임베디드 데이터베이스 활용 앱 만들기

임베디드(embedded) 데이터베이스는 기기에 내장되는 데이터베이스로써 PC용 프로그램, 네비게이션, 전자사전 등 다양한 기기에 널리 사용되고 있으며 제한된 메모리와 저장 공간을 가진 모바일 디바이스에도 널리 사용되고 있습니다.

이 장에서는 임베디드 데이터베이스를 이용해 앱을 만들고, 모바일 앱 배포 시 임베디드 데이터베이스도 함께 배포하는 방법을 알아보겠습니다.

1. 임베디드 데이터베이스 사용

PC용 응용프로그램은 물론 모바일 앱에 있어서 가장 기본적이고도 빈번하게 사용하는 기능은 원하는 정보를 입력하고 저장하고 제공하는 데이터와 관련된 기능일 것입니다. 그리고 이런 기능을 구현하는 가장 보편적인 방법은 DBMS를 활용하는 것이며 모바일 앱은 디바이스의 특성 상 외부의 DBMS를 연동하기 보다는 기기 내에 탑재 (Embedded) 되는 DBMS를 주로 사용합니다. 물론 외부의 DBMS에 연동하거나 웹서비스를 이용하는 방법도 자주 사용되며, 이 부분은 다른 교육에서 학습하게 됩니다. 현재 모바일 시장에서 주로 사용하는 대표적인 임베디드 데이터베이스는 Interbase ToGo와 SQLite가 있으며, 두 가지 모두 델파이 모바일 앱에서 사용할 수 있습니다. Interbase ToGo의 경 우 델파이 설치 시 옵션(기본으로 선택되어 있음)으로 같이 설치할 수 있으며, 데이터베이스 엔진과 개발환경이 모두 제공되므로 별도로 개발환경을 구축해야 할 필요가 없습니다. 이번에는 모바일에 내장되어 있는 SQLite를 사용하도록 하겠습니다. (인터베이스도 데이터베이스를 만들고 연결하는 과정만 차이가 있을 뿐 개발 방식은 동일합니다.)

IBLite 소개

InterBase ToGo는 엠바카데로 테크놀러지의 인터베이스(InterBase) 데이터베이스 에디션 중 하나입니다. InTerBase ToGo는 고성능, 멀티플랫폼(Windows, Mac OS X, iOS, Android) 지원 등 강력한 기능을 제공하면서도 DBA가 필요 없을 만큼 높은 신뢰성과 성능을 보장합니다. (http://www.devgear.co.kr/products/interbase)
IBLite와 InterBase ToGo는 모바일에서 사용 할 수 있는 임베디드 형 데이터베이스로 암호화 등 몇 가지 기능에 있어서 차이가 있습니다. 차이는 아래의 표와 같습니다.(제품의 스펙은 변경될 수 있습니다)

기능	IBLite	InterBase ToGo
가격 정책	무료	유료
CPU 코어 사용 권한	1	4
허용하는 동시 사용자 수	1	2
사용자 당 연결 수	1	4
데이터베이스 열 암호화(AES)	X	O
네트워크 암호화	X	O
데이터베이스 파일크기 제한	100 MB	제한 없음
연결 당 트랜잭션 제한	1	제한 없음
서비스 API 사용	X	O
OTW/SSL 지원	X	O

IBLite 와 SQLite

모바일 임베디드 DBMS 는 다양한 제품들이 출시되고 있습니다만 주로 사용되는 제품으로는 SQLite, IBLite 등이 있습니다. SQLite 는 현재 모바일 앱에서 가장 많이 사용되는 임베디드 DB 이지만 오픈소스로서 품질이나 기술 지원 등을 보장할 수는 없는 반면에, IBLite 는 엠바카데로의 Inetrbase DBM 의 에디션 중 하나로써 강력한 기능은 기본이며 엠바카데로라는 신뢰할 수 있는 회사가 제품의 품질과 지원을 보장되기 때문에 기업체나 공공기관에서 상당히 선호하는 제품입니다.

IBLite 라이선스 파일 등록하기 IBLite는 배포 라이선스가 무료(정품인 경우) 데이터베이스입니다. 배포 라이선스는 무료지만 최초 1회 라이선스를 등록하고 라이선스 파일을 다운로드하는 과정이 필요합니다.

1. 배포 라이선스는 데브기어 테크사이트의 InterBase 설치 방법 1단계 ~ 3단계"(http://tech.devgear. co.kr/344937)을 참고해 다운로드 받습니다.

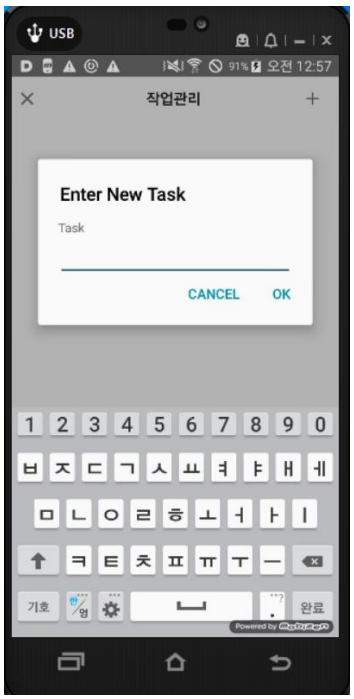
2. reg_nnnnnnnn.txt 형태로 다운로드 받은 파일을 reg_iblite.txt로 이름을 변경합니다.

3. reg_iblite.txt 파일을 "C:₩UserswPublic₩Documents₩EmbarcaderowinterBase₩redistv

₩InterBaseXE3"에 복사합니다.

따라하기

작업 목록을 표시하고, 새로운 작업을 등록하고, 완료 항목은 옆으로 밀어 삭제하는 기능을 구현해 보자.



1. . File > New > Multi-Device Application - Delphi(XE6 이하 버전은 FireMonkey Mobile Application - Delphi) > Blank Application 을 클릭합니다

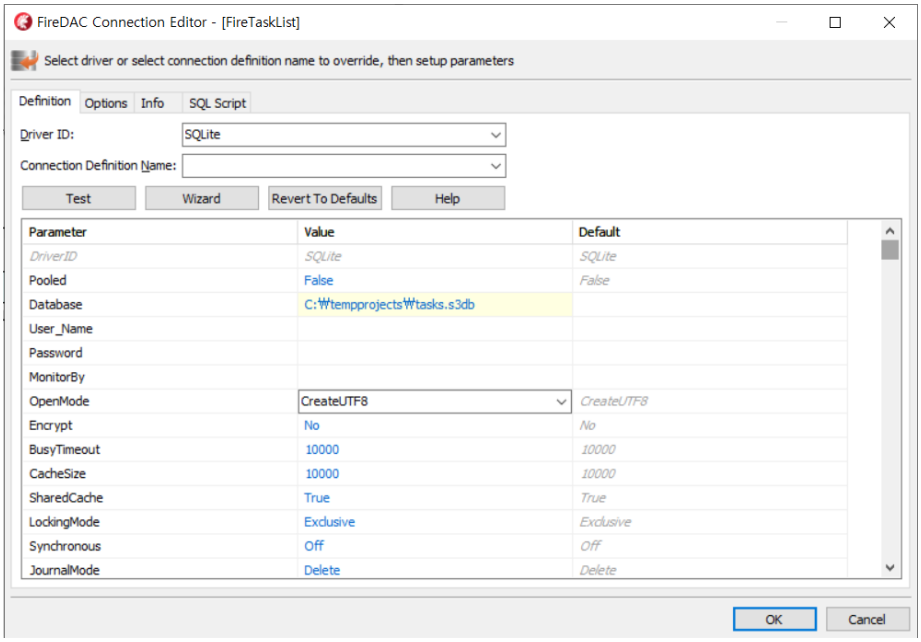
2. 컴포넌트를 추가하고 아래와 같이 UI 를 작성한다.

상위 컴포넌트	컴포넌트	속성	값(또는 설명)
Form1	ToolBar1		
ToolBar1	Label1	Align	Contents
		Text	작업관리
		TextSettings.HorzAlign	Center
	Button1	Name	btnAdd
		Align	Right
		StyleLookup	AddToolButton
	Button2	Name	btnDelete
		Align	Left
		StyleLookup	deletetoolbutton

Form1	ListBox1	Align	Client
		DefaultItemStyles.ItemStyle	listboxitemnodetail
	FDConnection1	데이터베이스 연결은 뒤에서 설명	
	FDGUIxWaitCursor1		UI에서 DB사용시 커서 표시
	FDPhysSQLiteDriverLink1		해당DB에따라(새 버전에서는 반드시 필요하지는 않음
	FDTable1	Name	FDTableTask
		Connection	FDConnection1
		TableName	Task
	FDQuery1	Name	FDQueryInsert
	FDQuery2	Name	FDQueryDelete

3. FDConnection 의 속성을 설정해 데이터베이스와 연결합니다.

- 연결 설정을 위해 FDConnection1 을 더블 클릭하고 아래와 같이 속성 변경 후 OK 버튼을 클릭한다.



속성	값
Driver ID	SQLite
Database	프로그램에서 코드로 연결
OpenMode	CreateUTF8
LoginPrompt	false

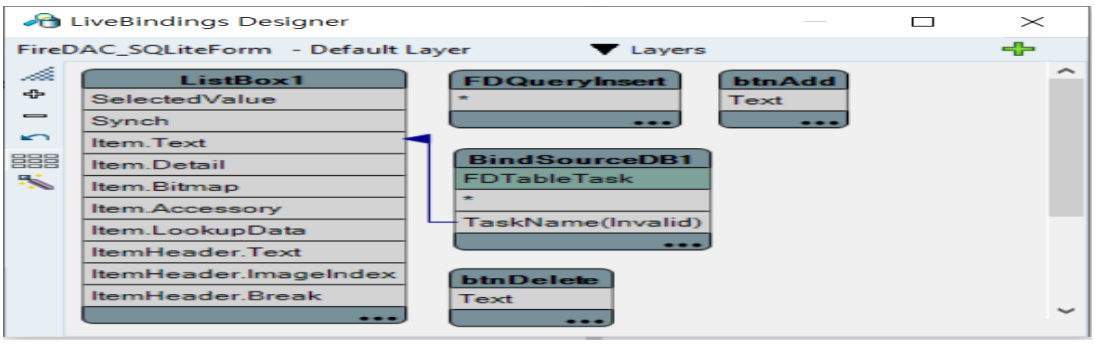
4. QueryInsert 를 새로운 작업추가 쿼리를 수행하도록 SQL 속성에 다음과 같이 코드를 입력한다.

```
insert into task (TaskName) values (:TaskName)
```

5. FDQueryDelete 를 작업 삭제 쿼리 수행하도록 SQL 속성에 다음과 같은 코드를 입력한다.(컴포넌트를 더블 클릭하여 SQL Command 에 입력가능)

```
delete from task where TaskName = (:TaskName)
```

6. 팝업메뉴에서 Bind Visually..를 선택하여 라이브바인딩 디자이너를 열고 아래와 같이 FDTablel 의 TASKNAME 항목을 마우스 드래그 하여 ListVeiw1 의 Item.Text 로 연결한다.



7. 아이템 추가 버튼(Button1)을 더블 클릭하여 클릭 이벤트 핸들러를 아래와 같이 구현한다.

```
procedure TForm1.btnAddClick(Sender: TObject);
var
    TaskName: string;
begin
  try
      TDialogService.InputQuery('새로운작업입력', ['작업관리'], [''],
      procedure(const AResult: TModalResult; const AValues: array of string)
      begin
        if AResult = mrOk then
          TaskName := AValues[0]
        else
          TaskName := '';
        if not (TaskName.Trim = '') then
        begin
          FDQueryInsert.ParamByName('TaskName').AsString := TaskName;
          FDQueryInsert.ExecSQL();
          FDTableTask.Refresh;
          LinkFillControlToField1.BindList.FillList;
        end;
      end);
  except
    on e: Exception do
    begin
      ShowMessage(e.Message);
    end;
  end;
end;
```

8. 작업항목 삭제 기능을 구현하기 위해 btnDelete 버튼의 클릭 이벤트 핸들러를 다음과 같이 구현한다.

```
procedure TForm1.btnDeleteClick(Sender: TObject);
var
  TaskName: string;
  LIndex: Integer;
begin
  TaskName := ListBox1.Selected.Text;
  try
    FDQueryDelete.ParamByName('TaskName').AsString := TaskName;
    FDQueryDelete.ExecSQL();
    FDTableTask.Refresh;
    LinkFillControlToField1.BindList.FillList;
    if (ListBox1.Selected = nil) and (ListBox1.Count > 0) then
      // Select last item
      ListBox1.ItemIndex := ListBox1.Count - 1;
  except
    on e: Exception do
    begin
      SHowMessage(e.Message);
    end;
  end;
end;
```

9. 폼의 Create 이벤트 핸들러를 생성하고 아래와 같이 구현한다.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  try
  // For unidirectional dataset, don't refill automatically when dataset is activated
  // because dataset is reactivated everytime use DataSet.First.
        LinkFillControlToField1.AutoActivate := False;
        LinkFillControlToField1.AutoFill := False;
        FDConnection1.Connected := True;
        FDTableTask.Active := True;
        LinkFillControlToField1.BindList.FillList;
  except
      on e: Exception do
```

```
        begin
          SHowMessage(e.Message);
        end;
    end;
end;
```

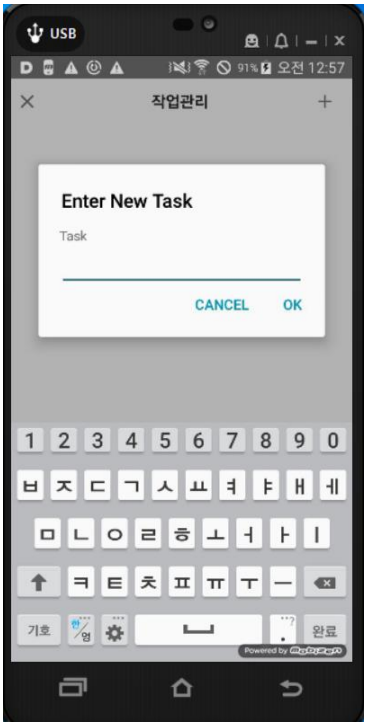
10. FDConnection1 의 AfterConnect 이벤트 핸들러를 생성하고 Task 테이블을 생성하는 코드를 구현한다.

```
procedure TForm1.TaskListAfterConnect(Sender: TObject);
begin
  FDConnection1.ExecSQL ('CREATE TABLE IF NOT EXISTS Task (TaskName TEXT NOT NULL)');
end;
```

11. 현재 데이터베이스 파일 경로는 윈도우를 기준으로 설정되어 있습니다. 이 경로를 모바일에 맞게 변경하기 위해 화면으로 돌아와 FDConnection1 의 BeforeConnect 이벤트 핸들러를 생성하고 아래의 코드를 입력합니다. (TPath 사용을 위해 SystemIOUtils 을 구현부 implementation uses 절에 추가합니다.)

```
procedure TForm1.TaskListBeforeConnect(Sender: TObject);
begin
{$IF DEFINED(IOS) or DEFINED(ANDROID)}
  FDConnection1.Params.Values['Database'] := TPath.GetDocumentsPath+PathDelim +
'tasks.s3db';
{$ELSE}
  FDConnection1.Params.Values['Database'] := 'tasks.s3db';
{$ENDIF}
End;
```

12. 프로그램을 실행하여 작업을 입력하고 삭제해본다.



팁 : 배포(데이터베이스와 IBLite 라이브러리)와 배포 경로 설정

모바일 앱에서 데이터베이스 사용을 하려면 모바일 디바이스에 데이터베이스 파일과 IBLite 라이브러리, 라이선스를 배포해야 합니다. 파일 배포를 위해 Project > Deployment 메뉴를 통해 Deployment 창을 표시한다.

- Add files(F) 버튼을 누르고, 앞에서 생성한 데이터베이스 파일인 TAKS.GDB 를 선택하여 추가하고 플랫폼 별 Remote Path 를 아래와 같이 지정한다.
  - Ios 플랫폼 : 'StartUp₩Documents'
  - 안드로이드 플랫폼: 'Assets₩Internal'

- IBLite 관련 기능파일(라이브러리, 라이선스)을 추가하기 위해 **Add Featured Files** 버튼을 클릭하고 Featured Files 창을 표시한다.

- Featured Files **창에** InterBase 중 iOS Device, Android 두 개의 항목에 대해 reg_ibtogo.txt 를 제외하고 나머지 모두를 선택한 후 OK 버튼을 클릭하여 목록에 추가한다..