

이제 기능적인 부분들을 추가하도록 하겠다.

우선 Text Control을 달아보는데 Frame Designer  를 Click후 Palette window의 Basic Controls tab에서 2번째 아이콘인 wx.TextCtrl을 선택 후 Frame Designer에 Click하면 그림처럼 보인다. textCtrl1의 Inspector window의 Name을 txtEditor로 수정 후 별도의 Size조절 없이 저장 후 바로 실행한다. 실행시 Auto Size로 처리된다.

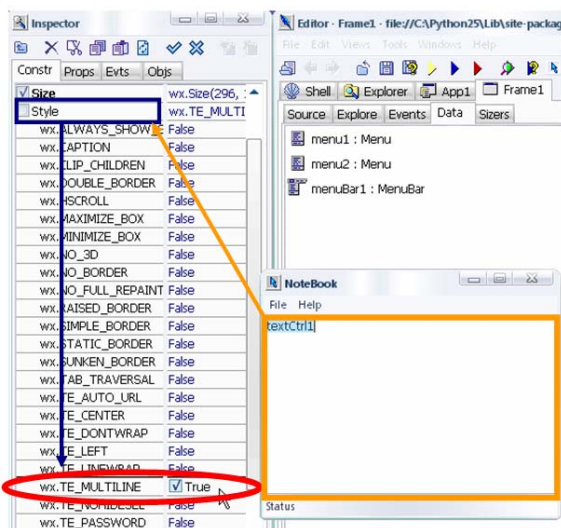


[Text Control 생성화면]

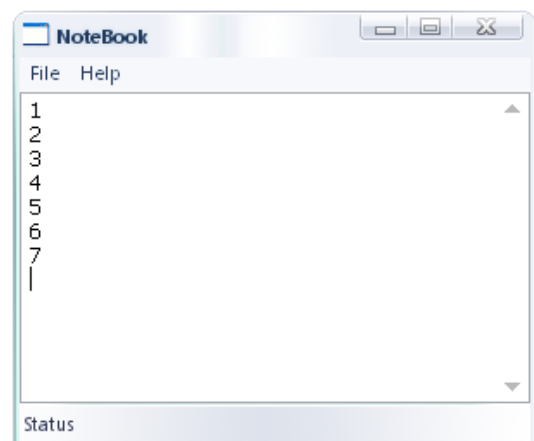


[Text Control의 Name 수정]

실행해보면 MultiLine 입력이 되지 않는데 관련된 속성을 수정하기 위해서 textCtrl1의 Inspector window의 Style 속성값을 수정한다. 수정방법은 Style의 체크박스를 클릭하여 wxTE\_MULTILINE의 값을 True로 바꾸면 된다. 저장 후 실행하면 MultiLine으로 입력할 수 있다.



[Style 속성수정]



[Text Control 실행화면]

## 사용방법

### 4. Event Handler 처리

이제 DragMenu별로 발생하는 Event를 처리하는 Handler를 각각 Coding한다.

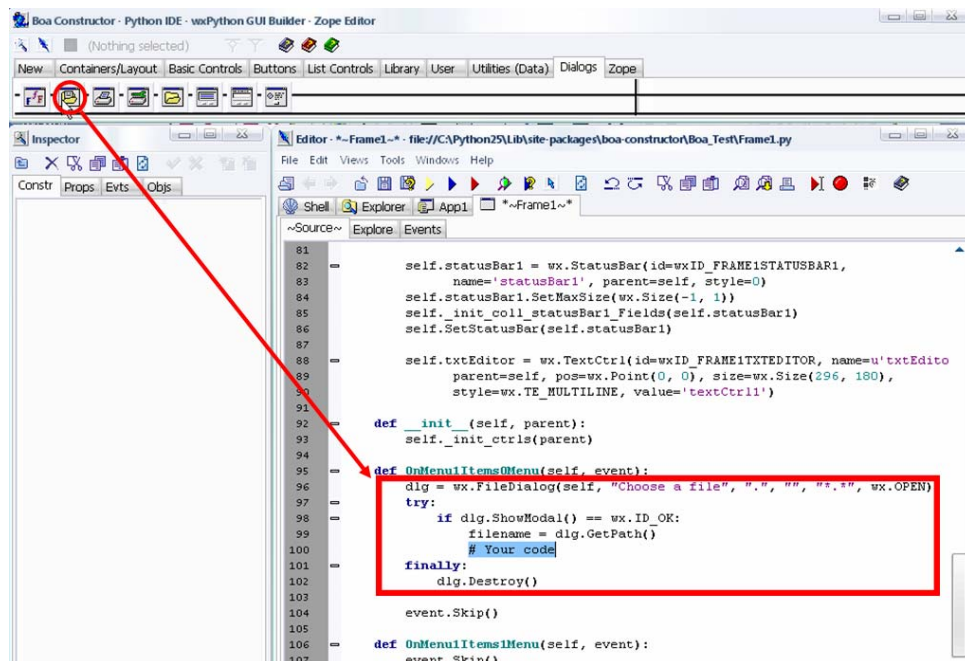
Frame1의 Editor Window에서 자동으로 생성되는데 Event Handler는 각각의 메뉴 특성에 맞게 Coding한다. 먼저 Open시 사용되는 File Dialog는 별도의 Coding없이 Palette Window에서 가져온다.

```
def OnMenuItems0Menu(self, event):
```

```
    #마우스 커서를 이곳으로 옮긴다.
```

```
    event.skip()
```

Palette window의 Dialog Tab에서 3번째 아이콘인 wx.FileDialog를 Click하면 아래에 반전으로 표시된 코드가 자동 생성된다.

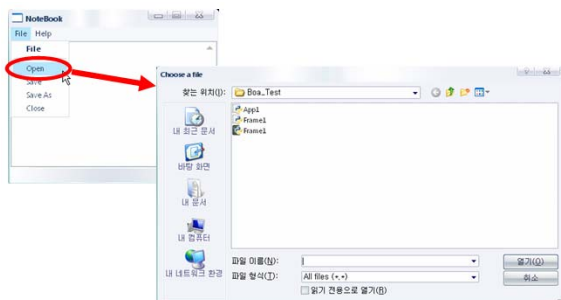


# Your code 부분을 다음과 같이 수정 추가한다.

```
self.txtEditor.LoadFile(filename)
```

```
self.FileName = filename
```

이렇게 입력하고 저장후 정상 동작되는지 실행해본다.



## 사용방법

Event Handler의 전체 Source Code는 아래와 같은 예제로 작성한다.

```
# Open
def OnMenuItems0Menu(self, event):
    dlg = wxDirDialog(self)
    try:
        if dlg.ShowModal() == wxID_OK:
            dir = dlg.GetPath()
            self.txtEditor.LoadFile(filename)
            self.FileName = filename
    finally:
        dlg.Destroy()

# Save
def OnMenuItems1Menu(self, event):
    if self.FileName == None:
        return self.OnFileitem2Menu(event)
    else :
        self.txtEditor.SaveFile(self.FileName)

# Save As : wxFileDialog 사용
def OnMenuItems2Menu(self, event):
    dlg = wxFileDialog(self, "Save File As", ".", "", " *.*", wxSAVE)
    try:
        if dlg.ShowModal() == wxID_OK:
            filename = dlg.GetPath()
            self.txtEditor.SaveFile(filename)
            self.FileName = filename
    finally:
        dlg.Destroy()

# Close
def OnMenuItems4Menu(self, event):
    self.Close()

# Help-About Menu에 사용되어질 Event Handler
def OnMenu2items0Menu(self, event):
    event.Skip()
```

다음 강좌는 Dialog window 생성 방법을 배워보도록 하겠습니다.