



НИЕ ВЯРВАМЕ ВЪВ ВАШЕТО БЪДЕЩЕ



# css animation

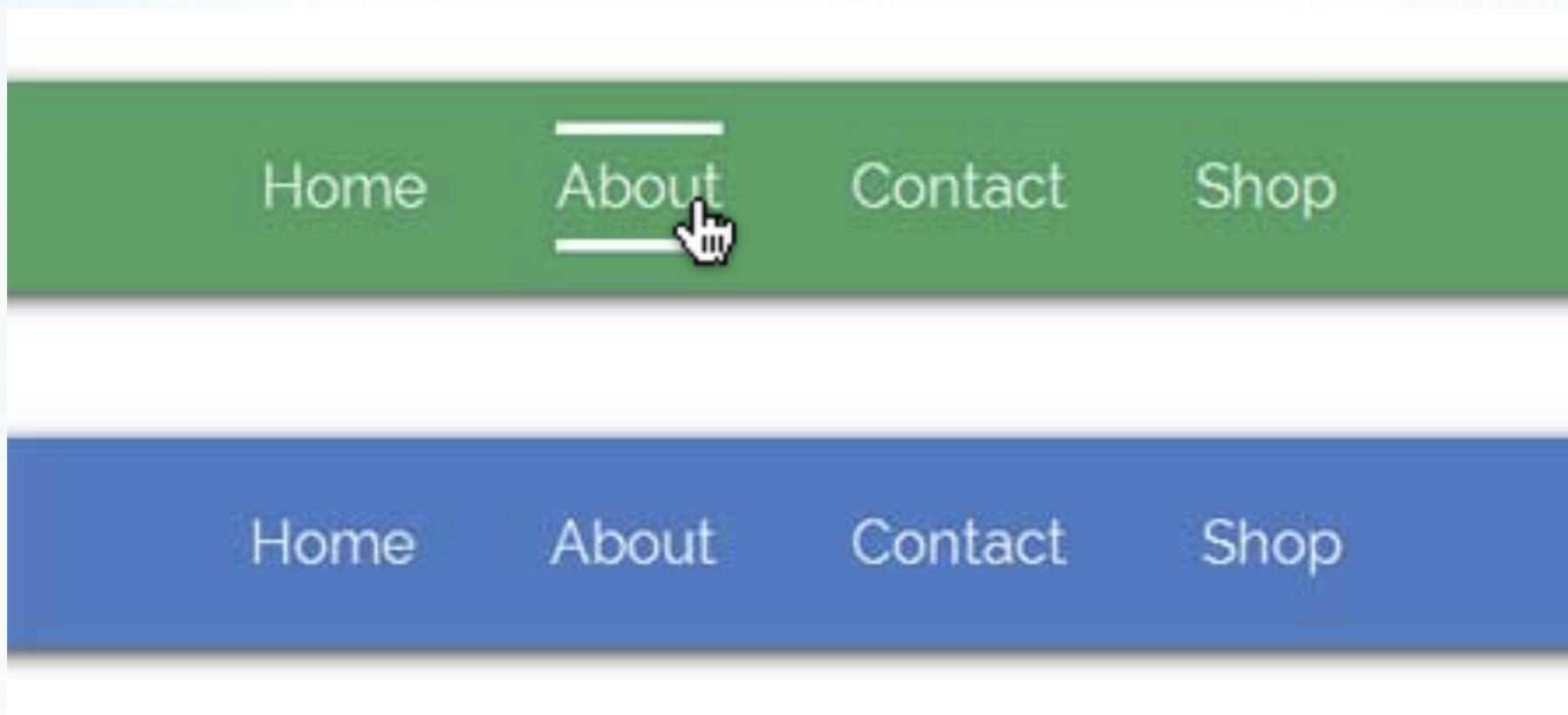
# Анимация

В CSS анимацията представлява последователност от транзишъни (transitions)

# Transition

наричаме преминаването на CSS property-тата  
на даден елемент от едно състояние в друго

# CSS transitions



# Синтаксис

- `transition <PROPERTY> <TIME> <METHOD>;`
- където
  - PROPERTY е CSS пропъртито, което ще се променя
  - TIME е колко секунди ще трае прехода
  - METHOD е функция (алгоритъм) на прехода - дали постепенно ще се случва (linear) или ще забавя в началото и края (ease-in-out)
- Пример:

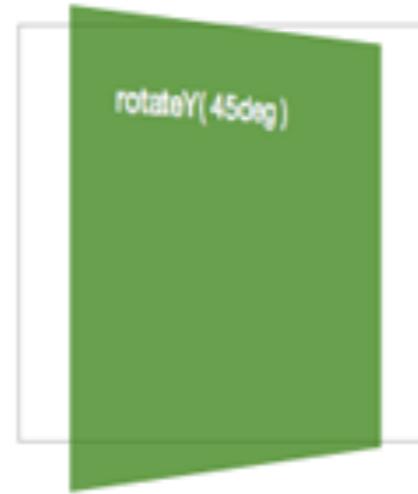
```
p { transition: all 2s; }
p:hover { font-size: 2em; }
```

- Ако прехода ни трае 0.1s - то промяната ще се случи много бързо (само ще премигне). Ако заложим 2s - промяната ще настъпва постепенно в рамките на 2 секунди
  - Можем да зададем transition както само за определени properties\*, така и за всички наведнъж:  
`transition: width 2s ease-in;`  
`transition: all 1s linear;`
- \* въпросните пропъртита трябва да имат различни стойности в двете състояния
- Състоянията се превключват чрез превключване на CSS класа или псевдокласа (hover, focus) на елемента.

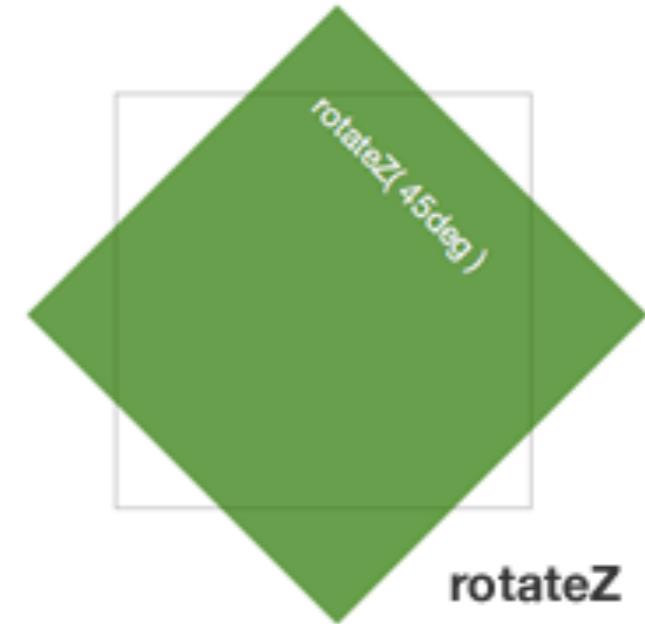
# CSS transforms



`rotateX( 45deg )`



`rotateY( 45deg )`



`rotateZ( 45deg )`



`translateX(30px)`



`translateY(30px)`

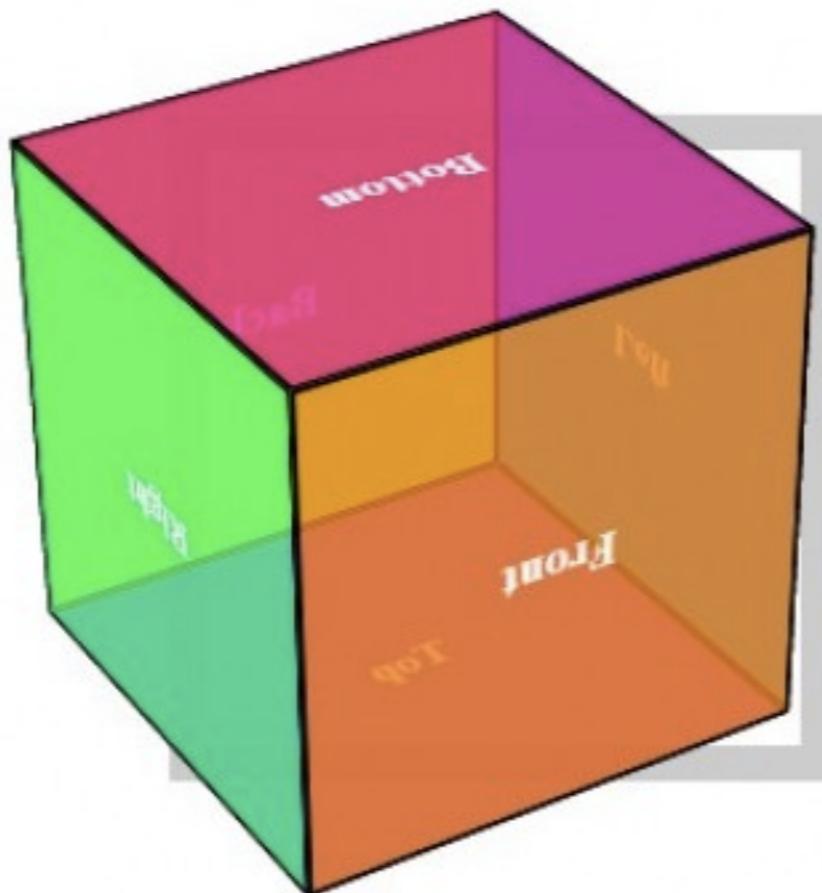


`translateZ(50px)`

# Трансформации

- 2d трансформации
  - translate(x,y) - преместваме по някоя от координатите
  - scale(x,y) - промяна на размера на някое от измеренията (височина, ширина, дълбочина)
  - rotate(angle) - завъртане
  - skew(x-angle,y-angle) - накланяне
- 3d трансформации: translate3d, rotate3d, perspective
- *Важно: трансформациите не се поддържат от всички браузъри и освен това понякога изискват префикс!!*

# CSS animation



- Ако комбинираме **transitions** и **transforms**, можем да направим интересни ефекти наподобяващи анимация
- Тъй като за една анимация са ни необходими повече от 2 състояния, които да се сменят, ни трябва специален синтаксис за да можем да дефинираме отделните състояния (кадри)
- Това е възможно благодарение на правилото **@keyframes**
- Можем да приложим повече от една анимации едновременно върху един елемент
- **Важно:** *Използва префикси в различните браузъри!!*  
<http://caniuse.com/#search=animation>

# {LESS}

- Пре-процесор за CSS
- Програмен език за css, който се компилира до чист css
- Дава възможност за използването на променливи, функции и наследяване при изготвянето на css-а за дадена страница
- Премахва досадните повторения в css и прави стиловете по-гъвкави и автономни
- Има 3 начина за ползване
  - с JavaScript компилатор: <http://lesscss.org/#client-side-usage>
  - с бекенд компилатор (less плъгин за бекенд приложението)
  - ръчно компилиране от командния ред

# LET'S DO SOME LESS



- сайт:  
[lesscss.org/](http://lesscss.org/)
- JS CDN:  
`<script src="//cdnjs.cloudflare.com/ajax/libs/less.js/2.5.3/less.min.js"></script>`
- Online interpreter:  
<http://lesscss.org/less-preview/>
- Документация:  
<http://lesscss.org/functions/>

# Canvas



- canvas буквално означава платно за рисуване
- в HTML canvas е елемент, който се използва за динамично генерирано графично съдържание и се поддържа от всички съвременни браузъри
- Представлява едно празно правоъгълно пространство, върху което рисуваме чрез javascript
- Пример:

```
<canvas width="650" height="200">  
    Your browser doesn't support canvas.  
</canvas>
```

(забележете, че винаги трябва да задаваме *width* и *height*!)

# Особености

- Ако браузърът не поддържа canvas, ще се изпише текста между двета тага (иначе текста не се вижда)
- За да рисуваме в canvas-а, трябва да използваме едно негово основно пропърти - **context**:

```
var canvas = document.getElementsByTagName('canvas')[0];
var ctx = canvas.getContext("2d");
ctx.fillRect(20, 20, 30, 40);
```

- Чудесна референция за **canvas** има в MDN (ако я изчетете, ще го научите, обещавам! :))  
[https://developer.mozilla.org/en-US/docs/Web/API/Canvas\\_API/Tutorial/Drawing\\_shapes](https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API/Tutorial/Drawing_shapes)

# Интересни примери

- Часовник  
[http://www.w3schools.com/canvas/canvas\\_clock.asp](http://www.w3schools.com/canvas/canvas_clock.asp)
- Анимация с орбити и въртене  
<http://ocanvas.org/demos/4>
- 3D анимация с кубове  
<http://cssdeck.com/labs/html5-canvas-3d-cubes>
- 2D canvas игра:  
[https://developer.mozilla.org/en-US/docs/Games/Tutorials/2D\\_Breakout\\_game\\_pure\\_JavaScript](https://developer.mozilla.org/en-US/docs/Games/Tutorials/2D_Breakout_game_pure_JavaScript)



# Въпроси?

# Полезни връзки

- Stop drawing a dead fish

<https://www.youtube.com/watch?v=ZfytHvgHybA&t=23s>

- Inventing on principle

<https://www.youtube.com/watch?v=PUv66718DII>

- Light table

<http://lighttable.com/>

# Примери

<http://zenlabs.pro/courses/sa-fe/lessons/lesson20/examples.zip>

<https://codepen.io/jenie/pen/vKNoNK?editors=1100>

<https://codepen.io/jenie/pen/RRraye>

<https://codepen.io/jenie/pen/GNQyQX>