STANDARDNI KORISNIČKI INTERFEJSI

Predavanje broj: 04

Nastavna jedinica: HTML5

Nastavne teme:

Canvas: pravougaonik,boje, gradijenti, putanje, složene krive, stilovi linija, osobine teksta, iscrtavanje teksta, senčenje, transformacije, obnavljanje stanja, kompozitne operacije, slike. SVG: krug, pravougaonik, poligon, gradijent, linija, staza, tekst, link, filteri. Video: tipovi, track, source, metode, svojstva. Audio.

Predavač: prof. dr Perica S. Štrbac, dipl. ing.

Literatura:

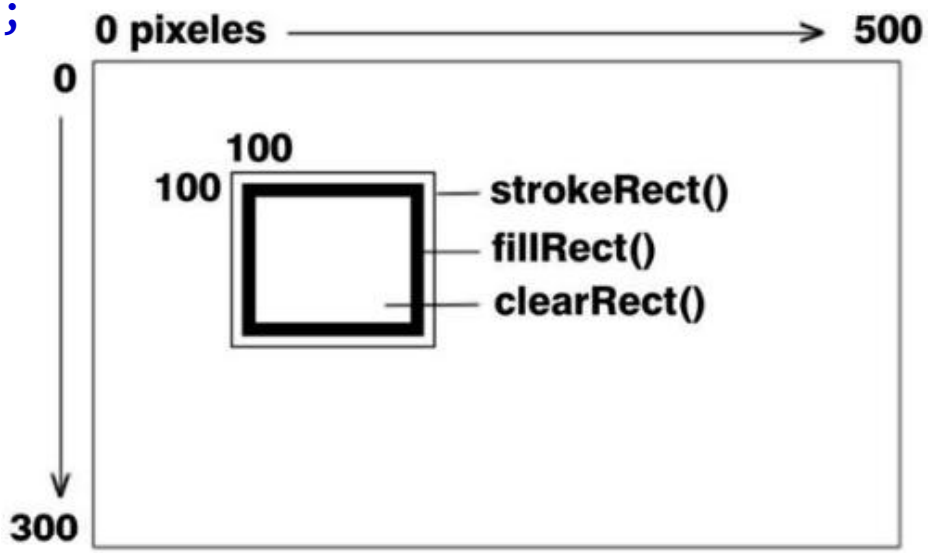
J. D. Gauchat, "Integrisane tehnologije za izradu WEB strana", Mikroknjiga, Beograd, 2014.

W3C Tutorials, Internet, 2014.

<canvas> pravougaonik

- Kreiranje pravougaonika:
  - nepopunjenog sa `strokeRect(x,y, sirina, visina)`
  - popunjenog sa `fillRect(x,y, sirina, visina)`
  - brisanje pravougaonog područja sa `clearRect(x,y, sirina, visina)`

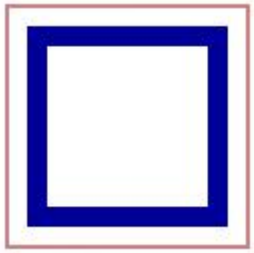
```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');
  k2d.strokeRect(100,100,120,120);
  k2d.fillRect(110,110,100,100);
  k2d.clearRect(120,120,80,80);
</script>
```



Boje

- Boje u kanvasu se mogu definisati kao što sledi:
  - strokeStyle, definiše boju linija datog oblika
  - fillStyle, definiše boju unutrašnjosti datog oblika
  - globalAlpha, zadaje providnost za sve oblike nacrtane na kanvasu

```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');

  k2d.fillStyle="#000099";
  k2d.strokeStyle="#990000";

  k2d.strokeRect(100,100,120,120);
  k2d.fillRect(110,110,100,100);
  k2d.clearRect(120,120,80,80);
</script>
```



- Moguće je boje zadati i sa:

```
strokeStyle="rgba(255,165,0,1)";
strokeStyle="rgb(255,165,0)";
```

HTML5: <canvas> gradijenti

- Gradijenti se mogu koristiti za popunjavanje pravougaonika, krugova, linija, teksta itd. Oblici na platnu nisu ograničeni na osnovne boje.
- Postoje dva tipa gradijenta:
  - `createLinearGradient(x1,y1, x2,y2)` – stvara linearni gradijent
  - `createRadialGradient(x1,y1,r1, x2,y2,r2)` - stvara radial/circular gradient
- Kada se dobije gradijent objekat, potrebno je dodati barem dva graničnika boje metodom `addColorStop(pozicija, boja)`
  - Metoda *addColorStop* specificira graničnik boje i njegovu poziciju u gradijentu.
  - Pozicija gradijenta može biti bilo gde između 0 i 1 i određuje početak slabljenja date boje.
- Za korišćenje gradijenta, potrebno mu je postaviti *fillStyle* ili *strokeStyle* svojstvo, posle čega se mogu nacrtati oblici kao što su pravougaonik, tekst ili linija.

HTML5: createLinearGradient(), createRadialGradient()

- Kreirati linearni gradijent. Popuniti pravougaonik korišćenjem gradijenta:

```
<script>
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  var grd = ctx.createLinearGradient(0,0, 200,0);
  grd.addColorStop(0,"red");
  grd.addColorStop(1,"white");
  ctx.fillStyle = grd;
  ctx.fillRect(10,10,150,80);
</script>
```

- Kreirati radijalni/kružni gradijent. Popuniti pravougaonik korišćenjem gradijenta:

```
<script>
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  var grd = ctx.createRadialGradient(75,50,5, 90,60,100);
  grd.addColorStop(0,"red");
  grd.addColorStop(1,"white");
  ctx.fillStyle = grd;
  ctx.fillRect(10,10,150,80);
</script>
```

Crtanje putanja

- Za crtanje putanja koriste se metode kao što sledi:
  - `beginPath()` , započinje opis novog oblika
  - `closePath()` , zatvara putanju generišući pravu liniju od poslednje do početne tačke. Može se izostaviti ako se hoće nacrtati otvoreni oblik ili ako je korišćena metoda fill().
  - `stroke()` , iscrtava putanju kao konturu.
  - `fill()` , iscrtava putanju kao oblik punog tela. Zatvaranje putanje je automatski.
  - `clip()` , zadaje površinu za isecanje u datom kontekstu. Pravi masku za isecanje koja je inicijalno ceo kontekst.
- Kod za crtanje putanje izgleda kao što sledi:

```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');
  k2d.beginPath();
    // ovde dolazi putanja
  k2d.stroke();
</script>
```

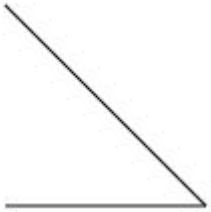
Crtanje putanje

- Za crtanje putanje na raspolaganju su sledeće metode:
  - `moveTo(x, y)` - pomera pero na zadatu poziciju
  - `lineTo(x, y)`
    - generiše pravu liniju od tekuće pozicije pera do nove pozicije zadate atributima x, y
  - `rect(x, y, sirina, visina)`
    - generiše pravougaonik kao deo putanje na zadatoj poziciji i u zadatoj veličini
  - `arc(x, y, radius, pocetniugao, zavrsniugao, smer)`
    - generiše luk ili krug sa zadatim parametrima u smeru kretanja kazaljke na satu (true) ili suprotnom (false)
  - `quadraticCurveTo(cpx, cpy, x, y)`
    - generiše kvadratnu Bezjeovu krivu koja počinje od tekuće pozicije pera a završava u poziciji datoj atributima x,y. Atributi cpx, cpy određuju kontrolnu tačku krive.
  - `bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y)`
    - Bezjeova kriva sa dve kontrolne tačke.

Crtanje putanje

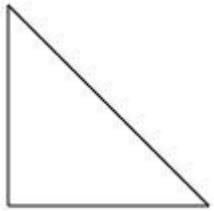
- Primer crtanja otvorene putanje

```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');
  k2d.beginPath();
    k2d.moveTo(100,100);
    k2d.lineTo(200,200);
    k2d.lineTo(100,200);
  k2d.stroke();
</script>
```



- Primer crtanja zatvorene putanje

```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');
  k2d.beginPath();
    k2d.moveTo(100,100);
    k2d.lineTo(200,200);
    k2d.lineTo(100,200);
  k2d.closePath();
  k2d.stroke();
</script>
```



Crtanje putanje

- Primer crtanja popunjene konture (putanje)

```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');
  k2d.fillStyle="rgb(0,128,0)";
  k2d.beginPath();
    k2d.moveTo(100,100);
    k2d.lineTo(200,200);
    k2d.lineTo(100,200);
  k2d.fill();
</script>
```

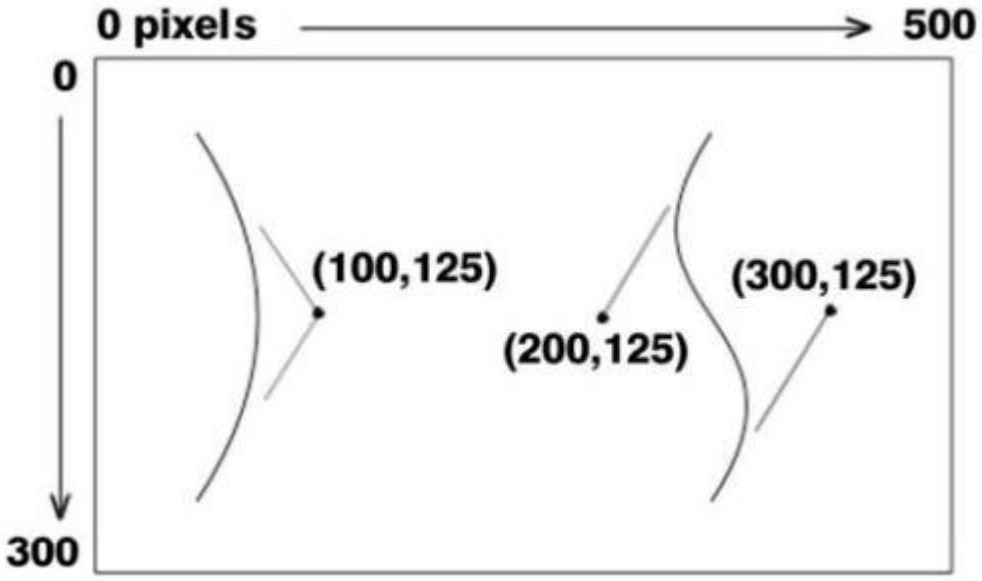
- Primer putanje brisanja

```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');
  k2d.beginPath();  k2d.moveTo(100,100);  k2d.lineTo(200,200);
                    k2d.lineTo(100,200);  k2d.clip();
  k2d.beginPath();  k2d.arc(100,100, 75, 0,Math.PI*2, true);
  k2d.stroke();
</script>
```

Crtanje složenih krivih

- Primer kvadratne i kubne Bezjeove krive.

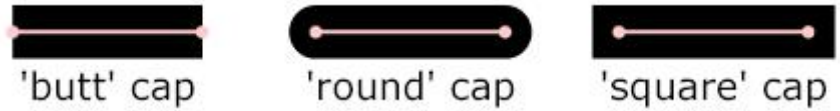
```
<script>
  var element=document.getElementById('myCanvas');
  k2d=element.getContext('2d');
  k2d.beginPath();
    k2d.moveTo(50,50);
    k2d.quadraticCurveTo(100,125, 50,200);
    k2d.moveTo(250,50);
    k2d.bezierCurveTo(200,125, 300,125, 250,200);
  k2d.stroke();
</script>
```



- ovde su podaci za Bezjeovu kvadratnu krivu:
  startna tačka (50,50)
  završna tačka (50,200)
  kontrolna tačka (100,125)
- ovde su podaci za Bezjeovu kubnu krivu:
  startna tačka (250,50)
  završna tačka (250,200)
  prva kontrolna tačka (200,125), druga kontrolan tačka (300,125)

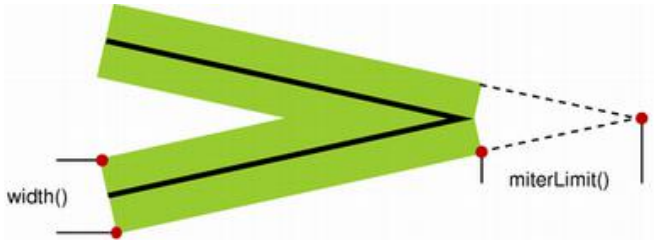
Stilovi linija

- Svojstva za stilove linija su kao što sledi:
  - lineWidth - definiše debljinu linije
  - lineCap



    - definiše oblik završetka linije. Moguće vrednosti su: butt, round, square.
  - lineJoin

    - definiše oblik veze između dve linije. Moguće vrednosti su: round, bevel, miter.
  - miterLimit



    - zajedno sa svojstvom lineJoin određuje koliko će veze između dve linije biti produžene kada se za lineJoin zada vrednost miter.

lineWidth, lineCap, lineJoin

- Primer koji koristi lineWidth, lineCap, lineJoin

```
<script>
  var element=document.getElementById('c2d');
  c2d=element.getContext('2d'); c2d.strokeStyle="rgb(255,0,0)";
  c2d.beginPath();
  c2d.arc(200,150,50,0,Math.PI*2, false);
  c2d.stroke(); c2d.strokeStyle="rgb(0,0,255)";
        c2d.lineWidth=10;
        c2d.lineCap="round";
        c2d.beginPath();
        c2d.arc(200,150,30,0,Math.PI, false);
        c2d.stroke();
  c2d.strokeStyle="rgb(0,255,0)";
  c2d.lineWidth=5;
  c2d.lineJoin="miter";
  c2d.beginPath();
  c2d.moveTo(195,135);
  c2d.lineTo(215,155);
  c2d.lineTo(195,155);
  c2d.stroke();
</script>
```

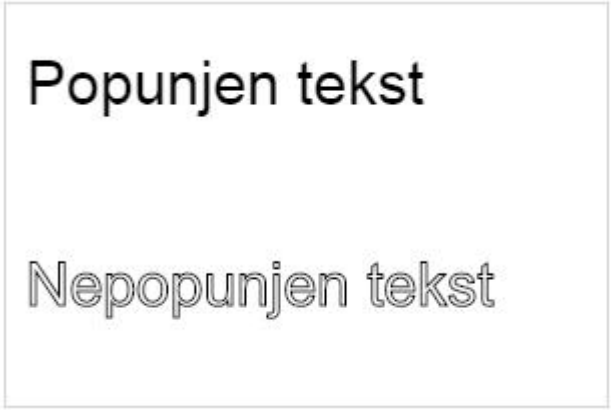
Osobine teksta

- Konfigurisanje teksta je kao što sledi:
  - `font`
    - definiše tip fonta koji se koristi.
  - `textAlign`
    - poravnava tekst horizontalno, vrednosti su: start, end, left, right, center.
  - `textBaseline`
    - definiše vertikalno poravnavanje, a vrednosti mogu biti: top, hanging, middle, alphabetic, bottom.

  - `strokeText(text, x, y)`
    - ispisuje zadati tekst kao konturni oblik na poziciji x,y pri čemu može da sadrži i četvrtu vrednost koja određuje maksimalnu veličinu.
  - `fillText(text, x, y)`
    - slično prethodnoj metodi ali se ispisuje popunjen tekst.
  - `measureText(text)`
    - vraća informacije o veličini teksta koji je prosleđen kao parametar.

HTML5: <canvas> tekst

- Primer:
  - Koristeći font "Arial" veličine 30 piksela ispisati
    - popunjen tekst na kanvasu, na poziciji 10,50
    - nepopunjen tekst na kanvasu, na poziciji 10,150




```
<script>
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  ctx.font = "30px Arial";
  ctx.fillText("Popunjen tekst",10,50);
  ctx.strokeText("Nepopunjen tekst",10,150);
</script>
```

Iscrtavanje teksta

- U sledećem primeru tekst **start** će se ispisati fontom verdana počevši od pozicije 100,100, a ako on nije dostupan onda fontom sans-sherif. Da je vrednost textAlign="end" onda bi tekst završio na poziciji 100,100.

```
<script>
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  ctx.font="bold 24px verdana, sans-serif";
  ctx.textAlign="start";
  ctx.fillText("moja poruka", 100,100);
</script>
```

- Primer merenja teksta:

```
 <script>
  var element=document.getElementById('myCanvas');
  ctx=element.getContext('2d');
  ctx.font="bold 24px verdana, sans-serif";
  ctx.textAlign="start";
  ctx.textBaseline="bottom";
  ctx.fillText("moja poruka", 100,124);
  var taman=ctx.measureText("moja poruka");
  ctx.strokeRect(100,100,taman.width,24);
</script>
```

Senke

- Kanvas sadrži sledeća svojstva za senke:
  - shadowColor
    - deklariše boju senke
  - shadowOffsetX
    - broj koji pokazuje koliko će senka biti horizontalno udaljena od objekta
  - shadowOffsetY
    - broj koji pokazuje koliko će senka biti vertikalno udaljena od objekta
  - shadowBlur
    - pravi efekat zamagljivanja za senku

```
<script>
 var element=document.getElementById('ctx');
 ctx=element.getContext('2d');
 ctx.shadowColor="rgba(0,0,0, 0.5)";
 ctx.shadowOffsetX=4;
 ctx.shadowOffsetY=4;
 ctx.shadowBlur=5;
 ctx.font="bold 50px verdana, sans-serif";
 ctx.fillText("moja poruka", 100,100);
</script>
```

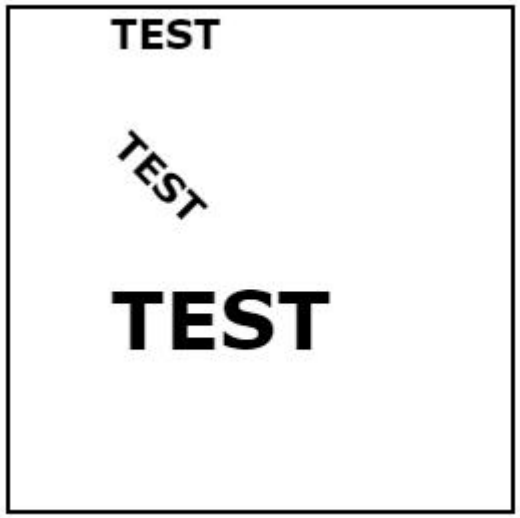
Transformacije

- Metode za transformacije su kao što sledi:
  - translate(x, y)
    - premešta koordinatni početak koji je podrazumevano (0,0).
  - rotate(alfa)
    - rotira kanvas oko koordinatnog početka za datu vrednost u radijanima
  - scale(x, y)
    - skaliranje sadržaja kanvasa
    - može se iskoristiti i za simetrična preslikavanja
  - transform(m1, m2, m3, m4, dx, dy)
    - zadavanje nove matrice transformacije koja se primenjuje na kanvas
  - setTransform(m1, m2, m3, m4, dx, dy)
    - resetuje tekuću transformaciju i zadaje novu na osnovu vrednosti svojih atributa

Transformacije

- Primer translacije, rotiranja i menjanja veličine

```
<script>
  var element=document.getElementById('myCanvas');
  ctx=element.getContext('2d');
  ctx.font="bold 20px verdana, sans-serif";
  ctx.fillText("TEST",50,20);
  ctx.translate(50,70);
  ctx.rotate((Math.PI/180)*45);
  ctx.fillText("TEST",0,0);
  ctx.rotate((-Math.PI/180)*45);
  ctx.translate(0,100);
  ctx.scale(2,2);
  ctx.fillText("TEST",0,0);
</script>
```



- Svaka transformacija je kumulativna.
  - ako bismo 2 puta pozvali metodu scale(2,2) rezultat bi bio da bi se učetvorostručila razmera sledećeg sadržaja kanvasa

transform, setTransform

- Podrazumevana matrica kanvasa je 1, 0, 0, 1, 0, 0.

```
<script>
  var element=document.getElementById('myCanvas');
  ctx=element.getContext('2d');
  ctx.transform(3,0,0,1,0,0);
  ctx.font="bold 20px verdana, sans-serif";
  ctx.fillText("TEST1",20,20);
  ctx.transform(1,0,0,10,0,0);
  ctx.fillText("TEST2",20,20);
  ctx.setTransform(1,0,0,1,0,0);
  ctx.fillText("TEST3",0,20);
</script>
```

- Prvo: razvlači se sledeći sadržaj kanvasa 3 puta po horizontali
- Drugo: još se razvlači sledeći sadržaj kanvasa i po vertikali 10 puta
  - računa se kumulativan efekat
- Treće: resetuje se matrica kanvasa i postavlja na 1,0,0,1,0,0 (podrazumevana).

Obnavljanje stanja

- Metode za snimanje i vraćanje stanja su:
  - save()
    - snima stanja kanvasa uključujući:
      - transformacije koje su već primenjene,
      - vrednosti svojstava stilizovanja
      - aktivnu stazu za isecanje (oblast napravljena pomoću metode clip())
  - restore()
    - vraća poslednje snimljeno stanje

```
<script>
  var element=document.getElementById('myCanvas');
  ctx=element.getContext('2d');
  ctx.save();
  ctx.translate(50,70);
  ctx.font="bold 20px verdana, sans-serif";
  ctx.fillText("TEST1",0,30);
  ctx.restore();
  ctx.fillText("TEST2",0,30);
</script>
```

globalCompositeOperation

- Kako se oblik pozicionira i kombinuje sa prthodnim oblicima na kanvasu:
  - globalCompositeOperation sa svojstvima:
    - source-over

      - novi oblik se crta preko onoga koji je na kanvasu (podrazumevano).
    - source-in

      - crta se samo onaj deo novog oblika koji preklapa postojeći oblik, a preostali delovi novog i postojećeg oblika postaju providni.
    - source-out

      - crta se samo onaj deo novog oblika koji ne preklapa postojeći oblik, a preostali delovi novog i postojećeg oblika postaju providni.
    - source-atop

      - crta se samo onaj deo novog oblika koji preklapa postojeći oblik pri čemu postojeći oblik ostaje sačuvan a ostali delovi novog oblika postaju providni.
    - lighter

      - crtaju se oba oblika ali boja preklopljenih boja dobija se sabiranjem boja.

globalCompositeOperation

- xor
  - crtaju se oba oblika ali delovi koji se poklapaju postaju providni.
- destination-over
  - novi oblik se crta iza postojećeg oblika na kanvasu
- destination-in
  - delovi postojećeg oblika na kanvasu koji se preklapaju sa novim oblikom ostaju isti, a ostali delovi uključujući i nov oblik postaju providni.
- destination-out
  - delovi postojećeg oblika na kanvasu koji se ne preklapaju sa novim oblikom ostaju isti , a ostali delovi uključujući i nov oblik postaju providni.
- destination-atop
  - unija destination-in i source-out
- darker
  - crtaju se oba oblika ali boja delova koji se preklapaju određuje se oduzimanjem vrednosti boja
- copy - crta se samo novi oblik a postojeći oblik postaje providan

globalCompositeOperation, destination-atop

- Primer sa svojstvom destination-atop


```
<script>
  var element=document.getElementById('myCanvas');
  ctx=element.getContext('2d');
  ctx.fillStyle="#990000";
  ctx.fillRect(100,100,300,100);
  ctx.globalCompositeOperation="destination-atop";
  ctx.fillStyle="#AAAAFF";
  ctx.font="bold 80px verdana, sans-serif";
  ctx.textAlign="center";
  ctx.textBaseline="middle";
  ctx.fillText("TESTtestTEST",250,110);
</script>
```

HTML5: <canvas> slike

- Za crtanje slike na kanvasu, koriste se sledeće metode:
  - `drawImage(image,x,y)`
    - crta sliku *image* na poziciji x,y
  - `drawImage(image,x,y, sirina, visina)`
    - crta sliku *image* na poziciji x,y u zadatoj visini i širini
  - `drawImage(image,x1,y1,sirina1,visina1,x2,y2,sirina2,visina2)`
    - odseca deo slike *image* od pozicije x, y i date širine1 i visine1 a onda taj odsečeni deo crta na kanvasu na koordinatama x2, y2 date širine2 i visine2.

- Primer**:** Nacrtati sliku unutar kanvasa:

```
<script>
   var c = document.getElementById("myCanvas");
   var ctx = c.getContext("2d");
   var img = document.getElementById("scream");
   ctx.drawImage(img,10,10);
</script>
```

HTML5: <canvas>, slike

```
<!DOCTYPE html>
<html><body>
<p>Image to use:</p>
<img id="scream" src="img_the_scream.jpg" alt="The Scream"
width="220" height="277">

<p>Canvas to fill:</p>
<canvas id="myCanvas" width="200" height="200"
        style="border:1px solid #d3d3d3;">
        Your browser does not support the HTML5 canvas tag.
</canvas>
<p><button onclick="myCanvas()">Try it</button></p>
<script>
  function myCanvas() {
    var c = document.getElementById("myCanvas");
    var ctx = c.getContext("2d");
    var img = document.getElementById("scream");
    ctx.drawImage(img,10,10);
  }
</script>
</body></html>
```

Rezultat koda

Image to use:

Canvas to fill:

Try it

Image to use:

Canvas:

Try it

```
ctx.drawImage(img,10,10,150,180);
```

Image to use:

Canvas:

```
ctx.drawImage(img,90,130,50,60,10,10,50,60);
```

HTML5: <svg> circle, rect

- SVG znači Scalable Vector Graphics, skalabilna vektorska grafika
- SVG crtanje kruga.

```
<!DOCTYPE html>
<html><body>
<svg width="100" height="100">
  <circle cx="50" cy="50" r="40"
          stroke="green" stroke-width="4" fill="yellow" />
  Sorry, your browser does not support inline SVG.
</svg>
</body></html>
```

- SVG crtanje pravougaonika.

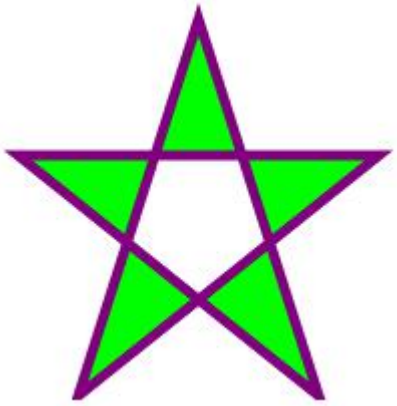
```
<!DOCTYPE html>
<html><body>
<svg width="400" height="100">
  <rect width="400" height="100"
     style="stroke:rgb(0,0,0);stroke-width:10;fill:rgb(0,0,255)"/>
</svg>
</body></html>
```

HTML5: <svg> rect polygon

- SVG crtanje zaobljenog pravougaonika (poluprovidnog). mogu se koristiti i fill-opacity i stroke-opacity.

```
<!DOCTYPE html>
<html><body>
<svg width="400" height="180">
  <rect x="50" y="20"
        rx="20" ry="20"
        width="150" height="150"
        style="fill:red;stroke:black;stroke-width:5;opacity:0.5" />
</svg>
</body></html>
```

- SVG poligon (pazite, tačke se navode kao x1,y1 razmak x2,y2, ...).

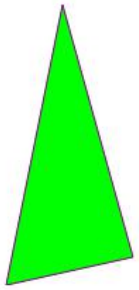


```
<!DOCTYPE html>
<html><body>
<svg width="300" height="200">
  <polygon points="100,10 40,198 190,78 10,78 160,198"
  style="fill:lime;stroke:purple;stroke-width:5;
         fill-rule:evenodd;" />
</svg>
</body></html>
```

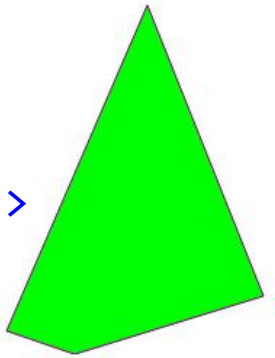
HTML5: <svg> polygon

- Neki poligoni

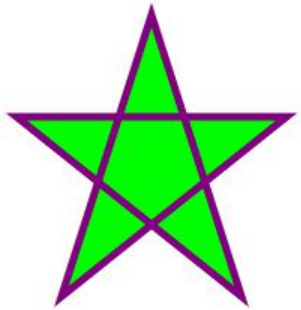
```
<svg height="210" width="500">
  <polygon points="200,10 250,190 160,210"
           style="fill:lime;stroke:purple;stroke-width:1" />
</svg>
```



```
<svg height="250" width="500">
  <polygon points="220,10 300,210 170,250 123,234"
           style="fill:lime;stroke:purple;stroke-width:1" />
</svg>
```



```
<svg height="210" width="500">
  <polygon points="100,10 40,198 190,78 10,78 160,198"
          style="fill:lime;stroke:purple;stroke-width:5;
                 fill-rule:nonzero;" />
</svg>
```



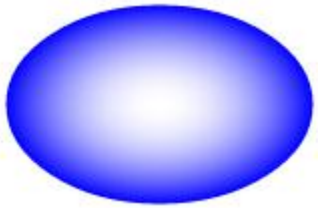
HTML5: <svg> lineargradient ellipse text

- SVG gradijenti, elipsa i tekst.

```
<!DOCTYPE html>
<html><body>
<svg height="130" width="500">
  <defs>
    <linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="0%">
      <stop offset="0%"
            style="stop-color:rgb(255,255,0);stop-opacity:1" />
      <stop offset="100%"
            style="stop-color:rgb(255,0,0)  ;stop-opacity:1" />
    </linearGradient>
  </defs>

  <ellipse cx="100" cy="70"
           rx="85"  ry="55" fill="url(#grad1)" />

  <text fill="#ffffff"
       font-size="45" font-family="Verdana"
       x="50" y="86">SVG</text>
  Sorry, your browser does not support inline SVG.
</svg></body></html>
```

HTML5: <svg> filteri

- Filter: <radialGradient>



```
<svg height="150" width="500">
  <defs>
    <radialGradient id="grad1"
                    cx="50%" cy="50%" r="50%"
                    fx="50%" fy="50%">
      <stop offset="0%"
            style="stop-color:rgb(255,255,255); stop-opacity:0" />
      <stop offset="100%"
            style="stop-color:rgb(0,0,255);     stop-opacity:1" />
    </radialGradient>
  </defs>
  <ellipse cx="200" cy="70" rx="85" ry="55" fill="url(#grad1)" />
</svg>
```

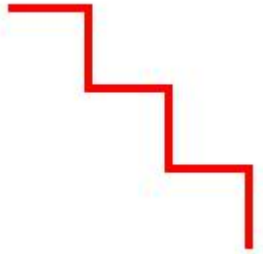
- Vrednosti fx i fy odnose se na fokalnu tačku

HTML5: <svg> line

- SVG linija.

```
<!DOCTYPE html>
<html><body>
<svg height="210" width="500">
  <line x1="0" y1="0" x2="200" y2="200"
        style="stroke:rgb(255,0,0);stroke-width:2" />
</svg>
</body></html>
```

- SVG polyline (probajte fill:blue)



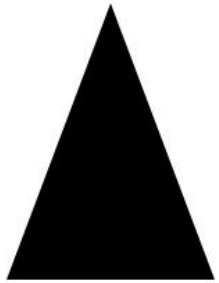
```
<!DOCTYPE html>
<html><body>
<svg height="180" width="500">
  <polyline points="0,40 40,40 40,80 80,80 80,120 120,120 120,160"
            style="fill:white;stroke:red;stroke-width:4" />
</svg>
</body></html>
```

HTML5: <svg> <path>

- Tag <path> omogućuje definisanje staze.
- Sledi lista prefiksa u listi vrednosti koje se navode za formiranje staze koji predstavljaju komande (za atribut d):
  - M = moveto
  - L = lineto
  - H = horizontal lineto
  - V = vertical lineto
  - C = curveto
  - S = smooth curveto
  - Q = quadratic Bézier curve
  - T = smooth quadratic Bézier curveto
  - A = elliptical arc (dx, dy ellipserot large,right endx,endy)
  - Z = closepath
- Sve prethodno prikazane komande mogu biti izražene:
  - malim slovima što predstavlja relativnu pozicioniranost,
  - velikim slovima što predstavlja apsolutnu pozicioniranost.

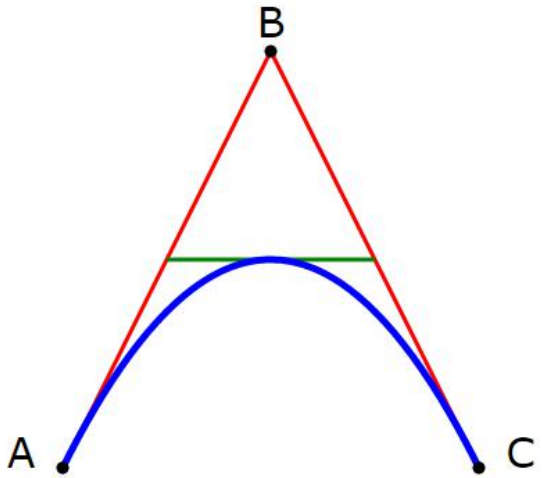
HTML5: <svg> <path>

- Primer:



```
<!DOCTYPE html>
<html><body>
<svg height="210" width="400">
  <path d="M150 0 L75 200 L225 200 Z" />
</svg>
</body></html>
```

- Da bi se nacrtala Bezjeova kriva kao na slici



koristimo sledeći kod:

HTML5: path

```
<svg height="400" width="450">
  <path id="lineAB" d="M 100 350 l 150 -300"
                    stroke="red" stroke-width="3"   fill="none" />
  <path id="lineBC" d="M 250 50 l 150 300"
                    stroke="red" stroke-width="3"   fill="none" />
  <path             d="M 175 200 l 150 0"
                    stroke="green" stroke-width="3" fill="none" />
  <path             d="M 100 350 q 150 -300 300 0"
                    stroke="blue"  stroke-width="5" fill="none" />
  <!-- relevantne tacke -->
  <g stroke="black" stroke-width="3" fill="black">
    <circle id="pointA" cx="100" cy="350" r="3" />
    <circle id="pointB" cx="250" cy= "50" r="3" />
    <circle id="pointC" cx="400" cy="350" r="3" />
  </g>
  <!-- labele tacaka -->
  <g font-size="30" font="sans-serif" fill="black"
     stroke="none"  text-anchor="middle">
    <text x="100" y="350" dx="-30">A</text>
    <text x="250" y="50"  dy="-10">B</text>
    <text x="400" y="350" dx="30">C</text>
</g> </svg>
```

HTML5: svg text

- SVG tekst.

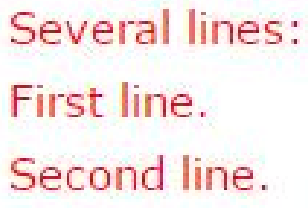
```
<svg height="30" width="200">
  <text x="0" y="15" fill="red">I love SVG!</text>
</svg>
```

- SVG rotirani tekst.

```
<svg height="60" width="200">
  <text x="0" y="15"
        fill="red"
        transform="rotate(30 20,40)">I love SVG</text>
</svg>
```

- SVG više linija teksta.

```
<svg height="90" width="200">
  <text x="10" y="20" style="fill:red;">Several lines:
    <tspan x="10" y="45">First line.</tspan>
    <tspan x="10" y="70">Second line.</tspan>
  </text>
</svg>
```



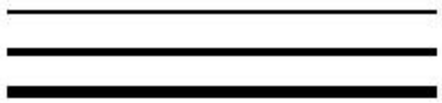
HTML5: <svg> stroke stroke-width

- Linije u boji

```
<!DOCTYPE html>
<html><body>
<svg height="80" width="300">
  <g fill="none">
    <path stroke="red"    d="M5 20 l215 0" />
    <path stroke="green"  d="M5 40 l215 0" />
    <path stroke="black"  d="M5 60 l215 0" />
  </g>
</svg>
</body></html>
```

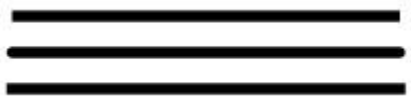
- Linije različite širine

```
<!DOCTYPE html>
<html><body>
<svg height="80" width="300">
  <g fill="none" stroke="black">
    <path stroke-width="2" d="M5 20 l215 0" />
    <path stroke-width="4" d="M5 40 l215 0" />
    <path stroke-width="6" d="M5 60 l215 0" />
  </g></svg> </body></html>
```



HTML5: <svg> stroke-linecap stroke-dasharray

- Linije sa različitim krajevima.



```
<!DOCTYPE html>
<html><body>
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="6">
    <path stroke-linecap="butt"   d="M5 20 l215 0" />
    <path stroke-linecap="round"  d="M5 40 l215 0" />
    <path stroke-linecap="square" d="M5 60 l215 0" />
  </g></svg>
</body></html>
```

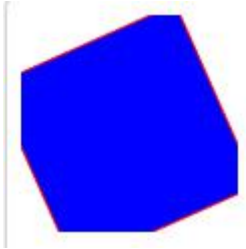
- Linije definisanog izgleda.

```
<!DOCTYPE html>
<html><body>
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="4">
    <path stroke-dasharray="5,5"             d="M5 20 l215 0" />
    <path stroke-dasharray="10,10"           d="M5 40 l215 0" />
    <path stroke-dasharray="20,10,5,5,5,10" d="M5 60 l215 0" />
  </g>
</svg>
</body></html>
```

SVG, animacija

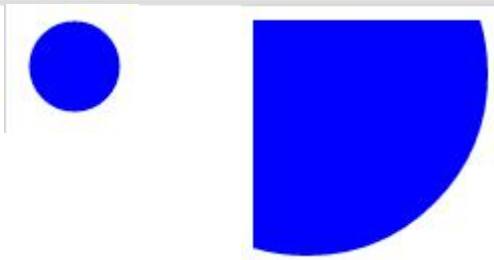
- Animacija, rotacija pravougaonika

```
<!DOCTYPE html>
<html><body>

<svg height="120" width="120">
<rect x="10" y="10" height="110" width="110"
      style="stroke:#ff0000; fill:#0000ff">

        <animateTransform
            attributeName="transform"
            begin="0s"
            dur="20s"
            type="rotate"
            from="0 60 60"
            to="360 60 60"
            repeatCount="indefinite"
        />
</rect>
</svg>

</body></html>
```



SVG, animacija

- Animacija, povećanje kruga nakon 5 s.



```
<svg height="520" width="520">
<circle cx="30" cy="30" r="25" style="stroke: none; fill:
#0000ff;">
    <set attributeName="r" attributeType="XML"
         to="100"
         begin="5s"  />
</circle>
</svg>
```

- Animacija, krug se ponovljeno pomera po x osi (stazu pređe za 5s)

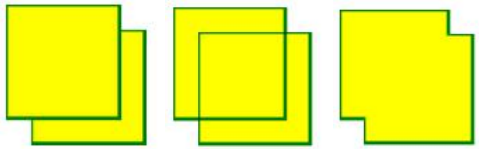
```
<svg height="520" width="520">
<circle cx="30" cy="30" r="25" style="stroke:none; fill: #0000ff;">
    <animate attributeName="cx" attributeType="XML"
             from="30"  to="470"
             begin="0s" dur="5s"
             fill="remove" repeatCount="indefinite"/>
</circle>
</svg>
```

HTML5: <svg> filteri

- Filteri: <feGaussianBlur>


```
<svg height="110" width="110">
  <defs>
    <filter id="f1" x="0" y="0">
      <feGaussianBlur in="SourceGraphic" stdDeviation="15" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
        fill="yellow" filter="url(#f1)" />
</svg>
```

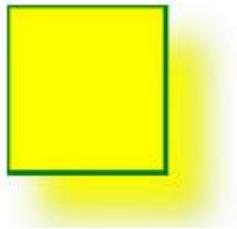
- Filteri: <feOffset> <feBlend> (mode: normal, multiply,screen)



```
<svg height="120" width="120">
  <defs>
    <filter id="f1" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
      <feBlend  in="SourceGraphic" in2="offOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
        fill="yellow" filter="url(#f1)" />
</svg>
```

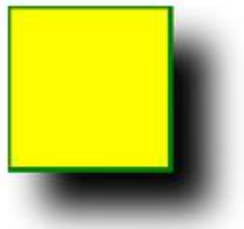
HTML5: <svg> filteri

- Filter: <feGaussianBlur>



```
<svg height="140" width="140">
  <defs>
    <filter id="f2" x="0" y="0" width="200%" height="200%">
      <feOffset in="SourceGraphic" result="offOut" dx="20" dy="20" />
      <feGaussianBlur in="offOut" result="blurOut"
stdDeviation="10" />
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
        fill="yellow" filter="url(#f2)" />
</svg>
```

HTML5: <svg> filteri

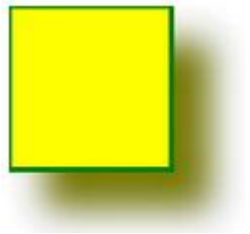


- Filter: <feGaussianBlur>

```
<svg height="140" width="140">
  <defs>
    <filter id="f3" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceAlpha" dx="20" dy="20" />
      <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" /
>
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
        fill="yellow" filter="url(#f3)" />
</svg>
```

HTML5: <svg> filteri

- Filter: <feGaussianBlur>



```
<svg height="140" width="140">
 <defs>
  <filter id="f4" x="0" y="0" width="200%" height="200%">
   <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
   <feColorMatrix result="matrixOut" in="offOut" type="matrix"
     values="0.2 0 0 0 0 0 0.2 0 0 0 0 0 0.2 0 0 0 0 0 1 0" />
   <feGaussianBlur result="blurOut" in="matrixOut" stdDeviation="10" /
>
   <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
  </filter>
 </defs>
 <rect width="90" height="90" stroke="green" stroke-width="3"
       fill="yellow" filter="url(#f4)" />
</svg>
```

$$\begin{pmatrix} R' \\ G' \\ B' \\ A' \\ 1 \end{pmatrix} = \begin{pmatrix} a00 & a01 & a02 & a03 & a04 \\ a10 & a11 & a12 & a13 & a14 \\ a20 & a21 & a22 & a23 & a24 \\ a30 & a31 & a32 & a33 & a34 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} R \\ G \\ B \\ A \\ 1 \end{pmatrix}$$

HTML5: <video>

- Standardi pre HTML5, nisu podržavali prikazivanje videa/fimova u okviru web stranice.
- Pre standarda HTML5, video se mogao prikazivati u web stranici jedino preko dodatka (npr. flash). Pri tome su različiti web čitači zahtevali različite dodatke (plug-ins).
- Trenutno, postoje tri video formata koja su podžana za <video> element u HTML5:
  - MP4 = MPEG 4 datoteke sa H264 video kodekom i AAC audio kodekom
  - WebM = WebM datoteke sa VP8 video kodekom i Vorbis audio kodekom
  - Ogg = Ogg datoteke sa Theora video kodekom i Vorbis audio kodekom

  Pripadni MIME-tip (Multipurpose Internet Mail Extensions, višenamenska proširenja za elektronsku poštu) je :
  - MP4 , video/mp4
  - WebM, video/webm
  - Ogg, video/ogg

HTML5: MIME tipovi za video formate

- HTML5 definiše novi element koji specificira standardni način za integrisanje videa i filmova u veb stranicu: <video> element.
- U HTML5, za prikazivanje videa, dovoljno je uneti sledeći kod:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>video</title>
</head>
<body>
  <video width="320" height="240" controls>
    <source src="movie.mp4" type="video/mp4">
    <source src="movie.ogg" type="video/ogg">
    Your browser does not support the video tag.
  </video>
</body></html>
```

- Atributi videa:
  - controls prikazuje kontrole videa
  - autoplay automatski startuje video

HTML5: <video> objašnjenje

- Atribut *control* dodaje video kontrole, kao što su play, pause i volume.
- Dobra praksa podrazumeva zadavanje atributa za visinu i širinu (*width* i *height*).
  - U tom slučaju se rezerviše prostor za sliku u fazi učitavanja stranice.
  - U suprotnom, veb čitač nije u stanju da odredi veličinu slike, pa će se prikaz slike menjati u fazi učitavanja slike.
- Takođe je potrebno uneti sadržaj teksta između tagova
  <video> i </video>
  za veb čitače koji ne podržavaju <video> element.
- Element <video> dozvoljava višestruke <source> elemente.
- Elementi <source> mogu povezati različite video datoteke.
  - Web čitač će pokrenuti prvu datoteku sa njemu prepoznatljivim formatom.
- Ekstenzija vtt predstavlja **Web V**ideo **T**ext **T**rack format zapisa.

```
WEBVTT

00:11.000 --> 00:13.000
- We are in New York City
```

HTML5: <video> <track>

- Tag <track> omogućuje ubacivanje prevoda.

```
<video width="320" height="240" controls>
  <source src="forrest_gump.mp4" type="video/mp4">
  <source src="forrest_gump.ogg" type="video/ogg">
  <track
src="subtitles_en.vtt" kind="subtitles" srclang="en" label="English"
default>
  <track src="subtitles_no.vtt" kind="subtitles" srclang="no" label="Norweg
ian">
```

Atribut	Vrednost	Opis
default	default	Podrazumevana traka
kind	captions, chapters, descriptions, metadata, subtitles	Specificira tekst traku
label	*text*	Specificira naslov tekst trake
src	*URL*	Obavezan atribut, URL fajla tekst trake
srclang	*language_code*	Specificira jezik podataka tekst trake (zahteva se navođenje ako se koristi kind="subtitles")

HTML5: <video><track> kind

- Atribut kind može imati sledeće vrednosti:
  - captions
    - traka definiše prevod dijalog i zvučnih efekata
      - podesno za gluve korisnike
  - chapters
    - traka definiše naslove poglavlja
      - podesno za navigaciju po izvoru medija
  - descriptions
    - traka definiše tekstualni opis video sadržaja
  - metadata
    - metapodaci, definiše se sadržaj koristeći skript. Nije vidljivo korisniku.
  - subtitles
    - traka definiše prevod (subtitle) koji se koristi za video

HTML5: <video> DOM metode, svojstva, događaji

- HTML5 poseduje: metode, svojstva (properties) i događaje za <video> i <audio> elemente.
- Ove metode, svojstva i događaji omogućuju korišćenje <video> i <audio> elemenata preko JavaScripta.
- Postoje metode za pokretanje, pauziranje i punjenje sadržaja i svojstva kao što su vreme trajanja i jačina zvuka.
- Takođe postoje događaji za obaveštavanje o trenutku pokretanja, pauziranja ili završetka <video> elementa.
- Primer koji sledi na jednostavan način ilustruje adresiranje <video> elementa, čitanje i postavljanje svojstva, kao i poziv metoda:
  - *play()*
  - *pause()*
  - takođe, pozivaju se dva svojstva: *paused* i *width.*

HTML5: <video> DOM metode, svojstva, događaji

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>video</title>
  <script>
    function init(){myVideo=document.getElementById("video1"); }
    function playPause()
    {
      if (myVideo.paused)  myVideo.play();
      else                 myVideo.pause();
    }
    function makeBig()   { myVideo.width=560; }
    function makeSmall() { myVideo.width=320; }
    function makeNormal(){ myVideo.width=420; }
    addEventListener('load',init)
  </script>
</head>
```

HTML5: <video> DOM metode, svojstva, događaji

```
<body>
 <div style="text-align:center">
    <button onclick="playPause()">Play/Pause</button>
    <button onclick="makeBig()">Big</button>
    <button onclick="makeSmall()">Small</button>
    <button onclick="makeNormal()">Normal</button>
    <br>
    <video id="video1" width="420">
      <source src="mov_bbb.mp4" type="video/mp4">
      <source src="mov_bbb.ogg" type="video/ogg">
      Your browser does not support HTML5 video.
    </video>
 </div>
</body>
</html>
```

HTML5: audio MIME tipovi za audio formate

- MIME tipovi za audio formate

Format	MIME-type
MP3	audio/mpeg
Ogg	audio/ogg
Wav	audio/wav

- Audio tagovi

Tag	Opis
<audio>	Definiše zvučni sadržaj
<source>	Definiše izvore za <video> i <audio>

HTML5: audio

- Pre standarda HTML5, audio datoteke su u web stranici mogle biti pokretane jedino preko dodatka (npr. flash). Pri tome su različiti web čitači zahtevali različite dodatke (plug-ins).
- HTML5 definiše novi element koji specificira standardni način za integrisanje audio datoteka u veb stranicu: <audio> element

Primer


```
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio>
```

- Atribut *controls* dodaje audio kontrole, kao što su play, pause i volume
- Takođe je potrebno uneti sadržaj teksta između tagova <audio> i </audio> za veb čitače koji ne podržavaju <audio> element
- Element <audio> dozvoljava višestruke <source> elemente koji mogu povezati različite audio datoteke. Web čitač će pokrenuti prvu datoteku sa njemu prepoznatljivim formatom.