Grafika – canvas (cz. 1)

Czym właściwie jest canvas

Canvas (inaczej płótno) to element HTML5 przypominający obszar roboczy programów graficznych, umożliwiający tworzenie grafik z wykorzystaniem JavaScript. Zachowuje się jak img, pozwala na wgrywanie grafik, rysowanie kształtów, pisanie po nim itp.

Element canvas ma następującą strukturę:

```
<canvas width="400" height="200" id="canvas">
</canvas>
```

Możemy podać wymiary: szerokość i wysokość elementu oraz identyfikator, na podstawie którego uzyskamy do niego dostęp.

Odwołanie się do canvasu

Na początku należy pobrać zawartość elementu. Do tego celu służy funkcja `getContext("2d")`. Oznacza to, że mamy do czynienia z trybem rysowania w dwóch wymiarach.

```
const canvas = document.querySelector("#canvas");
const ctx = canvas.getContext("2d");
```

Metoda querySelector jest już nam znana – pobieramy element o id canvas (można to również uczynić z wykorzystaniem getElementById). Następnie do zmiennej ctx przypisujemy zawartość elementu pobraną za pomocą metody getContext.

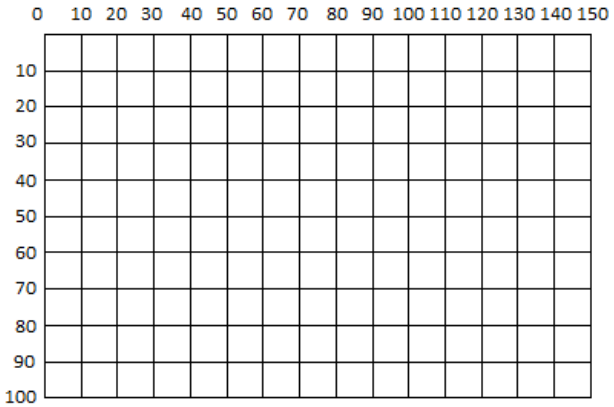
Rysowanie ścieżek

Rysowanie ścieżek wykonujemy w kilku krokach:

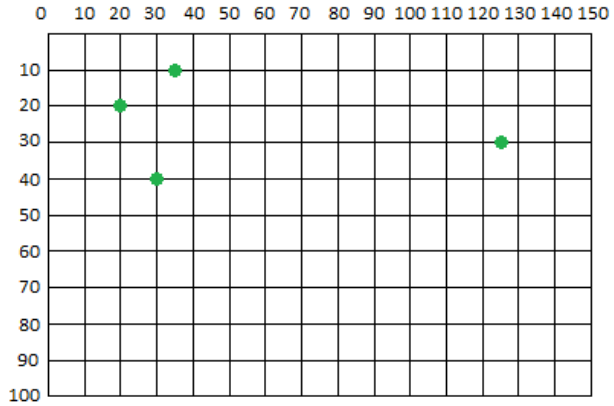
1. Rozpoczynamy rysowanie ścieżki za pomocą metody `beginPath()`
2. Używamy metod do rysowania ścieżki (np. `lineTo(x, y)`)
3. Obrysowujemy `stroke()` lub wypełniamy `fill()` naszą ścieżkę

Jeżeli chcemy zacząć rysować kolejną ścieżkę, poprzednią obrysowujemy/wypełniamy i zaczynamy rysować kolejny path.

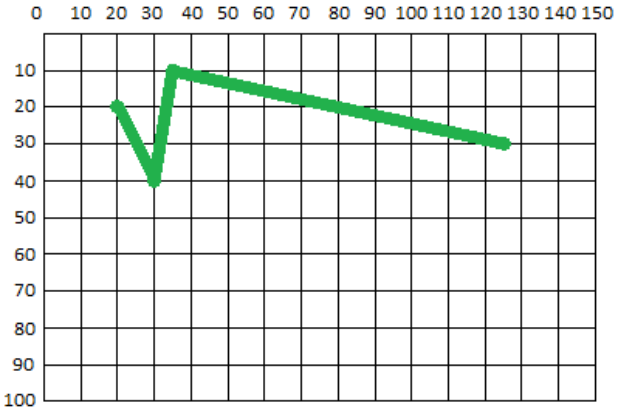
Jak łatwiej zrozumieć działanie canvasa? Spróbuj wyobrazić sobie układ współrzędnych z narysowaną siatką co 10 punktów:



Przypuśćmy, że chcemy narysować łamaną zamkniętą – jej rysowanie chcemy rozpocząć w punkcie (20,20). Należy najpierw umieścić długopis w tym punkcie. Później chcemy przejść do kolejnego punktu (30,40), następnie (35,10), a na końcu (125,30). Na rysunku będzie wyglądać to tak:



Następnie musimy dokończyć rysowanie – wystarczy połączyć punkty:



Bardzo podobnie działa canvas – poniżej przedstawiłam fragment kodu, który rysuje dokładnie tą samą łamaną, ale w JS:

```
ctx.beginPath();
ctx.moveTo(20, 20); //stawiamy piórko w punkcie x: 20 y: 20
ctx.lineTo(30, 40); //zaczynamy rysować niewidzialną linię do x : 30, y: 40
ctx.lineTo(35, 10); //kolejna linia
ctx.lineTo(125, 30); //i kolejna
ctx.stroke(); //po zakończeniu rysowania obrysowujemy linię
```

Jeżeli dodatkowo chcielibyśmy połączyć początkowy i końcowy punkt ścieżki, możemy skorzystać z funkcji `closePath()`, która zamyka całą ścieżkę.

ĆWICZENIE 1 – podstawy canvas

Utwórz plik o nazwie cw1_canvas.html. Utwórz prostą stronę zawierającą dwa elementy canvas o wymiarze 300x200. Korzystając z wymienionych wyżej metod w obu elementach narysuj prostokąt o współrzędnych (30,30), (270, 30), (270, 170), (30, 170). W pierwszym przypadku prostokąt powinien posiadać obrys (stroke), a w drugim wypełnienie (fill). Do zamknięcia figury wykorzystaj closePath().

Gotowy plik zamieść na platformie moodle.

Rysowanie prostokątów

W poprzednim ćwiczeniu rysowaliśmy prostokąty jako osobne kreski. Zamiast rozpoczynać path, rysować ręcznie każdy bok, i zamykać path, możemy też użyć gotowych funkcji, które wykonają powyższe kroki za nas:

- ✓ `fillRect(x, y, width, height)` rysuje wypełniony prostokąt
- ✓ `strokeRect(x, y, width, height)` rysuje obrysowany prostokąt
- ✓ `clearRect(x, y, width, height)` czyści określony obszar

Jak to wygląda w praktyce? Rysowanie wypełnionej figury:

```
ctx.clearRect(0, 0, canvas.width, canvas.height); //czyścimy canvas
ctx.fillRect(x, y, width, height); //wypełniona figura
```

Oraz rysowanie obrysowanej figury:

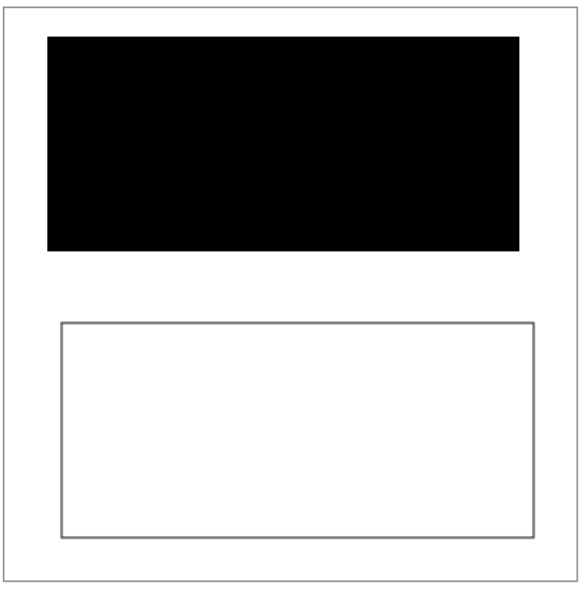
```
ctx.strokeRect(x, y, width, height); //obrysowana figura
```

W obu przypadkach za x, y, width oraz height podajemy nasze własne wymiary – x, y to współrzędne punktu, w którym rozpoczynamy rysowanie, a width oraz height to szerokość i wysokość naszej figury. Funkcja `clearRect(0, 0, canvas.width, canvas.height)` pozwala na wyczyszczenie całego używanego przez nas obszaru (jeśli jest to konieczne do dalszych prac).

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

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

    ctx.fillRect(30, 20, 330, 150); //wypełniona figura
    ctx. strokeRect(40, 220, 330, 150); //obrysowana figura
</script>
```



ĆWICZENIE 2 – rysowanie prostokątów

Utwórz plik o nazwie cw2_canvas.html. Utwórz prostą stronę zawierającą element canvas o dowolnych wymiarach. Korzystając z wymienionych wyżej metod narysuj:

1) wypełniony kwadrat rozpoczynający się w punkcie (x, y), gdzie x to dzień, a y to miesiąc Twojego urodzenia,
2) obrysowany prostokąt znajdujący się poniżej kwadratu.

Wymiary obu figur dowolne (byle mieszczące się w elemencie canvas). Miejsce rozpoczęcia prostokąta również dowolne. Gotowy plik zamieść na platformie moodle.

Kolory w canvas

W canvasie oczywiście mamy możliwość wykorzystywania kolorów – zarówno dla linii, jak i wypełnień. Aby zdefiniować kolor wypełnienia należy dla danego elementu zastosować `fillStyle` oraz podać kolor elementu, przykładowo: `ctx.fillStyle = "#FF0000"`. Oznacza to, że wypełnienie naszego elementu powinno mieć kolor czerwony. Kolor linii możemy określić za pomocą `strokeStyle`, przykładowo `ctx.strokeStyle = "green"`.

Zobaczmy przykład wypełnienia oraz obramowania:

```
<canvas width="500" height="500" id="canvas">
</canvas>

<script>
    const canvas = document.getElementById('canvas');
    const ctx = canvas.getContext('2d');

    ctx.fillStyle = "red";
    ctx.fillRect(30, 20, 300, 100);
    ctx.strokeStyle = "#ff009b";
    ctx.strokeRect(40, 150, 300, 100);
</script>
```

ĆWICZENIE 3 – kolory

Utwórz plik o nazwie cw3_canvas.html. Utwórz prostą stronę zawierającą element canvas o dowolnych wymiarach. Korzystając z wymienionych wyżej metod narysuj kilka figur geometrycznych. Wykorzystaj przedstawione metody do pokolorowania/obrysowania elementów. Gotowy plik zamieść na platformie moodle.