IRunes App

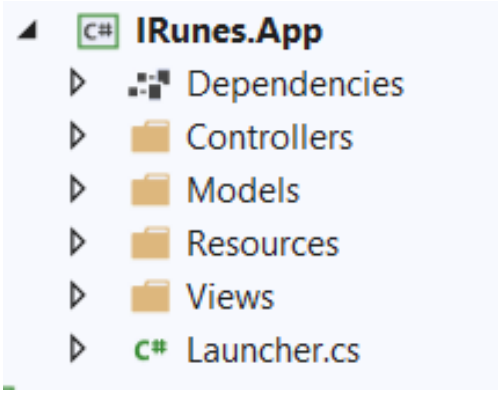
След като изградихме нашият HTTP Server е време да го впрегнем в действие и да видим как работи.

1. Архитектура

Първо нека да създадем архитектурата на нашият проект. Към вече съществуващият sln. добавете нов проект и го кръстете **IRunes.App**

2. IRunes.App Архитектура

IRunes.App ще съдържа логиката за контролер, моделите, ресурсите и html файловете. Ще имаме един клас "Launcher" от където ще започва нашата логика.



Нека да започнем с папката Controllers:

Controllers папка

"**Controllers**" папката, ще съдържа нашите контролери, благодарение на тях ние ще може да обработваме данни, да пренасочване ресурси и т.н. За сега ще създадем един клас, който ще се казва **`BaseController`**

BaseController клас

Създайте абстрактен клас "**`BaseController`**". Това ще бъде базовият клас за всички контролери.

```
public abstract class BaseController
{
    protected BaseController()
    {
        this.ViewData = new Dictionary<string, object>();
    }

    protected Dictionary<string, object> ViewData;

    private string ParseTemplate(string viewContent)...

    protected IHttpResponse View([CallerMemberName] string view = null)...

    protected IHttpResponse Redirect(string url)...
}
```

ViewData пропърти

То ще помогне, когато искаме да визуализираме данни. Първо ще ги добавим в този асоциативен масив и после **`ParseTemplate`** метода ще ги обработи.

ParseTemplate метод

Този метод ще обходи целият речник и там, където засече **`@Model.`** в нашият html, ще го замени с данните, които искаме. Този метод ще стане по-ясен, когато стигнем до описването на html файлове.

```
private string ParseTemplate(string viewContent)
{
    foreach (var param in this.ViewData)
    {
        viewContent = viewContent.Replace($"@Model.{param.Key}",
            param.Value.ToString());
    }

    return viewContent;
}
```

View метод

Това е най-сложният метод в контролер класа. Той е отговорен да върне правилният html и с правилно заредени данни.

```
protected IHttpResponse View([CallerMemberName] string view = null)
{
    string controllerName = this.GetType().Name.Replace("Controller", string.Empty);
    string viewName = view;

    string viewContent = File.ReadAllText("Views/" + controllerName + "/" + viewName + ".html");

    viewContent = this.ParseTemplate(viewContent);

    HtmlResult htmlResult = new HtmlResult(viewContent, HttpResponseStatusCode.Ok);

    return htmlResult;
}
```

"**`CallerMemberName`**" е атрибут, който ще върне информация откъде е бил извикан този метод. Така след, като знаем името на контролера и името на html файла, ние може да открием къде се намира и го върнем на потребителя, но при това използваме нашият метод **`ParseTemplate`**, който ще помогне за зараждане на допълнителна информация.

Redirect метод

```
protected IHttpResponse Redirect( string url)
{
    return new RedirectResult(url);
}
```

Models папка

"**`Models`**" папката, ще съдържа нашите моделите, които ще си комуникират с базата за в бъдеще. Сега създайте един клас **`Album`** със следните пропъртита:

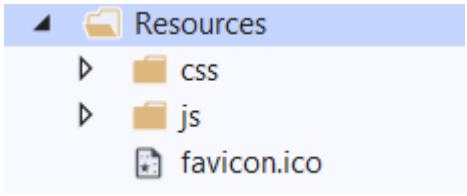
```
public class Album
{
    public string Id { get; set; }

    public string Name { get; set; }

    public string Cover { get; set; }

    public decimal Price { get; set; }
}
```

Resources папка



Изтеглете от ресурсите на сайта тази папка и я добавете към вашият проект.

Views папка

Изтеглете от ресурсите на сайта тази папка и я добавете към вашият проект. В нея ще откриете вече разписаните html файлове.

Controllers папка

В Controllers папката добавете два нови класа – **`HomeController`** и **`AlbumsController`**

`HomeController` класа трябва да съдържа метод – **`IHttpResponse Index(IHttpRequest httpRequest)`**

`AlbumsController` класа трябва да съдържа два метода – **`IHttpResponse Create(IHttpRequest httpRequest)`** и **`CreateConfirm(IHttpRequest httpRequest)`**

Launcher клас

Този е клас е нашата стартираща точка. Той съдържа **`Main`** метод в себе си. Създайте нова инстанция на **`ServerRoutingTable`**. След това регистрирайте пътищата до папките:

```
public static void Main(string[] args)
{
    ServerRoutingTable serverRoutingTable = new ServerRoutingTable();

    serverRoutingTable.Add(HttpRequestMethod.Get, "/", request => new RedirectResult("/Home/Index"));
    serverRoutingTable.Add(HttpRequestMethod.Get, "/Home/Index", request => new HomeController().Index(request));

    serverRoutingTable.Add(HttpRequestMethod.Get, "/Albums/Create", request => new AlbumsController().Create(request));
    serverRoutingTable.Add(HttpRequestMethod.Post, "/Albums/Create", request => new AlbumsController().CreateConfirm(request));

    Server server = new Server(8000, serverRoutingTable);
    server.Run();
}
```

Сега ако стартирате проекта и имплементирате правилно Index метода в HomeController трябва да ви покаже днешната дата и час.

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист".

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под **свободен лиценз CC-BY-NC-SA** (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).