Canvas Draw Lines

Zeichnet 3 unterschiedliche Linien. Zeichnet bei Veränderung der Bildschirmgröße die Linien neu

```
(function(){

    let joa = document.querySelectorAll('.job-offer-line');
    if(joa != undefined && joa != null){
        joa.forEach((jo) => drawJobOfferLine(jo));
    }

    let ola = document.querySelectorAll('.one-line-line');
    if(ola != undefined && ola != null) {
        ola.forEach((ol) => drawOneLineLine(ol));
    }

    let dia = document.querySelectorAll('.pictureDivider-line');
    if(dia != undefined && dia != null) {
        dia.forEach((di) => drawDividerLineLine(di));
    }

    function drawJobOfferLine(jo) {
        let cvWidth = jo.offsetWidth;
        let fWidth = jo.parentElement.parentElement.offsetWidth;
        let cvHeight = jo.parentElement.offsetHeight;
        jo.width = cvWidth;
        jo.height = cvHeight;
        let cx = jo.getContext('2d');


        cx.lineWidth = 5;
        cx.strokeStyle = '#1B6895';

        cx.beginPath();
        if(fWidth >= 1000){
            cx.moveTo(0, 0);
            cx.lineTo(cvWidth, 0);
            cx.lineTo(cvWidth * 0.75, cvHeight);
            cx.lineTo(0, cvHeight);
        }
        else {
            cx.moveTo(0, 0);
            cx.lineTo(cvWidth * 0.75, 0);
```

```
        cx.lineTo(cvWidth * 0.5, cvHeight);
        cx.lineTo(0, cvHeight);
    }
    cx.stroke();
}

function drawOneLineLine(ol){
    let cvWidth = ol.parentElement.offsetWidth;
    let fWidth = ol.parentElement.parentElement.offsetWidth;
    let cvHeight = ol.parentElement.offsetHeight;
    ol.width = fWidth;
    ol.height = cvHeight;
    let cx = ol.getContext('2d');

    cx.clearRect(0, 0, ol.width, ol.height);

    cx.lineWidth = 5;
    cx.strokeStyle = '#1B6895';

    cx.beginPath();
    if(fWidth >= 1000){
        cx.moveTo(fWidth, 0);
        cx.lineTo(cvWidth * 1.85, 0);
        cx.lineTo(cvWidth * 1.6, cvHeight);
        cx.lineTo(0, cvHeight);
    }
    else {
        cx.moveTo(fWidth, 0);
        cx.lineTo(cvWidth * 0.75, 0);
        cx.lineTo(cvWidth * 0.5, cvHeight);
        cx.lineTo(0, cvHeight);
    }
    cx.stroke();
}

function drawDividerLineLine(di){
    let cvWidth = di.parentElement.offsetWidth;
    let cvHeight = di.parentElement.offsetWidth * 0.1;
    di.width = cvWidth;
    di.height = cvHeight;
    let cx = di.getContext('2d');


    cx.lineWidth = 5;
    cx.strokeStyle = '#1B6895';
```

```
        cx.beginPath();
        cx.moveTo(cvWidth, 0);
        cx.lineTo(cvWidth * 0.66 - 10, 0);
        cx.lineTo(cvWidth * 0.33 - 10, cvHeight - 10);
        cx.lineTo(0, cvHeight - 10);
        cx.stroke();

        cx.beginPath();
        cx.moveTo(cvWidth, 10);
        cx.lineTo(cvWidth * 0.66, 10);
        cx.lineTo(cvWidth * 0.33, cvHeight);
        cx.lineTo(0, cvHeight);
        cx.stroke();
    }

    window.addEventListener("resize", (e) => {
        joa = document.querySelectorAll('.job-offer-line');
        if(joa != undefined && joa != null) {
            joa.forEach((jo) => drawJobOfferLine(jo));
        }

        ola = document.querySelectorAll('.one-line-line');
        if(ola != undefined && ola != null) {
            ola.forEach((ol) => drawOneLineLine(ol));
        }

        dia = document.querySelectorAll('.pictureDivider-line');
        if(dia != undefined && dia != null) {
            dia.forEach((di) => drawDividerLineLine(di));
        }
    });
})()
```

Revision #1
Created 2024-02-19 14:42:01 UTC
Updated 2024-02-19 14:47:37 UTC