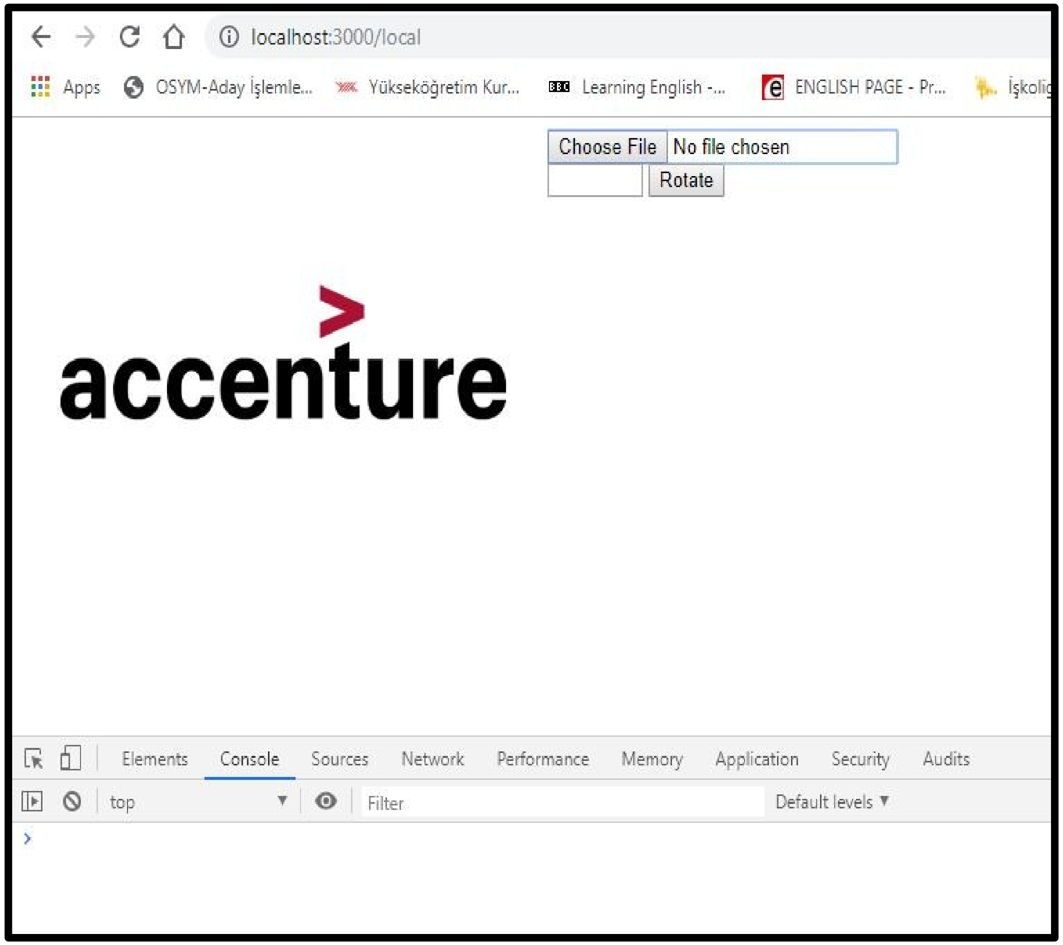
Accenture - Rotation Task

Ahmet Alp

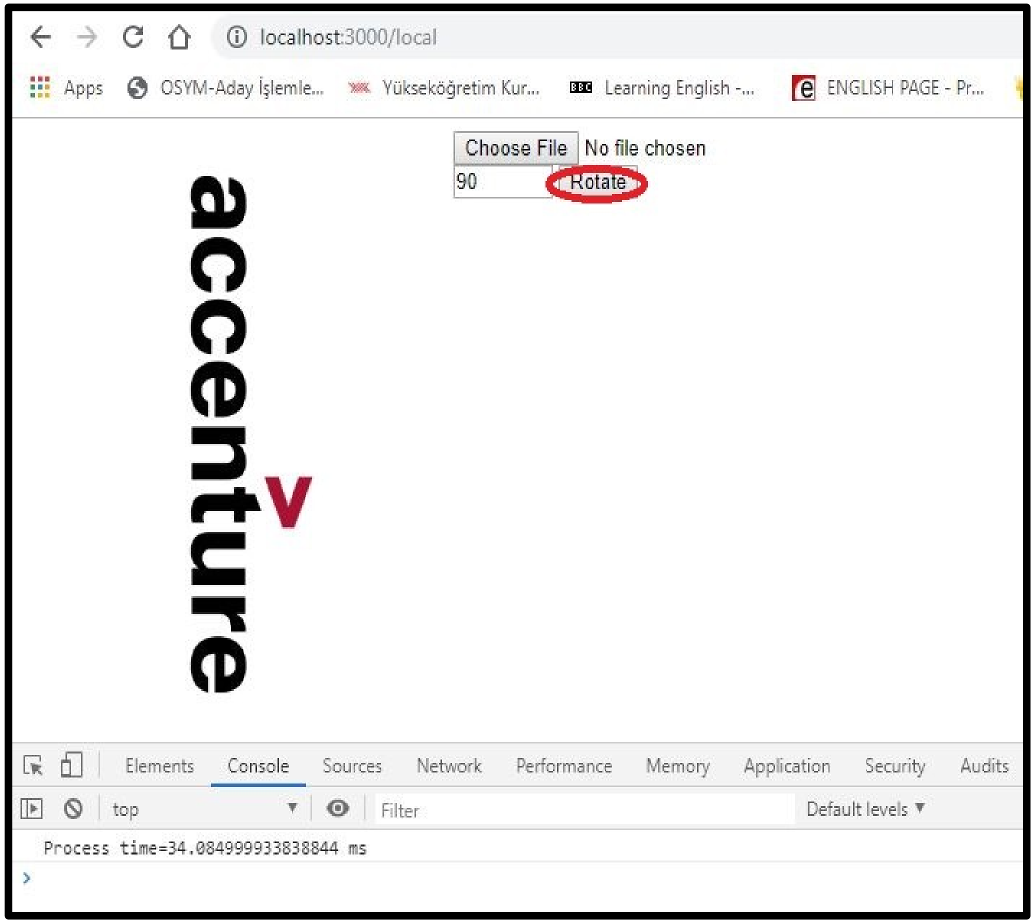
Rotation Task - Usage

- **Run on Node.js**
  - http://localhost:3000/ for rotate function in server(Node.js) side
    - Image rotation time written in Node.js console with console.log()
  - http://localhost:3000/local for rotate function in client(html) side
    - Image rotation time written in browser console with console.log()
- **Run configuration**
  - npm run dev
  - For unit testing: npm run test

Rotation Task - Usage



Before



```
Process time=34.084999933838844 ms
```

After

Rotation Task - Approach

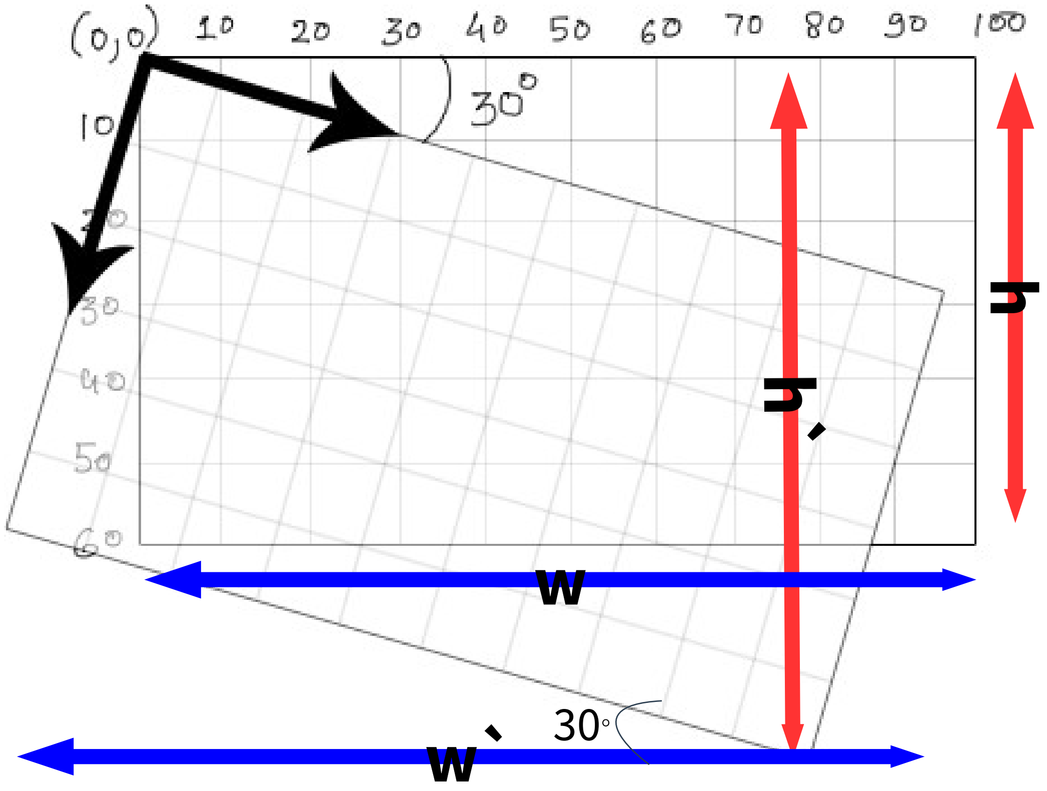
- **Load image on File Upload**
  - Create context from file and draw
  - Save uploaded ImageData in global variable
  - Pre calculate maxWidth and maxHight
    - 45 degree gives maxWitdh and maxHight
- **Rotate Image**
  - Get angle from inbox
  - Call rotate(image: ImageData, angle: double) function

Rotation Task - Algorithm

- **Calculate the Size of a Rotated Image**
  - For 180 and 360 degrees, size is same
  - For 90 and 270 degrees, width and height are reversed
  - For acute angles
    - new-width=$w \times \cos\Theta + h \times \sin\Theta$
    - new-height=$w \times \sin\Theta + h \times \cos\Theta$
  - For obtuse angles
    - new-width=$h \times \cos\Theta + w \times \sin\Theta$
    - new-height=$h \times \sin\Theta + w \times \cos\Theta$

Rotation Task - Algorithm

- **Calculate the Size of a Rotated Image**



For acute angle

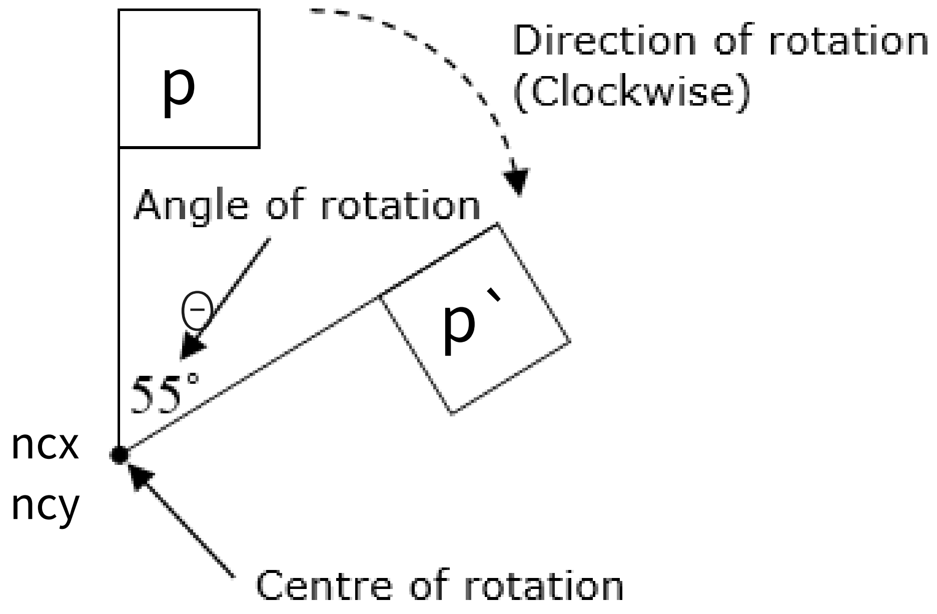
- w`=w x cosΘ + h x sinΘ
- h`=w x sinΘ + h x cosΘ

For obtuse angle

- h`=w x cosΘ + h x sinΘ
- w`=w x sinΘ + h x cosΘ

Rotation Task - Algorithm

- **Rotation Point Calculation**



- p`.x = xx * cos(Θ) - yy * sin(Θ) + ncx
- p`.y = xx * sin(Θ) + yy * cos(Θ) + ncy

Rotation Task - Algorithm

- **Rotation Point Calculation**
  - For each point
    - xx = difference between point.x and center point.x
    - yy = difference between point.y and center point.y
    - ncx = rotated image center point.x
    - ncy = rotated image center point.x
    - New point.x = xx * Math.cos(Θ) - yy * Math.sin(Θ) + ncx
    - New point.y = xx * Math.sin(Θ) + yy * Math.cos(Θ) + ncy
    - Copy new point to new array

Rotation Task – Unit Test

- **Test.js**
- **Mocha and jsdom are used**
- **Run with**
  - $npm run test

```
> nodejs@1.0.0 test c:\nodejs
> mocha

Rotation Task Test
 √ should return width=10 height=5
 √ should return in 1 s (83ms)
 √ should return in 1s(local)
3 passing (97ms)
```

Rotation Task – Statistics

- **On PC**
  - Intel i5 4210
  - 8 GB
  - Windows 10 64 bit
  - Chrome browser
- **324 Kb image**
  - 1045 x 640 pixel
  - 300 DPI
  - Process time average = 55 ms

Rotate Task - References

- **Rotation Border Size Calculation**
  - https://iiif.io/api/annex/notes/rotation/
- **Rotation Point Calculation**
  - https://medium.com/possible-cee/geometry-done-right-with-js-16706b33e88

Thank you