

BALTIC TALENTS ACADEMY

---

# CSS ANIMACIJOS

# KARTOJIMAS

- ▶ float
- ▶ clear
- ▶ position

## ELEMENTŲ IŠDĖSTYMAS - FLOAT

Elementų išdėstymą taip pat galima kontroliuoti su **float** savybe.

- ▶ **left** - elementai dėstomi iš kairės pusės vienas po kito. Kai netelpa puslapyje tai dėstome vėl naujoje eilutėje iš kairės pusės
- ▶ **right** - tas pats tik iš dešinės. Reikia tik atkreipti dėmesį į tai, kad pirmas elementas bus prie pat dešinio krašto, jam iš kairės antras ir t.t.

## ELEMENTŲ IŠDĖSTYMAS - CLEAR

Jei norim kad kiti elementai būtų dėliojami atsižvelgiant į plaukiojančius (t.y. turinčius savybę **float**), reikia jiems panaudoti savybę **clear**:

- ▶ **none** - reikšmė pagal nutylėjimą - plaukiojantys elementai gali būti iš abiejų pusių
- ▶ **left** - plaukiojantys elementai neleidžiami iš kairės pusės, t.y. jei tokių yra tai elementas dedamas į naują eilutę
- ▶ **right** - tas pats tik iš dešinės
- ▶ **both** - neleidžiami iš abiejų pusių

# ELEMENTŲ IŠDĖSTYMAS

CSS savybė **position** nusako kaip elementas bus patalpintas html puslapyje:

- ▶ **static** - standartinis išdėstymas - kai nenurodyta
- ▶ **relative** - standartinis išdėstymas
- ▶ **fixed** - elementas neužima vietos ir padedamas fiksuotoje pozicijoje ekrano atžvilgiu
- ▶ **absolute** - elementas neužima vietos ir padedamas atžvilgiu tėvo ar pirmo protėvio kuris turi nustatytą poziciją (**relative**, **absolute** ar **fixed**)

# ELEMENTŲ IŠDĖSTYMAS

Elemento išdėstymas koreguojamas su **left, top, right, bottom**.

- ▶ Jei elementas turi **relative** poziciją, tai šios reikšmės nurodo kiek ir kur pastumti elementą iš standartinės pozicijos kurioje jis būtų jei reikšmės nebūtų nurodytos (arba nurodyta 0)
- ▶ Jei elementas turi **absolute** arba **fixed** poziciją, tai šios reikšmės nurodo kiek ir kur elementą pastumti pagal atitinkamus tėvinio elemento kraštus

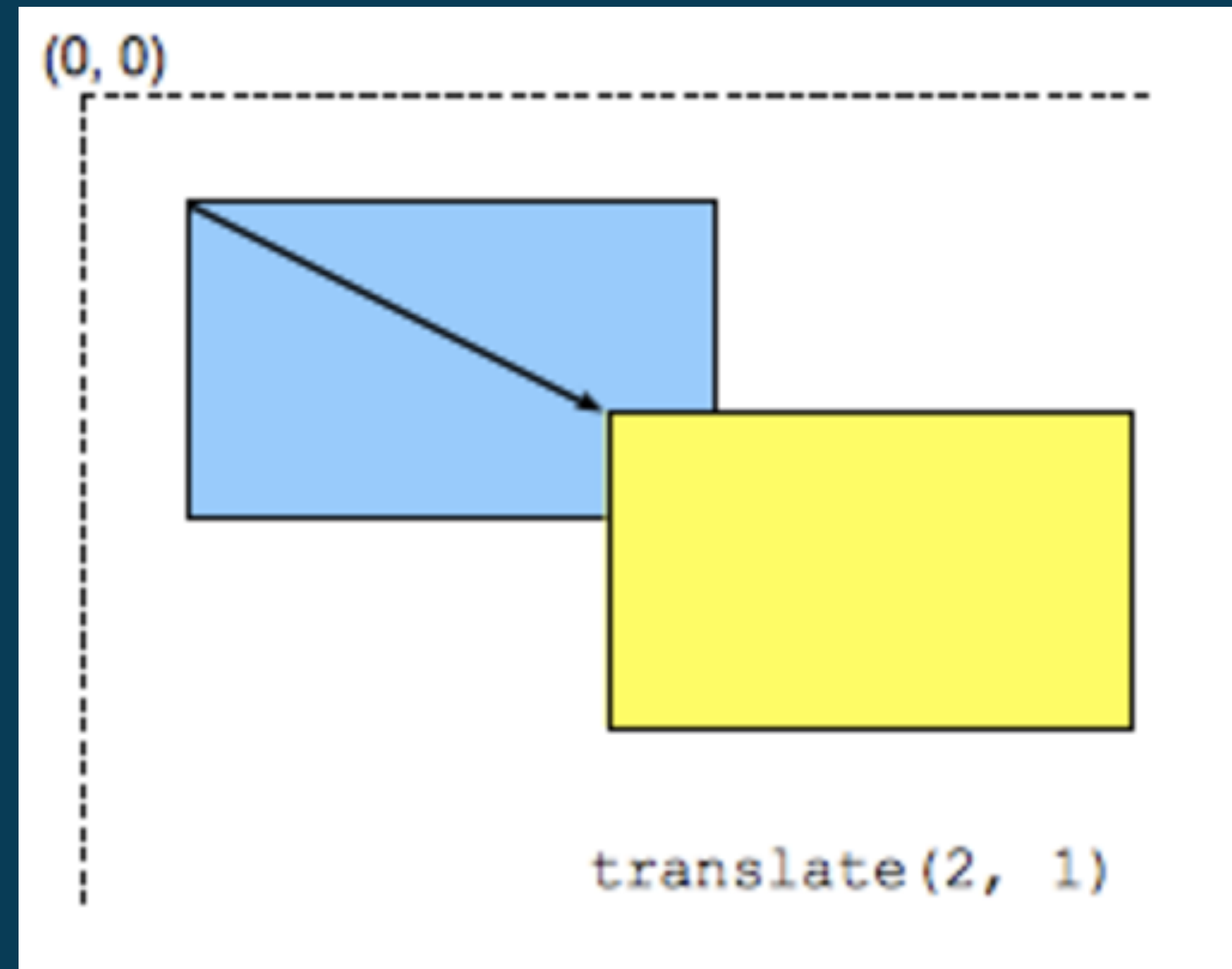
# TRANSFORMACIJOS

2D

3D

## TRANSFORMACIJOS 2D - PERKELTI (TRANSLATE)

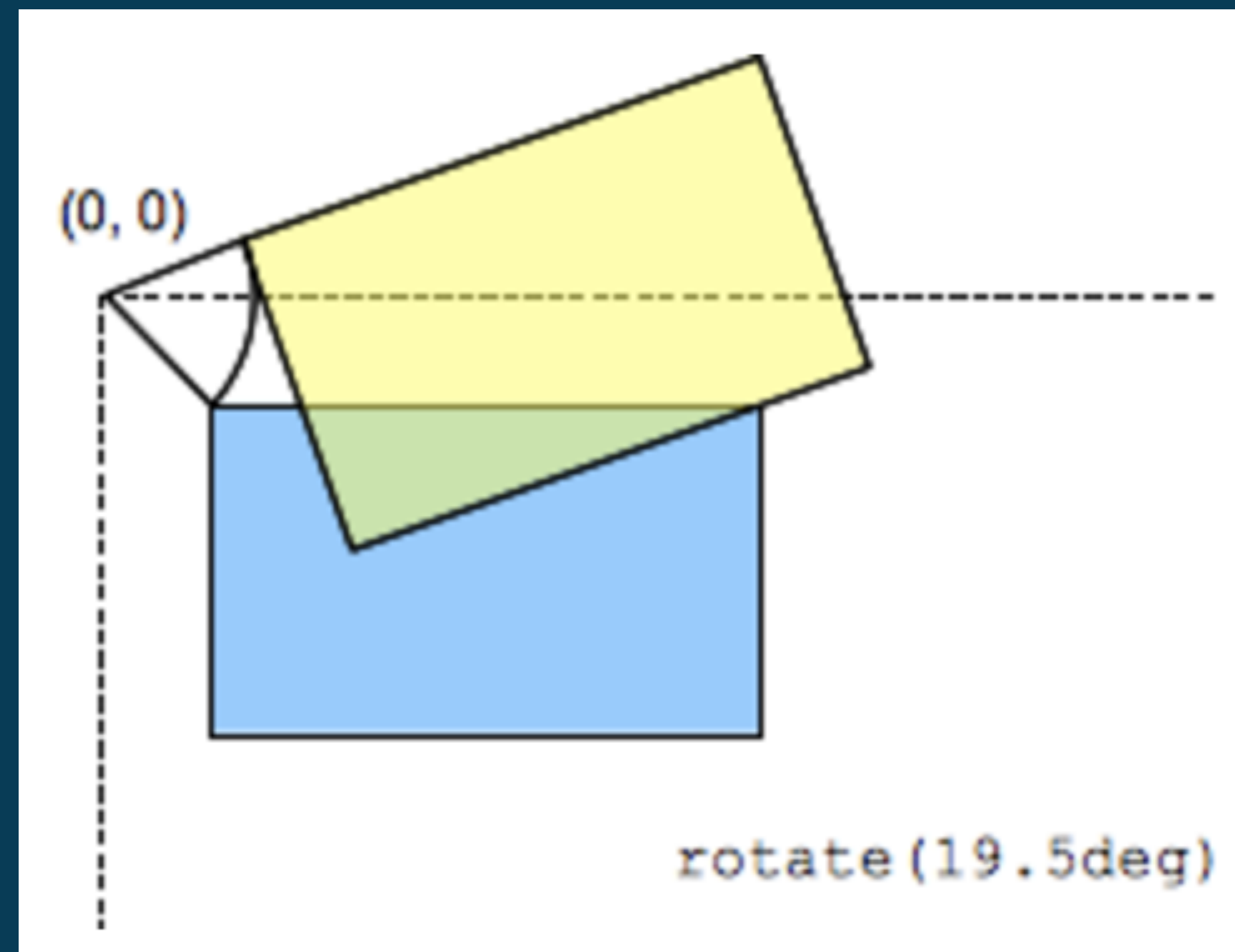
- ▶ `transform: translate(tx, ty);`
- ▶ `transform: translateX(tx);`
- ▶ `transform: translateY(ty);`





# TRANSFORMACIJOS 2D - PASUKTI (ROTATE)

► `transform: rotate(19.5deg);`



## TRANSFORMACIJOS 2D - CENTRAS

- ▶ Sukimo transformacijai būtina žinoti kur yra transformacijos centras, t.y. taškas aplink kurį sukamas (transformuojamas) objektas. Toks taškas pagal nutylėjimą yra objekto viduryje, bet jį galima apibrėžti ir kitur:
- ▶ `transform-origin: x-axis y-axis;`
  - ▶ *x-axis*: left | center | right | *length* | %
  - ▶ *y-axis*: top | center | bottom | *length* | %

## TRANSFORMACIJOS 2D - DIDINTI / MAŽINTI (SCALE)

- ▶ `transform: scale(sx, sy);`
- ▶ `transform: scaleX(sx);`
- ▶ `transform: scaleY(sy);`

**sx, sy** - tai didinimo/mažinimo koeficientai pagal x ir y ašį atitinkamai, pvz.:

- 1 - niekas nesikeičia,
- 3 - tris kartus didinamas,
- 0.5 - du kart mažinamas

# PERĖJIMAI (TRANSITIONS)

Animacija - tai elemento savybių pasikeitimas per nustatytą laiką.

Kaip turi vykti elemento kažkokios savybės pakeitimas yra nurodoma perėjimo (transitions) savybėse:

- ▶ *transition: property duration function delay, ...;*

pvz.: *transition: color 1s;*

- ▶ *transition-property: name;*

- ▶ *transition-duration: time;*

- ▶ *transition-timing-function: function-name;*

- ▶ *transition-delay: time;*

Daugiau info: [https://www.w3schools.com/cssref/css3\\_pr\\_transition.asp](https://www.w3schools.com/cssref/css3_pr_transition.asp)

# PRATIMAS

Aprašykite HTML struktūrą ir sukurkite stilius:

- ▶ Puslapyje vienas pilkas kvadratas 25% naršyklės lango pločio;
- ▶ Užėjus su pele ant jo, jis turi pradėti sukstis ir per 2 sekundes pavirsti į žalią apskritimą
- ▶ Nuvedus pelę jis vėl turi sukstis atgal ir vėl pavirsti į pilką kvadratą

Panaudokite `:hover`, `transitions` ir `transform`

# ANIMACIJOS

- ▶ `@keyframes name {  
 from { background-color: red; width: 10px; }  
 to { background-color: blue; width: 200px; }  
}`
- ▶ `@keyframes name {  
 0% { background-color: red; width: 10px; }  
 50% { background-color: yellow; width: 100px; }  
 100% { width: 200px; }  
}`

# ANIMACIJOS

- ▶ *animation: name duration timing-function delay iteration-count direction fill-mode play-state*
- ▶ *animation-name: name*
- ▶ *animation-duration: time*
- ▶ *animation-timing-function: ease | linear | ease-in | ease-out | ease-in-out*
- ▶ *animation-delay: time* - reikšmė pagal nutylėjimą **0s**
- ▶ *animation-iteration-count: number | infinite* - reikšmė pagal nutylėjimą **1**
- ▶ *animation-play-state: paused | running*

# ANIMACIJOS

- ▶ `animation-direction`: **normal** | `reverse` | `alternate` | `alternate-reverse` | ...
  - ▶ `alternate` - animuojama pirmyn, o po to atgal į pradinės reikšmės
  - ▶ `alternate-reverse` - animuojama atgal, o po to pirmyn
- ▶ `animation-fill-mode`: **none** | `forwards` | `backwards` | `both` | ...
  - ▶ **none** - savybės prieš ir po animacijos nesikeičia
  - ▶ `forwards` - savybės įgyja naujas animacijos pabaigos reikšmes
  - ▶ `backwards` - pirskiriamos pradinės savybės prieš pradedant animaciją laiko intervalui nurodytam `animation-delay`
  - ▶ `both` - veikia ir `forwards` ir `backwards` taisyklės

Daugiau info: [https://www.w3schools.com/cssref/css3\\_pr\\_animation.asp](https://www.w3schools.com/cssref/css3_pr_animation.asp)



# UŽDAVINYS

Aprašykite HTML struktūrą ir sukurkite stilius. Pradinė padėtis atidarius puslapį - visi apskritimai yra centre ir užlipę vienas ant kito. Praėjus kelioms sekundėms jie turi “nuvažiuoti” į parodytas pozicijas.

★) Pabandykite animacijai panaudoti bezier funkcijas, kad animacija atrodytų kai spyruoklė (<http://cubic-bezier.com/>)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce vestibulum in nisi id sagittis. Nunc ac iaculis orci. Cras eros massa, auctor rutrum nulla aliquam, feugiat laoreet urna. Proin a molestie quam. Nam ac nisi vel odio finibus vehicula. Fusce scelerisque odio vitae nunc finibus, eu varius lacus vehicula. Pellentesque id sapien magna. Nam rhoncus congue est, id interdum orci lobortis eu. Proin accumsan ut tellus in imperdiet.

