

JS 05 – Canvas - Animacja (14)

Gdyby nie grafika na stronach WWW smartfony z dużymi i pożerającymi energię ekranami byłyby zbędne. Canvas (płótno) pozwala rysować po ekranie za pomocą pikseli, jak w grafice rastrowej. Piksele płótna tworzą układ współrzędnych: lewy górny róg, to punkt o współrzędnych (0,0).

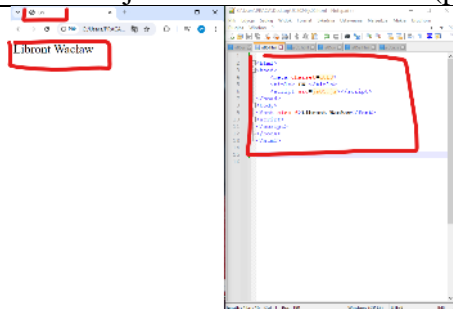
Pamiętaj o tym, by zrzut ekranu DOKUMENTOWAŁ Twoją pracę

Pliki (1)

- W swoim folderze utwórz 2 nowe dokumenty: **js05.html** i **js05.js**
- Otwórz oba dokumenty w notatniku, a dokument HTML w przeglądarce
- Notatnik i okno przeglądarki ustaw po obu stronach ekranu
- Do dokumentu **HTML** wklej tekst z ramki

```
<html>
<head>
  <meta charset=utf8>
  <title> CANVAS </title>
  <script src=js05.js></script>
</head>
<body>
<font size=6>Libront Waclaw</font>
<script>
</script>
</body>
</html>
```

- Zmień tytuł strony **CANVAS** na swoje **inicjały**
- Wpisz swoje nazwisko i imię
- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Canvas (1)

- Do dokumentu **HTML** przed znacznikiem **<script>** `` wpisz tekst `<canvas width=600 height=600 id=CAN></canvas>`
`<canvas>` graficzna ramka na stronie WWW o wymiarach 600x600 pikseli

- Do dokumentu **HTML** pomiędzy znaczniki **<script>** `<script>` wpisz tekst `var C = CAN.getContext("2d");`
`C.fillStyle="blue";`
`C.fillRect(25,25,100,100);`
`C.clearRect(45,45,60,60);`
`C.strokeStyle="red";`
`C.strokeRect(20,20,110,110);`

`var C = CAN.getContext("2d");`

`C.fillStyle(kolor)`

`C.strokeStyle(kolor)`

`C.fillRect(x,y,s,w);`

`C.strokeRect(x,y,s,w);`

obiekt C, za pomocą którego rysujemy w ramce za pomocą JS

kolor wypełnienia

kolor ramki

wypełniony prostokąt, lewy górny róg (x,y) szerokość i wysokość (s,w)

prostokątna ramka

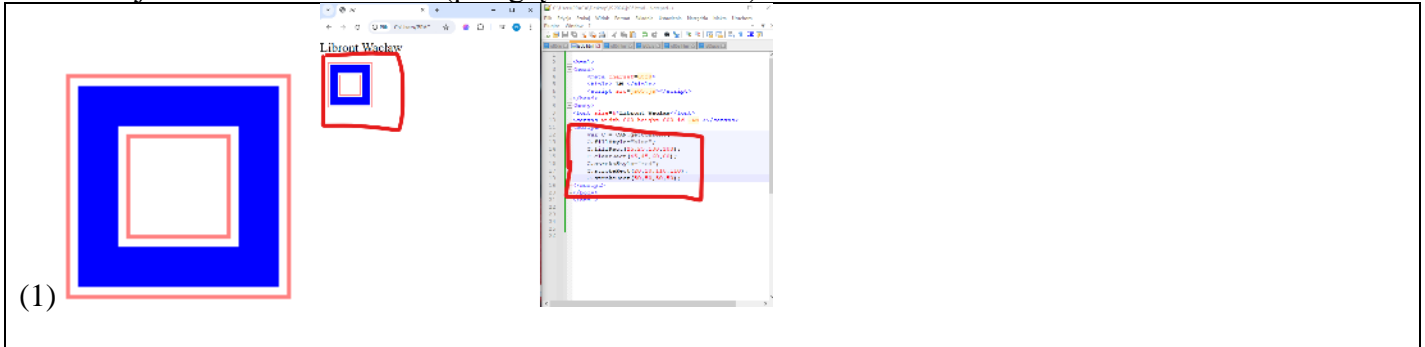
`C.clearRect(x,y,s,w)`

wymazywanie prostokątnego obszaru

UWAGA

- wielkość liter w JS ma znaczenie
- jeżeli pojawią się błędy użyj odpluskwiacza

- (1) Narysuj:
 - kwadrat o czerwonym brzegu
 - lewy górny róg (50,50), szerokość i wysokość po 50 pikseli
- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



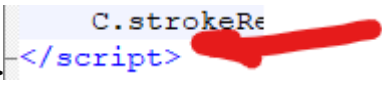
Kolorowe koła (1)

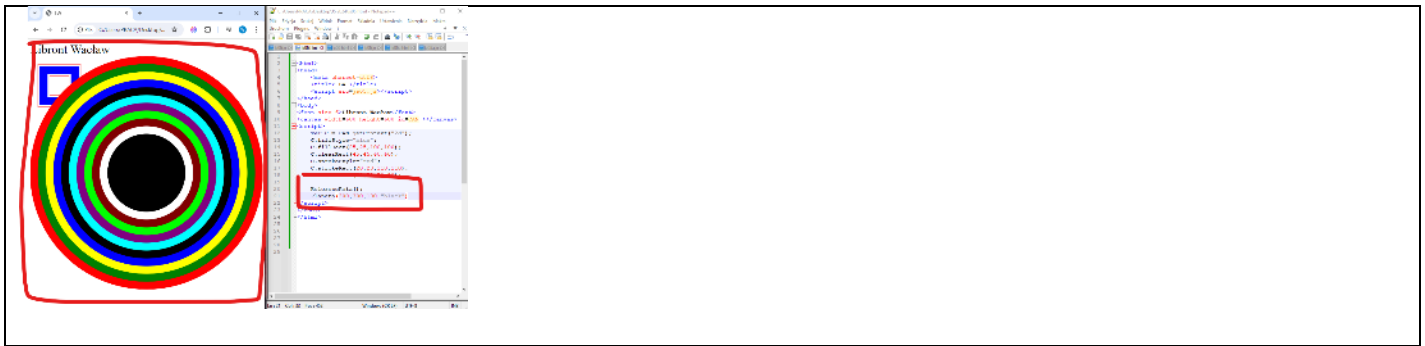
- Do dokumentu JS wklej funkcje z ramki rysujące kolorowe koła

```
function Planeta(x,y,r,k) {
    C.beginPath();
    C.strokeStyle=k;
    C.fillStyle=k;
    C.arc(x,y,r,0,2*Math.PI);
    C.stroke();
    C.fill();
};
function KoloroweKoła() {
    Planeta(300,300,300,"red");
    Planeta(300,300,280,"green");
    Planeta(300,300,260,"yellow");
    Planeta(300,300,240,"blue");
    Planeta(300,300,220,"black");
    Planeta(300,300,200,"aqua");
    Planeta(300,300,180,"purple");
    Planeta(300,300,160,"lime");
    Planeta(300,300,140,"maroon");
    Planeta(300,300,120,"white");
}
```

funkcja `Planeta()` rysuje kolorowe koło o środku w punkcie (x,y) , promieniu r w kolorze k

funkcja `KoloroweKoła()` rysuje kolorowe koła o środku w punkcie $(300,300)$ i promieniach od 300 do 12 co 20 w różnych kolorach

- Do dokumentu HTML przed znacznik `</script>` wklej `KoloroweKoła();`  wpisz tekst
- Zapisz dokumenty i odśwież przeglądarkę
- Za pomocą funkcji `Planeta()` narysuj:
 - czarne koło, o środku w punkcie $(300,300)$ i promieniu 100
- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Animacja - szablon (1)

Na czym polega animacja? Rysujemy obiekt, potem wymazujemy ekran, ustawiamy nowe współrzędne i powtarzamy te trzy instrukcje wiele razy. W JS animację realizujemy rekurencyjnie. Funkcje `clearTimeout` i `setTimeout` odpowiedzialne są za uruchamianie animacji, co określony przedział czasu.

- Do dokumentu JS wklej tekst z ramki
szablon funkcji animacyjnej

```
function animacja() {
  C.clearRect(0,0,w,h);
  C.strokeRect(0,0,w,h);
  KoloroweKoła();
  Planeta(x,300,100,"black");
  x=x+1;
  clearTimeout(czas);
  czas = setTimeout(animacja, skok);
}
```

`clearRect()` i `strokeRect()` wymazujemy canvas i rysujemy ramkę

rysujemy kolorowe koła i osobno czarne koło w środku

`x=x+1`

obliczanie współrzędnej poziomej, w następnej klatce animacji rysunek pojawi się w innym miejscu

`clearTimeout(czas)` czyszczenie zmiennej czas

`setTimeout()`

rekurencyjne uruchamianie animacji co czas milisekund

- Do dokumentu HTML przed znacznik `</script>`  wpisz instrukcje

```
var w=C.canvas.width;
var h=C.canvas.height;
var skok=10;
var czas;
var x=300;
animacja();
```

`w` i `h`

w zmiennych zapamiętujemy szerokość i wysokość obszaru canvas

`skok` i `czas`

zmiennie służą do obsługi animacji za pomocą instrukcji `clearTimeout()` i `setTimeout()`

`var x=300`

zmienna `x` jest współrzędną początkową animowanego obiektu

`animacja()`

funkcja rekurencyjna – wywołuje samą siebie za pomocą `setTimeout()`

funkcja musi być uruchamiana pierwszy raz

- Zapisz dokumenty i odśwież przeglądarkę

czarne koło przesuwa się od środka do prawego brzegu obszaru canvas - zmienna `x` przyjmuje wartości od 300 do ...

ZADANIE

- W dokumencie JS popraw instrukcję `x=x+1;`
 - aby obiekt poruszał się w lewo
 - za każdym razem o 2 piksele
- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik), gdy czarne koło jest z lewej strony



Odbicia X (1) – odbicia od prawego i lewego brzegu

- W dokumencie **HTML**, przed funkcją **animacja()**;

```
var vx=5;
```

deklaracja zmiennej vx z początkową wartością 5
zmienna definiuje szybkość ruchu - o ile pikseli przesuwa się obiekt za każdym obrotem pętli

- W dokumencie **JS**, w funkcji **animacja()**

- popraw instrukcję przesuwania koła z poprzedniego zadania

```
x=x+vx;
```

obiekt przesuwa się zgodnie ze zmienną vx, gdy dodatnie to w prawo, gdy ujemne to w lewo

- W dokumencie **JS**, w funkcji **animacja()**, przed instrukcją **clearTimeout** wpisz dwie instrukcje warunkowe:

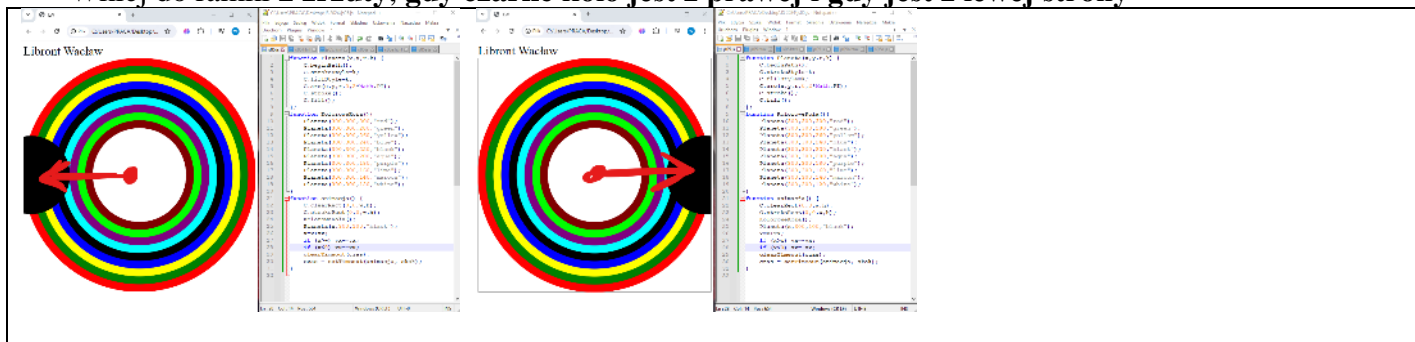
```
if (x>w) vx=-vx;
if (x<0) vx=-vx;
```

„odbijanie” obiektu od brzegów

gdy środek koła osiągnie lewy brzeg ($x < 0$), to zmienia się kierunek na przeciwny, podobnie dla brzegu z prawej strony ($x > w$)

- Zapisz dokumenty i odśwież przeglądarkę

- Wklej do ramki 2 zrzuty, gdy czarne koło jest z prawej i gdy jest z lewej strony



Odbicia Y (1) – odbicia od wszystkich brzegów

- W dokumencie **HTML**, przed funkcją **animacja()**;

- zadeklaruj zmienną y i przypisz jej wartość 300
początkowe położenie koła

- zadeklaruj zmienną vy i przypisz jej wartość 3
początkowa prędkość koła

- W dokumencie **JS** w funkcji **animacja()**

- popraw funkcję **Planeta()**

```
Planeta(x, y, 100, "black");
```

czarne koło może poruszać się w pionie - współrzędna y

- wpisz instrukcję obliczającą ruch w pionie:

```
y=y+vy;
```

ruch w kierunku pionowym

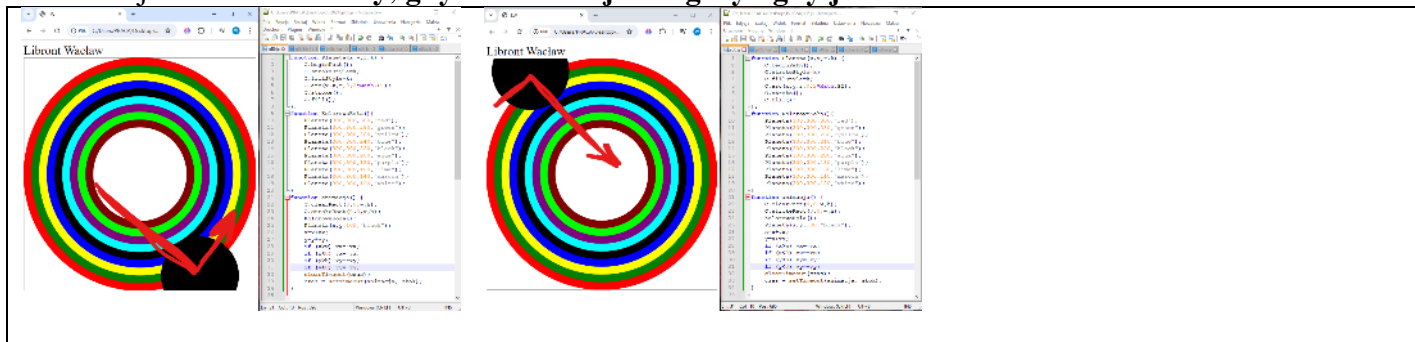
```
var vx=5;
animacja(),
```

- wpisz 2 instrukcje warunkowe odbić:

```
if (y>h) vy=-vy;  
if (y<0) vy=-vy;
```

od dolnego i górnego brzegu

- Wklej do ramki 2 zrzuty, gdy białe koło jest u góry i gdy jest u dołu obszaru canvas



Animacja - Kwadrat (1)

Aby kwadrat się przesuwał wystarczy zmieniać jego współrzędne (x,y).

Aby kwadrat zamieniał wymiary, wystarczy zmieniać jego szerokość i wysokość

Obroty realizujemy za pomocą transformacji

Gdy obiekt dociera do brzegu powinien się „odbić”, tzn. zmienić kierunek ruchu na przeciwny (matematycznie zmienić znak).

Aby odbicia były płynne należy uwzględnić szerokość i wysokość obiektu.

- Usuń **wszystkie instrukcje** pomiędzy znacznikami `<script>...</script>` możesz także „zamknąć” je w komentarzu: `/*...*/` wiele wierszy lub `//` pojedynczy wiersz
- Do dokumentu **HTML**, pomiędzy znaczniki `<script>...</script>` wklej instrukcje

```
var C = CAN.getContext("2d");  
var w=C.canvas.width;  
var h=C.canvas.height;  
var skok=10;  
var czas;  
var x=100;  
var y=100;  
var vx=5;  
var vy=4;  
var kat=0;  
var vkat=5;  
var bok=100;  
var kol="black";  
var b2=bok/2;  
animacja();
```

(x,y) początkowe położenie

(vx,vy) początkowy kierunek ruchu (szybkość)

(kat,vkat) początkowy kąt i szybkość obrotu

b2 połowa boku kwadratu, aby program był przejrzysty

- W dokumencie **JS** usuń całą funkcję **animacja()** lub zmień jej nazwę np. na **animacjaKoła()**
- Do dokumentu **JS** wklej tekst z ramki **definicja obrotowego kwadratu i nowa funkcja animacyjna**

```
function KWA(x,y,bok,kat,kol) {  
  C.save();  
  C.translate(x,y);  
  C.rotate(Math.PI*kat/180);  
  C.fillStyle=kol;  
  C.fillRect(-bok/2,-bok/2,bok,bok);  
  C.restore();  
}  
function animacja() {  
  C.clearRect(0,0,w,h);  
  C.strokeRect(0,0,w,h);  
  KWA(x,y,bok,kat,kol);  
  x=x+vx;  
  y=y+vy;  
  kat=kat+vkat;  
  if (x+b2 > w || x < b2) {vx=-vx;}  
  if (y+b2 > h || y < b2) {vy=-vy;}  
}
```

```

if (kat > 360){kat=kat-360;}
clearTimeout(czas);
czas = setTimeout(animacja, skok);
}

```

Za pomocą `fillRect` i `strokeRect` rysujemy kwadraty od lewego, górnego rogu podając do funkcji (x,y)

Chcemy, aby te współrzędne opisywały środek (np. będzie łatwiej obracać),

Funkcja `KWA` zanim narysuje kwadrat przesuwa współrzędne i obraca je (`translate`, `rotate`)

Instrukcje `save` i `restore` służą do zapisywania i odtwarzania stanu `canvas`

<code>Animacja</code>	przed narysowaniem kolejnej sceny czyścimy obszar
<code>KWA()</code>	definicja obrotowego kwadratu – jego centrum znajduje się w środku
<code>x=x+vx;</code>	przesuwanie – nowe położenie=poprzednie położenie + szybkość
<code>kat=kat+vk;</code>	obracanie – nowy kąt=poprzedni kąt +skok
<code>if (x+b2 > w x < b2){vx=-vx;}</code>	odbijanie środka kwadratu od prawego lub od lewego brzegu
<code>x+b2 > w</code>	środek kwadratu wychodzi za prawy brzeg <code>canvas</code>
<code>x < b2</code>	środek kwadratu wychodzi za lewy brzeg <code>canvas</code>
<code>vx=-vx</code>	zmiana kierunku ruchu – liczba przeciwna

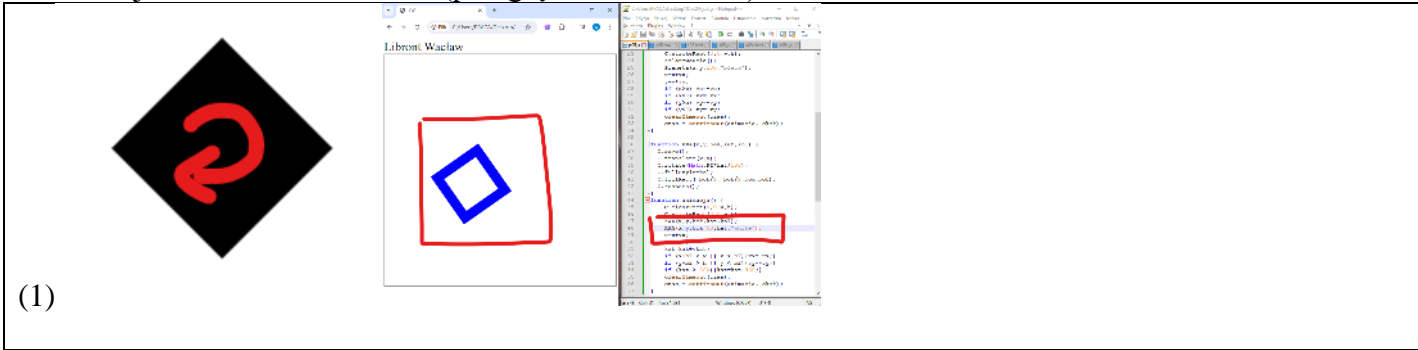
- (1) Zapisz dokumenty i odśwież przeglądarkę
- Zmień kolor kwadratu na **niebieski**
- Zwiększ bok kwadratu do **150**
- W dokumencie **JS**, w funkcji **animacja()**, po instrukcji rysującej kwadrat - dopisz instrukcję rysującą biały kwadrat

```

KWA(x,y,bok-50, kat, "white");

```

- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Dużo kwadratów (1)

Jak sprawić, aby po ekranie „fruwało” 100 kwadratów? Definiować tyle zestawów zmiennych? Z pomocą przychodzą tablice, które są zestawem komórek.

Przygotujemy tablicę na 100 kwadratowych obiektów.

W każdym obiekcie zapamiętamy: $x, y, vx, vy, kat, vk, bok$ i kolor

Początkowy układ kwadratów zostanie rozlosowany.

`var TK=[]` definicja pustej tablicy, jej elementy są numerowane od zera np. `TK[0]`

`var K={}` definicja obiektu - współrzędnych kwadratu, jego elementy mogą być nazwane, np. `K.x`

Jeżeli elementem tablicy `TK` będzie obiekt `K`, to `TK[9].x` oznacza dostęp do współrzędnej x dziesiątego kwadratu

- W dokumencie **HTML**
 - usuń wszystkie instrukcje pomiędzy znacznikami `<script>...</script>`
 - lub weź w komentarz
 - pomiędzy znaczniki `<script>...</script>` wklej instrukcje z ramki

```

var C = CAN.getContext("2d");
var w=C.canvas.width;
var h=C.canvas.height;
var skok=10;
var czas;
var ile=40;
var TK=[];
losujKWA();
animacja();

```

`ile` ile kwadratów jest animowanych
`TK` tablica na parametry kwadratów
`losujKWA` funkcja losująca początkowe parametry

- W dokumencie **JS**
 - usuń funkcję `animacja()`

lub zmień jej nazwę np. na animacjaKwa()

- wklej nowe funkcje z ramki

```
function losowa(p,k) {
  return Math.floor(Math.random()*(k-p+1)+p);
}
function losRGBA(){
  var r = losowa(0,255);
  var g = losowa(0,255);
  var b = losowa(0,255);
  var a = Math.random();
  return "rgb("+r+", "+g+", "+b+", "+a+" )";
}
function losujKWA(){
  for (var i=0;i<ile;i++){
    var K={};
    K.x=losowa(100,500);
    K.y=losowa(100,500);
    K.vx=losowa(-5,5);
    K.vy=losowa(-5,5);
    K.kat=losowa(0,45);
    K.vkat=losowa(-5,5);
    K.bok=losowa(5,100);
    K.kol=losRGBA();
    TK[i]=K;
  }
}
function animacja() {
  C.clearRect(0,0,w,h);
  C.strokeRect(0,0,w,h);
  for (var i=0;i<ile;i++){
    KWA(TK[i].x,TK[i].y,TK[i].bok,TK[i].kat,TK[i].kol);
    TK[i].x=TK[i].x+TK[i].vx;
    TK[i].y=TK[i].y+TK[i].vy;
    TK[i].kat=TK[i].kat+TK[i].vkat;
    var b2=TK[i].bok/2;
    if (TK[i].x+b2 > w || TK[i].x < b2){TK[i].vx=-TK[i].vx;}
    if (TK[i].y+b2 > h || TK[i].y < b2){TK[i].vy=-TK[i].vy;}
    if (TK[i].kat > 360){TK[i].kat=TK[i].kat-360;}
  }
  clearTimeout(czas);
  czas = setTimeout(animacja, skok);
}
```

losowa funkcja losująca liczby z przedziału <p..k>

losRGBA funkcja losująca kolor R-czerwony G-zielony B-niebieski A-przeźroczystość

losujKWA funkcja losująca początkowe parametry kwadratów
pętla FOR, w której „i” zmienia się od 0 do 99

var K={}; definiujemy pusty obiekt K (na parametry)

losujemy do niego wszystkie początkowe parametry

TK[i]=K obiekt K zapisujemy w kolejnej komórce tablicy TK

animacja funkcja animująca kwadraty z tablicy TK – działa w identyczny sposób: czyszczenie, rysowanie, obliczanie
ponieważ musimy analizować wiele kwadratów, dlatego pętla FOR

wszystkie zmienne zastąpiono ich tablicowymi odpowiednikami: x TK[i].x itd.

- (1) Zapisz dokumenty i odśwież przeglądarkę

- W funkcji **animacja()**, **dopisz kolejną instrukcję**

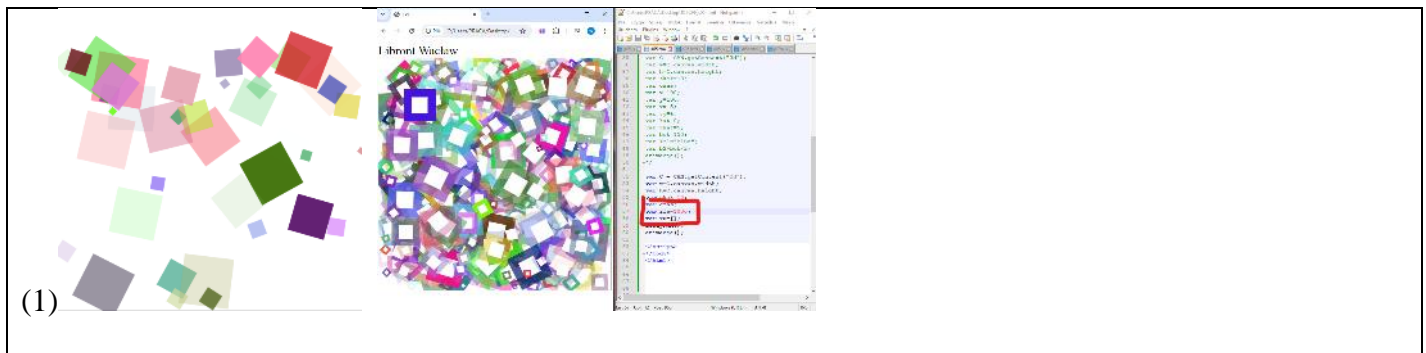
```
KWA(TK[i].x,TK[i].y,TK[i].bok-TK[i].bok/2,TK[i].kat,"white");
```

instrukcja rysuje biały kwadrat w środku kolorowego

- W dokumencie HTML, zmień liczbę kwadratów na **1000**

```
var ile=1000;
```

- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



(1)

Nowe pliki (1)

Jak animować dowolne grafiki? Z pomocą przychodzi instrukcja `drawImage()`

- W swoim folderze utwórz **2 nowe** dokumenty: **js05gr.html** i **js05gr.js**
- Otwórz oba dokumenty w notatniku, a dokument HTML w przeglądarce
- Do dokumentu **HTML** wklej tekst z ramki

```
<html>
<head>
  <meta charset=utf8>
  <title> GRAFIKA </title>
  <script src=js05gr.js></script>
</head>
<body>
  <canvas width=400 height=400 id=GRAFIKA></canvas>
<script>
  var C = GRAFIKA.getContext("2d");
  var w=C.canvas.width;
  var h=C.canvas.height;
  var skok=10;
  var czas;

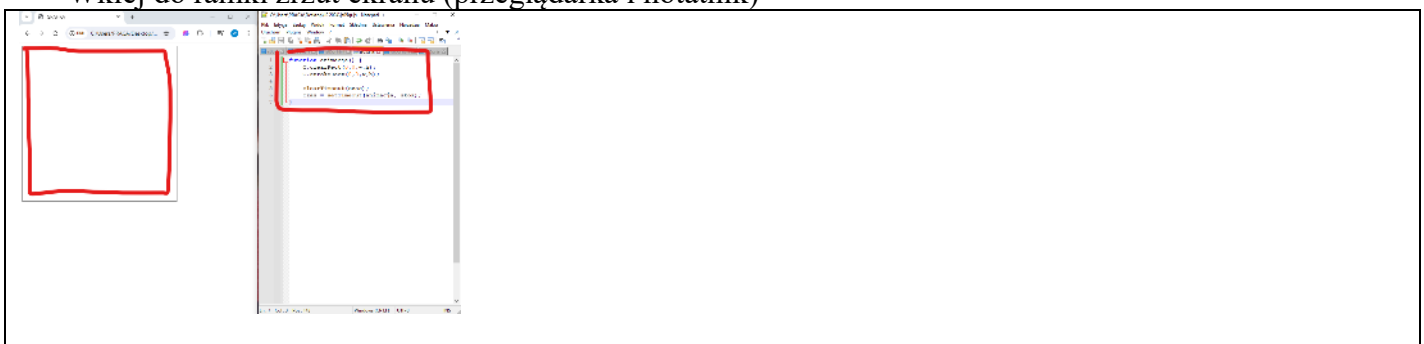
  animacja();
</script>
</body>
</html>
```

- Do dokumentu **JS** wklej funkcję z ramki

```
function animacja() {
  C.clearRect(0,0,w,h);
  C.strokeRect(0,0,w,h);

  clearTimeout(czas);
  czas = setTimeout(animacja, skok);
}
```

- Zapisz dokumenty i odśwież przeglądarkę
ramka otaczająca obszar canvas
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Wczytanie i animacja grafiki (1)

- Pobierz do swojego foldera obrazek: <http://zsobobowa.eu/pliki/program/tarcza.png>


```
var czas;  
animacja();
```

- Do dokumentu **HTML**, przed funkcję **animacja()** wklej tekst z ramki

```
var x=100;  
var y=100;  
var vx=2;  
var vy=3;  
var kat=0;  
var vk=1;  
var img=new Image();  
img.src='tarcza.png';  
img.onload=function () {  
    C.drawImage(img,0,0);  
}  
var WY=100; var SZ=80;
```

x,y,vx,... parametry ruchu jednostajnego prostoliniowego
var img=new Image(); deklaracja zmiennej, do której załadujemy obrazek
img.src='tarcza.png' jaki obrazek jest ładowany do zmiennej *img*
img.onload funkcja ładująca obrazek
S.drawImage wyświetlenie obrazka na canvas

- Do dokumentu **JS**, wklej tekst z ramki

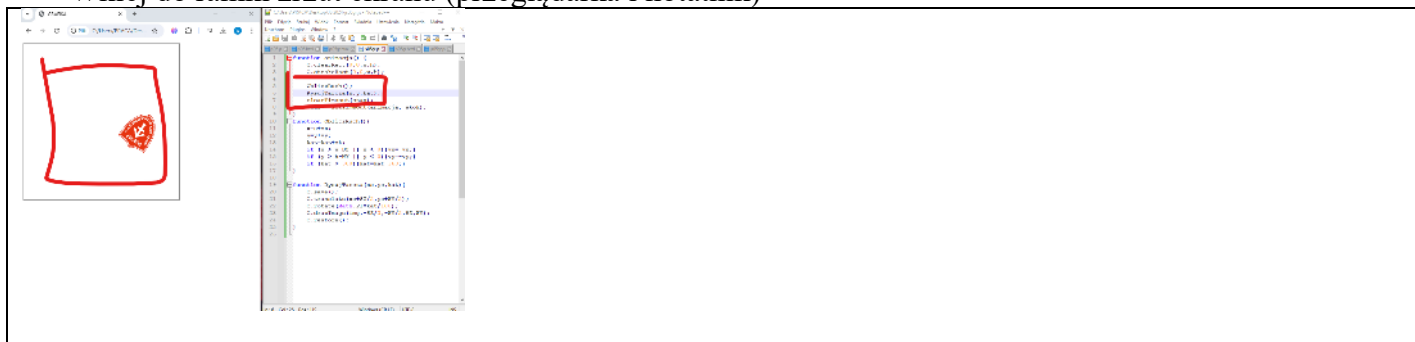
```
function ObliczRuch() {  
    x=x+vx;  
    y=y+vy;  
    kat=kat+vk;  
    if (x > w-SZ || x < 0) {vx=-vx;}  
    if (y > h-WY || y < 0) {vy=-vy;}  
    if (kat > 360) {kat=kat-360;}  
}  
  
function RysujTarcza(xs, ys, kat) {  
    C.save();  
    C.translate(xs+SZ/2, ys+WY/2);  
    C.rotate(Math.PI*kat/180);  
    C.drawImage(img, -SZ/2, -WY/2, SZ, WY);  
    C.restore();  
}
```

funkcja ObliczRuch() oblicza nowe położenie i kąt, a także sprawdza odbicia od brzegów
funkcja RysujTarcza() obraca tarczę o kąt i rysuje uwzględniając środek
funkcja drawImage() rysuje obrazek na canvas
drawImage(img,x,y) lewy górny róg obrazka w punkcie (x,y)
drawImage(img,x,y,sz,wy) ustalamy szerokość i wysokość obrazka
drawImage(img,a,b,s,w,x,y,sz,wy) z obrazka wycinamy obszar od punktu (a,b) o wymiarach (s,w) i wklejamy do obszaru (x,y) i sz,wy

- W dokumencie **JS**, w funkcji **animacja()**, **clearTimeout(czas);** wpisz dwie instrukcje:

```
ObliczRuch();  
RysujTarcza(x,y,kat);
```

- Zapisz dokumenty i odśwież przeglądarkę
szkolna tarcza odbija się od brzegów i obraca
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Animacja poklatkowa (1)

- Pobierz do swojego foldera obrazek: <http://zsobobowa.eu/pliki/program/sportowiec.png>

Wszystkie klatki (o jednakowych szerokościach) znajdują się w jednym obrazku i za pomocą `drawImage` wycinamy z niego odpowiedni obszar

- Do dokumentu **HTML**, przed funkcją `animacja()` `animacja();` wklej tekst z ramki

```
var spo = new Image();
spo.src = 'sportowiec.png';
spo.onload=function () {
    C.drawImage(spo,0,0);
}
var ramka=0;
```

załadowanie obrazka do zmiennej `spo`
zmienna `ramka` służy do indeksowania kolejnych obrazków

- Do dokumentu **JS** wklej funkcję z ramki

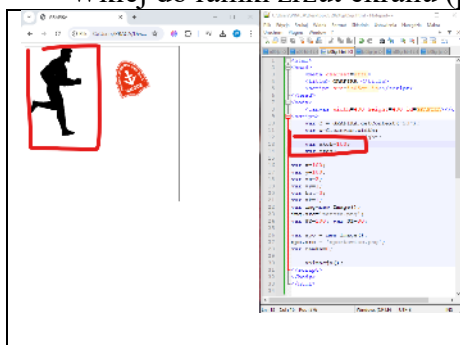
```
function Sportowiec() {
    var xr=ramka*300;
    if (ramka==5) ramka=0; else ramka++;
    C.drawImage(spo,xr,0,300,480,0,0,150,240);
}
```

funkcja `Sportowiec`

zmienna `xr` od którego pikseli w poziomie rozpoczyna się kolejna klatka – każda ma po 300 pikseli
klatki numerowane są od 0 do 5 – 6 klatek

z obrazka `sportowiec` wycinamy klatkę od punktu `(xr,0)` o wymiarach `(300,480)` i wklejamy do obszaru `canvas` od punktu `(0,0)` i skalujemy do wymiarów `(150,240)` – zmniejszamy o połowę

- W dokumencie **JS**, w funkcji `animacja()`, `clearTimeout(czas);` wpisz instrukcję `Sportowiec();`
- Zwiększ skok czasu do 100 milisekund, aby sportowiec nieco wolniej biegał
- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Sportowiec (1)

- Otwórz plik `sportowiec.png` w dowolnym edytorze grafiki
- (1) Na każdej klatce sportowca namaluj swoje inicjały w kolorze czerwonym
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Panorama (1)

- Pobierz do swojego foldera obrazek: <http://zsobobowa.eu/pliki/program/raclawice.png>

plik zawiera panoramę raclawicką – początek i koniec (prawie) identyczne

- Do dokumentu **HTML**, za znacznikiem `<body>` `<canvas width=` wpisz tekst

```
<body>  
<canvas width=400 height=500 id=PANORAMA></canvas>  
<br>
```

canvas o ID=PANORAMA o wymiarach 400x500

- Do dokumentu **HTML**, przed funkcją `animacja()` `animacja();` wklej tekst z ramki

```
var P = PANORAMA.getContext("2d");  
var pan = new Image();  
pan.src="raclawice.png";  
pan.onload=function () {  
    P.drawImage(pan,0,0);  
}  
var pw=500;  
var ps=2800;  
var px=0;
```

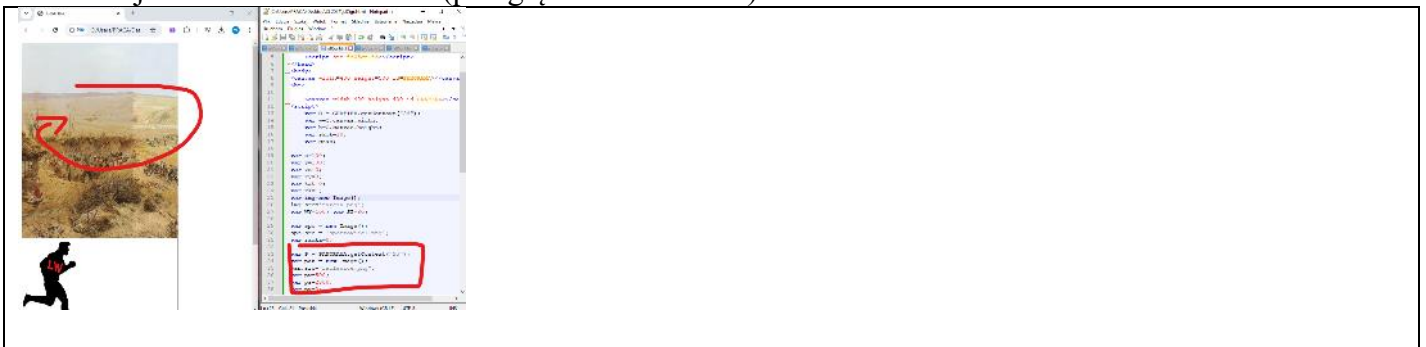
zmienna `P` pełni rolę „uchwyty” do nowego obszaru `canvas`
ładujemy obrazek do zmiennej `pan` o wymiarach 2800x500
zmienna `px` pełni rolę wskaźnika na obszar obrazka, który będzie pokazywany

- Do dokumentu **JS**, wklej funkcję z ramki

```
function Panorama () {  
    P.drawImage (pan,px,0);  
    P.drawImage (pan,px-ps,0);  
    if (px>=ps) px=0; else px++;  
}
```

funkcja `Panorama()` wyświetla obok siebie dwa obrazki
zmienna `px` pełni rolę wskaźnika na obszar obrazka, który będzie wyświetlany

- W dokumencie **JS**, w funkcji `animacja()` `clearTimeout(czas);` wpisz instrukcję `Panorama();`
- Zmniejsz skok czasu do 10 milisekund, aby panorama szybciej się przesuwała
- Zapisz dokumenty i odśwież przeglądarkę
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)



Panorama2 (1)

- Wykonaj zrzut całego ekranu (razem z datą i godziną) i zapisz go w swoim folderze z dowolną nazwą
- Zaanimuj ten zrzut ekranu
podmień nazwę w dokumencie **HTML**
- Wklej do ramki zrzut ekranu (przeglądarka i notatnik)

