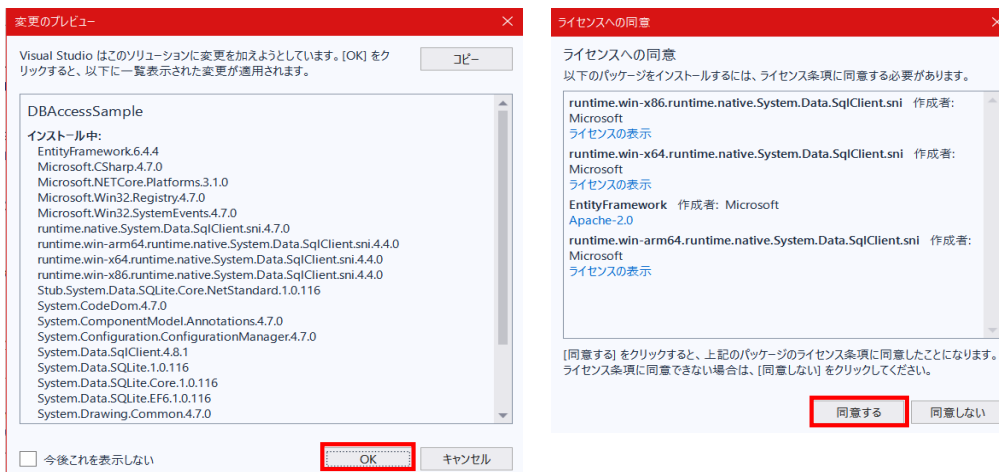
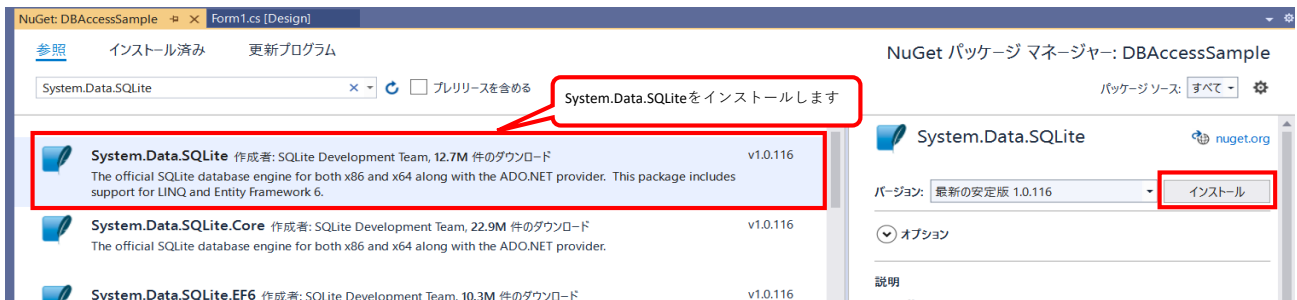
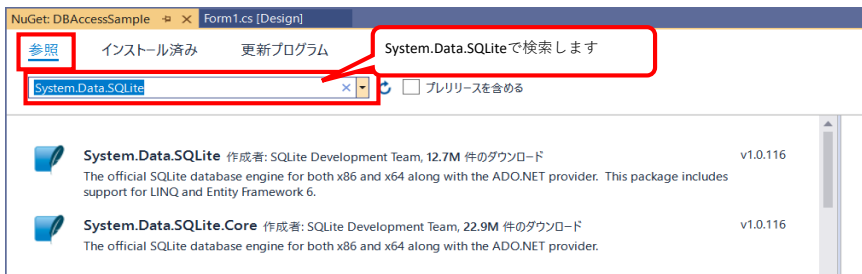
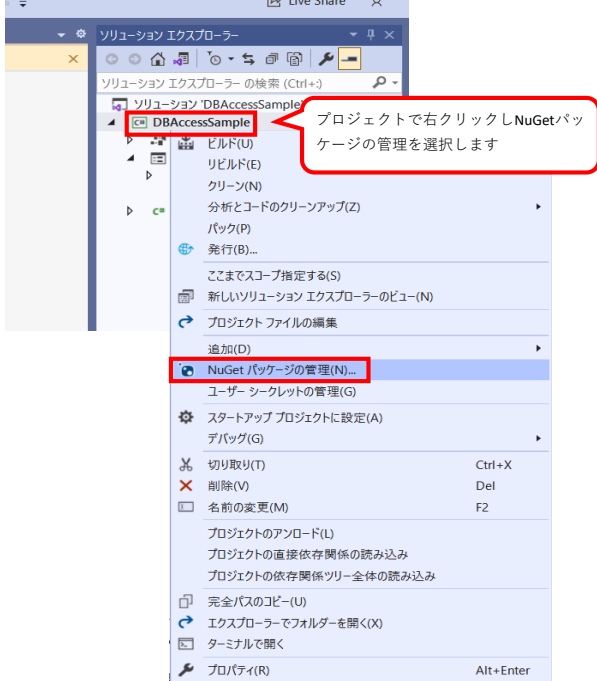
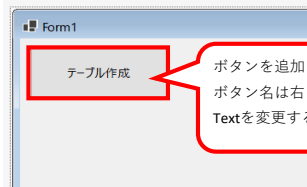


SQLiteのインストール方法



Form1に部品を設置してDBにアクセスします

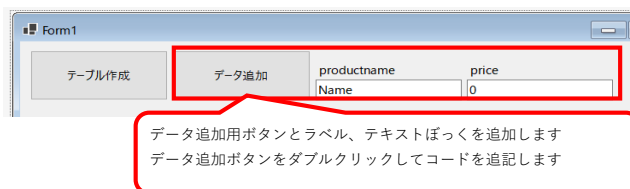
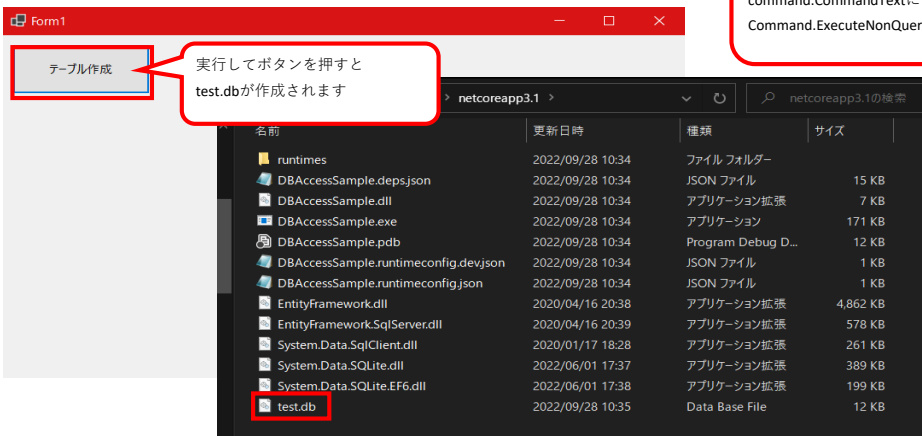


```
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10 using System.Data.SQLite;
11
12 namespace DBAccessSample {
13     3 個の参照
14     public partial class Form1 : Form {
15         1 個の参照
16         public Form1() {
17             InitializeComponent();
18         }
19
20         1 個の参照
21         private void button1_Click(object sender, EventArgs e) {
22             // コネクションを開いてテーブル作成して閉じる
23             using (var con = new SQLiteConnection("Data Source=test.db")) {
24                 con.Open();
25                 using (SQLiteCommand command = con.CreateCommand()) {
26                     command.CommandText =
27                         "create table t_product(CD INTEGER PRIMARY KEY AUTOINCREMENT, productname TEXT, price INTEGER)";
28                     command.ExecuteNonQuery();
29                 }
30                 con.Close();
31             }
32         }
33     }
34 }
```

Data Source=で指定した名前がDBのファイル名になります

t_productテーブルを作成します

con.Open();でDBを接続しcon.Close();でDB接続を切断します
使い終わったら必ず接続を切断しましょう
command.CommandTextには実行するSQL文を入れます
Command.ExecuteNonQuery();でSQLを実行します



```
1 個の参照
private void button2_Click(object sender, EventArgs e) {
    using (SQLiteConnection con = new SQLiteConnection("Data Source=test.db")) {
        con.Open();
        using (SQLiteTransaction trans = con.BeginTransaction()) {
            SQLiteCommand cmd = con.CreateCommand();
            // インサート
            cmd.CommandText = "INSERT INTO t_product (productname, price) VALUES (@Product, @Price)";
            // パラメータセット
            cmd.Parameters.Add("Product", System.Data.DbType.String);
            cmd.Parameters.Add("Price", System.Data.DbType.Int64);
            // データ追加
            cmd.Parameters["Product"].Value = textBox1.Text;
            cmd.Parameters["Price"].Value = int.Parse(textBox2.Text);
            cmd.ExecuteNonQuery();
            // コミット
            trans.Commit();
        }
    }
}
```

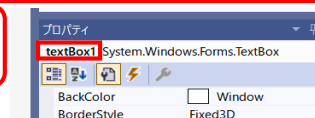
@をつけることで後述のParametersでセットした値をプログラム実行時に付加してSQLを実行できます

ParametersにProductとPriceを追加します

textBox1に入力されている文字列をParametersに設定します

textBox1とtextBox2は追加したテキストボックスの名称です
デザイナー画面のプロパティから確認できます

trans.commit();でDBの変更を確定します



Form1

テーブル作成

データ追加

productname price

test 100

テキストボックスにデータを入力してデータ追加ボタンを押すとDBにデータが登録されます
※データがあるかの確認は次にやります

Form1

テーブル作成

データ追加

productname price

Name 0

データ読み込み

ボタンとデータグリッドビューを追加します
データグリッドが見つからばい場合はツールボックスの検索欄に「data」と入力すると見つかります

ツールボックス

data

All Windows Forms

DataGridview

1 個の参照

```
private void button3_Click(object sender, EventArgs e) {
    using (SQLiteConnection con = new SQLiteConnection("Data Source=test.db")) {
        // DataTableを生成します。
        var dataTable = new DataTable();
        // SQLの実行
        var adapter = new SQLiteDataAdapter("SELECT * FROM t_product", con);
        adapter.Fill(dataTable);
        dataGridview1.DataSource = dataTable;
    }
}
```

SQLiteDataAdapterでSelect文を発行し、結果をdataTableに取得します
その後データグリッドビューに設定します
<https://learn.microsoft.com/ja-jp/dotnet/api/system.data.sqlclient.sqldataadapter?view=dotnet-plat-ext-6.0>

Form1

テーブル作成

データ追加

productname price

Name 0

データ読み込み

CD	productname
1	test

productname	price
test	100

データ読み込みボタンを押すと取得したデータをグリッドビューに表示します
※CDは自動で発行される通番です

```
private void button4_Click(object sender, EventArgs e) {
    using (SQLiteConnection con = new SQLiteConnection("Data Source=test.db")) {
        con.Open();
        using (SQLiteTransaction trans = con.BeginTransaction()) {
            SQLiteCommand cmd = con.CreateCommand();
            // インサート
            cmd.CommandText = "UPDATE FROM t_product set productname = @Product, price = @Price WHERE CD = @Cd";
            // パラメータセット
            cmd.Parameters.Add("Product", System.Data.DbType.String);
            cmd.Parameters.Add("Price", System.Data.DbType.Int64);
            cmd.Parameters.Add("Cd", System.Data.DbType.Int64);
            // データ追加
            cmd.Parameters["Product"].Value = textBox3.Text;
            cmd.Parameters["Price"].Value = int.Parse(textBox4.Text);
            cmd.Parameters["Cd"].Value = int.Parse(textBox5.Text);
            cmd.ExecuteNonQuery();
            // コミット
            trans.Commit();
        }
    }
}
```

UPDATE文を作ることができます

The screenshot shows a Windows Form titled 'Form1'. It has several buttons: 'テーブル作成' (Table Creation), 'データ追加' (Data Addition), 'データ読み込み' (Data Loading), 'データ修正' (Data Modification), and 'データ削除' (Data Deletion). The 'データ読み込み' button is highlighted with a blue border. Below the buttons, there are several data entry and display sections. One section shows a table with columns 'CD' and 'productname', containing a single row with '1' and 'test2'. Another section shows a table with columns 'productname' and 'price', containing a single row with 'test2' and '9999'. There are also input fields for '検索CD' (Search CD) and 'productname' and 'price' fields.

```
private void button5_Click(object sender, EventArgs e) {
    using (SQLiteConnection con = new SQLiteConnection("Data Source=test.db")) {
        con.Open();
        using (SQLiteTransaction trans = con.BeginTransaction()) {
            SQLiteCommand cmd = con.CreateCommand();
            // インサート
            cmd.CommandText = "DELETE FROM t_product WHERE CD = @Cd;";
            // パラメータセット
            cmd.Parameters.Add("Cd", System.Data.DbType.Int64);
            // データ削除
            cmd.Parameters["Cd"].Value = int.Parse(textBox6.Text);
            cmd.ExecuteNonQuery();
            // コミット
            trans.Commit();
        }
    }
}
```

DELETE文を作ることができます

The screenshot shows the same Windows Form 'Form1'. The 'データ読み込み' button is still highlighted. The data sections are now empty, indicating that the data has been deleted. The '検索CD' field now contains '1', and the 'productname' and 'price' fields are empty.

```
private void button6_Click(object sender, EventArgs e) {
    // コネクションを開いてテーブル削除して閉じる
    using (var con = new SQLiteConnection("Data Source=test.db")) {
        con.Open();
        using (SQLiteCommand command = con.CreateCommand()) {
            command.CommandText =
                "drop table t_product";
            command.ExecuteNonQuery();
        }
        con.Close();
    }
}
```

テーブルを削除するときはdrop tableを使います