

Avancerad webbutveckling

Övningsuppgift 3

Introduktion

Övningsuppgiften går ut på att öva på några olika koncept och tekniker som vi hittills har gått igenom i kursen. Uppgiften behandlar utöver det grundläggande inom MVC som t.ex. routing, vyer och controllers med tillhörande action methods även implementation av ett middleware, bearbeta HttpContext-objektet, deserialisera JSON från tredjepart, applikationsinställningar via appsettings, egenbyggd tag helpers, skicka custom request headers till servern och användning av System.Reflection.

TL;DR

Applikationen kommer gå ut på att med hjälp av ett plugin i Chrome så skickas en speciell header med för varje request till servern. Headern i fråga innehåller ett godtyckligt (riktigt) IP-nummer. Ett middleware kommer att inspektera requesten och fånga upp IP-numret.

Sedan kommer det på programmatisk väg göras ett API-anrop till en IP/Geo-tjänst dit IP-numret skickas. Tillbaka får man JSON innehållande diverse information om vart IP-numret härstammar; t.ex. land, landkod, latitud, longitud med mera.

JSON-responsen i fråga ska deserialiseras till en instans av en modell/POCO och tilldelas till HttpContexten.

När request pipelin väl har kommit till controller/action så ska objektet som tidigare lades till i HttpContexten skickas ut till vyn.

Väl i vyn ska objektet och dess egenskaper/värden genereras på dynamiskt vis i en tabell. Tabellen kan och ska byggas upp med hjälp av System.Reflektion.

I vyn ska en tag helper finnas som kommer ha till uppgift att rita ut en karta som motsvarar det land som IP-numret tillhör.

Se bild på sista sidan för idé och inspiration.

Grunduppsättning

Skapa en tom ASP.NET Core-applikation eller bygg vidare på applikation från tidigare övningsuppgift. Skapa sedan upp en ny controller med tillhörande action method och vy och ställ in routingen så att just denna controller och action används som "startside".

Skapa en appsettings-fil innehållande en sektion med ett vettigt namn, t.ex. "CustomAppSettings". Sektionen ska åtminstone innehålla en egenskap med tillhörande värde som pekar på endpointen för IP-API:et. Se länkar längre ner.

Lägg till pluginet "ModHeader" i Google Chrome. Lägg till en egen header med exempelvis namnet "spoof-ip-address", och tilldela denna ett värde. Se länkar längre ner för sida innehållande IP-adresser till olika länder.

Middleware

Skapa ett middleware som kan användas på följande vis:

```
app.UseIpLookup();
```

Middlewareet ska köras i samband med varje request och ska som beskrivet ovan titta på om spoof-ip-address headern finns - och om den finns, köra en sökning mot IP-API:et och deserialisera JSON-svaret till ett objekt och lägga till i HttpContexten via .Items.

Endpoint-adressen till API:et ska läsas från appsettings och DI via IOptions-mönstret.

Middlewareet efter det att den gjort sitt passa vidare requesten till nästa middleware i kedjan.

Vyn

Vyn ska vara hårt typad, och lämpligtvis av samma typ som JSON-deserialiseringen gav. System.Reflection ska användas för att bygga upp en tabell med avseende på modellens egenskaper/värden.

Pseudokod:

```
foreach(var prop in Model.Properties){  
    key: @prop.Name, value: @prop.Value  
}
```

Tag helper för flagga

En tag helper ska byggas som kan användas i stil med följande, där CountryCode sätts till det värde som modellen har:

```
<flag country-code="SE"></flag>
```

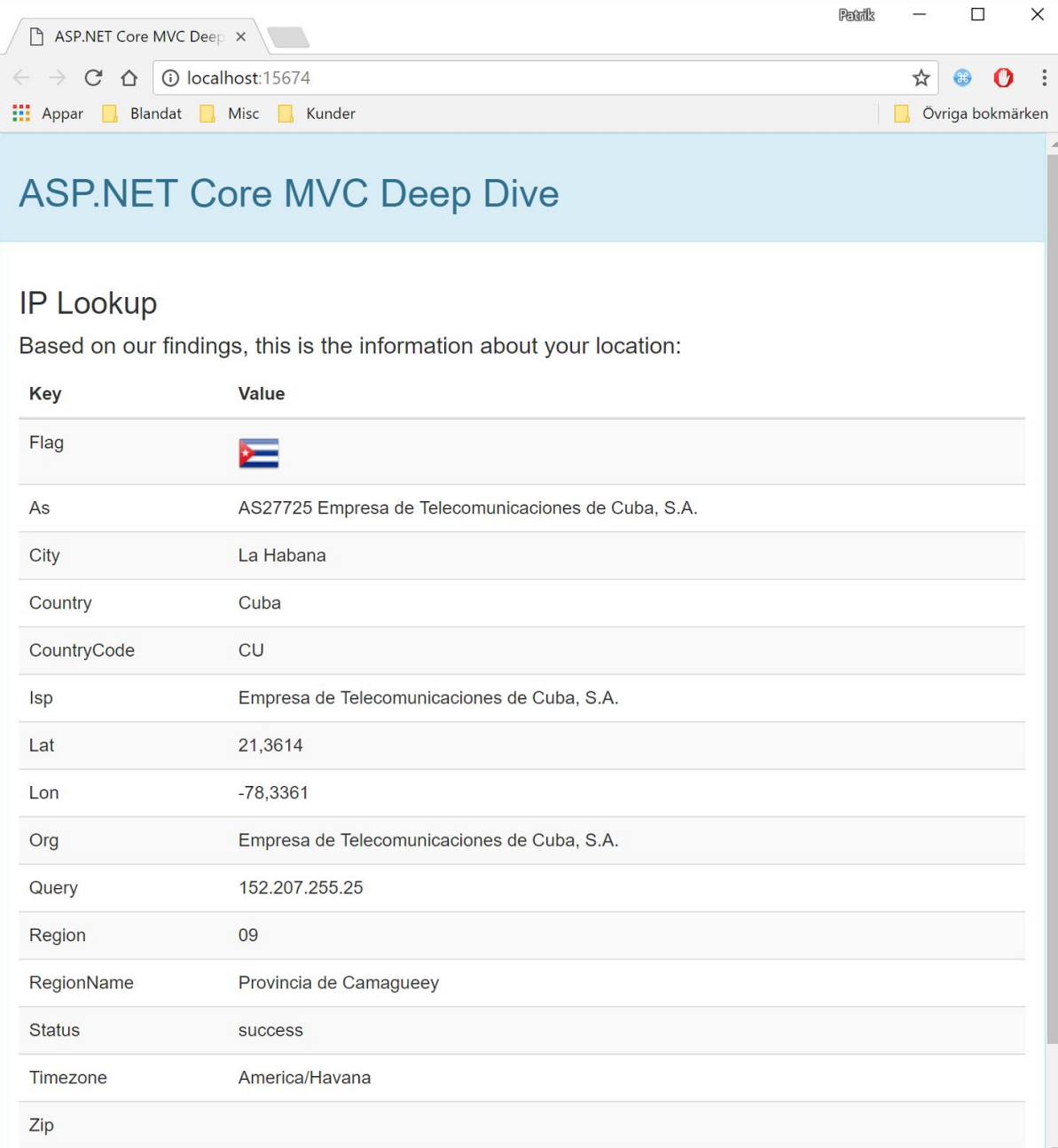
Se länk nedan för en smidig och färdig CSS-lösning för att visa kartor med hjälp av olika CSS-klasser.

Resurser

- Google Chrome-plugin för att sätta extra request headers:
https://chrome.google.com/webstore/detail/modheader/idgpnmonknjnojddfkpgkljpfnnfcklj?utm_source=chrome-app-launcher-info-dialog
- Lista med landskoder som ni bör känna igen från svaret från IP-API:et
https://en.wikipedia.org/wiki/ISO_3166-1_alpha-2
- CSS-sprite för flaggor
<https://github.com/lafeber/world-flags-sprite>
- Online-verktyg för att göra om JSON till en POCO
<http://json2csharp.com/>
- Ovan kan även åstadkommas i VS om man har Web Essentials:
<https://stackoverflow.com/questions/21611674/how-to-auto-generate-a-c-sharp-class-file-from-a-json-object-string/21611680#21611680>
- Olika länders IP-adresser
<http://www.nirsoft.net/countryip/>

- IP-API
<http://ip-api.com/>
- IP-API exempel på GET som returnerar JSON för Kuba
<http://ip-api.com/json/152.207.255.255>


Exempel på resultat



ASP.NET Core MVC Deep Dive

IP Lookup

Based on our findings, this is the information about your location:

Key	Value
Flag	
As	AS27725 Empresa de Telecomunicaciones de Cuba, S.A.
City	La Habana
Country	Cuba
CountryCode	CU
Isp	Empresa de Telecomunicaciones de Cuba, S.A.
Lat	21,3614
Lon	-78,3361
Org	Empresa de Telecomunicaciones de Cuba, S.A.
Query	152.207.255.25
Region	09
RegionName	Provincia de Camagueey
Status	success
Timezone	America/Havana
Zip	