ASP.NET Core Request Localization Options

Asked 5 years, 1 month ago Active 1 year, 1 month ago Viewed 25k times

23

6

Here is my custom request culture provider which returns "en" as a default culture if no culture specified in url (for example http://sypalo.com/ru or http://sypalo.com/en). My idea to show website on that language which is default in user's browser, so I'm looking a way how to determine it and return it instead of: return Task.FromResult(new ProviderCultureResult("en", "en"));

```
services.Configure<RequestLocalizationOptions>(options =>
{
    var supportedCultures = new List<CultureInfo>
    {
        new CultureInfo("en"),
        new CultureInfo("ru")
    };

    options.DefaultRequestCulture = new RequestCulture(culture: "en", uiCulture: "en");
    options.SupportedCultures = supportedCultures;
    options.SupportedUICultures = supportedCultures;

    options.RequestCultureProviders.Insert(0, new CustomRequestCultureProvider(context
=>
    {
        var pathSegments = context.Request.Path.Value.Split('/');
        if (pathSegments.Count() > 0)
        if (supportedCultures.Select(x =>
x.TwoLetterISOLanguageName).Contains((pathSegments[1])))
            return Task.FromResult(new ProviderCultureResult(pathSegments[1],
pathSegments[1]));
       return Task.FromResult(new ProviderCultureResult("en", "en"));
   }));
});
```

c# asp.net asp.net-mvc asp.net-core

Share Edit Follow

asked Aug 17 '16 at 21:39

Sergey Sypalo

911 4 11 27

4 Answers

Active Oldest Votes

33

✓

You can get `Accept-Language` header from the current `Request` and set default language. Your code should be something like this:

```
services.Configure<RequestLocalizationOptions>(options =>
{
    //...

    options.RequestCultureProviders.Insert(0, new CustomRequestCultureProvider(context
=>
    {
       //...
       var userLangs = context.Request.Headers["Accept-Language"].ToString();
       var firstLang = userLangs.Split(',').FirstOrDefault();
       var defaultLang = string.IsNullOrEmpty(firstLang) ? "en" : firstLang;
       return Task.FromResult(new ProviderCultureResult(defaultLang, defaultLang));
   }));
});
```

Share Edit Follow

answered Aug 18 '16 at 6:54

adem caglin

18.6k 8 47 69

Thanks a lot Adem, this is exactly what I was looking for! – Sergey Sypalo Aug 18 '16 at 18:07

27

Another way to get the Accept-Language header with a framework call (ASP.NET Core):

```
HttpContext.Request.GetTypedHeaders().AcceptLanguage
```

Share Edit Follow

answered May 15 '18 at 13:12

Dennis de Laat

393 4 8

12 Do not forget to order languages by its priority `q`, e.g. `OrderByDescending(x => x.Quality ?? 1)`. – Evgeni Nabokov Mar 10 '19 at 6:32

5

Inside your `public class Startup` :

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    //...

    app.Use((context, next) =>
    {
        //get client prefered language
        var userLangs = context.Request.Headers["Accept-Language"].ToString();
        var firstLang = userLangs.Split(',').FirstOrDefault();

        //set allowed alanguage
        var lang = "en"; //default
        switch (firstLang)
        {
            case "hy": //allowed
            case "ru": //allowed
                lang = firstLang;
                break;
            default:
                //client language not supported
                lang = "en"; //use our default
                break;
        }

        //switch culture
        Thread.CurrentThread.CurrentCulture = new
System.Globalization.CultureInfo(lang);
        Thread.CurrentThread.CurrentUICulture = Thread.CurrentThread.CurrentCulture;

        //save for later use
        context.Items["ClientLang"] = lang;
        context.Items["ClientCulture"] = Thread.CurrentThread.CurrentUICulture.Name;

        // Call the next delegate/middleware in the pipeline
        return next();
    });

    //... then goes app.UseMvc etc..

}
```

Share Edit Follow

edited Aug 12 '20 at 15:28

answered Mar 26 '19 at 6:56

Nick Kovalsky
3,174 17 36

3

as an additional option, Here is a full solution: [Handle culture in route (URL) via RequestCultureProviders]

as an option as well, here is a simplified code for CustomRequestCultureProvider:

```
options.RequestCultureProviders.Insert(0, new CustomRequestCultureProvider(context =>
{
    var lang =
context.Request.GetTypedHeaders().AcceptLanguage.FirstOrDefault()?.Value.Value ??
Constants.Languages.EN_US;

    return Task.FromResult(new ProviderCultureResult(lang, lang));

}));
```

Share Edit Follow

answered Dec 2 '18 at 15:49

Dongdong
1,811 13 20