https://www.halvorsen.blog

Consuming PHP Web API in WinForms App

Hans-Petter Halvorsen

Contents

- We have in a previous Tutorial made a simple CRUD Web API using PHP and MySQL.
  - CRUD means Create, Read, Update and Delete data in the Database.
- In a previous Tutorial I have also made a Windows Forms CRUD Application that has direct access to a SQL Database and the Windows Forms App communicate directly with the database using the ADO.NET package in Visual Studio/C#.
- In real-life scenarios you normally don't have direct access to the database due to security issues.
- Also to be able to get direct access to the database you need to specify access to your IP address in the server firewall settings.
  - That may be OK for 1 or 2 computers, but what if hundreds or thousands of computer need access?
- So, the focus in this Tutorial is to make an updated version of the previous Windows Forms CRUD Application that has direct access to a SQL Database where we use the CRUD PHP Web API instead.
  - The CRUD PHP Web API relays entirely on HTTP so no IP access, etc. need to be set up.

Introduction

Hans-Petter Halvorsen

Table of Contents

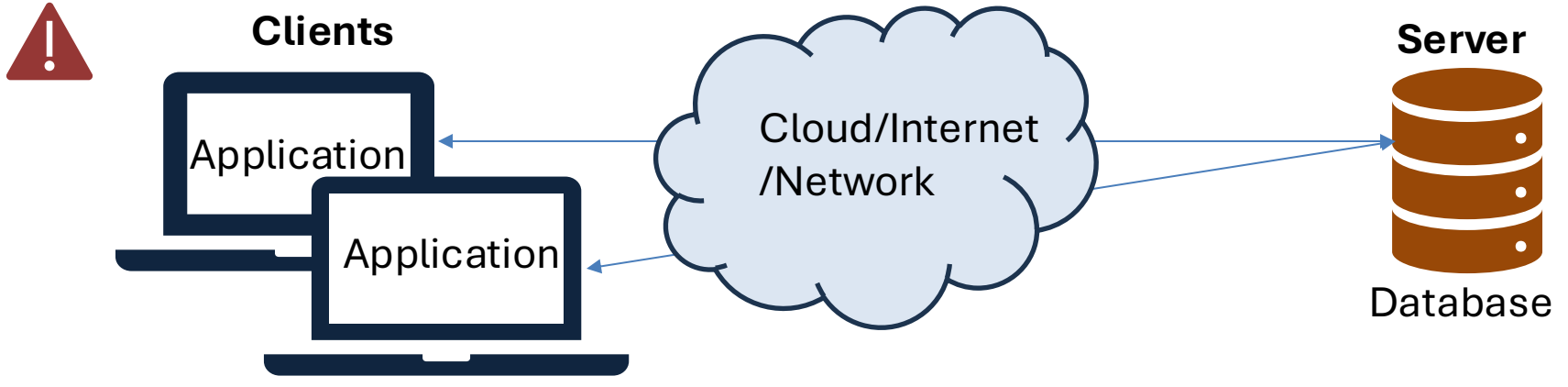
Web API

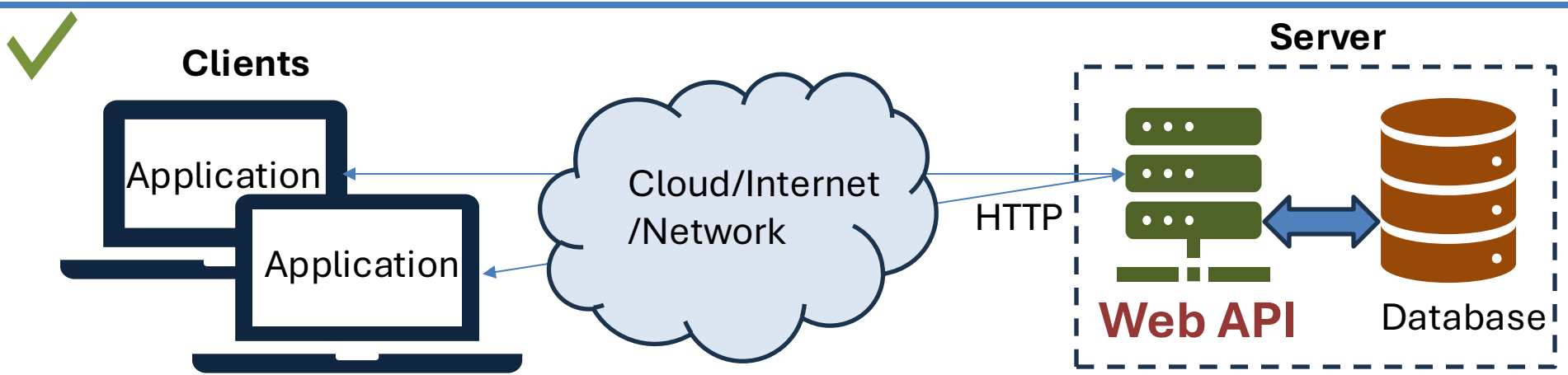
- We can create/use APIs for internal use inside an Application or between 2 or more Applications.
- Basically, an API can be just a Class with Methods that you use several places inside an Application or that you share between multiple Applications.
- A set of Stored Procedures in a Database can also be an API.
- When the Application that consume/use the API is on a local PC and the API itself is located on a Server, we can talk about so-called “Web APIs”.
- Such Web APIs also very often perform CRUD operations against a Database located on the Web.
- Normally it is not allowed to connect directly to a Database located in the Cloud from a local computer unless you configure and give access to the IP addresses for those clients.

CRUD: **C**reate, **R**ead, **U**pdate, **D**elete Data

Web API

Normally it is not allowed to connect directly to a Database located in the Cloud from a local computer unless you configure and give access to the IP addresses for those clients.





References

- Make HTTP requests with the HttpClient class: https://learn.microsoft.com/en-us/dotnet/fundamentals/networking/http/httpclient

Backend/Server-side

Table of Contents

Hans-Petter Halvorsen

Database

A simple Database with the following Table is used:

```
CREATE TABLE BOOK
(
    BookId int PRIMARY KEY AUTO_INCREMENT,
    Title varchar(100) NOT NULL,
    Author varchar(100) NOT NULL,
    Topic varchar(100) NOT NULL
);
```

CRUD PHP Web API

In a previous Tutorial we have made a simple Web API in PHP that has CRUD functionality and communicated with a MYSQL Database.

The following methods was created in the Web API:

- **GetBooks**.php
- **GetBookById**.php
- **InsertBook**.php
- **UpdateBook**.php
- **DeleteBook**.php

https://www.halvorsen.blog

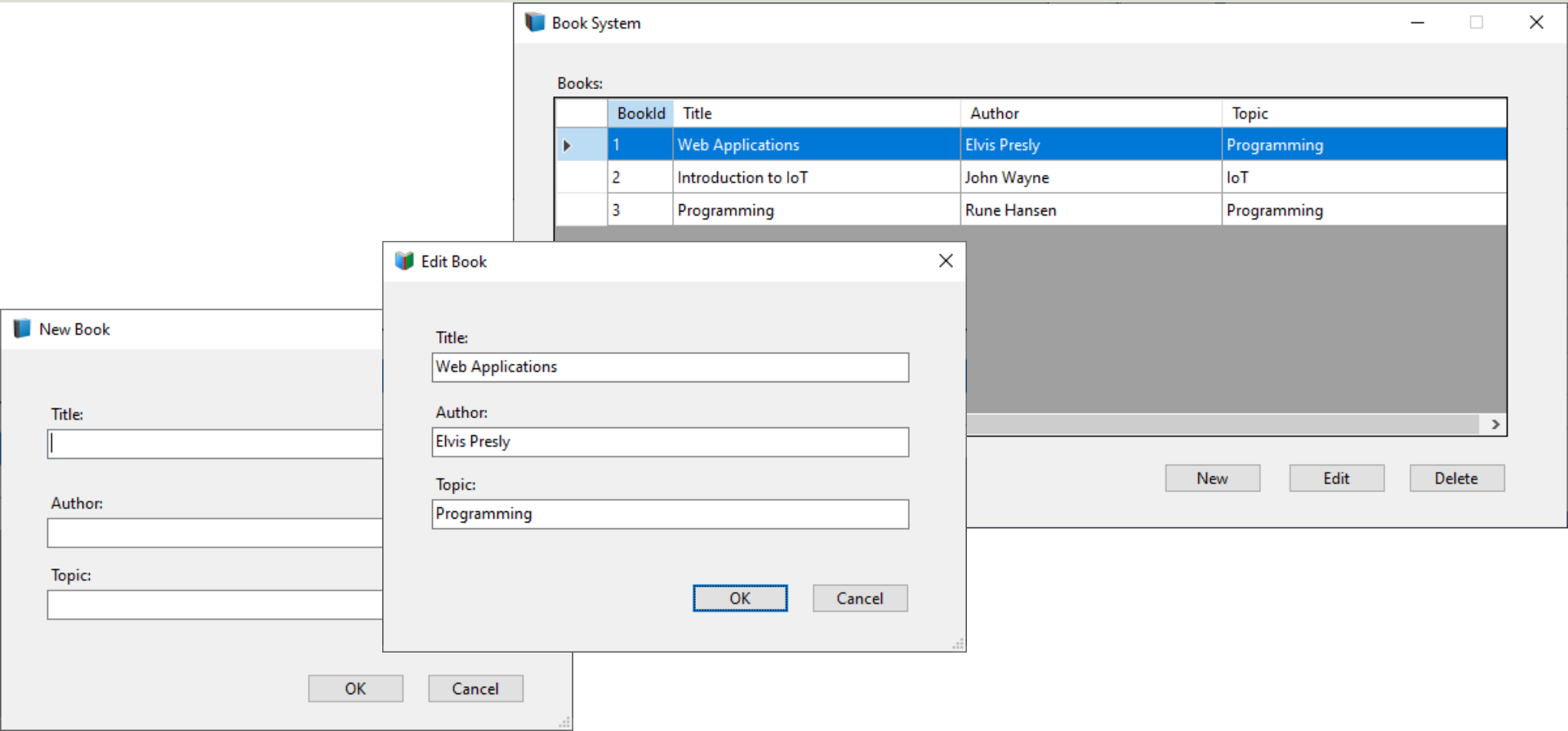
Windows Forms App

with direct Database Communication

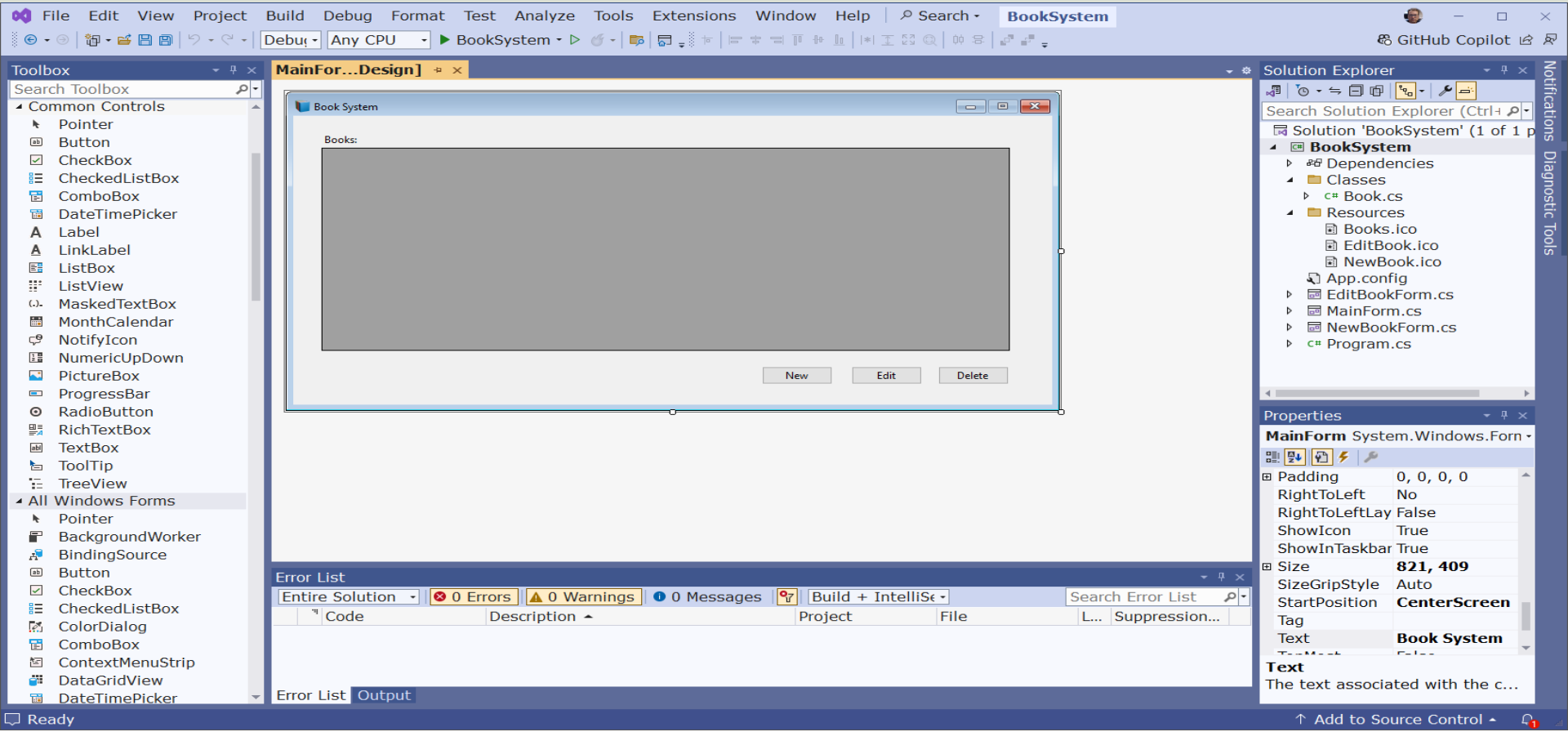
Hans-Petter Halvorsen

Table of Contents

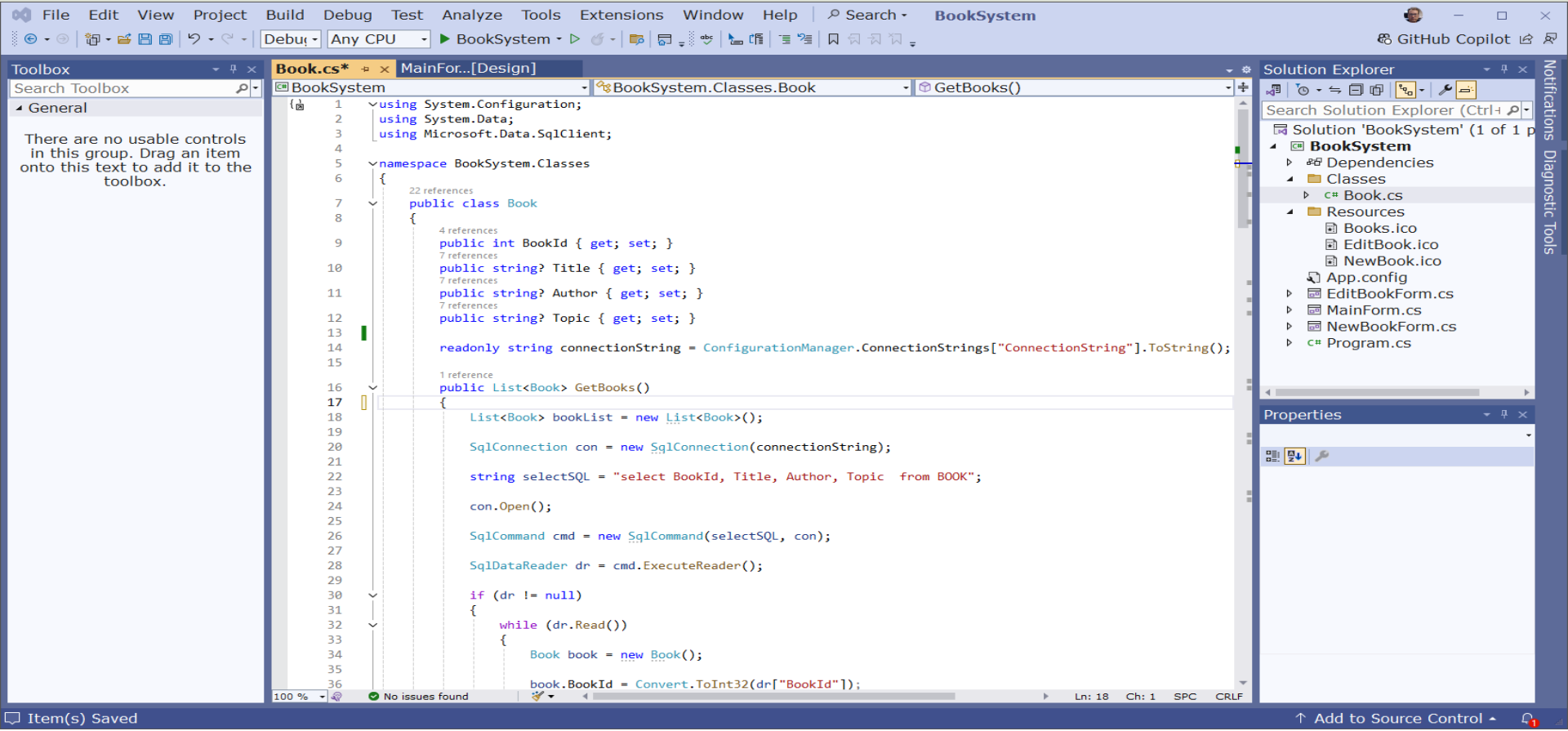
WinForm App


Book System
Books:
BookId
Title
Author
Topic
1
Web Applications
Elvis Presly
Programming
2
Introduction to IoT
John Wayne
IoT
3
Programming
Rune Hansen
Programming
New
Edit
Delete
Edit Book
Title:
Web Applications
Author:
Elvis Presly
Topic:
Programming
OK
Cancel
New Book
Title:
Author:
Topic:
OK
Cancel

WinForm App Visual Studio



Book Class deals with Database



Book Class

Book.cs

BookSystem | BookSystem.Classes.Book | Topic

```
1   using System.Configuration;
2   using System.Data;
3   using Microsoft.Data.SqlClient;
4
5   namespace BookSystem.Classes
6   {
        22 references
7       public class Book
8       {
            4 references
9           public int BookId { get; set; }
            7 references
10          public string? Title { get; set; }
            7 references
11          public string? Author { get; set; }
            7 references
12          public string? Topic { get; set; }
13
14          readonly string connectionString = ConfigurationManager.ConnectionStrings["ConnectionString"].ToString();
15
            1 reference
16          public List<Book> GetBooks()...
48
            1 reference
49          public Book GetBookData(int bookId)...
78
            1 reference
79          public void CreateBook(Book book)...
94
            1 reference
95          public void EditBook(Book book)...
111
            1 reference
112         public void DeleteBook(int bookId)...
125
126     }
127 }
```

Methods:

- GetBooks
- GetBookData
- CreateBook
- EditBook
- DeleteBook

https://www.halvorsen.blog

Windows Forms App

using simple PHP Web API

Hans-Petter Halvorsen

Table of Contents

Updated WinForms App

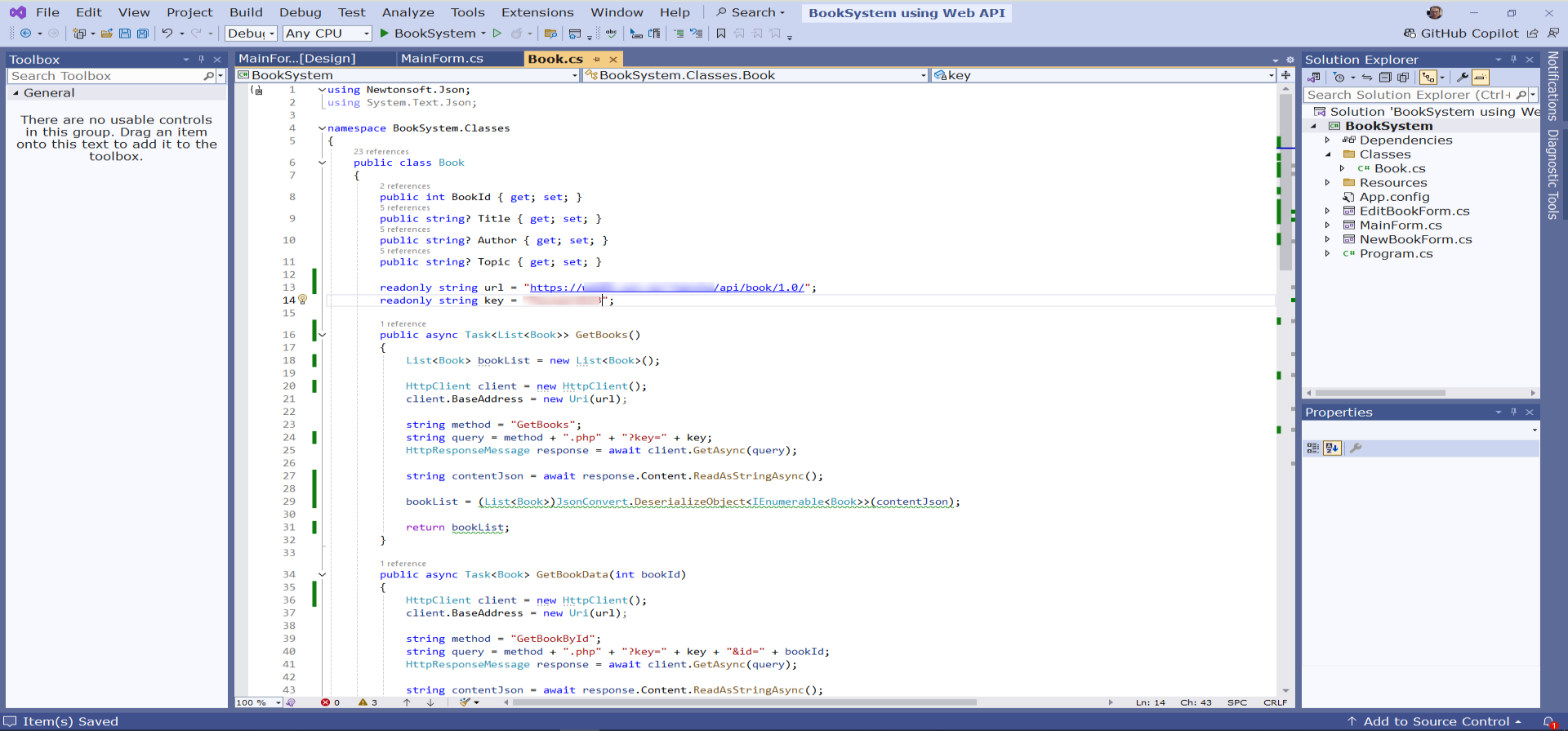
- Now we will update the WinForms App using the PHP Web API instead.
- We have already tested the API in the Web Browser and in Python.
- Now wee need to implement API communication in Visual Studio/C#.
- We use the built-in **HttpClient** Class in C#.
- The only thing we need to do in our WinForm App is to update the Book Class that's communicates with the API instead of direct Database access.

HttpClient

```
HttpClient client = new HttpClient();

string urlApi = "https://server/api/book/1.0/";
client.BaseAddress = new Uri(urlApi);

string apiMethod = "GetBooks.php";
HttpResponseMessage response = await client.GetAsync(apiMethod);

string contentJson = await response.Content.ReadAsStringAsync();
```

[{ "BookId": "1", "Title": "Arduino", "Author": "Hans-Petter", "Topic": "Programming" }, { "BookId": "2", "Title": "Raspberry Pi", "Author": "John Wayne", "Topic": "IoT" }, { "BookId": "22", "Title": "Arduino", "Author": "Hans-Petter", "Topic": "Microcontrollers" }, { "BookId": "24", "Title": "SQL", "Author": "Hans-Petter", "Topic": "Programming" }]

Visual Studio



Book Class

Book.cs

BookSystem | BookSystem.Classes.Book | url

```
public class Book
{
    2 references
    public int BookId { get; set; }
    5 references
    public string? Title { get; set; }
    5 references
    public string? Author { get; set; }
    5 references
    public string? Topic { get; set; }

    readonly string url = "https://[redacted]/api/book/1.0/";
    readonly string key = "[redacted]";

    1 reference
    public async Task<List<Book>> GetBooks()...

    1 reference
    public async Task<Book> GetBookData(int bookId)...

    1 reference
    public async void CreateBook(Book book)...

    1 reference
    public async void EditBook(Book book)...

    1 reference
    public async void DeleteBook(int bookId)...

}
```

Methods:

- GetBooks
- GetBookData
- CreateBook
- EditBook
- DeleteBook

GetBooks Method

```
public async Task<List<Book>> GetBooks()
{
   List<Book> bookList = new List<Book>();

   HttpClient client = new HttpClient();
   client.BaseAddress = new Uri(url);

   string method = "GetBooks";
   string query = method + ".php" + "?key=" + key;
   HttpResponseMessage response = await client.GetAsync(query);

   string contentJson = await response.Content.ReadAsStringAsync();

   bookList =
(List<Book>)JsonConvert.DeserializeObject<IEnumerable<Book>>(contentJson);

   return bookList;
}
```

GetBookData Method

```
public async Task<Book> GetBookData(int bookId)
{
    HttpClient client = new HttpClient();
    client.BaseAddress = new Uri(url);

    string method = "GetBookById";
    string query = method + ".php" + "?key=" + key + "&id=" + bookId;
    HttpResponseMessage response = await client.GetAsync(query);

    string contentJson = await response.Content.ReadAsStringAsync();
    contentJson = contentJson.Replace("[", "");
    contentJson = contentJson.Replace("]", "");

    Book? book = new Book();
    book = Newtonsoft.Json.JsonConvert.DeserializeObject<Book>(contentJson);

    return book;
}
```

CreateBook Method

```
public async void CreateBook(Book book)
{
    HttpClient client = new HttpClient();
    client.BaseAddress = new Uri(url);

    string method = "InsertBook";
    string query = method + ".php" + "?key=" + key + "&title=" +
      book.Title + "&author=" + book.Author + "&topic=" +
      book.Topic;

    HttpResponseMessage response = await client.GetAsync(query);

    string result = await response.Content.ReadAsStringAsync();
}
```

EditBook Method

```
public async void EditBook(Book book)
{
    HttpClient client = new HttpClient();
    client.BaseAddress = new Uri(url);

    string method = "UpdateBook";
    string query = method + ".php" + "?key=" + key + "&id=" +
      book.BookId + "&title=" + book.Title + "&author=" +
      book.Author + "&topic=" + book.Topic;

    HttpResponseMessage response = await client.GetAsync(query);

    string result = await response.Content.ReadAsStringAsync();
}
```

DeleteBook Method

```
public async void DeleteBook(int bookId)
{
    HttpClient client = new HttpClient();
    client.BaseAddress = new Uri(url);

    string method = "DeleteBook";
    string query = method + ".php" + "?key=" + key + "&id=" +
       bookId;

    HttpResponseMessage response = await client.GetAsync(query);

    string result = await response.Content.ReadAsStringAsync();
}
```

Summary

- In previous Tutorials we have made
  - A basic CRUD WinForm Desktop App that communicates directly with a SQL Database
    - This is “bad practice” and very often not allowed
  - So, In another Tutorial we made a simple PHP CRUD Web API
- In this Tutorial we updated the WinForm App using the Web API instead
- The code is very basic and don’t follow best practice, can be better structured, include error handling, etc.
- The code is made simple to illustrate the basic principles using Web APIs

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog