**# Blazor Area Selector**

**## Requirements**

.net core SDK 3.1

to add Blazor client-side template (optional):

```bash

dotnet new -i Microsoft.AspNetCore.Blazor.Templates::3.1.0-preview4.19579.2

```

there are two projects:

- NotWorking/BlazorAreaSelectionApp (contains Blazor server side, not working)

- BlazorCSAreaSelector (contains Blazor client side, is working)

**## Usage**

select BlazorCSAreaSelector.Server as startup project and run it

**## Code Challenge Analysis**

With Blazor server side i got stuck on how to refresh the image coming from the server. maybe is it possible with JSinterop? this was my first choise because is production ready.

Anyway, with Blazor client side is working (more or less). I'm assuming that the server is a box that is doing "magics tricks", indeed all my effort was front-end focused.

So, first I need to find a solution that allows users to draw shapes in the browser. I've started exploring WebGL, Canvas and SVG.

The proposed solution use a library to work with canvas [BlazorExtensions/Canvas](https://github.com/BlazorExtensions/Canvas). I haven't found a reliable way to manage canvas inside blazor.

The component for area selection is taking one parameter in input (filename - malaria.png)

and is configured as:

```html

<div>

<canvasComponent/>

</div>

```

the div allow us to bind necessary events to makes things works and is taking care of the image background also (One-Way binding).

Both elements are locked to 960px X 960px, I didn't explore a way to define a container size at run time (for example based on image properties).

As soon as the component is rendered the offset(X-Y) is calculated (jsInterop.js), I'm calling these two functions with JSinterop into OnAfterRenderAsync.

Using other mouse related events, I'm tracking the user defined area and as soon as the button is released the server is called with HttpClient.

Now, the server is receiving data and will draw a black rectangular into image and send back the response with the new file name.

The image is refreshed because the div background is bound to myImage variable.

In this case is also possible to instruct the client to draw the image, maybe with data sent into response, after server elaboration.

**## Issues**

There are a lot of glitches into UI, especially if the user start drawing the rectangular and move the mouse outside the container, or if the mouse is moved chaotically (users do all this kind of stuff into UI)

The loading message use the logic of myImage variable, in this case the user is not fully aware if the client is waiting data. Unfortunately, I haven't found a better solution to handle this.

Concurrent work at the same image is not handled, this solution is not able to handle this case.

No Unit test nor Integration test are present.