Wycinanie – clip

Metoda `clip()` sprawia, że aktualnie rysowana ścieżka staje się maską, która przycina dany obrazek do jej krawędzi.

```
<canvas width="250" height="300" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    // Tworzymy podstawowy obszar, z którego będziemy wycinać
    ctx.fillStyle = '#ffd700'
    ctx.arc(canvas.width/2, canvas.height/2, 100, 0, 2*Math.PI)
    ctx.fill()

    // Tworzymy wycinany obszar
    ctx.clip()
    ctx.fillStyle = '#000'
    ctx.beginPath()
    ctx.arc(canvas.width/2+50, canvas.height/2-70, 100, 0, 2*Math.PI)
    ctx.fill()
</script>
```

Maska stworzona za pomocą clip() działa na wszystko co po niej stworzymy. Jeśli chcemy wyłączyć coś z działania metody, musimy skorzystać z zapisu i przywrócenia stanu.

Zobaczmy działanie kodu bez zastosowania `save()` oraz `restore()`:

```
<canvas width="450" height="300" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    // Tworzymy podstawowy obszar, z którego będziemy wycinać
    ctx.fillStyle = '#ffd700'
    ctx.arc(canvas.width/2, canvas.height/2, 100, 0, 2*Math.PI)
    ctx.fill()

    // Tworzymy wycinany obszar
    ctx.clip()
    ctx.fillStyle = '#000'
    ctx.beginPath()
    ctx.arc(canvas.width/2+50, canvas.height/2-70, 100, 0, 2*Math.PI)
    ctx.fill()

    // Tworzymy tekst
    ctx.textAlign = "center"
    ctx.textBaseline = "middle"
    ctx.font = `bold 100px Arial, sans-serif`
    ctx.fillStyle = "rgba(255, 0, 0, 0.6)"
    ctx.fillText("CANVAS", canvas.width/2, canvas.height/2)
</script>
```

Teraz wykorzystajmy metody `save()` i `restore()`:

```
<canvas width="450" height="300" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    // Tworzymy podstawowy obszar, z którego będziemy wycinać
    ctx.fillStyle = '#ffd700'
    ctx.arc(canvas.width/2, canvas.height/2, 100, 0, 2*Math.PI)
    ctx.fill()

    ctx.save()

    // Tworzymy wycinany obszar
    ctx.clip()
    ctx.fillStyle = '#000'
    ctx.beginPath()
    ctx.arc(canvas.width/2+50, canvas.height/2-70, 100, 0, 2*Math.PI)
    ctx.fill()

    ctx.restore()

    // Tworzymy tekst
    ctx.textAlign = "center"
    ctx.textBaseline = "middle"
    ctx.font = `bold 100px Arial, sans-serif`
    ctx.fillStyle = "rgba(255, 0, 0, 0.6)"
    ctx.fillText("CANVAS", canvas.width/2, canvas.height/2)
</script>
```

Jak widać, po wykorzystaniu metod udało się uzyskać cały zamierzony efekt. Zachowana została całość napisu.

Kompozycja

Canvas udostępnia dwie właściwości, które umożliwiają określenie sposobu rysowania:

- ✓ `globalAlpha` ustawia ogólną przezroczystość nowo rysowanych figur, definiujemy ją z przedziału 0-1; zamiast tego możemy używać rgba lub hsla
- ✓ `globalCompositeOperation` określa jak mają być nanoszone nowe kształty na płótno

`globalCompositeOperation` może przyjmować następujące wartości:

- ✓ `copy` pokazuje rysowaną figurę oraz usuwa wszystko co się z nią zazębiało
- ✓ `destination-atop` rysuje figurę oraz tą część canvasa, która zazębiała się z rysowaną figurą
- ✓ `destination-in` pozostawia na canvasie tą część figur, które zazębiały się z rysowaną figurą
- ✓ `destination-out` pozostawia na canvasie te części figur, które nie zazębiały się z rysowaną figurą
- ✓ `destination-over` rysuje figurę pod figurami z canvasu
- ✓ `lighter` w miejscach zazębiania się sumuje kolory rysowanej figury i canvasu
- ✓ `source-atop` pokazuje część canvasa która kolidowała z rysowaną figurą nad nią
- ✓ `source-in` rysuje figurę tylko w miejscach gdzie zazębiała się z figurami narysowanymi na canvasie.
- ✓ `source-out` pokazuje rysowaną figurę w miejscach, gdzie canvas był transparenty, w innych miejscach pokazuje przezroczystość
- ✓ `source-over` domyślna wartość, pokazuje rysowaną figurę w miejscu rysowania, źródło zostaje zachowane

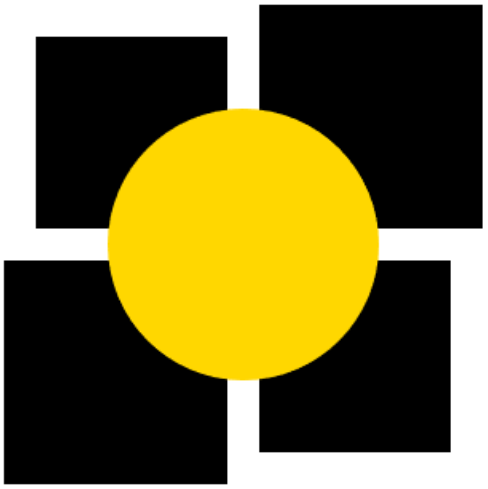
Naszą wyjściową figurą będzie konstrukcja przedstawiona niżej:

```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



Stworzyliśmy cztery kwadraty oraz jedno koło. Za każdym razem zmieniać będziemy tylko i wyłącznie wartość `globalCompositeOperation`.

`source-over` domyślna wartość, pokazuje rysowaną figurę w miejscu rysowania, źródło zostaje zachowane

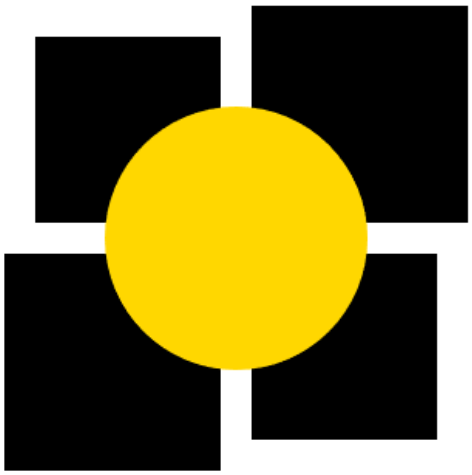
```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "source-over";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



`copy` pokazuje rysowaną figurę oraz usuwa wszystko co się z nią zazębiało

```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "copy";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```

`destination-atop` rysuje figurę oraz tą część canvasa, która zazębiała się z rysowaną figurą

```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "destination-atop";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```

`destination-in` pozostawia na canvasie tą część figur, które zazębiały się z rysowaną figurą

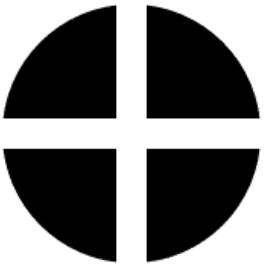
```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "destination-in";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



`destination-out` pozostawia na canvasie te części figur, które nie zazębiały się z rysowaną figurą

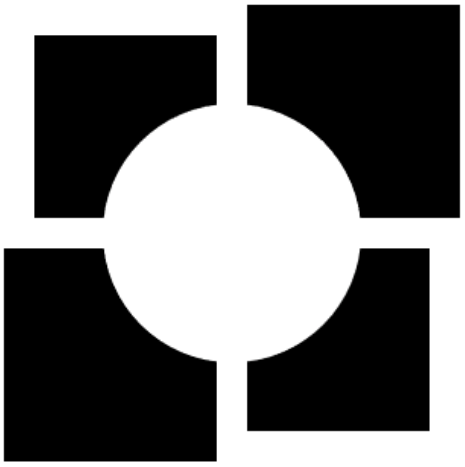
```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "destination-out";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



`destination-over` rysuje figurę pod figurami z canvasu

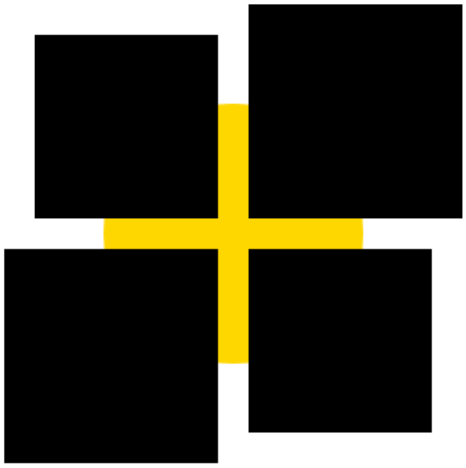
```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "destination-over";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



`lighter` w miejscach zazębiania się sumuje kolory rysowanej figury i canvasu

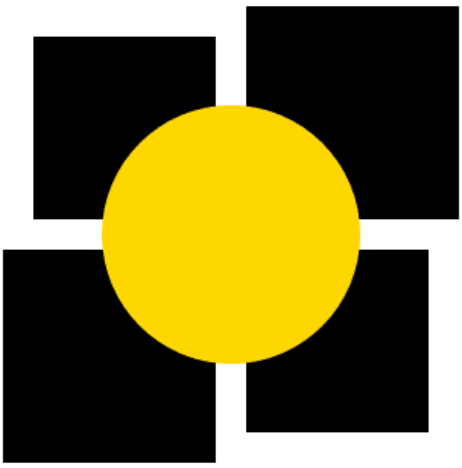
```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "lighter";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



Właściwość lighter dla czarnego koloru nie daje pożądanego efektu. Spróbujmy trochę rozjaśnić kolor naszych kwadratów.

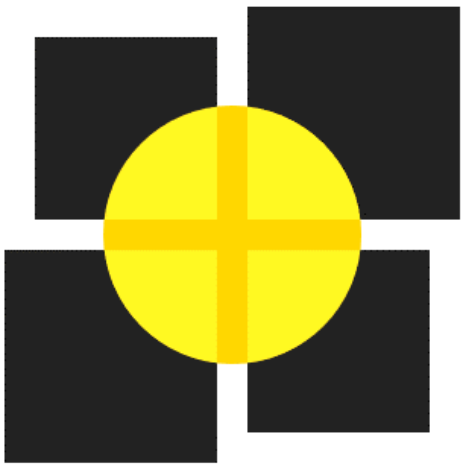
```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#222"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "lighter";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



source-atop pokazuje część canvasa która kolidowała z rysowaną figurą nad nią

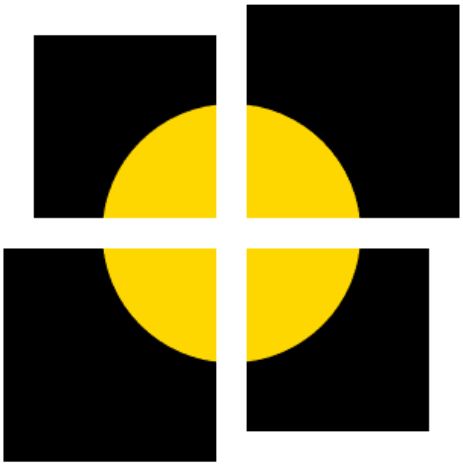
```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "source-atop";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```



source-in rysuje figurę tylko w miejscach gdzie zazębiała się z figurami narysowanymi na canvasie

```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "source-in";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```


source-out pokazuje rysowaną figurę w miejscach, gdzie canvas był transparenty, w innych miejscach pokazuje przezroczystość

```
<canvas width="450" height="400" id="canvas"></canvas>

<script>
    const canvas = document.querySelector("canvas");
    const ctx = canvas.getContext("2d");

    ctx.fillStyle = "#000"
    ctx.fillRect(60, 60, 120, 120)
    ctx.fillRect(200, 40, 140, 140)
    ctx.fillRect(40, 200, 140, 140)
    ctx.fillRect(200, 200, 120, 120)

    ctx.globalCompositeOperation = "source-out";

    ctx.fillStyle = "gold";
    ctx.arc(190, 190, 85, 0, 2*Math.PI)
    ctx.fill()
</script>
```

ĆWICZENIE 3 – cień, clip, kompozycja

Utwórz plik o nazwie imie_nazwisko_cw3.html. Utwórz prostą stronę zawierającą element canvas o dowolnych wymiarach. Korzystając ze znanych Ci wiadomości spróbuj zaprojektować proste logo (nie musi być z niczym konkretnie związane). Wykorzystaj wycinanie za pomocą metody clip(), tekst, cień, kompozycję, zapisywanie i odzyskiwanie stanu. Gotowy plik zamieść na moodle.