

50 ULTIMATE CSS THREADS BY PRATHAM

Compilation of 50+ threads on
Web Development



@Prathkum





Pratham 🎨🚀 @Prathkum

18 Mar • 10 tweets • [Prathkum/status/1372500279212511233](https://twitter.com/Prathkum/status/1372500279212511233)

Tr

5 cool things you can do with CSS 🎨

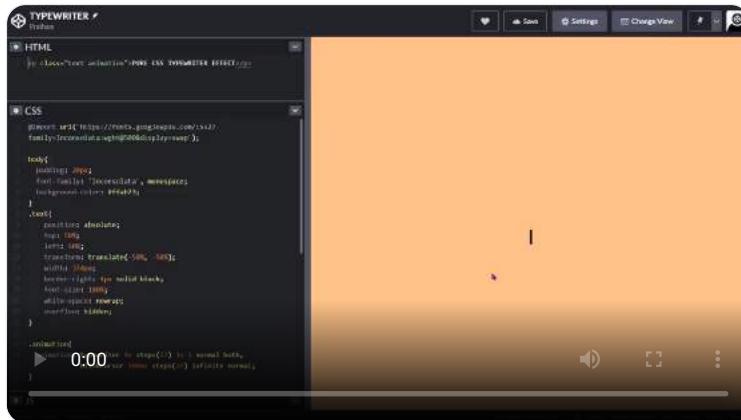
THREAD 🧶

1 Typewriter effect using CSS only

- You can create cool typing effect using steps() function of animation-timing-function

Displays an animation iteration along n stops along the transition, displaying each stop for equal lengths of time

{ 2 / 10 }



The code is pretty simple. Check it out here ↴

<https://codepen.io/prathamkumar/pen/qBqGep>

{ 3 / 10 }

2 Feature Testing

- The @ supports CSS at-rule lets you specify declarations that depend on a browser's support for one or more specific CSS features

{ 4 / 10 }

```
@supports (display: grid) {  
  div {  
    display: grid;  
  }  
}
```

3 Tooltip

You probably notice that when you hover over something, an additional piece of information corresponding to that element pops up. This is called Tooltip

{ 5 / 10 }

Hence tooltips guide your visitors to take action and hence provides extra layer of guidance without any difficulties

- You can create tooltip using CSS. It pretty easy



Pratham
@Prathkum



A tooltip is used to specify the extra bit of information typically when user moves the mouse over it

Tooltips guide your visitors to take action and hence provides extra layer of guidance without any difficulties

Let's see how we can create this  



I AM PRATHAM

7:52 AM · Mar 10, 2021

1 A M

①

Heart 85 Reply 2 ⌂ Copy link to Tweet

{ 6 / 10 }



I AM PRATHAM

▶ 0:00

Speaker icon

⋮

4 Image carousel using 2 lines of CSS

- A carousel is a set of rotating banners, or a slideshow. It allows you to display as many slides as you want.

Creating images carousel was considered as a tedious task in previous time but not now

{ 7 / 10 }

Now you can create image carousel without using JavaScript as well

Moreover this smooth scroll as well 😊

🔗 <https://codepen.io/prathamkumar/pen/bGBozXj>

{ 8 / 10 }

```
● ● ●
.container {
  scroll-snap-type: x mandatory;
}

.slides {
  scroll-snap-align: start;
}
```

5 Select elements like a pro😎

- Have you heard about :is pseudo-class. It takes a selector list as its argument, and selects any element that can be selected by one of the selectors in that list

For example, (next tweet)

{ 9 / 10 }

Instead of doing this ✘

```
.container h1,
.container h2,
.container h3 {
  color: red;
}
```

Do this ✓

```
.container :is(h1, h2, h3) {
  color: red;
}
```

{ 10 / 10 }

```
● ● ●  
.container :is(h1, h2, h3) {  
  color: red;  
}
```

• • •



Pratham 🎨🚀 @Prathkum

28 Mar • 26 tweets • [Prathkum/status/1376149828514811909](https://twitter.com/Prathkum/status/1376149828514811909)

Tr

100 Free CSS resources you will love🎨

Mega Thread 🧶👉

- Colors
- AI integrated color picker
- Code Snippets
- Document and notes
- CSS generator
- Design Inspiration
- GitHub Repo
- Cheat Sheets
- Validators
- Chrome Extension
- YouTube
- Animation Libraries
- Threads
- Website
- Learn by playing
- Templates

◆ Colors

1. Color Hunt

Color Hunt - Color Palettes for Designers and Artists
Color Hunt is a free and open platform for color inspiration with thousands of trendy hand-picked color palettes
<https://colorhunt.co>

2. Coolors

Colors - The super fast color schemes generator!
Generate or browse beautiful color combinations for your designs.
<https://color.co>

3. HTML color codes



HTML Color Codes
Easily find HTML color codes for your website using our color picker, color chart and HTML color names with Hex color codes, RGB and HSL values.
<https://htmlcolorcodes.com>

4. UI Gradients

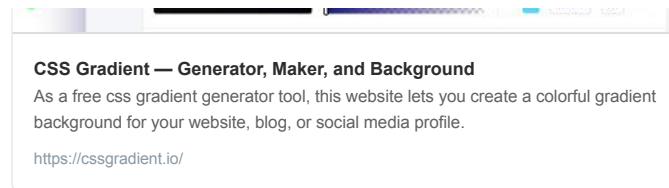


uiGradients - Beautiful colored gradients
A handpicked collection of beautiful color gradients for designers and developers
<https://uigradients.com/>

5. Gradient generator



CSS Gradient
SWATCHES EXAMPLES BLOG RESOURCES



CSS Gradient — Generator, Maker, and Background
As a free css gradient generator tool, this website lets you create a colorful gradient background for your website, blog, or social media profile.
<https://cssgradient.io/>

6. Encycolorpedia

encycolorpedia.com

7. WebFx

<https://webfx.com/web-design/color-picker/>

8. Radial Gradient

css-gradient.com

9. COLORS

clrs.cc

10. Flat UI colors 2

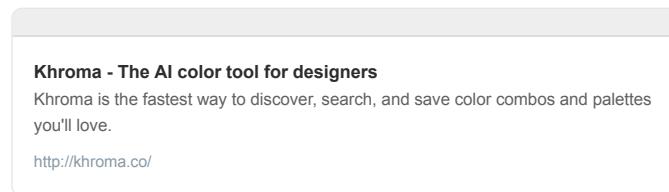


FLAT UI COLORS 2
280 Handpicked Colors, Ready for Copy & Paste.

Flat UI Colors 2 - 14 Color Palettes, 280 colors 
280 handpicked colors ready for COPY & PASTE
<https://flatuicolors.com/>

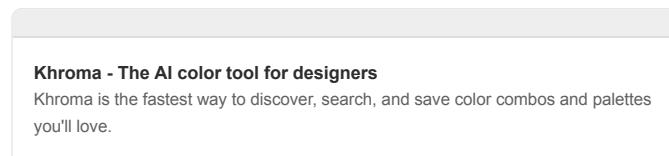
- AI integrated color picker

11. Khroma



Khroma - The AI color tool for designers
Khroma is the fastest way to discover, search, and save color combos and palettes you'll love.
<http://khroma.co/>

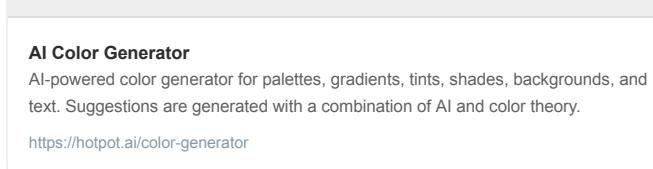
12. Colormind



Khroma - The AI color tool for designers
Khroma is the fastest way to discover, search, and save color combos and palettes you'll love.

13. Eva Design System
colors.eva.design

14. AI color generator



◆ Code Snippets

15. Web tools



Web Code Tools
The ultimate CSS generator. Learn CSS with our generators and preview your results! Start now without any CSS experience.

<https://webcode.tools/css-generator>

16. Little snippets
littlesnippets.net

17. Enjoy CSS
enjoycss.com

18. CSS Tricks



19. CSS Deck
cssdeck.com

20. W3 How to
w3schools.com/howto/

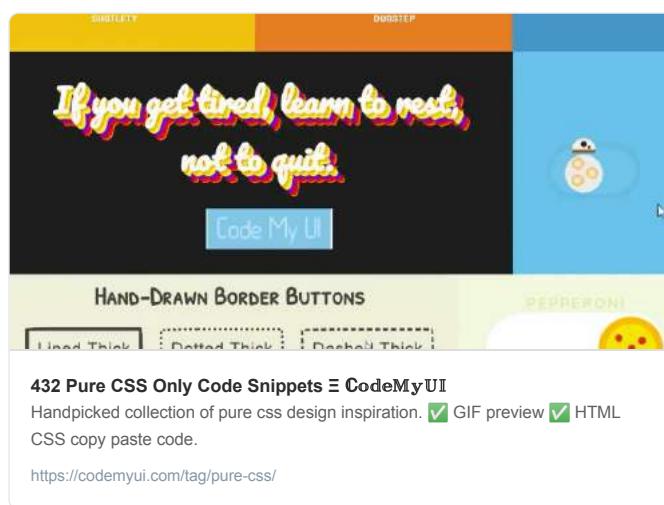
21. Snipplr
snipplr.com

22. Stack overflow



23. Codepen
<https://codepen.io/collection/DYpwPE>

24. Code my UI



- ♦ Document and notes

25. W3 Schools
w3schools.com/css/

26. MDN
<https://developer.mozilla.org/en-US/docs/Web/CSS>

27. DevDocs
devdocs.io/css/

28. Geeks for geeks



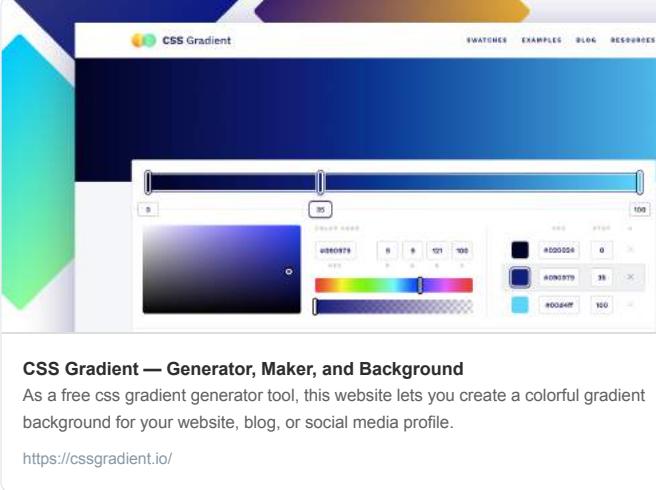
The screenshot shows the homepage of GeeksforGeeks' CSS tutorial section. It features a large, stylized '3' icon on the left and the text 'CSS TUTORIAL' in large, bold letters. Below this, there's a brief description: 'CSS Tutorials - GeeksforGeeks. A Computer Science portal for geeks. It contains well written, well thought and well explained computer science and programming articles, quizzes and practice/competitive programming/company interview...'. At the bottom, there's a link: <https://geeksforgeeks.org/css-tutorials/>.

29. Tutorials point
tutorialspoint.com/css/index.htm

- ◆ CSS generator

30. Stripes generator
stripesgenerator.com

31. Gradient generator

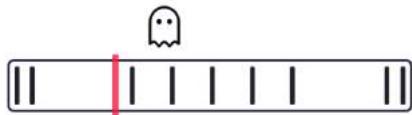


The screenshot shows the CSS Gradient Generator tool. It displays a preview of a gradient background with a color gradient from green to blue. Below the preview, there's a color editor interface with sliders for color, saturation, and brightness, and input fields for hex codes (#009977, #000024), HSL values (0, 0, 100), and RGB values (0, 0, 100). At the bottom, there's a description: 'CSS Gradient — Generator, Maker, and Background. As a free css gradient generator tool, this website lets you create a colorful gradient background for your website, blog, or social media profile.' and a link: <https://cssgradient.io/>.

32. Pattern generator
patternify.com

33. Animation generator

||| keyframes



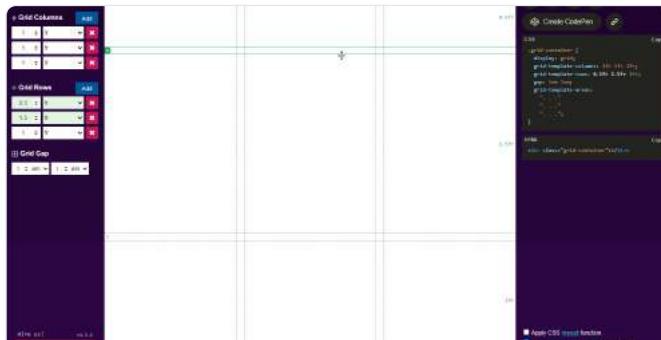
A simple CSS toolbox for generating animations, shadows, colors, & more.

Keyframes.app | CSS Toolbox

Keyframes helps you write better CSS with a suite of tools to create CSS @Keyframe animations, box shadows, colors, & more

<https://keyframes.app/>

34. Layout generator



Interactive CSS Grid Generator | Layoutit Grid

Quickly design web layouts, and get HTML and CSS code. Learn CSS Grid visually and build web layouts with our interactive CSS Grid Generator.

<https://grid.layoutit.com/>

35. CSS Accordion Slider Generator

accordionslider.com

36. Grid Layout generator

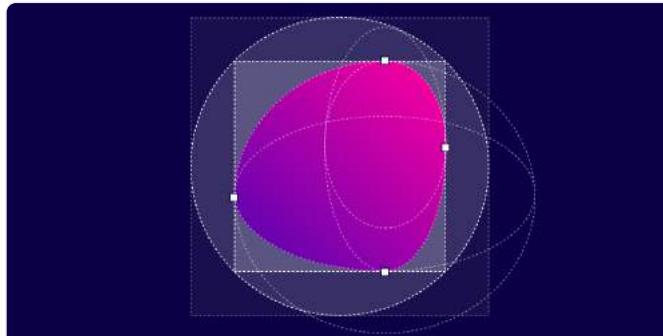


The Quickest & Easiest Way To Build Complex CSS Grid Layouts

The Quickest & Easiest Way To Build Complex CSS Grid Layouts

<https://css-grid-layout-generator.pw/>

37. Border radius



Fancy Border Radius Generator
A visual generator to build organic looking shapes with the help of CSS3 border-radius property
<https://9elements.github.io/fancy-border-radius/>

38. Neumorphism



Neumorphism/Soft UI CSS shadow generator
CSS code generator that will help with colors, gradients and shadows to adapt this new design trend or discover its possibilities.
<https://neumorphism.io/>

39. Shape maker

bennettfeely.com/clippy

40. Glassmorphism



Glassmorphism
CSS GENERATOR

Glassmorphism - simple CSS generator

Glassmorphism is a unified name for the popular Frosted Glass aesthetic.

<https://glassmorphism.com>

◆ Design Inspiration

41. CSS design awards

cssdesignawards.com

42. Awwwards



Best CSS3 Websites | Web Design Inspiration

Fantastic CSS3 Website Designs for Inspiration. Selection of Awwwards winning CSS3 websites. CSS3 is a powerful tool for web designers to enhance the appearance of a website.

<https://awwwards.com/websites/css3/>

43. CSS nectar



CSS Gallery for Web Design Inspiration - CSS Nectar

CSS Nectar is a css website design showcase for web designers and developers. Every day we select the best of the web design and add it to our gallery. Submit or suggest a website.

<https://cssnectar.com>

44. Design modo



Examples of CSS Website Designs for Inspiration - Designmodo
Some of the benefits of having the CSS websites designs are that it gives a good control over the layout. Here are a few good examples.
<https://designmodo.com/css-website-designs/>

45. CSS winner

csswinner.com

- ◆ GitHub Repo

46. Awesome CSS



awesome-css-group/awesome-css
:art: A curated contents of amazing CSS :). Contribute to awesome-css-group/awesome-css development by creating an account on GitHub.
<https://github.com/awesome-css-group/awesome-css>

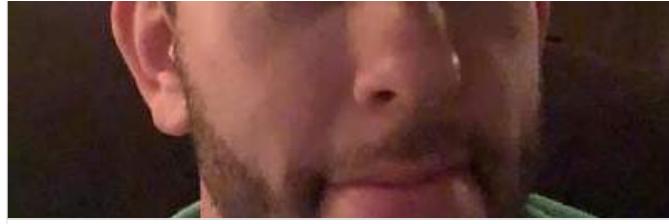
47. 30 seconds of CSS



30-seconds/30-seconds-of-css
Short CSS code snippets for all your development needs - 30-seconds/30-seconds-of-css
<https://github.com/30-seconds/30-seconds-of-css>

48. CSS Protips





AllThingsSmitty/css-protips

A collection of tips to help take your CSS skills pro - AllThingsSmitty/css-protips

<https://github.com/AllThingsSmitty/css-protips>

49. Awesome flexbox



afonsopacifer/awesome-flexbox

:eyeglasses: A curated list of CSS Flexible Box Layout Module or only Flexbox. - afonsopacifer/awesome-flexbox

<https://github.com/afonsopacifer/awesome-flexbox>

◆ Cheat Sheets

50. I love Coding

<https://ilovecoding.org/blog/htmlcss-cheatsheet>

51. Dev Hints

The screenshot shows a comprehensive CSS selector reference. It includes sections for:

- Selectors:** Lists basic CSS selector types like class, id, and attribute.
- Combinators:** Details various ways to combine multiple selectors, such as .parent + child, ~ sibling, and > child.
- Attribute selectors:** Shows how to target elements based on attributes, including [attribute="value"] and [attribute~=value].
- Pseudo-classes:** Lists pseudo-classes like :target, :first-child, and :nth-child(odd).
- Pseudo-class variations:** Lists specific variations like :first-of-type, :last-of-type, and :nth-of-type(2n+1).

CSS cheatsheet

margin, padding, border · div, .class, #id, [attr] · font, background · display: block, inline, flex · Selectors · Properties · One-page guide to CSS

<https://devhints.io/css>

52.

The screenshot shows the CSS Cheat Sheet website interface. It features several sections: 'RESET CSS' with code snippets; 'Media Queries' with a snippet for 'device-width'; 'Text Shadow' with a preview and generator; 'Background' with a snippet for 'background-size: cover'; 'Gradient' with a snippet for 'background: linear-gradient'; 'Transforms' with a generator for 'translateX'; and 'Button Generator' with a snippet for 'button:disabled'. A sidebar on the right includes links for 'HTML', 'JS', 'AMP', 'SEO', and 'CSS'.

Online Interactive CSS Cheat Sheet

CSS Cheat Sheet contains the most common style snippets: CSS gradient, background, button, font-family, border, radius, box and text shadow generators, color picker and more.

<https://htmlcheatsheet.com/css/>

53. Grid



GRID: A simple visual cheatsheet for CSS Grid Layout

Learn all about the properties available in CSS Grid Layout through simple visual examples.

<https://grid.malven.co/>

54. CSS Grid

The screenshot shows the Grid Cheatsheet website. It features a grid diagram with numbered cells (1-9) and labeled areas (a-d). Above the grid, there are labels for grid lines: 1, 2/2, 1/2/3/4, a, b, c, d, b-start, and b-start/main-begin. Below the grid, there are two boxes: one for 'b-start/main-begin/c-end' and another for 'b-start/main-begin/e-end/d-end'. To the right of the grid, there is a block of CSS code:

```
.parent {  
  display: grid;  
  grid-template-columns: [aside-begin] 1fr [aside-end main-begin]  
  1fr [main-end aside2-begin] 1fr  
  [aside2-end];  
  grid-template-rows: repeat(3,  
  1fr);  
  grid-template-areas:  
    "a a d"  
    "b b d"  
    "c c d";  
}
```

Grid Cheatsheet

Demos for W3C CSS Grid Specification <https://www.w3.org/TR/css-grid-1>

<https://yoksel.github.io/grid-cheatsheet/>

55. Flex cheat sheet

The flex-flow property is a shorthand for setting the `flex-direction` and `flex-wrap` properties, which together define the flex container's main and cross axes.

Note that the `flex-flow` directions are writing mode sensitive. In vertical Japanese, for example, a `row` flex container

Flex Cheatsheet
Demos for W3C CSS Flexbox Specification <http://www.w3.org/TR/css-flexbox-1/>
<https://yoksel.github.io/flex-cheatsheet/>

56.

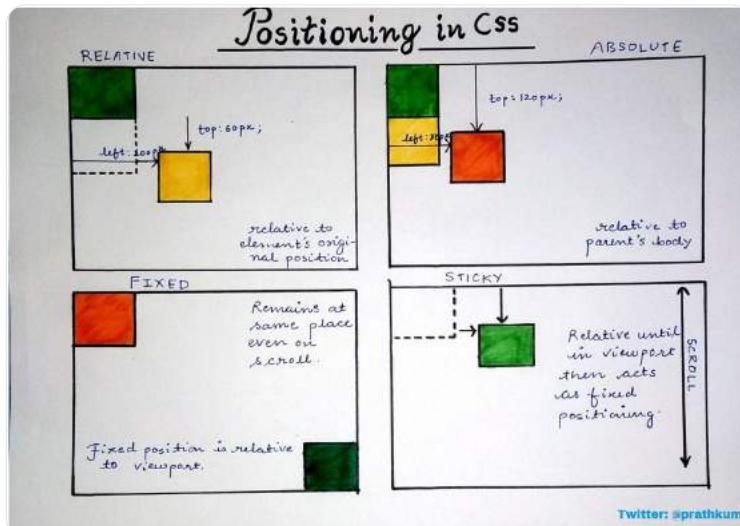


FLEX: A simple visual cheatsheet for flexbox

Learn all about the properties available in flexbox through simple visual examples.

<https://flexbox.malven.co/>

57. Position



- ◆ Validators

58. CSS Validators

<http://beautifytools.com/css-validator.php>

59. CSS Lint

<csslint.net>

60. Purify CSS

PurifyCSS Online
Remove unused CSS selectors from your stylesheets

Insert your website URL below and find out by how much of your CSS file size can be reduced

Website URL: Clean up CSS

Results:

<https://www.peterbagi.de/wp-content/themes/peterbagi/css/bootstrap.min.css>
before: **114.56 KB** after: **26.15 KB** compressed to: **22.83%** [show clean css code](#)

<https://www.peterbagi.de/wp-content/themes/peterbagi/css/flexslider.css>

PurifyCSS Online - Remove unused CSS
PurifyCSS is a website performance optimization tool that scans your HTML & JS source code, removes the unused CSS selectors and reduces the file size by up to 90%
<https://purifycss.online/>

- ◆ Chrome Extension

61. CSS Peeker



CSS Peeker
Extract CSS and build beautiful styleguides.
<https://chrome.google.com/webstore/detail/css-peeker/mbnbehikldjhnfefhhnaidhjhoofh...>

62. Pesticide





Pesticide for Chrome with autoupdate

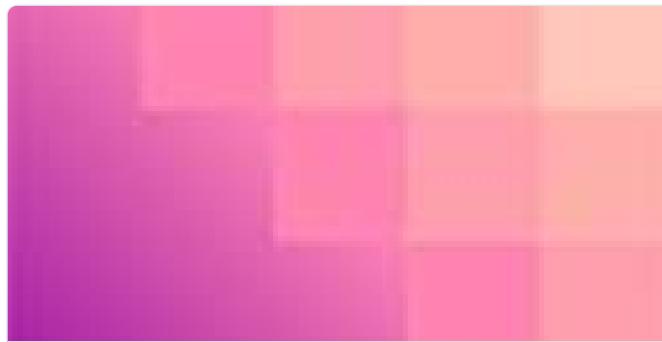
This extension inserts (with auto reload) the Pesticide CSS into the current page, outlining each element.

<https://chrome.google.com/webstore/detail/pesticide-for-chrome-with/eipbgplchlidkojm...>

63. Stylebot

<https://chrome.google.com/webstore/detail/stylebot/oiaejjdbmkiecgbjefoepgmdaleoha?hl=en>

64. Perfect Pixel

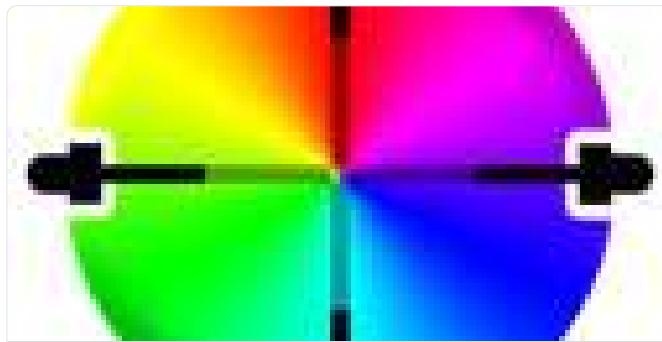


PerfectPixel by WellDoneCode (pixel perfect)

This extension helps develop your websites with pixel perfect accuracy!

<https://chrome.google.com/webstore/detail/perfectpixel-by-welldonec/dkaagdgjmgdmb...>

65. Colorpick eyedropper

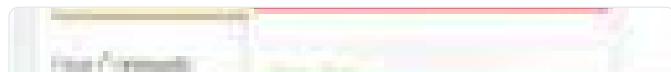


ColorPick Eyedropper

A zoomed eyedropper & color chooser tool that allows you to select color values from webpages and more.

<https://chrome.google.com/webstore/detail/colorpick-eyedropper/ohcpnigalekghcmgcd...>

66. CSS viewer





CSSViewer

A simple CSS property viewer.

<https://chrome.google.com/webstore/detail/cssviewer/ggfgjbjpiheegeflicemofobhmofgc...>

67. CSS Scan



CSS Scan

The fastest and easiest way to check, copy and edit CSS.

<https://chrome.google.com/webstore/detail/css-scan/gieabiemgggnpnminflinemaickipbe...>

- ◆ YouTube

68. Free code camp



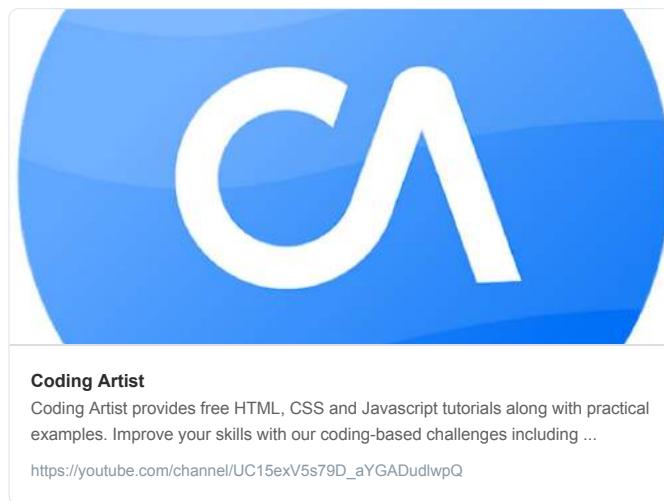
<https://www.youtube.com/embed/1Rs2ND1ryYc>

69. Traversy media



<https://www.youtube.com/embed/yfoY53QXEnI>

70. Coding artist

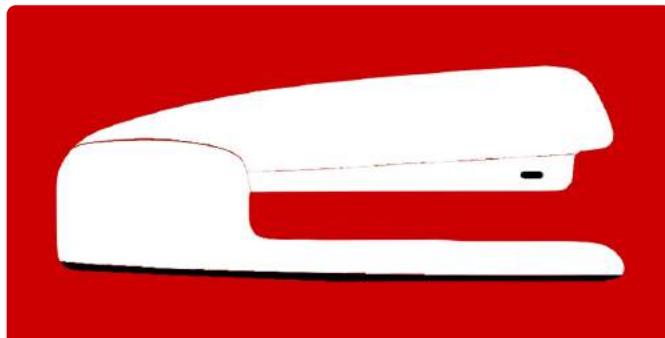


71. Edureka



https://www.youtube.com/embed/3_9znKVNe5g

72. Red Stapler



Red Stapler

How to and tutorial videos of cool CSS effect, Web Design ideas, JavaScript libraries, Node.js, HTML, dev tips and much more! New videos upload weekly. Monday...

<https://www.youtube.com/channel/UCRthRrv06q1iOl86-tTKJhg>

- ◆ Animation libraries

73. Animate.css



Animate.css | A cross-browser library of CSS animations.
Animate.css is a library of ready-to-use, cross-browser animations for you to use in your projects. Great for emphasis, home pages, sliders, and attention-guiding hints.

<https://animate.style/>

74. OBNOXIOUS.css
tholman.com/obnoxious/

75. CSShake
elrumordelaluz.github.io/cssshake/

76. Anim XYZ



AnimXYZ
The first composable CSS animation toolkit
<https://animxyz.com/>

77. Animista



Animista - CSS Animations on Demand
Animista is more than just a CSS animation library. It is a place where you can play with a collection of ready to use CSS animations, tweak them and download only those you will actually use.
<https://animista.net/>

78. Hover.CSS



Hover.css - A collection of CSS3 powered hover eff...
<https://ianlunn.github.io/Hover/>

79. Magic

Magic CSS

Magic CSS animations
Magic CSS are a set of simple animations to include in your web or app project's
<https://www.minimamente.com/project/magic/>

- ◆ Threads

80. CSS arts

Pratham 🎨🚀
@Prathkum

Create your first CSS art ✨🎨

A thread 🧵
It contains all resources you need in order to create your first CSS art 👇

11:27 AM · Sep 17, 2020

1.6K 448 Copy link to Tweet

81. Grid

Pratham 🎨🚀
@Prathkum

A complete beginner's guide to CSS Grid layout 👇

Thread 🧵

The diagram shows a 3x3 grid of colored cells labeled 1 through 9. Cell 1 is at the top center. Cells 2, 3, and 4 form a row below it. Cells 5 and 6 are in the middle-left column. Cell 7 is in the bottom-middle column. Cells 8 and 9 are at the bottom right.

9:20 AM · Mar 24, 2021

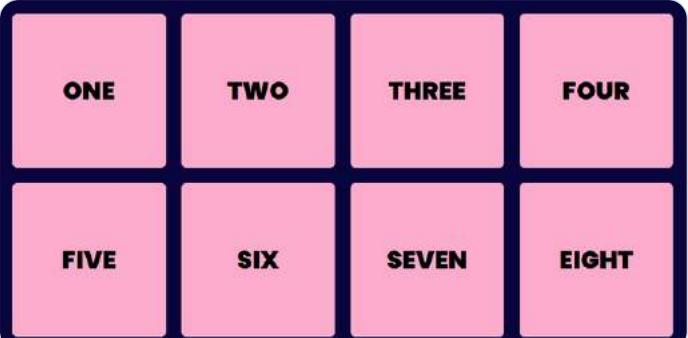
2.4K 684 Copy link to Tweet

82. Flex

 Pratham 🚀
@Prathkum

Everything you need to know about CSS Flexible Box Layout 

THREAD 



6:51 AM · Feb 17, 2021 

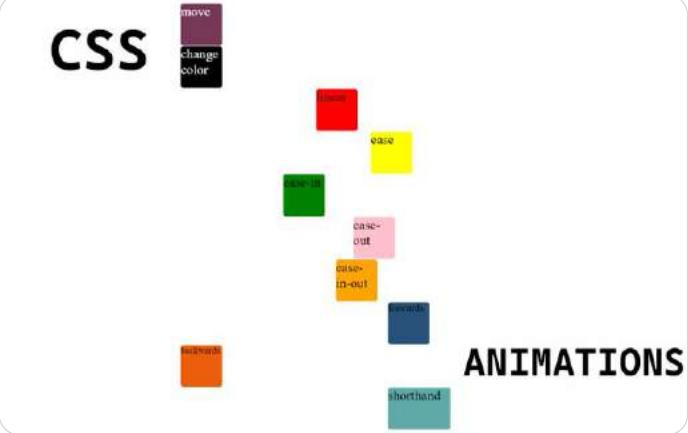
 1.3K  336  Copy link to Tweet

83. CSS Animations

 Pratham 🚀
@Prathkum

A quick start guide to CSS animations 

Thread 

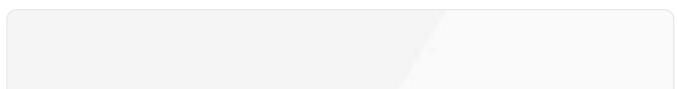


6:47 AM · Mar 1, 2021 

 721  186  Copy link to Tweet

- ◆ Websites

84. CSS reference





Free visual guide to the most popular **CSS3** properties

CSS Reference

CSS Reference is a free visual guide to CSS. It features the most popular properties, and explains them with illustrated and animated examples.

<https://cssreference.io/>

85. Learn to code

learn.shayhowe.com

86. CSS layouts

learnlayout.com

87. CSS tutorial



CSS Tutorial

Beginners CSS Tutorial For Web Designers.

[https://www.csstutorial.net/](http://www.csstutorial.net/)

89. CSS tricks



CSS-Tricks

Daily articles about CSS, HTML, JavaScript, and all things related to web design and development.

- ♦ Learn by playing

90. Grid Garden



Grid Garden

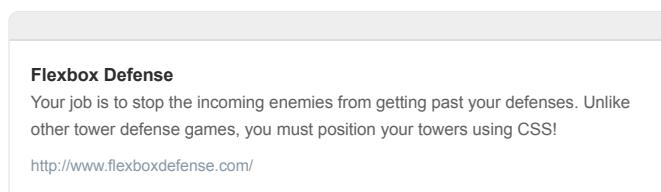
A game for learning CSS grid layout

<https://cssgridgarden.com/>

91. Unfold

rpl.github.io/unfold/

92. Flexbox defense



93. CSS Diner



94. Flexbox froggy

The screenshot shows a game titled "FLEXBOX FROGGY" with a progress bar at "Level 5 of 24". On the left, there's a code editor window containing the following CSS:

```
1<div>
2  <div>
3    display: flex;
4    align-items: flex-end
5  </div>
6</div>
```

On the right, three frogs (green, yellow, red) are lined up horizontally. Below them is a pond with three circular lily pads (green, yellow, red) arranged vertically.

Flexbox Froggy
A game for learning CSS flexbox
<https://flexboxfroggy.com/>

- ◆ [Templates](#)

95. Free CSS templates

<https://www.free-css.com/free-css-templates>

96. Templates

templated.co

97. Tooplate

The screenshot shows the homepage of Tooplate. The main feature is a large, stylized logo with the word "tooplate" where the "o" is orange and the rest is teal. Below the logo is the text "free templates for your websites".

Free HTML CSS Templates by tooplate
We provide free HTML CSS website templates for everyone. You can immediately download and use our templates for any purpose.
<https://www.tooplate.com/free-templates>

98. Nice page

The screenshot shows a dark-themed template page. At the top, there's a header section with social media icons (Facebook, Twitter, Instagram, LinkedIn, Pinterest) and a search bar. Below the header is a large, dark image of a landscape.

6500+ CSS Templates | Free CSS Templates
Free Download the biggest collection of CSS Templates 2021. Create your own CSS Template with the best web design software

99. Templatemo



548+ Free HTML CSS Templates by TemplateMo

Download 548+ free HTML CSS website templates that included 140+ responsive Bootstrap themes from templatemo and use them for your sites

<https://templatemo.com/>

100. Add one take this thread to 100 😊

It took me 2 hours to thread all these resources 😊

Share it with your connections ❤️

While writing this thread I was expecting around 3.5 - 4K likes. This is the largest resource thread ever shared on Twitter

Can we take it there? 😊

• • •



Pratham 🎨🚀 @Prathkum

24 Mar • 7 tweets • [Prathkum/status/1374726453649100809](#)

Tr

If you know CSS then you can use these amazing generators and save your time

Thread 🧶 ↗

1 CSS Box Shadow Generator

- Generate CSS3 Box Shadow code for your Div, Frame, Buttons or any other HTML element with Outline, and Inset (inner) type shadow effects

🔗 cssboxshadow.com



2 Glassmorphism

- Generate glassmorphic design easily

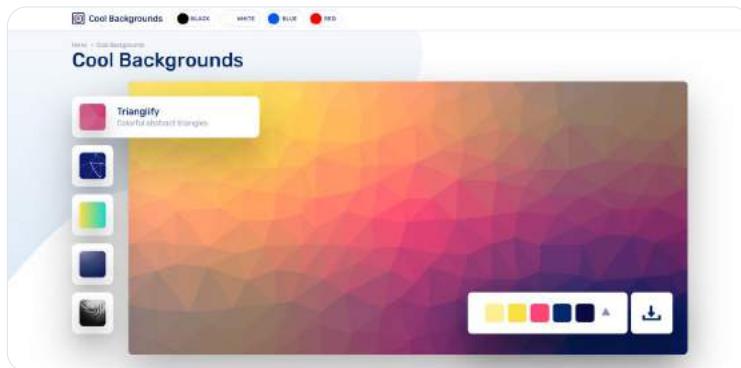
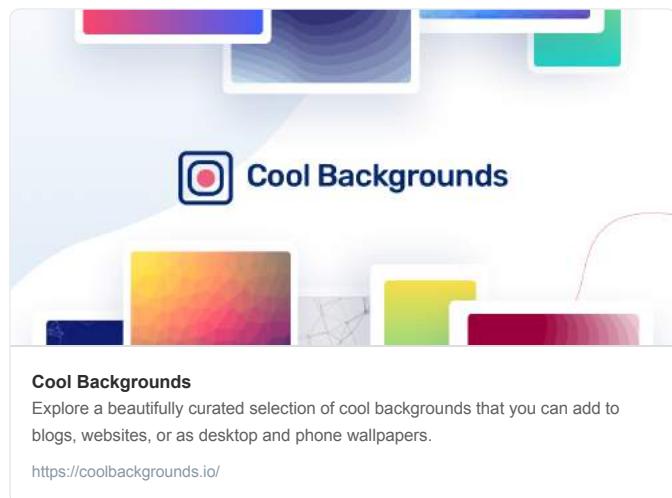
🔗 glassgenerator.netlify.app



3 Cool Backgrounds

- Explore a beautifully curated selection of cool backgrounds that you can add to your next project

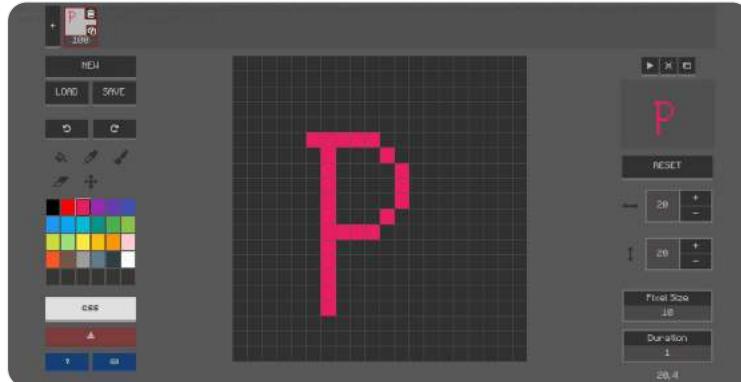
🔗



4 Pixel art

- Create CSS pixel art, export the results to CSS and download them.

🔗



5 SVG waves

- It's not a CSS generator but very handy tool. You can create the layers of waves and simply export either in SVG or PNG



Svg Wave - A free and beautiful gradient SVG wave Generator.

SVG Wave is a minimal svg wave generator with lot of customization. It lets you ability to generate and export pngs and svgs of beautiful waves. SVG wave also lets you layer multiple waves. Create SV...

<https://svgwave.in/>



6 □ Color theme generator

- A great tool for CSS dark/light theme created by [@tenphi](#)



CSS Theme Generator v1.0.0

open source • light/dark scheme • normal/high contrast
Based on [Numl.Design](#) theme generator by [@tenphi](#) and [HSLuv](#) color space by [@benjamin](#)

Colors Properties

Single tone • Duo tone

Main Hue: - 242 + Insert

Accent Hue: - 0 + Insert

Saturation: - 75 + Insert

Use pastel palette:

Type: Main Tint Tone Swap Special

Contrast: Soft Normal Strong

Emphasizing: None Normal Bold Only for Tone and Swap colors

Preview

Numl.Design Open REEL

Paragraph text Special paragraph text

Basic Card Clear Card

Basic badge Special badge

Basic button states

Default Hover Pressed

Toggled Focused

Disabled

Special button states

CSS Color Theme Generator by Numl.Design

Open Source CSS Color Theme Generator for creating lovely toned themes with dark scheme and high contrast mode.

<https://numl.design/theme-generator>



• • •



Pratham @Prathkum

20 Apr · 25 tweets · [Prathkum/status/1384464876450455552](#)

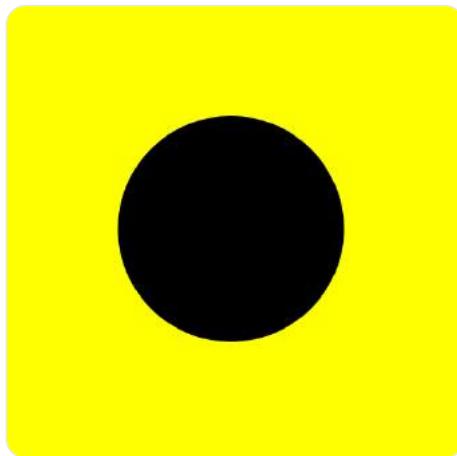
Tr

Let's create 22 shapes and figures using pure CSS 🎨



1. Circle 🟡

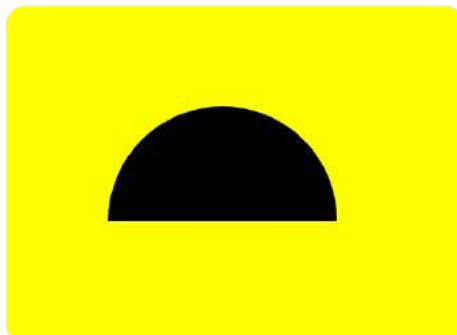
- Pretty simple, we just need to make a square and apply the border-radius 50% in order to give it a circular shape



```
● ● ●  
.circle {  
    height: 200px;  
    width: 200px;  
    border-radius: 50%;  
    background-color: black;  
}
```

2. Semi-circle 🟢

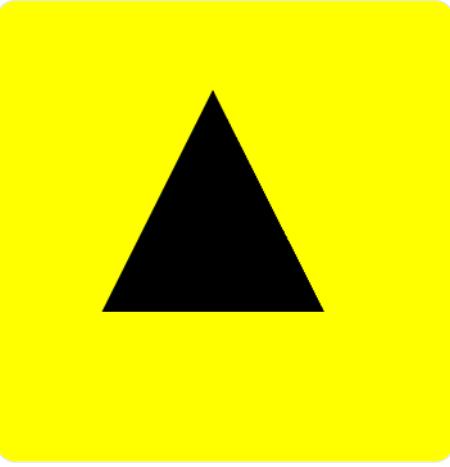
- Create a rectangle
- Apply border radius top left and top right same as height of the rectangle



```
● ● ●  
.semi-circle {  
    height: 100px;  
    width: 200px;  
    border-radius: 100px 100px 0 0;  
    background-color: black;  
}
```

3. Triangle ▲

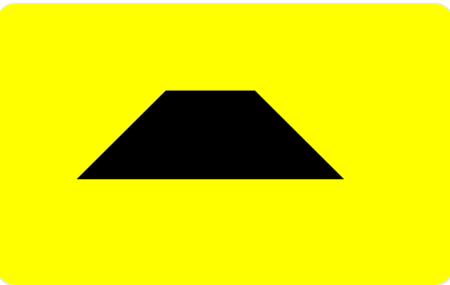
- Creating a triangle is little bit tricky
- Set height and width as zero
- To make this, we draw a solid border and make the side border transparent



```
.triangle {  
    width: 0;  
    height: 0;  
    border-bottom: 200px solid black;  
    border-left: 100px solid transparent;  
    border-right: 100px solid transparent;  
}
```

4. Trapezium

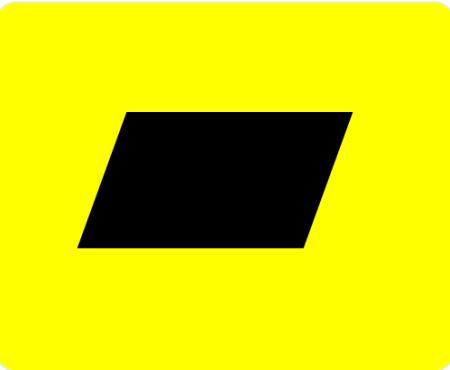
- Same as triangle but in this case we need to set some width



```
.trapezium {  
    width: 100px;  
    height: 0;  
    border-bottom: 100px solid black;  
    border-left: 100px solid transparent;  
    border-right: 100px solid transparent;  
}
```

5. Parallelogram

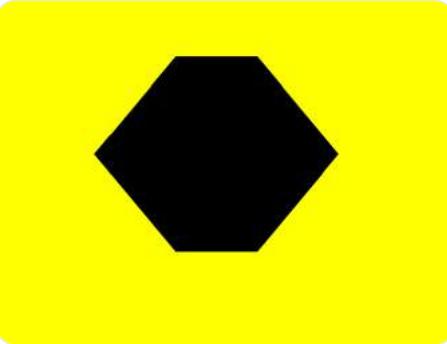
- Create a rectangle
- Apply skew in order to tilt it



```
.parallelogram {  
    height: 120px;  
    width: 200px;  
    background: black;  
    transform: skewx(-20deg);  
}
```

6. Hexagon

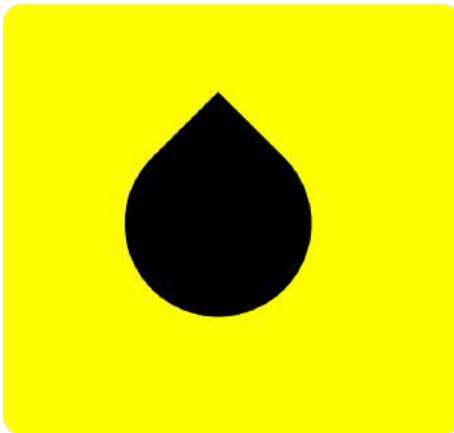
- Creating hexagon is very easy
- We need to make two trapeziums of same size but make sure that other trapezium should be upside down
- Align them perfectly



```
● ● ● .hexagon { position: relative; width: 100px; height: 0; border-top: 100px solid black; border-left: 100px solid transparent; border-right: 100px solid transparent; } .hexagon::before { content: ""; position: absolute; width: 100px; height: 0; border-bottom: 120px solid black; border-left: 100px solid transparent; border-right: 100px solid transparent; left: -10px; top: -24px; }
```

7. Drop 💧

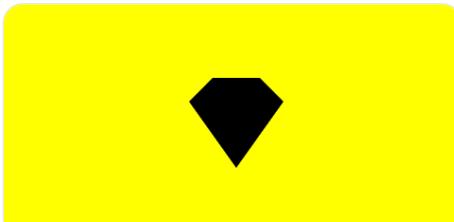
- Create a square
- Apply 50% border-radius to all the sides except one side
- Rotate in such a manner so that tip comes to top



```
● ● ● .drop { height: 200px; width: 200px; background: black; border-radius: 0 50% 50% 50%; transform: rotate(45deg); }
```

8. Diamond 💎

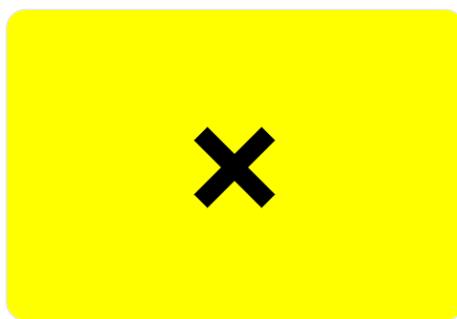
- Combination of Trapezium and triangle



```
● ● ● .diamond { border-style: solid; border-color: transparent transparent black transparent; border-width: 0 25px 25px 0; height: 0; width: 50px; position: relative; } .diamond::after { content: ""; position: absolute; top: 25px; left: -25px; width: 0; height: 0; border-style: solid; border-color: black transparent transparent transparent; border-width: 0 50px 50px 0; }
```

9. Cross ✕

- Create two rectangles
- Place them over each other vertically and horizontally
- Rotate 45deg

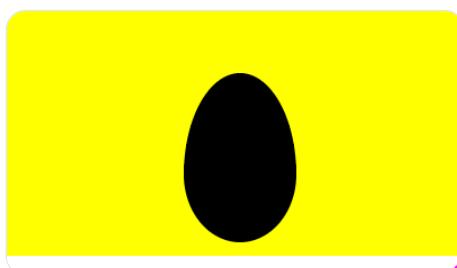


```
.cross {
  background: black;
  height: 100px;
  position: relative;
  width: 20px;
  transform: rotate(45deg);
}

.cross:after {
  background: black;
  content: "";
  height: 20px;
  left: -40px;
  position: absolute;
  top: 40px;
  width: 100px;
}
```

10. Egg 🥚

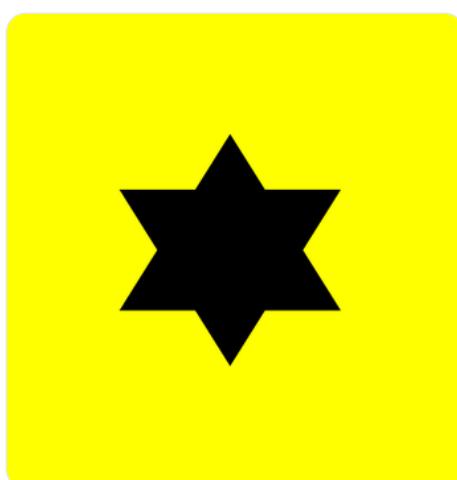
- Using advance border-radius technique



```
.egg {
  display: block;
  width: 180px;
  height: 150px;
  background-color: black;
  border-radius: 50% 50% 50% 50% / 66% 66% 44% 44%;
}
```

11. Star ✨

- Create two triangle upside down
- Align second triangle in the middle of the first one

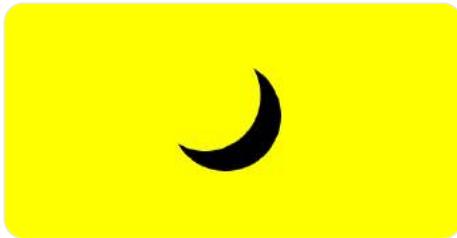


```
.star {
  position: relative;
  width: 0;
  height: 0;
  border-bottom: 160px solid black;
  border-left: 100px solid transparent;
  border-right: 100px solid transparent;
}

.star::before {
  content: "";
  position: absolute;
  left: -100px;
  top: 50px;
  width: 0;
  height: 0;
  border-top: 160px solid black;
  border-left: 100px solid transparent;
  border-right: 100px solid transparent;
}
```

12. Moon 🌙

- Transparent background
- Apply box shadow



```
.moon {  
    width: 120px;  
    height: 120px;  
    border-radius: 50%;  
    box-shadow: 22px 22px 0 0 black;  
}
```

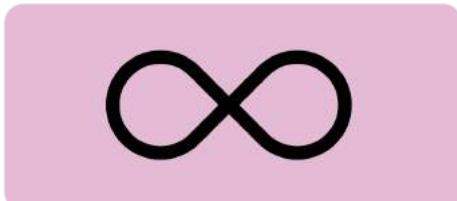
13. Coffee ☕



```
.coffee {  
    position: relative;  
    background: black;  
    height: 180px;  
    width: 180px;  
    border-radius: 50%;  
}  
  
.coffeebefore {  
    content: "*";  
    position: absolute;  
    height: 180px;  
    width: 180px;  
    border-radius: 50%;  
    background: gray;  
    background-size: 100%;  
    background-position: 50%;  
    top: -30px;  
    left: -30px;  
    filter: blur(4px);  
    box-shadow: 0px -2px 4px -1px gray, 2px -4px 4px -2px gray, -2px -4px 4px gray, 3px 0px 4px gray, 2px -3px 4px -2px gray;  
}  
  
.coffeearlier {  
    content: "*";  
    position: absolute;  
    height: 20px;  
    width: 20px;  
    border-radius: 50%;  
    background: gray;  
    background-size: 100%;  
    background-position: 50%;  
    top: -20px;  
    left: -20px;  
    filter: blur(4px);  
    box-shadow: 0px -2px 4px -1px gray, 2px -4px 4px -2px gray, -2px -4px 4px gray, 3px 0px 4px gray, 2px -3px 4px -2px gray;  
}  
  
.stand {  
    position: absolute;  
    height: 20px;  
    width: 20px;  
    background-color: black;  
    border-radius: 50% 50% 50% 50%;  
    right: -10px;  
    top: -10px;  
}
```

14. Infinity Symbol ∞

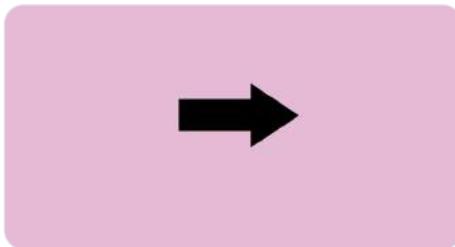
It may look tough but trust me I created this symbol using border-radius only



```
.infinity {  
    position: relative;  
}  
  
.infinity::before {  
    position: absolute;  
    height: 120px;  
    width: 120px;  
    border: 20px solid black;  
    content: "*";  
    border-radius: 50% 50% 50% 50%;  
    transform: rotate(45deg);  
}  
  
.infinity::after {  
    position: absolute;  
    height: 120px;  
    width: 120px;  
    border: 20px solid black;  
    content: "*";  
    border-radius: 50% 50% 50% 50%;  
    transform: rotate(45deg);  
    left: -20px;  
}
```

15. Arrow ➔

Combination of rectangle and a triangle

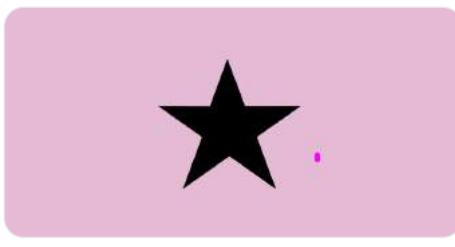


```
.arrow {
    height: 40px;
    width: 100px;
    background: black;
    position: relative;
}

.arrow::before {
    position: absolute;
    content: "";
    border-top: 40px solid black;
    border-right: 40px solid transparent;
    border-left: 40px solid transparent;
    transform: rotate(-90deg);
    right: -60px;
    top: -10px;
}
```

16. Star ⭐

Combination of two triangles



```
.star {
    position: relative;
    display: block;
    width: 80px;
    height: 80px;
    border-right: 100px solid transparent;
    border-bottom: 70px solid black;
    border-left: 100px solid transparent;
    transform: rotate(35deg);
}

.star:before {
    border-bottom: 80px solid black;
    border-left: 30px solid transparent;
    border-right: 30px solid transparent;
    position: absolute;
    height: 0;
    width: 0;
    top: -45px;
    left: -65px;
    display: block;
    content: '';
    transform: rotate(-35deg);
}

.star:after {
    position: absolute;
    display: block;
    top: 3px;
    left: -30px;
    width: 8px;
    height: 8px;
    border-right: 100px solid transparent;
    border-bottom: 70px solid black;
    border-left: 100px solid transparent;
    transform: rotate(-70deg);
    content: '';
}
```

17. Magnifying glass 🔎

You guessed it right! Its just a simple circle and small rectangle



```
.magnifying-glass {  
    height: 100px;  
    width: 100px;  
    border: 20px solid black;  
    border-radius: 50%;  
    position: relative;  
}  
  
.magnifying-glass::before {  
    height: 80px;  
    width: 20px;  
    background: black;  
    position: absolute;  
    content: "";  
    transform: rotate(-45deg);  
    right: -40px;  
    bottom: -60px;  
}
```

18. Heart ❤

- Two rectangles with top round border-radius



```
.star {  
    position: relative;  
    display: block;  
    width: 0px;  
    height: 0px;  
    border-right: 100px solid transparent;  
    border-bottom: 70px solid black;  
    border-left: 100px solid transparent;  
    transform: rotate(35deg);  
}  
  
.star:before {  
    border-bottom: 80px solid black;  
    border-left: 30px solid transparent;  
    border-right: 30px solid transparent;  
    position: absolute;  
    height: 0;  
    width: 0;  
    top: -45px;  
    left: -65px;  
    display: block;  
    content: "";  
    transform: rotate(-35deg);  
}  
  
.star:after {  
    position: absolute;  
    display: block;  
    top: 3px;  
    left: -105px;  
    width: 8px;  
    height: 0px;  
    border-right: 100px solid transparent;  
    border-bottom: 70px solid black;  
    border-left: 100px solid transparent;  
    transform: rotate(-70deg);  
    content: "";  
}
```

19. Pacman



```
.pacman {  
    width: 80px;  
    height: 60px;  
    border-right: 60px solid transparent;  
    border-top: 60px solid black;  
    border-left: 60px solid black;  
    border-bottom: 60px solid black;  
    border-top-left-radius: 60px;  
    border-top-right-radius: 60px;  
    border-bottom-left-radius: 60px;  
    border-bottom-right-radius: 60px;  
}
```

20. 8 Point Star ⭐

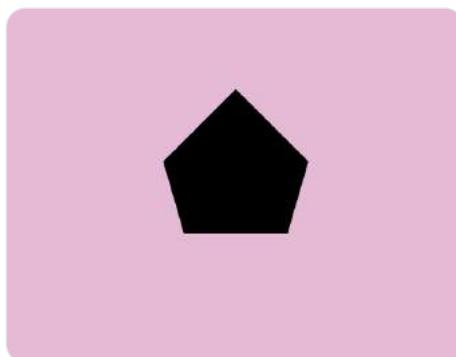
Two squares overlapping each other



```
.star-8 {
  background: black;
  width: 80px;
  height: 80px;
  position: relative;
  text-align: center;
  transform: rotate(20deg);
}
.star-8::before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  height: 80px;
  width: 80px;
  background: black;
  transform: rotate(135deg);
}
```

21. Pentagon ♦

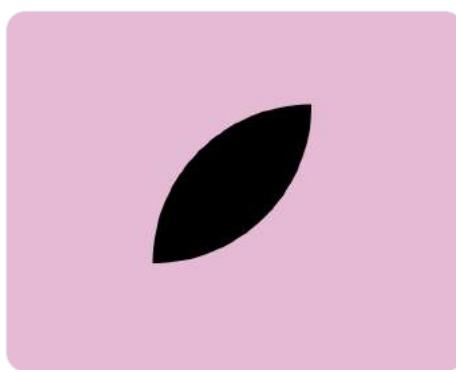
- Create a trapezium
- Create a triangle
- Merge them both



```
.pentagon {
  position: relative;
  width: 100px;
  border-left: 20px solid transparent;
  border-right: 20px solid transparent;
  border-top: 70px solid black;
}
.pentagon::before {
  content: "";
  position: absolute;
  width: 0;
  border-left: 70px solid transparent;
  border-right: 70px solid transparent;
  border-bottom: 70px solid black;
  top: -140px;
  left: -20px;
}
```

22. Leaf 🌱

- Just the matter or border-radius



```
.leaf {
  height: 200px;
  width: 200px;
  background: black;
  border-radius: 200px 0 200px 0;
```

I hope you will find it helpful, if so, share it ❤

Peace out 😊

In general, these shapes or CSS arts are of no use in practical terms, but

- It can increase your CSS learning process up to 10 times 🚀
- Helps you build better understanding 🧠

• • •



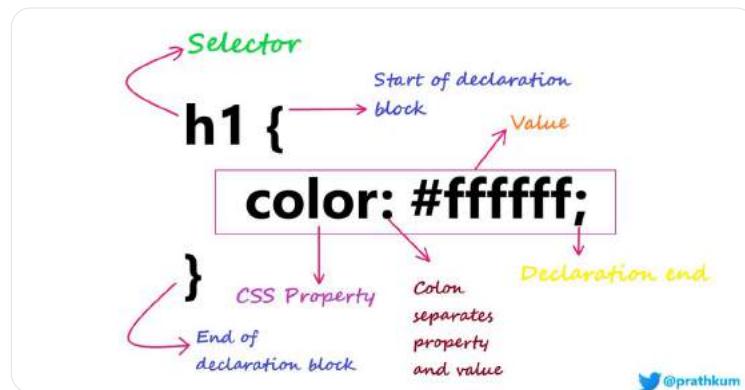
Pratham @Prathkum

15 May · 17 tweets · [Prathkum/status/1393521799187533833](https://twitter.com/Prathkum/status/1393521799187533833)

Tr

Are you planning to learn CSS for adding some cool styling in your website?

In this thread, I'm covering all the basics and proper plan you need to know in order to getting started with CSS.



We need to know the importance of CSS first. Why CSS?

Well I guess CSS is the only language that add some visuals in website. Without CSS, Without it, websites would still be plain text on white backgrounds.

{ 02 / 17 }

Let's compare a website with and without CSS

Image 1: With CSS

Image 2: Without CSS

{ 03 / 17 }



I this should be enough to explain you why CSS is so important. So lets start

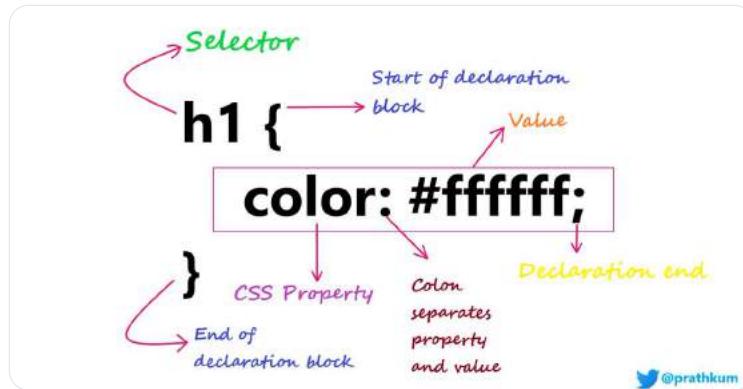
You can gain some decent amount of CSS knowledge in 30 days. And after that you will definitely able to create some good looking website. Let's see how 

{ 04 / 17 }

First things first, you need to learn basic stuff

- How to link your CSS files with HTML file
- How to select elements so that you can add styling

{ 05 / 17 }



Color and background are the two properties you can start learning with. Although background is a shorthand property and its deep. So intially you can start with background-color property. Later on, you can explore more

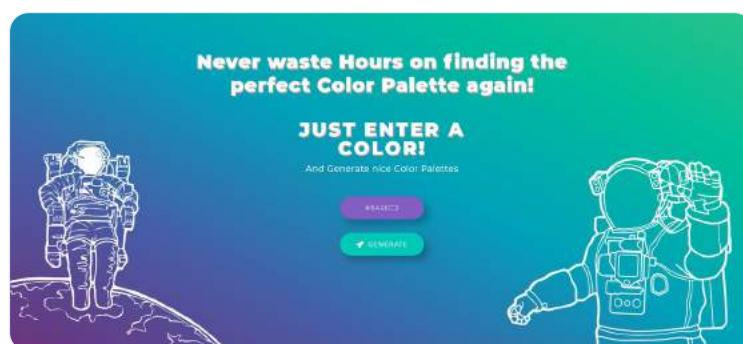
{ 06 / 17 }

There are a lot of great color palette out there using which you can generate pleasant color schemes

Check this great tool for generating accessible colors. Just enter a color and generate nice color palettes

 mycolor.space

{ 07 / 17 }



Box model is the next important concept you need to learn. As everything is a box in web development.

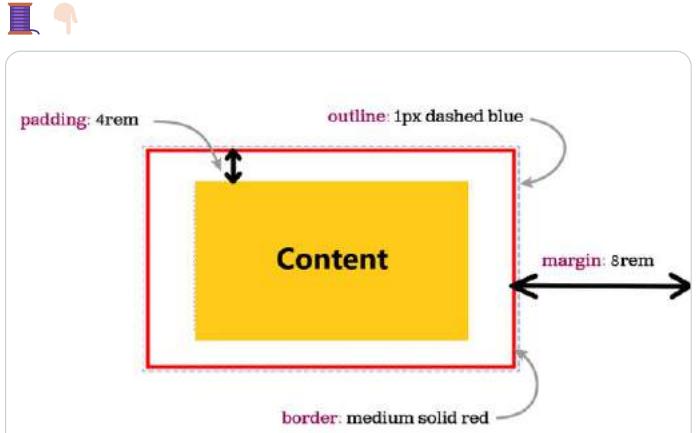
Check out this detailed thread if you want to learn about box model from my thread



Pratham
@Prathkum

Box Model

Everything on a website is a box. Therefore it is quite important to understand the box model concept in web development but it is confusing. Let's see some hidden facts about it in this thread



The diagram illustrates the Box Model with the following components and their properties:

- Content**: The innermost yellow area.
- padding: 4rem**: The space between the Content and the border.
- outline: 1px dashed blue**: A decorative border around the Content.
- border: medium solid red**: The main border of the box.
- margin: 8rem**: The space between the border and the outer edge of the box.

10:38 AM · May 14, 2021

1 ①

Heart 628 Comment 28 Copy link to Tweet

{ 08 / 17 }

Moving forward, typography is an essential thing of web page. A good font can make your webpage and establish a strong visual hierarchy, provide a graphic balance to the website, and set the product's overall tone

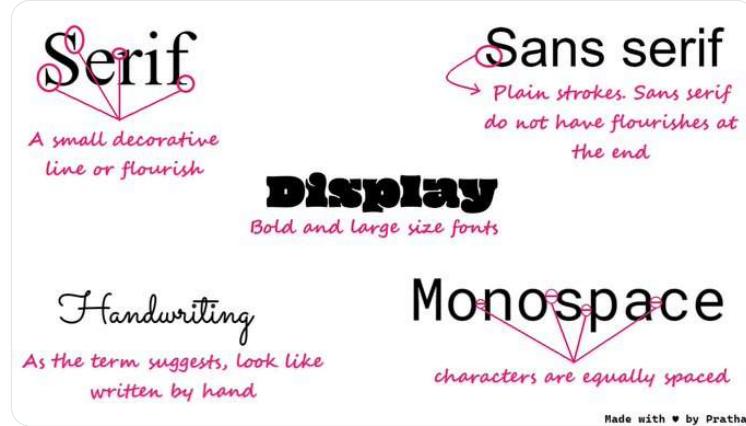
You can add free fonts from Google's official site

{ 09 / 17 }

There are five basic classifications of fonts:

1. serif
2. sans serif
3. script
4. monospaced
5. display

{ 10 / 17 }



Next thing we have in CSS is positioning concept, its yet another powerful and confusing. But don't worry I am here to make it easy for you. You can learn it from anywhere but I also have a thread on it

 **Pratham** @Prathkum 

Everything you need to know about CSS position property

Thread 



8:02 AM · Apr 4, 2021 

 1.2K  46  Copy link to Tweet

{ 11 / 17 }

Up to this point you have some decent knowledge of styling your website. It would be great if you learn about a layout system now.

Flex and Grid

Grid is little bit tough to master but flex isn't

{ 12 / 17 }

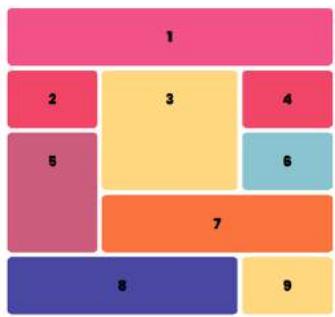
You don't even need to learn about Flex if you know Grid layout. Because you can achieve the Flex layout using Grid layout.



Pratham
@Prathkum

A complete beginner's guide to CSS Grid layout 

Thread 



The diagram illustrates a 3x3 grid layout using CSS Grid. The grid consists of nine cells, each containing a number from 1 to 9. The layout is as follows:

1		
2	3	4
5		6
	7	
8		9

Cell 1 is at the top center. Cell 2 is in the top-left corner. Cell 3 is in the middle column of the top row. Cell 4 is in the top-right corner. Cell 5 is in the middle-left column of the second row. Cell 6 is in the middle-right column of the second row. Cell 7 is in the bottom-left corner. Cell 8 is in the bottom-center. Cell 9 is in the bottom-right corner.

9:20 AM · Mar 24, 2021 

 3.7K  93  Copy link to Tweet

{ 13 / 17 }

Moving forward, there are million of device on which user will see your website. In order to deliver best experience, you need to make your website responsive. That is also handled by CSS.

{ 14 / 17 }

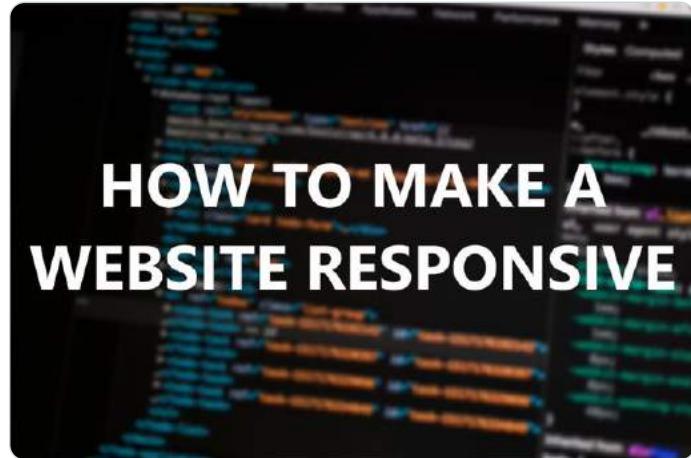
Few days back I wrote a thread on responsive web design. Give it read and learn about responsiveness in almost no time. This is a visual guide as well



Pratham
@Prathkum



Complete guide to Responsive Web Design



10:45 AM · Apr 10, 2021



2.2K 52 Copy link to Tweet

{ 15 / 17 }

CSS is deep but I am creating content on twitter in order to reduce the complexity for beginners

Check out this curated list of CSS threads written by me



Pratham
@Prathkum



You can learn 90% of CSS using these 10 threads



9:50 AM · Apr 19, 2021



9K 305 Copy link to Tweet

{ 16 / 17 }

Amazing! I think this is pretty much it for this thread. Don't think much just start with it, you'll enjoy the process for sure.

Thanks for reading this and share it with your connection it means a lot to me ❤️

Peace out 😊



Pratham @Prathkum

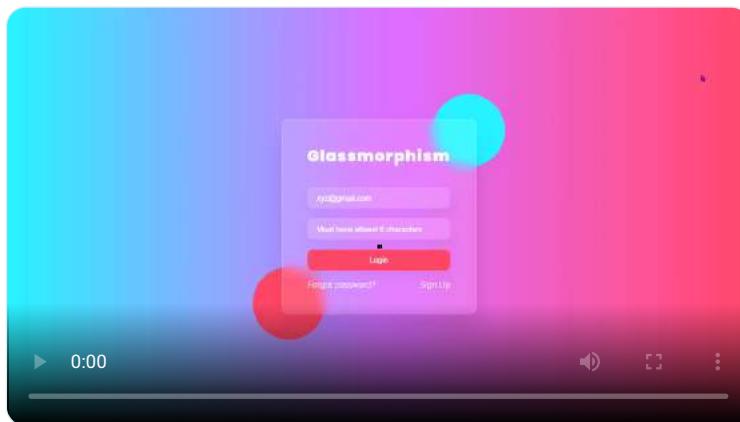
18 May · 11 tweets · [Prathkum/status/1394686217653002244](#)

Tr

Glassmorphism is a unified name for the popular frosted Glass design. It is trendy web design nowadays which looks super cool

Let's create a glassmorphic form using few simple CSS steps

A thread 



Few days back I have covered neumorphic design in one of my thread and now in this thread we will be creating glassmorphic form.

Glassmorphism design is all about one CSS property which is backdrop-filter. Let's start 

Step 1: Starting of the basic structure

Just start with writing all necessary tags for form layout. You can any number of input field according to you.

```
<div class="form">
  <div class="glassmorphic"></div>
  <form>
    <input type="text" placeholder="xyz@gmail.com" />
    <input type="password" placeholder="Must have atleast 6 characters" />
    <button type="submit" value="Login/Submit"></button>
    <div class="forget-signup">
      <a href="#">Forgot password?</a>
      <a href="#">Sign Up</a>
    </div>
  </form>
</div>
```

...

In order to create a glass like container, you just need to remember 3 things

1. White background color with almost zero opacity
2. Black shadow with low opacity and high blur radius
3. Add blur effect behind the glass (backdrop-filter property)

Now we know this, let's move

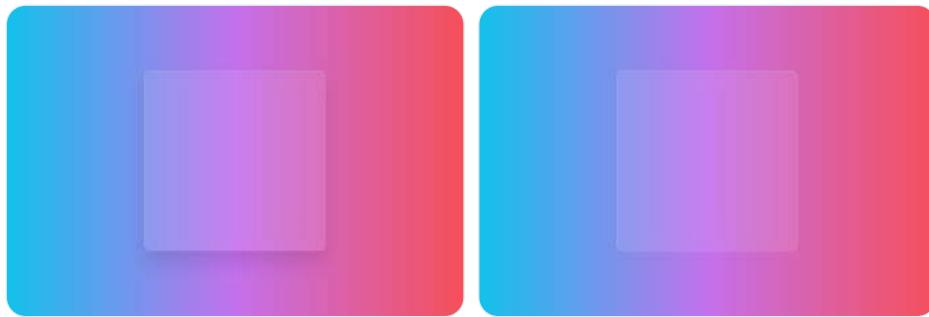
Step 2: Create glass

- Add thin white border
- Low opacity white background color
- Low opacity and large blur black shadow



Don't forget to add the shadow. Don't think that it won't make much difference.
Shadow is important as it add little 3d effect to the glass

Compare these two images ↪



Step 3: Style input field and button

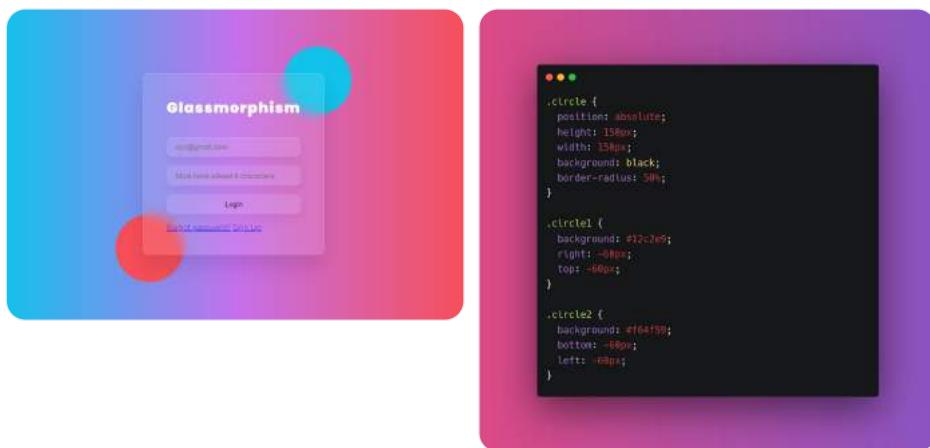
This is step same as step2. We need to add glass look in the input field as well.



Awesome! Our glassmorphic form is now taking shape.

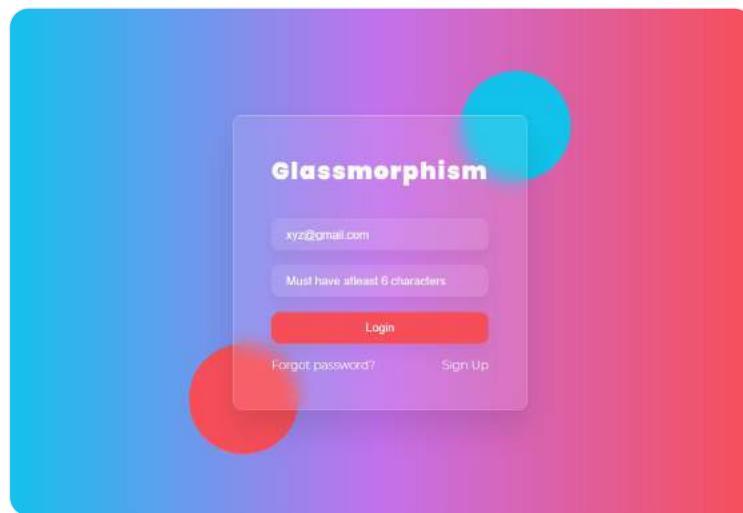
Step 4: Create shapes below glass.

Create any shapes or figure below the glass so that it look like the shape is placed under the glass. It will give our glass an actual and more appealing look



Step 5: Add some final detailing

- One consistent text color
- align "forgot password" and "signup"



You can check out the full source code [here](#)

🔗 <https://codepen.io/prathkum/pen/QWpKJPz>

Awesome! Wasn't it simple and beautiful? You should give it a go

Drop the screenshots of your creations and tag me in the post. I'll be more than happy to check it out ❤️

Peace out 😊



Pratham @Prathkum

3 May · 27 tweets · [Prathkum/status/1389278842611773440](https://twitter.com/Prathkum/status/1389278842611773440)

Tr

A complete beginner's guide to styling your website with CSS.

Thread  



CSS is an amazing and unique language that servers a great purpose. We can make our website visually good using CSS. It describe the presentation of web pages, including typography., layouts, color etc...

{ 2 / 27 }

First and foremost

The characterstic of a great website is it's color scheme. Forget about everything and learn about background and color properties initially.

The colors are something from which users interact first whenever they visit your webpage

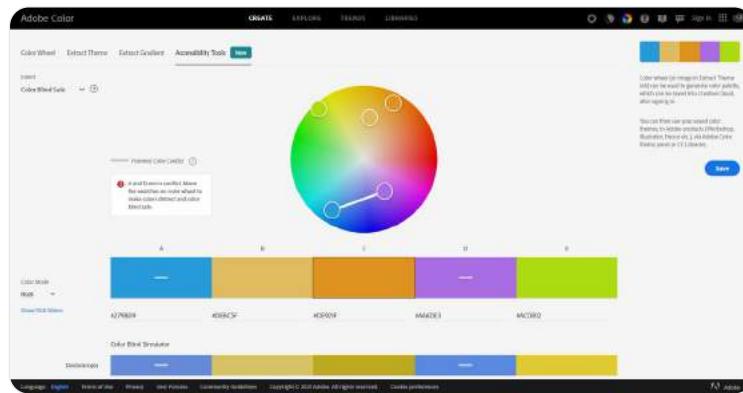
{ 3 / 27 }

There are a lot of great color palette out there using which you can generate pleasant color schemes

Check this great tool for generating accessible colors

<https://color.adobe.com/create/color-accessibility>

{ 4 / 27 }



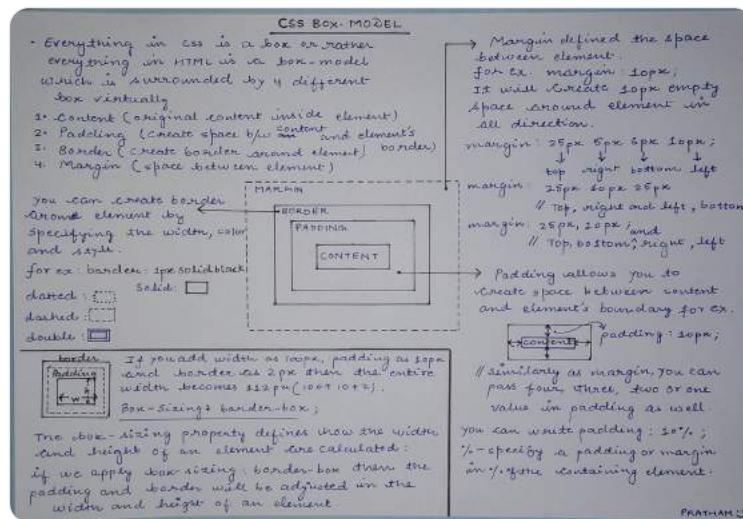
Don't think that background property is just for setting the solid color. Background is a shorthand property for background-image, background-position etc..

{ 5 / 27 }

Box model is one of the most important concept of CSS. It's not so tough to learn. The box-model covers

- Height
- Width
- Padding
- Border
- Margin

{ 6 / 27 }



Height and width property are pretty intuitive. These are used to set fixed height and width to the element

I suggest to give a look at max, min-width and max, min-height properties as well (we will cover these further in this thread)

{ 7 / 27 }

Proper and uniform separation of elements is something that can give your webpage a appealing look. Margin and padding can do this for you.

Give this article a short read for Definitive guide of padding and margin

PADDING — VS — MARGIN

Padding vs Margin: The Definitive Guide – UX Engineer

Padding and margin have two different purposes. Padding is for spacing within elements. Margin is for spacing between elements. However, there's more to it!

<https://uxengineer.com/padding-vs-margin/>

{ 8 / 27 }

Border are used to set the color, width and style to elements. You can learn it in 5 min
😊

Some good border selection can give your element a good pleasant look

{ 9 / 27 }

Moving forward, typography is an essential thing of web page. A good font can make your webpage and establish a strong visual hierarchy, provide a graphic balance to the website, and set the product's overall tone.

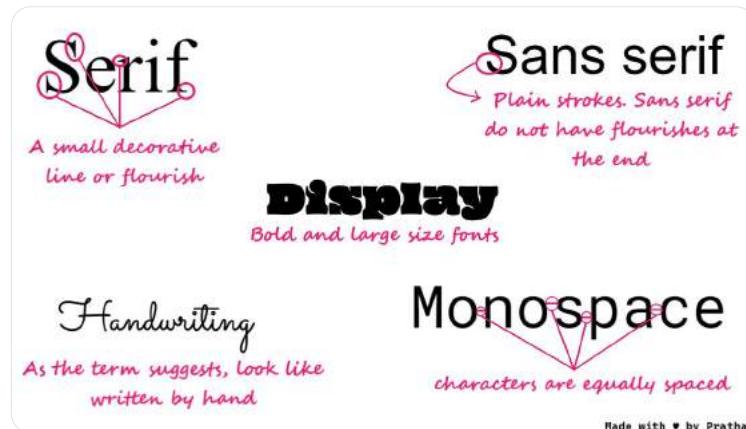
You can add free fonts from Google's official site

{ 10 / 27 }

There are five basic classifications of fonts:

1. serif
2. sans serif
3. script
4. monospaced
5. display

Give this article a read for more detailed explanation



Google Fonts
Making the web more beautiful, fast, and open through great typography
<http://fonts.google.com>

(Download free fonts from here)

You just need to look at few fonts properties. For ex,

- ◆ font-family
- ◆ font-weight
- ◆ font-size

Alright moving further, We have CSS positioning.

From here, a bit tricky CSS starts. Using CSS positioning you can change the position of your element. This might seem a bit tough but you can learn it in 2-3 days

I have already written a detailed thread on CSS positioning, If you're interested check it out



Pratham
@Prathkum

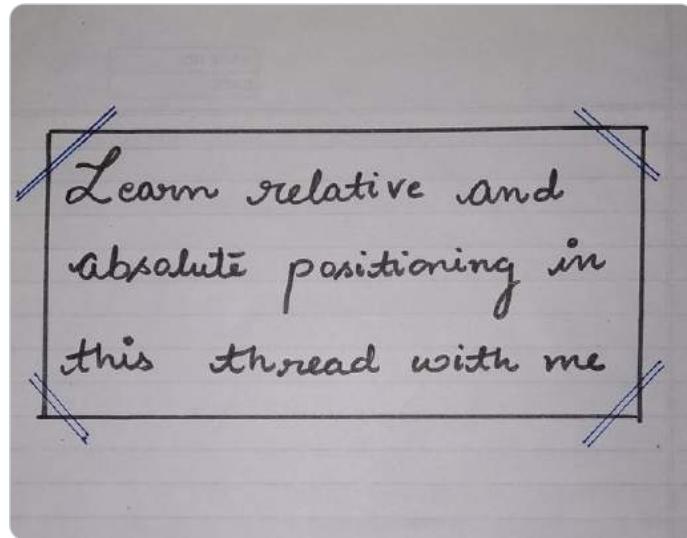


Positioning in CSS allows you to display your element wherever you want on the screen

But when I was learning it, I found it little bit confusing 😊

So in this thread I'll try to explain it in easiest manner with practical implementation. Let's start

THREAD  

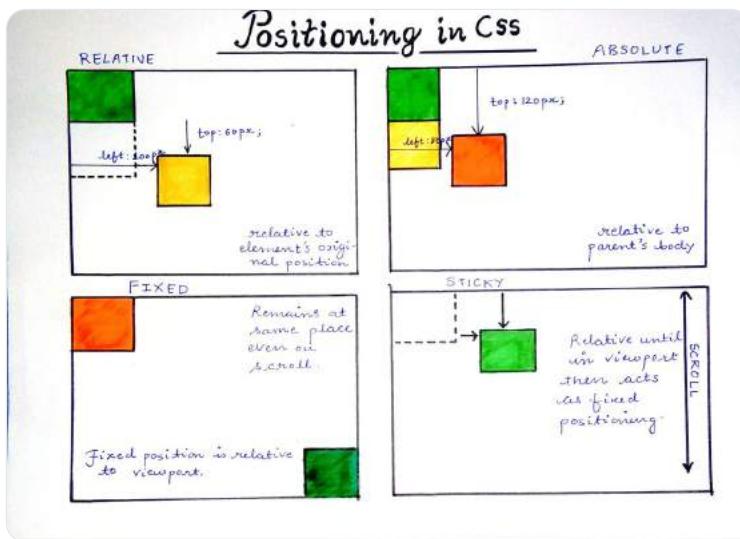


10:48 AM · Jan 31, 2021

(1)

948 38 Copy link to Tweet

{ 14 / 27 }



Up to this point you have some decent knowledge of styling your website. It would be great if you learn about a layout system now.

Flex and Grid

Grid is little bit tough to master but flex isn't

{ 15 / 27 }

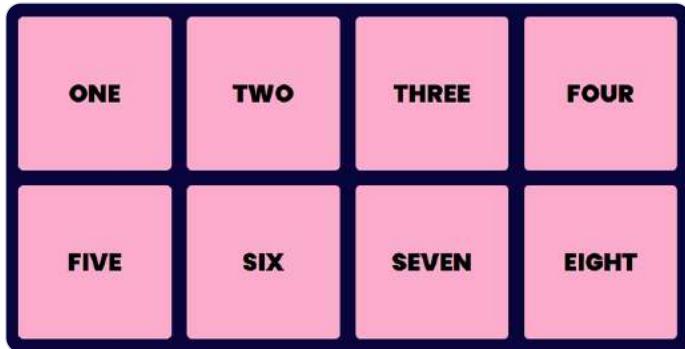
I have covered both the layout systems in my other threads. Check them out

Flex:

 **Pratham**
@Prathkum 

Everything you need to know about CSS Flex layout

A Thread 



7:38 PM · Apr 7, 2021 

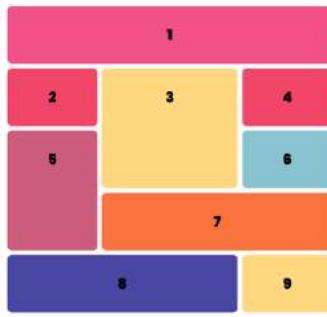
 1.3K  42  Copy link to Tweet

Grid:

 **Pratham**
@Prathkum 

A complete beginner's guide to CSS Grid layout 

Thread 



9:20 AM · Mar 24, 2021 

 3.7K  93  Copy link to Tweet

{ 16 / 27 }

We have covered almost everything upto this point except one thing.

📌 Responsive Web Design

This is the most trickiest part let's cover it in detail 👇

{ 17 / 27 }

In order to make a RWD, you just need to consider one thing.

"Ability of content to fit inside any device that look good and it will be for user to interact with that"

{ 18 / 27 }

Few quick points 👇

- 1 Use meta viewport element
- 2 Don't use large fixed width
- 3 Use <picture> tag
- 4 Responsive text size
- 5 Try to use layouts
- 6 Media Queries are saviour
- 7 Use "auto" in media

{ 19 / 27 }

1 First and foremost thing in order to make RWD is <meta> viewport element.

It forces page to follow the screen-width of the device.

{ 20 / 27 }



2 The second important point to note is that don't use large fixed width or height element.

It can cause trouble. Let's say an element having width 500px but user is viewing on a device having width smaller than 500px

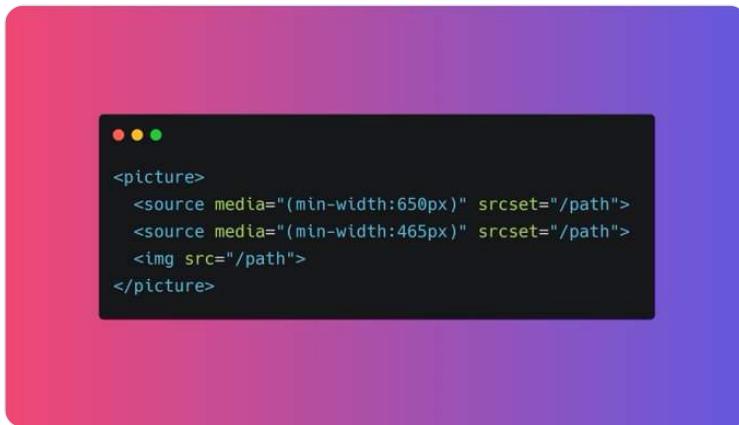
In such cases, use min-width or max-width

{ 21 / 27 }

3 Use HTML <picture> tag

The HTML <picture> element allows you to define different images for different browser window sizes.

{ 22 / 27 }



4 Responsive text size

Although you can make text responsive using media queries but you can also use "viewpoet" width as well.

```
h1 {
  font-size: 10vw;
}
```

{ 23 / 27 }

5 Try to use Layouts

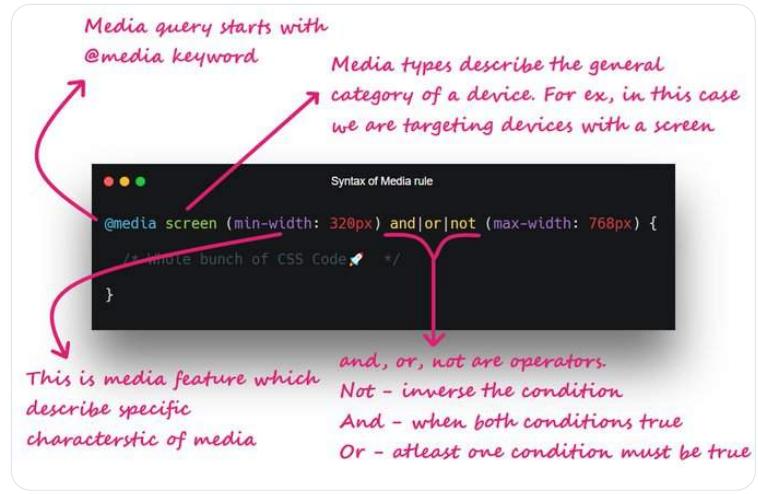
Using Grid or Flex layout always beneficial in order to make a web page responsive. Both these layout are not hard to learn. Try to use them

{ 24 / 27 }

6 Media Queries are saviour

Media query is a rule to include a block of CSS properties only if a certain condition is true. It is very useful in order to make a RWD.

{ 25 / 27 }



7 Use "auto" in media

Almost 99% a web page contains images or videos. And in order to make them responsive, use "auto"

If the width property is set to a percentage and the height is set to "auto", the image will be responsive

{ 26 / 27 }

Great!! I think that's pretty much it in order to give you a quick overview. If you have any doubts, feel free to drop a comment below

If you like this thread, a retweet means a lot ❤

{ 27 / 27 }

• • •



Pratham @Prathkum

8 May • 14 tweets • [Prathkum/status/1391049451825876992](https://twitter.com/Prathkum/status/1391049451825876992)

Tr

Complete introduction of CSS media queries for beginners

Thread 🧶



The biggest misconception among beginners are they think that media queries are for making a website responsive.

Though that's the one use case of media queries. You can use them modifying your site according to device media type. Let's see this in bit more details

Media types describe the general category of a device.

They are useful when you want to modify your site or app depending on a device's general type (such as print vs. screen) or specific characteristics and parameters (such as screen resolution or browser viewport width).

Let's see the media types first 👈

1. all
2. Print
3. Screen
4. Speech

There are some deprecated media types as well but they are not in use nowadays like tty, tv, projection, embossed, aural etc...

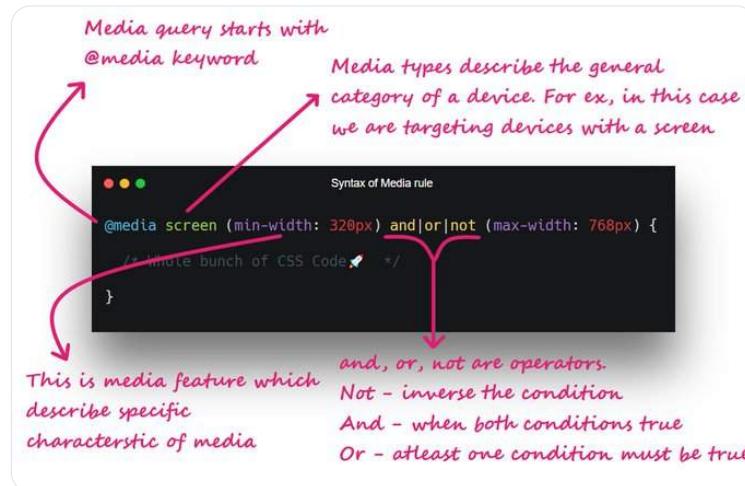
Alright moving forward, next thing we need to understand is media features.

As the term suggests, they are features of a particular media. They specifies the characteristic of the output device.

For example, width, height, color, aspect-ratio etc...

I hope this helped you understand that media queries are not just to make websites responsive. Although this is the biggest use case of media queries. When I was first learning it, I found some difficulties. Let see how we can use it in order to create a responsive website 

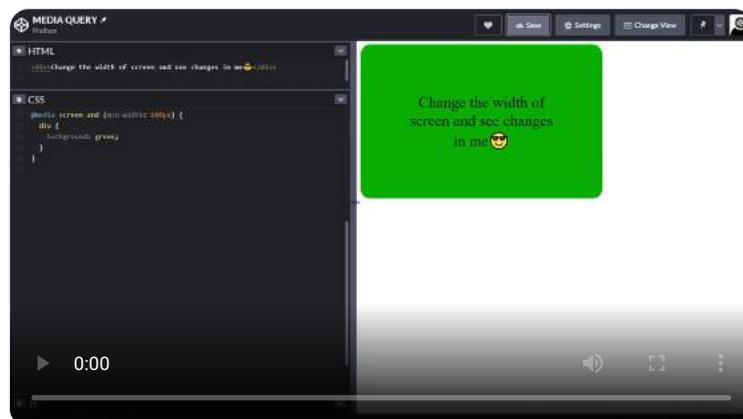
Let's start with the syntax first (See attached image)



Let's see an example into action. So that we can build strong hold on the concept

```
#####
@ media screen and (min-width: 600px) {
div {
background: green;
}
#####
#####
```

In this case the background will be green whenever the screen width is 600px or greater than 600px;



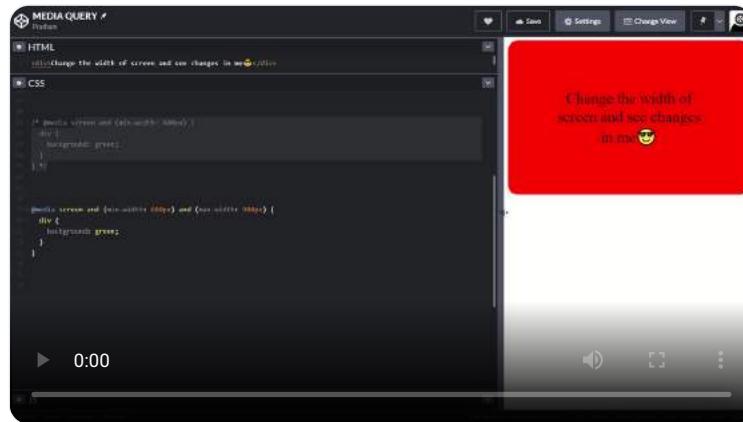
You can also combine two media features using "And", "Not" or ","

For example:

```
"""
@ media screen and (min-width: 600px) and (max-width: 900px) {
div {
background: green;
}
}
"""

```

In this case, the background will be green between 600px to 900px width



You can try bunch of media features

- any-hover
- color
- height
- width
- grid
- aspect-ratio
- orientation
- resolution, etc...

We can also write the nested media queries:

When min-width = 600px and user hover over element then solid black will create around div.



• • •

You can check my other thread if you want to learn more about how to make website responsive

 **Pratham**
@Prathkum

Complete guide to Responsive Web Design





10:45 AM · Apr 10, 2021

 2.2K  52  Copy link to Tweet

I guess that's pretty much it for quick overview. I hope you like it. Share this thread with your connection if you like it ❤️

Peace out 😊



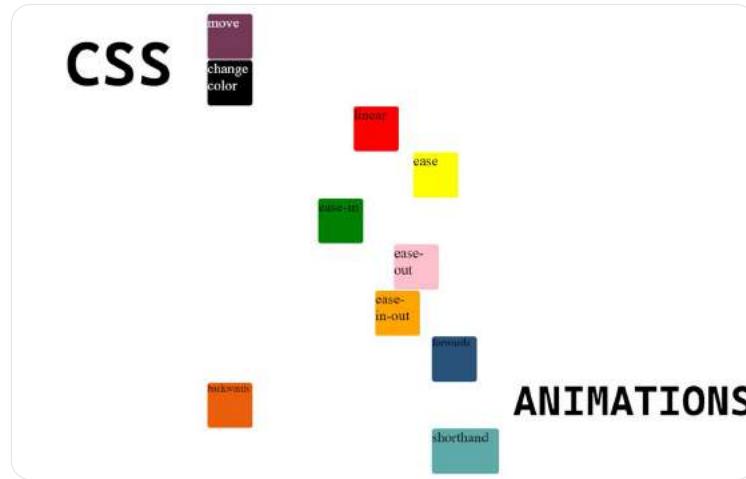
Pratham 🎨🚀 @Prathkum

1 Mar · 19 tweets · [Prathkum/status/1366278875467153413](#)

Tr

A quick start guide to CSS animations ⌂

Thread 🧶



Animations on your web pages or website can catch more audience

You can create some amazing animation using CSS itself. In this thread, we will try to learn some decent knowledge about it

Let's start ⌂

{ 2 / 19 }

Animation is all about changing one style to another at certain intervals or times

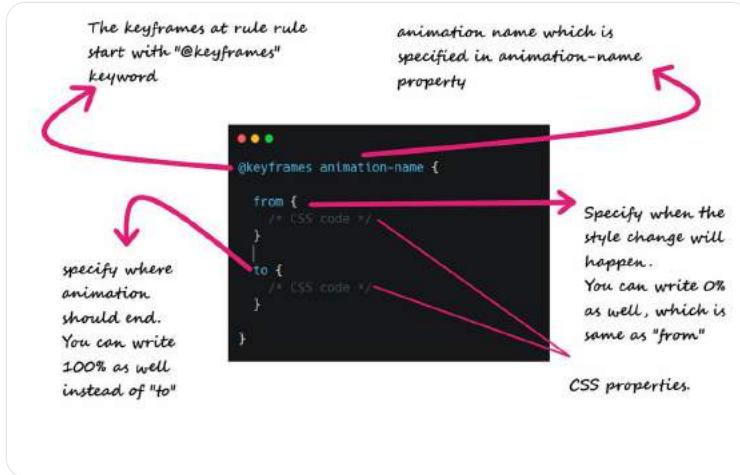
For doing that, first and foremost thing which we need to learn is @ keyframes at rule

{ 3 / 19 }

Let's understand the keyframes syntax first (See attached image)

Confusing? Don't worry! Let's move further everything will be crystal clear

{ 4 / 19 }



animation property is used to bind keyframes with a particular element

For example, suppose I want to move my element 500px left in four seconds. It's very easy (check video in next tweet)

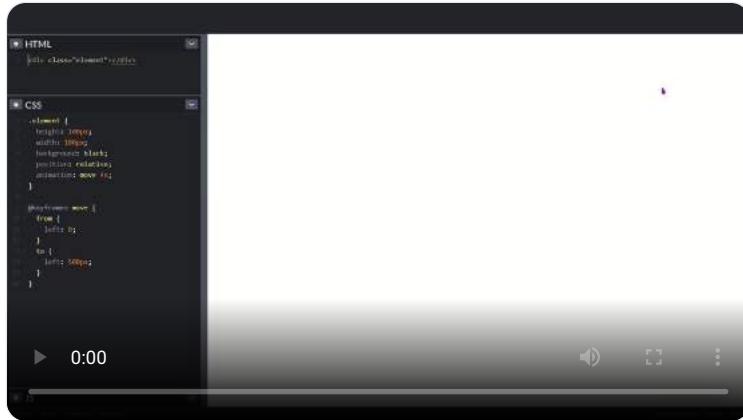
{ 5 / 19 }



The output of the above code

Play with code here <https://codepen.io/prathamkumar/pen/OJbvOoZ>

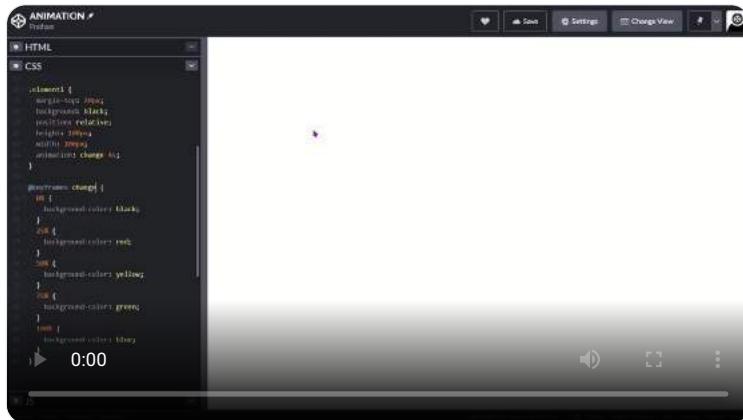
{ 6 / 19 }



Alright moving further, we can also pass percentage in keyframes rule and make some amazing animations

For example, suppose I want to change the background-color of an element

{ 7 / 19 }



One thing you might notice here is that I'm using animation property only and passing multiple values in that. Like

```
animation: move 4s;
```

This is because animation is a shorthand for setting all animation properties

{ 8 / 19 }

You can also specify the delay in your animation using animation delay property.

It specifies a delay before the start of an animation

{ 9 / 19 }

So far we have noticed that our animation only happens once.

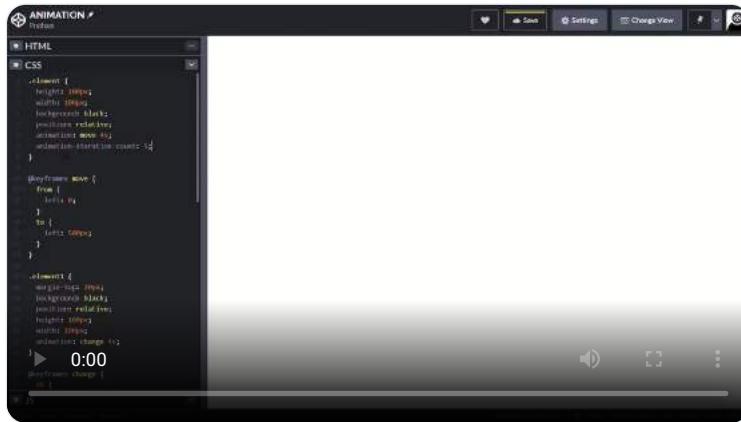
This is because we haven't applied the animation-iteration-count property

It specifies the number of times an animation should run.

For ex, animation-iteration-count: 5

(You can also pass infinite)

{10 / 19}



Creating a smooth animation is important and we have

animation-timing-function property for that.

This property specifies the speed curve of an animation. There are many functions that you can pass in this

{ 11 / 19 }

You can pass following values in animation-timing function

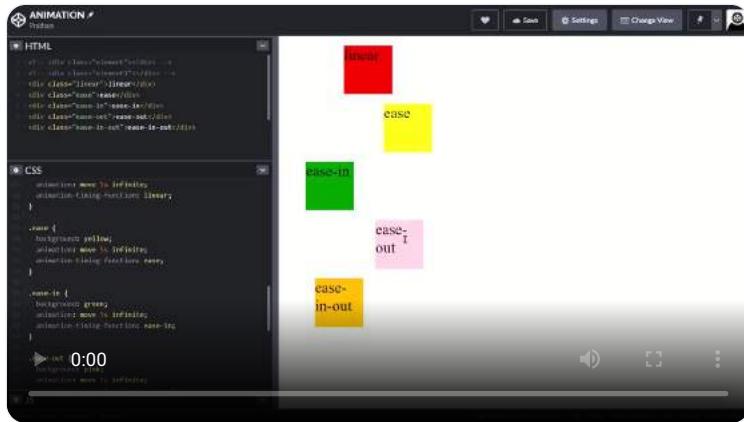
- ease (slow start, then fast, then slow)
- linear (same speed)
- ease-in (slow start)
- ease-out (slow end)
- ease-in-out (slow start and slow end)
- cubic-bezier (customizable)

{ 12 / 19 }

Check out this code and output for better understanding

<https://codepen.io/prathamkumar/pen/OJbvOoZ>

{ 13 / 19 }



One thing to note here is that animation do not affect element before or after the keyframes.

In order to persist the styling based on last or first keyframe, we have animation-fill-mode

{ 14 / 19 }

It accepts following value

- forwards (element will retain last keyframe styling)
- backwards (element will get the first keyframe value even in animation-delay period)
- both (both of the above)

{ 15 / 19 }

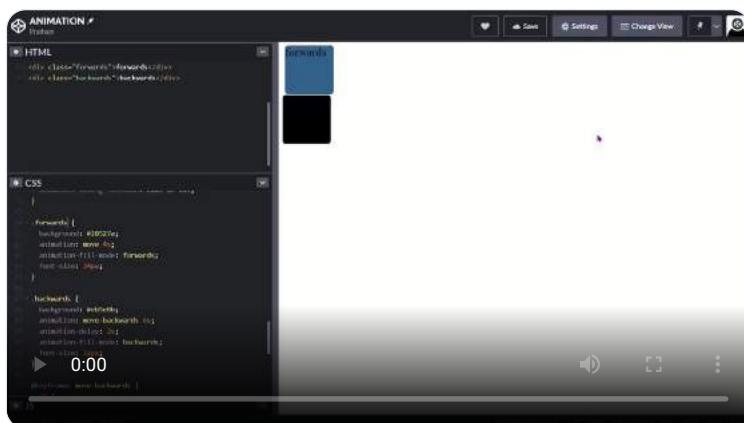
Let's play with animation-fill-mode

As you can see blue box stopped at left: 500px because forwards is begin applied on it
And orange box has black background even it's in a delay period of 2 seconds

Check code for better understanding

<https://codepen.io/prathamkumar/pen/OJbvOoZ>

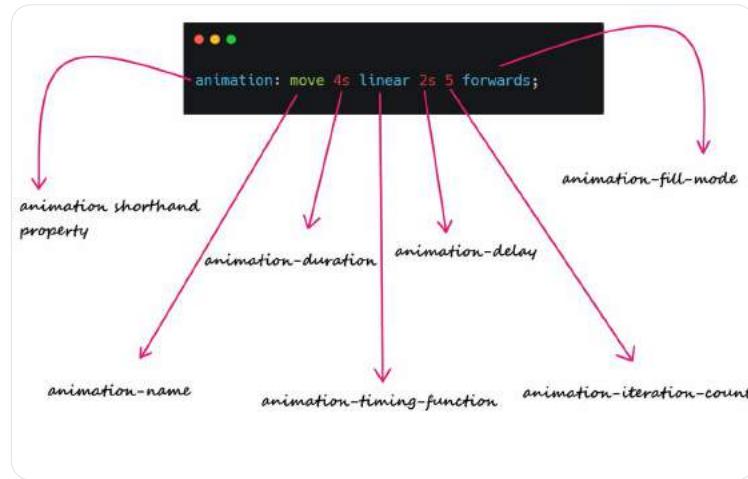
{ 16 / 19 }



Instead for using different animation properties you can use shorthand "animation" only

For example ↪

{ 17 / 19 }



This may seems little tough initially but practice of 2 - 3 days can make it easier.

Play around with code and try to write code by your own. This will definitely help you

<https://codepen.io/prathamkumar/pen/OJbvOoZ>

{ 18 / 19 }

This is pretty much it I guess. I hope you get some knowledge from this thread.

If you like it, share this thread with your connections, it means a lot to me ❤️

And thank for reading it 😊

*** END ***

• • •



Pratham @Prathkum

14 May · 13 tweets · [Prathkum/status/1393193386195161092](#)

Tr

Neumorphism is a trendy UI style nowadays. It is worth noting that neumorphism is not a replacement for flat design style, but an addition to app user interface design

Let's create a neumorphic form using few simple CSS steps

A thread 



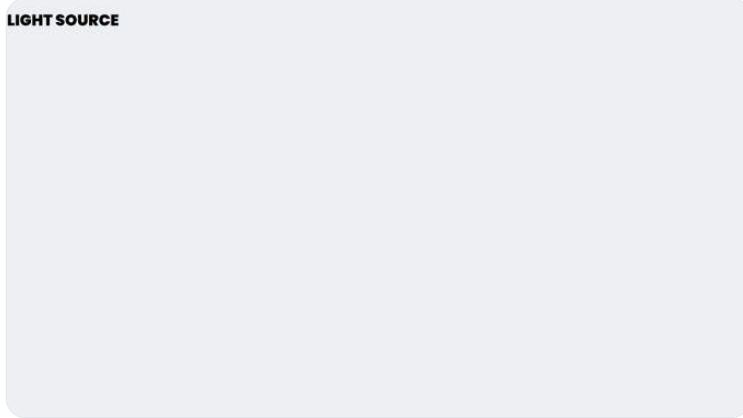
Neumorphic design is another UI design that has become popular these days.

Making neumorphic effect is actually pretty easy using only just CSS box-shadow

We'll be creating a neumorphic form in steps 

STEP 1:

Place a light source(virtually) on screen. In this example I'm considering that my light source is placed top left corner of the screen



STEP 2:

Create a container div and set background color slightly darker than white. I picked [#ecf0f3](#)

Make sure that the color of body and container should be same

As we placed our light source on top-left corner, make sure left and top border of container should be brighter than right and bottom border.

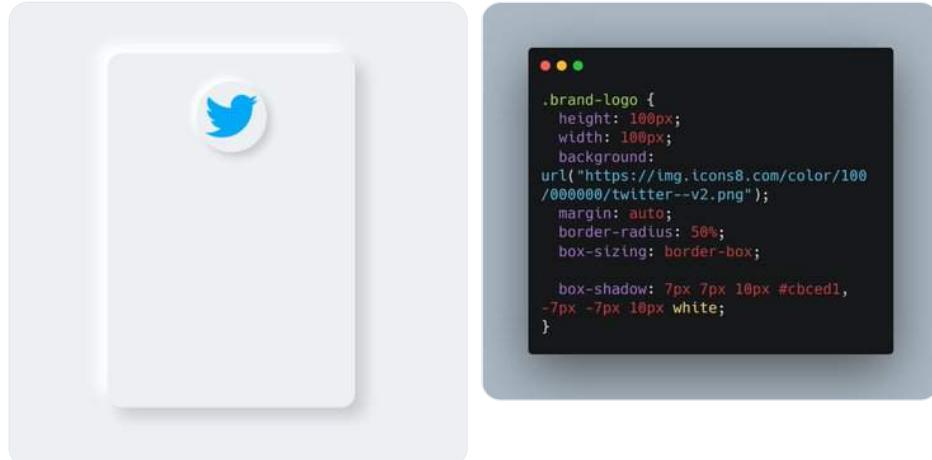
This will make the element looks like it has the light source illuminating from the top left corner of the screen.

```
.container {  
    width: 350px;  
    height: 500px;  
    padding: 40px;  
    border-radius: 20px;  
    box-sizing: border-box;  
    background: #ecf0f3;  
  
    THIS IS THE KEY PART ↴  
  
    box-shadow: 14px 14px 20px #cbced1,  
               -14px -14px 20px white;  
}
```

STEP 3:

Create a brand logo using same technique. Since brand going to protrude from the container, use same color i.e, [#ecf0f3](#)

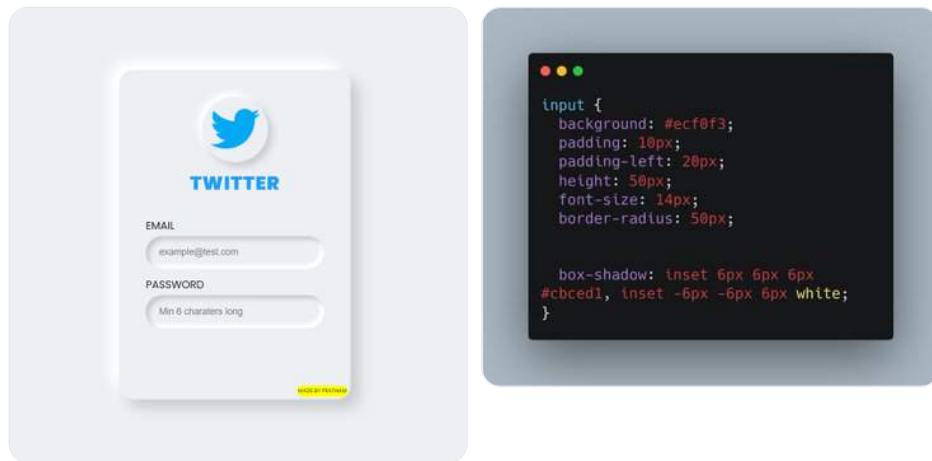
Add bright color shadow at the top left side and dark color shadow at the bottom right side



```
.brand-logo {  
    height: 100px;  
    width: 100px;  
    background:  
url("https://img.icons8.com/color/100/  
000000/twitter--v2.png");  
    margin: auto;  
    border-radius: 50%;  
    box-sizing: border-box;  
  
    box-shadow: 7px 7px 10px #cbced1,  
-7px -7px 10px white;  
}
```

STEP 4:

The input fields going to have a sunken effect. And to achieve this we will use inset box-shadow. As light is coming from top-left corner, hence right and bottom border will be bright in this case because input fields have sunken effect.



```
input {  
    background: #ecf0f3;  
    padding: 10px;  
    padding-left: 20px;  
    height: 50px;  
    font-size: 14px;  
    border-radius: 50px;  
  
    box-shadow: inset 6px 6px 6px #cbced1,   
    inset -6px -6px 6px white;  
}
```

STEP 5:

Create a button preferably of same color as logo because this will give you a more fascinating look to your design

Add hover selector and apply box-shadow none when you mouse over it



```
a {  
    color: inherit;  
    text-decoration: none;  
}  
  
button {  
    background-color: #00acee;  
    border: 1px solid #00acee;  
    border-radius: 50px;  
    color: inherit;  
    font-size: 14px;  
    padding: 10px;  
    width: 150px;  
}  
  
button:hover {  
    background-color: #00acee;  
    border: 1px solid #00acee;  
    border-radius: 50px;  
    color: inherit;  
    font-size: 14px;  
    padding: 10px;  
    width: 150px;  
}
```



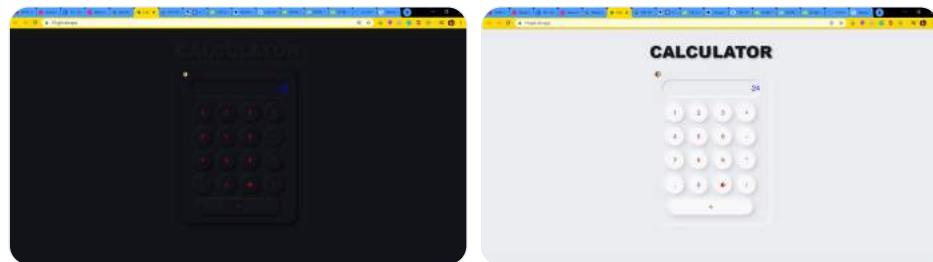
That's it!

Neumorphic design is all based on box-shadow property.
You just need to remember two things

- Protrude effect in outer element (normal box-shadow)
- Sunken effect in inner elements (Inset box-shadow)

You can create neumorphic design using any color scheme you just need to remember that shades should not differ much.

For example: check out this dark and light theme calculator



This is the end of this thread but not our creativity. Give a go to this awesome design and drop the link below ❤️

Peace out 😊

Check out the full source code:

<https://codepen.io/prathkum/pen/OJRvVzY>

Fleeting is an art

I often share more info in my fleet section as well. So be sure to be active there as well 😊

I just wrote a thread on

Neumorphic Design

Give it a read



Pratham @Prathkum 57m :

Neumorphism is a trendy UI style nowadays. It is worth noting that neumorphism is not a replacement for flat design style, but an addition to app user interface design

Let's create a neumorphic form using few simple CSS steps

A thread



6

45

165

...



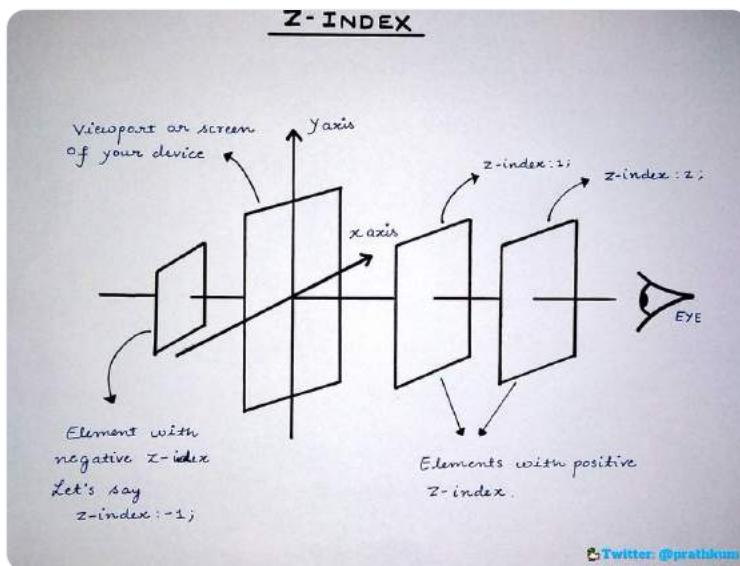
Pratham @Prathkum

1 May · 8 tweets · [Prathkum/status/1388489389702094849](#)

Tr

z-index is a powerful yet confusing concept of CSS. But this short thread will solve all your doubts related to it.

Thread



z-index is a CSS property that controls stacking order of elements along z axis.

Image a hypothetical line starting from your eye to screen, that is z-axis

Note that z-index only works on positioned elements.

You need to specify the position (relative, absolute, sticky, fixed) if you want to arrange an element using z-index

BAD

```
.element {  
  z-index: 1;  
}
```

z-index will not work as there is not position applied.

GOOD

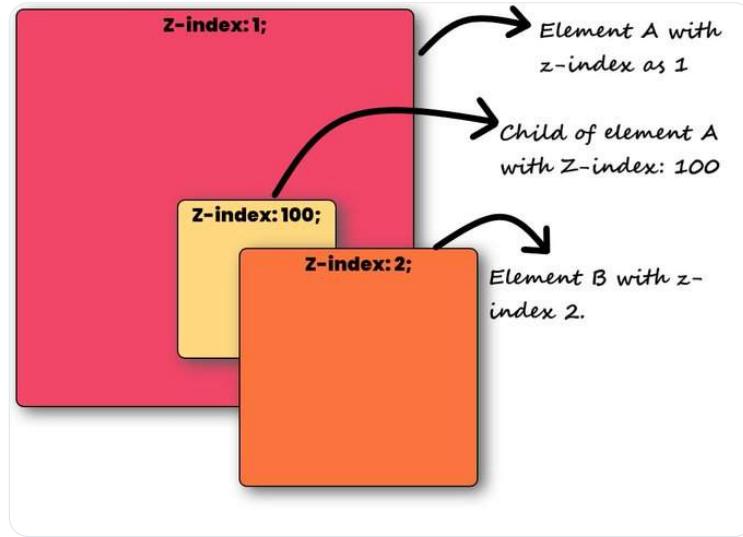
```
.element {  
  position: relative;  
  z-index: 1;  
}
```

z-index will work because position is applied

- z-index of nested elements

Let's say two elements A and B are siblings with element B written after element A in DOM, then the children of element A cannot be higher than element B no matter what z-index is being applied on children

See this in action 



- ◆ z-index of pseudo-elements

By default pseudo-elements(:before and ::after) display on the top of the parent.

You can set the stacking order of ::before and ::after as usual but you can't display pseudo-elements below the parent...



For example, this won't work 

```
● ● ● .parent { z-index: 0; } .parent::before { z-index: -1; }
```

In order to display the pseudo-elements below parent element, you just need to remove the z-index from parent itself. For example 



This is pretty much it. As simple as that 😊❤️

I hope you like it. Peace out 😊

• • •



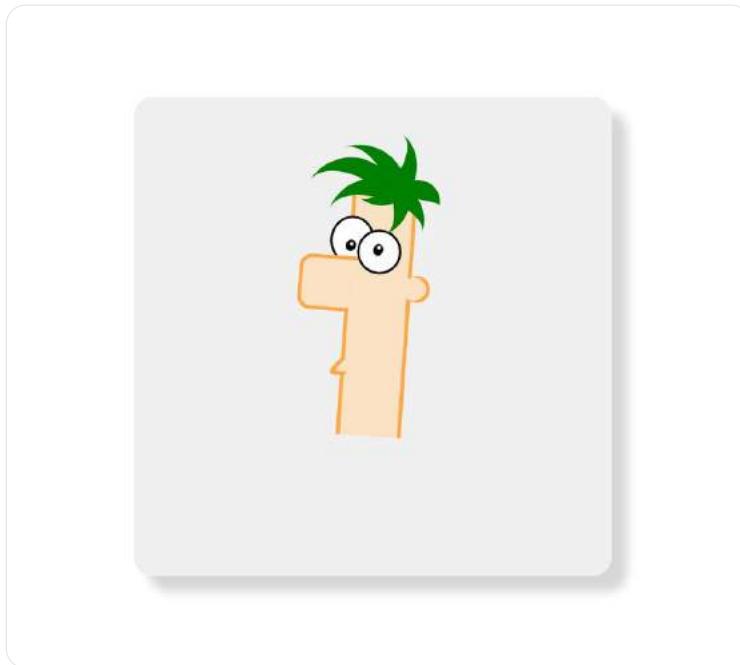
Pratham @Prathkum

23 May · 14 tweets · [Prathkum/status/1396304135520149506](https://twitter.com/Prathkum/status/1396304135520149506)

Tr

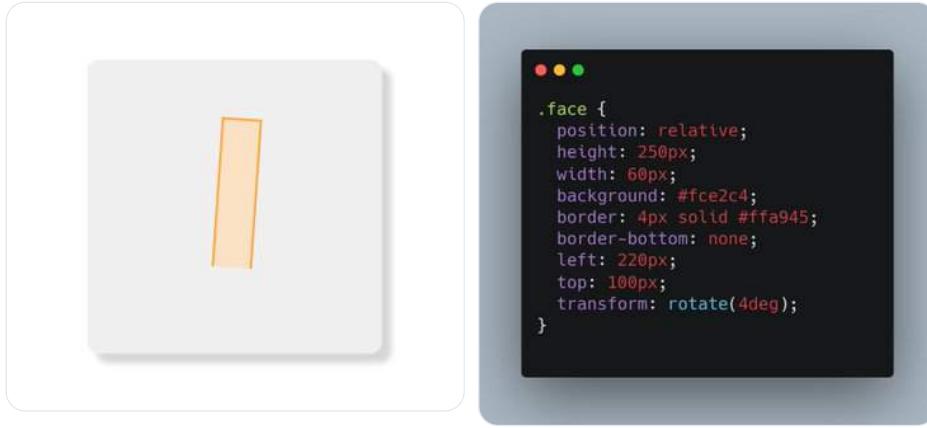
We can create cartoons using CSS as well 😊

Let's decode in this thread how we can create Ferb using few simple CSS steps. Let's go



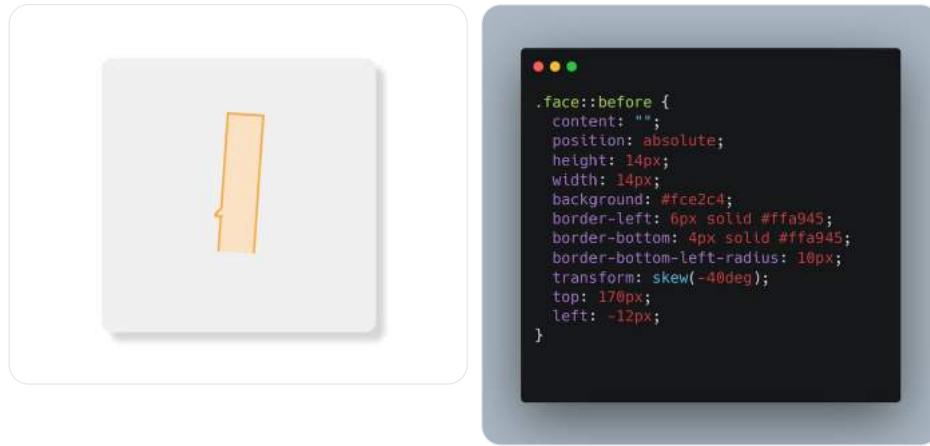
STEP 1: Create a Face of rectangular shape

- Create a rectangle and rotate it a little bit
- Give it a solid border without border-bottom
- Position it



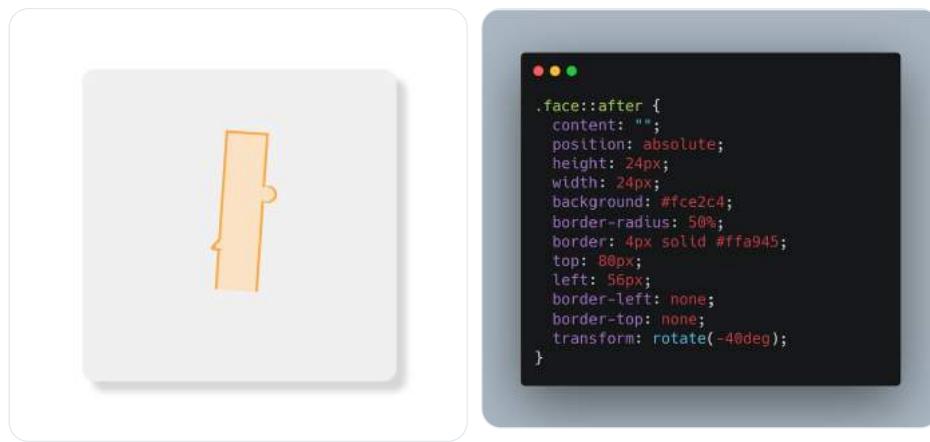
STEP 2: Create Lips

- Make a small square
- Add bottom and left border
- Skew it so that it seems like a lip
- Add little bit of bottom-left radius to give it a more realistic look



STEP 3: Create a Ear

- Create a circle
- Give it a solid right and bottom border of color same as face
- Rotate it a bit in order to match the boundary of ear to face



...

STEP 4: Create a Nose

- Make a square
- Apply solid border of same color as of face
- border-right: none - in order to match it with the face boundary
- Position and rotate it so that it sync with the face

The diagram shows a light gray rounded rectangle representing a face. Inside it, there is a smaller orange square rotated 4 degrees counter-clockwise, representing a nose. To the right of the face is a dark gray box containing the CSS code for the nose.

```
.nose {  
    position: relative;  
    width: 50px;  
    height: 50px;  
    background: #fce2c4;  
    border: 4px solid #ffa945;  
    left: 172px;  
    bottom: 90px;  
    border-right: none;  
    border-radius: 14px 0 0 14px;  
    transform: rotate(4deg);  
    z-index: 1;  
}
```


To the right of the nose icon is another dark gray box containing the HTML structure for the nose component.

```
<div class="container">  
  <div class="face"></div>  
  <div class="nose"></div>  
</div>
```

We need to extend the border of nose so that it looks like the nose is coming out of the face.

Again, we can do that using pseudo-elements

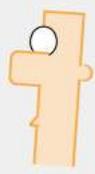
- Just create a small rectangle with same border and adjust it to the top border of nose

The diagram shows the same face and nose components as before, but now the nose has an additional vertical orange rectangle attached to its top edge, extending it upwards. To the right is a dark gray box with the CSS code for this pseudo-element.

```
.nose::after {  
    content: "";  
    position: absolute;  
    height: 10px;  
    width: 30px;  
    background: #fce2c4;  
    border-top: 4px solid #ffa945;  
    left: 30px;  
    top: -4px;  
}
```

STEP 5: Create an eye

- For eye we need an oval shape (height > width)
- Apply black border with white background color
- Position it behind the nose



```
<div class="container">
  <div class="face"></div>
  <div class="nose"></div>
  <div class="eye"></div>
</div>
```

```
.eye {
  position: relative;
  height: 44px;
  width: 40px;
  border-radius: 50%;
  background: white;
  border: 3px solid black;
  bottom: 188px;
  left: 206px;
  transform: rotate(4deg);
  z-index: 0;
}
```

- Add a small black oval inside the eye

* You can create black oval using pseudo-elements ::before



```
.eye::before {
  content: "";
  position: absolute;
  height: 12px;
  width: 10px;
  border-radius: 50%;
  background: black;
  left: 14px;
  top: 22px;
  transform: rotate(16deg);
}
```

- Add a even small white reflection in black oval(pupil) to give it a more realistic look

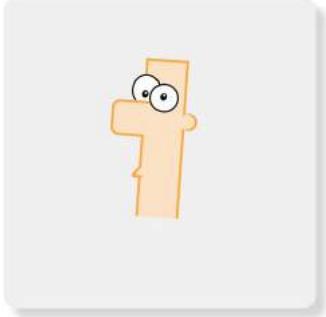
* You can create black oval using pseudo-elements ::after



```
.eye::after {
  content: "";
  position: absolute;
  height: 4px;
  width: 4px;
  border-radius: 50%;
  background: white;
  left: 18px;
  top: 24px;
}
```

STEP 6: Create second Eye

- Create second eye using same technique and code
- Just position it differently



```
<div class="container">
<div class="face"></div>
<div class="nose"></div>
<div class="eye"></div>
<div class="eye eye1"></div>
</div>
```

```
.eye1 {
width: 40px;
height: 40px;
bottom: 224px;
left: 234px;
z-index: 10;
}

.eye1::before {
top: 12px;
}

.eye1::after {
top: 14px;
}
```

STEP 7. Create hair spike

This is the easiest part in my opinions because all you need to make are just random spikes.

What I have done here is just created a border-top 20px and rotate it in order to give it a spike look.

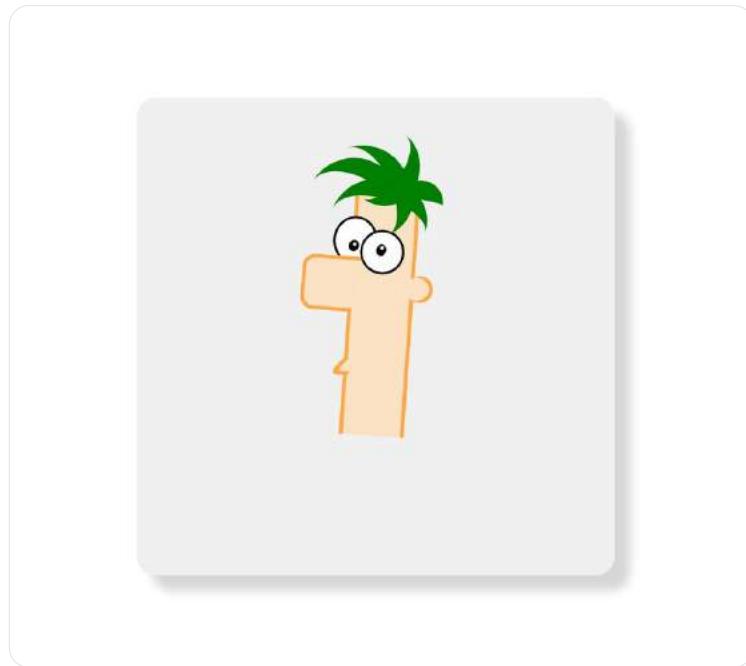


```
<div class="container">
<div class="face"></div>
<div class="nose"></div>
<div class="eye"></div>
<div class="eye eye1"></div>
<div class="hair">
<div></div>
<div></div>
<div></div>
<div></div>
<div></div>
</div>
</div>
```

```
.hair > div:nth-child(1) {  
    position: absolute;  
    background: transparent;  
    height: 30px;  
    width: 100px;  
    border-top: 20px solid green;  
    border-radius: 50px 40px 0 0;  
    bottom: 40px;  
    left: 180px;  
    transform: rotate(30deg);  
}
```

STEP 8. Create hairs

Repeat step 7 and create multiple spikes so that they look like hair



Done 😊

Wasn't it easy and fun. You should try it as well. It will help you improving your CSS skills

You can access the whole source code here:

<https://codepen.io/prathkum/pen/abmKoew>



Pratham 🎨🚀 @Prathkum

21 Mar • 16 tweets • [Prathkum/status/1373718310471221252](#)

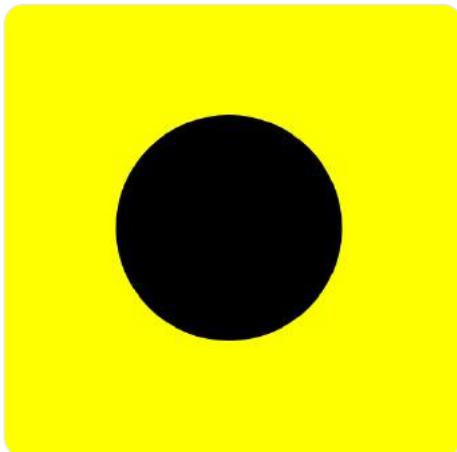
Tr

Let's create some easy and complex shapes using pure CSS 🎨

A Thread 🧶

1. Circle

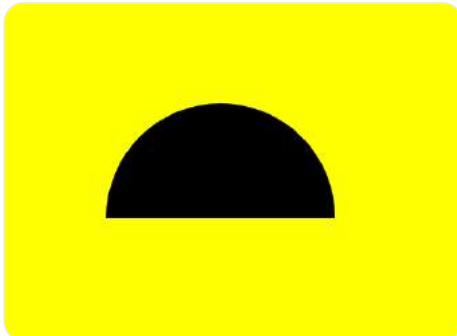
- Pretty simple, we just need to make a square and apply the border-radius 50% in order to give it a circular shape



```
...  
.circle {  
    height: 200px;  
    width: 200px;  
    border-radius: 50%;  
    background-color: black;  
}
```

2. Semi-circle

- Create a rectangle
- Apply border radius top left and top right same as height of the rectangle

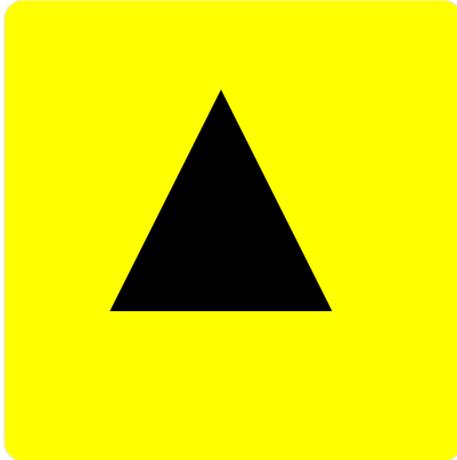


```
...  
.semi-circle {  
    height: 100px;  
    width: 200px;  
    border-radius: 100px 100px 0 0;  
    background-color: black;  
}
```

3. Triangle

- Creating a triangle is little bit tricky

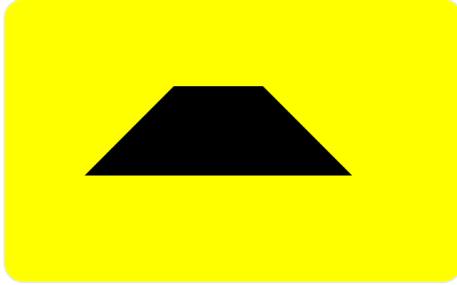
- Set height and width as zero
- To make this, we draw a solid border and make the side border transparent



```
.triangle {  
    width: 0;  
    height: 0;  
    border-bottom: 200px solid black;  
    border-left: 100px solid transparent;  
    border-right: 100px solid transparent;  
}
```

4. Trapezium

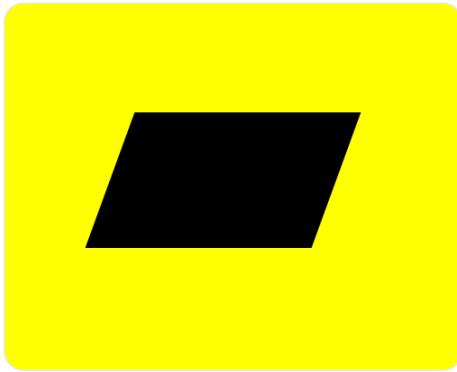
- Same as triangle but in this case we need to set some width



```
.trapezium {  
    width: 100px;  
    height: 0;  
    border-bottom: 100px solid black;  
    border-left: 100px solid transparent;  
    border-right: 100px solid transparent;  
}
```

5. Parallelogram

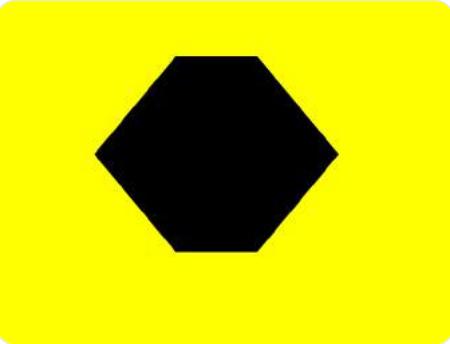
- Create a rectangle
- Apply skew in order to tilt it



```
.parallelogram {  
    height: 120px;  
    width: 200px;  
    background: black;  
    transform: skewx(-20deg);  
}
```

6. Hexagon

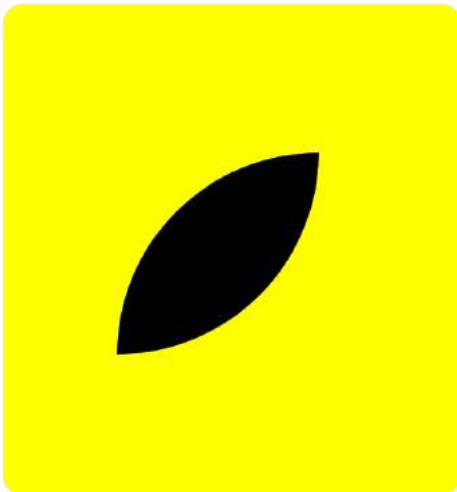
- Creating hexagon is very easy
- We need to make two trapeziums of same size but make sure that other trapezium should be upside down
- Align them perfectly



```
● ● ● .hexagon { position: relative; width: 100px; height: 0; border-top: 100px solid black; border-left: 100px solid transparent; border-right: 100px solid transparent; } .hexagon::before { content: ""; position: absolute; width: 100px; height: 0; border-bottom: 100px solid black; border-left: 100px solid transparent; border-right: 100px solid transparent; left: -100px; top: -240px; }
```

7. Leaf

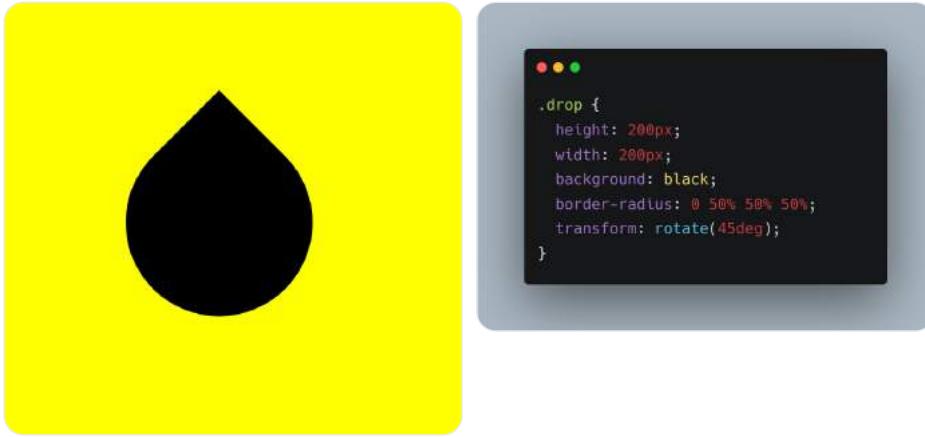
- Create a square
- Apply larger border-radius at top-left and bottom-right corner



```
● ● ● .leaf { height: 200px; width: 200px; background: black; border-radius: 250px 0 250px 0; }
```

8. Drop

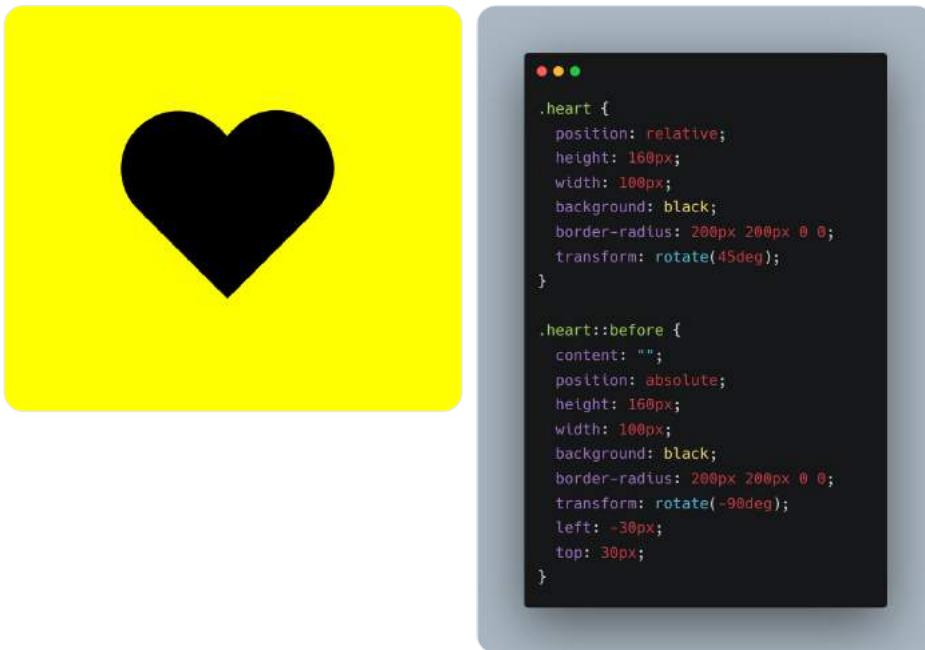
- Create a square
- Apply 50% border-radius to all the sides except one side
- Rotate in such a manner so that tip comes to top



```
.drop {  
    height: 200px;  
    width: 200px;  
    background: black;  
    border-radius: 0 50% 50% 50%;  
    transform: rotate(45deg);  
}
```

9. Heart

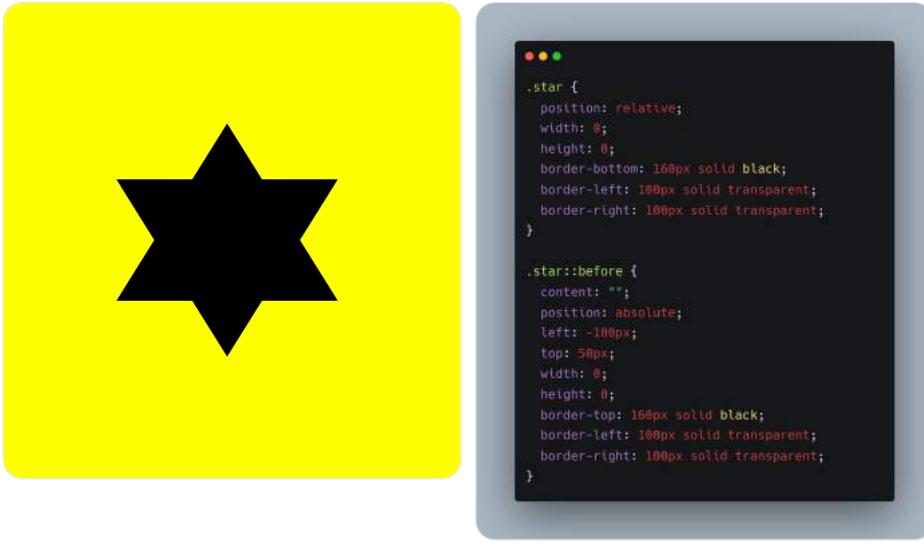
- Create a rectangle and apply circular border at top
- Rotate it 45deg
- Create another same rectangle with circular border and rotate in 45deg in other direction
- Align both of them



```
.heart {  
    position: relative;  
    height: 160px;  
    width: 100px;  
    background: black;  
    border-radius: 20px 20px 0 0;  
    transform: rotate(45deg);  
}  
  
.heart::before {  
    content: "";  
    position: absolute;  
    height: 160px;  
    width: 100px;  
    background: black;  
    border-radius: 20px 20px 0 0;  
    transform: rotate(-90deg);  
    left: -30px;  
    top: 30px;  
}
```

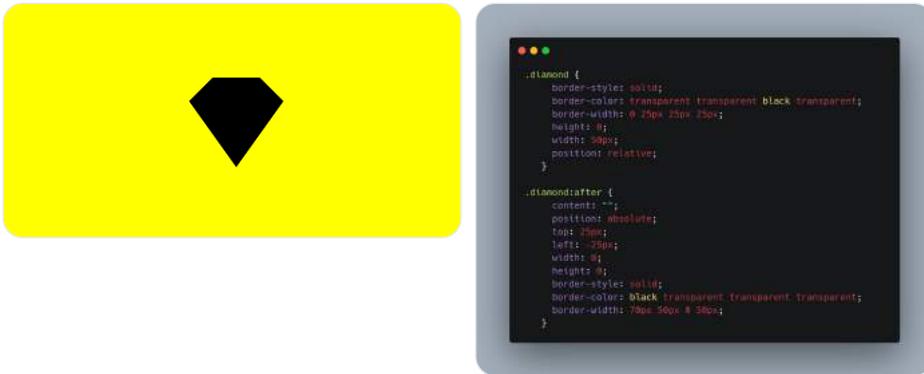
10. Star

- Create two triangle upside down
- Align second triangle in the middle of the first one



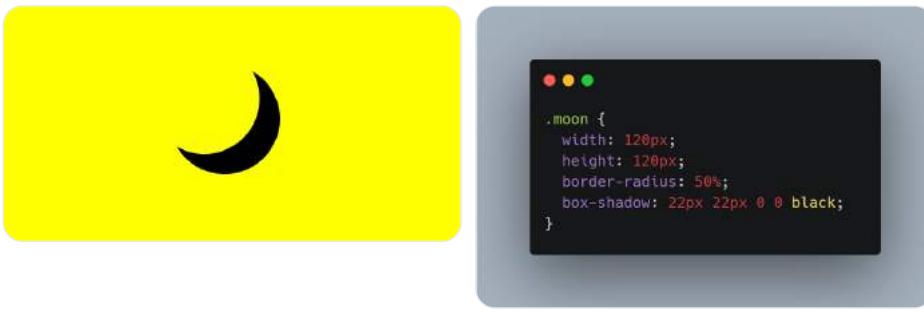
11. Diamond

- Combination of Trapezium and triangle



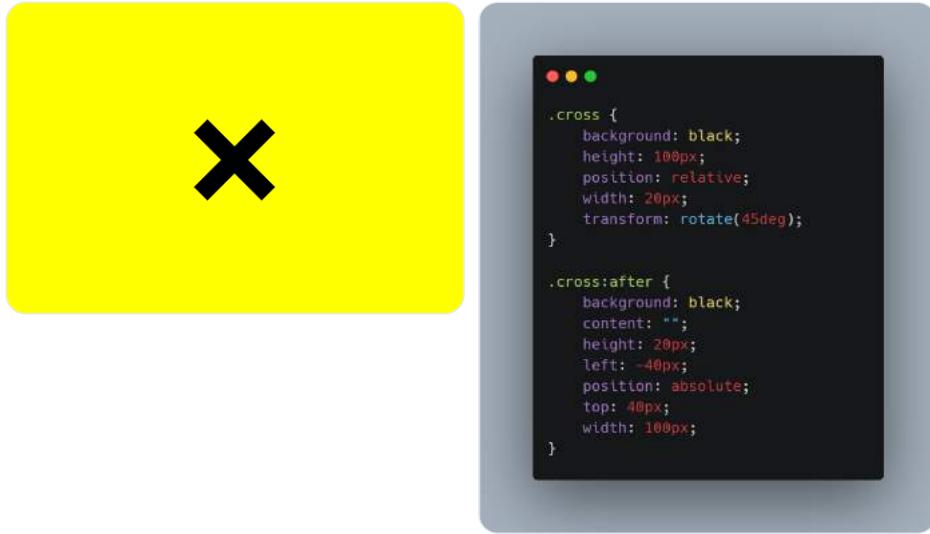
12. Moon

- Transparent background
- Apply box shadow



13. Cross

- Create two rectangles
- Place them over each other vertically and horizontally
- Rotate 45deg



14. Egg

- Using advance border-radius technique



I hope you will find it helpful, if so, share it ❤️

Peace out 😊

...



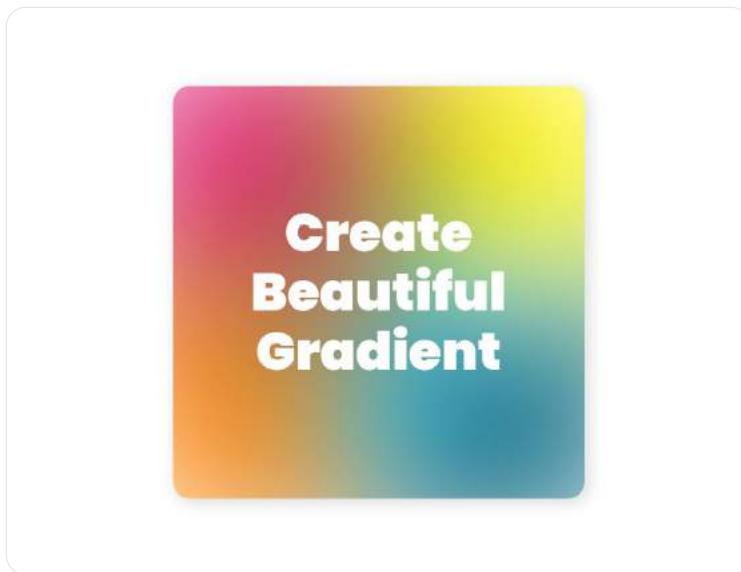
Pratham @Prathkum

6 May · 8 tweets · [Prathkum/status/1390291264218804225](#)

Tr

Aurora UI is a latest visual trend in 2021. Let's see how we can create that using CSS in 4 simple steps

Thread  



You can create beautiful aurora design hardly in 10 minutes. It is all about creating mesh gradients.

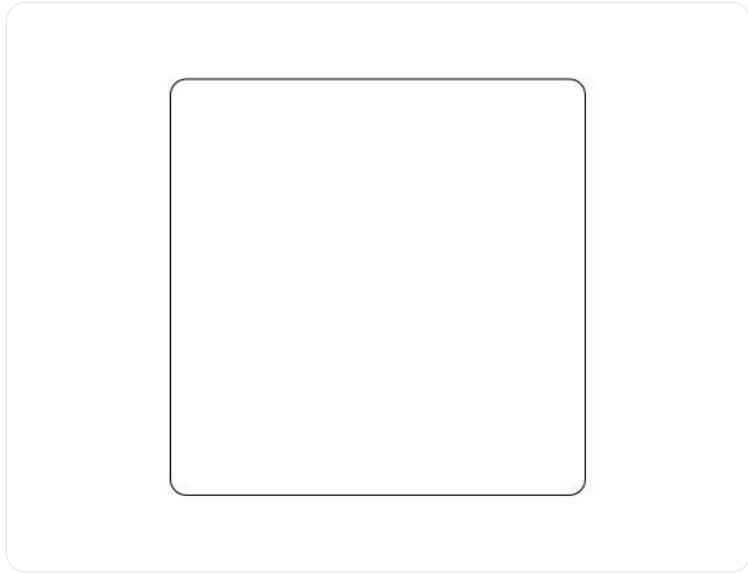
You can create gradient backgrounds using `linear-gradient()` or `radial-gradient()` but here we are creating cambered gradients

{ 2 / 8 }

Step 1: Create container

Just create a simple div with fixed height and width.

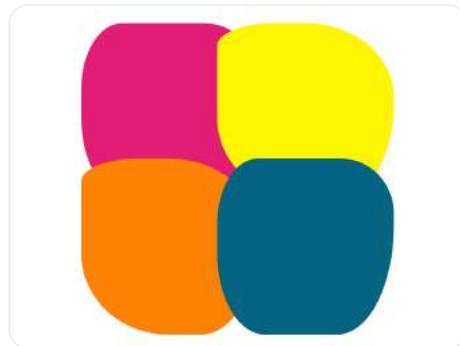
{ 3 / 8 }



Step 2: Create four distorted shapes

We need to create four random shapes inside container. You can create those shapes using height, width and border-radius property.

{ 4 / 8 }



HTML Structure

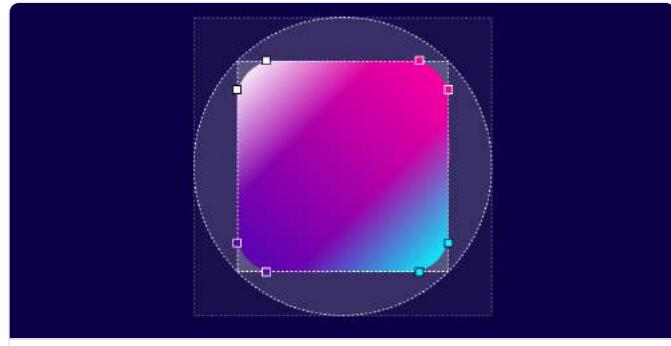
```
<div class="container">
<div class="c1"></div>
<div class="c2"></div>
<div class="c3"></div>
<div class="c4"></div>
</div>
```

CSS code for reference

```
.c1 {
  position: absolute;
  height: 350px;
  width: 350px;
  background: #e1d7ff;
  border-radius: 23% 38% 32% 27% / 37% 31% 62% 46%;
  left: -60px;
  top: -60px;
}
```

Check out this cool site for creating random shapes using border-radius





Border Radius Generator - Full Control

A visual generator to build organic looking shapes with the help of CSS3 border-radius property

<https://9elements.github.io/fancy-border-radius/full-control>

{ 5 / 8 }

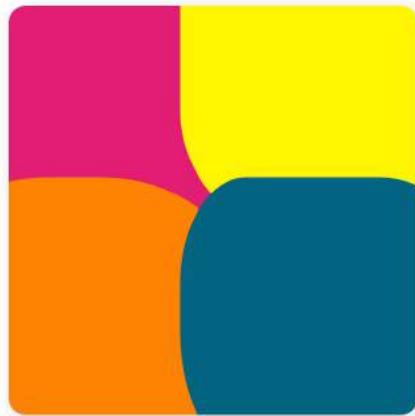


Alright moving further

Step 3:

Next thing you need to do is to add "overflow-hidden" in container element.

{ 6 / 8 }



Step 4: Blur

The final and last step is to add some blur and blending so that four shapes can merge into each other and show us the gradient look.

{ 7 / 8 }



```
.c1, .c2, .c3, .c4 {  
  filter: blur(10px);  
  mix-blend-mode: hue;  
}
```

As simple as that 😊

Check out the full source code here: <https://codepen.io/prathkum/pen/bGqGjoq>

• • •



Pratham 🎨🚀 @Prathkum
2 Jan · 11 tweets · [Prathkum/status/1345285253632569344](#)

Tr

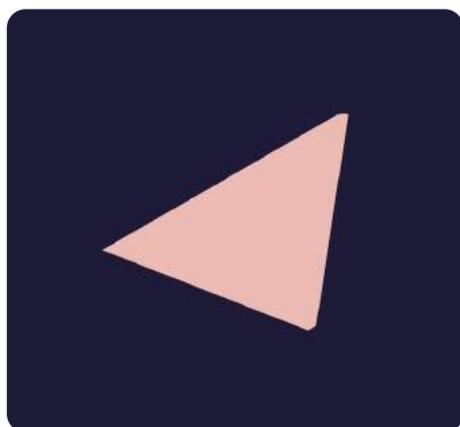
Let's create Phineas using CSS in few simple steps 😊

THREAD



STEP 1. Create a head

create a triangle and rotate it so that it looks like a head



```
● ● ●  
.head {  
  position: absolute;  
  height: 10px;  
  width: 10px;  
  border-bottom-left-radius: 20px;  
  border-bottom: 20px solid #ecbcb4;  
  border-left: 160px solid transparent;  
  border-right: 160px solid transparent;  
  transform: rotate(150deg);  
}
```

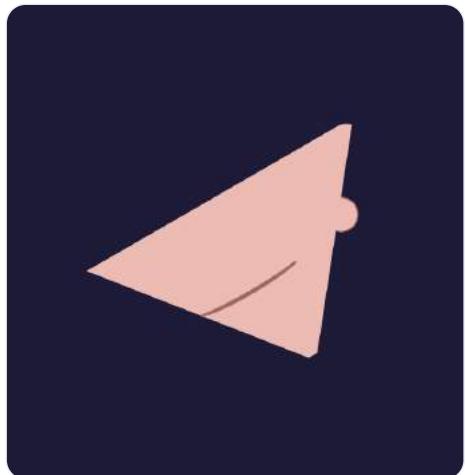
```
● ● ●  


</div>


```

STEP 2. Create smile and ear

We have a pseudo elements in CSS, so we can create smile and ear using head::after and head::before



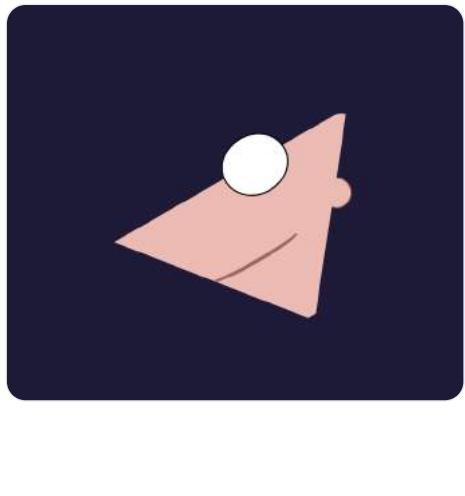
```
.head::before {  
    content: "";  
    position: absolute;  
    width: 120px;  
    height: 20px;  
    top: 100px;  
    left: -6px;  
    border: 2px solid #e66666;  
    border-color: #e66666 transparent transparent transparent;  
    border-radius: 50% / 10px 10px 0 0;  
}  
  
.head::after {  
    content: "";  
    position: absolute;  
    height: 20px;  
    width: 34px;  
    background-color: #e66666;  
    border-top: 2px solid #e66666;  
    border-bottom: none;  
    border-top-left-radius: 4px;  
    border-top-right-radius: 4px;  
    transform: rotate(-8deg);  
    top: 19px;  
    left: -12px;  
}
```

STEP 3. Create eyes

This is the most trickiest part because in order to make an eye we have to create

- One bigger white circle
- then a small black pupil inside white circle
- then a small white reflection in black pupil

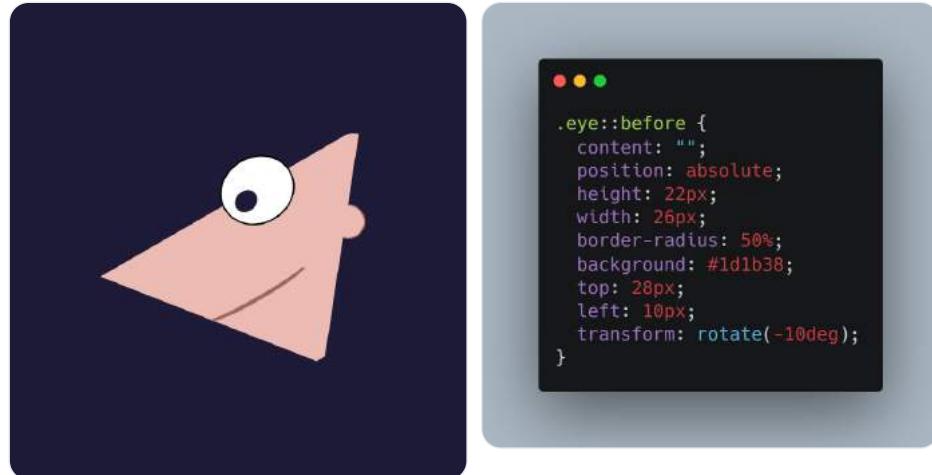
Let's create a outer white circle first



```
<div class="eye"></div>  
  
.eye {  
    position: absolute;  
    height: 70px;  
    width: 80px;  
    background: white;  
    border-radius: 50%;  
    border: 2px solid black;  
    transform: rotate(-30deg);  
    left: 100px;  
    top: -36px;  
}
```

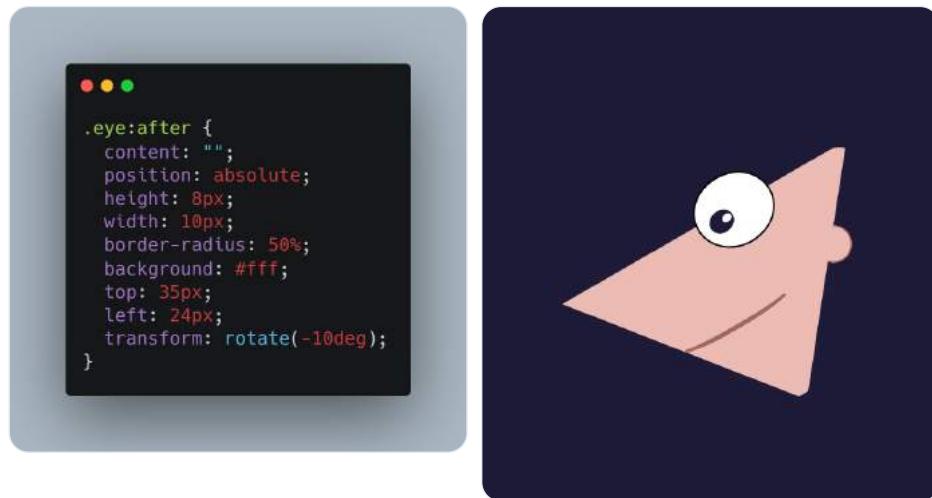
STEP 3(I). Create a black pupil

We can create pupil using pseudo element



STEP 3(II). Create white reflection in black pupil

Again make use of pseudo element and create small little white reflection inside pupil



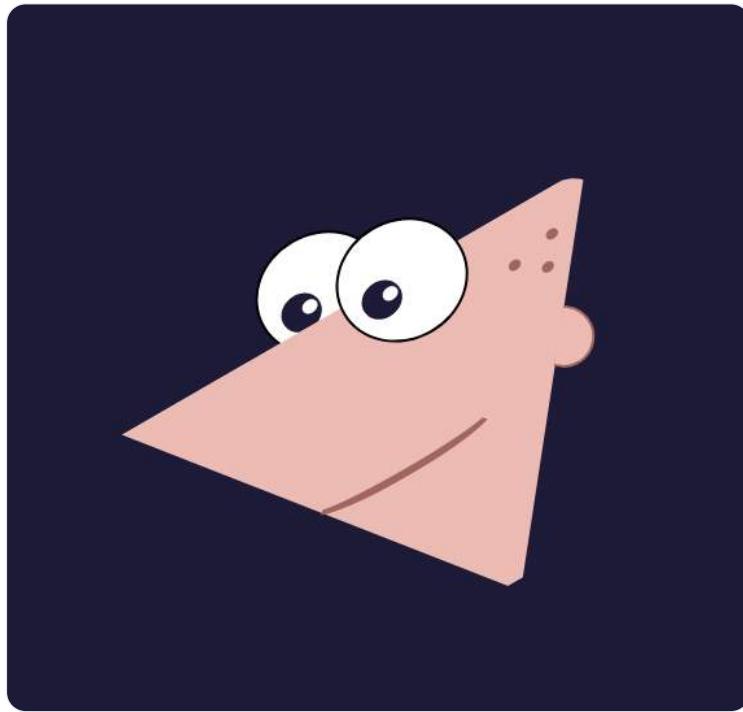
STEP 4. Create moles

Well CSS art is all about creative ideas and solution. In previous step we made oval shape white reflection. We can also create moles using shadow of the white element(reflection) and align them properly



STEP 5. Create second eye

Same as step 3



STEP 6. Create hair spike

This is the easiest part in my opinions because all you need to make are just random spikes.

What I have done here is just created a border-top 20px and rotate it in order to give it a spike look.

The image shows the same cartoon character from Step 5, now featuring a single red, spiky hair element on top of its head. To the right of the character are two code snippets. The top snippet is the HTML structure for the hair element:

```
<div class="hair">
<div></div>
<div></div>
<div></div>
<div></div>
<div></div>
</div>
```

The bottom snippet is the CSS styling for the first child of the hair container, which creates the red spike:

```
.hair>div:nth-child(1) {
  position: absolute;
  background: transparent;
  height: 30px;
  width: 100px;
  border-top: 20px solid red;
  border-radius: 50px 40px 0 0;
  bottom: 40px;
  left: 180px;
  transform: rotate(30deg);
}
```

...

STEP 7. Create hairs

Repeat step 6 and create multiple spikes so that they look like hairs 😊



That's all 😊

Thanks for reading this thread. Drop a link of your creations below. I love to check them out



Pratham 🎨🚀 @Prathkum

13 Jan • 6 tweets • Prathkum/status/1349283639192940544

Tr

Learn CSS by playing games🎨

THREAD🧵

📌 Grid Garden

- ◆ A game for learning CSS grid layout in 28 different levels



The screenshot shows a game interface titled "GRID GARDEN" at Level 4 of 24. On the left, there's a code editor with the following CSS:

```
gridarden { display: grid; grid-template-columns: 20% 20% 20% 20%; grid-template-rows: 20% 20% 20% 20%; } .carrot { grid-column-start: 1; }
```

Below the code editor, there's a text area with the instruction: "Try setting `grid-column-end` to a value less than 5 to water your carrots." On the right, there's a 4x4 grid representing a garden where several small green plants are growing.

Grid Garden
A game for learning CSS grid layout
<https://cssgridgarden.com/>

📌 Unfold

- ◆ Not a kind of game but here you will learn all about transform property in a more interactive manner



rupl.github.io/unfold/



📌 Flexbox Defense

- ◆ It covers the flex properties align-items, justify-content, flex-direction, align-self and order in 12 different levels



Flexbox Defense

Your job is to stop the incoming enemies from getting past your defenses. Unlike other tower defense games, you must position your towers using CSS!

<http://www.flexboxdefense.com/>

📌 CSS Diner

- ♦ In this game you will learn all about CSS selectors. It contains 32 levels. Try to master them



Select the apple on the plate

CSS Editor style.css HTML Viewer

```
1 Type in a CSS selector  enter
2 
3 /* Styles would go here. */
4 
```

```
1 <div class="table">
2   <bento></bento>
3   <plate>
4     <apple/>
```

CSS Diner
A fun game to help you learn and practice CSS selectors.
<https://flukeout.github.io/>

📌 Flexbox froggy

- ♦ In this game you will learn all about flexbox by writing small code snippet. Definitely check it out



FLEXBOX FROGGY Level 3 of 24

Now use `align-items` to help the frogs get to the bottom of the pond. This CSS property aligns items vertically and accepts the following values:

- `flex-start`: Items align to the top of the container.
- `flex-end`: Items align to the bottom of the container.
- `center`: Items align at the vertical center of the container.
- `baseline`: Items display at the baseline of the container.
- `stretch`: Items are stretched to fit the container.

```
1 #pond {
2   display: flex;
3   align-items: flex-end
4 }
```

Flexbox Froggy
A game for learning CSS flexbox
<https://flexboxfroggy.com/>

• • •



Pratham 🎨🚀 @Prathkum

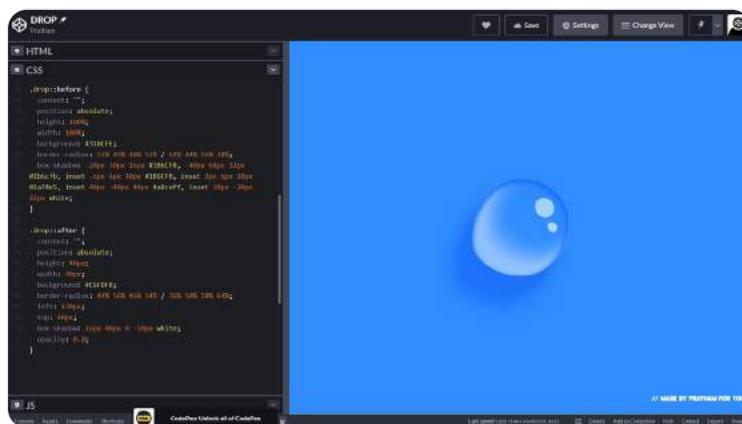
21 Jan • 13 tweets • [Prathkum/status/1352181742682202112](https://twitter.com/Prathkum/status/1352181742682202112)

Tr

Yesterday I invented a new design concept I named "Dropmorphism" 😅

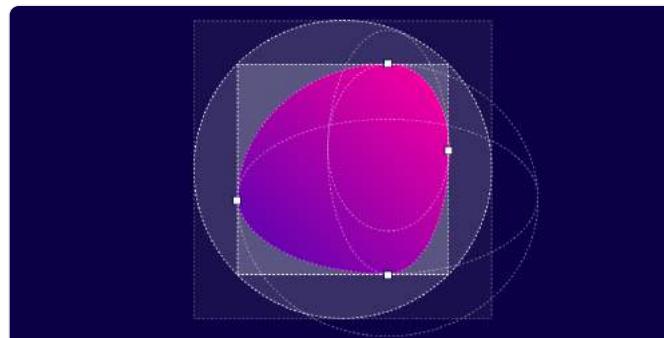
Let's see how I made it 👇

THREAD 🧶



⚠️ The heart of this art is box-shadow and border-radius

I created the shape of drop using Fancy border generator 👇



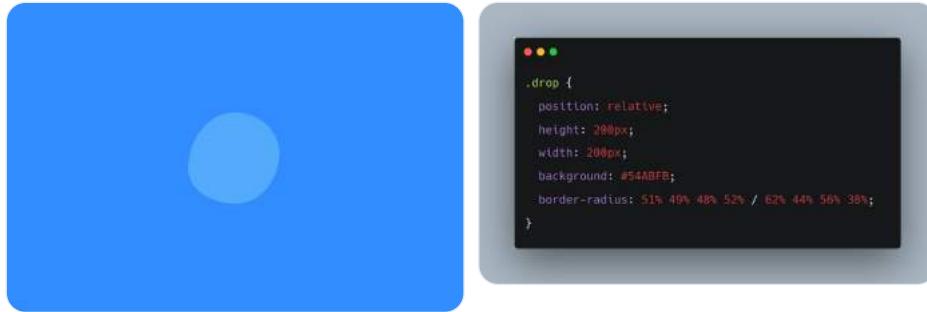
Fancy Border Radius Generator

A visual generator to build organic looking shapes with the help of CSS3 border-radius property

<https://9elements.github.io/fancy-border-radius/>

Now you a drop ready, We are good to go.

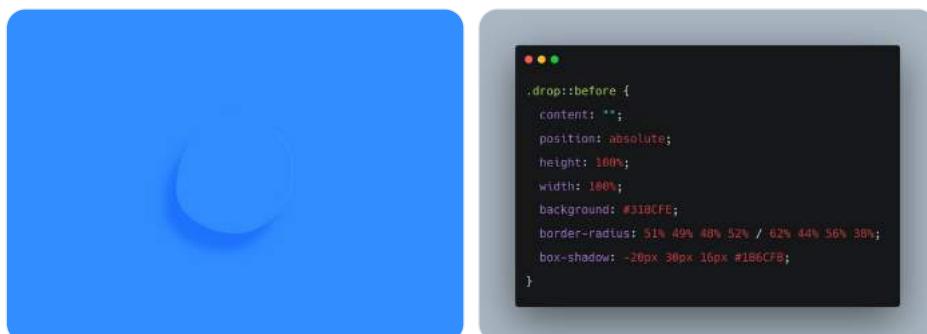
- Create a blue background and centralize your drop



- Add some opacity and border of almost same shade.
- Adding a border of approximately the same color will not be visible at a glance, but it will add some sort of volume to your drop

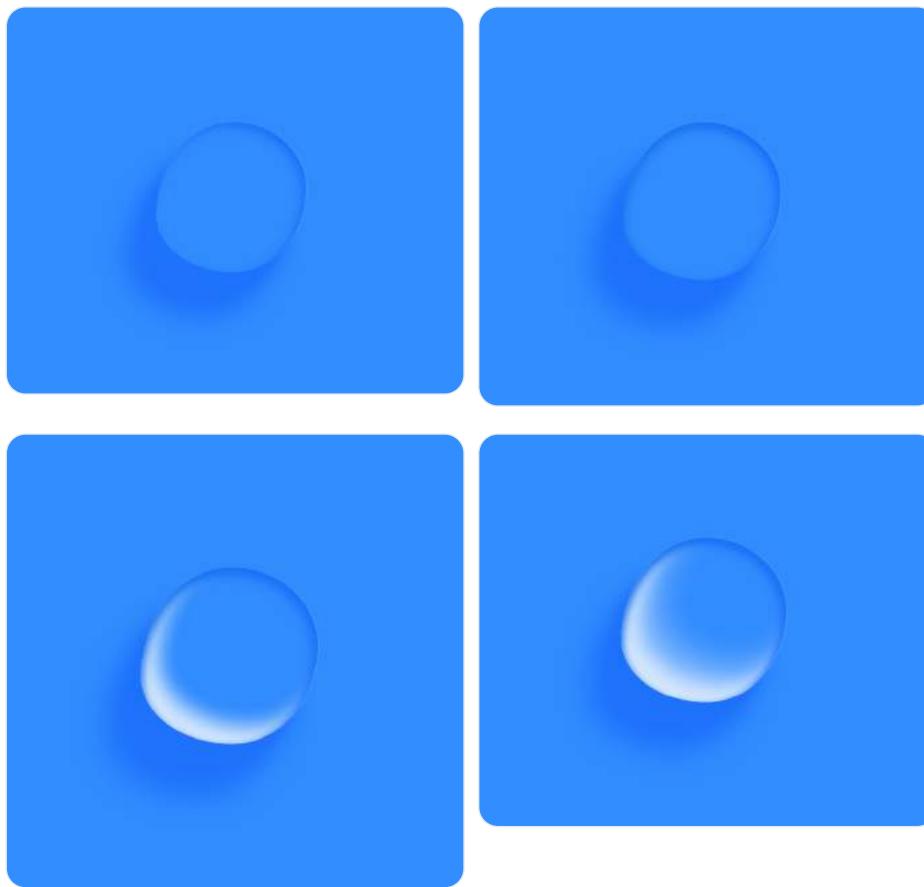


- Well it does not look like a drop. So how do we make it?
- We will use pseudo-elements for adding volume(Layers) in our drop
- Add a layer using ::before pseudo-element
- Height, width and border-radius should be same as drop
- Background color should be same as body color
- Add some dark blue shadow in order to give it a 3d look



- Extend the same shadow
- Now add four layers of inset shadows
 - 1. Blue inset shadow at right side
 - 2. Blue inset shadow on top

3. White inset shadow at bottom-left
4. White inset shadow at bottom-left with large intensity

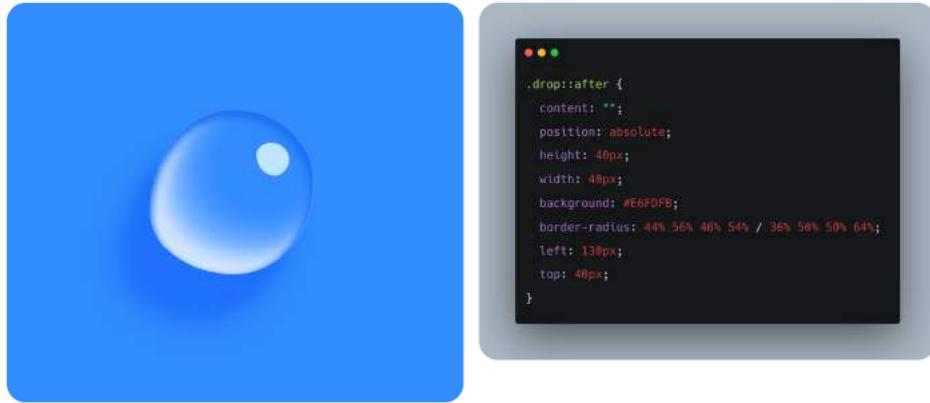


- CSS code for shadows mentioned in above tweet 

```
● ● ● .drop::before { box-shadow: -20px 30px 16px #1B6CFB, -40px 60px 32px #1b6cfb, inset -6px 6px 10px #1B6CFB, inset 2px 6px 10px #1a74e5, inset 20px -20px 22px #ffffff, inset 40px -40px 44px #a8ceff; }
```

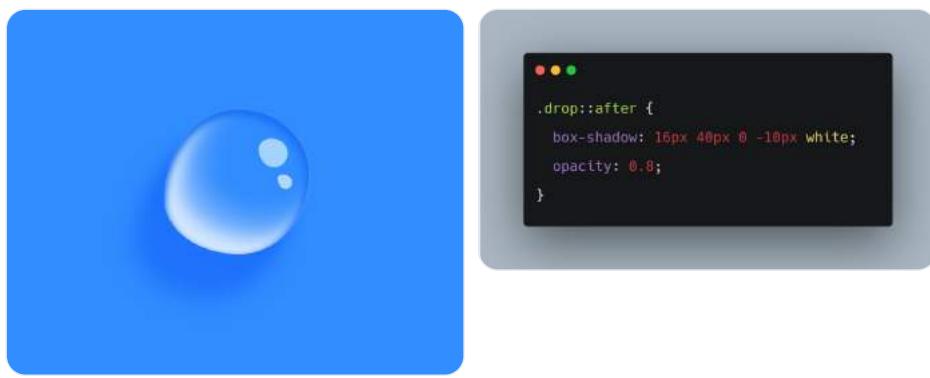
- Now will create a small white reflection using ::after pseudo-element.

- Create a blob and align it above the white intensity as shown in the given image below



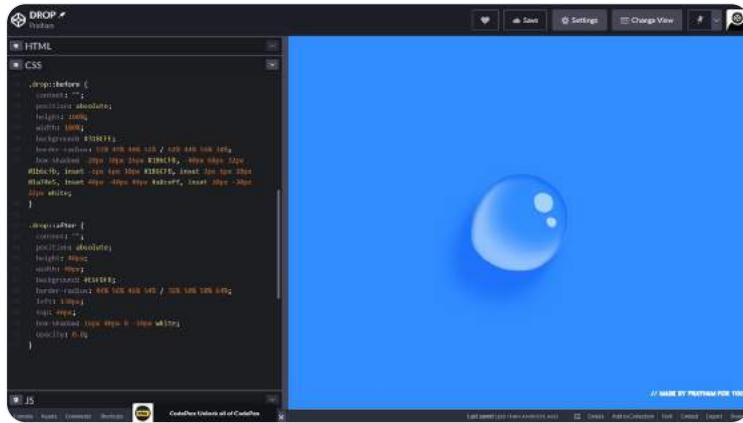
- Create another small white reflection using box-shadow of ::after-element

- Reduce opacity



Drop the link of your creations below. I'm more than happy to check it out

Final Output ↗



That's pretty much it. I hope you like it ❤️

I'll catch you with the next thread. Until then, keep coding 😎

SOURCE CODE:

<https://codepen.io/prathamkumar/pen/xxEMoZy>

→ Have* <—

...



Pratham 🎨🚀 @Prathkum

31 Jan • 14 tweets • [Prathkum/status/1355830282247692288](https://twitter.com/Prathkum/status/1355830282247692288)

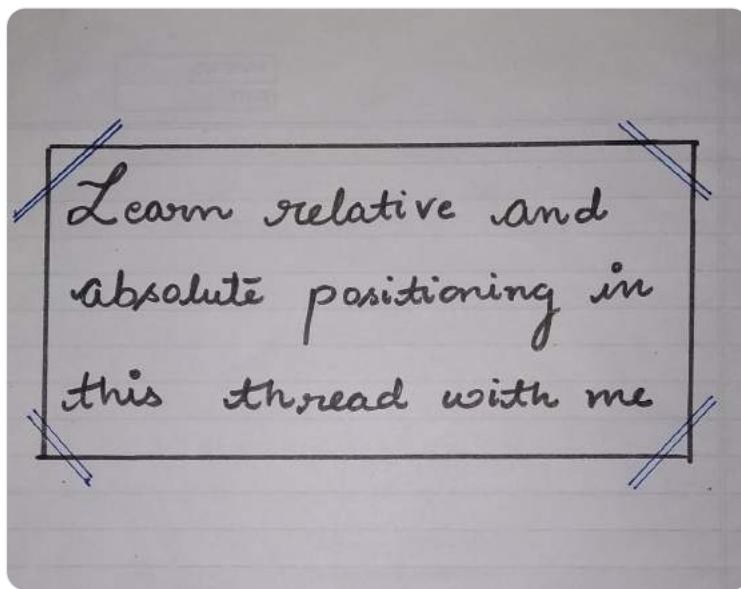
Tr

Positioning in CSS allows you to display your element wherever you want on the screen

But when I was learning it, I found it little bit confusing 😊

So in this thread I'll try to explain it in easiest manner with practical implementation. Let's start

THREAD



There are 5 values that you can pass in position property

- static
- relative
- absolute
- fixed
- sticky

In this thread we will be focusing on relative and absolute positioning as both are widely used

Let's start with understanding what document flow is?

👉 Elements are displayed on the screen as they written in the HTML document

Consider the following piece of code:

H1, P, H3 and div are displayed on the screen in exact order as they written in the HTML file.



As now you know about document flow, let's start with Relative positioning

👉 Relative Position

- Relative positioning do not take an element out of document flow
- Relative positioning is relative to element's original position which can be changed using offset
 - ◆ Relative position is relative to itself.

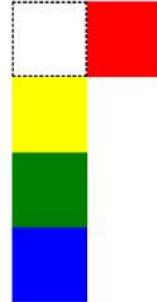
For example: Consider the code and output in the attached image below

As you can see red box is shifted 100px from left because I applied left offset after giving it relative positioning



In the attached image below, the black dotted area would be the original position of red box if I don't apply position relative in it.

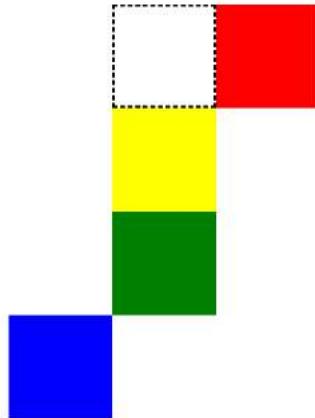
As you can see it proved that relative position is relative to itself



So now let me shift the blue box 100px towards left. So how can I do that? it's simple

```
.blue {  
position: relative;  
right: 100px;  
}
```

Notice here that document flow is as it is. So the relative position does not affect the document flow



📌 Absolute Position

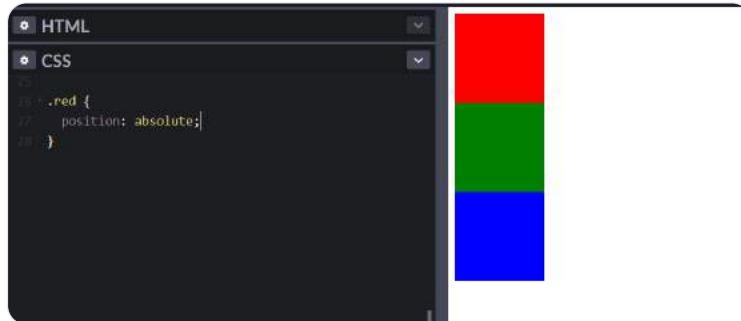
- The element is removed from the normal document flow
- You can consider it as, after applying absolute position the element will no longer in the flow and no space is created for the element in the page layout

For example:

If I apply absolute position in the red box, then the red box will be out of the flow and hence no space will be allocated to it.

See the image below, red box is out of flow and hence yellow box is at top and followed by green and blue

* Yellow box is below red



- The absolute position of an element is relative to its closest ancestor, which has some position property.

Consider the code below, Red is the parent div and black is the child div. In this particular case, body is the parent of red div



Now let me apply relative position to red(parent) div and absolute position to black(child) div.

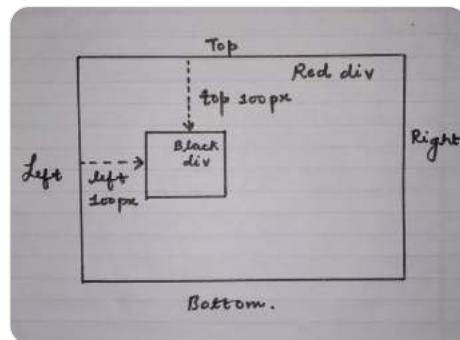
As I mentioned absolute position is relative to closest ancestor having some position property

The terminal window shows the following CSS code:

```
/* RED DIV */
.parent {
    position: relative;
}

/* BLACK DIV */
.child {
    position: absolute;
    top: 100px;
    left: 100px;
}
```

The browser preview window shows a red square containing a smaller black square at the top-left position defined by the CSS.



Let's understand it in little more details

Consider this piece of code.

Here green div is a parent of red and red div is a parent of black

The terminal window shows the following HTML and CSS code:

```
<div class="first-parent">
    <div class="second-parent">
        <div class="child"></div>
    </div>
</div>
```

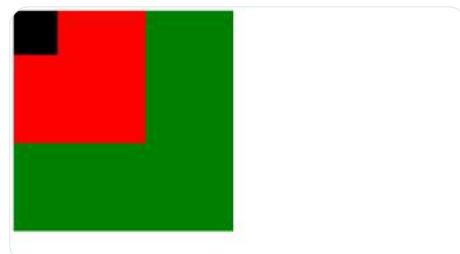
The right-hand code block shows the corresponding CSS:

```
.first-parent {
    width: 500px;
    height: 500px;
    background: green;
}

.second-parent {
    height: 300px;
    width: 300px;
    background: red;
}

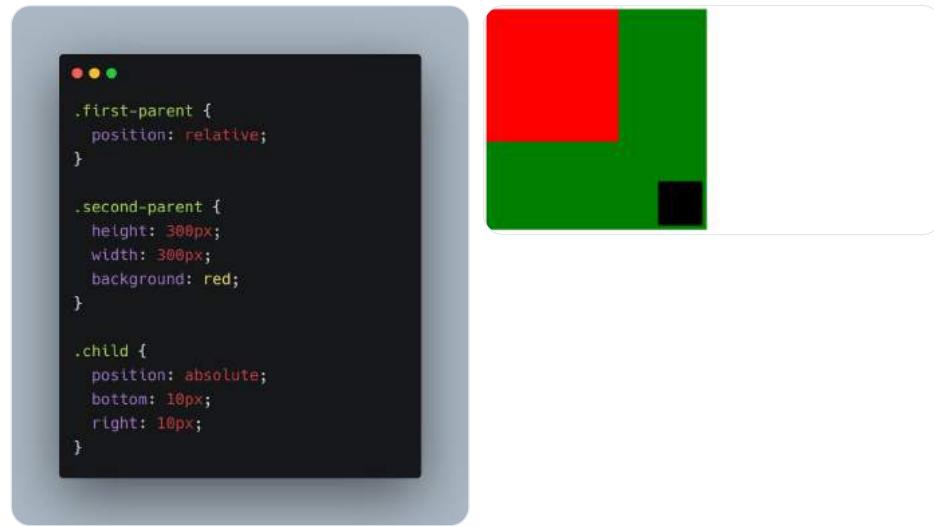
.child {
    height: 100px;
    width: 100px;
    background: black;
}
```

The browser preview window shows a green square containing a red square, which contains a small black square in the top-left corner.



So let me apply position property in green and black. In black div we have absolute position so in that case black div will be relative to green not red

Because here black's closest ancestor is green which has some position property



The image shows a screenshot of a code editor and a corresponding visual representation of the rendered HTML. The code editor on the left contains the following CSS:

```
.first-parent {  
    position: relative;  
}  
  
.second-parent {  
    height: 300px;  
    width: 300px;  
    background: red;  
}  
  
.child {  
    position: absolute;  
    bottom: 10px;  
    right: 10px;  
}
```

To the right of the code editor is a visual representation of the rendered HTML. It consists of three main colored boxes: a red box at the top, a green box below it, and a small black box in the bottom-right corner. The red and green boxes overlap, illustrating the concept of nested positioned elements.

I think that's pretty much it for this thread. I hope you get a overview of CSS positioning

This may sound a bit confusing but try to play with code. You'll be able to build better understanding 😊

Feel free to post your doubts below ❤️

• • •



Pratham 🎨🚀 @Prathkum

2 Feb · 9 tweets · [Prathkum/status/1356618477382078469](https://twitter.com/Prathkum/status/1356618477382078469)

Tr

Use the border radius effectively and create awesome shapes 😊

A Quick Thread 🧶 ↗



You might have used something like this before 👇

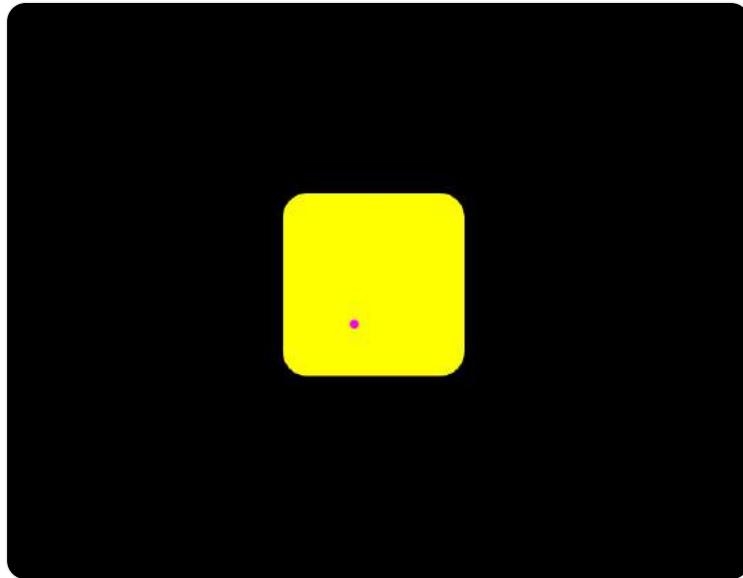
```
border-radius: 10px;  
border-radius: 50%;  
border-radius: 20px 10px;  
border-radius: 10px 20px 30px 40px;
```

but have you used this? 👇

```
border-radius: 40% 22% 33% 45% / 45% 69% 65% 36%;
```

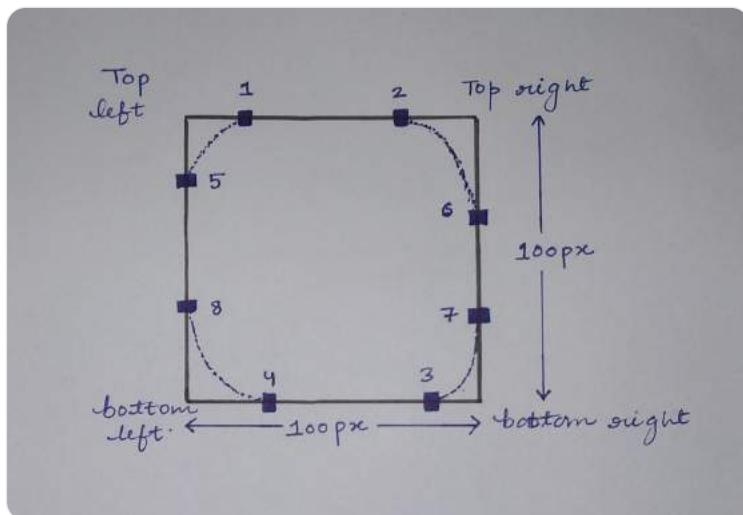
Attached image showing a rectangle with 10px border-radius all sides. This is what we probably all know.

```
border-radius: 20px;
```



In this thread I'll try to explain you the key points behind border-radius. Let's start

Consider border-radius as 8 clips(see the attached image)



We can shift the position of eight clips and make some amazing random shapes.

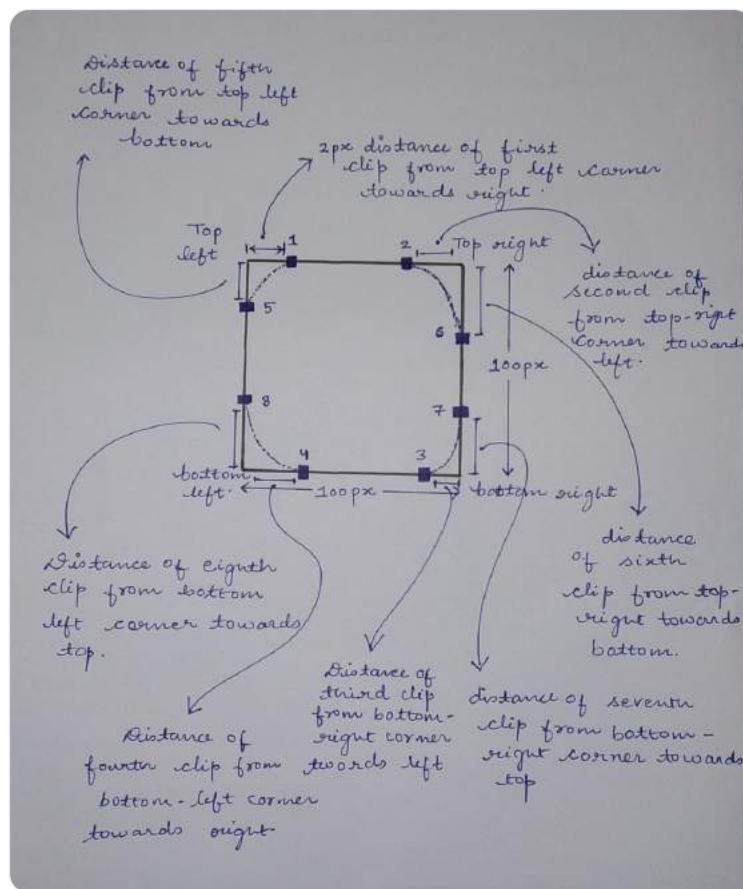
The syntax is pretty easy, we can consider it as border-radius accepts 8 value for each clip, four value before slash (/) and four values after slash



Let's decode what each value means.

border-radius: 2px 4px 6px 8px / 10px 12px 20px 16px;

The first 2px simply means the distance of first clip from the top left corner and second value that is 4px means the distance of second clip from top right corner (See attached image)



When we write simple border-radius: 10px, it means all clips and equal distance from their initial points.

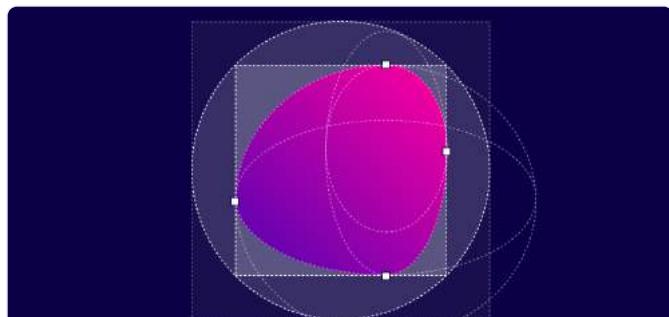
border-radius: 50% 50% 50% 50% / 50% 50% 50% 50%; is same as border-radius: 50%

I hope you get a quick overview of the border-radius and now you will be able to make some random shapes using only border-radius.

That's pretty much it for this thread. Drop your thoughts below ❤️

Extra Reading ↗

Check out this amazing website



Fancy Border Radius Generator

A visual generator to build organic looking shapes with the help of CSS3 border-radius property

<https://9elements.github.io/fancy-border-radius/>

Read more about border-radius

<https://developer.mozilla.org/en-us/docs/Web/CSS/border-radius>

• • •



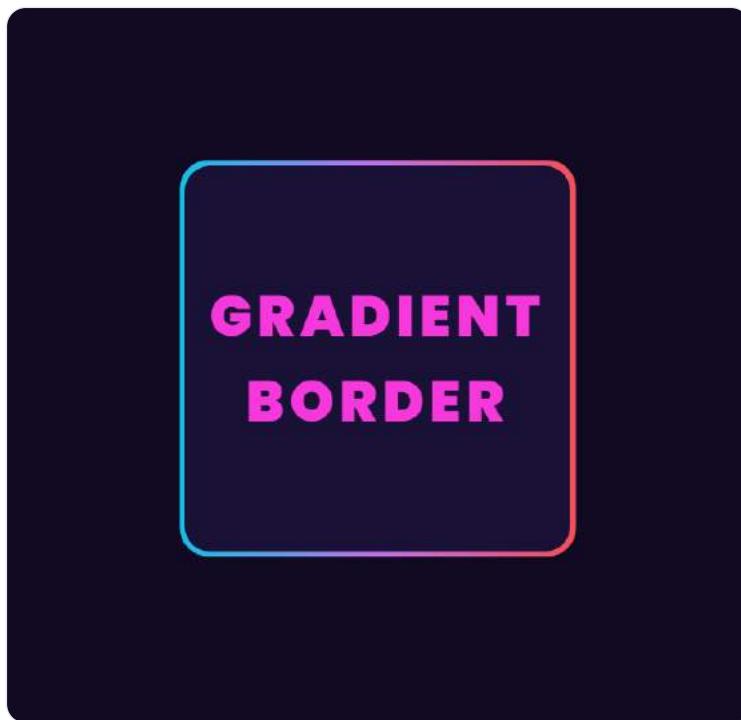
Pratham 🎨🚀 @Prathkum

6 Feb · 6 tweets · [Prathkum/status/1358077840856612866](#)

Tr

We can't create gradient border directly but we have a trick 😊

A SHORT THREAD 🧶



This is pretty easy. It will just take 5 minutes 😊

STEP 1:

- Create your element around which you want to create gradient border

In this example I have a square around which I'll create gradient border

STEP 2:

- Create pseudo-element with little extra height and width

Let's say if my square height width is 300px then I'll set 304px height and width of pseudo-element

STEP 3:

- Set gradient background of pseudo-element and place it at center

I hope now you get the trick 😊 If you continue reading 👇



```
.container::before {  
background: linear-gradient(to right, #13c2b9, #c471ed, #f64750);  
border-radius: 20px;  
left: 5%;  
top: 5%;  
transform: translate(-5%, -5%);  
}
```

STEP 4:

- Add z-index to pseudo-element and send it behind the actual element

That's all ❤️



```
.container::before {  
z-index: -1;  
}
```

Wasn't it easy and quick? 🎉

Thanks for reading this 🙏😊

• • •



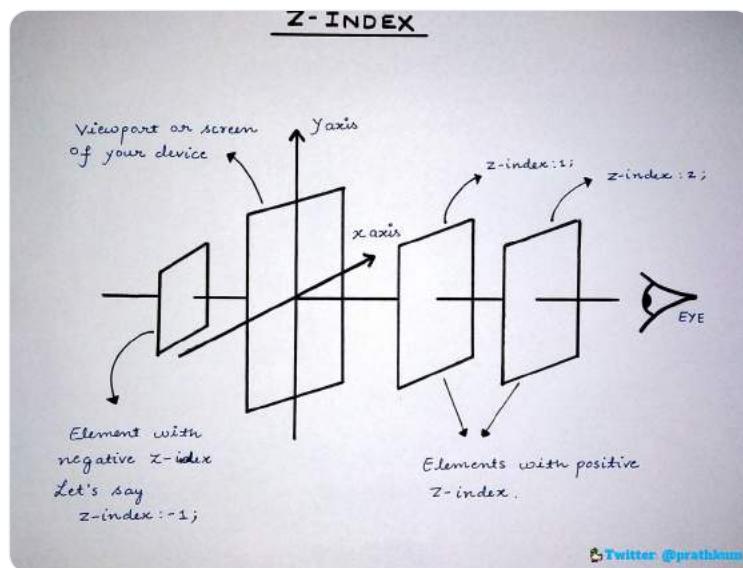
Pratham 🎨🚀 @Prathkum

19 Feb · 5 tweets · [Prathkum/status/1362686461544439809](https://twitter.com/Prathkum/status/1362686461544439809)

Tr

The Z-index is a powerful yet confusing concept of CSS

Let's make it easy in this quick thread 👇



Twitter: @prathkum

z-index is a CSS property that controls stacking order of elements along Z axis.

Image a hypothetical line starting from your eye to screen, that is Z-axis

★ Note that z-index only works on positioned elements.

You need to specify the position (relative, absolute, sticky, fixed) if you want to arrange an element using z-index

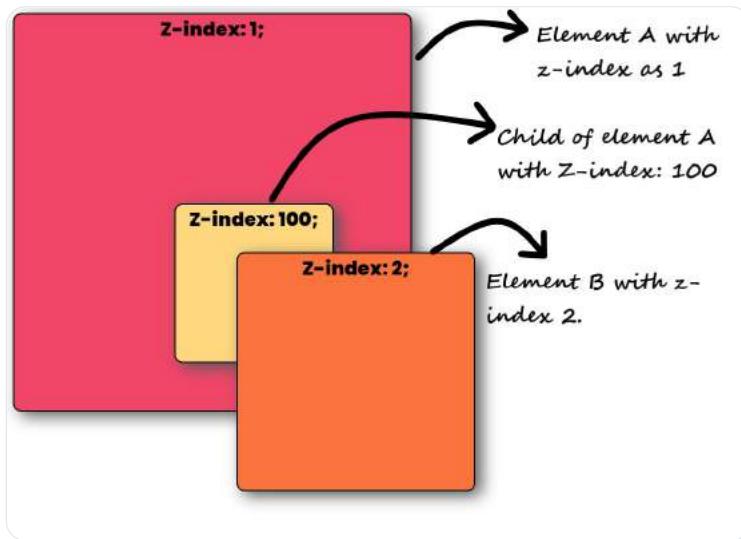
```
BAD
.element {
  z-index: 1;
}

GOOD
;element {
  position: relative;
  z-index: 1;
}
```

📌 z-index of nested elements

Let's say two elements A and B are siblings with element B written after element A in DOM, then the children of element A cannot be higher than element B no matter what z-index is being applied on children

See this in action 



Play around with code for better understanding. Any doubts? Post below 

<https://codepen.io/prathamkumar/pen/WNoOvYr>

• • •



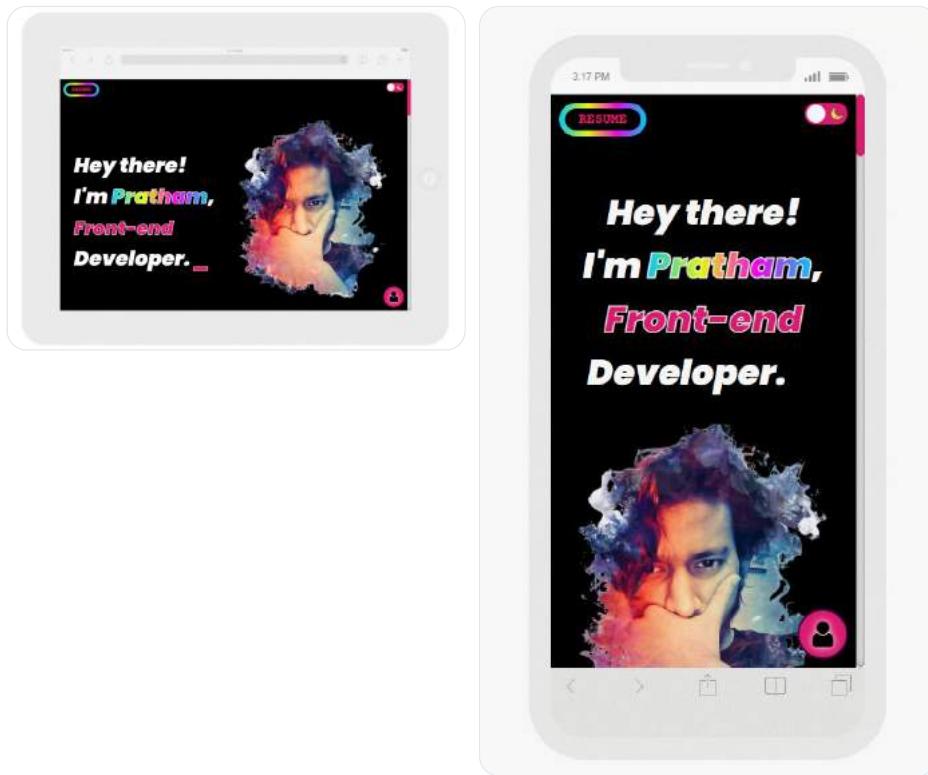
Pratham 🎨🚀 @Prathkum

20 Feb · 11 tweets · [Prathkum/status/1363063942994685952](https://twitter.com/Prathkum/status/1363063942994685952)

Tr

A quick beginner's guide to CSS Media Queries

THREAD



Media queries plays an important role in Responsive Web Design

Though this is not the only use of it. Media queries can also be used as

- Apply different styles for different media types
- Orientation (landscape or portrait mode)
- Resolution

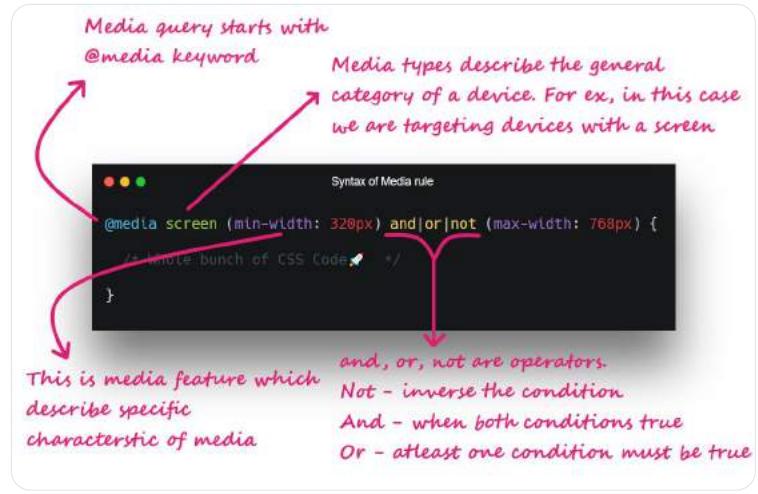
In this thread, our point of focus will be using Media Query in order to make a web page responsive.

Sounds interesting? Let's go 👏

Let's start with the syntax first (See attached image)

There are a lot of media features. Have a look at this table

https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries#media_features



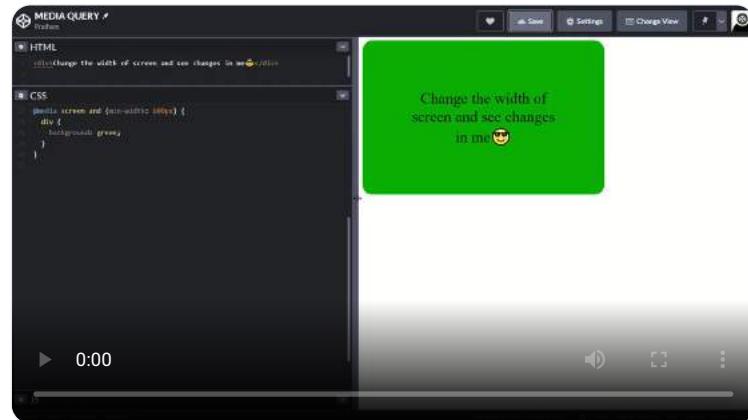
Let's see an example into action. So that we can build strong hold on the concept

.....

```
@ media screen and (min-width: 600px) {
div {
background: green;
}
}
```

.....

In this case the background will be green whenever the screen width is 600px or greater than 600px;



You can also combine two media features using "And", "Not" or ","

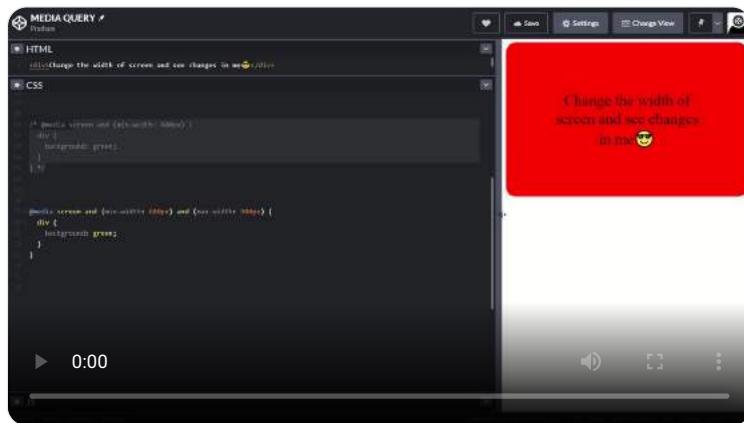
For example:

.....

```
@ media screen and (min-width: 600px) and (max-width: 900px) {
div {
background: green;
}
}
```

.....

In this case, the background will be green between 600px to 900px width



You can try bunch of media features

- any-hover
- color
- height
- width
- grid
- aspect-ratio
- orientation
- resolution, etc...

We can also write the nested media queries:

When min-width = 600px and user hover over element then solid black will create around div.



You can also invert the media query by simply adding the "not" keyword after @media

"not" keyword simply invert the condition. That is, in this we can say that now condition becomes max-width: 600px

```
@media not screen and (min-width: 600px) {  
    div {  
        border: 10px solid black;  
    }  
}
```

Sometimes managing media queries might be a tough task.
So whenever you're working on pure CSS, make sure to check some standard device viewports and try to minimize number of media queries in your code

I hope you get a quick overview of how media query works. The syntax of media query and stuff like that.

Read this MDN article for more info

https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries

• • •



Pratham 🎨🚀 @Prathkum

23 Feb · 14 tweets · [Prathkum/status/1364142557052219394](https://twitter.com/Prathkum/status/1364142557052219394)

Tr

All you need to know about CSS filter methods 🎨

A thread 🧶



Using filter functions you can change graphical effects of an image which can change the appearance of the input image.

You can do a lot of cool styling using one line of CSS. Let's see how

□ blur()

As the term suggests, blur function simply blurs your input image. The blur that is being applied on image is known as Gaussian blur.

The image shows a screenshot of a terminal window with a pink-to-purple gradient background. The window title is "Twitter: @prathkum". Inside, there is a black text area containing the following CSS code:

```
img {  
  filter: blur(2px);  
}
```

To the right of the terminal window is a large, rounded rectangular frame containing a portrait of a man with curly hair. The portrait has a soft, out-of-focus, and slightly blurry appearance, illustrating the effect of the blur filter applied to the image.

□ brightness()

Brightness function adds some visual perception in input image, making it appear brighter or darker

```
Twitter: @prathikam  
img {  
  filter: brightness(150%);  
}
```



contrast()

You can adjust the contrast of the input image. Contrast is like the difference in brightness between objects or regions

```
Twitter: @prathikam  
img {  
  filter: contrast(150%);  
}
```



drop-shadow()

box-shadow applies the shadow around rectangular box whereas drop-shadow automatically fits around the shape of an element

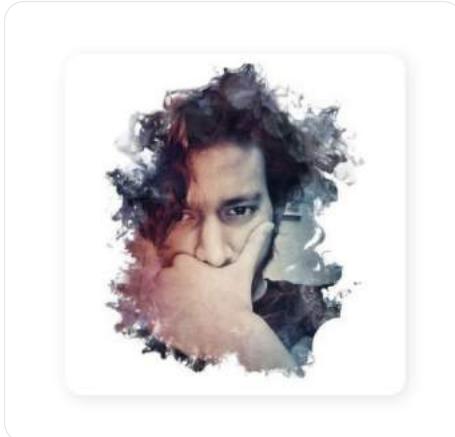
```
Twitter: @prathikam  
img {  
  filter: drop-shadow(20px 20px 20px #db3;);  
}
```





grayscale()

The grayscale() CSS function converts the input image to grayscale



hue-rotate()

Applies a hue rotation on the image. The value defines the number of degrees around the color circle the image samples will be adjusted

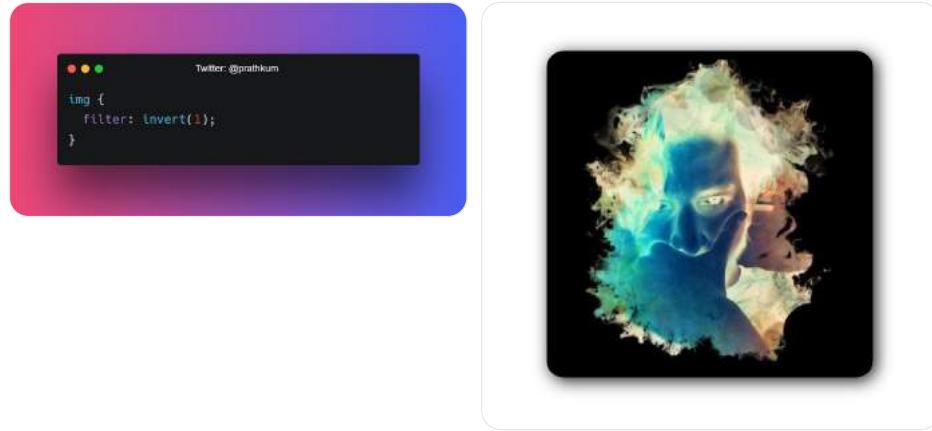


invert()

invert function simply invert the color of the element

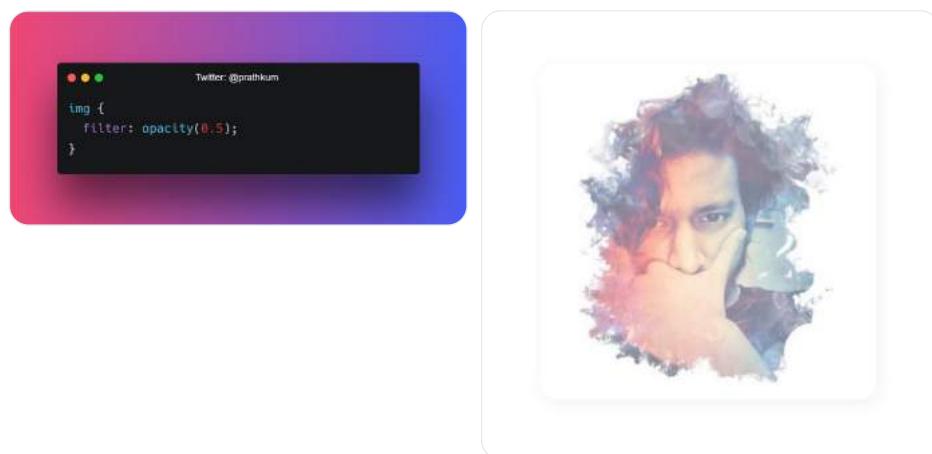
P.S.

The output image is littbe bit horrible 😅



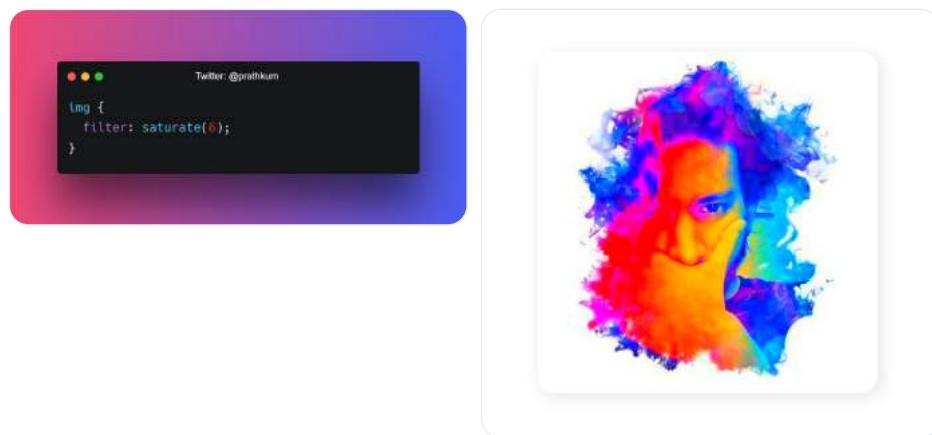
□ opacity()

opacity function applies transparency to the samples in the input image



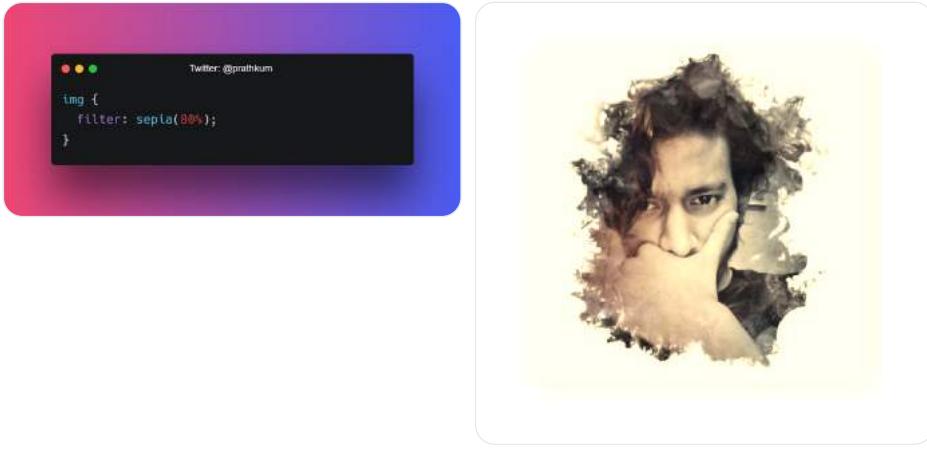
□ saturate()

The CSS saturate function helps to reduce or increase the saturation level of input image.



□ sepia()

The sepia() CSS function converts the input image to sepia, giving it a warmer, more yellow/brown appearance.



That pretty much it I guess. Did I miss something?

If you like this thread share it with your connection ❤

Play with with code here

<https://codepen.io/prathamkumar/pen/GRNvLMN>

• • •



Pratham 🎨🚀 @Prathkum

24 Feb · 18 tweets · [Prathkum/status/1364581954620907525](https://twitter.com/Prathkum/status/1364581954620907525)

Tr

You can learn some basic CSS and add some great styling in your web page in less than 10 days.

Let's see how 👇

Thread 🧶

First and foremost

The characteristic of a great website is its color scheme. Forget about everything and learn about background and color properties initially

{ 2 / 17 }

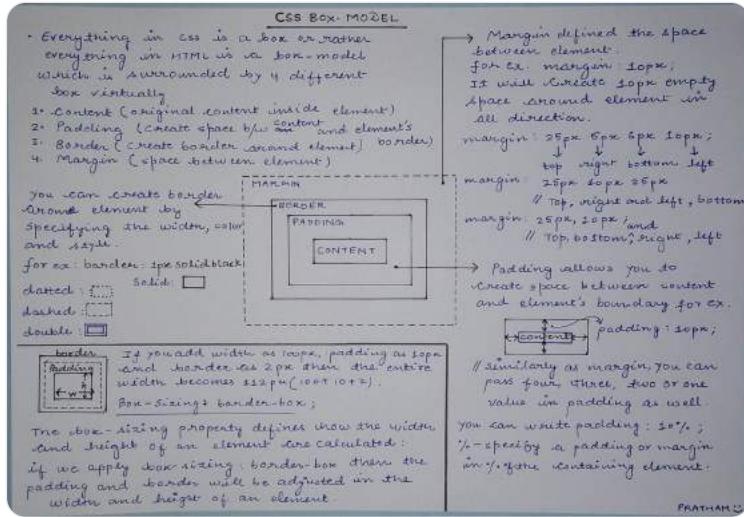
Don't think that background property is just for setting the solid color. Background is a shorthand property for background-image, background-position etc..

{ 3 / 17 }

Box model is one of the most important concept of CSS. It's not so tough to learn. The box-model covers

- Height
- Width
- Padding
- Border
- Margin

{ 4 / 17 }



Height and width property are pretty intuitive. These are used to set fixed height and width to the element

I suggest to give a look at max, min-width and max, min-height properties as well.

{ 5 / 17 }

Proper and uniform separation of elements is something that can give your webpage a appealing look. Margin and padding can do this for you.

Give this article a short read for Definitive guide of padding and margin



{ 6 / 17 }

Border are used to set the color, width and style to elements. You can learn it in 5 min 😊

Some good border selection can give your element a good pleasant look

{ 7 / 17 }

Moving forward, typography is an essential thing of web page. A good font can make your webpage and establish a strong visual hierarchy, provide a graphic balance to the website, and set the product's overall tone.

You can add free fonts from Google's official site

{ 8 / 17 }

There are five basic classifications of fonts:

1. serif
2. sans serif
3. script
4. monospaced
5. display

Give this article a read for more detailed explanation

<https://www.fonts.com/content/learning/fontology/level-1/type-anatomy/type-classifications>

{ 9 / 17 }

fonts .google.com (Download free fonts from here)

You just need to look at few fonts properties. For ex,

- ◆ font-family
- ◆ font-weight
- ◆ font-size

{ 10 / 17 }

Alright moving further, We have CSS positioning.

From here, a bit tricky CSS starts. Using CSS positioning you can change the position of your element. This might seem a bit tough but you can learn it in 2-3 days

{ 11 / 17 }

I have already written a detailed thread on CSS positioning, If you're interested check it out



Pratham
@Prathkum

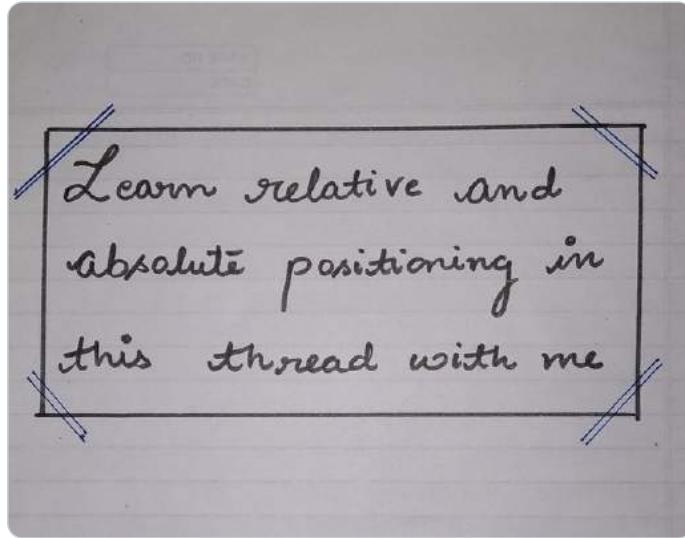


Positioning in CSS allows you to display your element wherever you want on the screen

But when I was learning it, I found it little bit confusing 😊

So in this thread I'll try to explain it in easiest manner with practical implementation. Let's start

THREAD  



10:48 AM · Jan 31, 2021

(i)

 948  38  Copy link to Tweet

{ 12 / 17 }

Up to this point you have some decent knowledge of styling your website. It would be great if you learn about a layout system now.

Flex and Grid

Grid is little bit tough to master but flex isn't

{ 13 / 17 }

Again 😊 I have a thread if flex layout as well. Check it out

<https://twitter.com/Prathkum/status/1361931227830358018?s=20>

{ 14 / 17 }

Hmm awesome!! I think you're all set to learn about responsiveness here.

Don't skip it. There are millions of devices on which your website is viewed.

{ 15 / 17 }

In this thread I have compiled some general tips to make your website responsive

<https://twitter.com/Prathkum/status/1362803479044104195?s=20>

{ 16 / 17 }

Great!! I think that's pretty much it in order to give you a quick overview.

If you like this thread, a retweet means a lot ❤

{ 17 / 17 }

Fun fact:

A few hours ago while I was writing this thread, my younger brother([@AmanKum1406](#)) comes and says that nobody will read it

Ruthless 😊

• • •



Pratham 🎨🚀 @Prathkum

28 Feb · 16 tweets · [Prathkum/status/1365851808963432451](https://twitter.com/Prathkum/status/1365851808963432451)

Tr

I have covered some important CSS topics in my thread in last couple of days.

Here is a quick sneak peek 🧵

📌 Relative and Absolute positioning

Positioning in CSS allows you to display your element wherever you want on the screen

But when I was learning it, I found it little bit confusing 😅

So in this thread I'll try to explain it in easiest manner with practical implementation. Let's start

THREAD🧵

Learn relative and absolute positioning in this thread with me

10:48 AM · Jan 31, 2021

1 ①

948 38 Copy link to Tweet

📌 Responsive web design

<https://twitter.com/Prathkum/status/1362803479044104195?s=20>

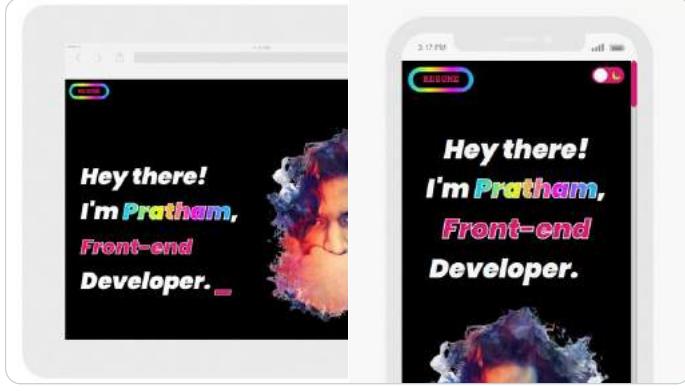
...

🐦 Media query



A quick beginner's guide to CSS Media Queries

THREAD 



9:52 AM · Feb 20, 2021 

Heart 535 Reply 20  Copy link to Tweet

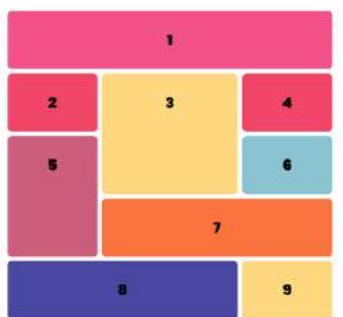
🐦 Flex box layout



<https://twitter.com/Prathkum/status/1361931227830358018?s=20>

A quick guide to CSS Grid layout 

Thread 



9:54 AM · Feb 27, 2021 

Heart 2.6K Reply 49  Copy link to Tweet

🐦 Filter functions



Pratham
@Prathkum



All you need to know about CSS filter methods 

A thread 



9:18 AM · Feb 23, 2021



 481  18 ⌂ Copy link to Tweet

📌 Basic CSS



Pratham
@Prathkum



You can learn some basic CSS and add some great styling in your web page in less than 10 days.

Let's see how 

Thread 

2:24 PM · Feb 24, 2021



 1K  18 ⌂ Copy link to Tweet

📌 Z-index

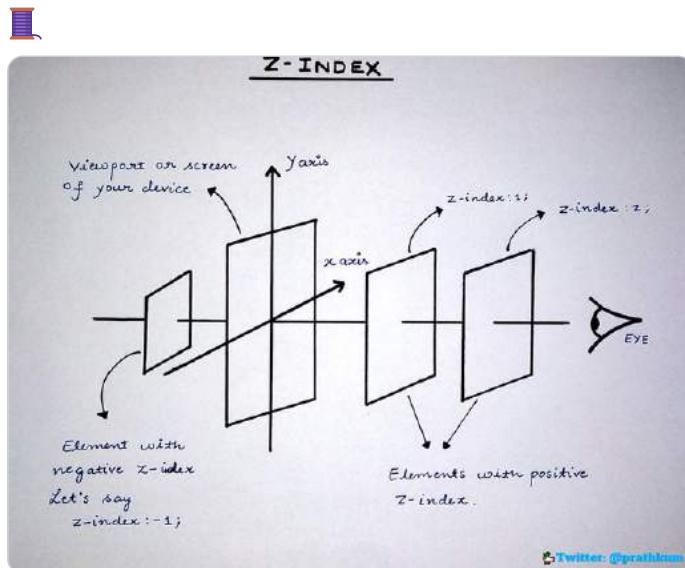


Pratham
@Prathkum



The Z-index is a powerful yet confusing concept of CSS

Let's make it easy in this quick thread 🙏



Twitter: @prathkum

8:52 AM · Feb 19, 2021



Heart 934 Comment 33 Copy link to Tweet

📌 Border-radius



Pratham
@Prathkum



Use the border radius effectively and create awesome shapes 😊

A Quick Thread 📜 🙏



3:00 PM · Feb 2, 2021



Heart 382 Comment 17 Copy link to Tweet

📌 CSS terminologies

 **Pratham**
@Prathkum 

Some common words used in CSS

Thread  



The diagram illustrates various CSS concepts arranged around a central word 'terminology'. At the top center is 'terminology'. To its left is 'rules' (with 'block selector pseudo' below it), and to its right is 'selector' (with 'elements combinator statement' above it). Below 'terminology' is 'CSS' (with 'property value rules' above it). To the left of 'CSS' is 'css terminology property' (with 'css terminology' below it), and to the right is 'statement keywords attribute'. In the center, below 'terminology', are 'css terminology css' and 'terminology css terminology'. To the left of these is 'property' (with 'terminology property property' above it), and to the right is 'pseudo' (with 'combinator statement keywords' above it). At the bottom, there are two small boxes: one for 'class' (with 'class pseudo' below it) and one for 'pseudoclass'.

6:48 AM · Feb 6, 2021 

 132  2  Copy link to Tweet

📌 How to make gradient border



Pratham
@Prathkum



We can't create gradient border directly but we have a trick 😊

A SHORT THREAD 



3:39 PM · Feb 6, 2021



 408  16  Copy link to Tweet

📌 Create different shapes

<https://twitter.com/Prathkum/status/1351109173841432577?s=20>

📌 Dropmorphism



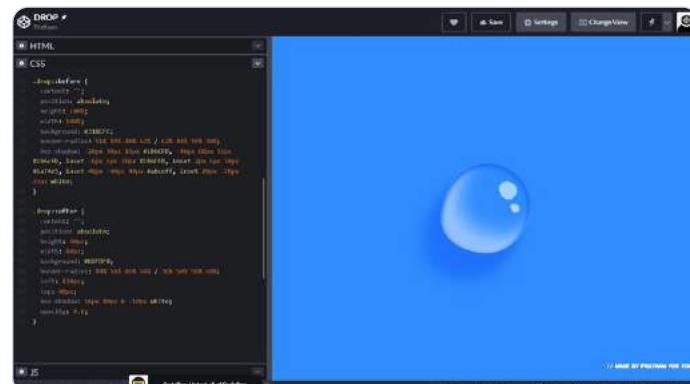
Pratham
@Prathkum



Yesterday I invented a new design concept I named "Dropmorphism" 😂

Let's see how I made it 👇

THREAD



A screenshot of a browser window. On the left, there is a CSS code editor with the following code:

```
.drop{position: absolute; width: 100px; height: 100px; background-color: #33CCCC; border-radius: 50%; border: 2px solid black; box-shadow: 0 0 10px 5px black; transition: width 1s, height 1s, border-radius 1s, border 1s;}.drop::before{content: ""; position: absolute; width: 10px; height: 10px; background-color: white; border-radius: 50%; border: 1px solid black; right: -5px; top: -5px; transform: rotate(45deg);}
```

The preview on the right shows a blue circle with a smaller white circle inside it, representing a drop or bubble.

9:10 AM · Jan 21, 2021



 481  18  Copy link to Tweet

👉 Play games



Pratham
@Prathkum



Learn CSS by playing games

THREAD

9:14 AM · Jan 13, 2021



 1.9K  33  Copy link to Tweet

I think there are many more, but I have not been able to find them 😅

Also if you want me to write on a specific topic tell me below 😊



Pratham 🎨🚀 @Prathkum

18 Mar · 9 tweets · Prathkum/status/1372624098568863752

Tr

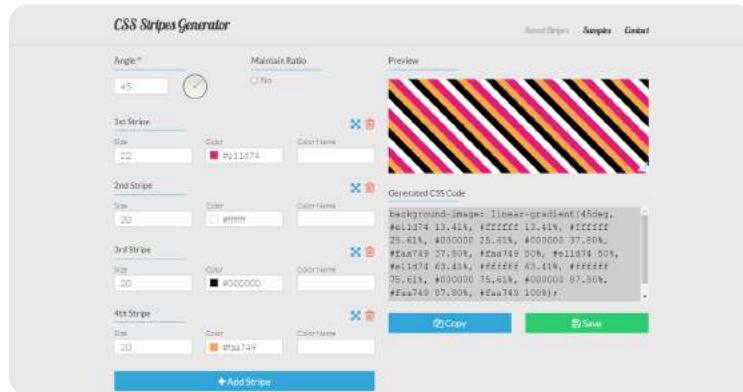
Do not write CSS code, use these free generators instead that can help you immensely

A Thread 🧵

1 Stripes generator

- Pure CSS Stripes Generator that you can use for backgrounds.

🔗 stripesgenerator.com

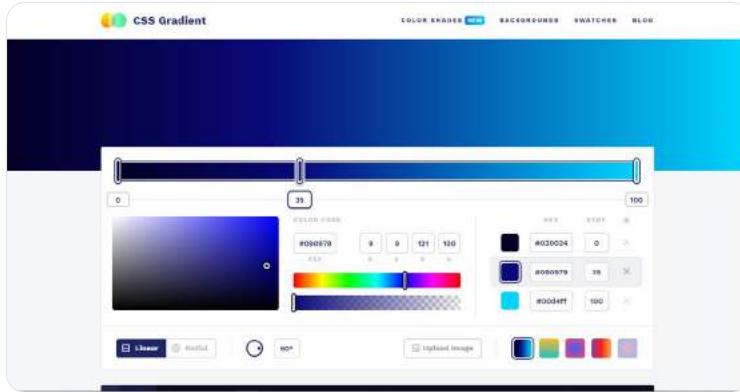


2 Gradient generator

- As a free css gradient generator tool, this website lets you create a colorful gradient background for your website, blog, or social media profile

🔗

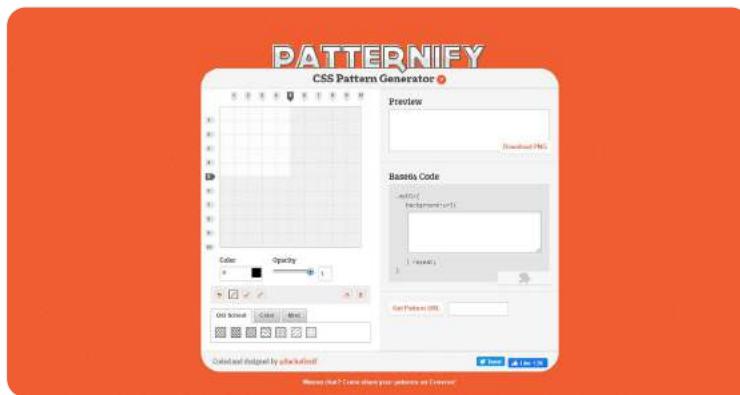
The screenshot shows the 'CSS Gradient' generator interface. It features a large preview area with a color gradient from green to blue. Below it is a color picker with sliders for 'Hue', 'Saturation', and 'Lightness'. To the right are color hex codes: #00E077, #00D95B, #40A977, and #00C64F. At the bottom, there is a summary section with the title 'CSS Gradient — Generator, Maker, and Background' and the text: 'As a free css gradient generator tool, this website lets you create a colorful gradient background for your website, blog, or social media profile.' Below that is the URL 'https://cssgradient.io/'.



3 Pattern generator

- It lets you create background pattern for free

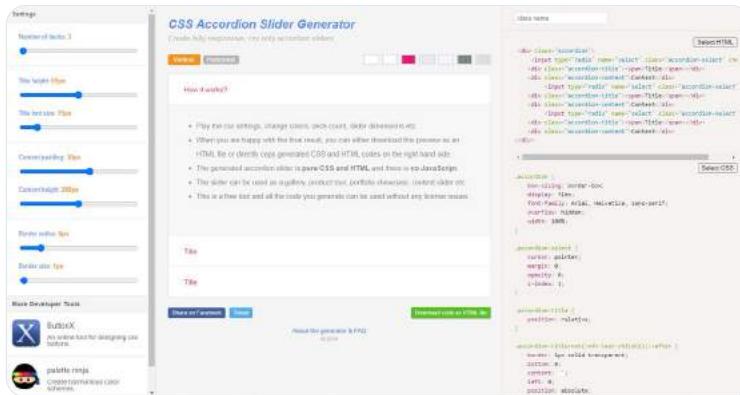
patternify.com



4 CSS Accordion Slider Generator

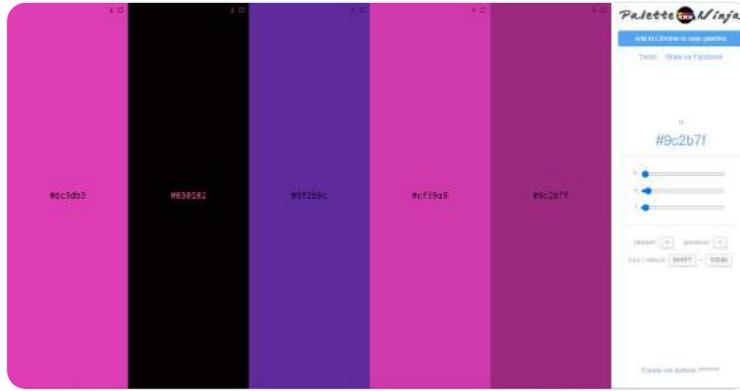
- Create fully responsive, css only accordion sliders

accordionslider.com



5 Palette Ninja

- Palette ninja is an online color scheme generator that allows you to create harmonious color schemes in seconds.



6 Layout generator

- Quickly design web layouts, and get HTML and CSS code. Learn CSS Grid visually and build web layouts with our interactive CSS Grid Generator.

Interactive CSS Grid Generator | Layoutit Grid

Quickly design web layouts, and get HTML and CSS code. Learn CSS Grid visually and build web layouts with our interactive CSS Grid Generator.

<https://grid.layoutit.com/>



7 □ Animation generator

- Dead simple visual tools to help you generate CSS for your projects.





A simple CSS toolbox for generating animations, shadows, colors, & more.

Keyframes.app | CSS Toolbox
Keyframes helps you write better CSS with a suite of tools to create CSS @Keyframe animations, box shadows, colors, & more
<https://keyframes.app>



8 □ Grid Layout generator

- The Quickest & Easiest Way To Build Complex CSS Grid Layouts.

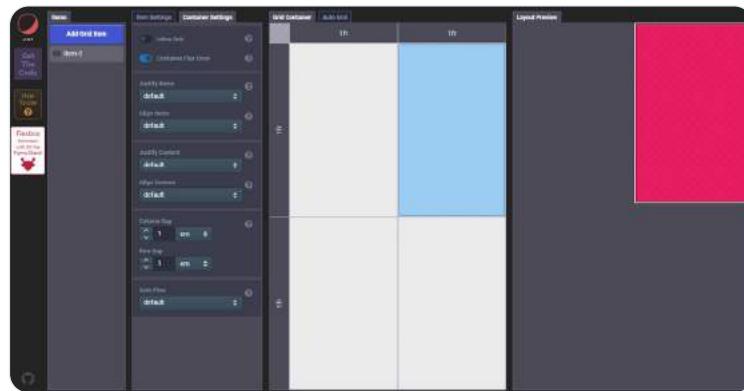




The Quickest & Easiest Way To Build Complex CSS Grid Layouts

The Quickest & Easiest Way To Build Complex CSS Grid Layouts

<https://css-grid-layout-generator.pw/>





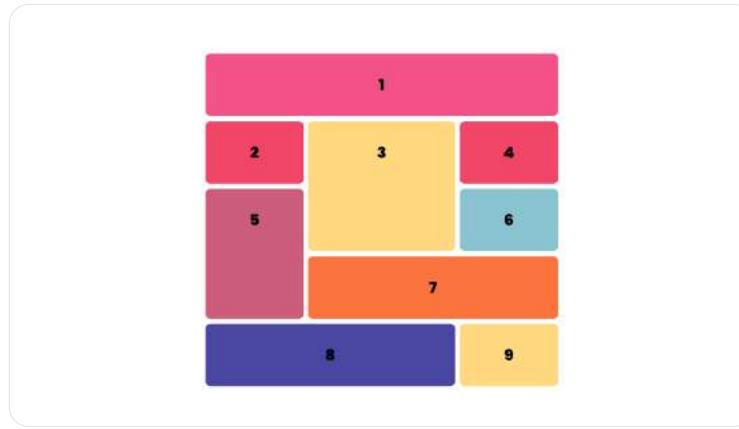
Pratham @Prathkum

24 Mar • 14 tweets • [Prathkum/status/1374652212928987137](https://twitter.com/Prathkum/status/1374652212928987137)

Tr

A complete beginner's guide to CSS Grid layout

Thread



Grid is used for making complex web design layouts more easily as it's not so hard to master

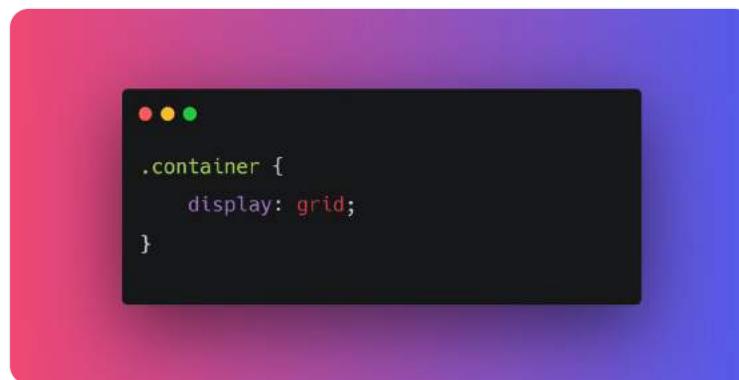
Using Flex you can make only 1D layout but Grid gives you full power of creating 2D layout

Let's start

{ 2 / 21 }

First things first, start with giving the display property "grid" to the container element or parent element

{ 3 / 21 }



Nothing will change after adding display: flex; in the parent container because we

need to define the width of columns. In order to set that columns width we have grid-template-columns property

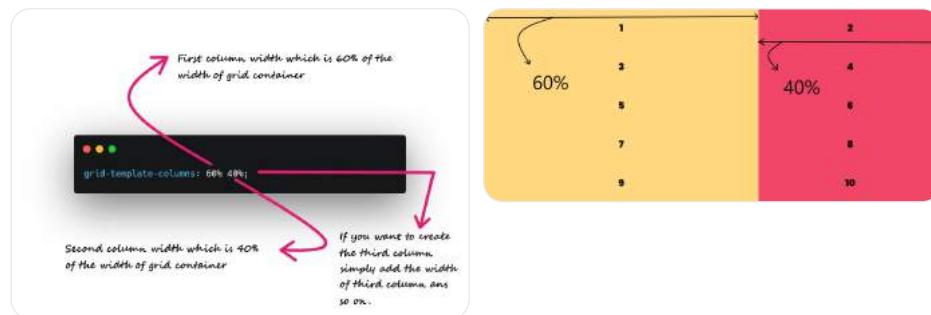
{ 4 / 21 }

Let's start with defining the width of our columns.

For example, let's say I need two columns of width 60% and 40% respectively

grid-template-columns: 60% 40%;

{ 5 / 21 }

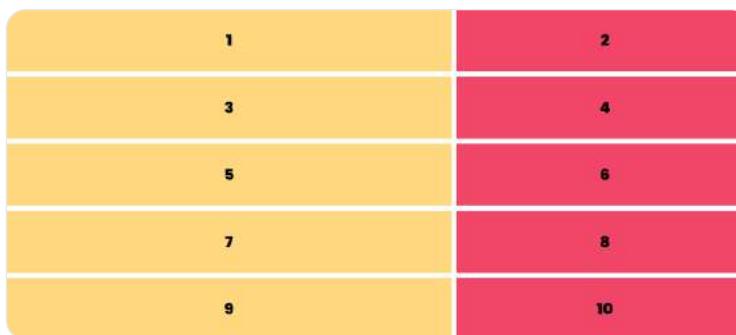


Ahh!! My grid items looks ugly as there is no spacing between them.

Here "grid-gap" property comes into play. For example I need 10px spacing along column and row

grid-gap: 10px;

{ 6 / 21 }



Similary we have grid-template-rows.

It is used to define the number of rows and height of rows.

grid-template-rows: 200px 400px;

{ 7 / 21 }

```
.container {
  grid-template-columns: 200px 200px 200px 200px 200px;
}
```

As you can see there is a lot of repeated code in
grid-template-columns: 200px 200px 200px 200px 200px;

Instead of this we can use repeat function

grid-template-columns: repeat(5, 200px);

{ 8 / 21 }

First param defines the number of repetition

Second param defines "what to repeat"

Hence it is equivalent to
grid-template-columns: 200px 200px 200px 200px 200px;

You might run into some responsiveness issues if you pass pixel unit or percentage in your grid-template-columns

In order to prevent this, it is recommended to use fraction values

For example

{ 9 / 21 }

```
grid-template-columns: 1fr 2fr 1fr;
```

You can use repeat function for fr as well

repeat(2, 1fr 2fr);

It will repeat 1fr 2fr two times.

{ 10 / 21 }

1	2	3	4
5	6	7	8
9	10		

Alright moving forward, you can set the height of grid element using grid-auto-rows

For ex, grid-auto-rows: 200px;

{ 11 / 21 }

1	2	3	4
5	6	7	8
9	10		

The height of each and every grid items is 200px because of
grid-auto-rows: 200px

Though there is a problem. By doing this, we are setting the fixed height so content inside items can be overflow.

For example ↗

{ 12 / 21 }

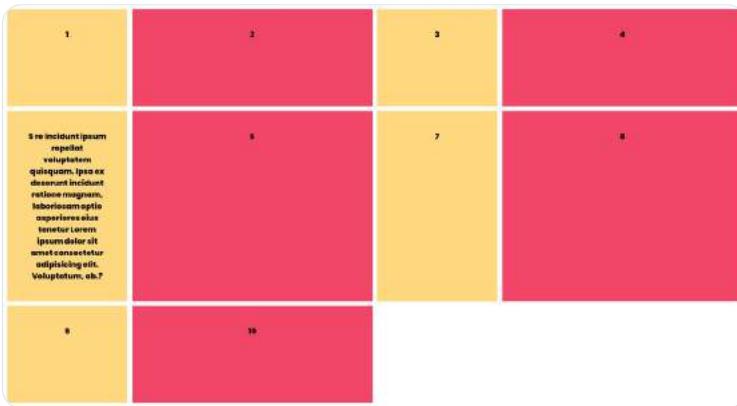
1	2	3	4
5 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Totam, accusantium, recidendi omnis.	6	7	8
9 consequatur lugit medi quis nemo non sequitur blanditiis.	10		

In order to prevent this kind of issues we have minmax function

grid-auto-rows: minmax(200px, auto);

It's pretty intuitive that the height of grid items will be 200px minimum and "auto" maximum(according to content)

{ 13 / 21 }



From 14 to 21

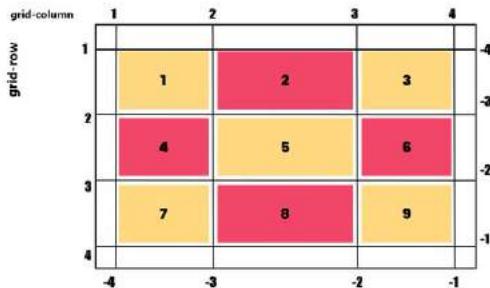
Pratham
@Prathkum



Well all that we have covered so far we can do that using flexbox also.

Let's understand the 2 dimensions of grid layout

{ 14 / 21 }



9:20 AM · Mar 24, 2021



Heart 85 Comment 4 Copy link to Tweet

• • •



Pratham @Prathkum
2 Apr · 23 tweets · [Prathkum/status/1377871173875335170](#)

Tr

A Complete Guide to Getting Started with CSS

Thread 🧶 ↴



CSS is an amazing and unique language that servers a great purpose. We can make our website visually good using CSS. It describe the presentation of web pages, including typography., layouts, color etc...

{ 2 / 23 }

We can't imagine web development without CSS. See two attached images

1. With CSS
2. Without CSS

Now imagine all websites without styling....

{ 3 / 23 }



I hope you got it why CSS is important for web development. Alright let's move onto

asctually discussion that how you start learning it

{ 4 / 23 }

First and foremost

The characterstic of a great website is it's color scheme. Forget about everything and learn about background and color properties initially.

The colors are something from which users interact first whenever they visit your webpage

{ 5 / 23 }

There are a lot of great color palette out there using which you can generate pleasant color schemes

Check this great tool for generating accessible colors

<https://color.adobe.com/create/color-accessibility>

{ 6 / 23 }



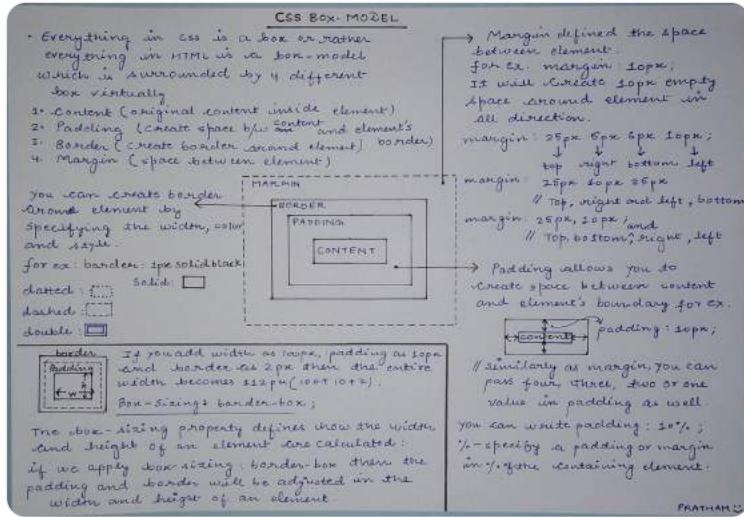
Don't think that background property is just for setting the solid color. Background is a shorthand property for background-image, background-position etc..

{ 7 / 23 }

Box model is one of the most important concept of CSS. It's not so tough to learn. The box-model covers

- Height
- Width
- Padding
- Border
- Margin

{ 8 / 23 }



Height and width property are pretty intuitive. These are used to set fixed height and width to the element

I suggest to give a look at max, min-width and max, min-height properties as well.

{ 9 / 23 }

Proper and uniform separation of elements is something that can give your webpage a appealing look. Margin and padding can do this for you.

Give this article a short read for Definitive guide of padding and margin



{ 10 / 23 }

Border are used to set the color, width and style to elements. You can learn it in 5 min 😊

Some good border selection can give your element a good pleasant look

{ 11 / 23 }

Moving forward, typography is an essential thing of web page. A good font can make your webpage and establish a strong visual hierarchy, provide a graphic balance to the website, and set the product's overall tone.

You can add free fonts from Google's official site

{ 12 / 23 }

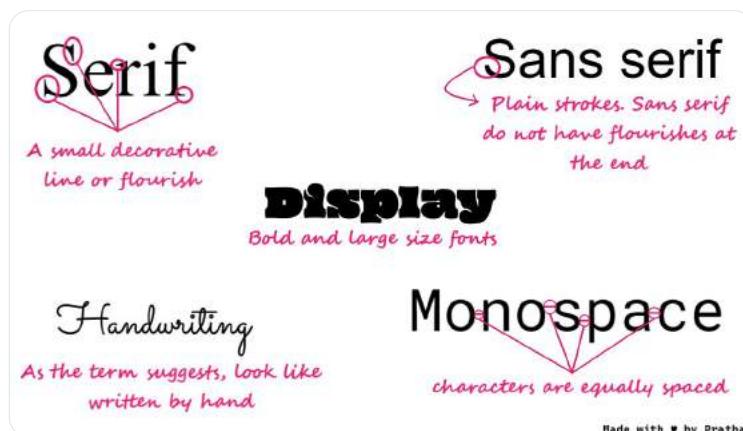
There are five basic classifications of fonts:

1. serif
2. sans serif
3. script
4. monospaced
5. display

Give this article a read for more detailed explanation

<https://www.fonts.com/content/learning/fontology/level-1/type-anatomy/type-classifications>

{ 13 / 23 }



fonts .google.com (Download free fonts from here)

You just need to look at few fonts properties. For ex,

- ◆ font-family
- ◆ font-weight
- ◆ font-size

{ 14 / 23 }

Alright moving further, We have CSS positioning.

From here, a bit tricky CSS starts. Using CSS positioning you can change the position of your element. This might seem a bit tough but you can learn it in 2-3 days

{ 15 / 23 }

I have already written a detailed thread on CSS positioning, If you're interested check it out



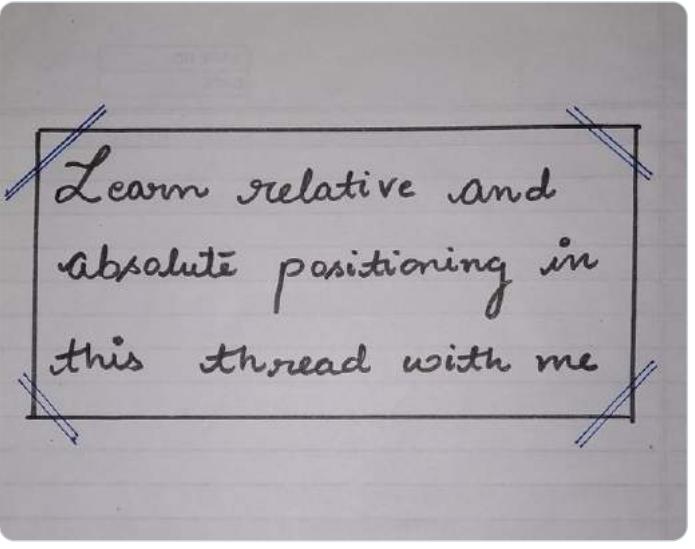
Pratham
@Prathkum

Positioning in CSS allows you to display your element wherever you want on the screen

But when I was learning it, I found it little bit confusing 😅

So in this thread I'll try to explain it in easiest manner with practical implementation. Let's start

THREAD 

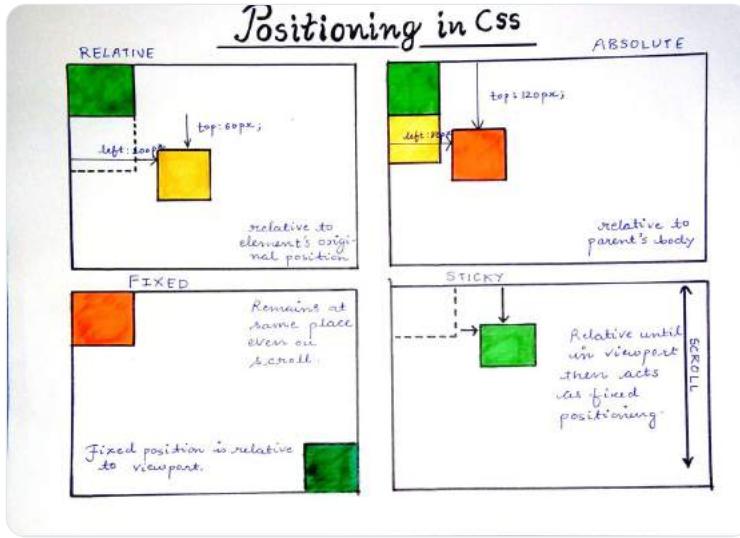


Learn relative and
absolute positioning in
this thread with me

10:48 AM · Jan 31, 2021

 923  36  Copy link to Tweet

{ 16 / 23 }



Up to this point you have some decent knowledge of styling your website. It would be great if you learn about a layout system now.

Flex and Grid

Grid is little bit tough to master but flex isn't

{ 17 / 23 }

Layouts can save your time as well. Its an important concept to learn because it plays an important role in responsive web design(RWD)

Pratham
 @Prathkum

A complete beginner's guide to CSS Grid layout

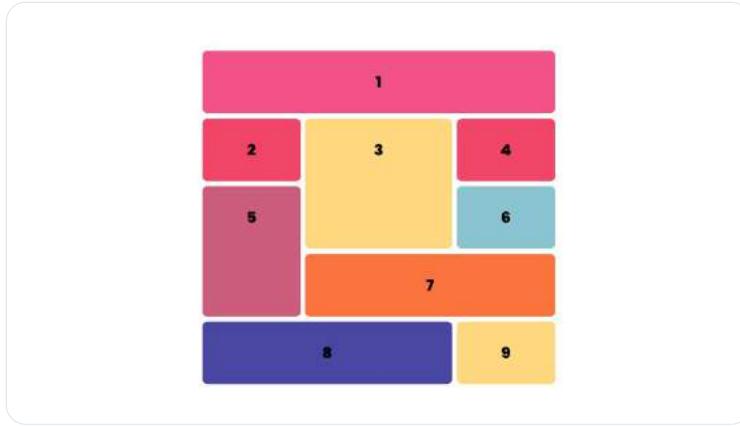
Thread

1		
2	3	4
5	7	6

9:20 AM · Mar 24, 2021

2.6K
 36
 Copy link to Tweet

{ 18 / 23 }



Hmm awesome!! I think you're all set to learn about responsiveness here.

Don't skip it. There are millions of devices on which your website is viewed.

{ 19 / 23 }

Responsive web design is little tricky but not so tough to master. There are few quick points you need to remember while writing CSS code

I would suggest you to take care of responsiveness while writing CSS rather than handling responsiveness in the end.

{ 20 / 23 }

- 1 Use meta viewport element
- 2 Don't use large fixed width
- 3 Try to use Layouts
- 4 Use box-sizing: border-box
- 5 Media Queries are saviour
- 6 Use "auto" in media
- 7 Use frameworks if possible

{ 21 / 23 }

In this thread I have compiled some general tips to make your website responsive

<https://twitter.com/Prathkum/status/1362803479044104195?s=20>

{ 22 / 23 }

Great!! I think that's pretty much it in order to give you a quick overview. If you have any doubts, feel free to drop a comment below

If you like this thread, a retweet means a lot ❤️

{ 23 / 23 }

...



Pratham 🎨🚀 @Prathkum

4 Apr · 19 tweets · [Prathkum/status/1378618845225746433](#)

Tr

Everything you need to know about CSS position property

Thread🧵👉



There are 5 values that you can pass in position property

- static
- relative
- absolute
- fixed
- sticky

In this thread we will be look at all of them

{ 2 / 19 }

Let's start with understanding what document flow is?

👉 Elements are displayed on the screen as they written in the HTML document

Consider the following piece of code:

H1, P, H3 and div are displayed on the screen in exact order as they written in the HTML file

{ 3 / 19 }



Alright let's start with position concept. The first value we have is "static"

HTML elements are positioned static by default. An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

{ 4 / 19 }

Moving forward, next we have is

📍 Relative Position

- Relative positioning do not take an element out of document flow
- Relative positioning is relative to element's original position which can be changed using offset

{ 5 / 19 }

- ◆ Relative position is relative to itself.

For example: Consider the code and output in the attached image below

As you can see red box is shifted 100px from left because I applied left offset after giving it relative positioning

{ 6 / 19 }

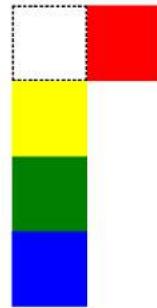


In the attached image below, the black dotted area would be the original position of

red box if I don't apply position relative in it.

As you can see it proved that relative position is relative to itself

{ 7 / 19 }

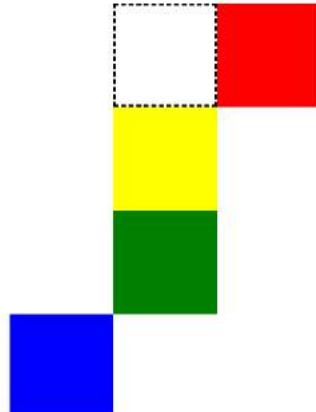


So now let me shift the blue box 100px towards left. So how can I do that? it's simple

```
.blue {  
position: relative;  
right: 100px;  
}
```

Notice here that document flow is as it is. So the relative position does not affect the document flow

{ 8 / 19 }



📌 Absolute Position

- The element is removed from the normal document flow
- You can consider it as, after applying absolute position the element will no longer in the flow and no space is created for the element in the page layout

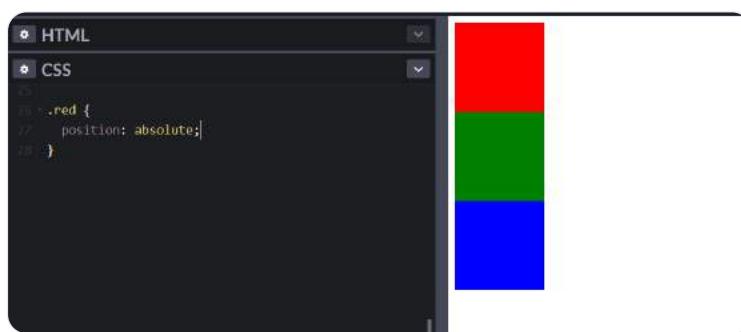
{ 9 / 19 }

For example:

If I apply absolute position in the red box, then the red box will be out of the flow and hence no space will be allocated to it.

See the image below, red box is out of flow and hence yellow box is at top and followed by green and blue

* Yellow box is below red



- The absolute position of an element is relative to its closest ancestor, which has some position property.

Consider the code below, Red is the parent div and black is the child div. In this particular case, body is the parent of red div

{ 11 / 19 }



Now let me apply relative position to red(parent) div and absolute position to black(child) div.

As I mentioned absolute position is relative to closest ancestor having some position property

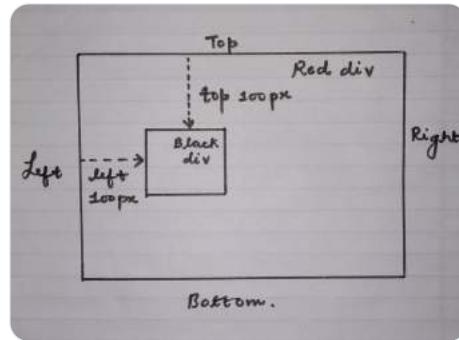
{ 12 / 19 }

The screenshot shows a code editor on the left and a browser preview on the right. The code editor contains the following CSS:

```
/* RED DIV */
.parent {
  position: relative;
}

/* BLACK DIV */
.child {
  position: absolute;
  top: 100px;
  left: 100px;
}
```

The browser preview shows a red square background with a black square centered at the coordinates (100px, 100px) relative to its parent element.



Let's understand it in little more details ↴

Consider this piece of code.

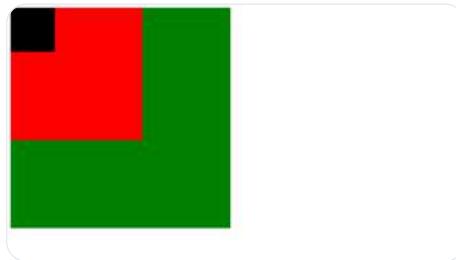
Here green div is a parent of red and red div is a parent of black

{ 13 / 19 }

The screenshot shows a code editor containing the following HTML structure:

```
<div class="first-parent">
  <div class="second-parent">
    <div class="child"></div>
  </div>
</div>
```

```
.first-parent {  
    width: 500px;  
    height: 500px;  
    background: green;  
}  
  
.second-parent {  
    height: 300px;  
    width: 300px;  
    background: red;  
}  
  
.child {  
    height: 100px;  
    width: 100px;  
    background: black;  
}
```

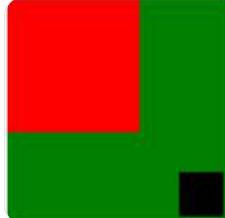


So let me apply position property in green and black. In black div we have absolute position so in that case black div will be relative to green not red

Because here black's closest ancestor is green which has some position property

{ 14 / 19 }

```
.first-parent {  
    position: relative;  
}  
  
.second-parent {  
    height: 300px;  
    width: 300px;  
    background: red;  
}  
  
.child {  
    position: absolute;  
    bottom: 10px;  
    right: 10px;  
}
```



Next we have is position: fixed;

Fixed position elements is always relative to the viewport. Which means it always stays in the same place even if the page is scrolled

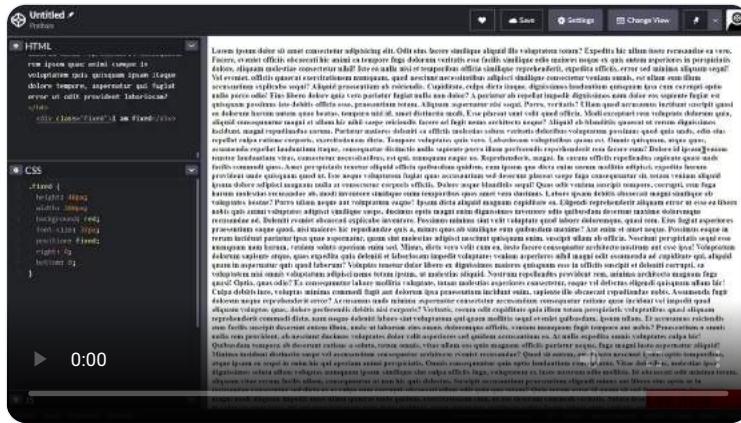
{ 15 / 19 }

For example, consider this red element. It will always fixed at same place even if the page is scrolled.

Fixed element also break the document flow.

<https://codepen.io/prathamkumar/pen/bGgWJoN>

{ 16 / 19 }



Great! Moving even further, Next we have is

📌 Position: sticky

Sticky position is the mixture of relative and fixed position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed)

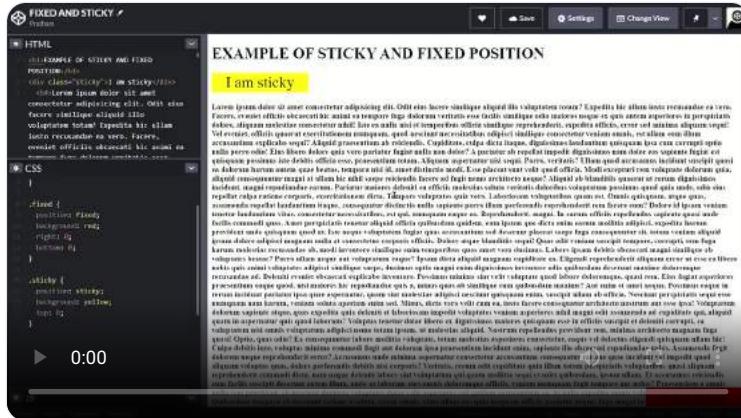
{ 17 / 19 }

The sticky state does not break the document flow.

Play around with the output here for better understanding

<https://codepen.io/prathamkumar/pen/bGgWJoN>

{ 18 / 19 }



I think that's pretty much it. Did I forget to add something? Feel free to add below.

Thanks for reading ❤️



Pratham 🎨🚀 @Prathkum

6 Apr · 36 tweets · [Prathkum/status/1379366024991158273](#)

Tr

Are you planning to learn React? If yes, these 35 tweets can make the process easier for you👉

Long Thread🧵



If you just started or planning to getting started with React, this thread might help you.

In this thread I'll try to give you a quick overview to the world of React.

So if you find that sound interesting, give this thread a read😊

{ 2 / 35 }

Before you go further into the React, make sure to check these things

- Basic knowledge of HTML and CSS
- JavaScript fundamentals and ES6 features

I would like to suggest you build a decent hold on JavaScript concepts because that is the backbone of React

{ 3 / 35 }

So what is React?

- React is a JavaScript front-end library for building user interfaces or UI components
- A typical React app contain many components. They are reusable and can interact

with each other

{ 4 / 35 }

Now the most obvious thing comes in mind is "What is component?"

📌 Component as a simple function that you can call with some input and they render some output

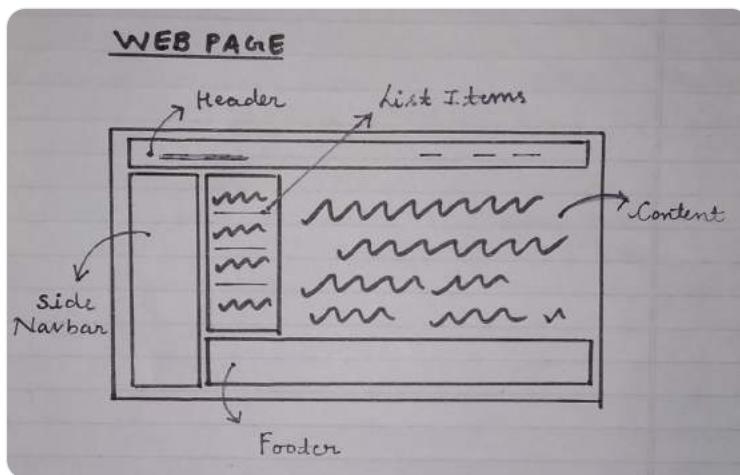
As they are reusable and interactable, so you can merge many components in order to make a entire React app

5 / 35

For example...

Attached image showing a typical React app with all different components.
As you can see this entire webpage is nothing but the mixture of different components

{ 6 / 35 }



In the above image there are 5 components

- Header
- Footer
- Content
- Sidebar
- List Items

I think you now have some understanding of the components

{ 7 / 35 }

Let's go further into the discussion

There are two types of components in React

- 1 Class based components
- 2 functional based components

Class-based components are defined using ES6 classes, whereas function components are basic JavaScript functions

{ 8 / 35 }

Before we breakdown each component, we need to understand difference between JS and JSX

Well, components are nothing but a JavaScript file but for our convenience we use JSX format of our components

{ 9 / 35 }

 JSX stands for JavaScript XML. It's basically nothing but the extension of JavaScript which allow us to write HTML code in JavaScript file.

Without JSX, creating large, nested HTML documents using JS syntax would be a tedious task that's why we use JSX

{ 10 / 35 }

```
const element = <h1>Hello, world!</h1>;
```

Consider this variable declaration. It's neither JS nor HTML. This is the mixture of JavaScript + XML = JSX

You can create JSX file with .JSX extension

{ 11 / 35 }

Now you know about JSX, let's breakdown two types of components

- 1 Class-based components

- Class-based components are defined using ES6 classes which implement a render function, which returns some JSX

A typical example of class based components 

{ 12 / 35 }

The screenshot shows a browser window with two tabs: 'ClassBasedComponent' and 'FunctionBasedComponent'. Both tabs display the same content: 'Hi, I'm a class based component' and 'Hi, I'm a class function component'. The code for each component is visible in the browser's developer tools.

```
1 import React from "react";
2
3 class ClassBasedComponent extends React.Component {
4   render() {
5     return <h1>Hi, I'm a class based component</h1>;
6   }
7 }
8
9 export default ClassBasedComponent;
```

```
1 import React from "react";
2
3 function FunctionBasedComponents() {
4   return <h1>Hi, I'm a class function component</h1>;
5 }
6
7
8 export default FunctionBasedComponents;
```

2 Functional-based components

- Functional components are nothing but simply a JavaScript function which takes some parameter will return some JSX code

If looks confusing, don't worry I'll explain each line of code further in this thread

{ 13 / 35 }

The screenshot shows a browser window with two tabs: 'ClassBasedComponent' and 'FunctionBasedComponent'. Both tabs display the same content: 'Hi, I'm a class based component' and 'Hi, I'm a class function component'. The code for each component is visible in the browser's developer tools.

```
1 import React from "react";
2
3 class ClassBasedComponent extends React.Component {
4   render() {
5     return <h1>Hi, I'm a class based component</h1>;
6   }
7 }
8
9 export default ClassBasedComponent;
```

```
1 import React from "react";
2
3 function FunctionBasedComponents() {
4   return <h1>Hi, I'm a class function component</h1>;
5 }
6
7
8 export default FunctionBasedComponents;
```

Components can be further divided into two categories

- ◆ Stateful components
- ◆ Stateless components

As the term suggest, one has state, and the other doesn't

{ 14 / 35 }

The Heart of every React component is "State" which is nothing the the object of observable properties. In simple terms we can say that state is nothing but the information/data that associated with a particular component

When the state object changes, the component re-renders

UNDERSTANDING THE VIRTUAL DOM

- You might have heard the term "DOM", virtual DOM is kind of similar. It uses a strategy that updates the DOM without having to redraw all the webpage elements

{ 16 / 35 }

Every time the DOM changes, browser need to recalculate entire layout and then repaint the web page which makes a web app slow.

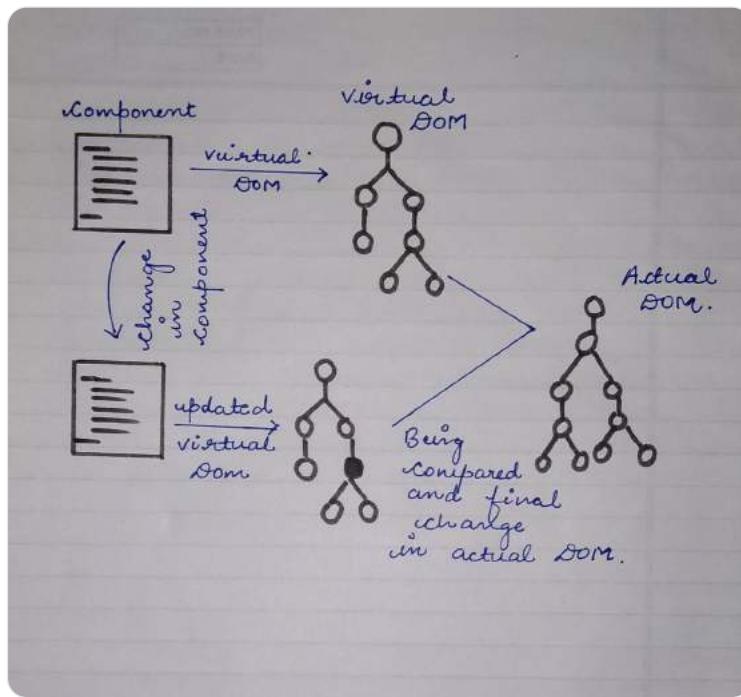
To overcome this we have a virtual DOM

- 📌 Every time the state of our application changes, the virtual DOM gets updated instead of the real DOM

17

Whenever the new element is added to the UI, a new virtual DOM associated with that element is created. If state of this element changes, a second new virtual DOM is created which will be compared with the previous virtual DOM

- It then updates ONLY the object on the real DOM



Sounds confusing??

[@moshhamedani](#) has an amazing article on this. Give it a read

<https://programmingwithmosh.com/javascript/stateful-stateless-components-react/>

{ 19 / 35 }

Let's see how we can create a React app in local machine?

- I'm assuming you have node environment set up and up-to-date version of npm. If

no, download it from here



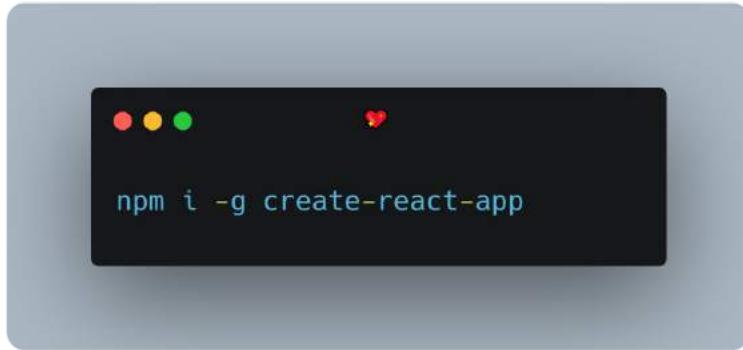
{ 20 / 35 }

Now we're good to go to create our first React app

- We are using Create React App, a tool that gives you a massive head start when building React apps. It saves you from time-consuming setup and configuration. You simply run one command

Install it first👉

{ 21 / 35 }

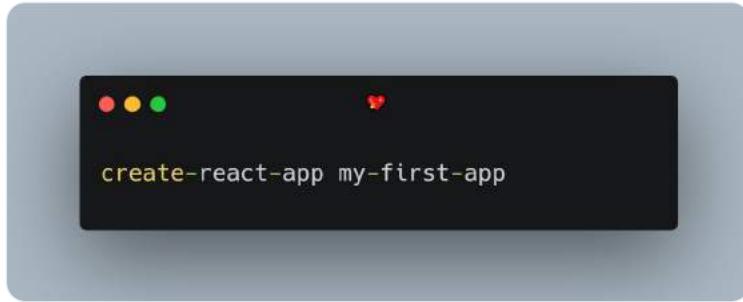


Now you have create-react-app installed in your machine, it's time to create your first React app

Command - "create-react-app app-name"

Depending upon your internet speed, this will take some minutes. So time to prepare a coffee for yourself☕

{ 22 / 35 }



Great!! 😊

After successful installation, your terminal will show you something like this

- Open this folder in your favorite editor and then we will write our first React code🌈

{ 23 / 35 }

```
f create-react-app my-first-app
  Creating a new React app in C:\Users\prash\Desktop\Web Development\React\Projects\my-first-app.

  Installing packages. This might take a couple of minutes.
  Installing react, react-dom, and react-scripts with cra-template...

> core-js@2.6.12 postinstall C:\Users\prash\Desktop\Web Development\React\Projects\my-first-app\node_modules\babel-runtime\node_modules\core-js
  core-js
> node -e "try{require('./postinstall')}catch(e){}"

> core-js-pure@2.0.3 postinstall C:\Users\prash\Desktop\Web Development\React\Projects\my-first-app\node_modules\core-js-pure
  core-js-pure
> node -e "try{require('./postinstall')}catch(e){}"

> ejar@2.7.4 postinstall C:\Users\prash\Desktop\Web Development\React\Projects\my-first-app\node_modules\ejar
  ejar
> node ./postinstall.js

+ cra-template@1.1.1
+ react-scripts@4.0.1
+ react@17.0.1
+ react-dom@17.0.1
added 1902 packages from 722 contributors and audited 1905 packages in
552.382s

120 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

The terminal shows the command 'create-react-app my-first-app' being run. It installs react, react-dom, and react-scripts with cra-template. It also installs core-js, core-js-pure, and ejar. The output includes version numbers and audit results.

Run this command to start your local server and run react app

" npm start "

Your default browser will launch automatically and you will see something like this at localhost:3000

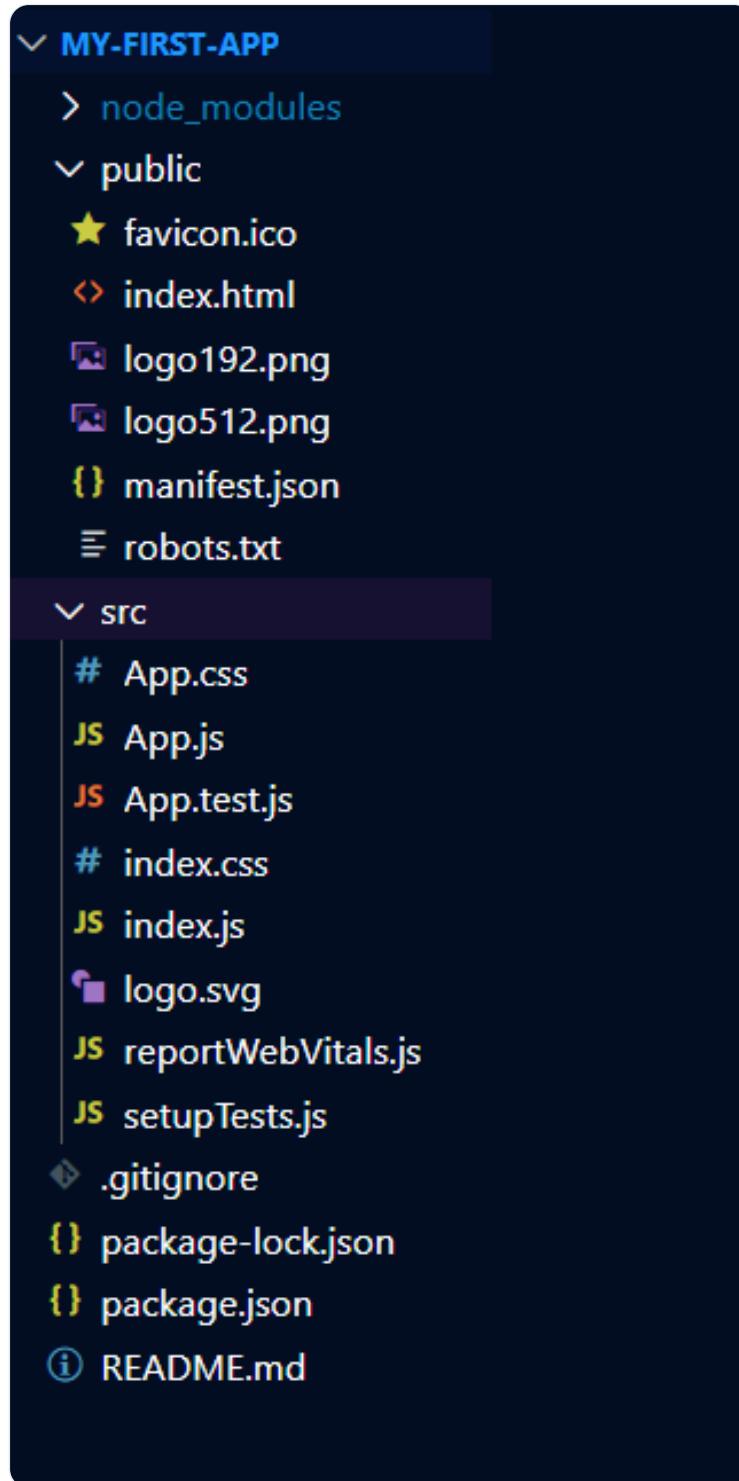
{ 24 / 35 }



You'll get many folder and files in your app. We're not going in the details as of now because that can create some confusion. You will understand each and everything automatically as you go further in the world of React

So let's start with the src folder

{ 25 / 35 }



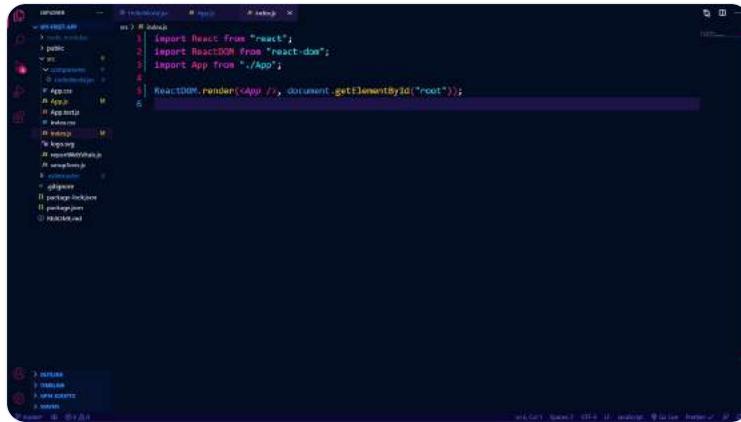
...

Inside the src folder you'll get index.js file

- Remove everything inside the file
- And add this four line of code

Let me try to explain each line of code of this file  (next tweet)

{ 26 / 35 }

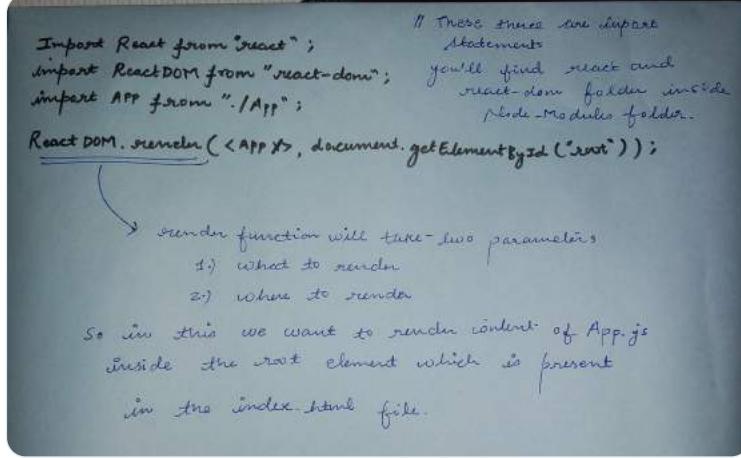


```
index.js
1 import React from "react";
2 import ReactDOM from "react-dom";
3 import App from "./App";
4
5 ReactDOM.render(, document.getElementById("root"));
```

- By importing React, you are telling your browser you're working with JSX so that your browser can compile it

- Import ReactDOM in order to user render method
- Import App.js so that we can print return JSX of it

- Finally render method takes two param(See attached image)



You can directly render things from inside the index.js file

* this is just from explaining purpose, don't do this in real projects

{ 28 / 35 }

The screenshot shows a code editor with an open file named index.js. The code is as follows:

```

1 import React from "react";
2 import ReactDOM from "react-dom";
3
4 ReactDOM.render(
5   <h1>Coding directly from index.js</h1>,
6   document.getElementById("root")
7 )
8

```

Next to it is a browser window displaying the rendered output: "Coding directly from index.js".

- Create a folder "Components" in the src folder

This is the best practice in React, suppose you're building a complex app that have more than 10 different components, then it's good to have component folder in order to manage each and every component effectively

{ 29 / 35 }

- Inside that component folder create file(component), HelloWorld.jsx

I am gonna use functional component. You can use class component as well but functional components are new way to making your components hence I'll recommend you to write functional component

{ 30 / 35 }



- Import React so that our browser can understand we are working with JSX

- Create a function with same name as of component i.e, HelloWorld(it is best practice)

- Write your code inside HelloWorld function

CONT...

{ 31 / 35 }

- I just wanna return "hello world"

- export HelloWorld so that we can use it outside the component

{ 32 / 35 }

```
src -> components -> HelloWorld.js (in edit)
1 import React from "react";
2
3 function HelloWorld() {
4   return <h1>Hello World</h1>;
5 }
6
7 export default HelloWorld;
```

Import HelloWorld inside App and return it

- You will see the "Hello world" is being printed on your web page😍

Congratulations🎉 you have just taken your first step into the world of React

{ 33 / 35 }



I think time to wrap up this thread. React is so deep, it can't be explained in a single thread

But I tried my best to give a quick overview of how things work in React

I hope you get a basic overview of what components are, JSX, virtual DOM and stuff like that

{ 34 / 35 }

Next Step?

- Props and hooks.

I'll try to write a thread on them. Thanks for reading this ❤️

This is the longest thread I've ever written.

If you are a beginner, my tweets can boost your learning process 🚀



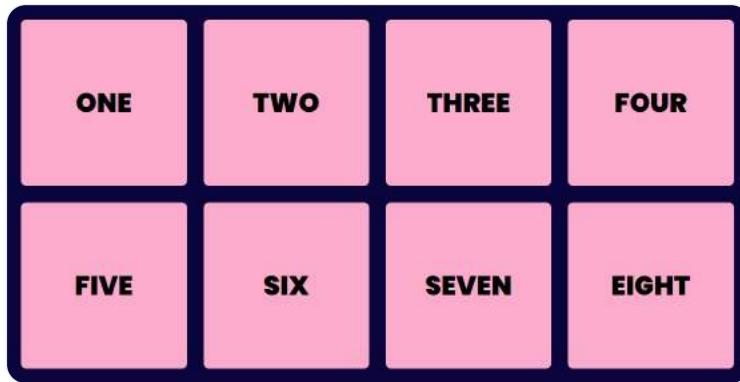
Pratham @Prathkum

7 Apr · 16 tweets · [Prathkum/status/1379881164912099334](#)

Tr

Everything you need to know about CSS Flex layout

A Thread



Start applying display: flex; property to parent element

This will take all the elements in horizontal direction

{ 2 / 16 }

The image consists of two cards. The left card shows a dark blue rectangle containing eight small white boxes labeled 'ONE' through 'EIGHT', arranged horizontally. The right card has a purple-to-blue gradient background and displays a snippet of CSS code:

```
.parent {  
  display: flex;  
}
```

You can also change the direction of elements. For doing so, you need to mention the "flex-direction" property

The flex-direction property specifies the direction of the items within the flex container

{ 3 / 16 }

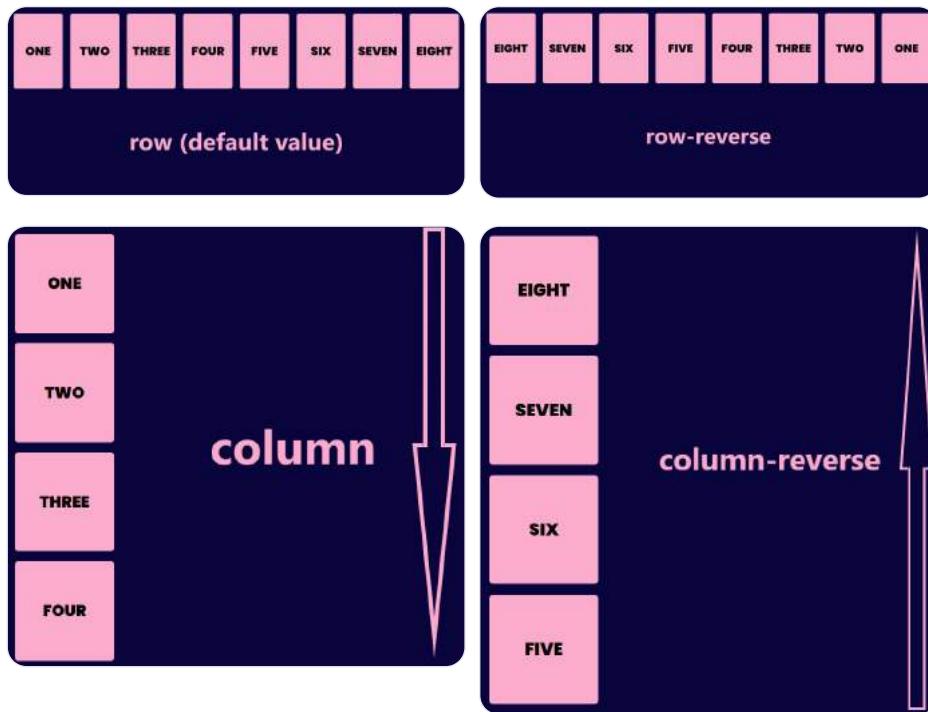
The image shows a card with a purple-to-blue gradient background. It displays a snippet of CSS code:

```
.parent {  
  flex-direction: row | row-reverse | column | column-reverse;  
}
```

After applying row, row-reverse, column and column-reverse in flex-direction

They all are pretty intuitive.

{ 4 / 16 }



If we have a large number of element in flex container, it can destroy the width of elements

As you can see our elements have shrunk a bit. This is because we haven't applied the flex-wrap property.

{ 5 / 16 }



📌 flex-wrap

It defines whether the flex items are forced in a single line or can be flowed into multiple lines depending on screen size. It preserves the width of elements

```
.parent {  
flex-wrap: wrap;  
}
```

{ 6 / 16 }

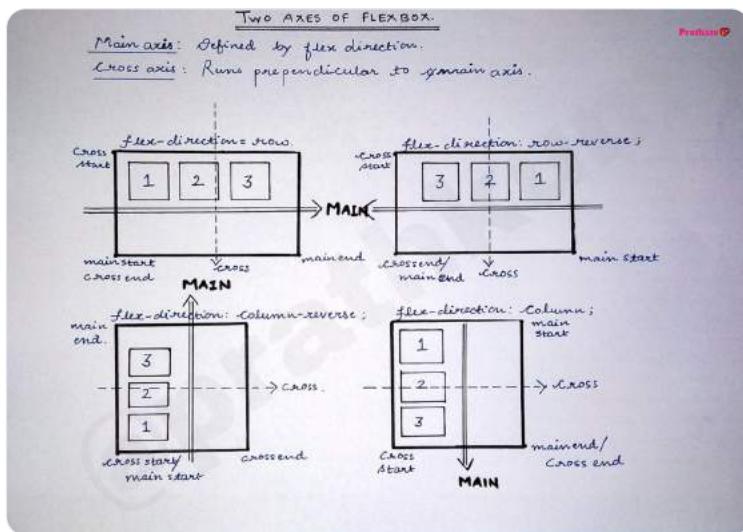


Moving forward, I think this is right time to understand what main and cross axes are



We can align elements within Flexbox in accordance with these two axis.

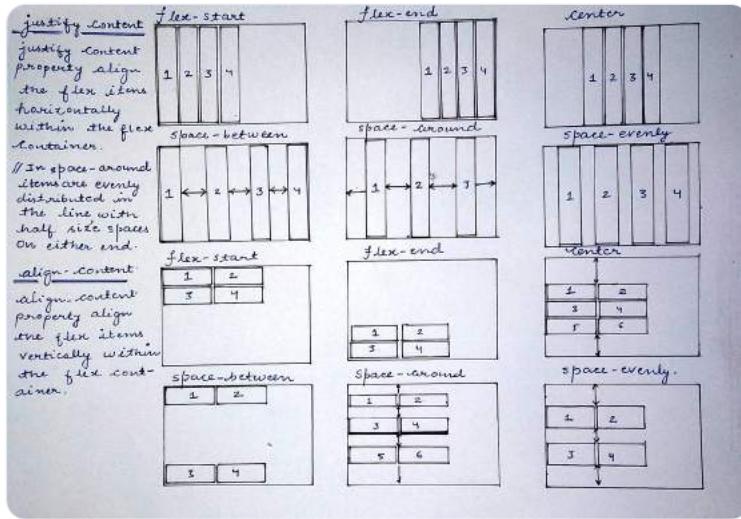
{ 7 / 16 }



Time to align flexible items within flex container

- ◆ Justify-content: For horizontal alignment
- ◆ Align-content: For vertical alignment

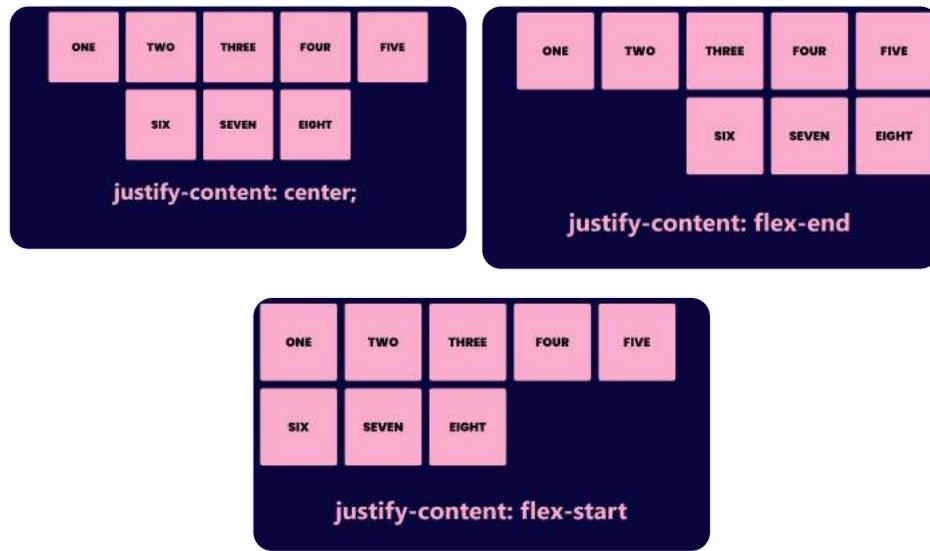
{ 8 / 16 }



The justify-content property aligns the items within flex container.

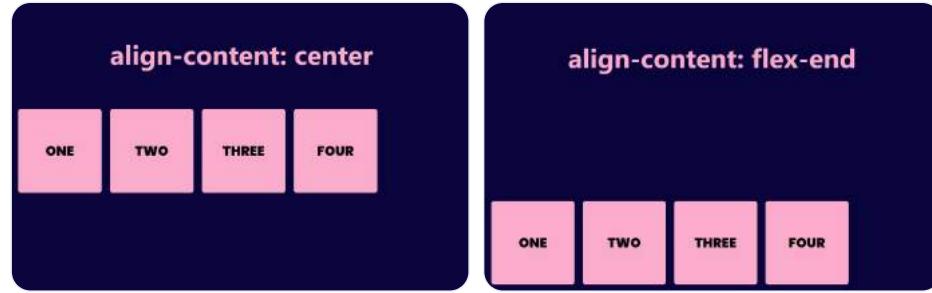
- flex-start
- flex-end
- center
- space-between (Items will equal space between them)
- space-around (equal space b/w elements with half left side and half right side)
- space-evenly

{ 9 / 16 }



align-content for vertical alignment. For example

{ 10 / 16 }

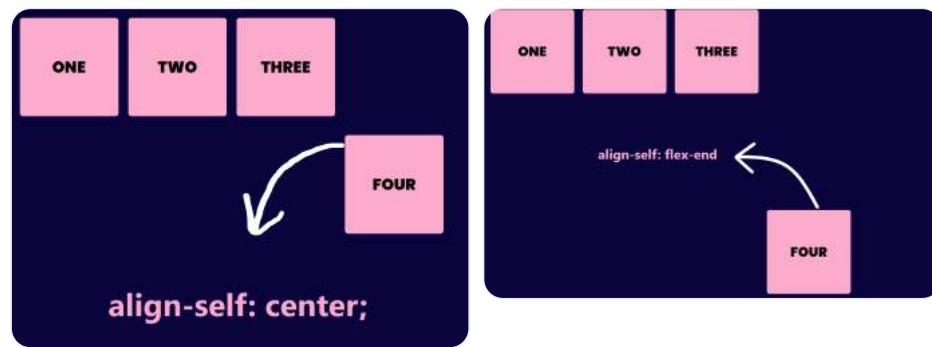


You can also position a particular item inside flex container.

For this we have, align-self property

For example:

{ 11 / 16 }



You can pass following values in align-self

- auto
- stretch
- center
- flex-start
- flex-end
- baseline
- initial

- inherit

Play with code here: https://www.w3schools.com/cssref/playit.asp?filename=playcss_align-self&preval=auto

{ 12 / 16 }

Alright! Moving even further next we have is "order"

If you want to change the ordering of the items within flex container. You can use order property that specifies the order of an item relative to the rest of the items inside the same flex container.

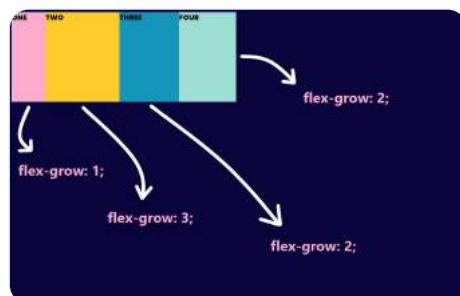
{ 13 / 16 }

```
.one {  
  order: 2;  
}  
  
.two {  
  order: 3;  
}  
  
.four {  
  order: 3;  
}  
  
.three {  
  order: 1;  
}
```

📌 flex-grow

The flex-grow property specifies how much the item will grow relative to the rest of the flexible items inside the same container.

{ 14 / 16 }



```
.one {  
  flex-grow: 1;  
}  
  
.two {  
  flex-grow: 3;  
}  
  
.three {  
  flex-grow: 2;  
}  
  
.four {  
  flex-grow: 2;  
}
```

Similarly we have "flex-shrink" property

The flex-shrink property specifies how the item will shrink relative to the rest of the flexible items inside the same container.

{ 15 / 16 }

I think that's pretty much it for this thread. Did I miss some points? Add below.

Thanks for reading this ❤

{ 16 / 16 }

...



Pratham @Prathkum

8 Apr · 31 tweets · [Prathkum/status/1380200517184458752](https://twitter.com/Prathkum/status/1380200517184458752)

Tr

These 30 free CSS generators can help you immensely and save you a lot of time

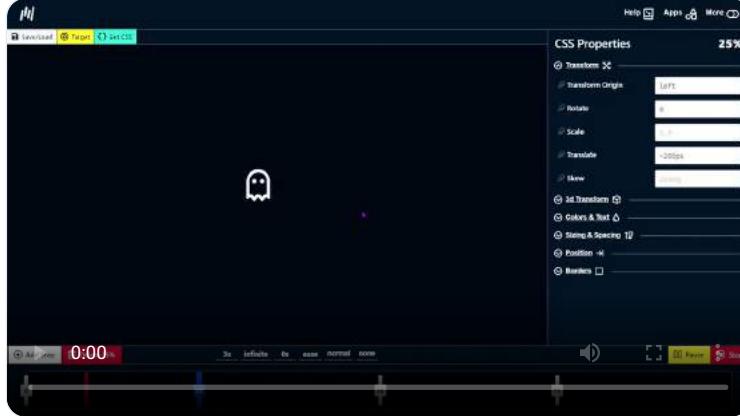
Mega Thread

1 Animation Generator

- Dead simple visual tools to help you generate CSS for your projects.



The screenshot shows the Keyframes.app interface. At the top, there's a logo with three vertical bars and the word "keyframes". Below it is a timeline with several keyframes represented by vertical bars. A small ghost icon is positioned above the timeline. The text "A simple CSS toolbox for generating animations, shadows, colors, & more." is displayed. At the bottom, there's a section titled "Keyframes.app | CSS Toolbox" with a brief description and a link: <https://keyframes.app/>.



2 Palette Ninja

- Palette ninja is an online color scheme generator that allows you to create harmonious color schemes in seconds.



The screenshot shows the homepage of palette.ninja. At the top is the logo "Palette Ninja" with a stylized ninja head icon. Below the logo is the tagline "Create Harmonious Color Schemes". A horizontal color palette is displayed, consisting of five vertical bars in blue, green, yellow, red, and purple. Below the palette, the URL "palette.ninja" is shown, followed by a brief description: "palette.ninja allows you to create harmonious color schemes." and the link "https://palette.ninja/".



3 Layout Generator

- Quickly design web layouts, and get HTML and CSS code. Learn CSS Grid visually and build web layouts with our interactive CSS Grid Generator.



The screenshot shows the Interactive CSS Grid Generator. On the left, there's a "Grid Editor" window titled "Explicit Grid" which contains two "grid-template-columns" sections, each with three columns of 1fr, 1fr, and 1fr. Below it is another "grid-template-columns" section with three columns of 1fr, 2fr, and 1fr. On the right, there's a "Code Editor" window titled "Create CodeSandBox" showing the generated CSS code for a grid layout with three columns of varying widths. The center of the screen displays a wireframe grid with three columns. At the bottom, there's a footer with the text "Interactive CSS Grid Generator | Layoutit Grid" and the URL "http://grid.layoutit.com".



4 Pattern generator

- It lets you create background pattern for free

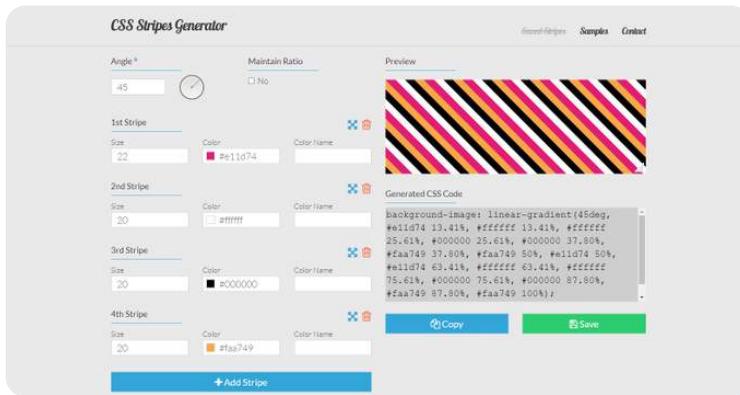
patternify.com



5 Stripes generator

- Pure CSS Stripes Generator that you can use for backgrounds.

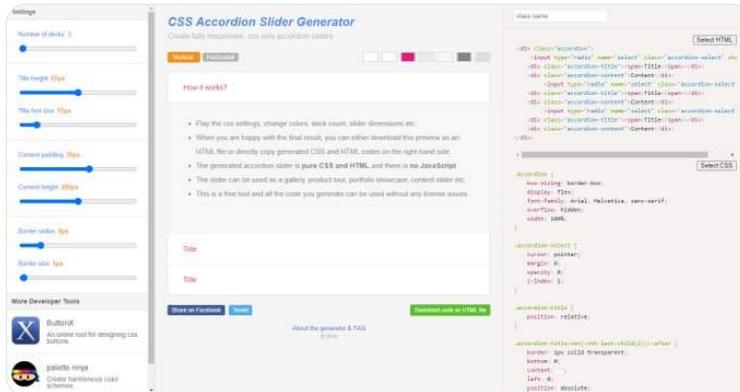
stripesgenerator.com



6 CSS Accordion Slider Generator

- Create fully responsive, css only accordion sliders

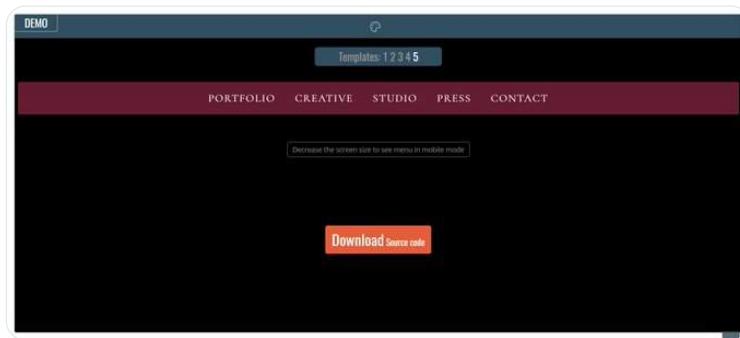
accordionslider.com



7 Navbar Generator

- You can generate 5 types of nav bar using few clicks, which are fully responsive

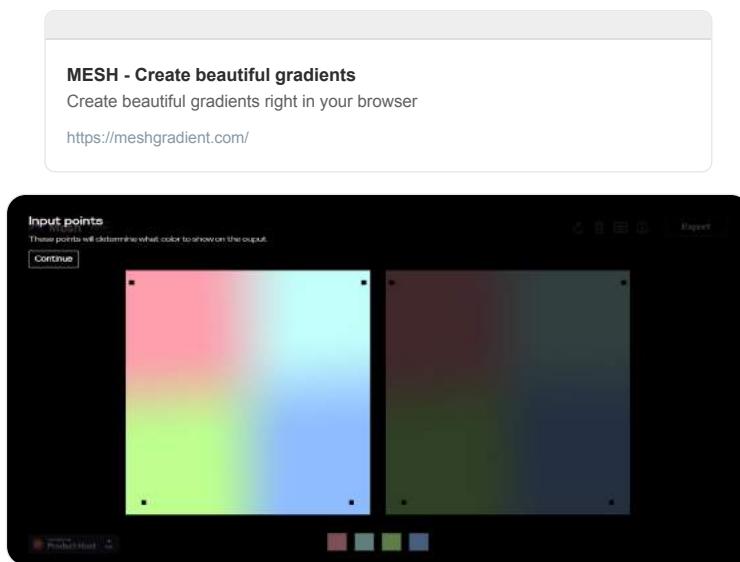
menucool.com/css-menu



8 Mesh Gradient

- Mesh is a simple way to create beautiful, unique gradients using shaders

<https://meshgradient.com/>



9 Flexbox

- Generate flexbox without actually writing the CSS



webflow

Meet the first visual flexbox builder

Easily build flexible, responsive layouts—without writing code.

Visual CSS flexbox builder | Webflow

Our CSS flexbox builder today lets you easily build websites with flexbox layouts—without actually writing the CSS. Try the Webflow flexbox generator now.

<https://flexbox.webflow.com/>



10 Shapes maker

- Create complex shapes using CSS clip-path property



bennettfeely.com/clippy/

11 CSS Box Shadow Generator

- Generate CSS3 Box Shadow code for your Div, Frame, Buttons or any other HTML element with Outline, and Inset (inner) type shadow effects

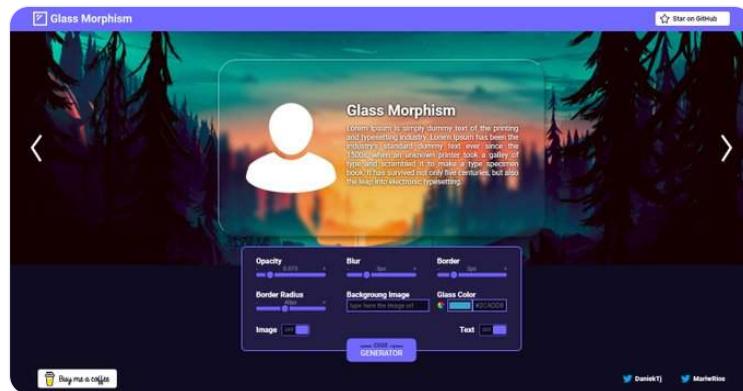
🔗 cssboxshadow.com



1 2 Glassmorphism

- Generate glassmorphic design easily

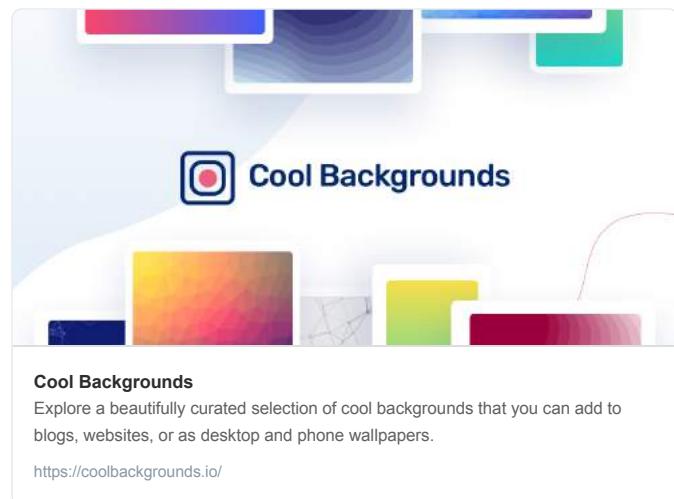
🔗 glassgenerator.netlify.app

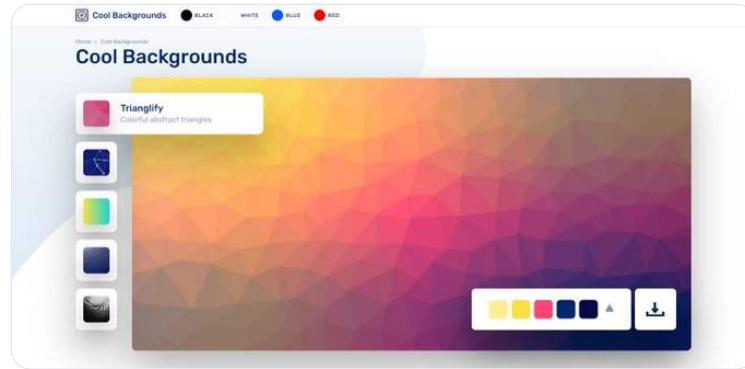


1 3 Cool Backgrounds

- Explore a beautifully curated selection of cool backgrounds that you can add to your next project

🔗



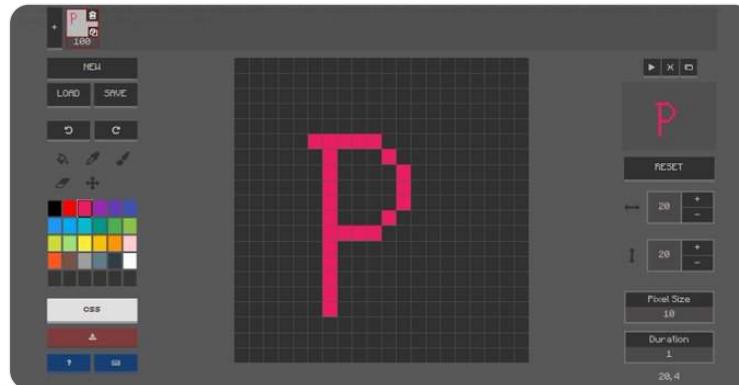


1 4 Pixel art

- Create CSS pixel art, export the results to CSS and download them.

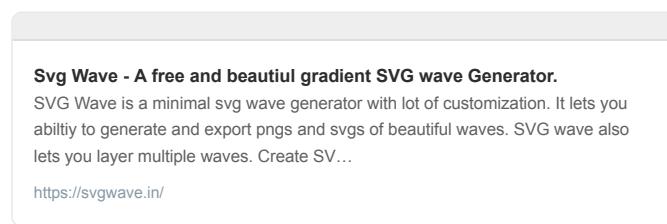


The screenshot shows the homepage of the Pixel Art to CSS website. It features a central grid where a pink letter 'P' is being drawn. To the left is a toolbar with various drawing tools and a color palette. To the right, there are controls for scaling the image up and down, and settings for 'Pixel Size' (set to 10) and 'Duration' (set to 1). Below the grid, there's a URL: <http://pixelartcss.com>.



1 5 SVG waves

- It's not a CSS generator but very handy tool. You can create the layers of waves and simply export either in SVG or PNG



The screenshot shows the homepage of the Svg Wave website. It features a central text area with the title 'Svg Wave - A free and beautiful gradient SVG wave Generator.' Below the title, there's a brief description: 'SVG Wave is a minimal svg wave generator with lot of customization. It lets you ability to generate and export pngs and svgs of beautiful waves. SVG wave also lets you layer multiple waves. Create SV...'. At the bottom, there's a URL: <https://svgwave.in/>.



16 Color theme generator

- A great tool for CSS dark/light theme



CSS Theme Generator v1.0.0

open source • light/dark scheme • normal/high contrast
Based on NumiDesign theme generator by @steveo and hsluv color space by @devesh

Colors Properties

Single tone • Duo tone

Main Hue: - 242 + Insert
Accent Hue: - 0 + Insert
Saturation: - 75 +

Type: Main Tint Tone Swap Special
Contrast: Soft Normal Strong
Emphasizing: Dim Normal Bold Only for Tone and Swap types.

Preview

Numi.Design Open REPL
Paragraph text Special paragraph text
Basic Card Clear Card
Basic badge Special badge
Basic button states Default Hover Pressed
Toggled Focused
Disabled
Special button states
Default Hover Pressed
Toggled Focused
Disabled

CSS Color Theme Generator by Numi.Design
Open Source CSS Color Theme Generator for creating lovely toned themes with dark scheme and high contrast mode.
<https://numi.design/theme-generator>

CSS Theme Generator v1.0.0

open source • light/dark scheme • normal/high contrast
Based on NumiDesign theme generator by @steveo and hsluv color space by @devesh

Options

Single tone • Duo tone

Hue: - 242 + Insert
Saturation: - 75 +
Use pastel palette:

Type: Main Tint Tone Swap Special
Contrast: Soft Normal Strong
Emphasizing: Dim Normal Bold Only for Tone and Swap types.

Output

CSS Colors Numi
Switch by CSS JS CSS JS CSS JS
The Switch by CSS declaration uses CSS Media Queries to adapt the theme to system preferences.
Note: CSS Media Query prefers-contrast is an experimental feature and currently not supported by all browsers.
See compatibility notes at caniuse.com prefers-contrast prefers-color-scheme

Preview

Numi.Design Open REPL
Paragraph text Special paragraph text
Basic Card Clear Card
Basic badge Special badge
Basic button states Default Hover Pressed
Toggled Focused
Disabled
Special button states Default Hover Pressed
Toggled Focused
Disabled
Various widgets 1 2 3 4

17 Neumorphic Design

- CSS code generator that will help with colors, gradients and shadows to adapt this new design trend or discover its possibilities.



Neumorphism/Soft UI CSS shadow generator
CSS code generator that will help with colors, gradients and shadows to adapt this new design trend or discover its possibilities.
<https://neumorphism.io/>

This project is now [Open-Source!](#) | [View on GitHub](#)

18 Smooth Shadow

- Another shadow generator with great UI

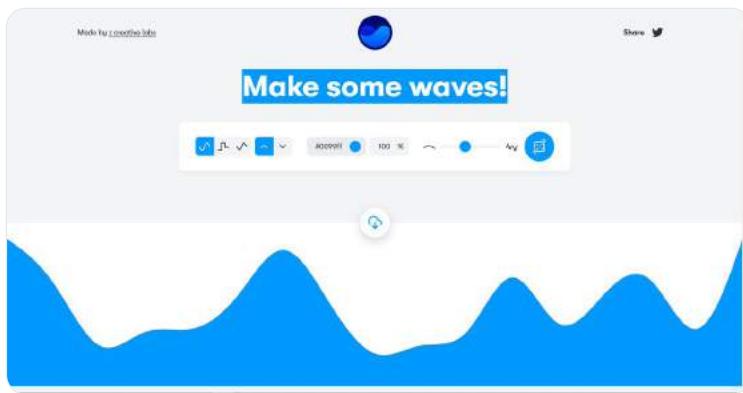
🔗 shadows.brumm.af

19 SVG waves

- Make some waves!

🔗

The screenshot shows a web application titled "Get Waves – Create SVG waves for your next design". The main heading is "Make some waves!". Below it is a control panel with various icons and sliders. A preview area shows a single blue wave. The text below the heading reads: "A free SVG wave generator to make unique SVG waves for your next web design. Choose a curve, adjust complexity, randomize!" followed by the URL "https://getwaves.io/".



20 CSS Button Generator

- You don't have to learn any complex CSS rules . Just click and slide to make CSS3 Buttons. Lots of pretty button samples.



The screenshot shows a web application titled "CSS Button Generator". The main heading is "CSS Button Generator". Below it is a brief description: "You don't have to learn any complex CSS rules. Just click and slide to make CSS3 Buttons. Lots of pretty button samples." followed by the URL "https://www.cssgenerators.net/".



2 1 Blob Maker

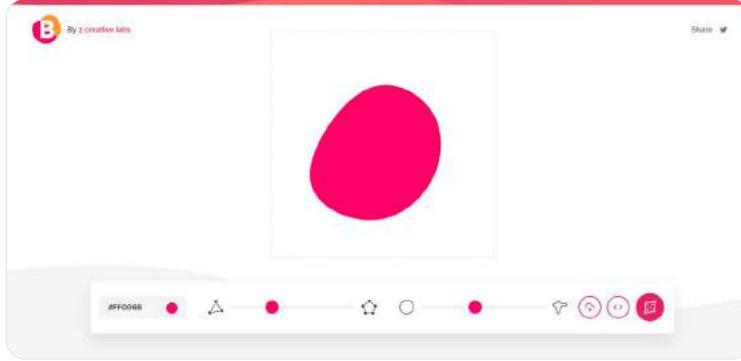
- Help you quickly create random, unique, and organic-looking SVG shapes.



Blobmaker - Make organic SVG shapes for your next design

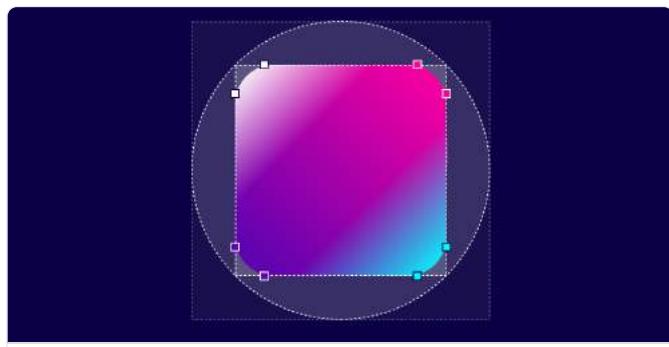
Make organic SVG shapes for your next design. Modify the complexity, contrast, and color, to generate unique SVG blobs every time.

<https://www.blobmaker.app/>



2 2 Border-radius

- Create fancy shapes using border-radius CSS property



Border Radius Generator - Full Control

A visual generator to build organic looking shapes with the help of CSS3 border-radius property

<https://9elements.github.io/fancy-border-radius/full-control.html>



2 3 Gradient generator

- As a free css gradient generator tool, this website lets you create a colorful gradient background for your website, blog, or social media profile



CSS Gradient — Generator, Maker, and Background
As a free css gradient generator tool, this website lets you create a colorful gradient background for your website, blog, or social media profile.

<https://cssgradient.io/>



2 4 Best css button generator

- CSS Button Generator is a free online tool that allows you to create cross browser HTML and CSS button styles.



**UI DESIGNER'S
BUTTON
GENERATOR**

A useful tool for designing css buttons
Button generator is a free online tool that allows you to create cross browser css button styles.

<https://www.bestcssbuttongenerator.com/>

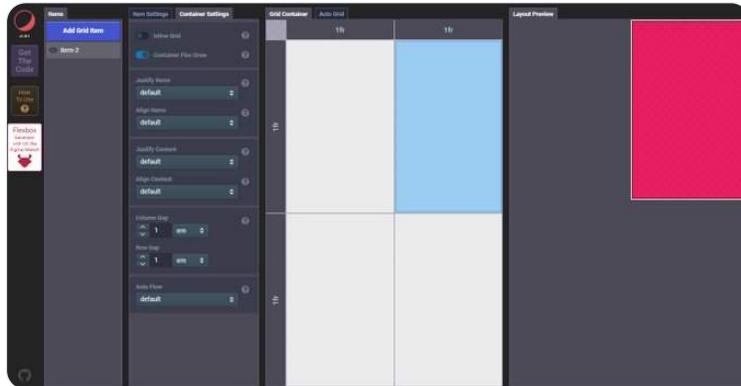


2 5 Grid Layout generator

- The Quickest & Easiest Way To Build Complex CSS Grid Layouts.



The Quickest & Easiest Way To Build Complex CSS Grid Layouts
The Quickest & Easiest Way To Build Complex CSS Grid Layouts
<https://css-grid-layout-generator.pw/>

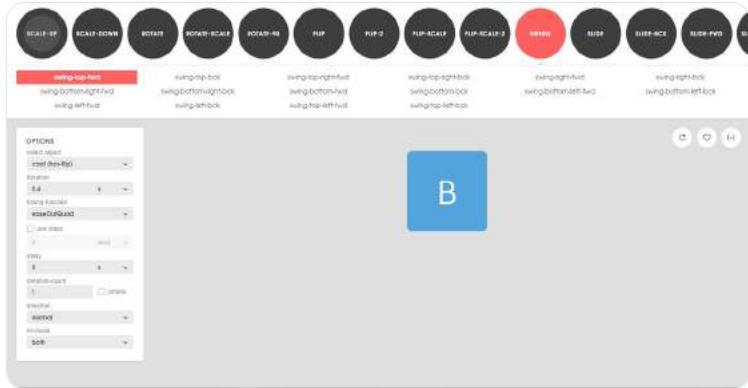


2 6 Animista

- Animista is a place where you can play with a collection of ready to use CSS animations, tweak them and download only those you will actually use.



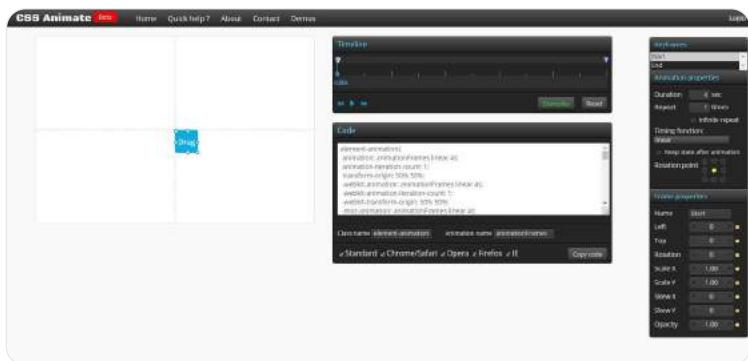
Animista - CSS Animations on Demand
Animista is a CSS animation library and a place where you can play with a collection of ready-made CSS animations and download only those you will use.
<https://animista.net/>



2 7 CSS animate

- Online tool for creating native CSS3 Keyframes Animation. You can easy and fast generate consistent CSS3 animation using simple UI without any coding.

🔗 cssanimate.com



2 8 Glassmorphism

- Glassmorphism is a unified name for the popular Frosted Glass aesthetic.

🔗

OFFICIAL

Glassmorphism

CSS GENERATOR

Glassmorphism - simple CSS generator

Glassmorphism is a unified name for the popular Frosted Glass aesthetic.

<https://glassmorphism.com/>



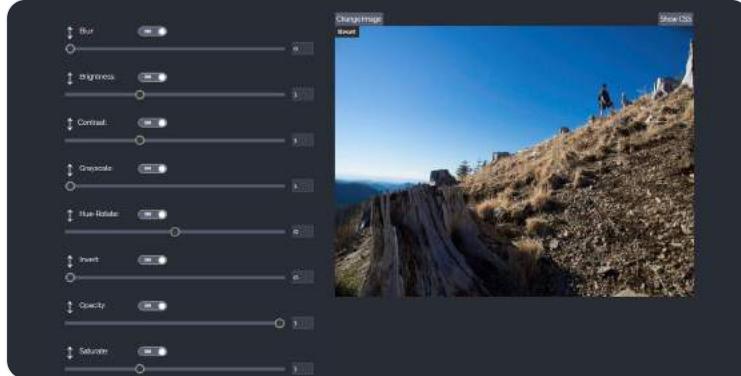
2 9 Image Filter

- Create a sharable URL of your custom filter using Create URL or Save your filter



CSSFilterGenerator.com
Beta V0.7 - prerelease - working but not beautiful - more to come!

CSSfiltergenerator.com
CSS Filter Generator produces CSS filters and mix-blend-mode overlay code using a graphical user interface.
<https://www.cssfiltergenerator.com/>

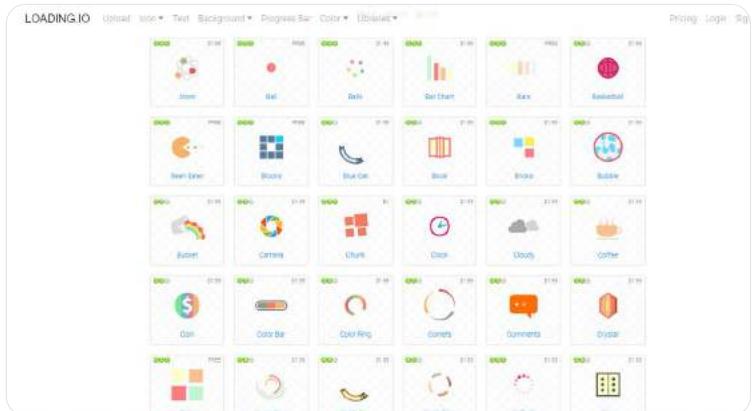


3 0 Loader

- Free CSS loaders



loading.io



• • •



Pratham @Prathkum

10 Apr · 17 tweets · [Prathkum/status/1380834198039171073](#)

Tr

Complete guide to Responsive Web Design



In order to make a RWD, you just need to consider one thing.

"Ability of content to fit inside any device that look good and it will be for user to interact with that"

{ 2 / 16 }

Responsive web design is not a kind of program or framework. We can say it's a combination of various concepts using which we try to make our web page look good on all devices

The great thing is that you can achieve RWD using HTML and CSS only ⚡

{ 3 / 16 }

1 First and foremost thing in order to make RWD is <meta> viewport element.

It forces page to follow the screen-width of the device.

{ 4 / 16 }

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

- 2 The second important point to note is that don't use large fixed width or height element.

It can cause trouble. Let's say an element having width 500px but user is viewing on a device having width smaller than 500px

In such cases, use min-width or max-width

{ 5 / 16 }

- 3 Use HTML <picture> tag

The HTML <picture> element allows you to define different images for different browser window sizes.

{ 6 / 16 }

```
<picture>
  <source media="(min-width:650px)" srcset="/path">
  <source media="(min-width:465px)" srcset="/path">
  
</picture>
```

- 4 Responsive text size

Although you can make text responsive using media queries but you can also use "viewpoet" width as well.

```
h1 {
  font-size: 10vw;
}
```

{ 7 / 16 }

- 5 Try to use Layouts

Using Grid or Flex layout always beneficial in order to make a web page responsive. Both these layout are not hard to learn. Try to use them 😊

{ 8 / 16 }

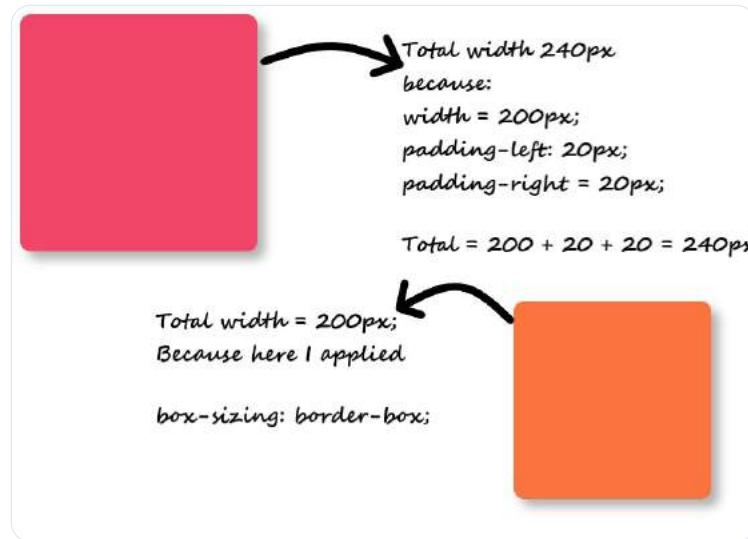
6 Use box-sizing: border-box

Don't know you consider with point valid or not but small thing can make big impact.

Box-sizing: border-box; forces an element to adjust its padding and border inside width and height of that element

CONT... ↗

{ 9 / 16 }



Even small 4px border can damage the responsiveness. Hence I always consider box-sizing border-box for RWD.

{ 10 / 16 }

7 Media Queries are saviour

Media query is a rule to include a block of CSS properties only if a certain condition is true. It is very useful in order to make a RWD.

{ 11 / 16 }

```
● ● ● @prathkum
@media screen and (min-width: 480px) {
  .element {
    width: 100px;
  }
}
```

Here is the basic syntax of media query

Check out the detailed thread on media query [▼](#)

🔗

 Pratham
@Prathkum

A quick beginner's guide to CSS Media Queries

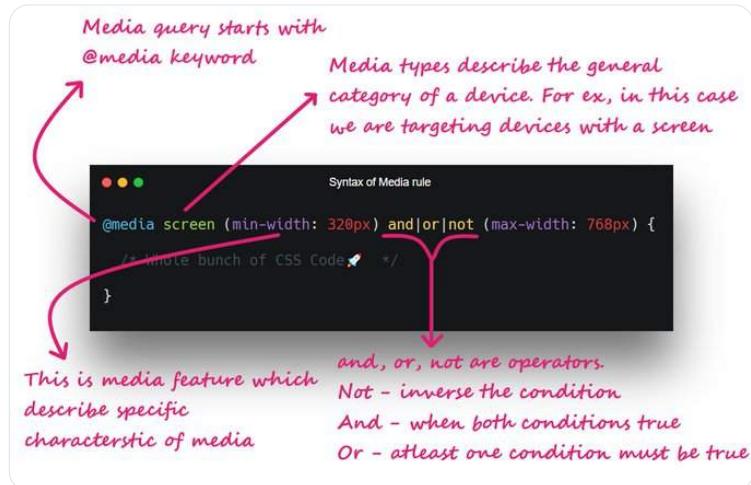
THREAD



9:52 AM · Feb 20, 2021

493 14 Copy link to Tweet

{ 12 / 16 }



8 Use "auto" in media

Almost 99% a web page contains images or videos. And in order to make them responsive, use "auto"

If the width property is set to a percentage and the height is set to "auto", the image will be responsive

{ 13 / 16 }

In order to improve further, you can use max-width in image so that quality of image will be persist. After all it can also be considered as responsive 😊

{ 14 / 16 }

9 Use frameworks if possible

Sometimes it might be a tedious task to handle responsiveness if you're writing pure CSS. Try to use frameworks because they can save a lot of time designing a responsive website

{ 15 / 16 }

I think that's pretty much it for this thread. Did I miss something?

Feel free to add your suggestion below and thanks for reading ❤️

I am glad you all are finding this thread helpful ❤️

Can we take it to 2K likes 😊

• • •



Pratham @Prathkum

19 Apr · 14 tweets · [Prathkum/status/1384081863640190976](#)

Tr

You can learn 90% of CSS using these 10 threads



1. All you need to know about CSS 🎨

Pratham @Prathkum

A Complete Guide to Getting Started with CSS

Thread



6:31 AM · Apr 2, 2021

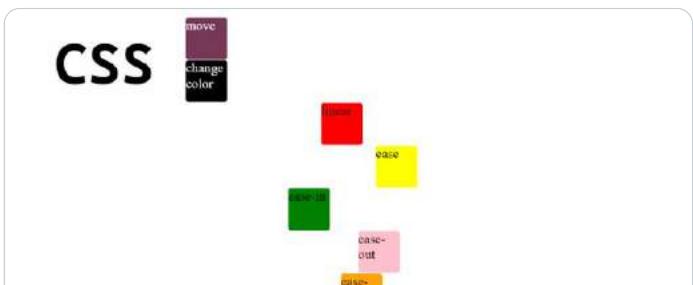
2.6K 68 Copy link to Tweet

2. Getting started with CSS animations 🎨

Pratham @Prathkum

A quick start guide to CSS animations 🎨

Thread



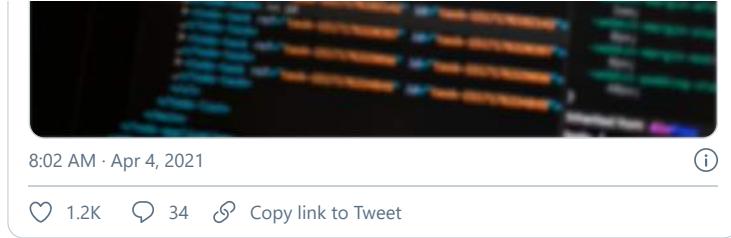
The header features four colored squares with text: 'in-out' (yellow), 'transitions' (dark blue), 'shorthand' (teal), and 'ANIMATIONS' (black). Below the header is the timestamp '6:47 AM · Mar 1, 2021'. At the bottom are engagement metrics: 769 likes, 25 retweets, and a 'Copy link to Tweet' button.

3. Z-index is tricky but this thread solve all your doubts 📈

The header shows the user's profile picture, name 'Pratham', handle '@Prathkum', and a blue Twitter icon. The first tweet reads: 'The Z-index is a powerful yet confusing concept of CSS' and 'Let's make it easy in this quick thread 👇'. Below the tweets is an illustration titled 'Z- INDEX' showing a 3D coordinate system with an eye at the origin. It illustrates how elements with negative z-index overlap the viewport, while elements with positive z-index overlap each other. The illustration includes labels for 'viewport on screen of your device', 'y axis', 'x axis', 'z-index:-1;', 'z-index:1;', 'z-index:2;', 'Element with negative z-index Let's say z-index:-1;', and 'Elements with positive z-index.' At the bottom right is a link to 'Twitter: @prathkum'.

4. CSS positioning concepts 📈

The header shows the user's profile picture, name 'Pratham', handle '@Prathkum', and a blue Twitter icon. The first tweet reads: 'Everything you need to know about CSS position property' and 'Thread 🧵 👇'. Below the tweets is a large image of a computer screen displaying a dark-themed developer interface with the text 'CSS Layout' and 'The Position Property' overlaid in white.



5. Learn how to make your website responsive 🖥️💻📱

A screenshot of a Twitter post from user @Prathkum. The post features a dark-themed image of a computer screen displaying code. The text "Complete guide to Responsive Web Design" is overlaid in white. Below the image, the text "10:45 AM · Apr 10, 2021" is displayed, along with engagement metrics: 2K likes, 34 comments, and a link to copy the tweet.

6. Media Query 📱💻🖥️

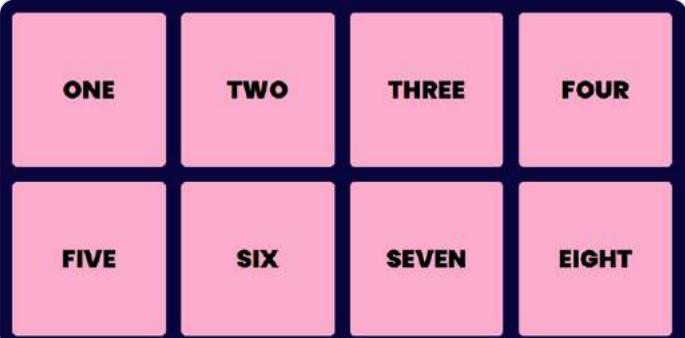
A screenshot of a Twitter post from user @Prathkum. The post features two side-by-side images showing a mobile phone and a desktop computer displaying a website. The text "A quick beginner's guide to CSS Media Queries" is overlaid in white. Below the images, the text "9:52 AM · Feb 20, 2021" is displayed, along with engagement metrics: 510 likes, 17 comments, and a link to copy the tweet.

7. Flexible box layout 📊

 **Pratham**
@Prathkum 

Everything you need to know about CSS Flex layout

A Thread 



7:38 PM · Apr 7, 2021 

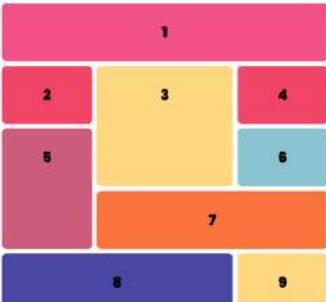
 1.3K  33  Copy link to Tweet

8. Grid layout 📊

 **Pratham**
@Prathkum 

A complete beginner's guide to CSS Grid layout 

Thread 



9:20 AM · Mar 24, 2021 

 2.6K  36  Copy link to Tweet

9. CSS filter method 🌈

 **Pratham**
@Prathkum 

All you need to know about CSS filter methods 

A thread 



10. 100 CSS resources that will help you throughout your life 🔨



Fun Fact 💡

On average, I spent more than 24 hours writing all these threads.

I hope you find these threads helpful 😊❤️

Thanks for the amazing response ❤️

It was worth spending every minute writing this thread

Yeah 😊

My first thread that got more than 5K likes 😳😍

• • •



Pratham @Prathkum

5 May · 6 tweets · [Prathkum/status/1389911003643748353](#)

Tr

If you know CSS then do not write CSS code, use these free generators instead that can help you tremendously

A Thread

1 Gradient Animator

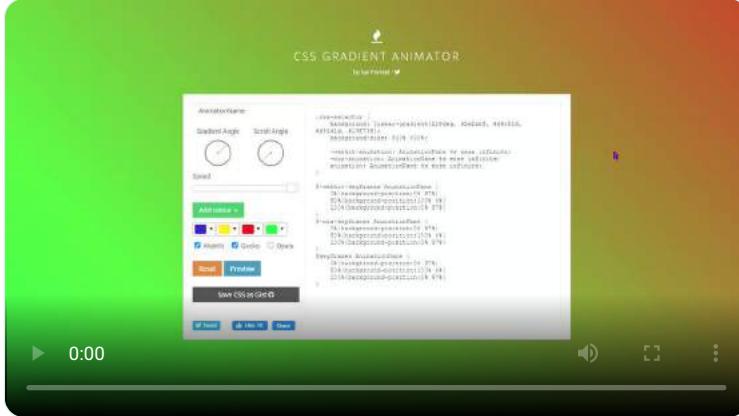
- A CSS generator to create beautiful animated gradients 



The screenshot shows the 'CSS Gradient Animator' interface. It features a 'Gradient Name' input field, two circular 'Start Angle' and 'End Angle' inputs with sliders, a 'Speed' slider, and a green 'Add Color +' button. Below this is a preview area with a color gradient and a 'CSS Gradient Animator' title. The main content area displays generated CSS code for a radial gradient:

```
background: radial-gradient( at 50% 50%, #0000ff 40%, #00ffff 40%, #0000ff 60%, #00ffff 60% );
```

Below the CSS is a note: 'A CSS generator to create beautiful animated gradients for use on your website.' and a link: <http://gradient-animator.com>.

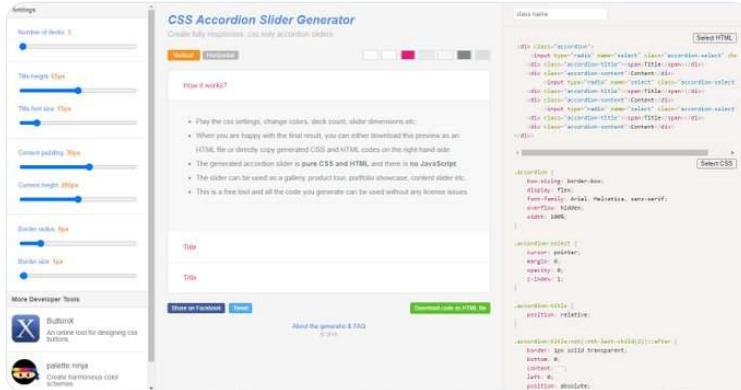


2 CSS Accordion Slider Generator

- Create fully responsive, css only accordion sliders



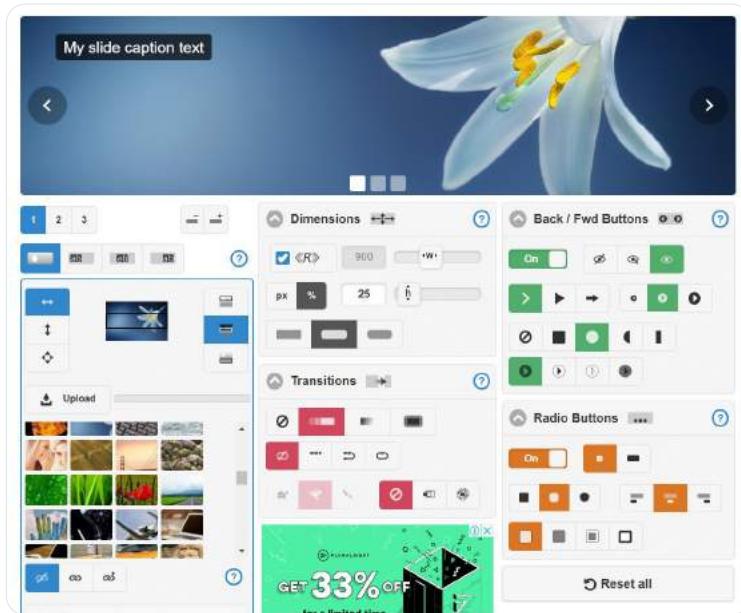
accordionslider.com



3 Image carousel

- Create your responsive image slider in few seconds

[🔗 imageslidermaker.com/v2](http://imageslidermaker.com/v2)

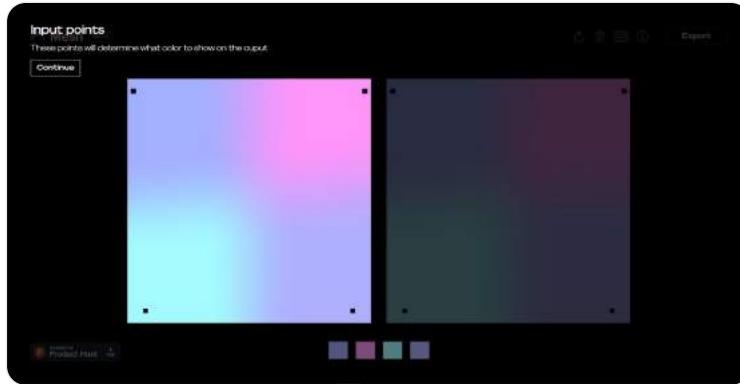


4 Mesh Gradients

- Create beautiful gradient like aurora UI in few clicks

[🔗 https://meshgradient.com](https://meshgradient.com)

MESH - Create beautiful gradients
Create beautiful gradients right in your browser
<https://meshgradient.com/>



5 Color theme generator

- A great tool for CSS dark/light theme

8

CSS Theme Generator v1.0.0

open source • light/dark scheme • normal/high contrast
Based on [NumiDesign](#) theme generator by [@kazuhiko](#) and [takuya](#) color space by [@herominne](#)

Colors Properties

single tone • **Duo tone**

Main Hue: - 242 + insert

Accent Hue: - 0 + insert

Saturation: - 75 +

Use pastel palette:

Type: Main Tint Tone Swap Special

Contrast: Soft Normal Strong

Emphasizing: Dim Normal Bold Only For Tone and Swap items

Preview

Numi.Design Open RPL

Paragraph text Special paragraph text

Basic Card Clear Card

Basic badge **Special badge**

Basic button states

Default Hover Pressed
Toggled Focused
Disabled

Special button states

CSS Color Theme Generator by Numi.Design

Open Source CSS Color Theme Generator for creating lovely toned themes with dark scheme and high contrast mode.

<https://numi.design/theme-generator>

CSS Theme Generator v1.0.0

open source • light/dark scheme • normal/high contrast
Based on [NumiDesign](#) theme generator by [@kazuhiko](#) and [takuya](#) color space by [@herominne](#)

Options

Single tone • **Duo tone**

Hue: - 242 + insert

Saturation: - 75 +

Use pastel palette:

Type: Main Tint Tone Swap Special

Contrast: Soft Normal Strong

Emphasizing: Dim Normal Bold Only For Tone and Swap items

Output

CSS Colors Numi

Switch by: CSS JS CSS JS

The switch by CSS declaration uses CSS Media Queries to adapt the theme to system preferences.
Note: CSS Media Query `prefers-contrast` is an experimental feature and currently is not supported by any browser.
See compatibility tables on [caniuse.com](#) `prefers-contrast` `prefers-color-scheme`

Preview

Numi.Design Open RPL

Paragraph text Special paragraph text

Basic Card Clear Card

Basic badge **Special badge**

Basic button states

Default Hover Pressed
Toggled Focused
Disabled

special button states

Default Hover Pressed
Toggled Focused
Disabled

various widgets

1 2 3 4

• • •



Pratham @Prathkum
6 May · 5 tweets · [Prathkum/status/1390341178730582017](#)

Tr

Do you have videos in your webpage?

Here are 4 tips to style video like a pro ↓

Adding subtitles is always good from accessibility point of view.

You can style the subtitles of video on your website using ::cue pseudo-element

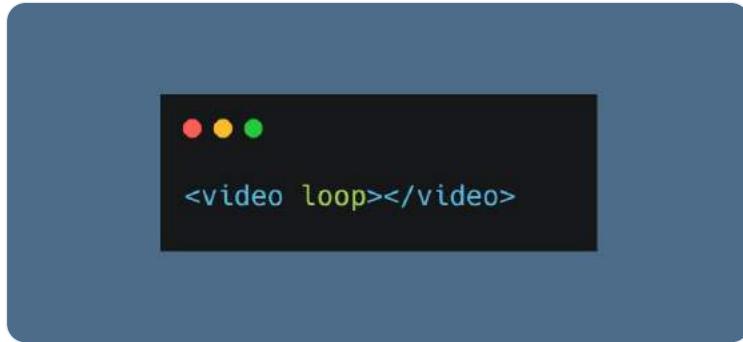
```
::cue {  
  color: black;  
  background: red;  
}
```

The :picture-in-picture CSS pseudo-class matches the element which is currently in picture-in-picture mode.

You can use it for style video on your webpage when its picture-in-picture mode

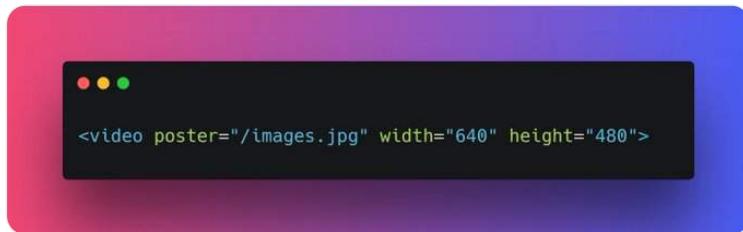
```
:picture-in-picture {  
  box-shadow: 0 0 0 5px red;  
}
```

Loop is a boolean attribute. When present, it specifies that the video will start over again, every time it is finished



Poster attribute specifies an image to be shown while the video is downloading, or until the user hits the play button.

You can consider it as a thumbnail





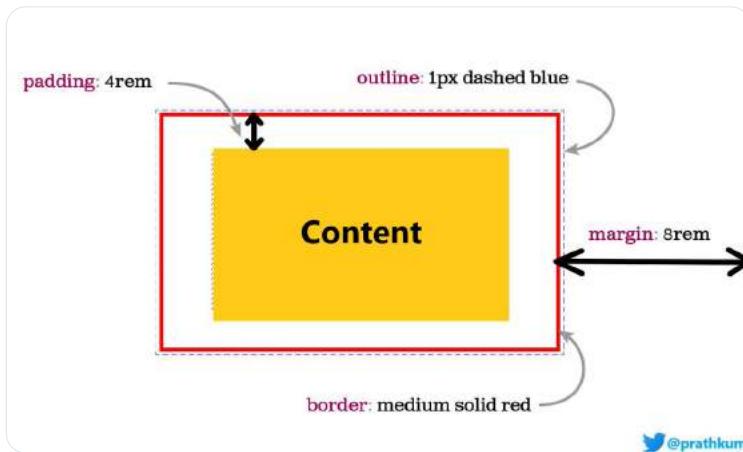
Pratham @Prathkum

14 May · 17 tweets · [Prathkum/status/1393153623748747264](https://twitter.com/Prathkum/status/1393153623748747264)

Tr

Box Model

Everything on a website is a box. Therefore it is quite important to understand the box model concept in web development but it is confusing. Let's see some hidden facts about it in this thread



When laying out a document, the browser's rendering engine represents each element as a rectangular box. You can visualize it by adding some solid in any element.

You can manipulate this box using the basic standards of CSS like color, border, background etc.

{ 02 / 17 }



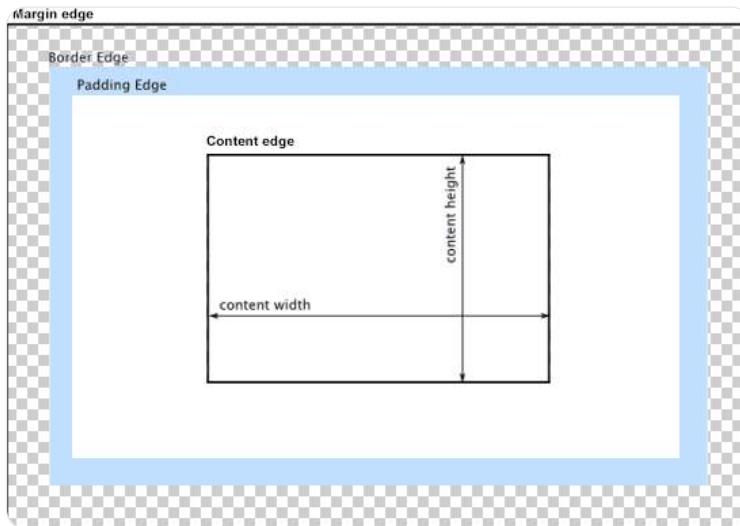
Every box comprised of four parts

- content dimensions
- padding
- border

- margin

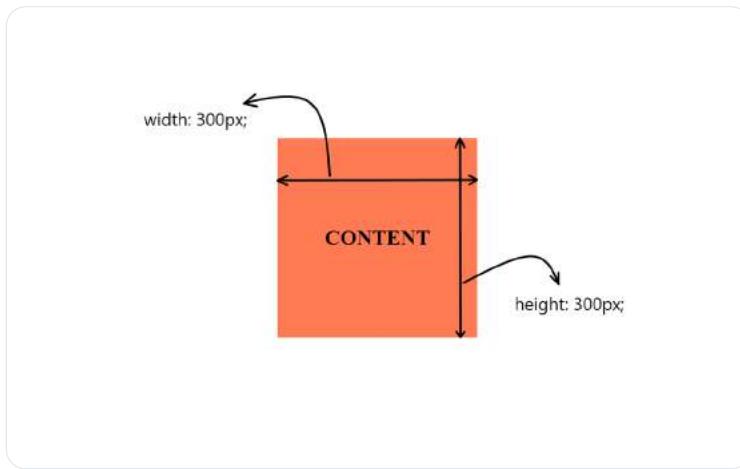
Although outline is not considered as the part of box model but we can talk little bit about it later on this thread.

{ 03 / 17 }



The content area contains the real content in form of text, videos, images. The dimensions of the content area is defined by the height and width CSS property.

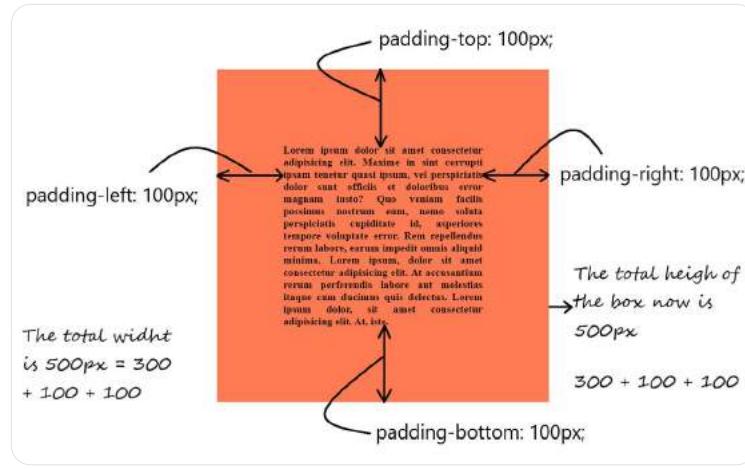
{ 04 / 17 }



Moving further, the next area we have in box is padding area. Basically padding is the empty area between content edge and border.

It is generally used to make the content looks good which eventually increase the readability of content.

{ 05 / 17 }



The padding area is covered with the same color as of element's background color hence we can say that padding area is visible.

{ 06 / 17 }

Moving onto "Border area", border is some fixed thickness separation between padding and margin. The thickness of the borders are determined by the border-width and shorthand border properties.

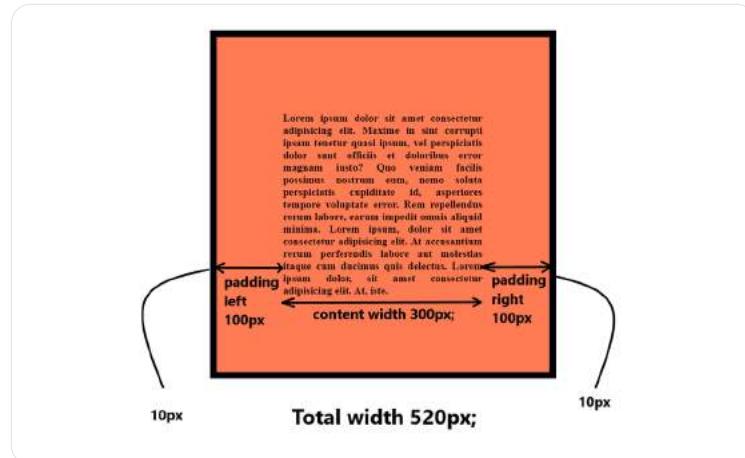
{ 07 / 17 }

The border width is also counted in the total dimension of the box. For example

```
width: 300px;
padding: 100px;
border-width: 10px;
```

then the total width of the box is 520px

{ 08 / 17 }



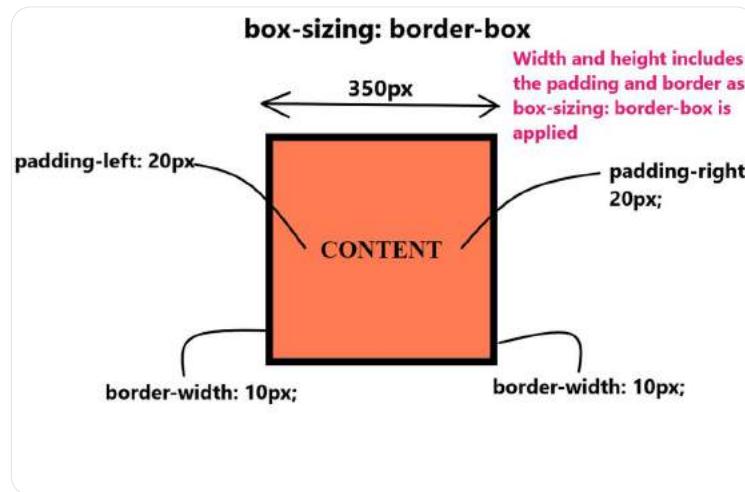
We have noticed that padding and border can increase the dimensions of the box. In order to prevent it and merge the thickness of padding and border within the element's fixed dimensions, we can use one single property known as box-sizing

{ 09 / 17 }

box-sizing: border-box;

The width and height properties include the content, padding, and border, but do not include the margin. Note that padding and border will be inside of the box.

{ 10 / 17 }



The margin area, bounded by the margin edge, extends the border area to include an empty area used to separate the element from its neighbors. Its dimensions are the margin-box width and the margin-box height.

The margin area is transparent.

{ 11 / 17 }

Let me tell you the unique concept of margin in CSS which probably you haven't heard yet... 

{ 12 / 17 }

The margin collapsing

The top and bottom margins of blocks are sometimes combined (collapsed) into a single margin whose size is the largest of the individual margins (or just one of them, if they are equal), a behavior known as margin collapsing.

{ 13 / 17 }

Although there are three main cases in which margin collapsing occurs. They are

1. Adjacent sibling's margin are collapsed
2. Empty blocks margin are generally collapsed
3. No content separating parent and descendants

Read more about it here:

This paragraph's bottom margin is collapsed.

This paragraph is 34px below the above text.

What are the rules of margin collapse in CSS ? - GeeksforGeeks
A Computer Science portal for geeks. It contains well written, well thought and well explained computer science and programming articles, quizzes and practice/competitive programming/company interview...

<http://geeksforgeeks.org/what-are-the-rules-of-margin-collapse-in-css/>

{ 14/17 }

Difference between border and outline are generally confusing. Let's talk little bit about it.

An outline is a line that is drawn around elements, OUTSIDE the borders, to make the element "stand out". outline styling is exactly same as border.

{ 15 / 17 }

The difference between outline and border

- Outline is drawn outside the element's border
- Outline may overlap other content
- Outline is not the part of element's dimensions
- Outline don't effect dimensions

{ 16 / 17 }

I think that's pretty much it for this thread. If you like it, share it with your connections. It means a lot to me ❤️

Peace out 😊

• • •



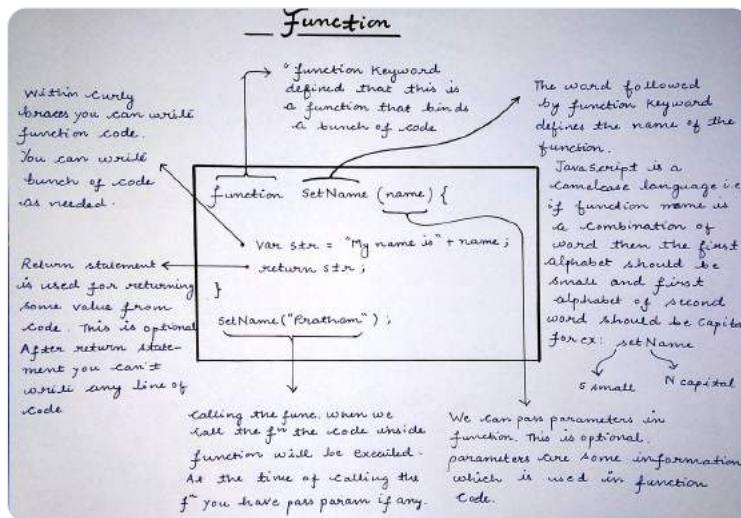
Prathum @Prathkum

18 May · 11 tweets · [Prathkum/status/1394733122911039490](#)

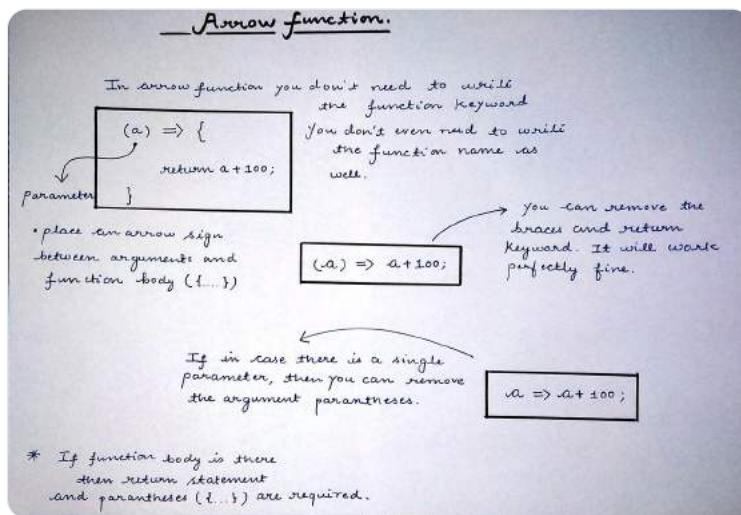
Tr

Some of my handmade JavaScript notes and cheat sheets that can help you 🧳 ↴

1. JavaScript Function Anatomy



2. JavaScript Arrow Function



3. JavaScript Array Methods

```

[1,2,3].map(x=>x*x) // [1,4,9]
[1,2,3].filter(x=>x<2) // [1]
[1,2,3].indexOf(2) // 1
[1,2,3].reduce((x,y)=>x+y) // 6
[1,2,3].reverse() // [3,2,1]

```

```

['P','R'].concat([1,2]) // ['P','R','1','2']
['P','R','A'].slice(1) // ['R','A']
['P','A','T'].splice(1,0,'R') // ['P','R','A','T']
['P','R'].join('') // P-R

```

Array Methods

```

[1,2,3,4].fill(0,2,4) // [1,2,0,0]
[1,2,3,4].find(x=>x>3) // 4
[1,2,3].findIndex(x=>x>1) // 1
[1,2,3].includes(2) // true
[1,2,3].some(x=>x*y==0) // true

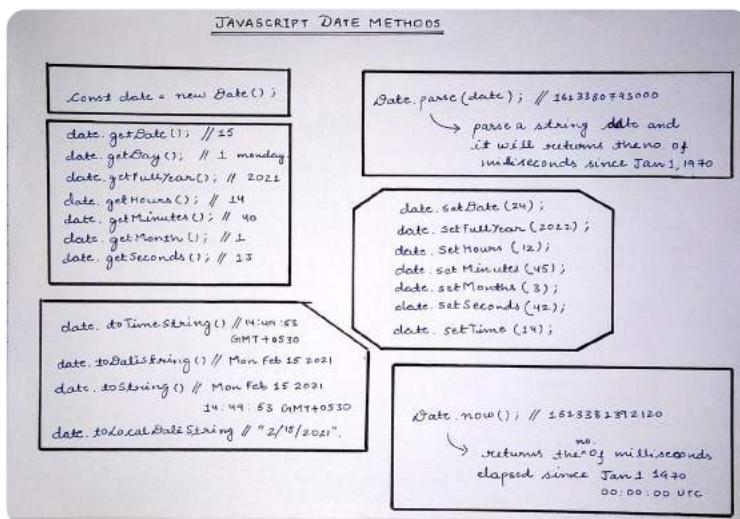
```

```

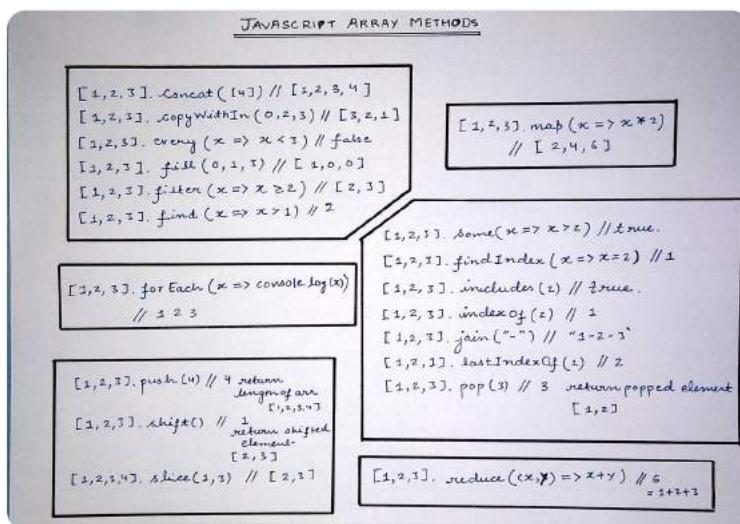
[1,2,3].forEach(x=> console.log(x)) // 1,2,3
[1,2].push(3) // [1,2,3]
[1,2,3].pop() // [1,2]
[1,2,3].shift() // [2,3]
[1,2,3].unshift(0) // [0,1,2,3]
[1,2,3].every(x=>x<5) // true

```

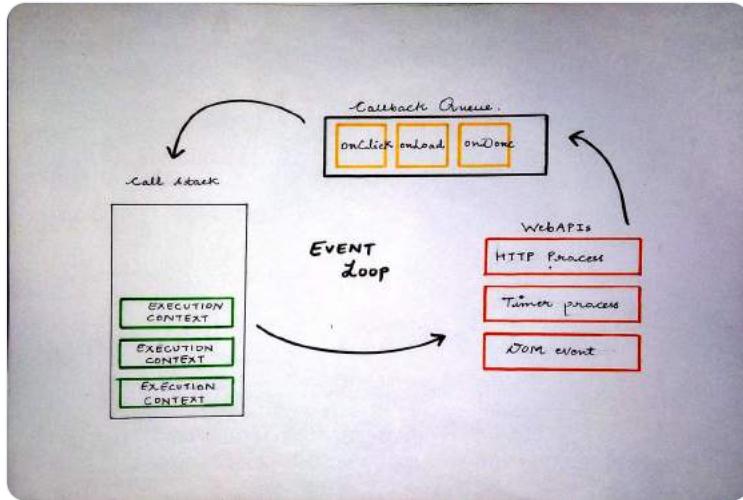
4. JavaScript Date Methods



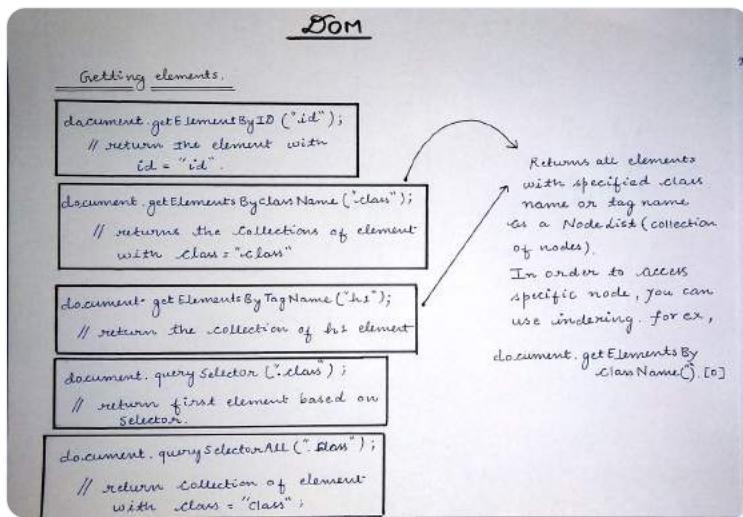
5. JavaScript Array Methods



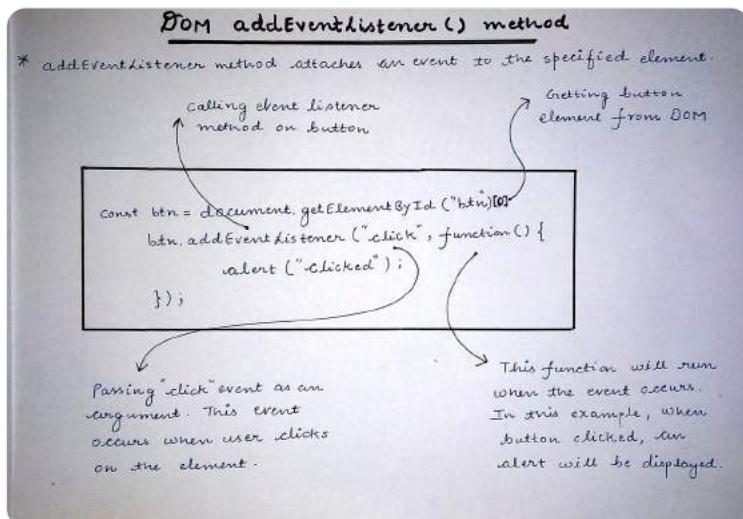
6. Event Loop



7. JavaScript DOM Methods



8. DOM addEventListener Method



More than 0.5 million views on this tweet ❤️🔥

Impressions

524,374

times people saw this Tweet on Twitter

Check it here in action



Greg Slonina
@gregslonina



Replying to @Prathkum

This is amazing. I was so inspired by this I turned it into code if anyone needs it gregslonina.com/javascript-che...

4:30 AM · May 20, 2021

(i)

5 2 Copy link to Tweet

Hey you JavaScript lovers.

I got inspired by a web developer by the twitter name handle of <https://twitter.com/Prathkum>.

He's been creating handwritten javascript cheatsheets that you can find on his twitter account.

Here's an **example**.

```
[1,2,3].map(x=>x*x) // [2,4,6]
[1,2,3].filter(x=>x<2) // [1]
[1,2,3].indexOf(2) // 1
[1,2,3].reduce((x,y)=>x+y) // 6
[1,2,3].reverse() // [3,2,1]
```

```
[P,R].concat([a]) // [P,R,a]
[R,A,a].slice(3) // [R,a]
[P,A,T].splice(3,0,R) // [P,R,A,T]
[P,R].join('') // P-R
[P,R,A,T,W,X,M].lastIndexOf('A') // 5
```

Array Methods

```
[1,2,3,4].fill(0,2,4) // [1,2,0,0]
[1,2,3,4].find(x=>x>3) // 4
[1,2,3].findIndex(x=>x>1) // 1
[1,2,3].includes(a) // true
[1,2,3].some(x=>x>y=2) // true
```

```
[1,x,z].forEach(x=>console.log(x)) // 1,2,3
[1,z].push(z) // [1,z,z]
[1,z,z].pop() // [1,z]
[1,z,z].shift() // [z,z]
[1,2,3].unshift(o) // [o,1,2,3]
[1,2,3].every(x=>x<5) // true
```

Pretty cool huh.

...



Pratham @Prathkum

20 May · 18 tweets · [Prathkum/status/1395372995821060100](#)

Tr

Introduction to React

React is a JavaScript library for building UI components.

The ecosystem of React is really immense which eventually makes it one of the best front-end libraries



Why React?

1. Reusable components
2. Fast due to virtual DOM
3. Huge ecosystem

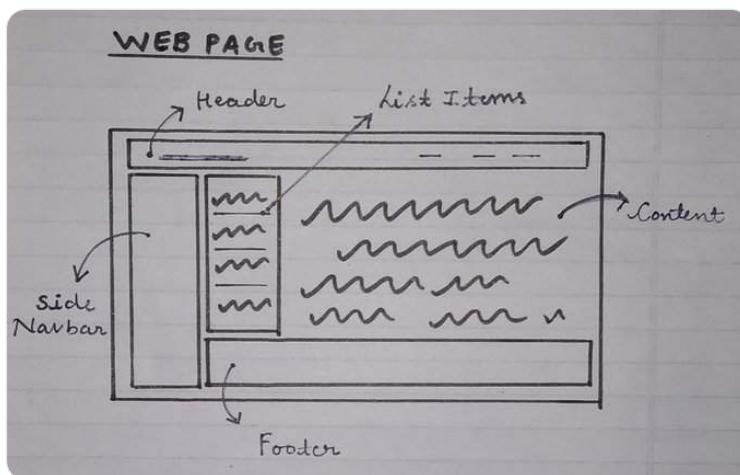
A typical React app contain many components. They are reusable and can interact with each other.

What is a component?

- Component as a simple function that you can call with some input and they render some output

Attached image showing a typical React app with all different components.

As you can see this entire webpage is nothing but the mixture of different components



Components are of two types:

- 1 Class based components
- 2 functional based components

Class-based components are defined using ES6 classes, whereas function components are basic JavaScript functions

Before diving deeper into it, let's talk a little bit about JSX

- JSX stands for JavaScript XML. It's basically nothing but the extension of JavaScript which allow us to write HTML code in JavaScript file.

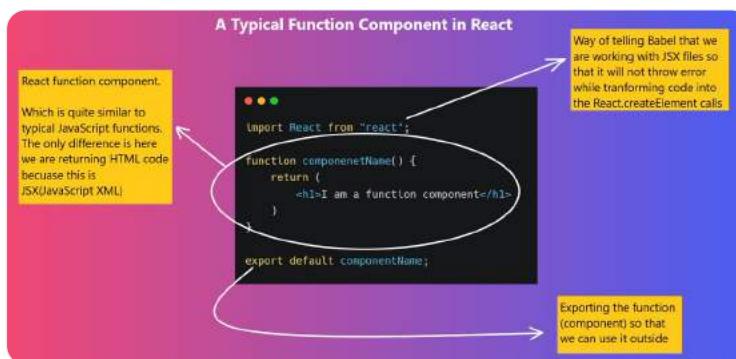
```
const element = <h1>Hello, world!</h1>;
```

Consider this variable declaration. It's neither JS nor HTML. This is the mixture of JavaScript + XML = JSX

Now we know JSX, let's move forward

- Functional components are nothing but simply a JavaScript function which takes some parameter will return some JSX code

A typical function component ↗



A very important concept in React

Virtual DOM

You might have heard the term "DOM", virtual DOM is kind of similar. It uses a strategy that updates the DOM without having to redraw all the webpage elements

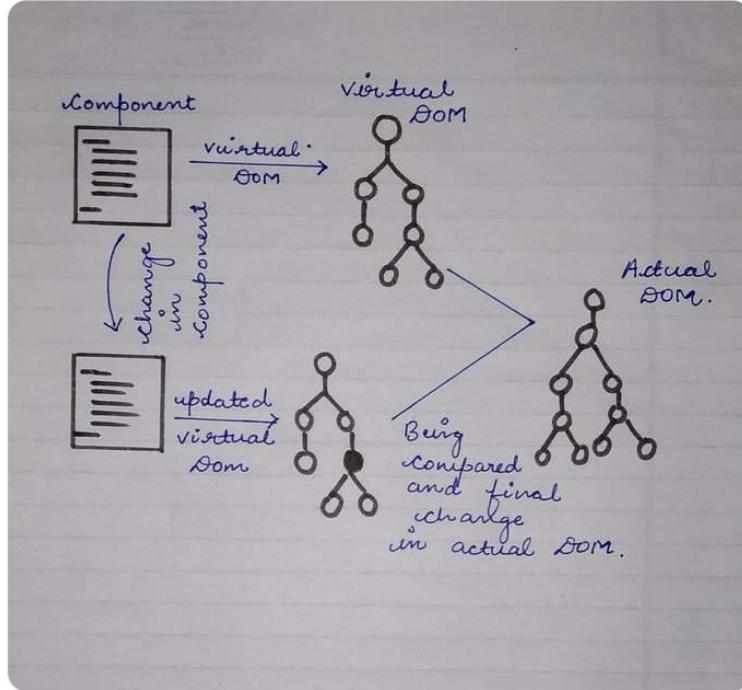
Every time the DOM changes, browser need to recalculate entire layout and then repaint the web page which makes a web app slow.

To overcome this we have virtual DOM

Every time the state of our application changes, the virtual DOM gets updated instead of the real DOM

Whenever the new element is added to the UI, a new virtual DOM associated with that element is created. If state of this element changes, a second new virtual DOM is created which will be compared with the previous virtual DOM

- It then updates ONLY the object on the real DOM



Setting up your first react project directory is quite confusing. Let's see how you can do it

I'm assuming you have node environment set up and up-to-date version of npm. If no, download it from here



Next thing you need to install is `create-react-app` is a tool helps you start building with React app. It set up all the tools that you need in order to get started

Now you have create-react-app installed in your machine, it's time to create your first React app

Command - "create-react-app app-name"

Depending upon your internet speed, this will take some minutes. So time to prepare a coffee for yourself ☕

Once done, run "npm start"

Your default browser will launch automatically and you will see something like this at localhost:3000



And that's it. You just created your first react app. I tried my best to give a quick overview of how things work in React

If you like this thread, drop a like and retweet, means a lot to me ❤️

Peace out 😊

You can start learning React right now but basic understanding of programming and HTML can make the process easier

Here are some important concept of JavaScript you can learn before diving into React

Some important topics of JavaScript before diving into React

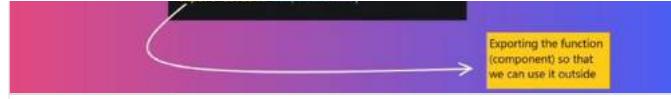
A Thread

9:41 AM · Feb 14, 2021

1K 32 Copy link to Tweet

I just converted this thread into a [@hashnode](#) article





Are you planning to learn React? Let me make the process easier for...

React is a JavaScript library for building UI components. The ecosystem of React is really immense which eventually makes it one of the best front-end libraries Why you should learn React? Reusable ...

<https://pratham.hashnode.dev/are-you-planning-to-learn-react-let-me-make-the-proces...>

• • •



Pratham @Prathkum

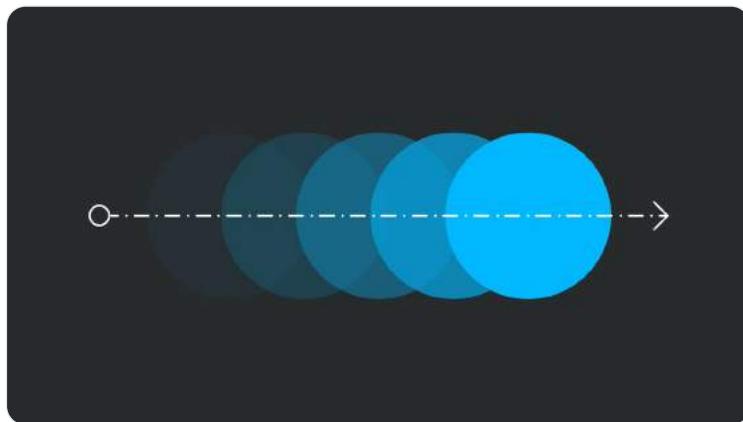
25 May · 22 tweets · [Prathkum/status/1397170589677281285](#)

Tr

A complete introduction to CSS animations 🚀

CSS animation is a module in CSS that lets you change the styling of elements and therefore creates an animate effect. It's little tricky but not much

Let's learn about it 🧳 ➡️



Before diving into it, we need to understand the importance of CSS animations

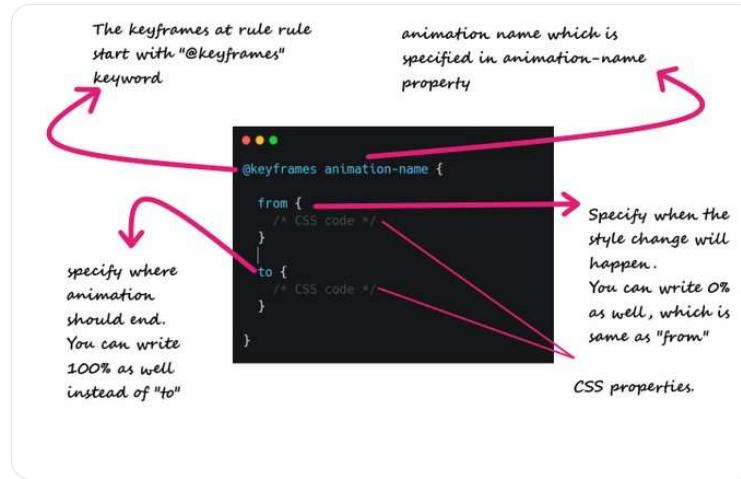
Animation can establish interaction between users and the user interface. This makes your website more interactive which ultimately attracts more visitors to your webpage.

CSS animations is little tricky but not too tough to master. In this thread, we will try to gain some decent knowledge about it.

Animation is all about changing one style to another at certain intervals or times. For doing that, we need to learn about `@ keyframes` at rule

The keyframe provides us animation by changing the style sequentially according to the mentioned interval.

The syntax of keyframes at-rule ➡️



In order to bind the keyframe with a particular animation, we need to define few things like name and timing of animations.

We have several properties for that like animation-name, animation-timing, etc...

Either we can use each property of animation or we can use animation shorthand property which includes:

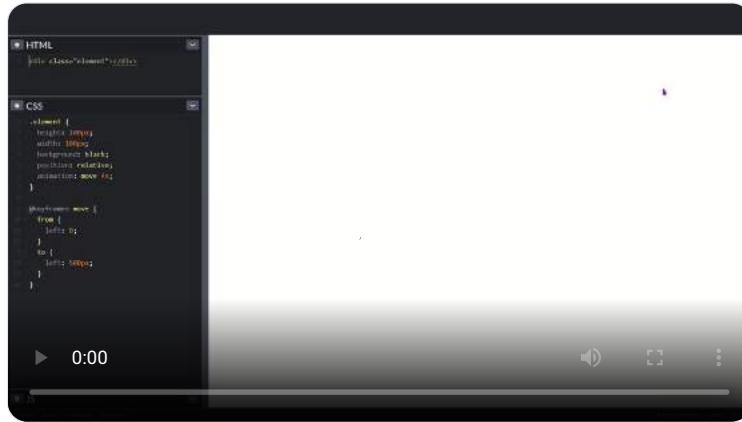
- animation-name
- animation-duration
- animation-timing-function
- animation-delay
- animation-iteration-count
- animation-direction
- animation-fill-mode
- animation-play-state

Lets create a simple animation which will solve all our doubts:

For example, suppose I want to move my element 500px left in four seconds. It's very easy



The output of above attached code:

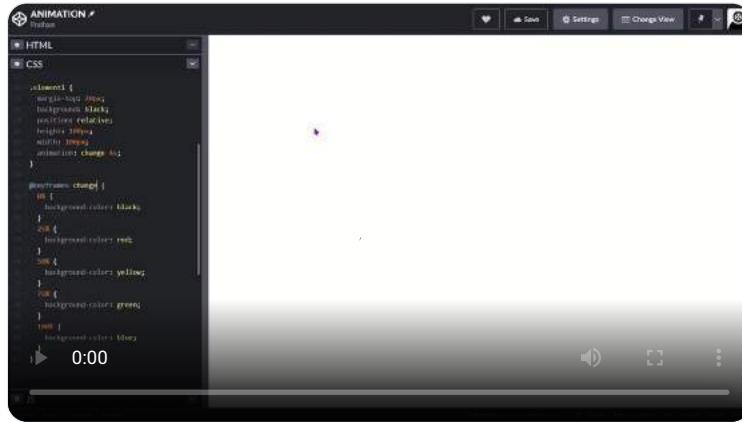


Explanation:

Black box is going 500px from left boundary in 4 seconds because we have mentioned 4 seconds in animation property and 500px in keyframe at rule.

Alright moving further, we can also pass percentage in keyframes rule and make some amazing animations

For example, suppose I want to change the background-color of an element

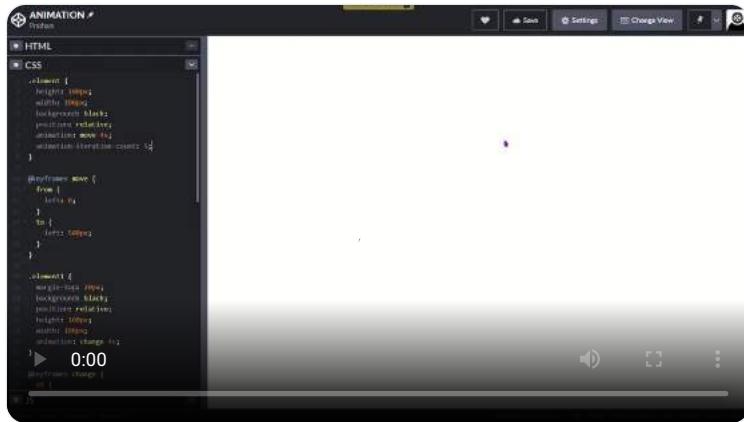


You can also specify the delay in your animation using animation delay property.

It specifies a delay before the start of an animation

We have another cool property called "animation-iteration-count"

It specifies the number of times an animation should run repeatedly. You can pass a numeral value or "infinite" in this property

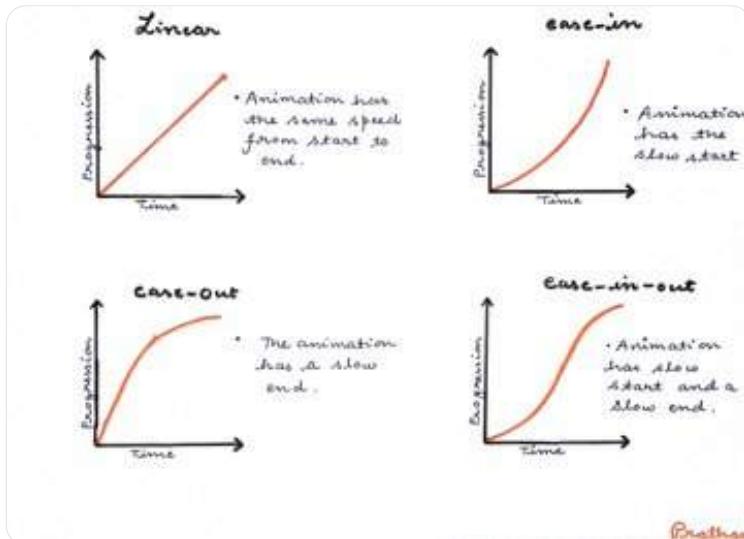


So far so good, But we can also change the speed curve of animation and can create more smooth animation. And we have animation-timing-function property for that.

You can pass following values in animation-timing function

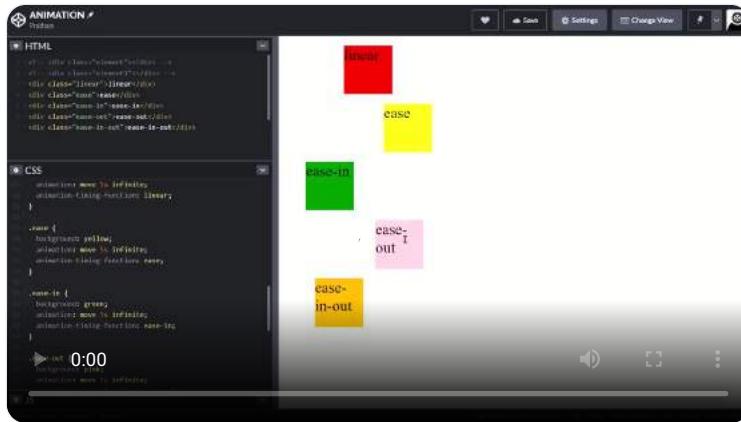
- ease (slow start, then fast, then slow)
- linear (same speed)
- ease-in (slow start)
- ease-out (slow end)
- ease-in-out (slow start and slow end)
- cubic-bezier (customizable)

Here are some visual representation of speed curve 



See this video for better understanding or you can play with code here

<https://codepen.io/prathkum/pen/OJbvOoZ>



One thing to note here is that animation do not affect element before or after the keyframes.

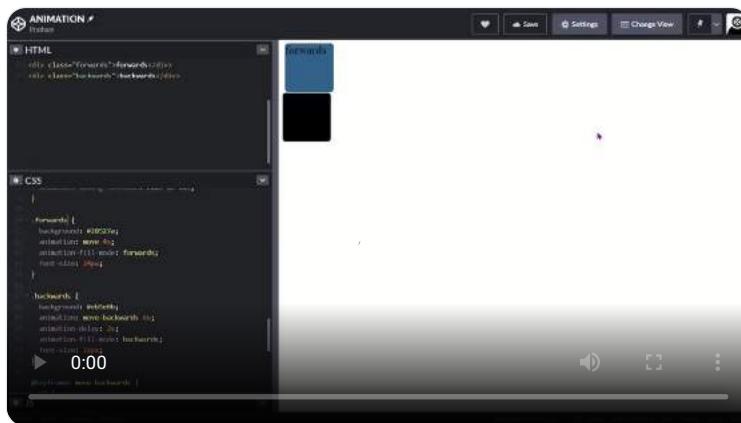
In order to persist the styling based on last or first keyframe, we have animation-fill-mode

It accepts following value

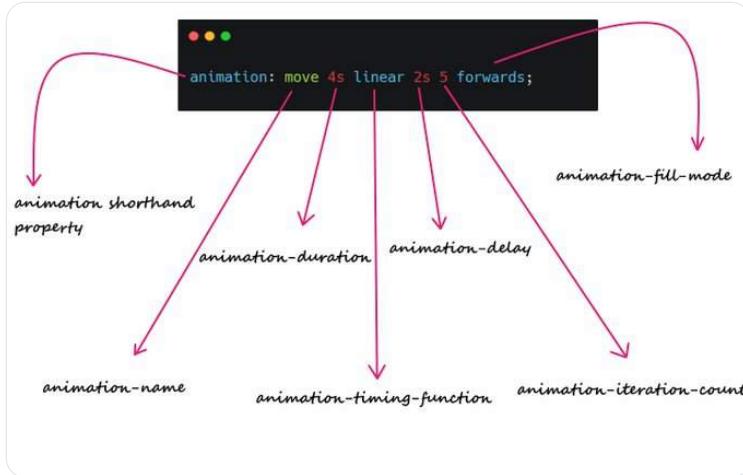
- forwards (element will retain last keyframe styling)
- backwards (element will get the first keyframe value even in animation-delay period)
- both (both of the above)

Let me explain animation-fill-mode by this video ↗

- As you can see blue box stopped at left: 500px because forwards is begin applied on it
- And orange box has black background even it's in a delay period of 2 seconds because backwards is being applied on it



This is the correct order to animation shorthand property ↗



And that's it. I tried my best to give a quick overview of how CSS animations works

If you like this thread, drop a like and retweet, means a lot to me ❤️

Peace out 😊

• • •

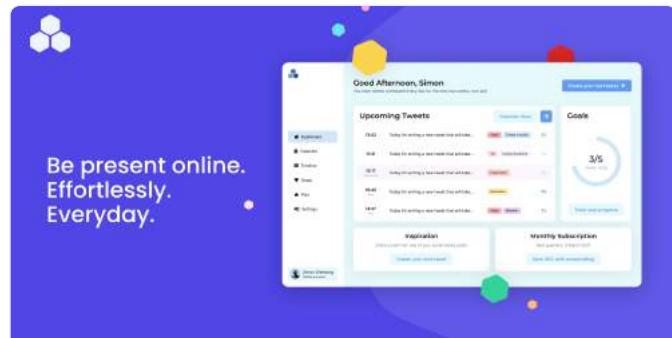


Pratham @Prathkum

30 May · 1 tweets · [Prathkum/status/1399098652908396547](https://twitter.com/Prathkum/status/1399098652908396547)

Tr

I compiled all my threads in the most logical order and created a beginner level CSS course for free



FeedHive

FeedHive.io

<https://app.feedhive.io/public/post/cn265tSmq1>



...



Pratham @Prathkum

3 Jun · 26 tweets · [Prathkum/status/1400394232469262340](https://twitter.com/Prathkum/status/1400394232469262340)

Tr

Are you new in the field of Web development or planning to start with it?

This is how you can start with total ease.



Getting started with web development is little tricky and tedious. I personally faced some difficulties when I was first learning it.

But you don't have to worry. In this thread we will make the process easier

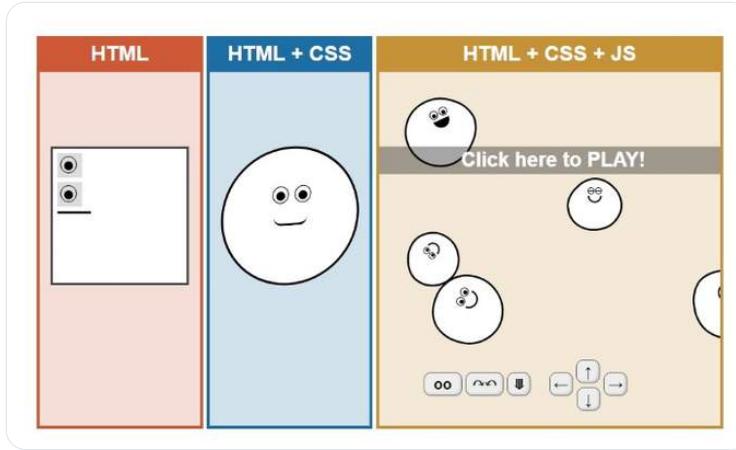
Three different languages you need to learn

HTML

CSS

JavaScript

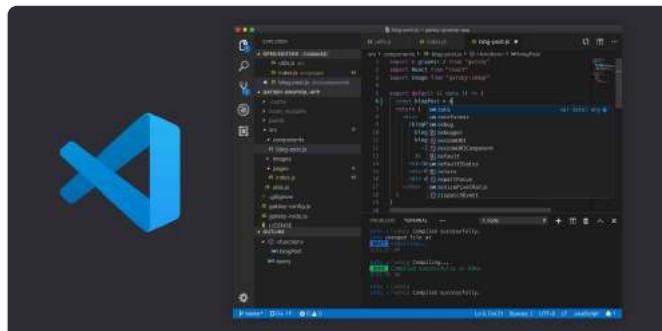
You need to learn these 3 things in a chronological order



First things first, You need an editor where you can write code. I'll recommend you to start with VS Code. Why?

- IntelliSense code completion
- Better environment

Download it from here:



Download Visual Studio Code - Mac, Linux, Windows

Visual Studio Code is free and available on your favorite platform - Linux, macOS, and Windows. Download Visual Studio Code to experience a redefined code editor, optimized for building and debuggi...

<https://code.visualstudio.com/download>

Start with learning HTML

Its pretty easy and you don't need to spend much days on it. You can also start learning it through any YouTube video right away.

Project based learning is the best

I suggest you to create a simple portfolio or personal site using HTML only. This way you can make projects while learning and later on you can include them in your Resume as well

Time to learn CSS

Although CSS is deep but it revolves around few concepts only.

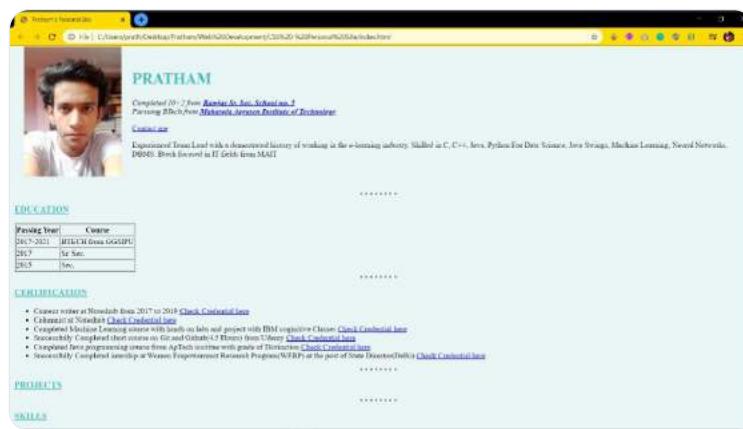
In the beginning you don't need to learn the entire CSS in one single go, learn basics so that you would be able to make basic websites

Some CSS topics you need to learn

- color
- background
- font
- layout
- positioning
- media query

Again, I suggest you to make project along side.

When I was learning HTML and CSS, I create this portfolio 



You can start learning CSS using this latest course



The image shows a promotional graphic for a CSS course. It features the text "Learn CSS!" in a large, bold font, with each letter in a different color. Below the text is the "web.dev" logo. At the bottom, there's a description of the course: "Learn CSS An evergreen CSS course and reference to level up your web styling expertise." followed by the URL "https://web.dev/learn/css/".

CSS Framework

CSS framework can save you a lot of time. You can use CSS frameworks as someone's code which you can use in your project

Learn Bootstrap or TailwindCSS, that's my recommendation

🐦 JavaScript

JavaScript is used to add functionality in your website so that user can interact with your webpage. There are plethora of course out there for JavaScript, you can pick any updated one

Here are some websites from where you can learn or improve your JavaScript knowledge

These websites will help you improve your JavaScript knowledge

1:22 PM · Mar 21, 2021

1.7K 33 ⏲ Copy link to Tweet

🐦 Document Object Model

So how we can change thing in HTML using JavaScript?

Here DOM comes into play

With the HTML DOM, JavaScript can access and change all the elements of an HTML document

By the combination of DOM and JavaScript you can perform a lot of cool things.

- You can change styling
- You can change content
- You can change tags
- You can change attributes
- ETC.....

You can learn DOM from here

codecademy

DOM Events with JavaScript | Codecademy

Learn to create webpage interactivity by leveraging JavaScript events in the browser.

<https://www.codecademy.com/learn/build-interactive-websites/modules/dom-javascript-...>

And again, you need to build some projects. When I was learning this, I made a lot of projects. You can find them on my GitHub link



PrathamKumar14 - Overview
I have a passion for web development and love to create websites for web and mobile devices - PrathamKumar14
<https://github.com/PrathamKumar14>

📌 Command Line Interface

Developers prefer CLI because it's faster and provides you extra power because you can control OS as well with some commands

📌 Version Control

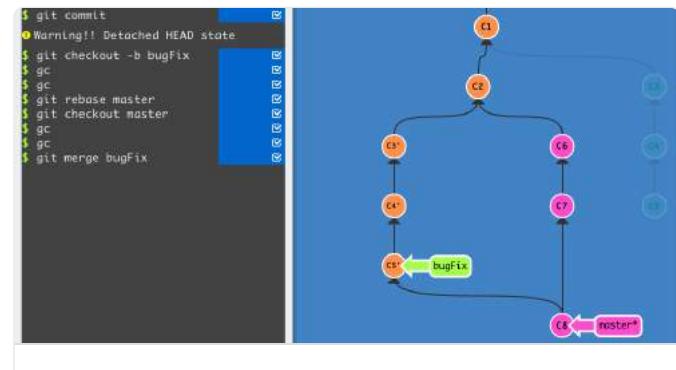
Version control, also known as source control, is the practice of tracking and managing changes to software code. In simple terms, you can store your code in other computer and it can help persist your data if you delete code from your computer.

There are a lot of version control softwares but Git is pretty famous and its ecosystem is great

GitHub is a code hosting platform for version control (Git)

Check out this great free website and learn git visually

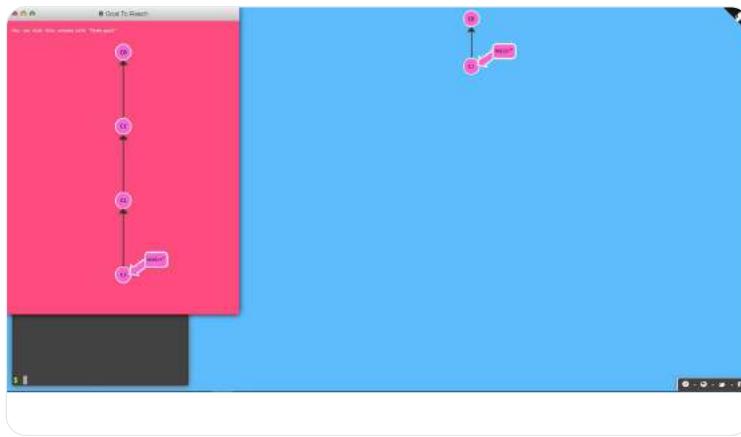




Learn Git Branching

An interactive Git visualization tool to educate and challenge!

<https://learngitbranching.js.org/>



So far so good. At this point you would be able to get a job, freelancing and other stuff. But in order to dive deep into the Web development ocean, we need to learn some front-end framework

I recommend React because it is fast, flexible and the community is really big.

I have a fully fledged thread on React. Check it out



Pratham
@Prathkum



Introduction to React 

React is a JavaScript library for building UI components. The ecosystem of React is really immense which eventually makes it one of the best front-end libraries

 
1:37 PM · May 20, 2021


 3.1K
 146
 Copy link to Tweet

Make more projects and test your skills. Include them in your Resume and start applying for a front-end role

• • •

That's pretty much it for this thread. Feel free to ask your doubts if any. I'll catch you with the next thread until then

Peace out 😊

With this thread, You can buy my CSS Cheat Sheet for free. Hurry up next 100 copies only

Discount Code: pratham ⚡



CSS Cheat Sheets
In this eBook you will find all my handwritten CSS notes
<https://gum.co/css-cheat>



Pratham @Prathkum

3 Jun · 6 tweets · [Prathkum/status/1400414868021387264](https://twitter.com/Prathkum/status/1400414868021387264)

Tr

5 great websites a web developer should visit

Thread 🧶 ↴

1. Can I use?

- A website that provides up-to-date browser support tables for support of front-end web technologies on desktop and mobile web browsers.

🔗 caniuse.com



2. Web Skills

Websites for all kind of learning resources for web developers

- Fundamentals
- Accessibility
- Web components
- Progressive web apps
- Frameworks and libraries
- Testing
- Architecture and Paradigm
- UI and UX
- DS and Algo

🔗



Web Skills

A visual overview of useful skills to learn as a web developer

<https://andreasbm.github.io/web-skills/>

A screenshot of a website titled "1. FUNDAMENTALS". It features three main sections: "HTML" with icons for basic tags like <h1>, , and <div>; "CSS" with icons for styling elements like borders, backgrounds, and transforms; and "JavaScript" with icons for functions, loops, conditionals, and objects. A central callout box for "Grid" provides a brief introduction to CSS Grid.

3. Learn Shell

Whether you are an experienced programmer or not, this website is intended for everyone who wishes to learn programming with Unix/Linux shell interpreters.

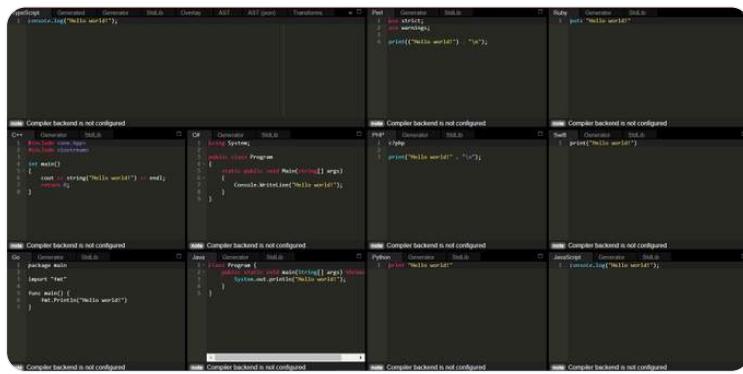


The homepage of learnshell.org. It features a large "LEARN SHELL .ORG" logo with a terminal icon to its left. Below the logo is a section titled "Learn Shell - Free Interactive Shell Tutorial" with a brief description and a link to the site.

4. Programming Language Converter

- This is a super cool tool where you can convert code from one programming language into another.

 ide.onelang.io



5. Gitignore

A collection of useful .gitignore templates for your project. Select from 442 Operating System, IDE, and Programming Language





gitignore.io
Create useful .gitignore files for your project

Search Operating Systems, IDEs, or Programming Languages Create

Source Code | Command Line Docs | Watch Video Tutorial

gitignore.io
Create useful .gitignore files for your project by selecting from 522 Operating System, IDE, and Programming Language .gitignore templates

<https://toptal.com/developers/gitignore>



gitignore.io
Create useful .gitignore files for your project

Search Operating Systems, IDEs, or Programming Languages Create

Source Code | Command Line Docs

• • •



Pratham @Prathkum

5 Jun · 17 tweets · [Prathkum/status/1401137125496602625](https://twitter.com/Prathkum/status/1401137125496602625)

Tr

CSS is also a deep module of web development. There are over 250 unique properties

Do we need to learn them all? Let's try to figure out how much CSS will be enough



CSS is an amazing and unique language that servers a great purpose. We can make our website visually good using CSS. It describe the presentation of web pages, including typography, layouts, color etc...

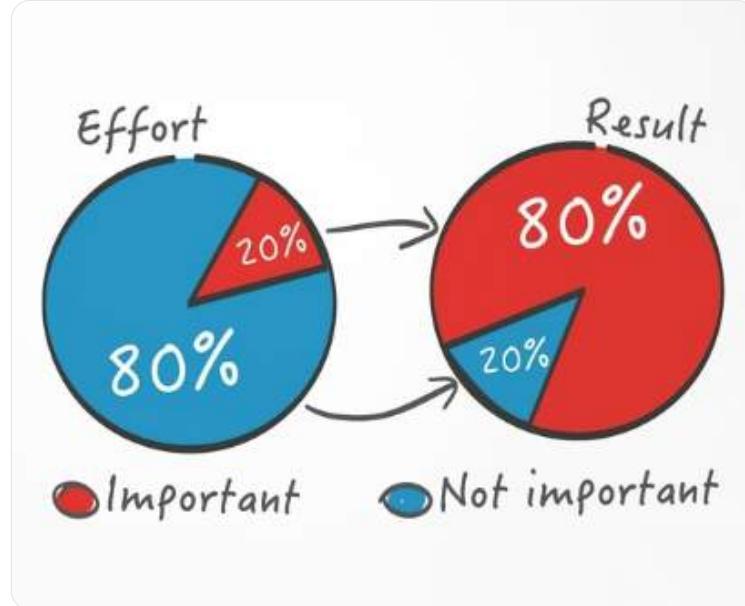
CSS is totally operates on properties value pair. And there are around ~300 distinct properties in CSS

Obviously remembering them all and their values is almost impossible.

In my opinion, entire CSS revolves around 20 properties only

You must have heard about "Pareto principle" which states that for many outcomes roughly 80% of consequences come from 20% of the causes

Similarly, in order to cover 80% of CSS, you need to learn only 20 properties



For example: Have you heard about "image-rendering property"?

If not, then its not a problem
if yes, then its great

This property basically defines the image scaling algorithm



There are a lot of this types of properties in CSS which you never gonna use for sure.

Like,

- 📌 caret-color
- 📌 mask-composite
- 📌 will-change

Lets talk about some important concepts which you should know to be a front-end developer ↪

First and foremost

- 📌 The background and color

The characteristic of a great website is its color scheme. Forget about everything and learn about background and color properties initially

The colors are something from which users interact first whenever they visit your webpage

The background is the shorthand property which covers background-color, background-attachment, etc

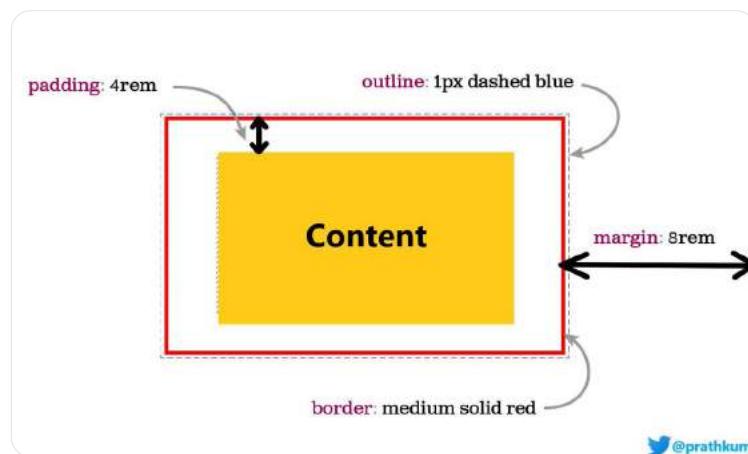
But again, you will rarely use background-attachment, background-origin, etc

👉 The box-model

You will see these every single time. Like a website is based on the boxes and that's why this is very important

The box-model covers

- Height
- Width
- Padding
- Border
- Margin



👉 Typography

Moving forward, typography is an essential thing of a web page. A good font can make your webpage stand out and establish a strong visual hierarchy, provide a graphic balance to the website, and set the product's overall tone

👉 Layout

- CSS positioning
- Flexible box layout
- Grid layout

Structuring your website is essential as it provides better navigation and visualization of the content available on the site. Let's talk a little more about it and learn how we can create a proper layout

📌 Responsive Web Design (RWD)

An extremely important topic in CSS. Responsive web design is little tricky but not so tough to master.

It covers

- media query
- layouts

You can check my detailed thread on RWD

Pratham
@Prathkum

Complete guide to Responsive Web Design

10:45 AM · Apr 10, 2021

2.2K 52 Copy link to Tweet

I guess after learning all these concepts, you are good to go to JavaScript.

But remember CSS is so deep and there is always a little margin of improvement

without taking much of your time, I guess this is pretty much it for this thread. If you like it, share it with your connections it means a lot to me ❤️

Peace out 😊

• • •



Pratham @Prathkum

19 Jun · 19 tweets · [Prathkum/status/1406298132737150976](#)

Tr

JavaScript is an enormous language but you don't need to learn everything in the beginning.

Here is the detailed explanation on JavaScript for Web Development  



JavaScript is a kind of language that as you progress deeper in the field, you will come to know a lot more cool concepts about it.

Prerequisites if you want to learn JavaScript for web development 

- HTML
- CSS

First and foremost, it's a programming language so you should know about basic programming concepts:

- Data types in JavaScript
- Variables
- Statements
- Control statements
- Operators
- And other basic stuff....

When I started learning web development, I had basic knowledge about Java language which helped me a lot because I was no more in a need to learn about these general programming concepts.

But you can start with the ZERO knowledge as well.

There are some advanced topics in JavaScript which you don't need to learn in the beginning. For ex:

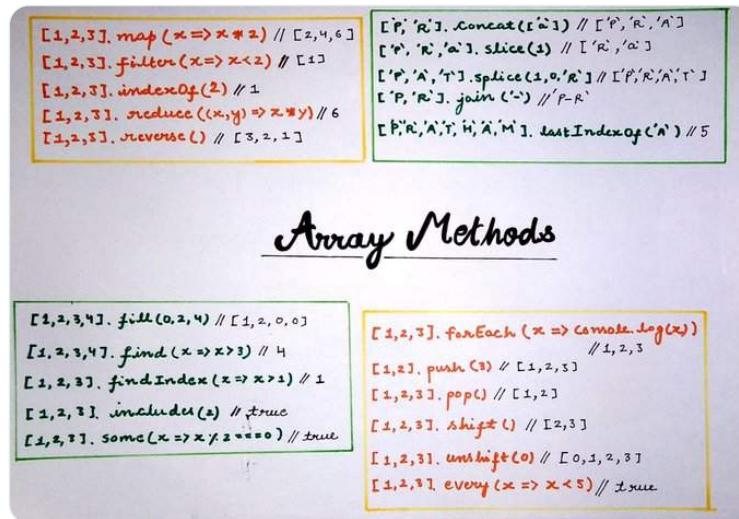
- Async/await
- callbacks, promises, etc...

These are some advance concept which you can learn after

You need to cover some intermediate topics so that you can add behaviour to your website. They are

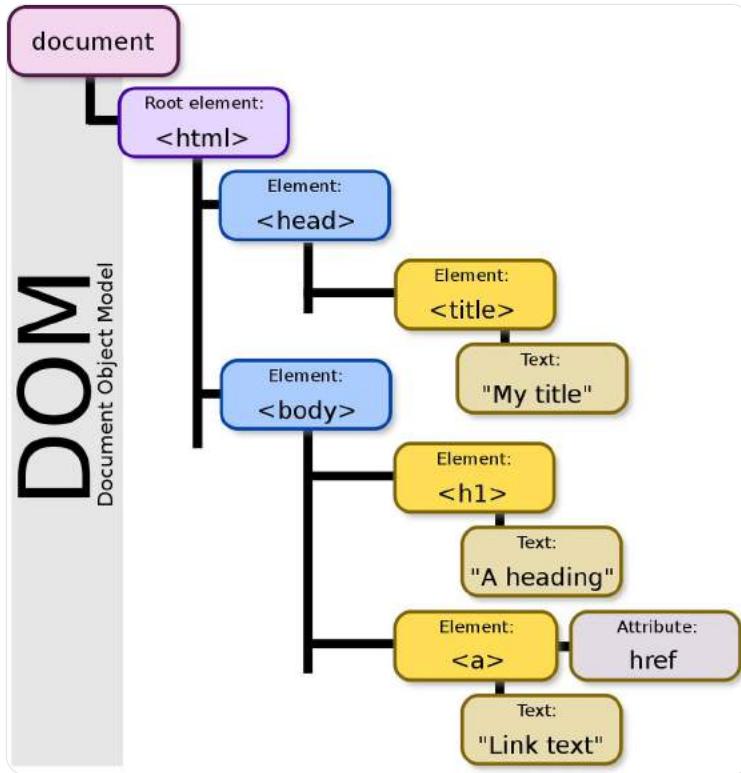
- Arrays and their methods
- Objects
- Functions
- Arrow function

These will help you to work with DOM.



Let's talk about DOM a little bit. It stands for Document Object Model.

Consider it as a tree that comprises the content of a particular web page (HTML)



DOM gives you a superpower to change your webpage at your convenience using JavaScript.

- You change the HTML elements, attribute, styles
- You can delete existing HTML elements
- You can add HTML elements

JavaScript and DOM are connected.

We use DOM to access the elements but write JavaScript code in order to modify them.

Here is a simple example:

```
const paragraphs = document.querySelectorAll("p");
// paragraphs[0] is the first <p> element
// paragraphs[1] is the second <p> element, etc.
alert(paragraphs[0].nodeName);
```

Don't get confused here. DOM is not a programming language it's just a model using which we can access and modify HTML elements,

Basic things you need to cover in DOM

Finding HTML Elements

- getElementsByTagName()

- getElementsById()
- getElementsByClassName()

📌 Changing HTML Element

- element.innerHTML
- element.attribute
- element.style.property
- element.setAttribute(attr, value)

Pushpin Adding and deleting elements

- document.createElement(element)
- document.removeChild(element)
- document.appendChild(element)

Just one last thing and then you will be able to make fully-fledged websites.

After learning these basic properties and methods, its time to move onto Events and Event Listener

The addEventListener() method attaches an event handler to the specified element.

Up to this point you will be able to make a fully functional website using JavaScript. But there are always some margin of improvement

Here are some advanced key concepts

- Hoisting
- Closures
- Callbacks
- Promises
- Async & Await
- Currying
- And other ES6 feature

Here is the complete general introduction to JavaScript



Pratham
@Prathkum

Introduction to JavaScript ☕

JavaScript is probably one of the most commonly used programming languages nowadays. It has a wide range of applications in almost all technologies

2:01 PM · May 22, 2021

 1.5K  72  Copy link to Tweet

The most important concept on which the entire JavaScript language based on

The Event Loop



Pratham
@Prathkum

Introduction to Event Loop 🐾

Event loop is the confusing concept in JavaScript but you need to know about it because whole JavaScript is based on this programming construct.

Let's try to understand it in a simple way 🧩 🌱

3:00 PM · Jun 5, 2021

1.5K 42 Copy link to Tweet

Project-based learning is the best. Here are some practice projects you can build.



40 JavaScript Projects for Beginners – Easy Ideas to Get Started Codi...

The best way to learn a new programming language is to build projects. I have created a list of 40 beginner friendly project tutorials in Vanilla JavaScript, React, and TypeScript. My advice for tu...

<https://www.freecodecamp.org/news/javascript-projects-for-beginners/>

And I guess that's pretty much it for this thread. Hope you find it helpful ❤️

Whoa! Half million views 😲❤️



Pratham @Prathkum

JavaScript is an enormous language but you don't need to learn everything in the beginning.

Here is the detailed explanation on JavaScript for Web Development pic.twitter.com/rNarhlHgiy

Impressions

544,735

times people saw this Tweet on Twitter

Total engagements

20,019

times people interacted with this Tweet

...

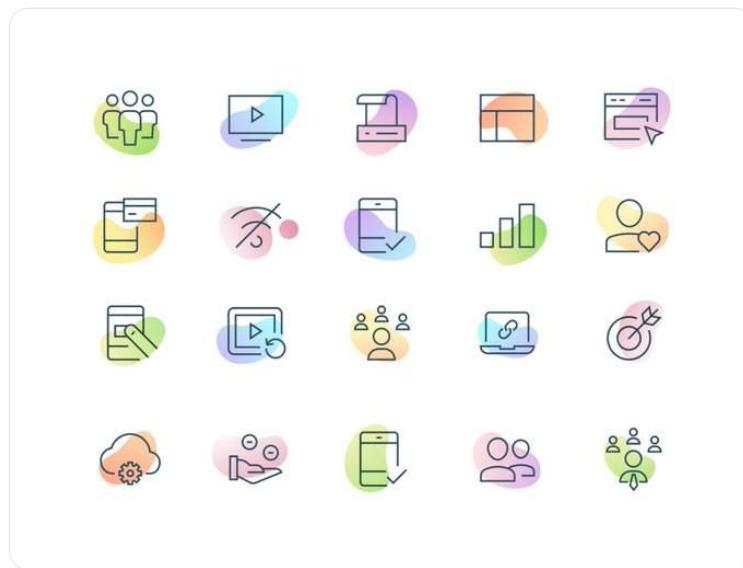
Pratham @Prathkum

22 Jun · 21 tweets · Prathkum/status/1407249651338727426

Tr

Here are some great websites that can help you in your next web development project.

Including colors, accessibility, backgrounds, icons, testing, and much more



1. Color hunt

- Color Hunt is a free and open platform for color inspiration with thousands of trendy hand-picked color palettes



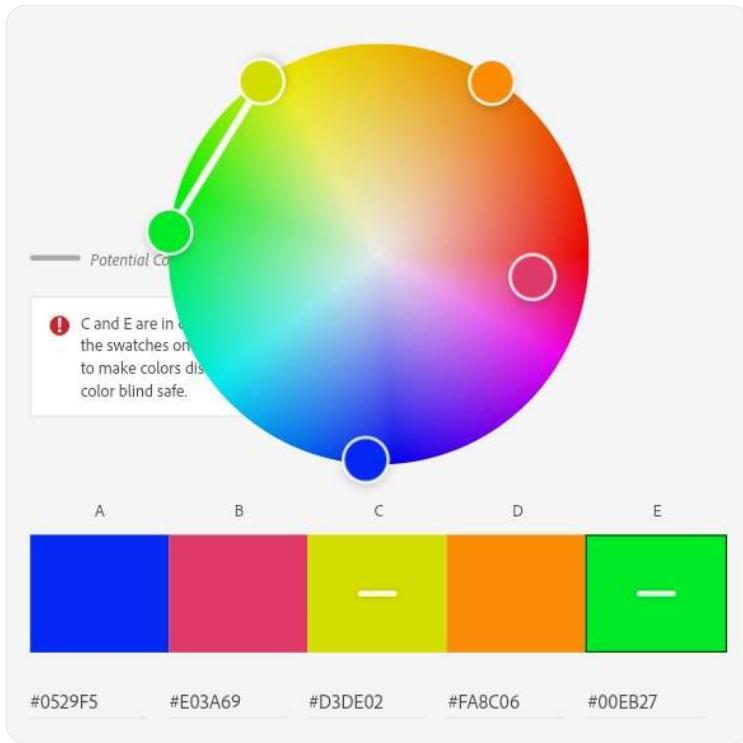
Color Palettes for Designers and Artists - Color Hunt
Discover the newest hand-picked color palettes of Color Hunt. Get color inspiration for your design and art projects.
<https://colorhunt.co>



2. Adobe color wheel

Explore and create accessible color palettes using color wheel, in a variety of color variations and contrast levels. It will tell you automatically if two colors are not accessible

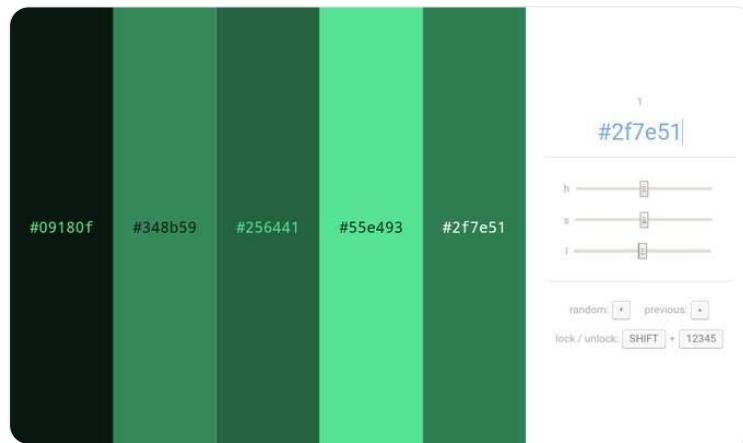
🔗 <https://color.adobe.com/create/color-accessibility>



3. Palette Ninja

Palette ninja is an online color scheme generator that allows you to create harmonious color schemes in seconds

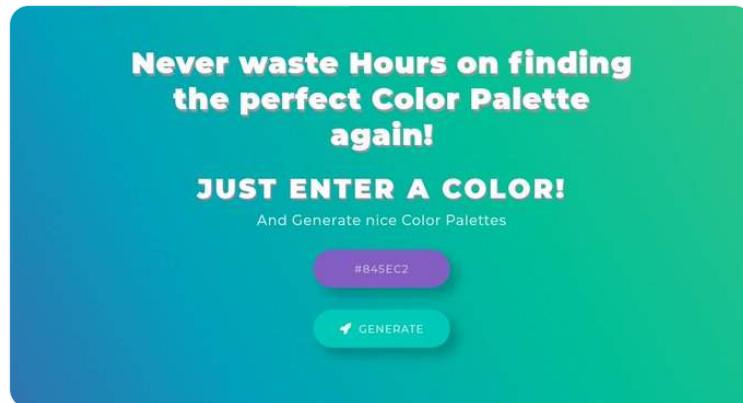
🔗



5. My color space

Here you can find the perfect matching color scheme for your next project! Generate nice color palettes, color gradients, and much more! Your space for everything that has to do with color

🔗 [mycolor.space](#)



5. Learn UI design

A tool for creating color variations on base color to meet WCAG AA or AAA color contrast ratio guidelines.

🔗 <https://learnui.design/tools/accessible-color-generator.html>

Show me the closest variations of #4ac4e2 that contrast against the color #fffff enough to meet AAA Guidelines AAA Guidelines

RESULTS

FOR LARGE/BOLD TEXT ⓘ
Try this combo instead:
 #00819d
 #fffff

FOR SMALL TEXT ⓘ
Try this combo instead:
 #005f7a
 #fffff

6. Magic pattern

Beautiful pure CSS background patterns that you can actually use in your projects. They are highly customizable as well.

🔗

MagicPattern

*Magical Visuals
With No Design Skills*

404 - MagicPattern page not found

Create Pro Visuals with MagicPattern. Generate SVG/CSS patterns, gradients and organic shapes to brand your product and social media posts.

<https://magicpattern.design/tools/css-backgrounds>

CSS Background Patterns

Beautiful pure CSS background patterns that you can actually use in your projects!

Created by @d_raptis

Product Hunt
#2 Product of the Day

Back Color:

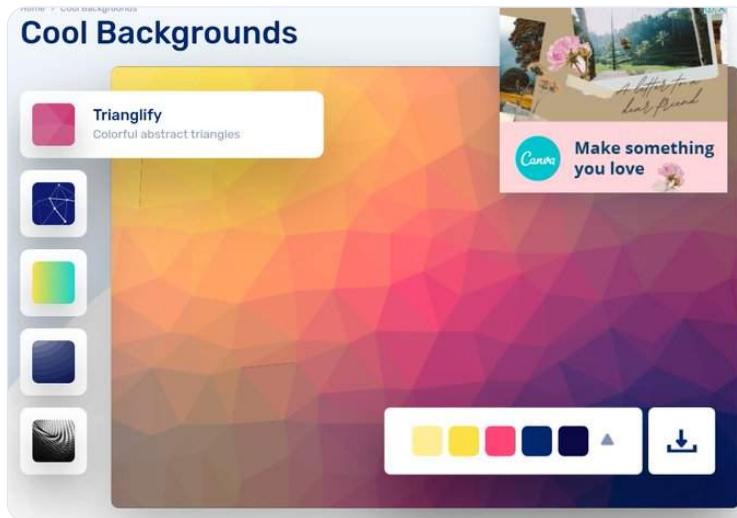
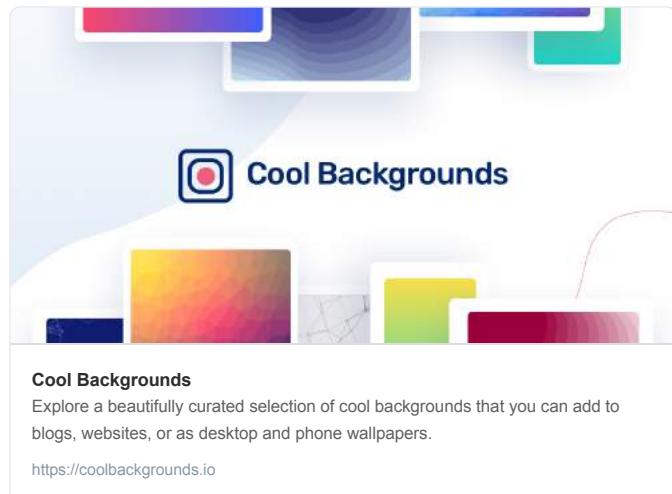
Front Color:

Opacity:

Spacing:

7. Cool backgrounds

Collection of cool backgrounds that you can add to blogs, websites, or as desktop and phone wallpapers



8. Gradients

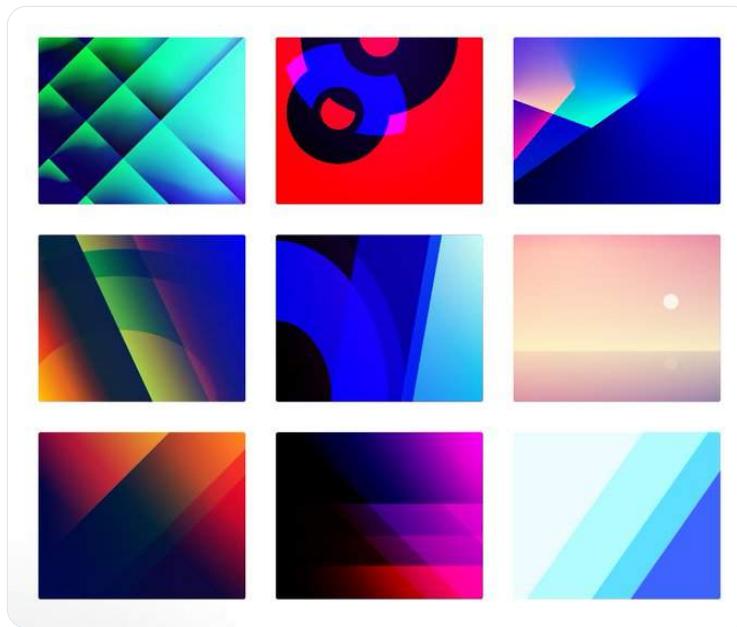
Use pure CSS gradient backgrounds for your next website or app, as a JPG image or CSS code, no attribute required





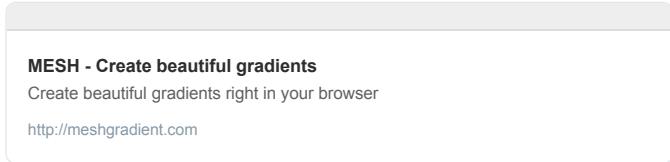
Gradienta
CSS Gradients With SVG Downloads

Multicolor CSS Gradients, JPG Downloads, 100% Free!
Use pure CSS gradient backgrounds for your next website or app, as a JPG image or CSS code, no attribute required!
<https://gradienta.io>

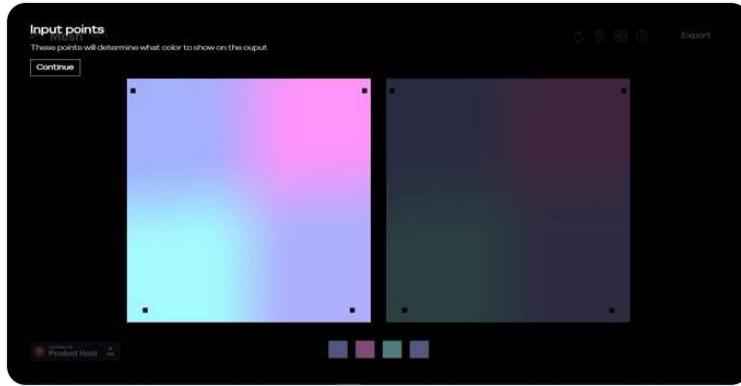


9. Mesh gradient

Create aurora UI like gradients using few simple clicks



MESH - Create beautiful gradients
Create beautiful gradients right in your browser
<http://meshgradient.com>



10. Stripes generator

Pure CSS Stripes Generator that you can use for backgrounds.

🔗 stripesgenerator.com

The screenshot shows the 'CSS Stripes Generator' interface. It includes fields for 'Angle' (set to 45), 'Maintain Ratio' (unchecked), and four stripe sections. Each section has 'Size' (22, 20, 20, 20) and 'Color' (hex codes #e11d74, #fffffe, #000000, #fbaa74). A preview window shows a pattern of alternating black, yellow, pink, and white stripes. Below the preview is a 'Generated CSS Code' panel containing:

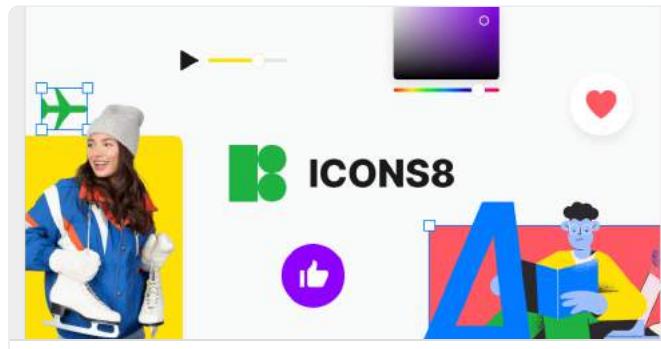
```
background-image: linear-gradient(45deg,
#e11d74 13.41%, #fffffe 13.41%, #fffffe
25.61%, #000000 25.61%, #000000 37.80%,
#faa749 37.80%, #faa749 50%, #e11d74 50%,
#e11d74 63.41%, #fffffe 63.41%, #fffffe
75.61%, #000000 75.61%, #000000 87.80%,
#faa749 87.80%, #faa749 100%);
```

Buttons for 'Copy' and 'Save' are at the bottom.

11. Icons 8

- Icons8 is just more than icons. You can download illustrations, vector images, music and much more

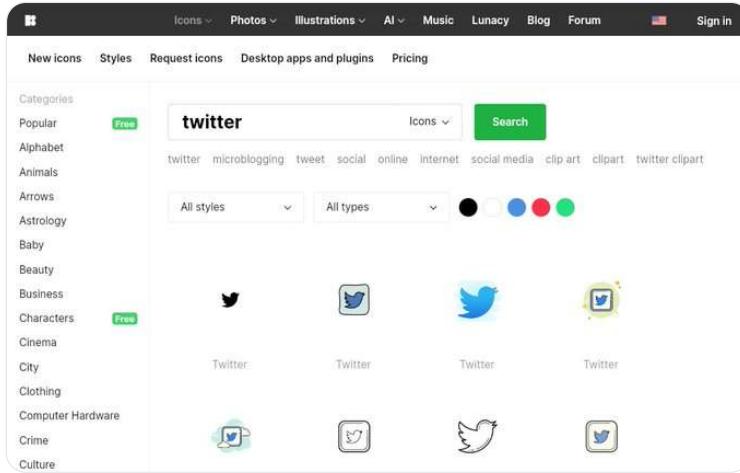
🔗



Free Icons, Clipart Illustrations, Photos, and Music

Download design elements for free: icons, photos, vector illustrations, and music for your videos. All the assets made by designers → consistent quality ⚡

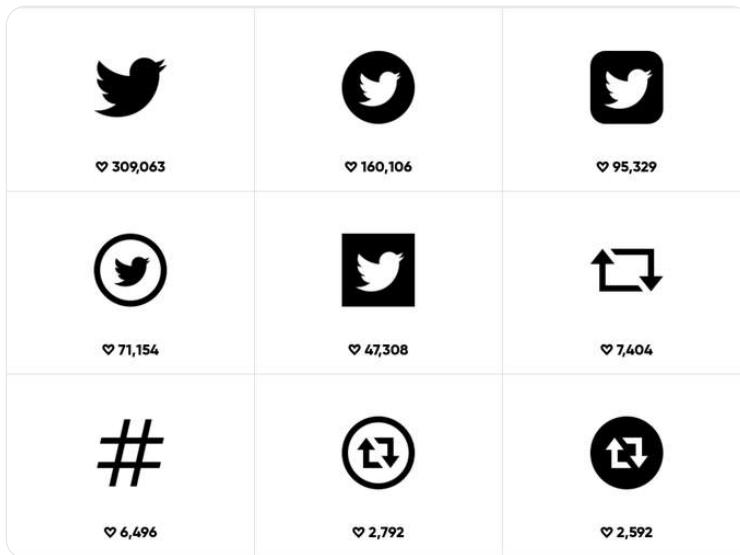
<https://icons8.com>



12. Icon Monstr

- Black and white themed minimal icons which look super great. You can also customize the thickness

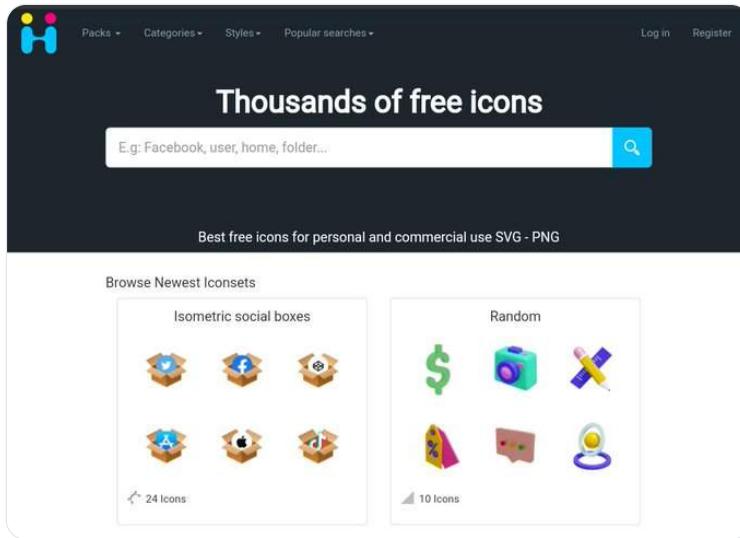
[🔗 iconmonstr.com](http://iconmonstr.com)



13. Icon Icons

- Over one thousand free icons which you can download as icons or images

[🔗 icon-icons.com](http://icon-icons.com)

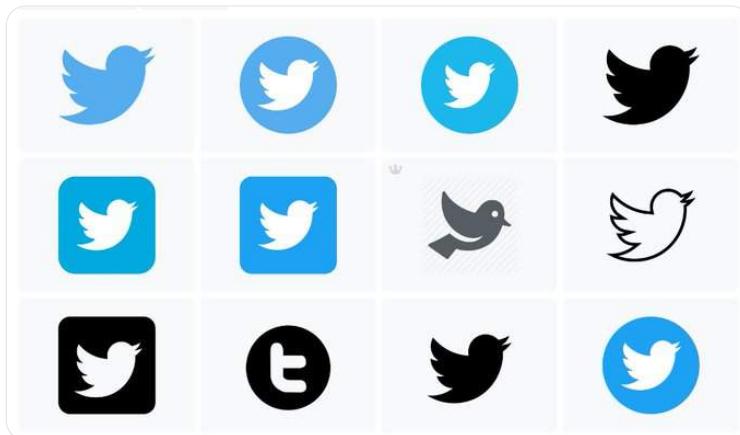


14. Icons finder

Filter through the world's largest marketplace for icons with flexibility and ease. Made up by submissions from top designers around the world, and curated by the team



A screenshot of the Iconfinder website. The header features the 'ICONFINDER' logo and the text '5 million vector icons'. Below the header is a search interface with filters like 'All prices', 'Free', 'Premium', 'All styles', 'Solid', 'semi Solid', 'Outline', 'Web icon', and 'More...'. A specific category 'Rocket icons' is selected, showing a grid of various rocket-related vector icons. Below the grid, the text '5,700,000+ free and premium vector icons - Iconfinder' and the description 'Iconfinder is the world's largest marketplace for vector and raster icons in SVG and PNG formats.' are displayed. At the bottom, the URL 'https://iconfinder.com' is provided.

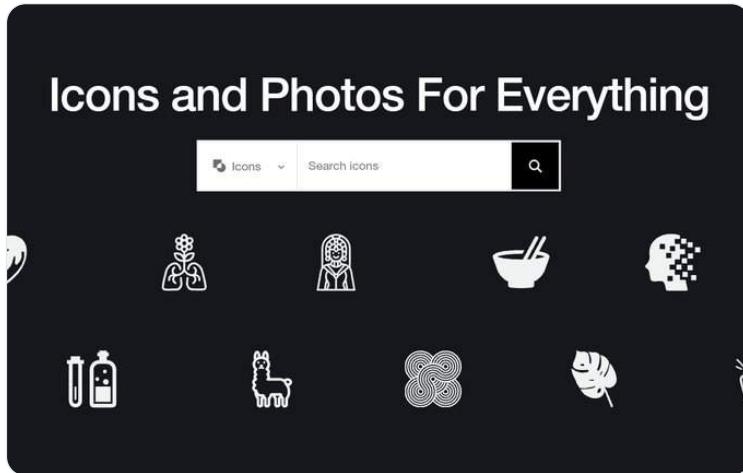


...

15. The Noun Project

- Search a noun and find an icon as simple as that

🔗 thenounproject.com



16. SEO Site Checkup

- SEO Site Checkup runs through a fast audit of your site, checking for proper tags and surfacing any errors that might come up

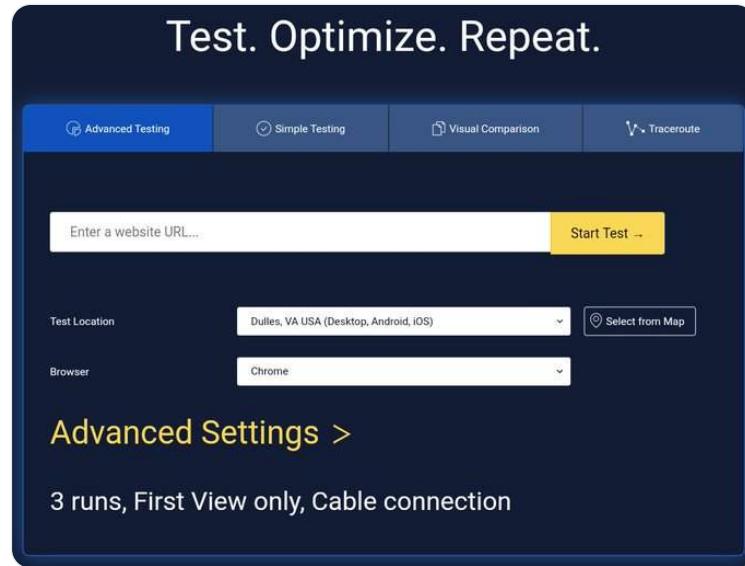
🔗 seositecheckup.com



17. Web Page Test

- Run a free website speed test from around the globe using real browsers at consumer connection speeds with detailed optimization recommendations.

🔗 webpagetest.org



18. GT metrix

- Easily test your web page speed



GTmetrix | Website Performance Testing and Monitoring
GTmetrix is a free tool to test and monitor your page's performance. Using Lighthouse, GTmetrix generates scores for your pages and offers actionable recommendations on how to optimize them.
<https://gtmetrix.com>

How fast does your website load?
Find out with GTmetrix

See how your site performs, reveal why it's slow and discover optimization opportunities.

Enter URL to Analyze...

Options: Vancouver, Canada using Chrome (Desktop) [Log in to change options](#)

19. Dareboost

- Website Speed Test and Website AnalysisTest, analyze and optimize your website performance

Run a Website Speed Test and get your analysis report for free



Quality and Performance report:
<https://dareboost.com> | [View report](#)

100% Perfect Congrats

0 Issues 0 Improvements 87 Business

See your priorities ▾

SIMULATED VISITOR: Chrome | Port: 4个 800ms Latency (Average) | 38% Metrics

Requests: 23 | Weight: 341 KB

First Byte: 0.39 sec | Start Render: 0.93 sec | Fully loaded: 1.59 sec

HTML: 13.5 KB | Scripts: 1.8 KB | Images: 240 KB | Other: 1.7 KB

Timeline: Unprefixed | More metrics | Timings & Video

Website Speed Test and Website Analysis – Free test | Dareboost
All-in-one service for website speed test, web performance monitoring and website analysis (speed, SEO, quality, security). Web performance has never been so easy.
<https://dareboost.com/en>

Website Speed Test and Website Analysis
Test, analyze and optimize your website performance

Run a Website Speed Test and get your analysis report for free

https://your-website.com Test my page

20. Pingdom

- Pingdom offers a cost and performance monitoring tool for your website



tools.pingdom.com

solarwinds pingdom Product ▾ Pricing Resources ▾ Support Tools ▾

Pingdom Website Speed Test

Enter a URL to test the page load time, analyze it, and find bottlenecks.

URL: Test from: START TEST

The Internet is fragile. Be the first to know when your site is in danger.

START YOUR FREE 14-DAY TRIAL

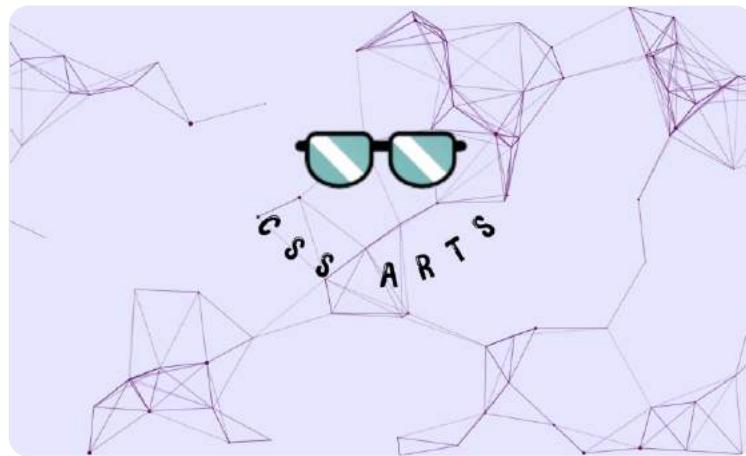
Pratham @Prathkum

22 Jun · 28 tweets · [Prathkum/status/1407411219758202885](#)

Tr

Are you planning to create your first CSS art for fun? Start with the basic shapes.

Here are 26 easy to hard shapes and figures you can try first



In general, analyze any image and try to think how can you make it using different shapes.

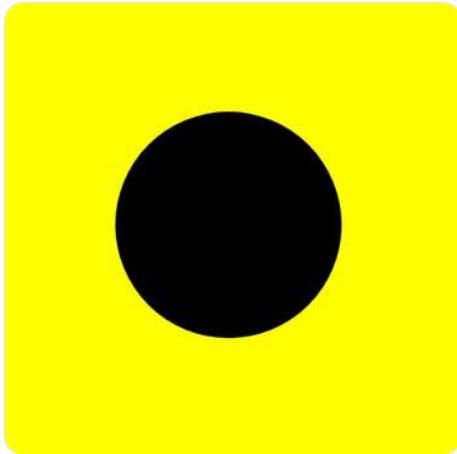
Two concepts that you need to master

1. border-radius
2. linear-gradient

1. Circle Yellow

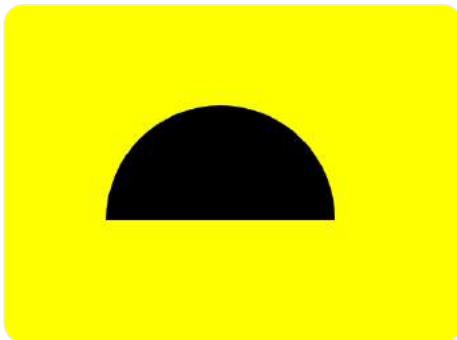
- Pretty simple, we just need to make a square and apply the border-radius 50% in order to give it a circular shape

```
● ○ ●  
.circle {  
    height: 200px;  
    width: 200px;  
    border-radius: 50%;  
    background-color: black;  
}
```



2. Semi-circle First 🌙

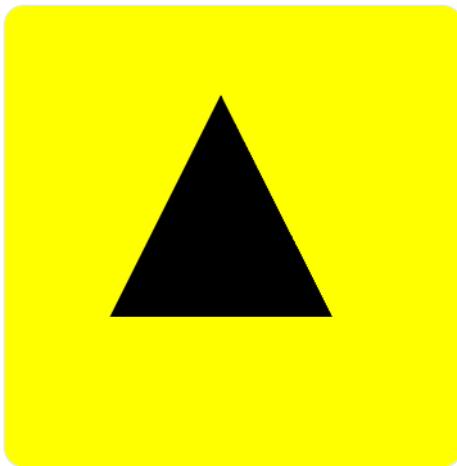
- Create a rectangle
- Apply border-radius top left and top right same as the height of the rectangle



```
● ● ●
.semi-circle {
  height: 100px;
  width: 200px;
  border-radius: 100px 100px 0 0;
  background-color: black;
}
```

3. Triangle ▲

- Creating a triangle is little bit tricky
- Set height and width as zero
- To make this, we draw a solid border and make the side border transparent



```
● ● ●
.triangle {
  width: 0;
  height: 0;
  border-bottom: 200px solid black;
  border-left: 100px solid transparent;
  border-right: 100px solid transparent;
}
```

4. Trapezium

- Same as a triangle but in this case we need to set some width



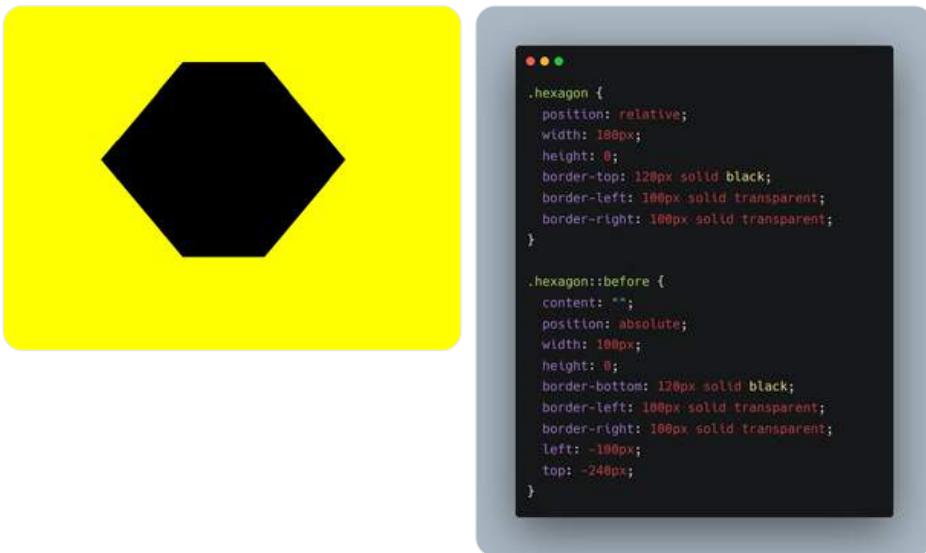
5. Parallelogram ■

- Create a rectangle
- Apply skew in order to tilt it



6. Hexagon ♦

- Creating a hexagon is very easy
- We need to make two trapeziums of the same size but make sure that the other trapezium should be upside down
- Align them perfectly



7. Drop 💧

- Create a square

- Apply 50% border-radius to all the sides except one side
- Rotate in such a manner so that tip comes to top



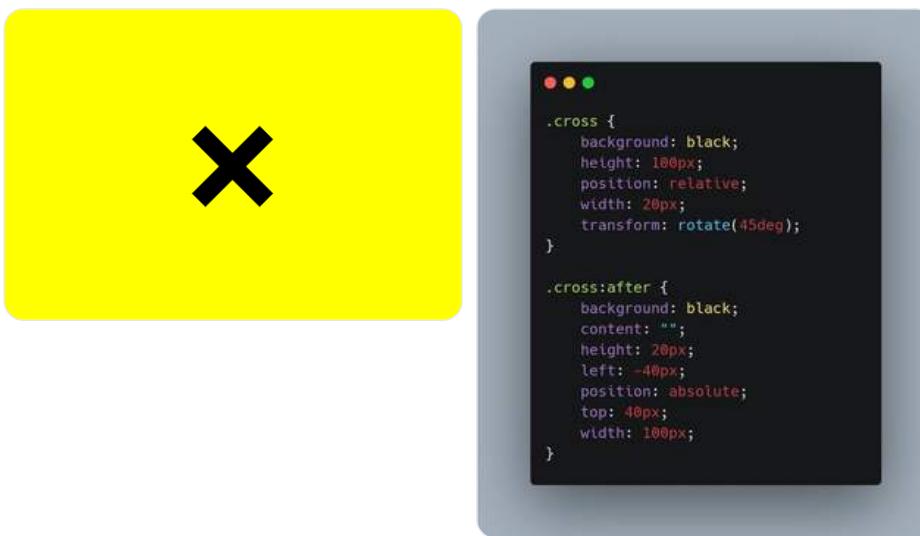
8. Diamond

- Combination of Trapezium and triangle



9. Cross

- Create two rectangles
- Place them over each other vertically and horizontally
- Rotate 45deg



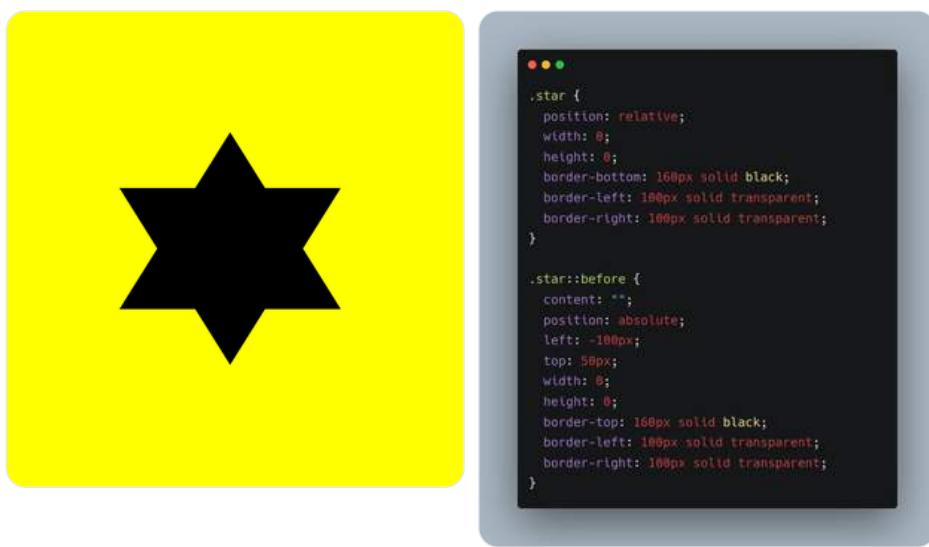
10. Egg 🥚

- Using an advanced border-radius technique



11. Star Glowing 🌟

- Create two triangle upside down
- Align the second triangle in the middle of the first one

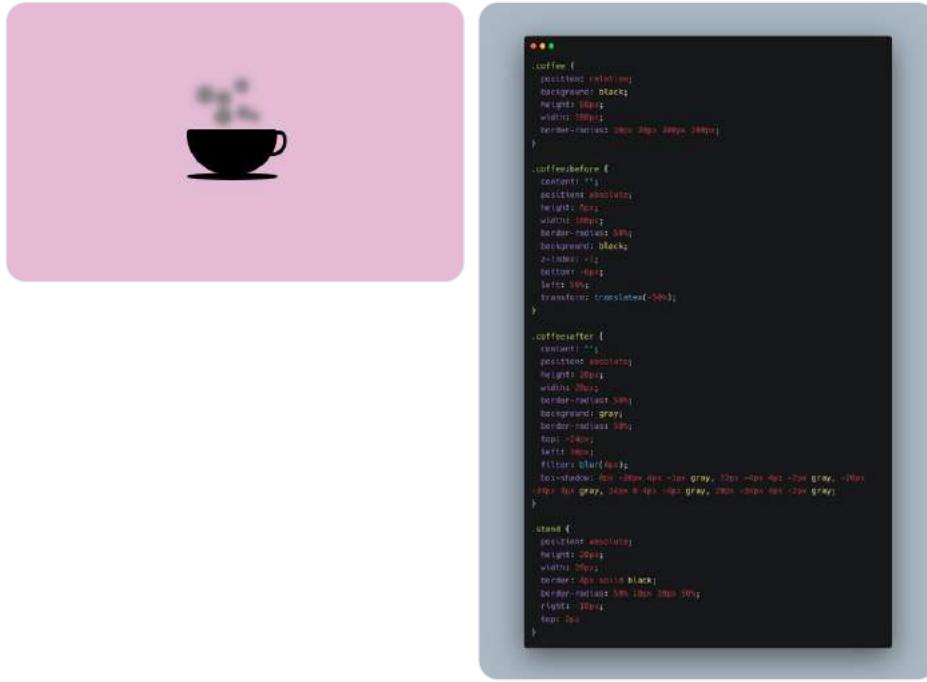


12. Moon 🌙

- Transparent background
- Apply box-shadow

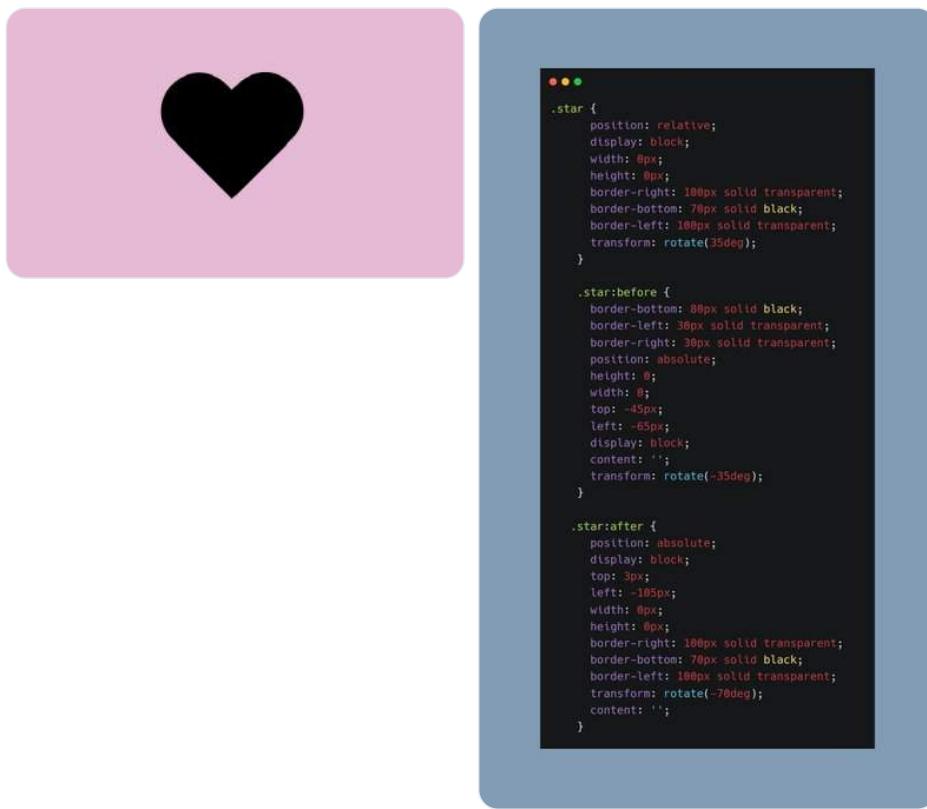


13. Coffee ☕



14. Heart ❤

- Two rectangles with top round border-radius



15. Infinity ∞

It may look tough but trust me I created this symbol using border-radius only



```
.infinity {  
    position: relative;  
}  
  
.infinity::before {  
    position: absolute;  
    height: 120px;  
    width: 120px;  
    border: 20px solid black;  
    content: "";  
    border-radius: 50% 50% 50% 50%;  
    transform: rotate(45deg);  
}  
  
.infinity::after {  
    position: absolute;  
    height: 120px;  
    width: 120px;  
    border: 20px solid black;  
    content: "";  
    border-radius: 50% 50% 50% 50%;  
    transform: rotate(-45deg);  
    left: -20px;  
}
```

16. Arrow ▶

Combination of the rectangle and a triangle



```
.arrow {  
    height: 48px;  
    width: 100px;  
    background: black;  
    position: relative;  
}  
  
.arrow::before {  
    position: absolute;  
    content: "";  
    border-top: 60px solid black;  
    border-right: 40px solid transparent;  
    border-left: 40px solid transparent;  
    transform: rotate(-90deg);  
    right: -60px;  
    top: -10px;  
}
```

17. Star ★

Combination of two triangles



```
.star {  
    position: relative;  
    display: block;  
    width: 0px;  
    height: 0px;  
    border-right: 100px solid transparent;  
    border-bottom: 70px solid black;  
    border-left: 100px solid transparent;  
    transform: rotate(35deg);  
}  
  
.star:before {  
    border-bottom: 80px solid black;  
    border-left: 36px solid transparent;  
    border-right: 30px solid transparent;  
    position: absolute;  
    height: 0;  
    width: 0;  
    top: -45px;  
    left: -65px;  
    display: block;  
    content: '';  
    transform: rotate(-35deg);  
}  
  
.star:after {  
    position: absolute;  
    display: block;  
    top: 3px;  
    left: -105px;  
    width: 0px;  
    height: 0px;  
    border-right: 100px solid transparent;  
    border-bottom: 70px solid black;  
    border-left: 100px solid transparent;  
    transform: rotate(-70deg);  
    content: '';  
}
```

18. Magnifying glass 🔎

You guessed it right! It's just a simple circle and small rectangle



```
.magnifying-glass {  
    height: 100px;  
    width: 100px;  
    border: 20px solid black;  
    border-radius: 50%;  
    position: relative;  
}  
  
.magnifying-glass::before {  
    height: 80px;  
    width: 20px;  
    background: black;  
    position: absolute;  
    content: '';  
    transform: rotate(-45deg);  
    right: -40px;  
    bottom: -60px;  
}
```

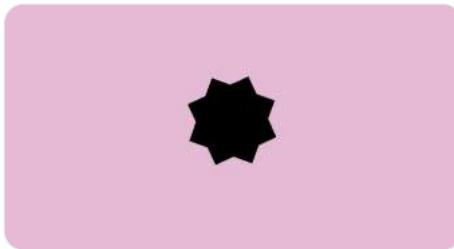
19. Pacman



```
.pacman {  
    width: 0px;  
    height: 0px;  
    border-right: 60px solid transparent;  
    border-top: 60px solid black;  
    border-left: 60px solid black;  
    border-bottom: 60px solid black;  
    border-top-left-radius: 60px;  
    border-top-right-radius: 60px;  
    border-bottom-left-radius: 60px;  
    border-bottom-right-radius: 60px;  
}
```

20. 8 Point ☀

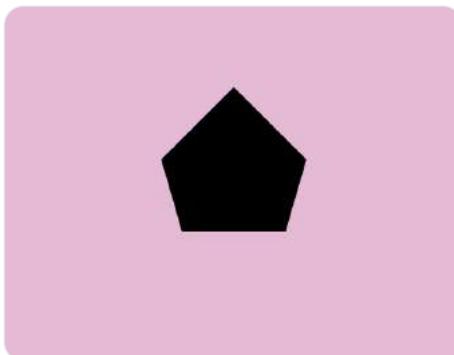
Two squares overlapping each other



```
.star-8 {
  background: black;
  width: 80px;
  height: 80px;
  position: relative;
  text-align: center;
  transform: rotate(20deg);
}
.star-8:before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  height: 80px;
  width: 80px;
  background: black;
  transform: rotate(135deg);
}
```

21. Pentagon ♦

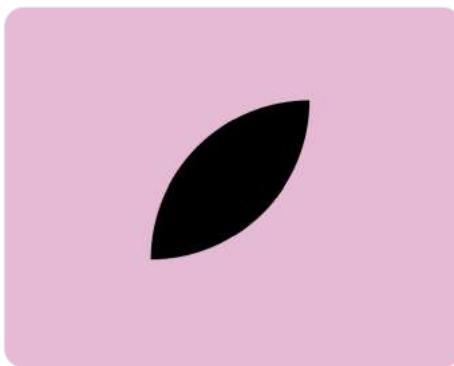
- Create a trapezium
- Create a triangle
- Merge them both



```
.pentagon {
  position: relative;
  width: 100px;
  border-left: 20px solid transparent;
  border-right: 20px solid transparent;
  border-top: 70px solid black;
}
.pentagon:before {
  content: "";
  position: absolute;
  width: 0;
  border-left: 70px solid transparent;
  border-right: 70px solid transparent;
  border-bottom: 70px solid black;
  top: -140px;
  left: -20px;
}
```

22. Leaf 🌱

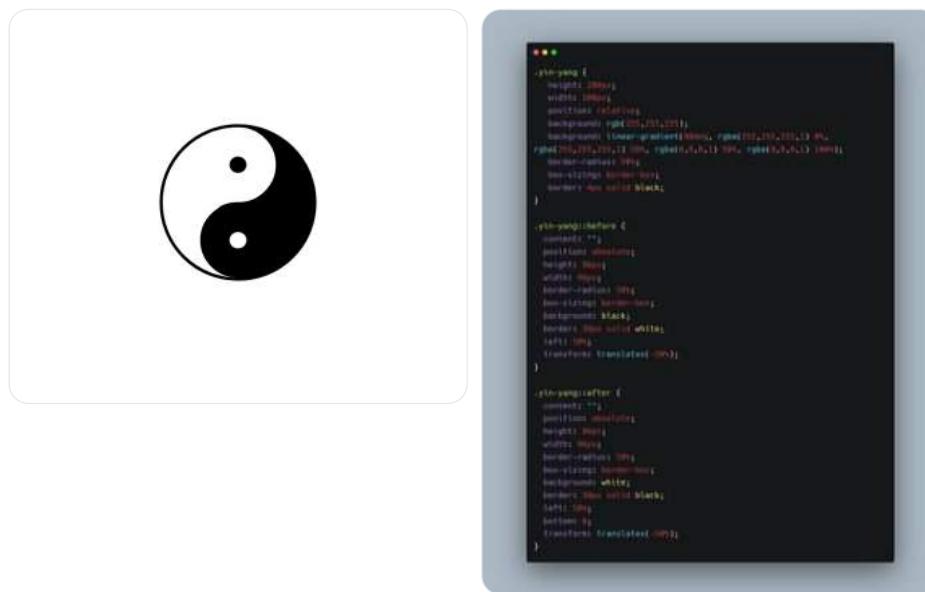
- Just the matter or border-radius



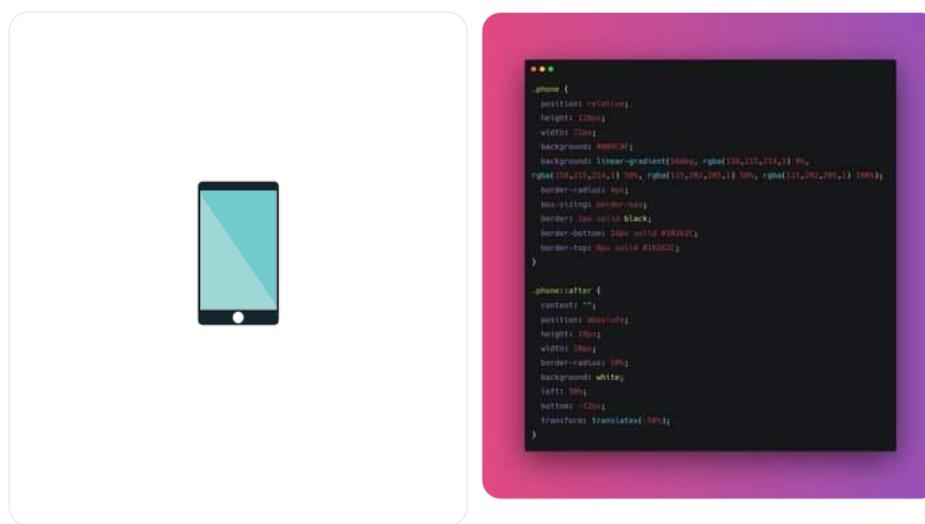
```
.leaf {
  height: 200px;
  width: 200px;
  background: black;
  border-radius: 200px 0 200px 0;
}
```

23. Yin and Yang

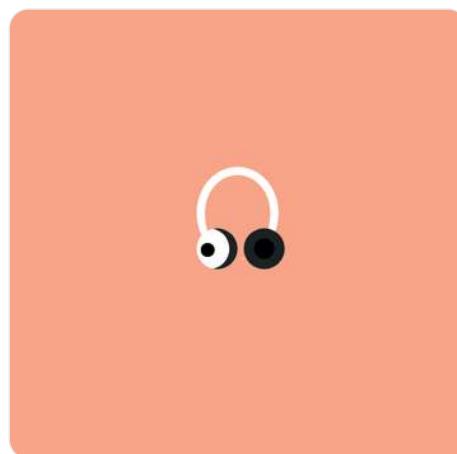
Looks tough but actually, it is not



24. Mobile Phone 📱



25. Headphones 🎧



```
headphones {  
    height: 10px;  
    width: 10px;  
    background-color: white;  
    border-radius: 50%;  
    position: relative;  
    overflow: hidden; -o-clip-path: polygon(50% 0%, 95% 45%, 50% 100%, 5%, 45%); clip-path: polygon(50% 0%, 95% 45%, 50% 100%, 5% 45%);  
}  
  
.headphones::after {  
    content: " ";  
    position: absolute;  
    height: 10px;  
    width: 10px;  
    background-color: black;  
    border-radius: 50%;  
    top: 10px;  
    left: 45px;  
}  
  
.headphones::before {  
    content: " ";  
    position: absolute;  
    height: 10px;  
    width: 10px;  
    border-radius: 50%;  
    background-color: white;  
    border: 1px solid white;  
    border-bottom: none;  
    z-index: 1;  
    top: -10px;  
}
```

26. Spectacles ☺



```
screen {  
    height: 10px;  
    width: 10px;  
    background-color: black;  
    position: relative;  
    border-radius: 50%;  
}  
  
.spectacles {  
    content: " ";  
    position: absolute;  
    height: 10px;  
    width: 10px;  
    background: linear-gradient(to top, rgba(10,100,100,1) 0%,  
        rgba(25,100,100,1) 40%, rgba(255,255,255,1) 40%, rgba(255,255,255,1) 60%,  
        rgba(25,100,100,1) 60%, rgba(255,255,255,1) 100%);  
    border: 1px solid black;  
    border-radius: 50%;  
    top: -5px;  
    left: 5px;  
}  
  
.spectacles::before {  
    content: " ";  
    position: absolute;  
    height: 10px;  
    width: 10px;  
    background: linear-gradient(to top, rgba(10,100,100,1) 0%,  
        rgba(25,100,100,1) 40%, rgba(255,255,255,1) 40%, rgba(255,255,255,1) 60%,  
        rgba(25,100,100,1) 60%, rgba(255,255,255,1) 100%);  
    border: 1px solid black;  
    border-radius: 50%;  
    top: -5px;  
    left: 5px;  
}
```

• • •



Pratham @Prathkum

23 Jun • 6 tweets • [Prathkum/status/1407585116403257345](https://twitter.com/Prathkum/status/1407585116403257345)

Tr

I again brought 5 awesome GitHub repositories for beginners



1. Project-based learning

A curated list of project-based tutorials covering JavaScript, Python and many other popular languages



tuvtran/project-based-learning
Curated list of project-based tutorials

64 Contributors | 24 Issues | 1 Discussion | 52k Stars | 8k Forks

tuvtran/project-based-learning
Curated list of project-based tutorials. Contribute to tuvtran/project-based-learning development by creating an account on GitHub.
<https://github.com/tuvtran/project-based-learning>

tuvtran / project-based-learning

Code Issues Pull requests Discussions Actions Wiki Security Insights

Project Based Learning

A list of programming tutorials in which learners build an application from scratch. These tutorials are divided into different primary programming languages. Some have intermediate technologies and languages.

To get started, simply fork this repo. Please refer to CONTRIBUTING.md for contribution guidelines.

Table of Contents:

- C
- C/C++
- ObjC

About

Curated list of project-based tutorials

Contributors

Releases

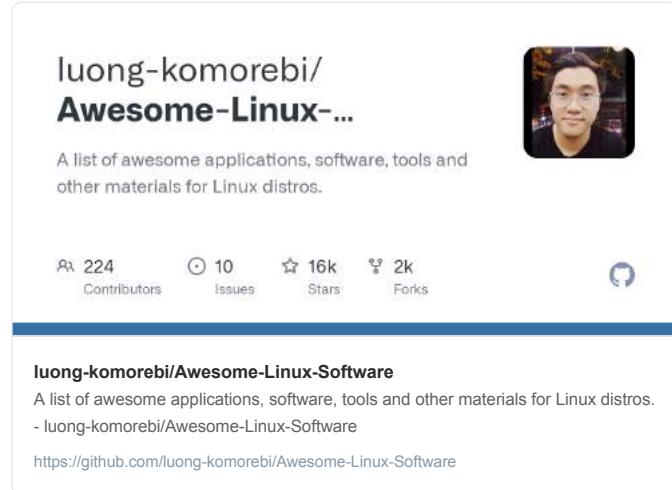
Packages

Contributors

2. Awesome Linux software

A list of awesome applications, software, tools, and other materials for Linux distros





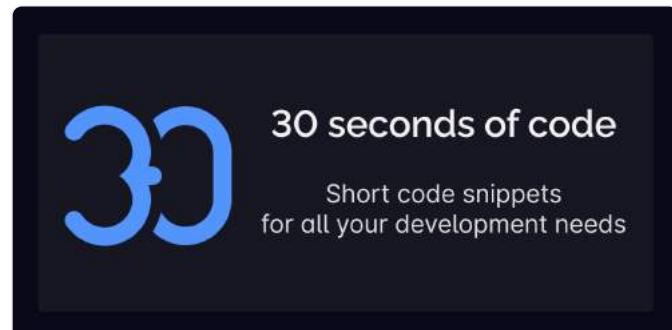
A screenshot of the GitHub repository code view for 'luong-komorebi/Awesome-Linux-Software'. The interface shows a list of commits in the master branch. The commits are as follows:

- luong-komorebi Merge pull request #547 from Fabiolalike/patch3 - 6 days ago
- luong-komorebi Test removing rdm - 2 years ago
- luong-komorebi Trigger grilla build - 16 months ago
- luong-komorebi Creating a separate Contributions guideline - 4 years ago
- luong-komorebi Add some links and sort alphabetically - last month
- luong-komorebi Update Arc theme link - last month
- luong-komorebi Update Arc theme link - last month
- luong-komorebi Update Arc theme link - last month
- luong-komorebi Update Arc theme link - last month
- luong-komorebi Set theme jdyjl-theme-dark - 16 months ago
- luong-komorebi README.md - 16 months ago

The right sidebar includes sections for 'About', 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Security', 'Insights', 'About', 'Releases', and 'Packages'.

3. 30 Seconds of React

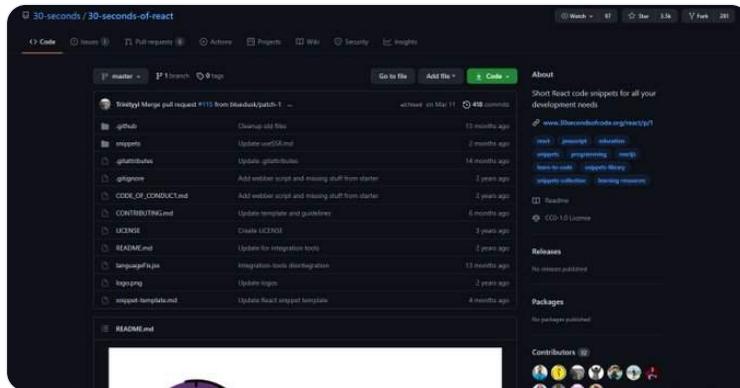
Short React code snippets for all your development needs



30-seconds/30-seconds-of-react

Short React code snippets for all your development needs - 30-seconds/30-seconds-of-react

<https://github.com/30-seconds/30-seconds-of-react>



4. 50 Projects

50+ mini web projects using HTML, CSS & JS



**bradtraversy/
50projects50days**



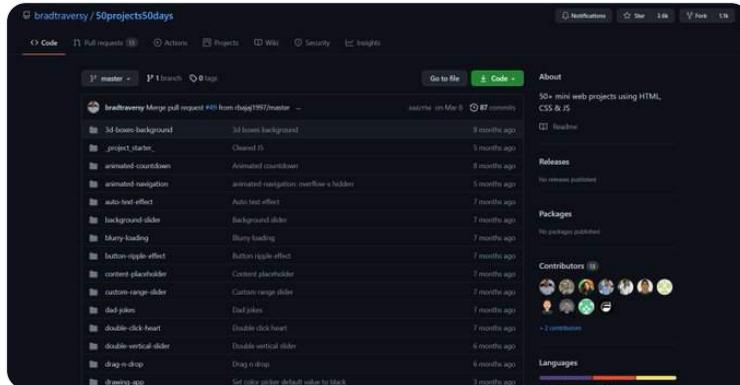
50+ mini web projects using HTML, CSS & JS

14 Contributors | 0 Issues | 5k Stars | 1k Forks

bradtraversy/50projects50days

50+ mini web projects using HTML, CSS & JS. Contribute to bradtraversy/50projects50days development by creating an account on GitHub.

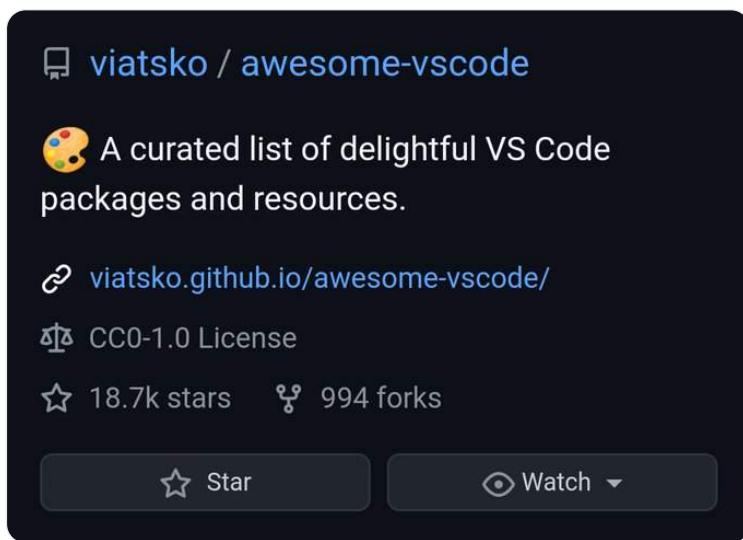
<https://github.com/bradtraversy/50projects50days>



5. Awesome VS code

A curated list of delightful VS Code packages and resources





...



Pratham @Prathkum

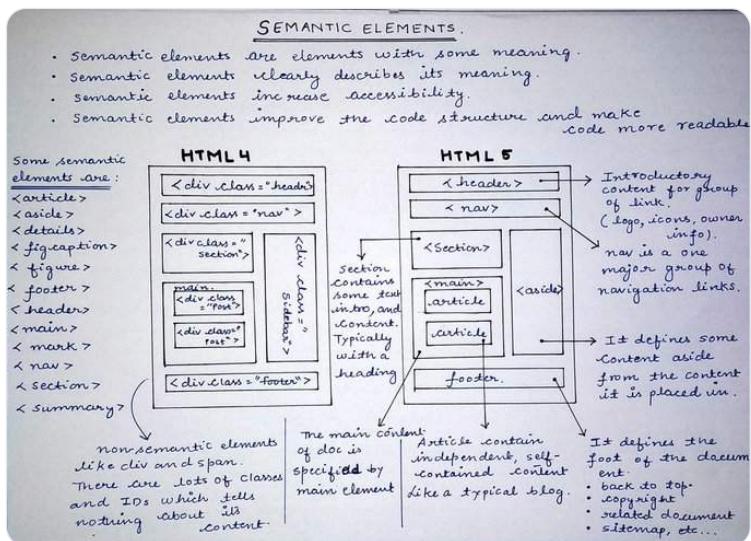
23 Jun · 13 tweets · [Prathkum/status/1407795508551294982](https://twitter.com/Prathkum/status/1407795508551294982)

Tr

I have created enormous handmade notes for Web development beginners.

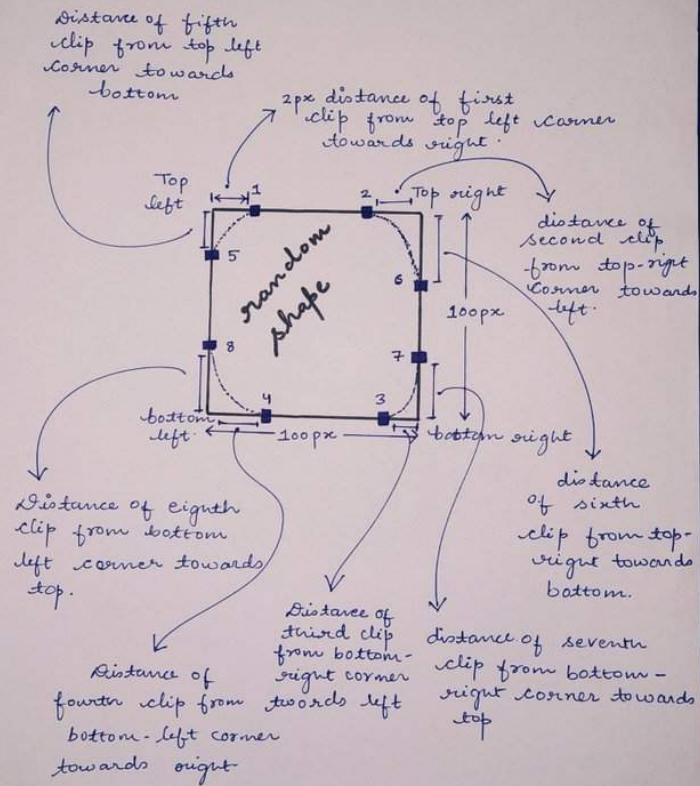
Here are some of my handmade JavaScript and CSS notes/cheat sheets that can help you 

0. Semantic HTML

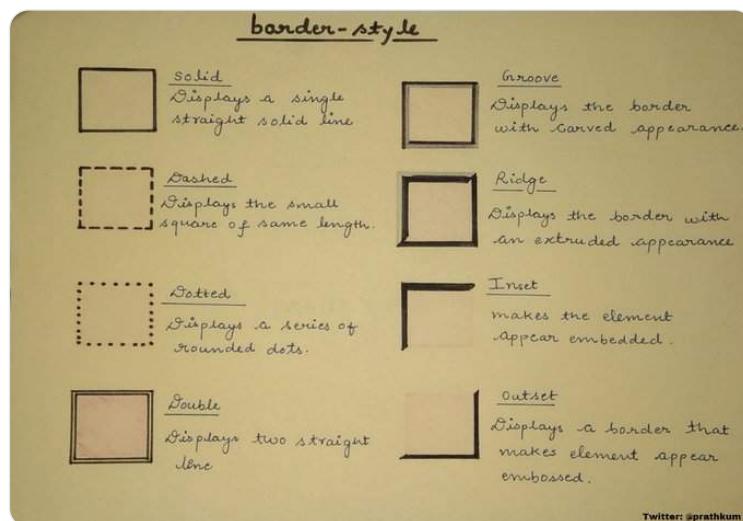


2. CSS border-radius

border-radius: 1 2 3 4 / 5 6 7 8 & clips position

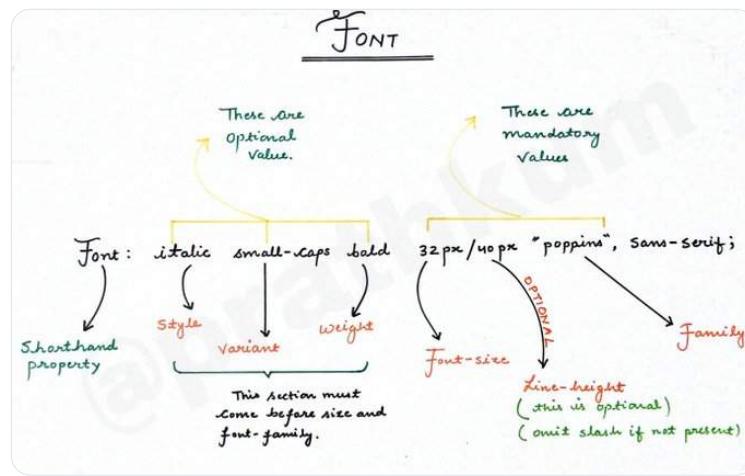


3. Border-style

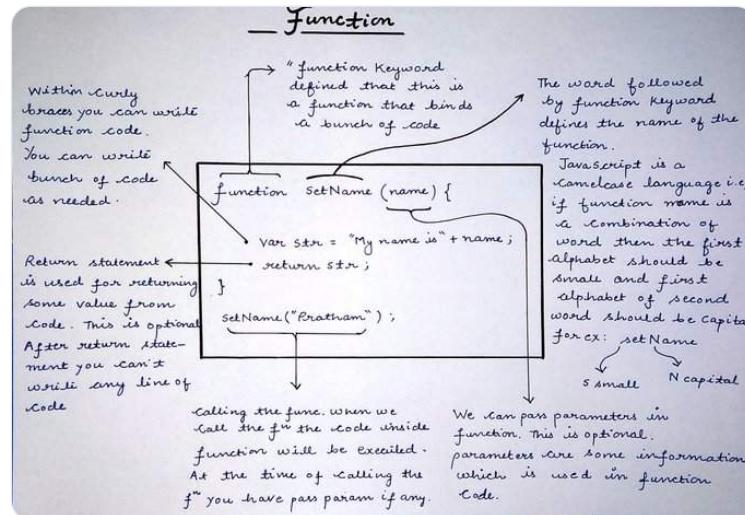


Twitter: @prathkum

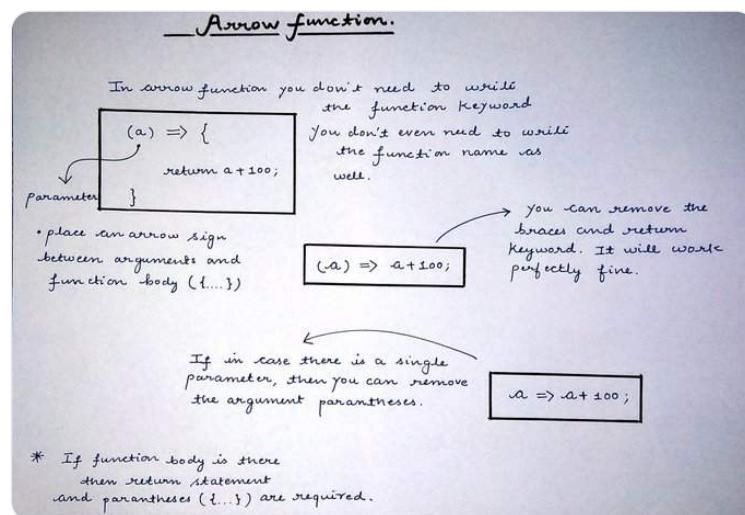
4. Font shorthand property



5. JavaScript Function Anatomy



6. JavaScript Arrow Function



7. JavaScript Array Methods

```

[1,2,3].map(x=>x*x) // [2,4,6]
[1,2,3].filter(x=>x<2) // [1]
[1,2,3].indexOf(2) // 1
[1,2,3].reduce((x,y)=>x*y) // 6
[1,2,3].reverse() // [3,2,1]

```

```

['P','R'].concat(['A']) // ['P','R','A']
['P','R','A'].slice(1) // ['R','A']
['P','A','T'].splice(1,0,'R') // ['P','R','A','T']
['P','R'].join('-') // 'P-R'

```

Array Methods

```

[1,2,3,4].fill(0,2,4) // [1,2,0,0]
[1,2,3,4].find(x=>x>3) // 4
[1,2,3].findIndex(x=>x>1) // 1
[1,2,3].includes(a) // true
[1,2,3].some(x=>x*x==0) // true

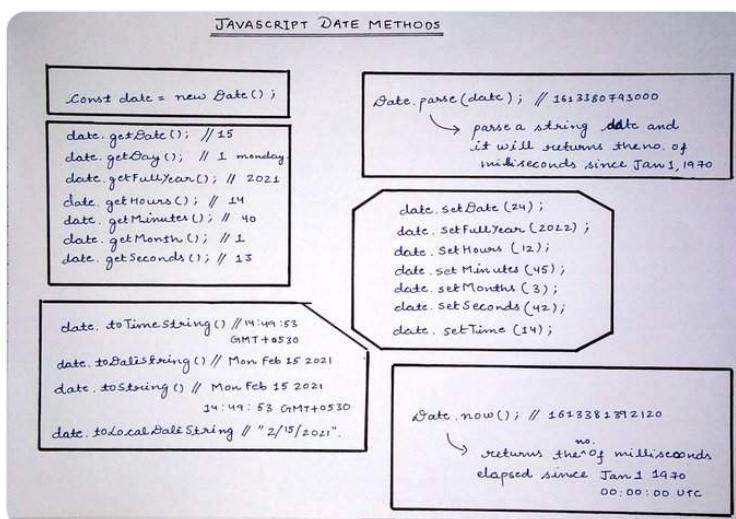
```

```

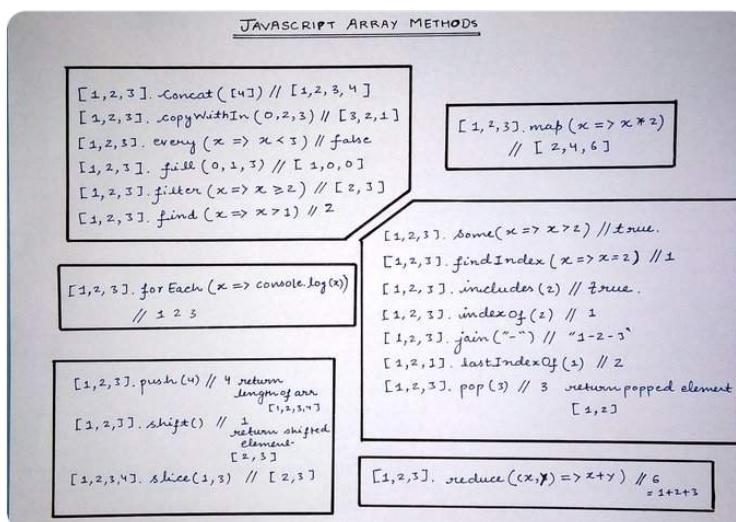
[1,2,3].forEach(x=> console.log(x)) // 1,2,3
[1,2].push(3) // [1,2,3]
[1,2,3].pop() // [1,2]
[1,2,3].shift() // [2,3]
[1,2,3].unshift(0) // [0,1,2,3]
[1,2,3].every(x=>x<5) // true

```

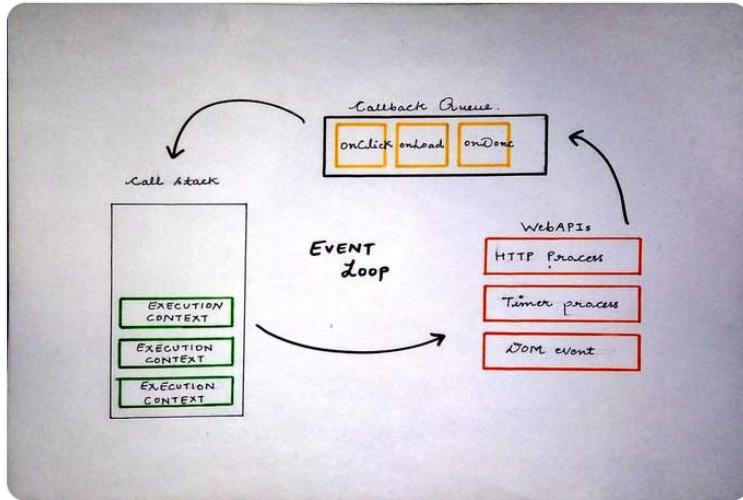
8. JavaScript Date Methods



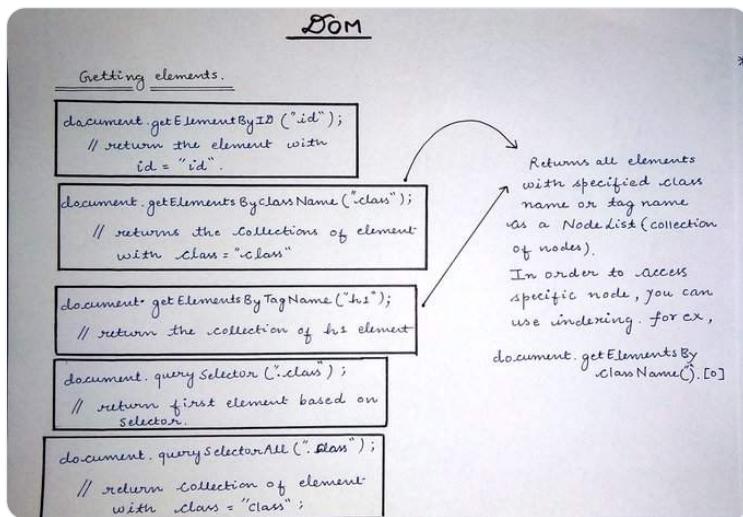
9. JavaScript Array Methods



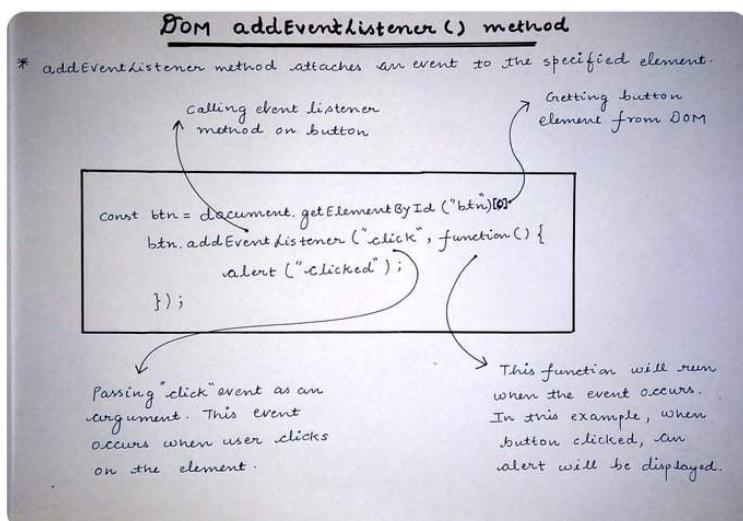
10. Event Loop



11. JavaScript DOM Methods



12. DOM addEventListener Method



• • •

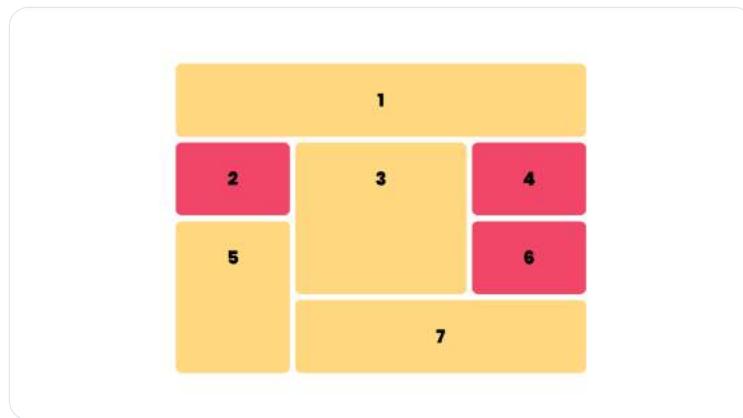


Pratham @Prathkum

25 Jun · 28 tweets · [Prathkum/status/1408352178364981250](#)

Tr

Next 27 tweets are the Complete Introduction to CSS Grid Layouts including everything



Grid is used for making complex web design layouts more easily as it's not so hard to master

Using Flex you can make only 1D layout but Grid gives you the full power of creating a 2D layout

Let's start

{ 02 / 28 }

First things first, start with giving the display property "grid" to the container element or parent element.

{ 03 / 28 }



Nothing will change after adding `display: flex`; in the parent container because we need to define the width of columns. In order to set that column's width we have `grid-template-columns` property.

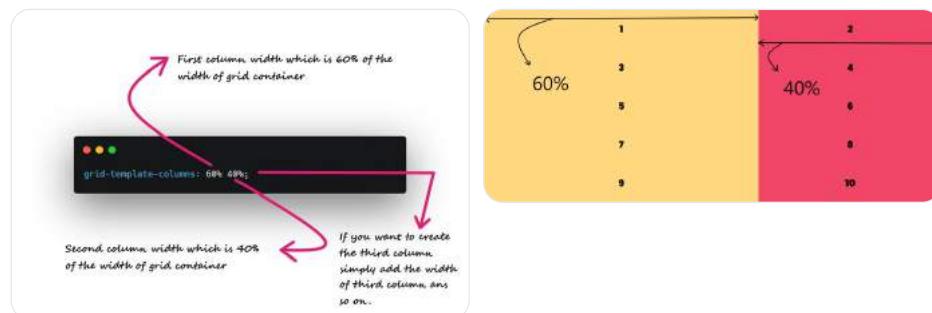
{ 04 / 28 }

Let's start with defining the width of our columns.

For example, let's say I need two columns of width 60% and 40% respectively

`grid-template-columns: 60% 40%;`

{ 05 / 28 }

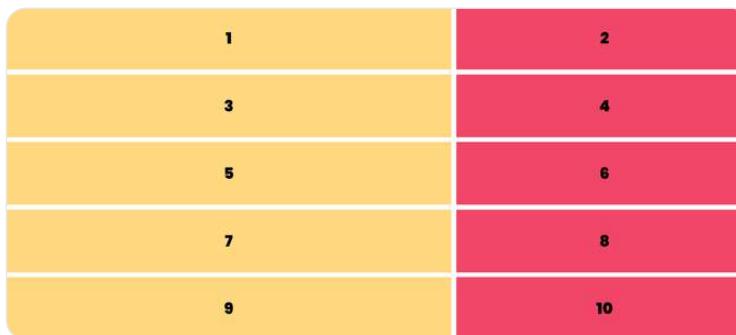


Ahh!! My grid items look ugly as there is no spacing between them.

Here "grid-gap" property comes into play. For example, I need 10px spacing along column and row

`grid-gap: 10px;`

{ 06 / 28 }



Similarly we have `grid-template-rows`.

It is used to define the number of rows and height of rows.

`grid-template-rows: 200px 400px;`

{ 07 / 28 }

```
.container {
  grid-template-columns: 200px 200px 200px 200px 200px;
}
```

As you can see there is a lot of repeated code in
grid-template-columns: 200px 200px 200px 200px 200px;

Instead of this, we can use the repeat function

grid-template-columns: repeat(5, 200px);

{ 08 / 28 }

First param defines the number of repetition

Second param defines "what to repeat"

Hence it is equivalent to
grid-template-columns: 200px 200px 200px 200px 200px;

You might run into some responsiveness issues if you pass pixel unit or percentage in your grid-template-columns

In order to prevent this, it is recommended to use fraction values

For example:

{ 09 / 28 }

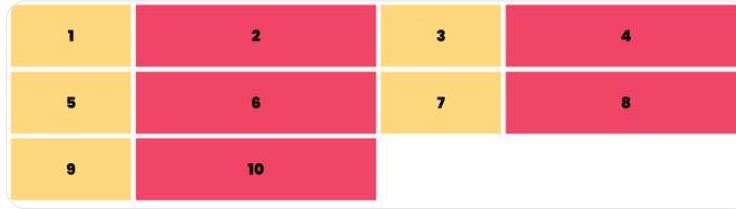
```
grid-template-columns: 1fr 2fr 1fr;
```

You can use repeat function for fr as well

repeat(2, 1fr 2fr);

It will repeat 1fr 2fr two times.

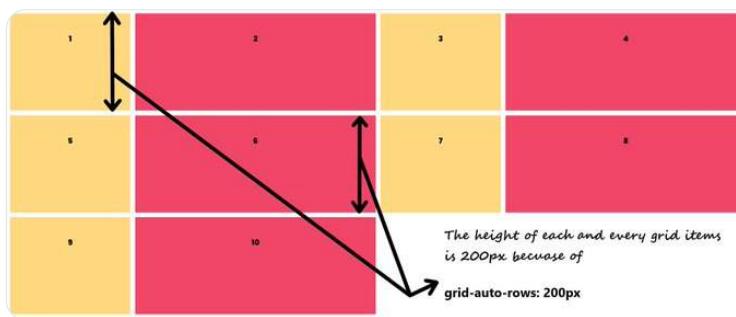
{ 10 / 28 }



Alright moving forward, you can set the height of the grid element using grid-auto-rows

For ex, grid-auto-rows: 200px;

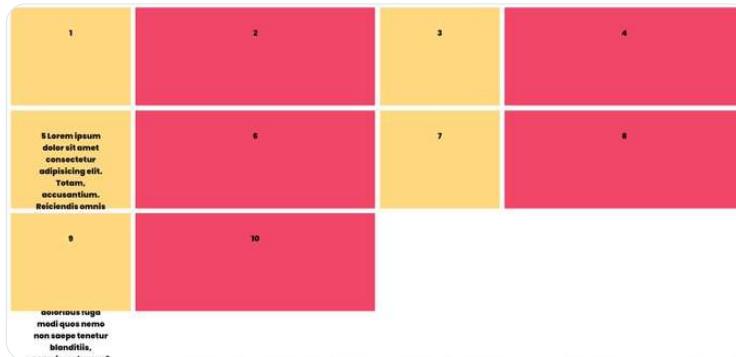
{ 11 / 28 }



Though there is a problem. By doing this, we are setting the fixed height so content inside items can be overflow.

For example:

{ 12 / 28 }

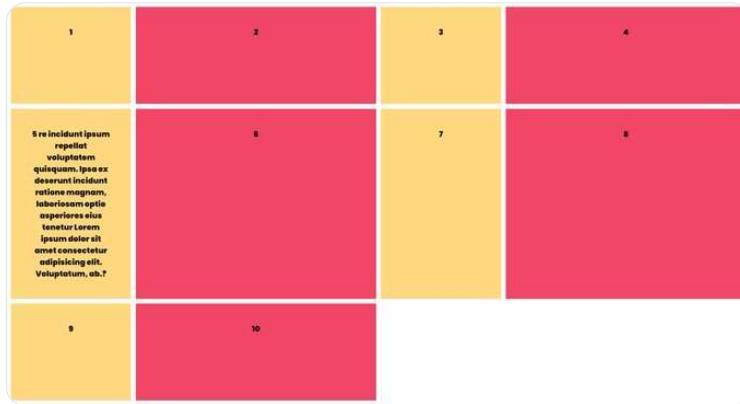


In order to prevent this kind of issues we have minmax function

grid-auto-rows: minmax(200px, auto);

It's pretty intuitive that the height of grid items will be 200px minimum and "auto" maximum(according to content)

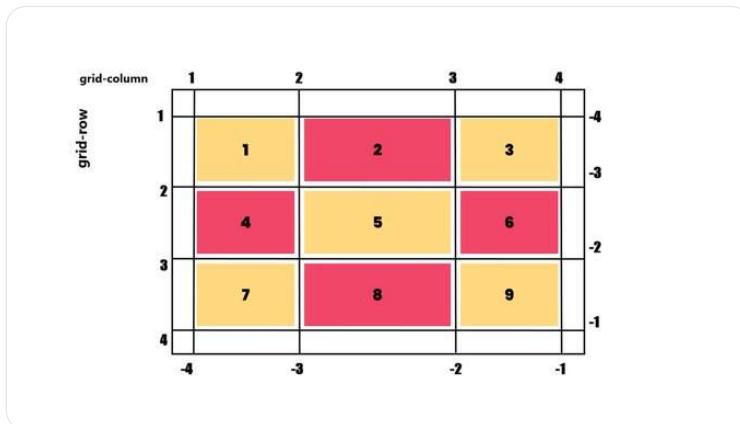
{ 13 / 28 }



Well, all that we have covered so far we can do that using flexbox also.

Let's understand the 2 dimensions of grid layout

{ 14 / 28 }

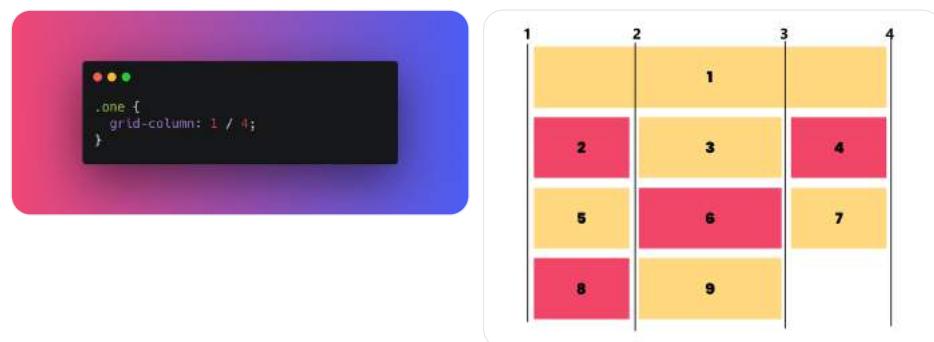


We can change the position of a particular item in accordance of row and column

For example, I want my first item to take up entire row, that is from the first column to the last column

grid-column: 1 / 4;

{ 15 / 28 }



Alright moving forward, The next property we have is `grid-template-areas` which specifies the areas within the grid layout.

Each row is defined by apostrophes ('')

Sounds confusing? Let see this in action

{ 16 / 28 }

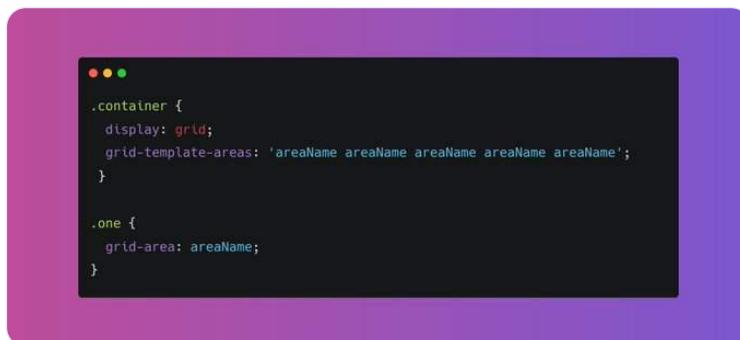
Let's breakdown this code

```
grid-template-areas: 'areaName areaName areaName areaName areaName';
```

- Each row is defined by apostrophes ('')
- Hence only one row in this case as there is only one pair of apostrophes
- Five "areaName" hence five columns

CONT...

{ 17 / 28 }



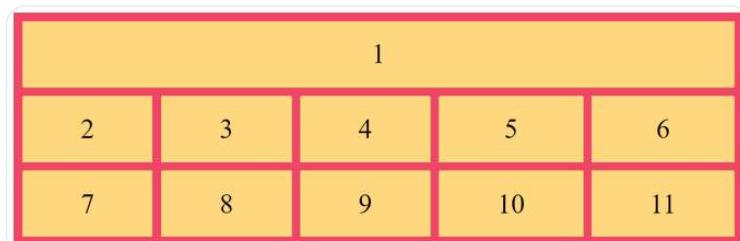
Now let's apply this area (areaName) to the first grid item

```
.one {  
  grid-area: areaName;  
}
```

As you can see item1 takes one entire row and 5 columns. As simple as that

<https://codepen.io/prathkum/pen/LYxVjGd>

{ 18 / 28 }



Alright, moving further let's talk about in the context of Grid items.

We can set the ordering and alignment of a particular item. Let's dive into it

{ 19 / 28 }

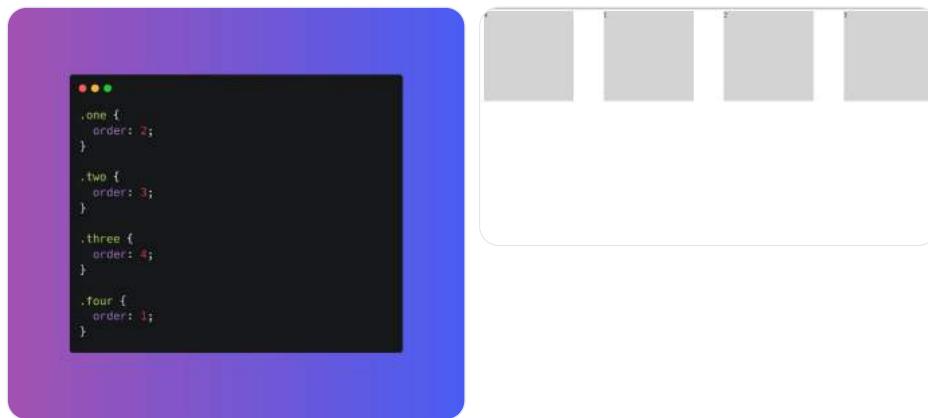
📌 The Order of the items

The order is nothing but used to set the order of an item within the grid container

{ 20 / 28 }

Here is the visual example, by default the correct order should be 1 2 3 4 but I have applied order property to change the actual ordering.

{ 21 / 28 }



📌 Alignment in CSS Grid Layout

This is a little confusing thing but we will cover everything in this thread.

justify-content
align-items
justify-self
align-self

{ 22 / 28 }

The `justify-content` is used to align the container's items when the items do not use all available space on the main-axis (horizontally).

🔗 https://www.w3schools.com/cssref/playit.asp?filename=playcss_justify-content&preval=flex-start

{ 23 / 28 }

CSS Property:

justify-content:

- flex-start
- flex-end
- center
- space-between
- space-around
- space-evenly
- initial

Result:

The `align-items` property specifies the default alignment for items inside the flexible container.

🔗 https://www.w3schools.com/cssref/playit.asp?filename=playcss_align-items&preval=center

{ 24 / 28 }

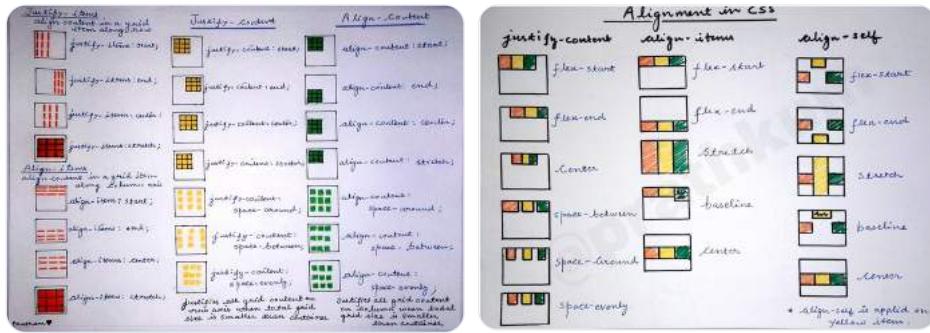
CSS Property:

- flex-start
- flex-end
- stretch
- baseline
- center
- initial

Result:

We can use justify-self and align-self in order to set the alignment of a particular item with the grid container

{ 25 / 28 }



Let's tackle the trickiest part:

How to center horizontally and vertically using one line of CSS?

It's simple using only one line of code

place-content: center;

{ 26 / 28 }



Try to play around with code here. This might be a little confusing in the beginning but once you get used to it, it all becomes pretty easy

<https://codepen.io/prathkum/pen/YzpeNWV>

{ 27 / 28 }

I think that was pretty much it, If you like this thread share it with your connections



Peace out 😊

{ 28 / 28 }

* Improved version of previous thread

• • •



Pratham @Prathkum

1 Jul · 8 tweets · [Prathkum/status/1410516717764218880](https://twitter.com/Prathkum/status/1410516717764218880)

Tr

7 GitHub repositories that can help you if you're learning Web Development



1. HTML reference

- A free guide to all HTML5 elements and attributes.



jgthms/html-reference



HTML Reference: a free guide to all HTML5 elements and attributes

1 Contributor 13 Issues 749 Stars 105 Forks

<https://github.com/jgthms/html-reference>





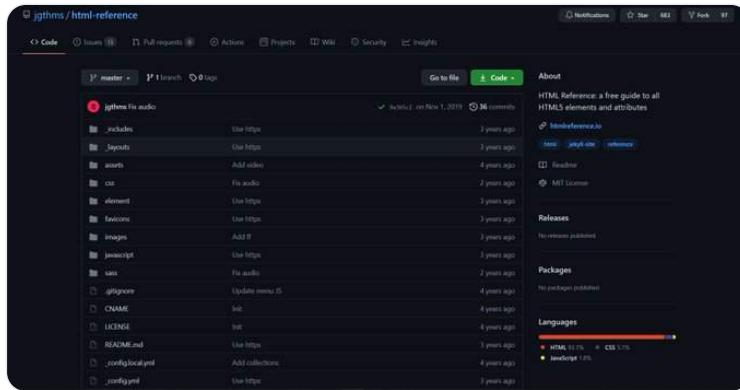
htmlreference.io

Free guide to all **HTML5** elements and attributes

HTML Reference

A free guide to all HTML elements and attributes.

<https://htmlreference.io>



2. HTML and CSS Code Guide

- Standards for developing consistent, flexible, and sustainable HTML and CSS.



Code Guide
Standards for developing consistent, flexible, and sustainable HTML and CSS.

mdo/code-guide
Standards for developing consistent, flexible, and sustainable HTML and CSS. -
mdo/code-guide
<https://github.com/mdo/code-guide>



[codeguide.co](#)

Code Guide
Standards for developing consistent, flexible, and sustainable HTML and CSS.

mdo/code-guide
Standards for developing consistent, flexible, and sustainable HTML and CSS. -
mdo/code-guide
<https://github.com/mdo/code-guide>

3. CSS Tips

- A collection of tips to help take your CSS skills pro



AllThingsSmitty/css-
protips



A collection of tips to help take your CSS skills pro

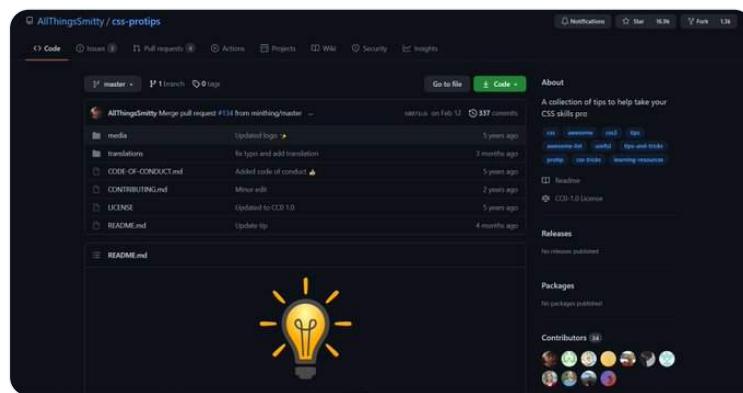
34 Contributors 4 Issues 19k Stars 1k Forks



AllThingsSmitty/css-protips

A collection of tips to help take your CSS skills pro - AllThingsSmitty/css-protips

<https://github.com/AllThingsSmitty/css-protips>



4. JavaScript Best Practice

- Comprehensive and exhaustive JavaScript & Node.js testing best practices



goldbergoni/javascript- testing-best-practices



Comprehensive and exhaustive JavaScript & Node.js testing best practices (June 2021)

53 Contributors 23 Issues 12k Stars 1k Forks

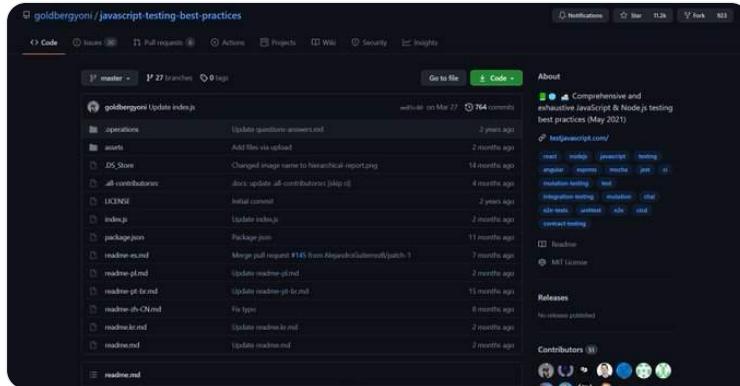


goldbergoni/javascript-testing-best-practices

Comprehensive and exhaustive JavaScript & Node.js testing best practices (June 2021) - goldbergoni/javascript-testing-best-practices

<https://github.com/goldbergoni/javascript-testing-best-practices>

 testjavascript.com



5. 50 Projects

- 50+ mini web projects using HTML, CSS & JS



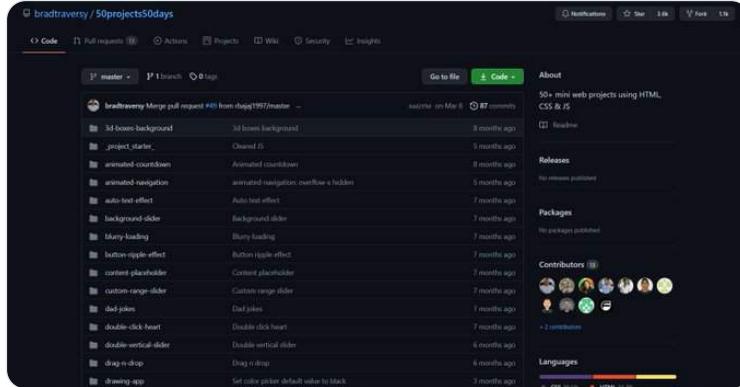
**bradtraversy/
50projects50days**



50+ mini web projects using HTML, CSS & JS

14 Contributors 0 Issues 5k Stars 1k Forks

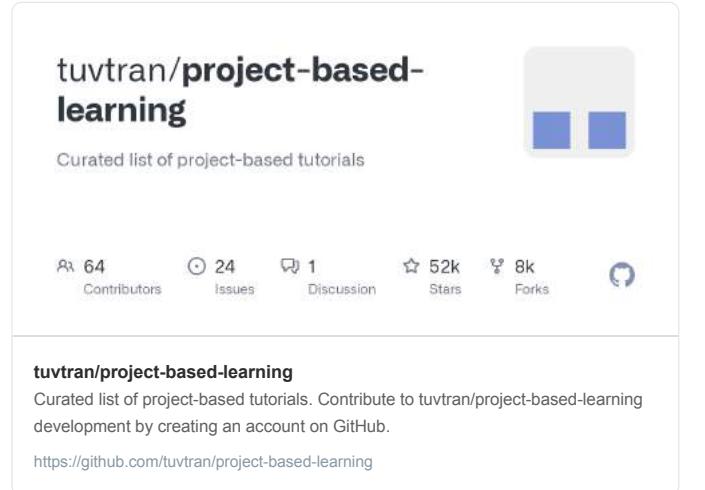
bradtraversy/50projects50days
50+ mini web projects using HTML, CSS & JS. Contribute to bradtraversy/50projects50days development by creating an account on GitHub.
<https://github.com/bradtraversy/50projects50days>



6. Project-based learning

- Curated list of project-based tutorials





This screenshot shows the GitHub repository 'tuvtran/project-based-learning' in dark mode. It displays the repository's contents, including files like 'ghpages', 'LICENSE.md', and 'README.md'. The 'README.md' file contains a section titled 'Project Based Learning' with a description of the curated list of tutorials. The right sidebar shows the repository's 'About' section, which includes links for 'contributors', 'releases', and 'packages'.

7. Microsoft web dev course

- This is a super cool GitHub repo by Microsoft for learning web development for beginners

This screenshot shows the GitHub repository 'microsoft/Web-Dev-For-Beginners'. The repository has a large, colorful graphic background featuring overlapping triangles in yellow, orange, blue, and red. The title 'Web Development for Beginners' is prominently displayed in white text on a dark blue background within the graphic. Below the graphic, the repository's name is listed as 'microsoft/Web-Dev-For-Beginners', followed by a brief description: '24 Lessons, 12 Weeks, Get Started as a Web Developer - microsoft/Web-Dev-For-Beginners'. A link to the repository is provided at the bottom: <https://github.com/microsoft/Web-Dev-For-Beginners>.

Search or jump to... Pull requests Issues Marketplace Explore

microsoft / Web-Dev-For-Beginners

Code Issues Pull requests Actions Projects Wiki Security Insights

Use this template

1 main · 1 branch · 0 tags · Go to file · Add file · + Code · About · Use this template

1 pull request for translation files

commit management · ✓ written yesterday · 72 commits · last month

1 getting-started-lessons · fixing path for translation files · yesterday

2 j-p-basics · fix Markdown translation links in jc-basics lesson · 12 days ago

3-terminus · 🚧 WIP · 21 days ago

4-typing-game · adding localized json keys for Italian translation · 16 days ago

5-browser-extension · merge branch: main into loc-query-string-6 · 15 days ago

6-4-track-game · Update README.md · 13 days ago

7-book-project · adding localized json keys for Italian translation · 16 days ago

lesson-template · removing stray hard-coded json · 23 days ago

spike-mp · Chinese translation to Spanish · 8 days ago

translations · 🚧 WIP: 韩语の翻訳を削除 · 22 days ago

gh-prone · adding a ghp app for more interactive issues · 2 months ago

aspify · preliminary work for docx implementation (ap) to space permit · 4 months ago

CODE_OF_CONDUCT.md · 4 months ago

About · 24 Lessons, 12 Weeks; Get Started as a Web Developer · 72 commits · last month

Releases · No releases published

Packages · No packages published

Contributors · 35 · + 44 contributors

This screenshot shows the GitHub repository page for 'Web-Dev-For-Beginners'. The 'Code' tab is selected, displaying a list of 72 commits. The commits are organized into several branches: 'main', 'branch management', 'getting-started-lessons', 'j-p-basics', 'terminus', 'typing-game', 'browser-extension', '4-track-game', 'book-project', 'lesson-template', 'spike-mp', 'translations', 'gh-prone', 'aspify', and 'CODE_OF_CONDUCT.md'. Each commit includes a commit message, author, date, and a link to the commit details. To the right of the commit list, there are sections for 'About' (describing the repository as '24 Lessons, 12 Weeks; Get Started as a Web Developer'), 'Releases' (none), 'Packages' (none), and 'Contributors' (35 total, with 44 more listed). The overall interface is dark-themed.

Credits

The motivation behind compiling all these resources was for our internal team at Fueler.

We then shared it with our Fueler Community so that more number of aspiring developers can learn and start their journey in Web Development.

Then one of us thought of an Idea why not share it with the world so that more number of individuals can share the benefit of all these amazing resources

Thanks to Pratham for his immense support toward the community. His sheer hardwork and passion for what he does is something inspire us a lot.

We genuinely want to take this opportunity to thank him from the whole Dev Community for his selfless commitment and contribution.

**Thanks to the amazing team at Thread Reader.
We are a proud supporter of you guys.**

Namaste!

