# .NET MAUI - reading glass status

Once you're playing around and get some life in the glasses, there are moments you're not sure if your glasses aren't listening or wonder about the battery or status.

The resulting project is this one: ExerciseGlassesInfoCallBack.zip

In this project, we'll extend the previous project and add a button to read the status of the glasses.

You will need VuzixSDK 1.0.2 - it has the extra classes and the classes are organized a little more logical.
To migrate from 1.0.1 to 1.0.2 you only need to change one namespace.

We continue here from the tutorial .NET MAUI - reporting events to the GUI

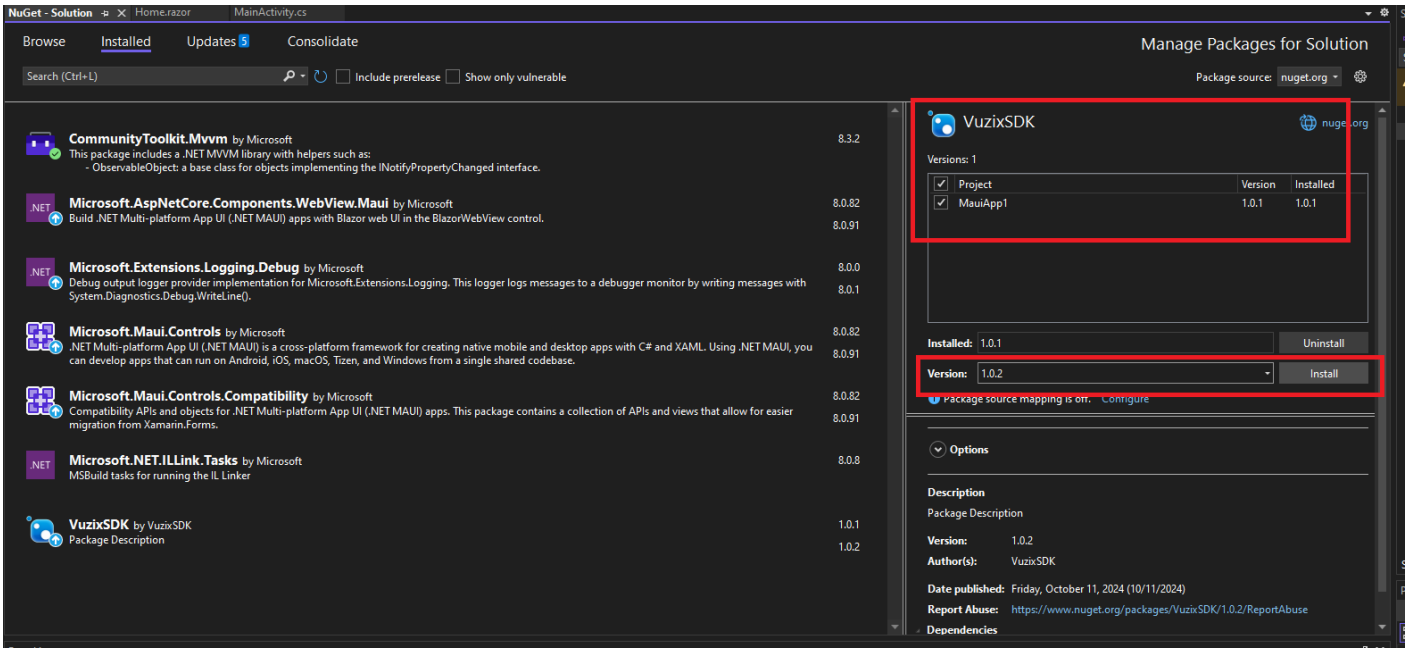
This is the code base to aadjust ExcersisTapCallback.zip

**Let's get to it!**

## Set up the android space

We will add the class to listen for. And when we receive it, send the information of the SDK back in Android towards the GUI/.NET stack. (Blazor, MAUI, .. how you want to label it: "the thing in the front")

1. Open your (familiar) solution and update the VuzixSDK package

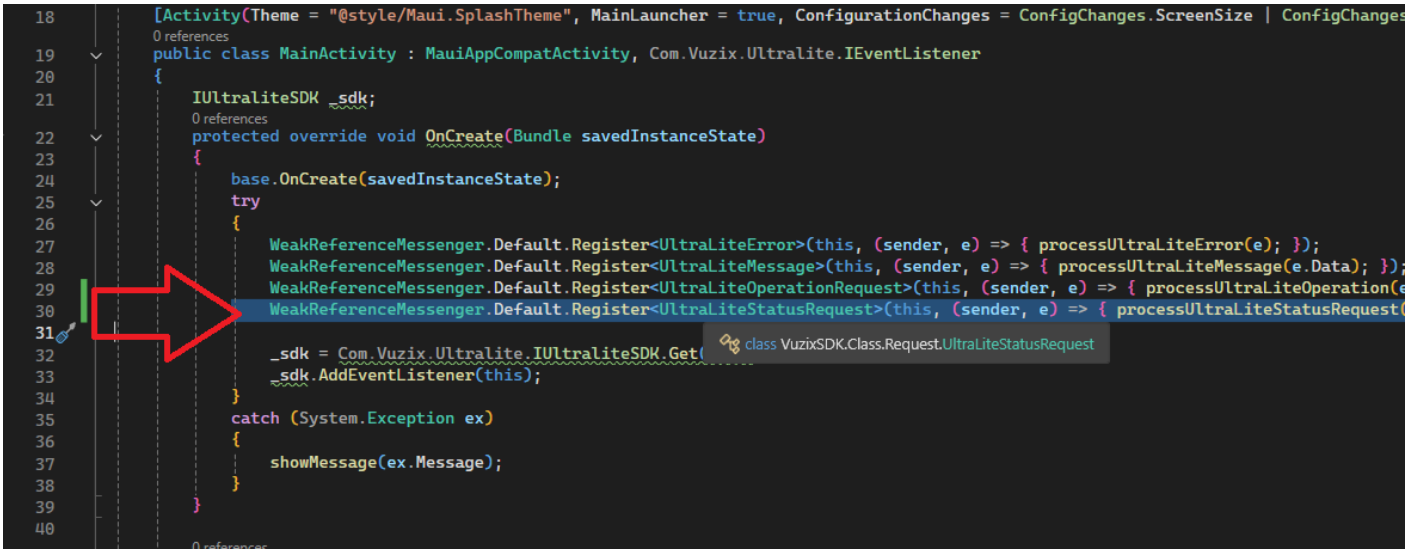



2. Include the new namspace in your MainActivity.cs

```
using VuzixSDK.Class.Request;
```

3. and add a listener for the new request in your MainActivity.cs

```
WeakReferenceMessenger.Default.Register(this, (sender, e) => { processUltraLiteStatusRequest()});
```




4. Write the function that sends the information back

```
    private void processUltraLiteStatusRequest()
    {
        if(_sdk != null)
        {
            WeakReferenceMessenger.Default.Send<UltraLiteStatusResponse>(new UltraLiteStatusResponse()
            {
                isAvailable = _sdk.IsAvailable,
                isCharging = _sdk.IsCharging,
                isConnected = _sdk.IsConnected,
                isControlled = _sdk.IsControlled,
                isControlledByMe = _sdk.IsControlledByMe,
                isLinked = _sdk.IsLinked,
                BatteryLevel = _sdk.BatteryLevel,
                Name = _sdk.Name
            });
        }
    }
```

Now your MainActivity.cs should look as follows:

```
using Android.App;
using Android.Content.PM;
using Android.OS;
using Android.Widget;
using CommunityToolkit.Mvvm.Messaging;
using VuzixSDK.Class;
using Com.Vuzix.Ultralite;
using Layout = Com.Vuzix.Ultralite.Layout;
using TextAlignment = Com.Vuzix.Ultralite.TextAlignment;
using VuzixSDK.Enum;
using VuzixSDK.Class.Request;
using Android.Graphics;
using System.Diagnostics.Tracing;
using static System.Net.Mime.MediaTypeNames;
```

```
namespace MauiApp1
{
    [Activity(Theme = "@style/Maui.SplashTheme", MainLauncher = true, ConfigurationChanges =
ConfigChanges.ScreenSize | ConfigChanges.Orientation | ConfigChanges.UiMode | ConfigChanges.ScreenLayout |
ConfigChanges.SmallestScreenSize | ConfigChanges.Density)]
    public class MainActivity : MauiAppCompatActivity, Com.Vuzix.Ultralite.IEventListener
□{
        IUltraliteSDK _sdk;
□□protected override void OnCreate(Bundle savedInstanceState)
        {
            base.OnCreate(savedInstanceState);
            try
            {
                WeakReferenceMessenger.Default.Register<UltraLiteError>(this, (sender, e) => {
processUltraLiteError(e); });
                WeakReferenceMessenger.Default.Register<UltraLiteMessage>(this, (sender, e) => {
processUltraLiteMessage(e.Data); });
                WeakReferenceMessenger.Default.Register<UltraLiteOperationRequest>(this, (sender, e) => {
processUltraLiteOperation(e); });
                WeakReferenceMessenger.Default.Register<UltraLiteStatusRequest>(this, (sender, e) => {
processUltraLiteStatusRequest();  });

                _sdk = Com.Vuzix.Ultralite.IUltraliteSDK.Get(this);
□□□□_sdk.AddEventListener(this);
□□□}
□□□catch (System.Exception ex)
            {
                showMessage(ex.Message);
            }
        }

□□public void OnTap(int tapCount)
□□{
            WeakReferenceMessenger.Default.Send<VuzixSDK.Class.Event.UltraLiteOnTap>(new
VuzixSDK.Class.Event.UltraLiteOnTap()
            {
                tapCount = tapCount
            });
□□}
```

```
□□private void OnDisplayOff()
□□{
        WeakReferenceMessenger.Default.Send<VuzixSDK.Class.Event.UltraLiteOnDisplayOff>(new
VuzixSDK.Class.Event.UltraLiteOnDisplayOff());
    }
□□private void OnDisplayOn()
□□{
        WeakReferenceMessenger.Default.Send<VuzixSDK.Class.Event.UltraLiteOnDisplayOn>(new
VuzixSDK.Class.Event.UltraLiteOnDisplayOn());
    }
    private void OnDisplayTimeout()
□□{
        WeakReferenceMessenger.Default.Send<VuzixSDK.Class.Event.UltraLiteOnDisplayTimeout>(new
VuzixSDK.Class.Event.UltraLiteOnDisplayTimeout());
    }
    private void OnPowerButtonPress(bool turningOn)
□□{
        WeakReferenceMessenger.Default.Send<VuzixSDK.Class.Event.UltraLiteOnPowerButtonPress>(new
VuzixSDK.Class.Event.UltraLiteOnPowerButtonPress()
        {
            turningOn = turningOn
        });
□□}
□□private void OnScrolled(bool isScreenEmpty)
□□{
        WeakReferenceMessenger.Default.Send<VuzixSDK.Class.Event.UltraLiteOnScrolled>(new
VuzixSDK.Class.Event.UltraLiteOnScrolled()
        {
            isScreenEmpty = isScreenEmpty
        });
    }
    private void processUltraLiteStatusRequest()
    {
        if(_sdk != null)
        {
            WeakReferenceMessenger.Default.Send<UltraLiteStatusResponse>(new UltraLiteStatusResponse()
            {
                isAvailable = _sdk.IsAvailable,
                isCharging = _sdk.IsCharging,
                isConnected = _sdk.IsConnected,
```

```
                isControlled = _sdk.IsControlled,
                isControlledByMe = _sdk.IsControlledByMe,
                isLinked = _sdk.IsLinked,
                BatteryLevel = _sdk.BatteryLevel,
                Name = _sdk.Name
            });
        }
    }
    public void clearScreen()
    {
        _sdk.Canvas.RemoveText(_lastTextId);
        _sdk.Canvas.RemoveAnimation(_lastAnimationId);
        _sdk.Canvas.RemoveImage(_lastImageId);
    }
    public void showMessage(string message)
    {
        MainThread.BeginInvokeOnMainThread(() =>
        {
            var toast = Toast.MakeText(this, message, ToastLength.Short);
            toast.Show();
        });
    }
    protected void processUltraLiteError(UltraLiteError error)
    {
        if (_sdk.IsConnected)
        {
            if (!_sdk.IsControlledByMe)
            {
                _sdk.RequestControl();
            }
            if (_sdk.IsControlledByMe)
            {
                string _title = $"[Error]{(error.Source != null ? " " + error.Source : "")}";
                string _error = (error.Exception != null ? $"Exception : {error.Exception.Message}" : "Error
occured");
                _sdk.SendNotification(_title, _error);
            }
        }
    }
    int _lastTextId = -1;
```

```
        int _lastImageId = -1;
        int _lastAnimationId = -1;

    protected void processUltraLiteMessage(String message)
    {
        if (_sdk.IsConnected)
        {
            if (!_sdk.IsControlledByMe)
            {
                _sdk.RequestControl();
            }
            if (_sdk.IsControlledByMe)
            {
                _sdk.SetLayout(Layout.Canvas, 0, true);
                bool _messageSucceeded = false;
                if (_lastTextId >= 0)
                {
                    _messageSucceeded = _sdk.Canvas.UpdateText(_lastTextId, message);
                }
                else
                {
                    _lastTextId = _sdk.Canvas.CreateText(message, Anchor.Center);
                    _messageSucceeded = (_lastTextId != -1);
                }
                if (!_messageSucceeded)
                {
                    showMessage("Text failed");
                }
                _sdk.Canvas.Commit();
                SystemClock.Sleep(1000);
            }
        }
    }
    protected void processUltraLiteOperation(UltraLiteOperationRequest Request)
        {
            if (_sdk.IsConnected)
            {
                if (!_sdk.IsControlledByMe)
                {
                    _sdk.RequestControl();
```

```
            }
            if (_sdk.IsControlledByMe)
            {
                _sdk.SetLayout(Layout.Canvas, 0, true);
                if (Request.Operation == eUltraLiteOperation.ShowImage && Request.ImageBitMap != null)
                {
                    LVGLImage image = loadLVGLImage(Request.ImageBitMap);
                    bool _imageSucceeded = false;
                    if (_lastImageId >= 0)
                    {
□□□□□□□_imageSucceeded = _sdk.Canvas.UpdateImage(_lastImageId, image);
                    }
                    else
                    {
                        _lastAnimationId = _sdk.Canvas.CreateImage(image, Anchor.Center);
□□□□□□□_imageSucceeded = (_lastImageId != -1);
□□□□□□}
                    if(!_imageSucceeded) showMessage("Image failed");
                    _sdk.Canvas.Commit();
                }
                if (Request.Operation == eUltraLiteOperation.ShowAnimation && Request.AnimationBitMap != null)
                {
                    LVGLImage[] image = loadLVGLImage(Request.AnimationBitMap);
                    int _animationDelay = 500;
                    if(_lastAnimationId >= 0)
                    {
                        _sdk.Canvas.RemoveAnimation(_lastAnimationId);
□□□□□□}
□□□□□□_lastAnimationId = _sdk.Canvas.CreateAnimation(image, Anchor.Center, _animationDelay);

□□□□□□if (_lastAnimationId == -1)
                    {
                        showMessage("Animation failed");
                    }
                    _sdk.Canvas.Commit();
                }
            }

        }
        else
```

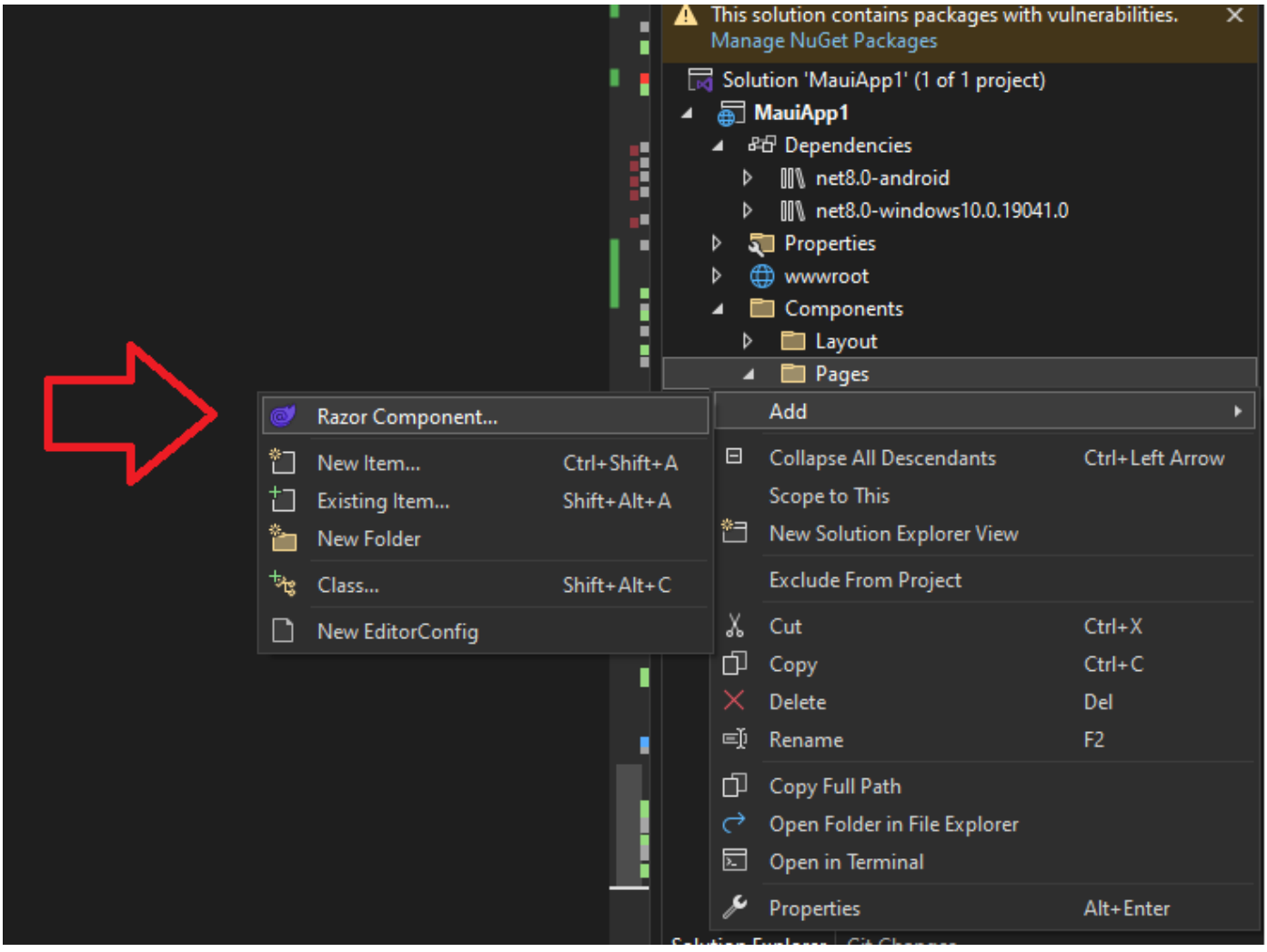
```
        {
            showMessage("SDK is not connected");
        }
    }
    private static Bitmap loadBitmap(byte[] bitmapbytes)
    {
        BitmapFactory.Options options = new BitmapFactory.Options();

        // https://proandroiddev.com/image-decoding-bitmaps-android-c039790ee07e
        options.InSampleSize = 2;

        //options.InTargetDensity = 640 * 2;
        //options.InTargetDensity = 480 * 8;
        //options.InScaled = true;
        options.InPreferredConfig = Bitmap.Config.Argb8888;
        /* options.InMutable = true;

        options.InSampleSize = 8;
□□□options.OutWidth = 600;
□□□options.OutHeight = 400;
□□□options.InScaled = scaled;*/
        Bitmap bmp = BitmapFactory.DecodeByteArray(bitmapbytes, 0, bitmapbytes.Length, options);

        return bmp;// resize(bmp, 640, 480);
    }
    private static LVGLImage[] loadLVGLImage(List<byte[]> images)
    {
        List<LVGLImage> _images = new List<LVGLImage>();
        foreach (var image in images)
        {
            _images.Add(LVGLImage.FromBitmap(loadBitmap(image), LVGLImage.CfIndexed1Bit));
        }
        return _images.ToArray();
    }
    private static LVGLImage loadLVGLImage(byte[] image)
    {
        //ColorObject[] _colors = { LVGLImage.IColorMapper.White, LVGLImage.IColorMapper.Mid };
        //LVGLImage _img = new LVGLImage(LVGLImage.CfIndexed1Bit, 480, 640, _colors, image);
        LVGLImage _img2 = LVGLImage.FromBitmap(loadBitmap(image), LVGLImage.CfIndexed1Bit);
        return _img2;
```

```
    }
  }
}
```

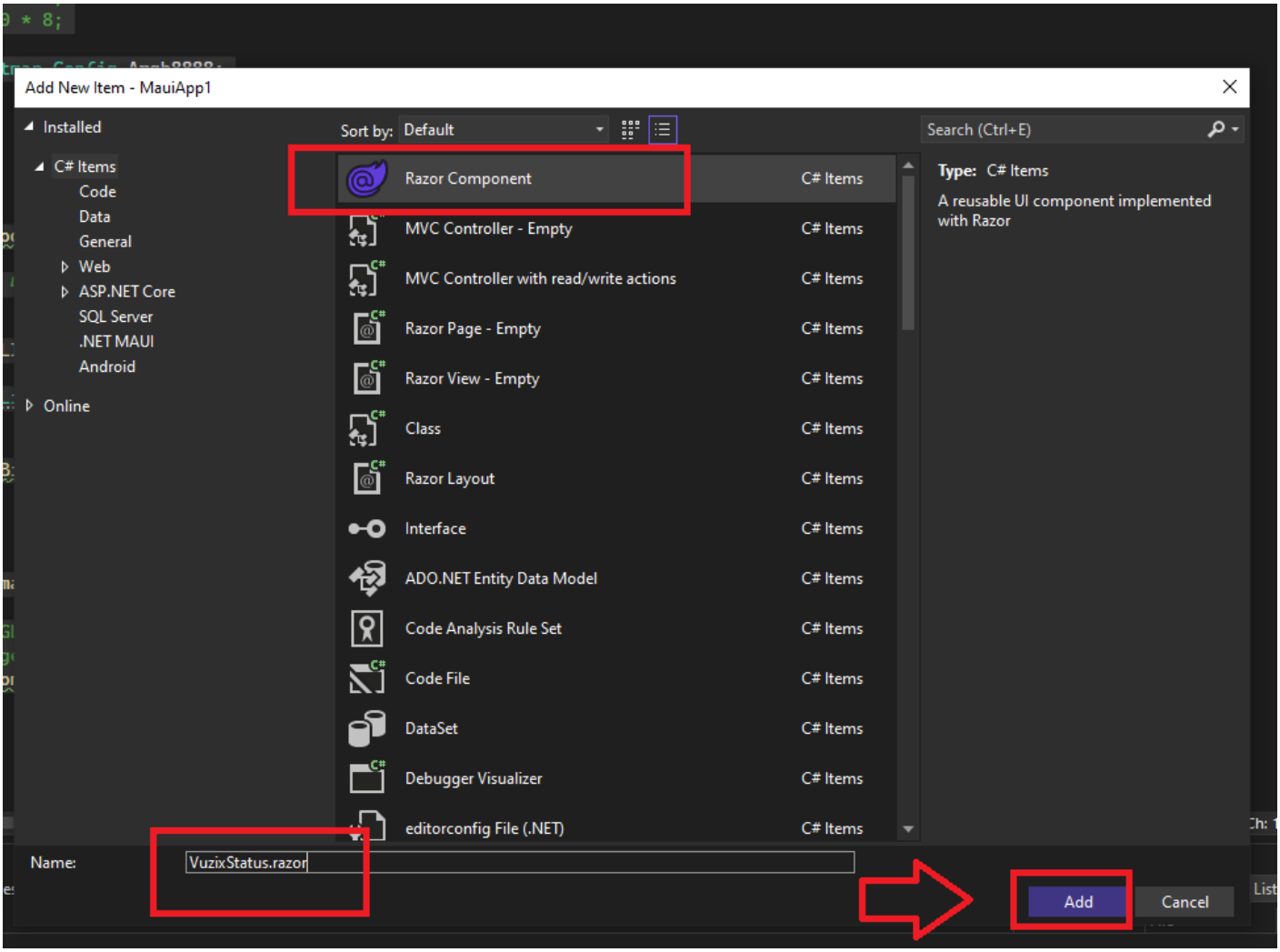
## Set up the pages and navigation in the UI

We now have to initiate the request for this information and then show it.
We'll make another page, add a button and write the information in text on the screen.

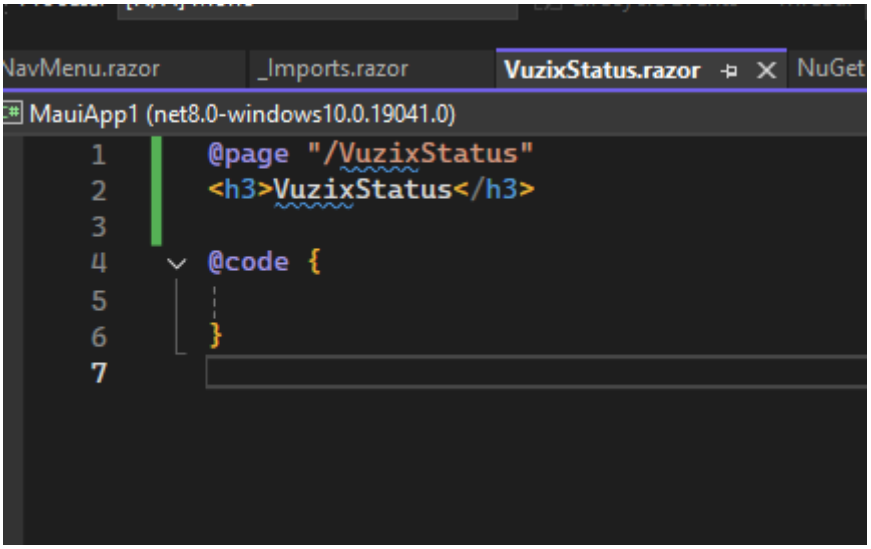
1. Right-click Pages and add Razor-component



2. Enter the name of the razor page and Add



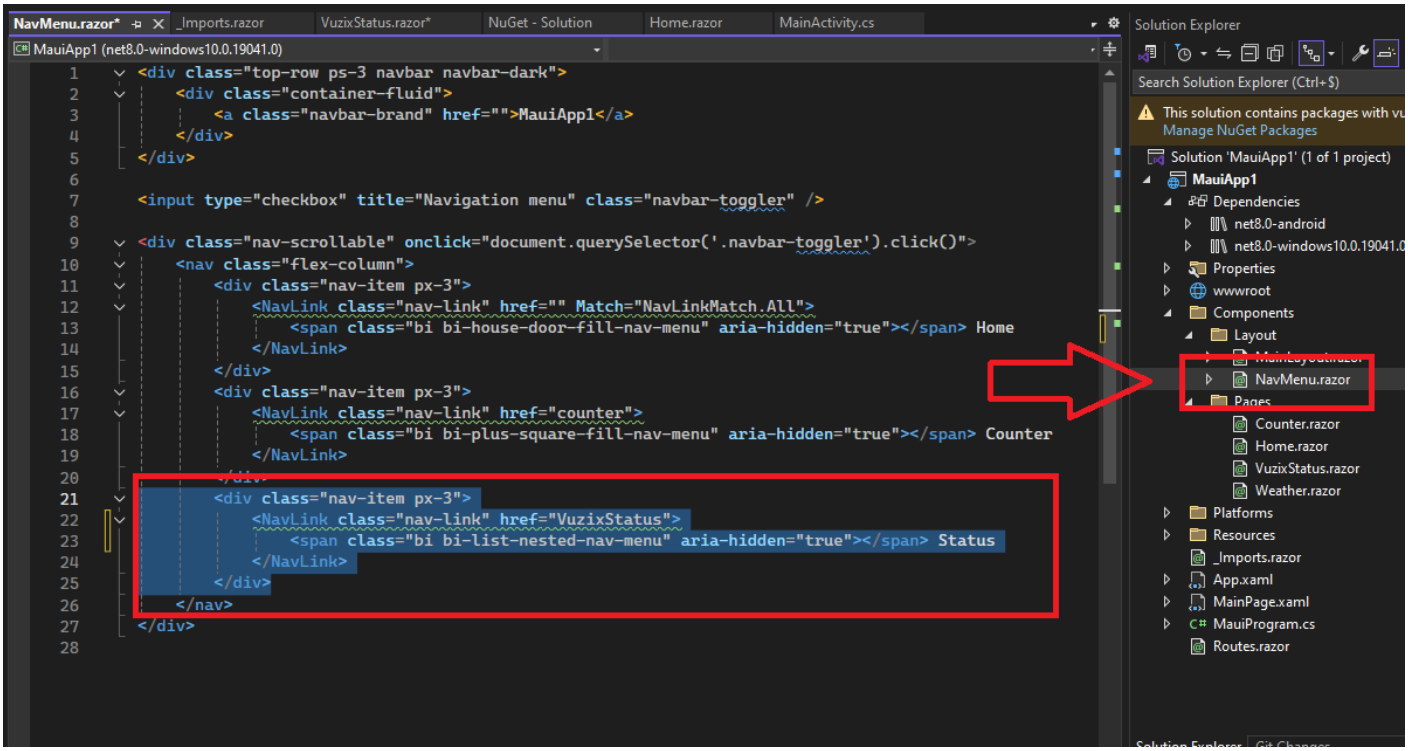

3. You'll see this as te page



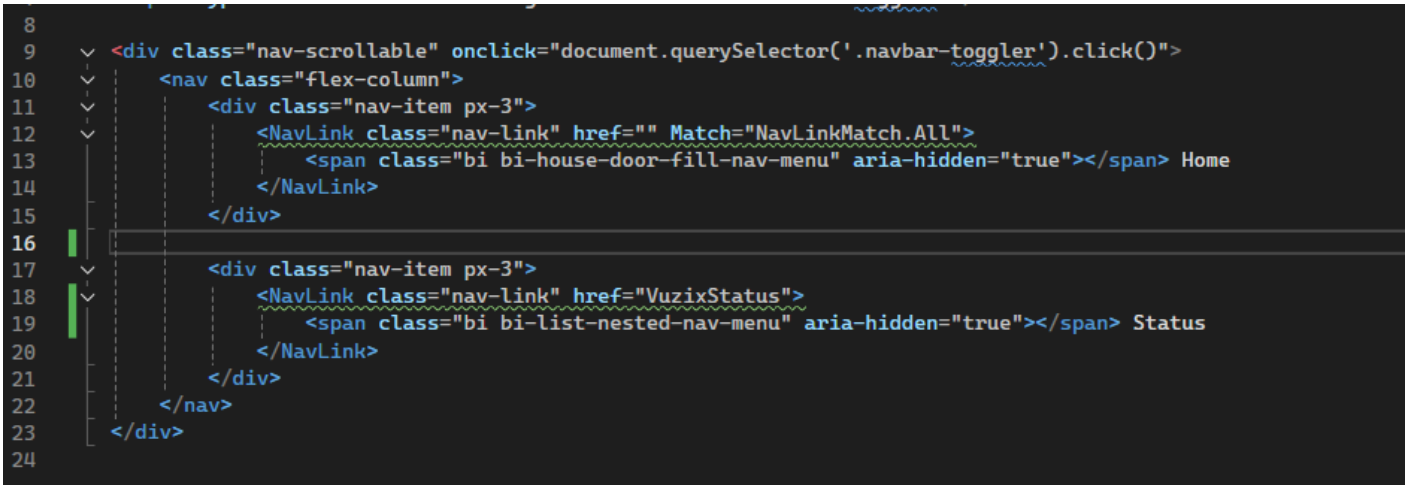

4. Add a link to the page in the navigation, through going in ComponentsLayout/NavMenu.razor.

Just copy paste an existing element and adjust the link and the text

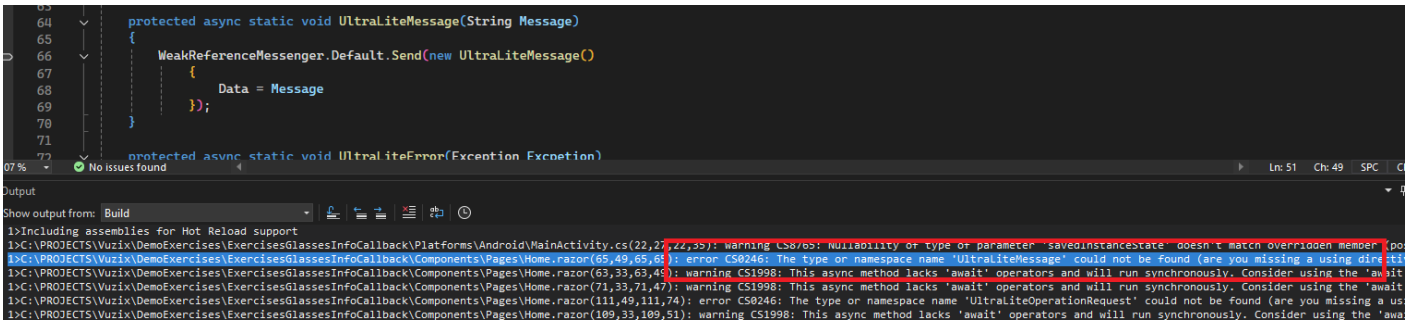
```
<div class="nav-item px-3">
    <NavLink class="nav-link" href="VuzixStatus">
        <span class="bi bi-list-nested-nav-menu" aria-hidden="true"></span> Status
    </NavLink>
</div>
```




While here, lets also remove the Counter item - we haven't been using it.



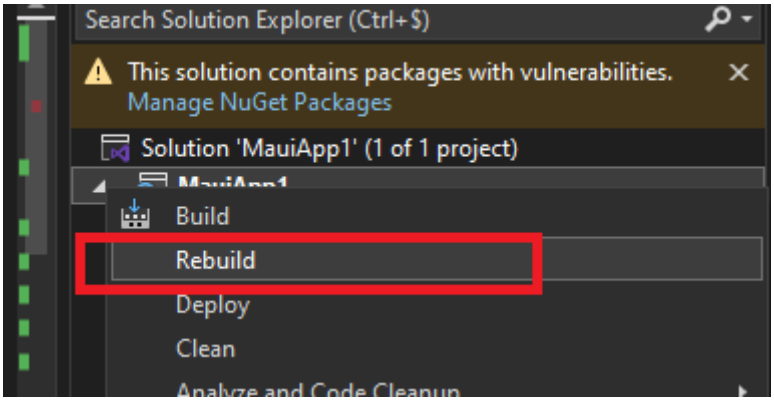
5. Now you could test if everything is in the right place and builds. - in this case we see we have to add the namespaces for the new nuget version.



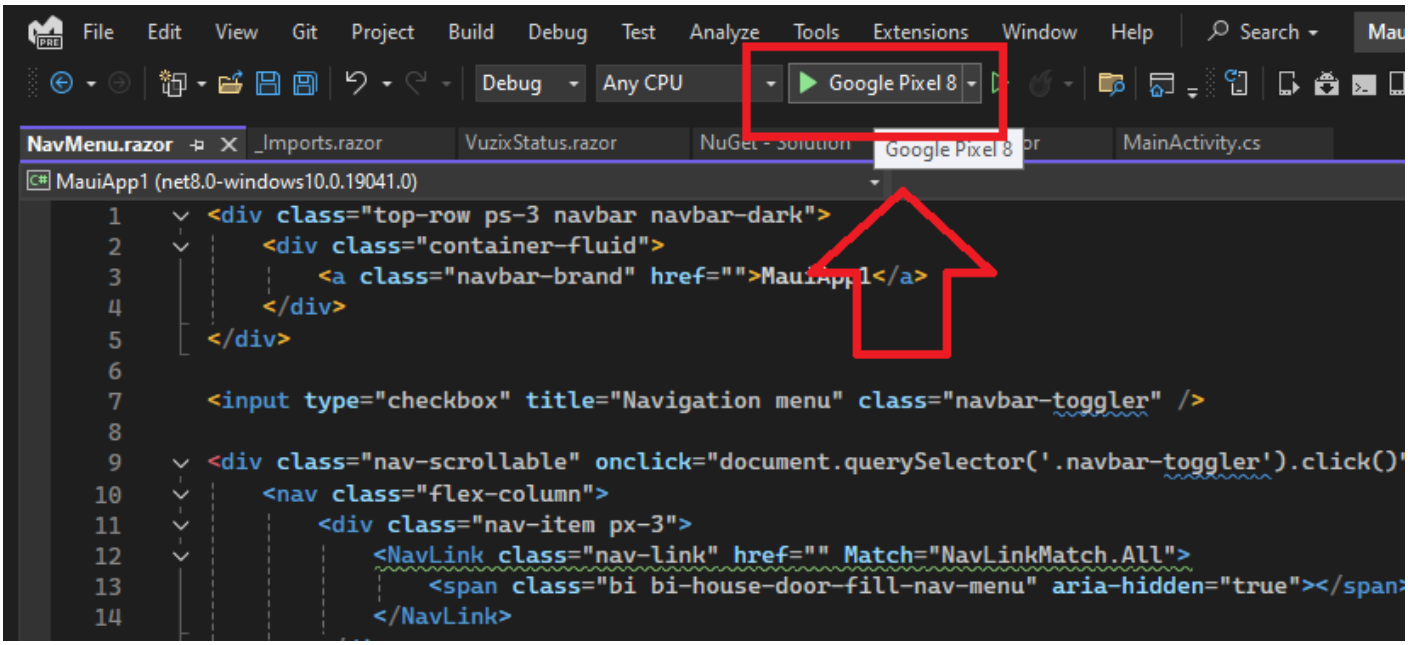

-> So add the includes

```
@page "/"
@using CommunityToolkit.Mvvm.Messaging;
@using VuzixSDK.Class.Event
@using VuzixSDK.Class
@using VuzixSDK.Class.Request
@using VuzixSDK.Enum
```

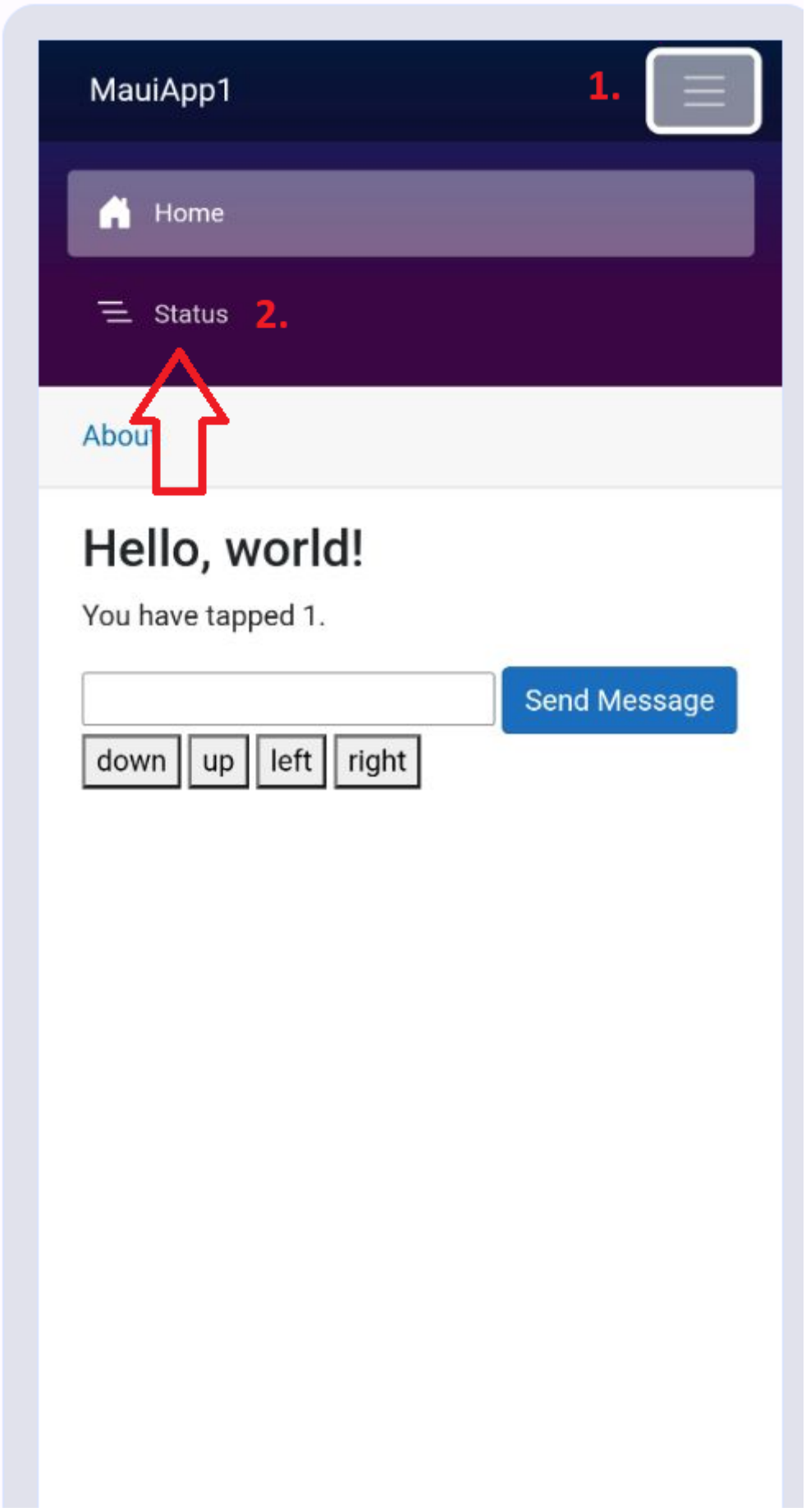
-> Rebuild.



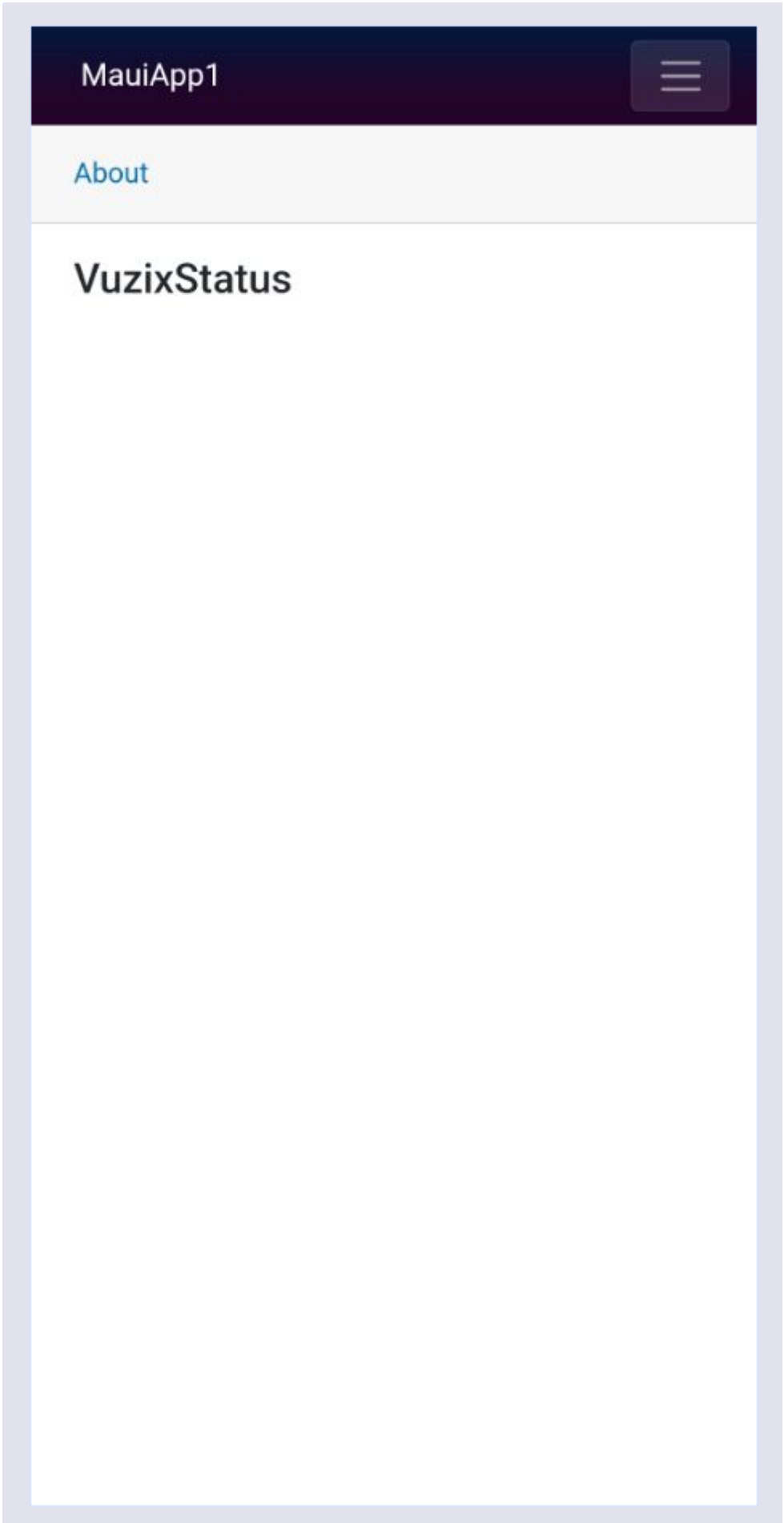

-> Start debug




You should now see the Status menu item if you press the hamburger menu.




And when pressed, see our page/component


MauiApp1
About
VuzixStatus

## Making the two ends meet together (trigger request and read response)

1. Include the namespaces in VuzixStatus.razor (or your name) and the Mvvm namespace

```
@page "/VuzixStatus"
@using VuzixSDK.Class.Event
@using VuzixSDK.Class
@using VuzixSDK.Class.Request
@using VuzixSDK.Enum
@using CommunityToolkit.Mvvm.Messaging;
```

2. Write a button to trigger the function to trigger the function: @onclick denotes it is ran on the server. So we can write this functin in a @code{ } segment

```
<h3>VuzixStatus</h3>
<button @onclick="GetZ100Info">Get Info</button><br />
@code {


    protected void GetZ100Info()
    {
        WeakReferenceMessenger.Default.Send(new UltraLiteStatusRequest());
    }
}
```

3. Declare a variable to hold the information and write the receiver in the OnInitialized function to set the variable in the @code{} segment

```
    UltraLiteStatusResponse Z100Status;
    protected override void OnInitialized()
    {
        WeakReferenceMessenger.Default.Register<UltraLiteStatusResponse>(this, (sender, e) =>
        {
            Z100Status = e;
            StateHasChanged();
        });
    }
```

4. Write the GUI code to display the variable, outside of the @code{} segment - the @ denotes server code. So we check the variable on the server and then render the GUI if not empty. (unset)

```
@if(Z100Status != null)
{
   <table>
      <tr>
         <td>Name</td>
         <td>@Z100Status.Name</td>
      </tr>
      <tr>
         <td>isAvailable</td>
         <td>@Z100Status.isAvailable</td>
      </tr>
      <tr>
         <td>isCharging</td>
         <td>@Z100Status.isCharging</td>
      </tr>
      <tr>
         <td>BatteryLevel</td>
         <td>@Z100Status.BatteryLevel %</td>
      </tr>
      <tr>
         <td>isConnected</td>
         <td>@Z100Status.isConnected</td>
      </tr>
      <tr>
         <td>isControlled</td>
         <td>@Z100Status.isControlled</td>
      </tr>       <tr>
         <td>isControlledByMe</td>
         <td>@Z100Status.isControlledByMe</td>
      </tr>
      <tr>
         <td>isLinked</td>
         <td>@Z100Status.isLinked</td>
      </tr>
      <tr>
         <td>isConnected</td>
         <td>@Z100Status.isConnected</td>
      </tr>
   </table>
```

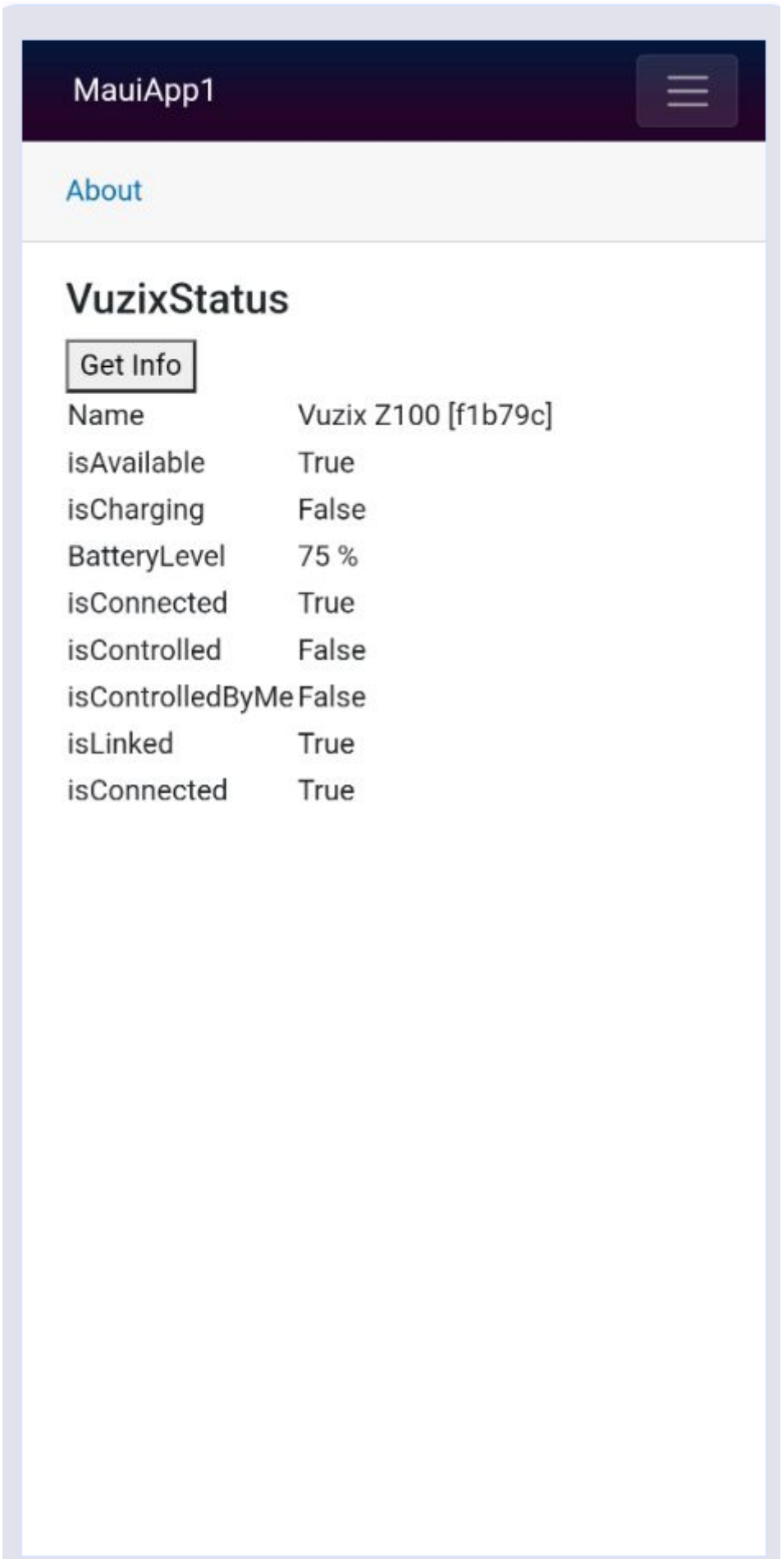
```
}
```

Your full razor page would look something as this:

```
@page "/VuzixStatus"
@using CommunityToolkit.Mvvm.Messaging;
@using VuzixSDK.Class.Event
@using VuzixSDK.Class
@using VuzixSDK.Class.Request
@using VuzixSDK.Enum

<h3>VuzixStatus</h3>
<button @onclick="GetZ100Info">Get Info</button><br />
@code {
    protected UltraLiteStatusResponse Z100Status;
    protected override void OnInitialized()
    {
        WeakReferenceMessenger.Default.Register<UltraLiteStatusResponse>(this, (sender, e) =>
        {
            Z100Status = e;
            StateHasChanged();
        });
    }
    protected void GetZ100Info()
    {
        WeakReferenceMessenger.Default.Send(new UltraLiteStatusRequest());
    }
}



@if(Z100Status != null)
{
    <table>
        <tr>
            <td>Name</td>
            <td>@Z100Status.Name</td>
        </tr>
        <tr>
            <td>isAvailable</td>
```

```
        <td>@Z100Status.isAvailable</td>
      </tr>
      <tr>
        <td>isCharging</td>
        <td>@Z100Status.isCharging</td>
      </tr>
      <tr>
        <td>BatteryLevel</td>
        <td>@Z100Status.BatteryLevel %</td>
      </tr>
      <tr>
        <td>isConnected</td>
        <td>@Z100Status.isConnected</td>
      </tr>
      <tr>
        <td>isControlled</td>
        <td>@Z100Status.isControlled</td>
      </tr>       <tr>
        <td>isControlledByMe</td>
        <td>@Z100Status.isControlledByMe</td>
      </tr>
      <tr>
        <td>isLinked</td>
        <td>@Z100Status.isLinked</td>
      </tr>
      <tr>
        <td>isConnected</td>
        <td>@Z100Status.isConnected</td>
      </tr>
    </table>

}
```

Run the application,  And select the status in the navigation. Press the button.


MauiApp1
About
VuzixStatus
Get Info
Name Vuzix Z100 [f1b79c]
isAvailable True
isCharging False
BatteryLevel 75 %
isConnected True
isControlled False
isControlledByMe False
isLinked True
isConnected True

Resulting project: ExerciseGlassesInfoCallBack.zip

---

Revision #2
Created 11 October 2024 18:52:42 by Tim
Updated 11 October 2024 20:05:42 by Tim