無料電子ブック

学習

CSS

Free unaffiliated eBook created from
Stack Overflow contributors.

#css

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: css

It is an unofficial and free CSS ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official CSS.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: CSSをいめる

スタイルは、ソースHTMLでされたときに、さまざまなのとをにする、いくつかのでオーサリングできます。 スタイルシートはHTMLでできます。 めみスタイルシートは、されたドキュメントにされます。 インラインスタイルは、された々のHTMLにのみされます。

バージョン

バージョン	
1	19961217
2	1998-05-12
3	2015-10-13

Examples

スタイルシート

HTMLスタイルシートは、HTMLに`<link>`をすることで、ののHTMLにできます。

`<link>`タグの`rel`は`"stylesheet"`し、 `href`はスタイルシートへのパスまたはパスにするがあります。 URLパスをするのははいですが、パスもできます。 HTML5では、 `type`をすることができます 。

HTMLファイルの`<head>`タグに`<link>`タグをして、スタイルがスタイルをするのにロードされるようにすることをおめします。さもなければ、 ユーザはスタイルのないコンテンツのフラッシュをるでしょう 。

hello-world.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <body>
        <h1>Hello world!</h1>
        <p>I ♥ CSS</p>
  </body>
</html>
```

style.css

```
h1 {
    color: green;
    text-decoration: underline;
}
p {
    font-size: 25px;
    font-family: 'Trebuchet MS', sans-serif;
}
```

CSSファイルへのしいパスをhrefにめるようにしてください。 CSSファイルがHTMLファイルとじフォルダにある、パスはありませんののようにが、フォルダにされているは、この `href="foldername/style.css"`ようにします。

```
<link rel="stylesheet" type="text/css" href="foldername/style.css">
```

スタイルシートは、CSSをするのとえられています。これはにながあります.1つのスタイルシートでされている100ページのサイトをしているときに、リンクのをからにしたいは、するのがずっとですあなたのCSSファイルにして、を100ページにって "カスケード"させ、それを100ページにけて100じをえます。あなたのウェブサイトのをにしたいは、この1つのファイルをするだけです。

にじてHTMLページにくのCSSファイルをロードできます。

```
<link rel="stylesheet" type="text/css" href="main.css">
<link rel="stylesheet" type="text/css" href="override.css">
```

CSSルールはいくつかのルールでされ、はです。たとえば、main.cssファイルにコードがまれているとします。

```
p.green { color: #00FF00; }
```

'green'クラスのすべてのはるいでかれますが、 main.cssのにするだけでの.cssファイルでこれをきできます。あなたはのコードをmain.cssのにけてoverride.cssをくことができます。

```
p.green { color: #006600; }
```

はあなたのすべてのはではなく、よりいでかれます。

なルール、、などののがされます。

かがにあなたのウェブサイトにアクセスすると、ブラウザはのページのHTMLとリンクされたCSSファイルをダウンロードします。その、のページにすると、ブラウザはそのページのHTMLをダウンロードするだけでみます。 CSSファイルはキャッシュされているので、ダウンロードするはありません。ブラウザはスタイルシートをキャッシュするので、ページのみみがします。

スタイル

HTMLドキュメントの<style></style>タグでまれたCSSは、スタイルシートのようにしますが、のファイルではなくスタイルけされたHTMLドキュメントにするため、。このは HTMLのために<head>になければならないことにしてください <body>されて<body>はのすべてのブラウザでします。

```
<head>
    <style>
        h1 {
            color: green;
            text-decoration: underline;
        }
        p {
            font-size: 25px;
            font-family: 'Trebuchet MS', sans-serif;
        }
    </style>
</head>
<body>
    <h1>Hello world!</h1>
    <p>I ♥ CSS</p>
</body>
```

インラインスタイル

インラインスタイルをして、のにスタイリングをします。これはではないことにしてください。コンテンツルールとプレゼンテーションルールをするために、 <style>タグまたはCSSファイルにスタイルルールをすることをおめします。

インラインスタイルは、 <style>タグまたはスタイルシートのCSSをきします。これはによってはですが、このはしばしばプロジェクトのをさせます。

ののスタイルは、それらがアタッチされているにされます。

```
<h1 style="color: green; text-decoration: underline;">Hello world!</h1>
<p style="font-size: 25px; font-family: 'Trebuchet MS';">I ♥ CSS</p>
```

インラインスタイルはに、さまざまなメールクライアント、プログラム、デバイスでのレンダリングのをするもなですが、きみにがかかり、するのがししいがあります。

CSS @importルールCSS at-ruleの1つ

@import CSS at-ruleは、のスタイルシートからスタイルルールをインポートするためにされます。これらのルールは、@ charsetルールのすべてのタイプのルールにするがあります。ネストされたステートメントではないので、@importはきのat-rulesでできません。 `@import` 。

@importのい

あなたは@importルールをのでうことができます

A.スタイルタグ

```
   <style>
    @import url('/css/styles.css');
   </style>
```

B.スタイルシートをする

のでは、ルートディレクトリの`additional-styles.css`というCSSファイルが、そのCSSファイルにインポートされます。

```
@import '/additional-styles.css';
```

CSSをインポートすることもです。なは、フォントファイルです。

```
@import 'https://fonts.googleapis.com/css?family=Lato';
```

`@import`ルールのオプションの2のは、メディアクエリのリストです

```
@import '/print-styles.css' print;
@import url('landscape.css') screen and (orientation:landscape);
```

JavaScriptによるCSSの

なJavaScript

の`style`プロパティをして、JavaScriptでCSSのプロパティを、、することはです。

```
var el = document.getElementById("element");
el.style.opacity = 0.5;
el.style.fontFamily = 'sans-serif';
```

スタイルのプロパティは、ラクダのスタイルでされています。このでは、CSSプロパティ`font-family`がjavascriptの`fontFamily`になることがわかり`fontFamily`。

をするわりに、JavaScriptで`<style>` `<link>`または`<link>`をし、HTMLの`<body>`または`<head>`にすることができます。

jQuery

jQueryをしてCSSプロパティをすると、さらににになります。

```
$('#element').css('margin', '5px');
```

のスタイルルールをするがある

```
$('#element').css({
    margin: "5px",
```

```
    padding: "10px",
    color: "black"
});
```

jQueryにはハイフンをむCSSルール `font-size` をする2つのがあります。スタイルルールをでむかキャメルケースにすることができます。

```
$('.example-class').css({
    "background-color": "blue",
    fontSize: "10px"
});
```

もしてください

- JavaScriptのドキュメント - CSSスタイルのみみと 。
- jQueryドキュメント - CSS

CSSによるリストのスタイル

リストの`list-style-type` 、 `list-style-image` `list-style-type` 、 `list-style-image` 、 `list-style-position` 3つのプロパティがあります。そのでするがあります。デフォルトはそれぞれdisc、outside、noneです。プロパティは、 々にすることも、 `list-style`プロパティをしてすることもできます。

`list-style-type`は、リストにされるきのまたはタイプをします。

`list-style-type`されるのいくつかはのとおりです。

- ディスク
- サークル
-
- の
- ローマン
- ローマ
- し

なリストについては、 W3Cwikiをしてください

たとえば、リストアイテムにいきをするには、のプロパティとのペアをします。

```
li {
    list-style-type: square;
}
```

`list-style-image`プロパティーは、リストアイコンにがされているかどうかをし、`none`またはをす

URLをけれます。

```
li {
    list-style-image: url(images/bullet.png);
}
```

---

list-style-positionプロパティは、リストアイテムマーカーのをし、 "inside"または "outside"という2つののいずれかをけれます。

```
li {
  list-style-position: inside;
}
```

オンラインでCSSをいめるをむ https://riptutorial.com/ja/css/topic/293/cssをいめる

2: 2D

-
- transformrotate<angle>
- をする
- transformtranslate<length-or-percentage> [、<length-or-percentage>]
- transformtranslateX<length-or-percentage>
- transformtranslateY<length-or-percentage>
- スキュー
- transformスキュー<angle> [、<angle>]
- transformskewX<angle>
- transformskewY<angle>
- スケール
- transformscale<scale-factor> [、<scale-factor>]
- transformscaleX<scale-factor>
- transformscaleY<scale-factor>
-
- transform<number> [、<number>] {5,5}

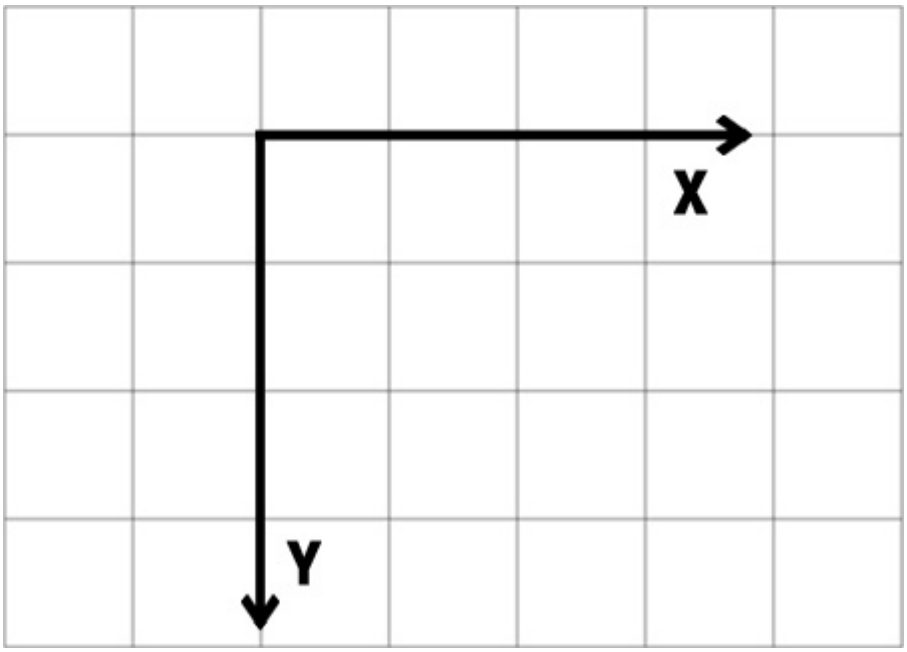
パラメーター

パラメータ	
`rotate(x)`	をZののりにするをします
`translate(x,y)`	XとYののをします。
`translateX(x)`	Xののをします。
`translateY(y)`	Yののをします。
`scale(x,y)`	XとYののサイズをします。
`scaleX(x)`	Xののサイズをします。
`scaleY(y)`	Yののサイズをします。
`skew(x,y)`	せんまたは、のをにのだけませる
`skewX(x)`	のをにあるだけませるマッピング
`skewY(y)`	のをにあるだけませる
`matrix()`	ので2Dをします。

/パラメータ	
	をまたはさせるするによってなる。は、 `deg` 、 `grad` 、ラジアン `rad` またはターン `turn` でできます。 `skew()` では、2のはオプションです。されていないは、Yにスキュ—0がありません。
さまたはパ—センテ—ジ	がされるべきさまたはパ—センテ—ジとしてされる。 `translate()` では、2のlength-or-percentageはオプションです。されていない、Yには0のはありません。
スケ—ルファクタ	がされたでスケ—ルされるべきかをする。 `scale()` では、2のはオプションです。されていないは、Yにものスケ—ルファクタがされます。

2Dコ—ディエンティシステム

は、2D X / Yにってわれます。のにすように、Xはからにし、Yはにします。



したがって、の`translateY()`はきになり、の`translateX()`はにみます。

ブラウザサポ—トとプレフィックス

- IEは、 `-ms-`きのIE9、このプロパティをサポ—トして`-ms-`ます。いバ—ジョンとEdgeではプレフィックスはありません
- Firefoxはバ—ジョン3.5をサポ—トしており、バ—ジョン15までは`-moz-`がです
- バ—ジョン4のChrome、バ—ジョン34までは`-webkit-`プレフィックスが`-webkit-`
- Safariはバ—ジョン8まで`-webkit-`がです
- Operaはバ—ジョン11.5では`-webkit-`、バ—ジョン15から22では`-o-`がです
- Androidはバ—ジョン2.1から4.4.4の`-webkit-`がです

きの

```
-webkit-transform: rotate(45deg);
    -ms-transform: rotate(45deg);
        transform: rotate(45deg);
```

Examples

する

HTML

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

CSS

```
.rotate {
    width: 100px;
    height: 100px;
    background: teal;
    transform: rotate(45deg);
}
```

このではdivをりに45します。のは、DIVのにある`50%`からと`50%`から。 `transform-origin`プロパティをすると、をできます。

```
transform-origin: 100% 50%;
```

のでは、をのにします。

HTML

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

CSS

```
.scale {
    width: 100px;
    height: 100px;
    background: teal;
    transform: scale(0.5, 1.3);
}
```

このでは、divをXで`100px * 0.5 = 50px`、Yで`100px * 1.3 = 130px`します。

のは、divのにある`50%`からと`50%`から。

HTML

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

CSS

```
.translate {
    width: 100px;
    height: 100px;
    background: teal;
    transform: translate(200px, 50%);
}
```

このでは、divをXで200px、Yで`100px * 50% = 50px`します。

1つのにをすることもできます。

Xの

```
.translate {
    transform: translateX(200px);
}
```

Yで

```
.translate {
    transform: translateY(50%);
}
```

ゆがみ

HTML

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

CSS

```
.skew {
    width: 100px;
    height: 100px;
    background: teal;
    transform: skew(20deg, -30deg);
}
```

このでは、divをXで20、Yで-30させます。
のは、divのにある`50%`からと`50%`から。

ここのをてください 。

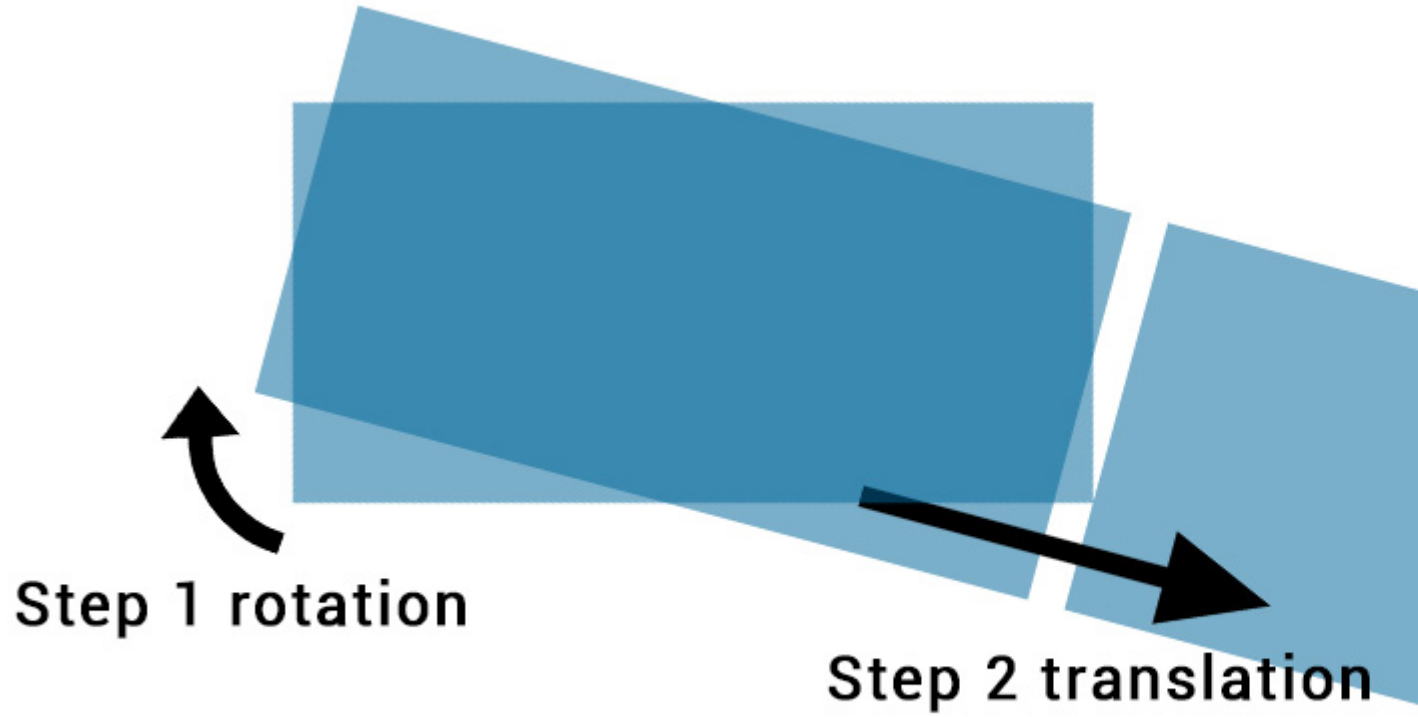
の

のように、1つのプロパティのにのをできます。

```
transform: rotate(15deg) translateX(200px);
```

をりに15させ、に200ピクセルにします。

では、はとともにします。つまり、はではなく、でののようにりに15します。



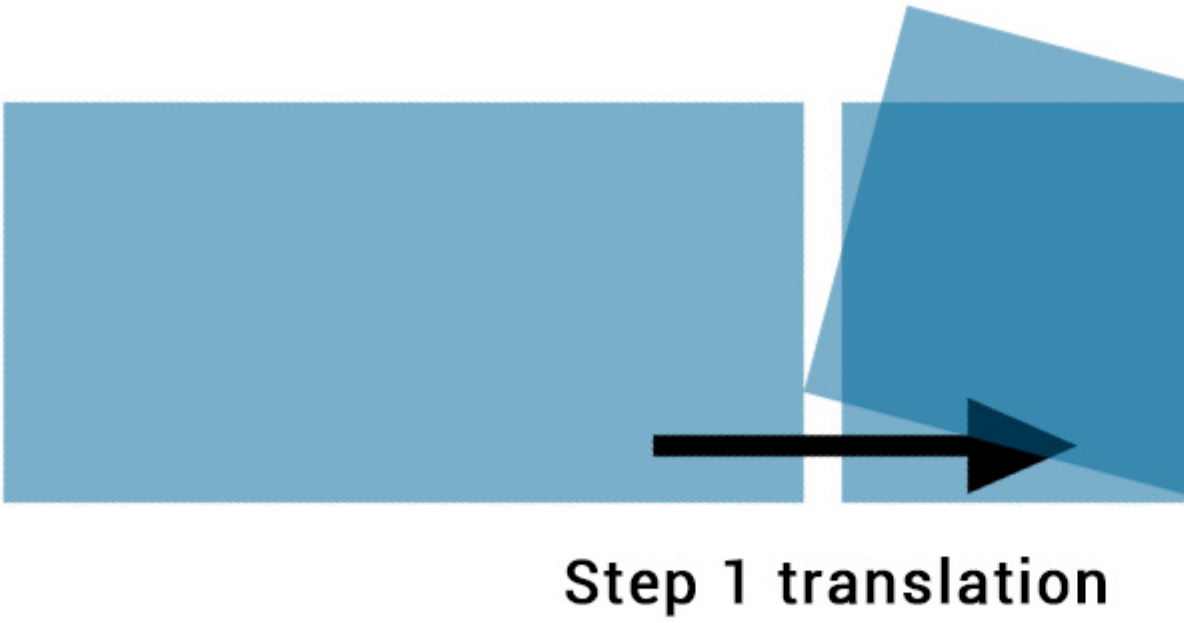
のをすると、がされます。のは、

```
transform: translateX(200px) rotate(15deg);
```

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

```
.transform {
  transform: rotate(15deg) translateX(200px);
}
```

このにすように



は、 `transform-origin`プロパティによってされたにしてわれます。

このプロパティは2つのをとります `transform-origin: XY;`

のでは、のdiv `.t1` は、 `transform-origin: 0 0;`をにし`transform-origin: 0 0;` 2の `.tr` は`transform-origin: 100% 0`に`transform-origin: 100% 0`。 ホバーでローテーションがされます

HTML

```
<div class="transform originl"></div>
<div class="transform origin2"></div>
```

CSS

```
.transform {
    display: inline-block;
    width: 200px;
    height: 100px;
    background: teal;
    transition: transform 1s;
}

.origin1 {
    transform-origin: 0 0;
}

.origin2 {
```

```
    transform-origin: 100% 0;
}

.transform:hover {
    transform: rotate(30deg);
}
```

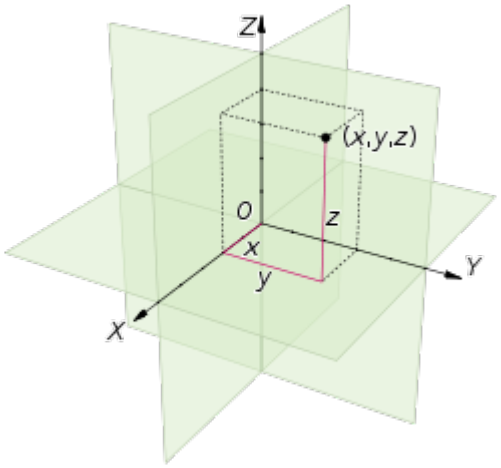
プロパティのデフォルトは、のである`50% 50%`です。

オンラインで2Dをむ https://riptutorial.com/ja/css/topic/938/2d

3: 3D

3Dは、 ユークリッドの `(x, y, z)` ベクトルシステムにってわれる。

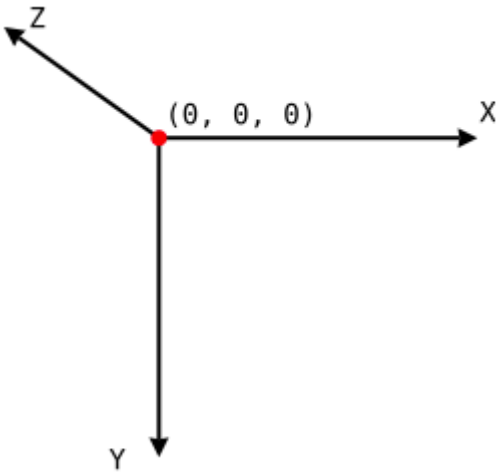
のは、ユークリッドののをしています。



CSSでは、

- `x`はおよび
- `y`は
- `z`は、き、

のは、これらのがCSSでどのようにされるかをしています。



Examples

3Dキューブ

3Dは、くの3Dをするためにできます。な3D CSSキューブのをにします。

HTML

```
<div class="cube">
  <div class="cubeFace"></div>
  <div class="cubeFace face2"></div>
</div>
```

CSS

```
body {
  perspective-origin: 50% 100%;
  perspective: 1500px;
  overflow: hidden;
}
.cube {
  position: relative;
  padding-bottom: 20%;
  transform-style: preserve-3d;
  transform-origin: 50% 100%;
  transform: rotateY(45deg) rotateX(0);
}
.cubeFace {
  position: absolute;
  top: 0;
  left: 40%;
  width: 20%;
  height: 100%;
  margin: 0 auto;
  transform-style: inherit;
  background: #C52329;
  box-shadow: inset 0 0 0 5px #333;
  transform-origin: 50% 50%;
  transform: rotateX(90deg);
  backface-visibility: hidden;
}
.face2 {
  transform-origin: 50% 50%;
  transform: rotatez(90deg) translateX(100%) rotateY(90deg);
}
.cubeFace:before, .cubeFace:after {
  content: '';
  position: absolute;
  width: 100%;
  height: 100%;
  transform-origin: 0 0;
  background: inherit;
  box-shadow: inherit;
  backface-visibility: inherit;
}
.cubeFace:before {
  top: 100%;
  left: 0;
  transform: rotateX(-90deg);
}
.cubeFace:after {
  top: 0;
  left: 100%;
  transform: rotateY(90deg);
}
```

このをる
のスタイリングがデモにされ、ホバーでがされてキューブの6がされます。

それにするがあります

- 4はでられています
- がされる

の

`backface-visibility`プロパティは3Dにします。

3Dと`backface-visibility`プロパティをすると、ののがにかないようにをさせることができます。

たとえば、これはをからします

JSFIDDLE

```
<div class="flip">Loren ipsum</div>
<div class="flip back">Lorem ipsum</div>
```

```
.flip {
  -webkit-transform: rotateY(180deg);
  -moz-transform: rotateY(180deg);
  -ms-transform: rotateY(180deg);
  -webkit-backface-visibility: visible;
  -moz-backface-visibility:    visible;
  -ms-backface-visibility:     visible;
}

.flip.back {
  -webkit-backface-visibility: hidden;
  -moz-backface-visibility:    hidden;
  -ms-backface-visibility:     hidden;
}
```

Firefox 10+およびIE 10+は、プレフィックスなしで`backface-visibility`をサポートします。 Opera、Chrome、Safari、iOS、Androidはすべて、 `-webkit-backface-visibility`が`-webkit-backface-visibility`。

それは4つのをっています

1. **visible** デフォルト - をいていなくてもはにされます。
2. **hidden** - スクリーンにしていないときにがされません。
3. **inherit** - プロパティはそのからそのをします
4. **initial** - プロパティをデフォルトにします。これはです

3Dをしたコンパスポインタまたは

CSS

```
div.needle {
  margin: 100px;
  height: 150px;
  width: 150px;
  transform: rotateY(85deg) rotateZ(45deg);
  /* presentational */
  background-image: linear-gradient(to top left, #555 0%, #555 40%, #444 50%, #333 97%);
  box-shadow: inset 6px 6px 22px 8px #272727;
}
```

HTML

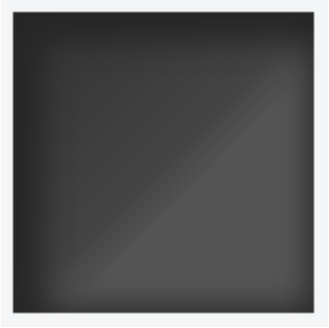
```
<div class='needle'></div>
```

のでは、またはコンパスのポインタが3Dをしてされています。に、に`rotate`をすると、 `rotate`はZでのみこり、せいぜいのみになります。しかし、 `rotateY`トランスフォームがそのにされると、はYでされ、のようにえます。 Yのがきくなればなるほど、はよりにえます。

ののは、そのにかれたであろう。ベースにあるをするには、をYにってではなくXにってうがあります。したがって、 `transform`プロパティのは`rotateX(85deg) rotateZ(45deg);`ようなものでなければなりません`rotateX(85deg) rotateZ(45deg);`。

このペンでもので、Safariのロゴやコンパスのダイヤルのようなものをします。

のないのスクリーンショット



2Dのみののスクリーンショット

3Dトランスフォームをしたのスクリーンショット

きの**3D**テキスト

HTML

```
<div id="title">
  <h1 data-content="HOVER">HOVER</h1>
</div>
```

CSS

```
*{margin:0;padding:0;}
html,body{height:100%;width:100%;overflow:hidden;background:#0099CC;}
#title{
  position:absolute;
  top:50%; left:50%;
  transform:translate(-50%,-50%);
  perspective-origin:50% 50%;
  perspective:300px;
}
h1{
  text-align:center;
  font-size:12vmin;
  font-family: 'Open Sans', sans-serif;
  color:rgba(0,0,0,0.8);
  line-height:1em;
```

```
  transform:rotateY(50deg);
  perspective:150px;
  perspective-origin:0% 50%;
}
h1:after{
  content:attr(data-content);
  position:absolute;
  left:0;top:0;
  transform-origin:50% 100%;
  transform:rotateX(-90deg);
  color:#0099CC;
}
#title:before{
  content:'';
  position:absolute;
  top:-150%; left:-25%;
  width:180%; height:328%;
  background:rgba(255,255,255,0.7);
  transform-origin: 0 100%;
  transform: translatez(-200px) rotate(40deg) skewX(35deg);
  border-radius:0 0 100% 0;
}
```

ホバーをしたをる

このでは、テキストは、ユーザーかられたにされるようにされています。

それにじてがされてテキストにいます。と`data`でられているので、それはそのH1タグからのをします。

い「」は`#title`のでられます。めになっており、みをびたコーナーにボーダーをします。

オンラインで3Dをむ https://riptutorial.com/ja/css/topic/2446/3d

4: CSSイメージスプライト

- //バックグラウンドをする
  url "sprite-image.png";
  バックグラウンドの-20px 50px;
- //バックグラウンドプロパティの
  url "sprite-image.png"-20px 50px;

いくつかのユースケースでは、スプライトはゆっくりとをい、アイコンのWebフォントやSVGイメージにきえられます。

Examples

な

イメージスプライトとはですか

イメージスプライトは、イメージスプライトシートにあるのアセットです。イメージスプライトシートは、そこからできるのアセットをむイメージファイルです。

えば

のはスプライトシートで、これらののそれぞれはスプライトシートのスプライトです。これらのスプライトシートは、ブラウザがとするがあるHTTPリクエストのをらすことによってパフォーマンスをさせるので、です。

では、どのようにしますかここにいくつかのコードがあります。

HTML

```
<div class="icon icon1"></div>
<div class="icon icon2"></div>
<div class="icon icon3"></div>
```

CSS

```
.icon {
    background: url(“icons-sprite.png”);
    display: inline-block;
    height: 20px;
    width: 20px;
}
.icon1 {
      background-position: 0px 0px;
}
.icon2 {
      background-position: -20px 0px;
}
.icon3 {
      background-position: -40px 0px;
}
```

スプライトのとさをし、CSSのbackground-positionプロパティxとyのををすると、CSSをしてスプライトシートからにスプライトをできます。

オンラインでCSSイメージスプライトをむ https://riptutorial.com/ja/css/topic/3690/cssイメージスプライト

5: CSSオブジェクトモデルCSSOM

CSS Object ModelCSSOMはそれのです。

のドラフトは、 https : //www.w3.org/TR/cssom-1/でごいただけます。

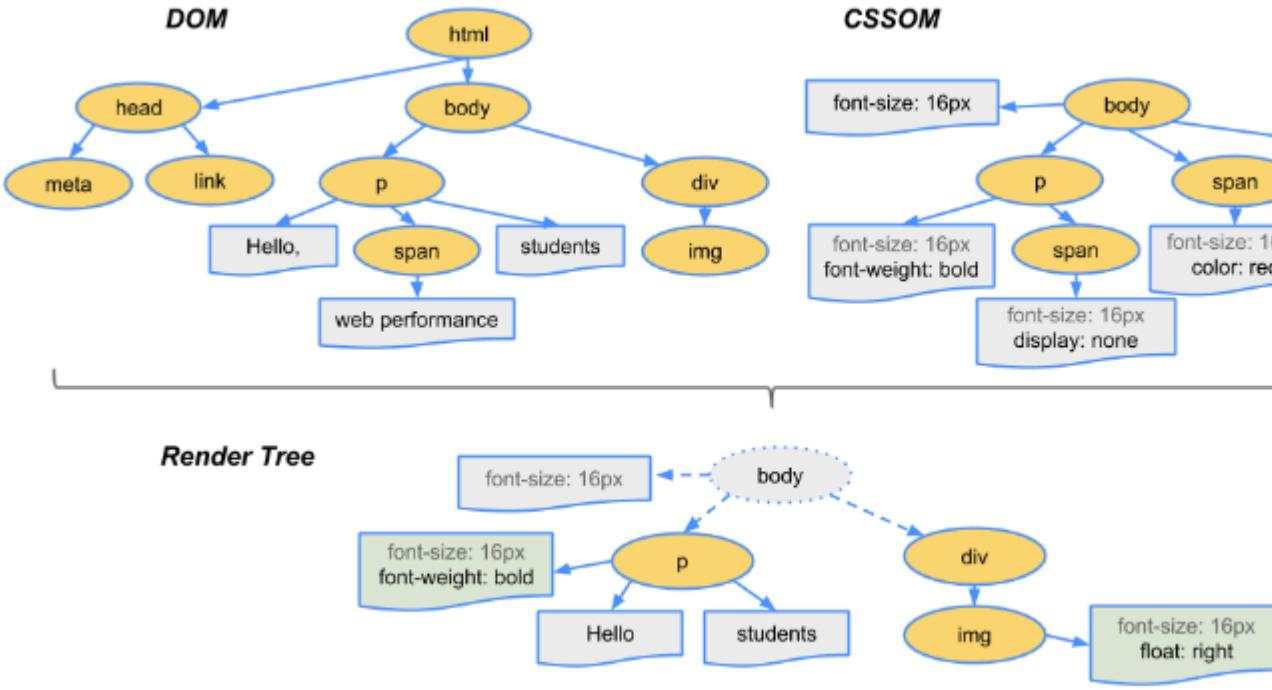
Examples

き

ブラウザはスタイルシートからトークンをし、それらをツリーにリンクされたノードにします。
ページのするスタイルをつすべてのノードのマップがCSSオブジェクトモデルになります。

Webページをするには、Webブラウザでのをします。

1. WebブラウザはHTMLをべ、DOMDocument Object Modelをします。
2. WebブラウザはCSSをべ、CSSOMCSS Object Modelをします。
3. Webブラウザは、DOMとCSSOMをみわせてレンダーツリーをします。 Webブラウザに Webページがされます。



CSSOMをしてルールをする

CSSOMをしてルールをするには、まずのスタイルシートのルールへのをします。

```
var stylesheet = document.styleSheets[0].cssRules;
```

に、スタイルシートののをします。

```
var end = stylesheet.length - 1;
```

に、bodyのルールをスタイルシートのにします。

```
stylesheet.insertRule("body { background-image:
url('http://cdn.sstatic.net/Sites/stackoverflow/img/favicon.ico'); }", end);
```

オンラインでCSSオブジェクトモデルCSSOMをむ https://riptutorial.com/ja/css/topic/4961/cssオブジェクトモデル-cssom-

6: CSSデザインパターン

き

これらのは、 BEM 、 OOCSS 、 SMACSSなどのCSSのデザインパターンをするためのものです。

これらのは、 BootstrapやFoundationのようなCSSフレームワークをするためのものではありません。

これらのは、CSSの/パターンをするためのものです。

これらのには、がまれますが、これにされません。

- BEM
- OOCSS
- SMACSS

これらのは、 BootstrapやFoundationのようなCSSフレームワークをするためのものではありません。 CSSフレームワークで1つのCSSメソドロジ/デザインパターンをするのをめることができますが、それらのは、そののフレームワークでのメソドロジ/デザインパターンとフレームワークのにをてることです。

Examples

BEM

BEMは`Blocks, Elements and Modifiers`です。これは、ロシアのYandexによってされたでしたが、のWebにもかなりのをもたらしました。

じで、BEMメゾロジーは、HTMLとCSSコードを3つのタイプのコンポーネントにコンポーネントすることにするものです。

- ブロックのをつスタンドアロンエンティティ

  は、 `header` 、 `container` 、 `menu` 、 `checkbox` 、 `textbox`

- スタンドアロンのをたず、ブロックににびついているブロックの。

  としては、 `menu item` 、 `list item` 、 `checkbox caption` と `header title`

- ブロックまたはのフラグで、ややをするためにされます。

  は `disabled` 、 `highlighted` 、 `checked` 、 `fixed` 、 `size big` 、 `color yellow`

BEMのは、CSSコードの、、をすることです。これをするは、のルールをすることです。

- ブロックスタイルはしてページののにしません
- ブロックはシンプルでいで、 _または-をけるがあります
- をスタイリングするときは、 `blockname__elementname`のセレクタをします
- スタイリングをするは、 `blockname--modifiername`および`blockname__elementname--modifiername`のセレクタをします
- をつまたはブロックは、のブロックまたはからすべてをするがあります

コード

BEMをフォームにすると、CSSセレクタはのようになります。

```
.form { }                       // Block
.form--theme-xmas { }           // Block + modifier
.form--simple { }               // Block + modifier
.form__input { }                // Block > element
.form__submit { }               // Block > element
.form__submit--disabled { }     // Block > element + modifier
```

するHTMLはのようになります。

```
<form class="form form--theme-xmas form--simple">
  <input class="form__input" type="text" />
  <input class="form__submit form__submit--disabled" type="submit" />
</form>
```

オンラインでCSSデザインパターーンをむ https://riptutorial.com/ja/css/topic/10823/cssデザインパターーン

7: CSSルールのと

みやすくするために、すべてのをセレクタから1レベルインデントし、じをのにインデントします。セレクタとコロンのに1つのスペースをし、ののにセミコロンをいてください。

い

```
p {
    color: maroon;
    font-size: 16px;
}
```

い

```
p{
 color: maroon;
font-size:16px }
```

ワンライナー

が1つか2つしかないは、このをしないでください。ほとんどのはおめできません。になりしてください。

```
p { color: maroon; font-size: 16px; }
```

Examples

ルール、セレクタ、およびブロック

CSS ルールは、セレクタ `h1` とブロック `{}` でされます。

```
h1 {}
```

プロパティリスト

いくつかのプロパティは、のをることができ、にプロパティリストとばれます。

```
/* Two values in this property list */
span {
    text-shadow: yellow 0 0 3px, green 4px 4px 10px;
```

```
}

/* Alternate Formatting */
span {
    text-shadow:
        yellow 0 0 3px,
        green 4px 4px 10px;
}
```

セレクタ

CSSセレクタをグループすると、スタイルシートでスタイルをりさずに、じスタイルをのなるにできます。グループされたのセレクタをカンマでります。

```
div, p { color: blue }
```

はすべての とすべての<p>にされます。カンマなしで である<p>はになります。

これはすべてのタイプのセレクタにもてはまります。

```
p, .blue, #first, div span{ color : blue }
```

このルールはのものにされます。

- <p>
- blueクラスの
- firstにIDをつ
- すべての

オンラインでCSSルールのとをむ https://riptutorial.com/ja/css/topic/4313/cssルールのと

8: Internet Explorerのハック

これらの「ハック」は、のブラウザ/クライアントをターゲットにするためにされます。のラッパーのいずれかにスタイルをすることで、ブラウザーのレンダリングのいをすることができます。

Examples

Internet Explorer 10のコントラストモード

Internet Explorer 10+およびEdgeでは、マイクロソフトが`-ms-high-contrast`メディアセレクタをしてブラウザからの "High Contrast"をします。これによりプログラマはサイトのスタイルをにできます。

`-ms-high-contrast`セレクタには、 `active` 、 `black-on-white` 、および`white-on-black` `black-on-white` 3つのがあります。 IE10 +ではこれも`none`でしたが、Edgeではこれがサポートされなくなりました。

```
@media screen and (-ms-high-contrast: active), (-ms-high-contrast: black-on-white) {
    .header{
       background: #fff;
       color: #000;
    }
}
```

コントラストモードがアクティブで、 `black-on-white`モードになっている、ヘッダーのがに、テキストのがにわります。

```
@media screen and (-ms-high-contrast: white-on-black) {
    .header{
       background: #000;
       color: #fff;
    }
}
```

のとですが、これはには`white-on-black`のみをし、ヘッダーカラーをいテキストのいにします。

しくは

`-ms-high-contrast`のMicrosoftドキュメント

Internet Explorer 6およびInternet Explorer 7のみ

Internet Explorer 6とInternet Explorer 7をにするには、 `*`プロパティをします。

```
.hide-on-ie6-and-ie7 {
```

```
    *display : none; // This line is processed only on IE6 and IE7
}
```

IE6とIE7では、のプレフィックスハイフンやアンダースコアはされるため、この`property: value`れていない`property: value`ペアにしてし`property: value`。

Internet Explorer 8のみ

Internet Explorer 8をターゲットにするには、セレクタを`@media \0 screen { }`にラップします。

```
@media \0 screen {
    .hide-on-ie8 {
        display : none;
    }
}
```

`@media \0 screen { }`はすべてIだけでされます

IE6とIE7にインラインブロックサポートをする

```
display: inline-block;
```

`inline-block`をつ`display`プロパティは、Internet Explorer 6および7ではサポートされていません。これをするには、のようにします。

```
zoom: 1;
*display: inline;
```

`zoom`プロパティは、の`hasLayout`をトリガし、Internet Explorerでのみできます。 `*display`は、なプロパティがをけるブラウザでのみされることをします。のブラウザはにルールをします。

オンラインでInternet Explorerのハックをむ https://riptutorial.com/ja/css/topic/5056/internet-explorerのハック

9: アニメーション

- `transition: <property> <duration> <timing-function> <delay>;`
- `@keyframes <identifier>`
- `[ [ from | to | <percentage> ] [, from | to | <percentage> ]* block ]*`

パラメーター

パラメータ	
プロパティ	の、またはするCSSプロパティのいずれか`all`すべてのなプロパティをします、。
	またはミリ。
タイミング	プロパティののをするをします。なは、 `ease` 、 `linear` 、 `step-end`です。しくは、 イージングのチートシートをしてください。
ディレイ	アニメーションをするまでのまたはミリ。
@keyframes	
[from | | `<percentage>`]	キーフレームのセットがされているにパーセント、または2つのパーセント、つまり`10%, 20%`でをすることができます。
`block`	キーフレームのCSSの。

Examples

プロパティをつアニメーション

シンプルなアニメーションになCSS `transition`プロパティでは、ベースのCSSプロパティをステートでアニメーションできます。

```
.Example{
    height: 100px;
    background: #fff;
}

.Example:hover{
    height: 120px;
    background: #ff0000;
```

```
}
```

をる

デフォルトでは、 `.Example`クラスをつに`.Example`と、のさはすぐに`120px`に、はに `#ff0000` ジャンプします。

`transition`プロパティをすることで、これらのがのとともにするがあります。

```
.Example{
    ...
    transition: all 400ms ease;
}
```

をる

`all`は、すべてののあるベースのプロパティにします。ののあるプロパティえば、 `height`は`top` このキーワードにすることができます。

`400ms`は、にするをします。この、のさのには400ミリかかります。

に、`ease`は、アニメーションのをするアニメーションです。 `ease`することは、ゆっくりとし、スピードアップし、びゆっくりとすることをするのは`linear` 、 `ease-out` 、および`ease-in` 。

ブラウザの

`transition`プロパティはに、IE 9をくなブラウザすべてでサポートされています。FirefoxおよびWebkitベースのブラウザののバージョンでは、のようなベンダープレフィックスをします。

```
.Example{
    transition:         all 400ms ease;
    -moz-transition:    all 400ms ease;
    -webkit-transition: all 400ms ease;
}
```

`transition`プロパティは、にかかわらず、の2つののをアニメートできます。また、 `100px`から`50vh` ようなのも`50vh` 。ただし、エレメントのさを`100px`から`auto`するなど、とデフォルトまたはのですることはできません。

`will-change`をったアニメーションのパフォーマンスの

アニメーションやそののGPUいアクションをするときは、 `will-change`をする`will-change`です。

CSSキーフレームと`transition`プロパティのがGPUアクセラレーションをします。をデバイスのGPUにオフロードすることで、パフォーマンスがします。これは、されるGPUにオフロードされ

るペイントレイヤーにレンダリングされるページのをすることによってわれます。 `will-change`プロパティは、ブラウザにアニメートするものをし、ブラウザがさなペイントをしてパフォーマンスをさせるようにします。

`will-change`プロパティは、アニメートされるプロパティのカンマりリストをけれます。たとえば、オブジェクトをしてをするは、のようにします。

```
.Example{
    ...
    will-change: transform, opacity;
}
```

`will-change`えめに`will-change`ください。ブラウザでのペイントレイヤーをしようとすると、GPUによるがにするがあるため、ページののの`will-change`するとパフォーマンスのがするがあります。

キーフレームきアニメーション

マルチステージCSSアニメーションでは、CSS `@keyframes`をできます。キーフレームでは、よりなアニメーションをするために、キーフレームとばれるのアニメーションポイントをできます。

な

このでは、すべてのをさせるなアニメーションをします。

```
@keyframes rainbow-background {
         0%     { background-color: #ff0000; }
      8.333%    { background-color: #ff8000; }
     16.667%    { background-color: #ffff00; }
     25.000%    { background-color: #80ff00; }
     33.333%    { background-color: #00ff00; }
     41.667%    { background-color: #00ff80; }
     50.000%    { background-color: #00ffff; }
     58.333%    { background-color: #0080ff; }
     66.667%    { background-color: #0000ff; }
     75.000%    { background-color: #8000ff; }
     83.333%    { background-color: #ff00ff; }
     91.667%    { background-color: #ff0080; }
     100.00%    { background-color: #ff0000; }
}

.RainbowBackground {
    animation: rainbow-background 5s infinite;
}
```

をる

ここですべきがいくつかあります。まず、の`@keyframes`。

```
@keyframes rainbow-background{
```

これにより、アニメーションのが`rainbow-background`ます。

```
0%     { background-color: #ff0000; }
```

これは、アニメーションのキーフレームのです。のは、ケースの`0%`で、アニメーションのキーフレームのをします。 0は、からアニメーションののの`0%`します。

アニメーションはキーフレームでにりわります。そのため、のを`8.333%`にすると、アニメーションはそのキーフレームをするの8.333をスムーズにめます。

```
.RainbowBackground {
    animation: rainbow-background 5s infinite;
}
```

このコードは、 `.RainbowBackground`クラスをつすべてのにアニメーションをします。

のアニメーションプロパティには、のがあります。

- **animation-name** たちのアニメーションの。この、 `rainbow-background`
- **animation-duration** アニメーションの このは5。
- **animation-iteration-count**オプション アニメーションがループする。この、アニメーションはにきます。デフォルトでは、アニメーションは1されます。
- **animation-delay**オプション アニメーションをするまでのをします。デフォルトは0で、のをとることができます。えば、 `-2s`そのループにアニメーション2をします。
- **animation-timing-function**オプション アニメーションのカーブをします。これがデフォルト`ease` 、アニメーションは、いまりくし、い。

こののでは、 `0%`と`100%`のキーフレームで`{ background-color: #ff0000; }` 。 2つのキーフレームがをするは、1つのステートメントですることができます。この、2つの`0%`と`100%`をこの1できえることができます

```
0%, 100%     { background-color: #ff0000; }
```

クロスブラウザとの

いWebKitベースのブラウザでは、のように`@keyframes`と`animation`プロパティのにベンダープレフィックスをするがあります。

```
@-webkit-keyframes{}

-webkit-animation: ...
```

のは、なすべてのプロパティ/パラメータをしたアニメーションのプロパティをしています。

```
  animation: 3s         ease-in          1s      2                reverse     both
```

```
paused       slidein;
  /*         duration | timing-function | delay | iteration-count | direction | fill-mode |
play-state | name   */
```

たちの2のはしシンプルで、いくつかのプロパティをできることをしています

```
  animation: 3s         linear            1s      slidein;
  /*         duration | timing-function | delay | name   */
```

3のは、のをしています。 animation-nameとanimation-durationをするがあることにしてください。

```
  animation: 3s         slidein;
  /*         duration | name */
```

また、アニメーションをするときは、プロパティのがいをむことにもするがあります。らかに、ブラウザはあなたのをあなたのれとするかもしれません。

さがあなたのものでないは、のプロパティをスキップして、プロパティをにきすこともできます。

```
animation-duration: 3s;
animation-timing-function: ease-in;
animation-delay: 1s;
animation-iteration-count: 2;
animation-direction: reverse;
animation-fill-mode: both;
animation-play-state: paused;
animation-name: slidein;
```

オンラインでアニメーションをむ https://riptutorial.com/ja/css/topic/590/アニメーション

10: インラインブロックレイアウト

Examples

みのナビゲーションバー

にされたナビゲーションメニューバーには、されるべきいくつかのがあります。のアイテムはコンテナにマージンをたず、アイテムはコンテナにマージンをたない。アイテムのは、アイテムのごとにしています。

HTML

```
<nav>
    <ul>
        <li>abc</li>
        <li>abcdefghijkl</li>
        <li>abcdef</li>
    </ul>
</nav>
```

CSS

```
nav {
    width: 100%;
    line-height: 1.4em;
}
ul {
    list-style: none;
    display: block;
    width: 100%;
    margin: 0;
    padding: 0;
    text-align: justify;
    margin-bottom: -1.4em;
}
ul:after {
    content: "";
    display: inline-block;
    width: 100%;
}
li {
    display: inline-block;
}
```

ノート

- `nav`、 `ul`、 `li`タグは、ナビゲーションのリストとしてされました。もちろん、のタグをすることもできます。
- `:after`は、 `ul`にな"をもたらし、したがってこのブロックのな、のさをもたらし、のコンテンツをしげます。これは、の`margin-bottom`によってされますが、 `margin-bottom`は`line-height`とじきさでなければならないただしの。
- ページがすぎてすべてのアイテムがまりきらない、アイテムはしいラインからまりますにされ、このラインでされます。メニューのさはにじてきくなります。

オンラインでインラインブロックレイアウトをむ https://riptutorial.com/ja/css/topic/3308/インラインブロックレイアウト

11: オーバーフロー

- オーバーフロー|された|スクロール|オート|の|する;

パラメーター

Overflow	
visible	のにあるすべてのオーバーフローコンテンツをします。
scroll	オーバーフローしているコンテンツをにし、スクロールバーをします。
hidden	オーバーフローしているコンテンツをにし、のスクロールバーがえてページがになります
auto	コンテンツがオーバーフローしたは`scroll`じですが、コンテンツがまるはスクロールバーをしません
inherit	Inheritは、このプロパティののです

> `overflow`プロパティは、コンテンツをクリップするか、スクロールバーをレンダリングするか、コンテナをしてブロックレベルのコンテナをオーバーフローさせたときにコンテンツをするかどうかをします。

がさすぎてをできないはどうなりますかデフォルトでは、コンテンツはオーバーフローしてのにされます。それはあなたのをくせます。あなたのはえがいいので、オーバーフローするコンテンツをましいでするようにオーバーフロープロパティをします。

`overflow`プロパティのは`overflow-x`プロパティと`overflow-y`プロパティのとじですにされます。

`overflow-wrap`プロパティは、 `word-wrap`プロパティとしてもられています。

> な visibleとはなるでoverflowプロパティをすると、しいブロックコンテキストがされます。

Examples

オーバーフロースクロール

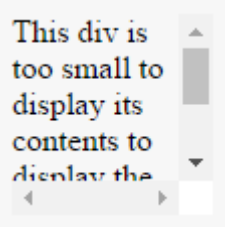
HTML

```
<div>
    This div is too small to display its contents to display the effects of the overflow
property.
```

```
</div>
```

CSS

```
div {
    width:100px;
    height:100px;
    overflow:scroll;
}
```



のコンテンツは、100px×100pxボックスでクリップされ、オ―バ―フロ―するコンテンツをするためのスクロ―ルがです。

ほとんどのデスクトップブラウザでは、コンテンツがクリップされているかどうかにかかわらず、スクロ―ルバ―とスクロ―ルバ―のがされます。これにより、でスクロ―ルバ―がされたりえたりするをできます。プリンタは、オ―バ―フロ―したコンテンツをすることがあります。

オ―バ―フロ―ラップ

`overflow-wrap`は、ののテキストのを、それのはられないののにすることができることをブラウザにします。コンテナのオ―バ―フロ―によるレイアウトのをきこすいのにちます。

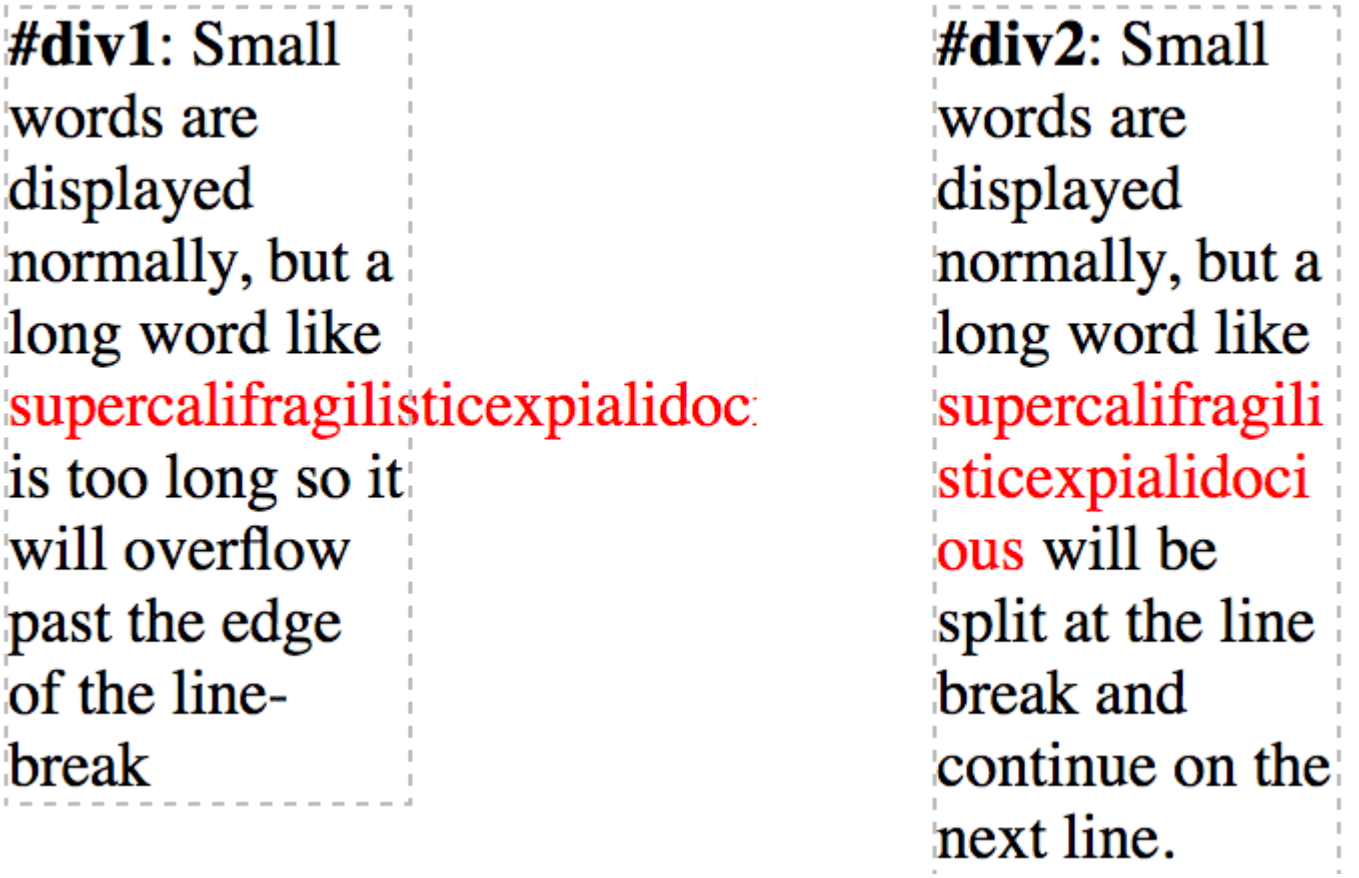
CSS

```
div {
    width:100px;
    outline: 1px dashed #bbb;
}

#div1 {
    overflow-wrap:normal;
}

#div2 {
    overflow-wrap:break-word;
}
```

HTML

```
<div id="div1">
      <strong>#div1</strong>: Small words are displayed normally, but a long word like <span
style="red;">supercalifragilisticexpialidocious</span> is too long so it will overflow past
the edge of the line-break
</div>
```

```
<div id="div2">
      <strong>#div2</strong>: Small words are displayed normally, but a long word like <span
style="red;">supercalifragilisticexpialidocious</span> will be split at the line break and
continue on the next line.
</div>
```



overflow-wrap -	
normal	よりもいは、のオーバーフローをします。
break-word	にじてをのにします。
inherit	このプロパティののをします。

オーバーフロー

HTML

```
<div>
     Even if this div is too small to display its contents, the content is not clipped.
</div>
```

CSS

```
div {
     width:50px;
     height:50px;
     overflow:visible;
}
```

コンテンツはクリップされず、コンテナサイズをえているはコンテンツボックスのにレンダリングされます。

オーバーフローでされたブロックコンテキスト

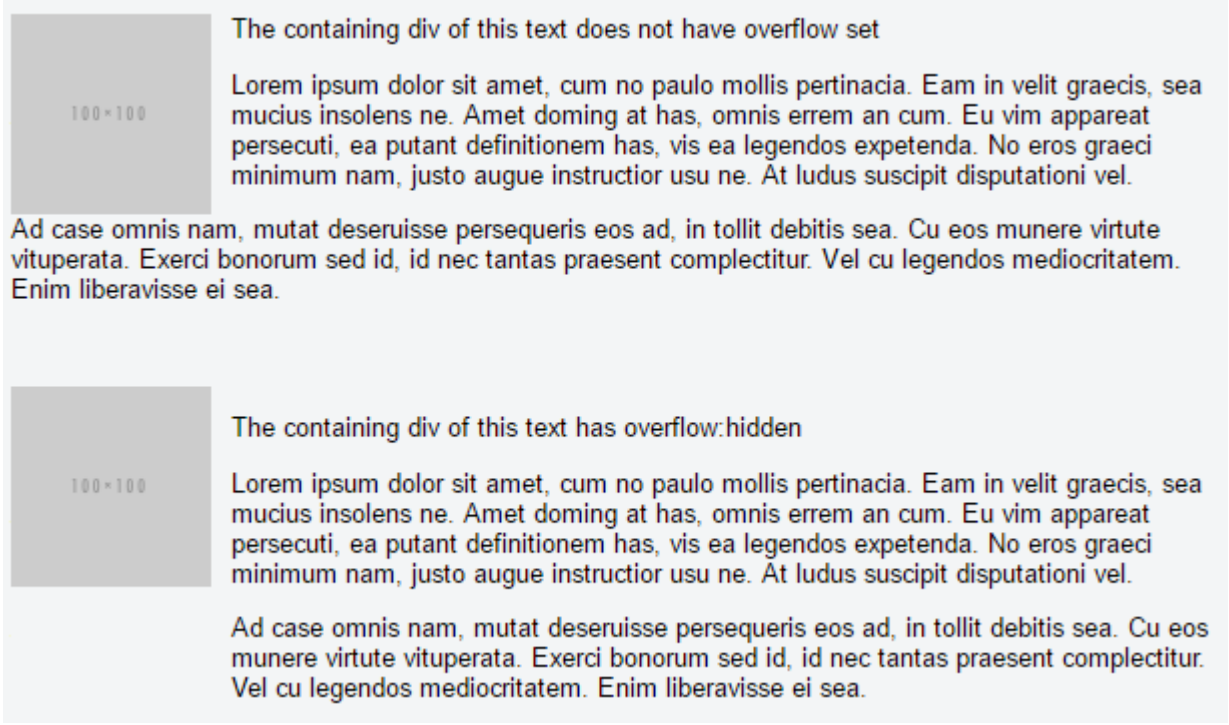
`overflow`プロパティを`visible`となるですると、しいブロックコンテキストがされます 。これは、のにあるブロックをさせるにです。

CSS

```
img {
    float:left;
    margin-right: 10px;
}
 div {
     overflow:hidden; /* creates block formatting context */
}
```

HTML

```
<img src="http://placehold.it/100x100">
<div>
  <p>Lorem ipsum dolor sit amet, cum no paulo mollis pertinacia.</p>
  <p>Ad case omnis nam, mutat deseruisse persequeris eos ad, in tollit debitis sea.</p>
</div>
```



このは、 `overflow`プロパティがされたdivのがイメージとどのようにするかをしています。

overflow-xおよびoverflow-y

これらの2つのプロパティは、 `overflow`プロパティとにし、じをけれます。 `overflow-x`パラメータは、xまたはからのでのみします。 `overflow-y`は、yまたはからのでします。

HTML

```
<div id="div-x">
    If this div is too small to display its contents,
    the content to the left and right will be clipped.
</div>

<div id="div-y">
    If this div is too small to display its contents,
    the content to the top and bottom will be clipped.
</div>
```

CSS

```
div {
    width: 200px;
    height: 200px;
}

#div-x {
    overflow-x: hidden;
}

#div-y {
    overflow-y: hidden;
}
```

オンライ ンでオーバーフローをむ https://riptutorial.com/ja/css/topic/4947/オーバーフロー

12: オブジェクトのフィットと

プロパティ`object-fit`および`object-position`は、Internet Explorerではサポートされていません。

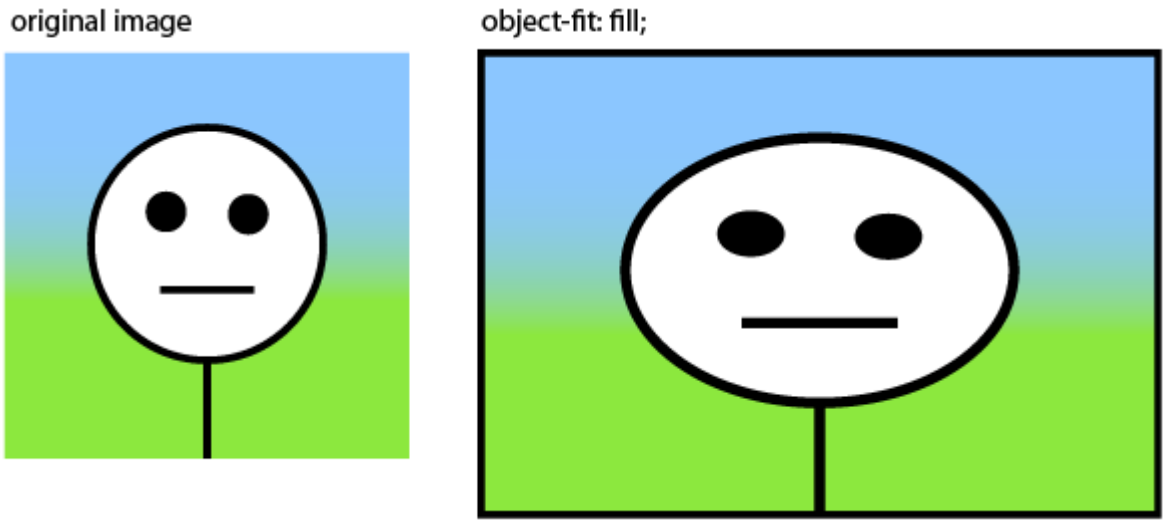
Examples

オブジェクトフィット

object-fitプロパティは、がさとがされたボックスにどのようにまるかをします。、イメージやビデオにされるオブジェクトフィットは、の5つのをけれます。

フィル

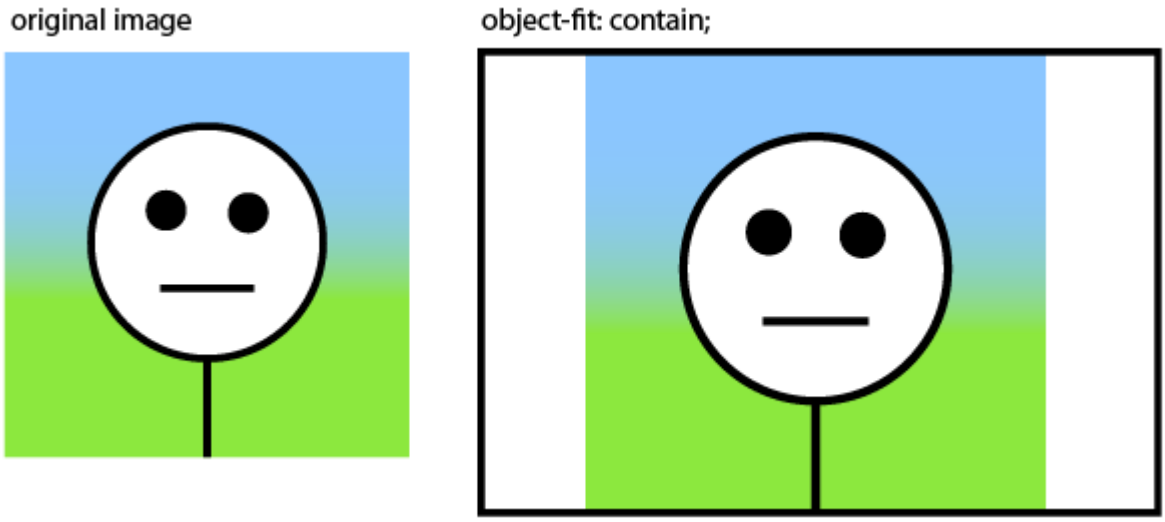
```
object-fit:fill;
```



ののアスペクトになく、をしてコンテンツボックスにわせます。

む

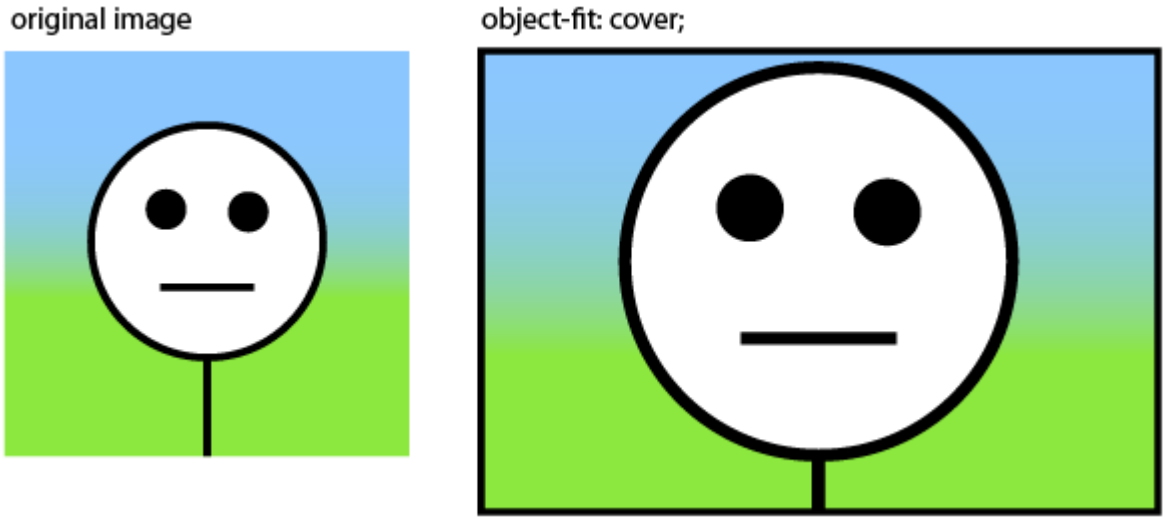
```
object-fit:contain;
```



Containは、のアスペクトをしながら、をボックスのさまたはにわせます。

カバー

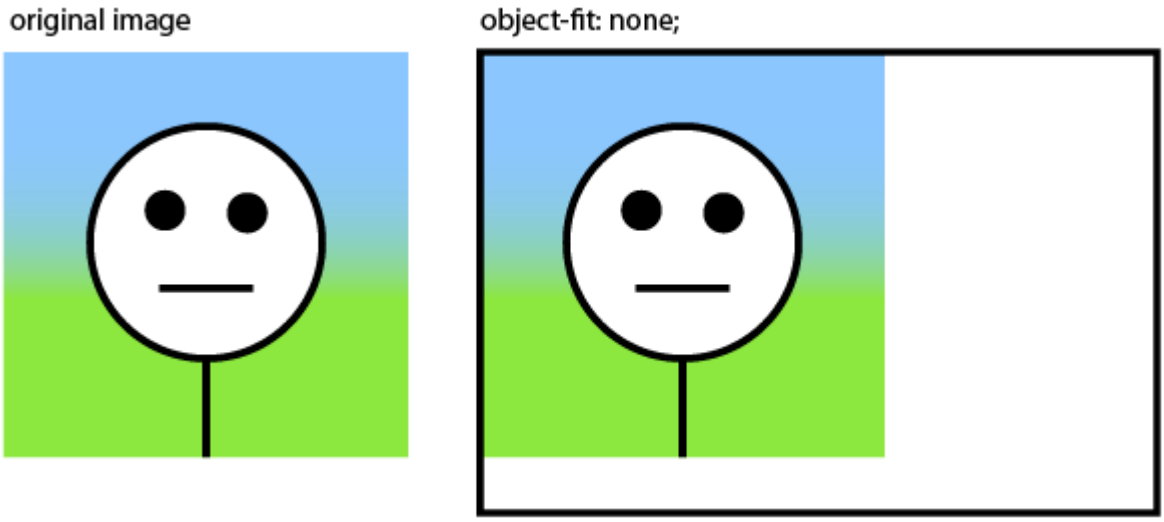
```
object-fit:cover;
```



Coverはをボックスでりつぶします。イメージのアスペクトはされますが、イメージはボックスのにわせてりられます。

し

```
object-fit:none;
```

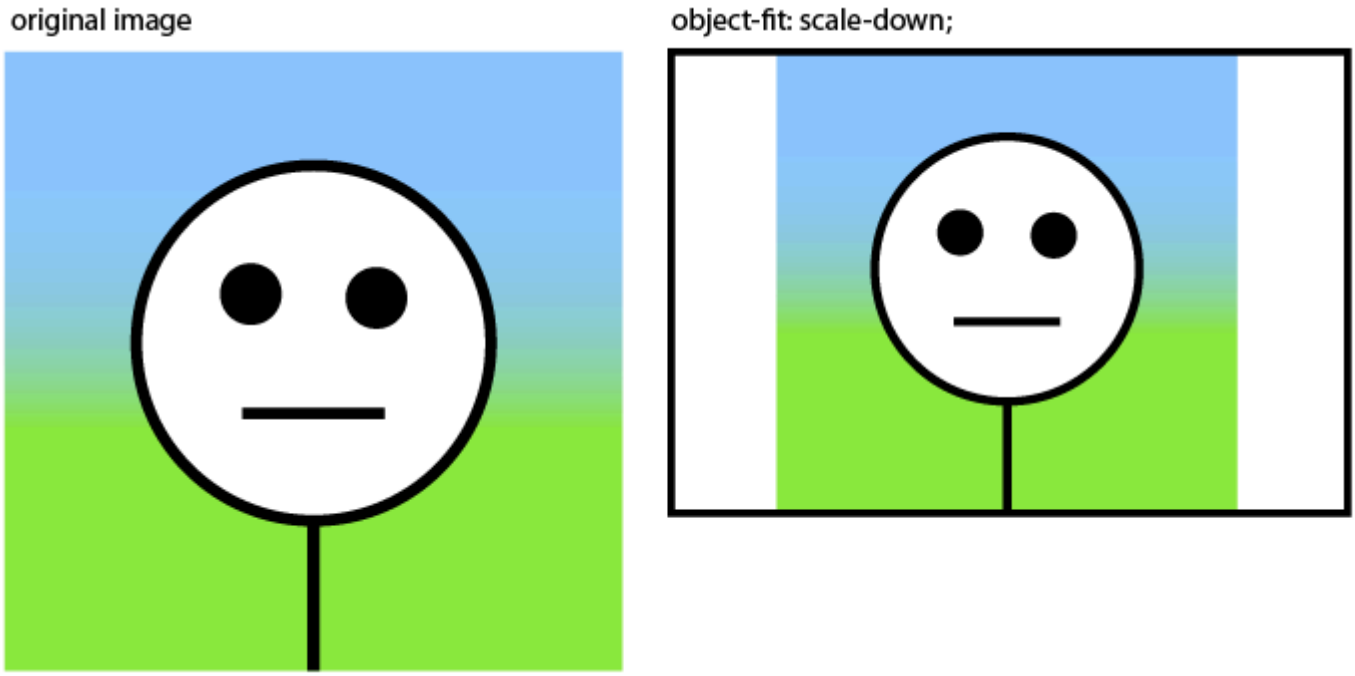


Noneはボックスのサイズをし、サイズされません。

スケールダウン

```
object-fit:scale-down;
```

Scale-downは、オブジェクトのサイズを`none`または`none`と`contain`。どちらのオプションをしたでも、サイズがさくなります。



オンラインでオブジェクトのフィットとをむ https://riptutorial.com/ja/css/topic/5520/オブジェクトのフィットと

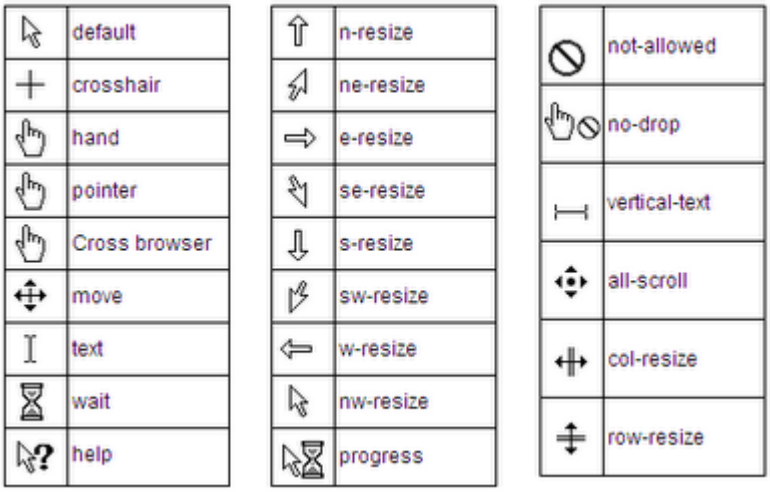
13: カーソルのスタイル

- カーソル|デフォルト|なし|コンテキストメニュー|ヘルプ|ポインタ||つ|セル| ||テキスト|きテキスト|エイリアス|コピー||ノードロップ| |リサイズ| n-resize | ne-resize | nw-resize | s-resize | se-resize | sw-resize | w-resize | ew-resize | ns-resize | nesw-resize | nwse-resize | col-resize |サイズ|すべてのスクロール| |ズームアウト| |つかむ;

Examples

カーソルタイプの

```
cursor: value;
```



し	にはカーソルはされません
オート	デフォルト。ブラウザがカーソルをする
けて	カーソルはヘルプがであることをします
つ	カーソルは、プログラムがビジーであることをします。
く	カーソルがかをすることをする
ポインタ	カーソルはポインタでリンクをします

ポインタイベント

pointer-eventsプロパティは、HTMLがマウス/タッチイベントにどのようにするかをできます。

```
.disabled {
```

```
  pointer-events: none;
}
```

このでは、

> 'none'は、されたHTML[1]のすべてのクリック、、およびカーソルオプションをします。

HTMIののなはのとおりです。

- ;
- する。

1. https://css-tricks.com/almanac/properties/p/pointer-events/

そののリソース

- https://developer.mozilla.org/en-US/docs/Web/CSS/pointer-events
- https://davidwalsh.name/pointer-events

キャレットカラー

キャレットカラーCSSプロパティは、ユーザーのやによってテキストやそののコンテンツがされるのポイントのインジケータであるキャレットのをします。

HTML

```
<input id="example" />
```

CSS

```
#example {
  caret-color: red;
}
```

リソース

- https://developer.mozilla.org/en-US/docs/Web/CSS/caret-color

オンラインでカーソルのスタイルをむ https://riptutorial.com/ja/css/topic/1742/カーソルのスタイル

14: カウンター

- カウンタセット[<カウンタ> <>] + |し
- カウンタリセット[<カウンタ> <>] + |し
- カウンタ[<カウンタ> <>] + |し
- counter<counter-name> [、<counter-style>]
- カウンタ<counter-name>、<connector-string> [、<counter-style>]

パラメーター

パラメータ	
カウンター—ネーム	これは、またはインクリメントまたはプリントするがあるカウンタのです。がむのカスタムをできます。
	このはオプションので、カウンタのにすると、カウンタの `counter-set` 、`counter-reset` プロパティまたはカウンタをインクリメントする `counter-increment` をし `counter-increment` 。
し	これは3つすべての `counter-*` プロパティのです。このを `counter-increment` すると、 `counter-increment` のがをけません。これをの2つにすると、カウンタはされません。
カウンター—スタイル	これは、カウンタをするがあるスタイルをします。これは、 `list-style-type` プロパティでサポートされているすべてのをサポートします。 `none` もされ `none` 、カウンタはくされません。
コネクタ	これは、2つのなるカウンタレベル "2.1.1"の "。"などののにするがあるをします。

カウンタはCSSのしいトピックではありません。これは、CSSレベル2には1のであり、したがってにいブラウザサポートをえています。

IE6とIE7をくすべてのブラウザは、CSSカウンタをサポートしています。

Examples

ローマのスタイリングをカウンターにする

CSS

```
body {
  counter-reset: item-counter;
}

.item {
  counter-increment: item-counter;
}

.item:before {
  content: counter(item-counter, upper-roman) ". "; /* by specifying the upper-roman as style
the output would be in roman numbers */
}
```

HTML

```
<div class='item'>Item No: 1</div>
<div class='item'>Item No: 2</div>
<div class='item'>Item No: 3</div>
```

のでは、がカウンタのスタイルをにしたので、カウンタのはの1,2,3のわりにI、II、IIIローマとしてされます。

CSSカウンタをしてにをける

CSS

```
body {
  counter-reset: item-counter; /* create the counter */
}
.item {
  counter-increment: item-counter; /* increment the counter every time an element with class
"item" is encountered */
}
.item-header:before {
  content: counter(item-counter) ". "; /* print the value of the counter before the header and
append a "." to it */
}

/* just for demo */

.item {
  border: 1px solid;
  height: 100px;
  margin-bottom: 10px;
}
.item-header {
  border-bottom: 1px solid;
  height: 40px;
```

```
  line-height: 40px;
  padding: 5px;
}
.item-content {
  padding: 8px;
}
```

HTML

```
<div class='item'>
  <div class='item-header'>Item 1 Header</div>
  <div class='item-content'>Lorem Ipsum Dolor Sit Amet....</div>
</div>
<div class='item'>
  <div class='item-header'>Item 2 Header</div>
  <div class='item-content'>Lorem Ipsum Dolor Sit Amet....</div>
</div>
<div class='item'>
  <div class='item-header'>Item 3 Header</div>
  <div class='item-content'>Lorem Ipsum Dolor Sit Amet....</div>
</div>
```

ののページのすべての「」として、そのヘッダのにアイテムのをするcontentの.item-headerエレメントのを:before。このコードのライブデモはこちらからできます。

CSSカウンタをしたマルチレベルのけの

CSS

```
ul {
  list-style: none;
  counter-reset: list-item-number; /* self nesting counter as name is same for all levels */
}
li {
  counter-increment: list-item-number;
}
li:before {
  content: counters(list-item-number, ".") " "; /* usage of counters() function means value of
counters at all higher levels are combined before printing */
}
```

HTML

```
<ul>
  <li>Level 1
    <ul>
      <li>Level 1.1
        <ul>
          <li>Level 1.1.1</li>
```

```
        </ul>
      </li>
    </ul>
  </li>
  <li>Level 2
    <ul>
      <li>Level 2.1
        <ul>
          <li>Level 2.1.1</li>
          <li>Level 2.1.2</li>
        </ul>
      </li>
    </ul>
  </li>
  <li>Level 3</li>
</ul>
```

は、CSSカウンタをしたマルチレベルのけのです。カウンター—のネスティングをしています。ネストは、あるがすでにされたのカウンタをっていて、のをしなければならないに、それをのカウンタのとしてするです。ここで、2レベルの`ul`にから`list-item-number`カウンタをしていますが、それの`list-item-number li` をしなければならないので、 `list-item-number[1]` for `list-item-number[0]` 1レベルのカウンタのにネストします。したがって、マルチレベルのけをします。

`counters()`は、をするときにすべてのカウンタののにくようにされているため、 `counter()` `counters()`のわりに`counters()`をしてをします。

オンラインでカウンター—をむ https://riptutorial.com/ja/css/topic/2575/カウンター—

15: カスケーディングと

CSSのは、いセットのなをができるようにすることでコードのさをし、のにしてオーバーライドすることをしています。

Examples

カスケーディング

カスケーディングとは、CSSスタイリングプロパティのなをするためににされます。また、CSSルールセットのをするためのメカニズムもします。

CSSみみ

スタイルはのソースからにみまれます。

1. ユーザーエージェントのスタイルシートブラウザベンダがするスタイル
2. ユーザースタイルシートユーザーがのブラウザーでしたのスタイル
3. スタイルシートはウェブページ/ウェブサイトのをする
    - たぶん1つの`.css`ファイル
    - HTMLドキュメントの`<style>`
4. インラインスタイルHTMLの`style`

ブラウザは、をレンダリングするときにするスタイルをルックアップします。

はどのようにされますか

1つのCSSルールセットだけがのスタイルをしようとすると、はなく、そのルールセットがされます。

するでのルールセットがつかったは、にSpecifictyルール、にカスケードルールをして、するスタイルをします。

1 - ルール

```
.mystyle { color: blue; }  /* specificity: 0, 0, 1, 0 */
div { color: red; }        /* specificity: 0, 0, 0, 1 */
```

```
<div class="mystyle">Hello World</div>
```

テキストはどのようなになりますか ホバーしてえをる

に、がされ、ものいものが「つ」。

2 - じセレクタをつカスケ―ド

のCSSファイル

```
.class {
  background: #FFF;
}
```

CSSHTMLファイル

```
<style>
.class {
  background: #000;
}
<style>
```

この、セレクタがの、カスケ―ドがし、にロ―ドされたものが「つ」とされます。

3 - ルールののカスケ―ドルール

```
body > .mystyle { background-color: blue; }   /* specificity: 0, 0, 1, 1 */
.otherstyle > div { background-color: red; }  /* specificity: 0, 0, 1, 1 */
```

```
<body class="otherstyle">
  <div class="mystyle">Hello World</div>
</body>
```

はどのようなですか

ルールをしたも、とのにはまだがあるため、カスケードルールはルールのにされます。カスケーディングは、じ.cssファイルか、スタイルソースのコレクションかになく、ルールのロードをべます。にロードされたものは、のものよりされます。この、 .otherstyle > divルールが「つ」。

のメモ

- セレクター―のはにされます。
- スタイルシートのブレークタイ。
- インラインスタイルはすべてをします。

な

!importantは、ルールにいをえることによって、スタイルシートののをきするためにされます。そのはのりです property : value !important;

```
#mydiv {
    font-weight: bold !important;    /* This property won't be overridden
                                        by the rule below */
}

#outerdiv #mydiv {
    font-weight: normal;            /* #mydiv font-weight won't be set to normal
                                       even if it has a higher specificity because
                                       of the !important declaration above */
}
```

!importantのをけることは、あなたのスタイルシートにをもたらすCSSルールのなれをげるため、にでないりくおめします。また、 !importantがのにしてじにされた、よりいをつがされることにすることがです。

!importantをできるいくつかのをにします。

- htmlの`style`のにされているのインラインスタイルによってルールをオーバーライドしてはならない。
- !importantをってのスタイルをきすることにより、フォントサイズのなど、Webアクセシビリティをよりにできるようにする。
- inspectをったテストとデバッグ。

- **W3C - 6**プロパティ、カスケード、およびのりて **- 6.4.2**なルール

セレクターの

々のCSSセレクタにはそれぞれのがあります。のセレクターは、のをさせる。セレクターは、 *A* 、 *B*および*C*の3つのなるグループの1つにされる。のセレクタシーケンスがのをするとき、ブラウザはながもいシーケンスによってされるスタイルをします。

グループ	からって	
A	IDセレクタ	`#foo`

B	クラスセレクタ
	セレクタ
	クラス
		`.bar`
		`[title]` 、 `[colspan="2"]`
		`:hover` 、 `:nth-child(2)`

C	タイプセレクタ	`div` 、 `li` `::before` 、 `::first-letter`

グループ*A*がもであり、にグループ*B* 、いでグループ*C*がきます 。

なセレクター `*` とコンビネーター `>`や`~`ようなはがありません。

1々なセレクターの

```
#foo #baz {}      /* a=2, b=0, c=0 */

#foo.bar {}       /* a=1, b=1, c=0 */

#foo {}           /* a=1, b=0, c=0 */

.bar:hover {}     /* a=0, b=2, c=0 */

div.bar {}        /* a=0, b=1, c=1 */

:hover {}         /* a=0, b=1, c=0 */

[title] {}        /* a=0, b=1, c=0 */

.bar {}           /* a=0, b=1, c=0 */

div ul + li {}    /* a=0, b=0, c=3 */

p::after {}       /* a=0, b=0, c=2 */

*::before {}      /* a=0, b=0, c=1 */

::before {}       /* a=0, b=0, c=1 */

div {}            /* a=0, b=0, c=1 */

* {}              /* a=0, b=0, c=0 */
```

2ブラウザがどのようにをするか

のCSSをしてみてください。

```
#foo {
  color: blue;
}

.bar {
  color: red;
  background: black;
}
```

ここでは、IDセレクタする`color` として、とするクラスセレクタ`color` として`background` ブラックを。

IDをつ`#foo`のクラス`.bar`のによってされるであろう。 IDセレクターはグループ*A*をし、クラスセレクター—はグループ*B*をする。 IDセレクタは、ののクラスセレクタをります。このため、`color:blue;` `#foo`セレクタと`background:black;` `.bar`セレクタからのがにされます。 IDセレクタのがいほど、ブラウザは`.bar`セレクタの`color`をします。

CSSのさまざまなをしてみましょう。

```
.bar {
  color: red;
  background: black;
}
```

```
.baz {
  background: white;
}
```

ここには2つのクラスセレクタがあります。つは`color`として`background`ブラックとして、が`background`として。

をつ`.bar`と`.baz`クラスはこれらののによってをけることになるが、しかし、々がっているは、そので`.bar`と`.baz`じグループBをっています。CSSのカスケードは、たちのためにこれをしますよう`.baz`にされて`.bar`、たちのがでわる`color`から`.bar`が、い`background`から`.baz`。

3をする

2からののは、たちをするためにすることができる`.bar`クラスセレクタの`color`がわりのものをいている`.baz`クラスセレクタ。

```
.bar {}         /* a=0, b=1, c=0 */
.baz {}         /* a=0, b=1, c=0 */
```

これをするもなは、`.bar`セレクタシーケンスにできるのセレクタをつけることです。たとえば、`.bar`クラスが`span`にのみされた、`.bar`セレクタを`span.bar`できます。これにより、`.baz`セレクタのをにするしいグループCのがえられます。

```
span.bar {}     /* a=0, b=1, c=1 */
.baz {}         /* a=0, b=1, c=0 */
```

しかし、`.bar`クラスをするでされるのセレクタをつけることはずしもではないかもしれません。このため、CSSではセレクタをしてをめることができます。わりに、ただの`.bar`、々はすることができます`.bar.bar`わりにしてくださいセレクタの、W3C。これは`.bar`クラスをつをしますが、グループBのを2にしました

```
.bar.bar {}     /* a=0, b=2, c=0 */
.baz {}         /* a=0, b=1, c=0 */
```

`!important`でインラインスタイルの

スタイルの`!important`フラグとHTML `style`でされたスタイルは、どのセレクタよりもいをつとなされます。これらがする、をけるスタイルは、そのになくのをにします。つまり、じにされるじプロパティにして`!important`フラグをむがあるをきます。そして、それらのにして、いにながされます。

らはをににするので、のユースケースでは`!important`のはうことになります。できるだけそれをうべきではありません。にってCSSコードをかつにつには、`!important`をするよりも、のセレクタのをめるがいでしょう。

これらのまれなのうちの1つは、 `!important`まされていませんが、したときにに1つのプロパティをオーバーライドするはずの`.hidden`または`.background-yellow`クラスのようなジェネリックヘルパークラスをするときです。それでも、あなたがしていることをるがあります。メンテナンスなCSSをくときににむことは、CSSに`!important`フラグをけることです。

のメモ

CSSのにするなは、グループ*A* 、 *B* 、および*C*のをみわせるがあるということです `a=1, b=5, c=1` => 151。これはてはまりません 。これがてはまる、グループ*B*または*C*セレクターの20をすることは、それぞれのグループ*A*または*B*セレクターをオーバーライドするのにである。 3つのグループはレベルのとみなすべきである。はのですることはできません。

CSSスタイルシートをするときは、できるだけをくつがあります。のメソッドをきするためにをしくするがあるは、それをくするためになりくします。のようなセレクタをつはありません。

```
body.page header.container nav div#main-nav li a {}
```

これにより、のがよりになり、そのCSSページがされます。

セレクターのはここでできます

よりなの

```
div {
    font-size: 7px;
    border: 3px dotted pink;
    background-color: yellow;
    color: purple;
}

body.mystyle > div.myotherstyle {
    font-size: 11px;
    background-color: green;
}

#elmnt1 {
    font-size: 24px;
    border-color: red;
}

.mystyle .myotherstyle {
    font-size: 16px;
    background-color: black;
    color: red;
}
```

```
<body class="mystyle">
    <div id="elmnt1" class="myotherstyle">
        Hello, world!
    </div>
</body>
```

どのような、、フォントサイズがテキストになりますか

フォントサイズ

`font-size: 24; #elmnt1`ルールセットはの`<div>`にしてもいをつため、ここのすべてのプロパティがされます。

`border: 3px dotted red;` 。 ボーダーカラーの`red`は`#elmnt1`ルールセットからされます。これはもいをつためです。 border、border-thickness、border-styleののプロパティは、 `div`ルールセットからのものです。

`background-color: green;` 。 `background-color`は、 `div` 、 `body.mystyle > div.myotherstyle` 、および`.mystyle .myotherstyle`ルールセットでされます。は0、0、10,2,20,2,0であるため、のものが「つ」。

`color: red;` 。 は、 `div .mystyle .myotherstyle`と`.mystyle .myotherstyle`でされます。は0、2、0と ""のいをっています。

オンラインでカスケーディングとをむ https://riptutorial.com/ja/css/topic/450/カスケーディングと

16: カスタムプロパティ

き

CSSをすると、CSSでできるなをすることができます。

たとえば、でのをすることは、CSSでです。 CSSのでは、これはドキュメントでじのをもすることをします。 CSSをすると、のをにしてのですることができます。これにより、のCSSをするよりもにをできるようになります。

- のグローバルなをにするクラス* /
- - ; / *をする* /
- var - 、 デフォルト / *されたをデフォルトfallbackでう* /

CSSは、なとえられています。

ブラウザーのサポート/

Firefoxバージョン**31** + デフォルトで

Mozillaよりしい

Chromeバージョン**49** + デフォルトで 。

「このはにすることで、テストのためにクロームバージョン48でにすることができ*experimental Web Platform*をします。 *chrome://flags/* *Chrome*のアドレスバーに、このにアクセスするには。 」

IEサポートされていません。

エッジ

Safariバージョン**9.1**

Examples

```
:root {
  --red: #b00;
  --blue: #4679bd;
  --grey: #ddd;
}
.Bx1 {
  color: var(--red);
  background: var(--grey);
  border: 1px solid var(--red);
}
```

```
:root {
  --W200: 200px;
  --W10: 10px;
}
.Bx2 {
  width: var(--W200);
  height: var(--W200);
  margin: var(--W10);
}
```

カスケーディング

CSSはのプロパティとじようにカスケードし、にびしできます。

をすることができ、されたにもいをつのみがされます。

このHTMLをすると

```
<a class="button">Button Green</a>
<a class="button button_red">Button Red</a>
<a class="button">Button Hovered On</a>
```

たちはこのCSSをくことができます

```
.button {
  --color: green;
  padding: .5rem;
  border: 1px solid var(--color);
  color: var(--color);
}

.button:hover {
  --color: blue;
}

.button_red {
  --color: red;
}
```

そして、このをる

/

けCSSにをけるときには、のCSSプロパティとじようにとダッシュだけがまれていますのさ、-moz-box-sizingがダブルダッシュ - でまります

```
//These are Invalids variable names
--123color: blue;
```

```
--#color: red;
--bg_color: yellow
--$width: 100px;

//Valid variable names
--color: red;
--bg-color: yellow
--width: 100px;
```

CSSはとをします。

```
/* The variable names below are all different variables */
--pcolor: ;
--Pcolor: ;
--pColor: ;
```

と

```
/* Invalid */
    --color:;

/* Valid */
  --color: ; /* space is assigned */
```

```
    /* Invalid - CSS doesn't support concatenation*/
    .logo{
        --logo-url: 'logo';
        background: url('assets/img/' var(--logo-url) '.png');
    }

    /* Invalid - CSS bug */
    .logo{
        --logo-url: 'assets/img/logo.png';
        background: url(var(--logo-url));
    }

    /* Valid */
    .logo{
        --logo-url: url('assets/img/logo.png');
        background: var(--logo-url);
    }
```

をするときはしてください

```
    /* Invalid */
    --width: 10;
    width: var(--width)px;

    /* Valid */
    --width: 10px;
    width: var(--width);

    /* Valid */
    --width: 10;
    width: calc(1px * var(--width)); /* multiply by 1 unit to convert */
    width: calc(1em * var(--width));
```

メディアクエリをする

メディアクエリのをし、それらがされているであればどこでもカスケードすることができます。これはプリプロセッサではです。

ここで、メディアクエリは、になグリッドをするためにされるをします。

HTML

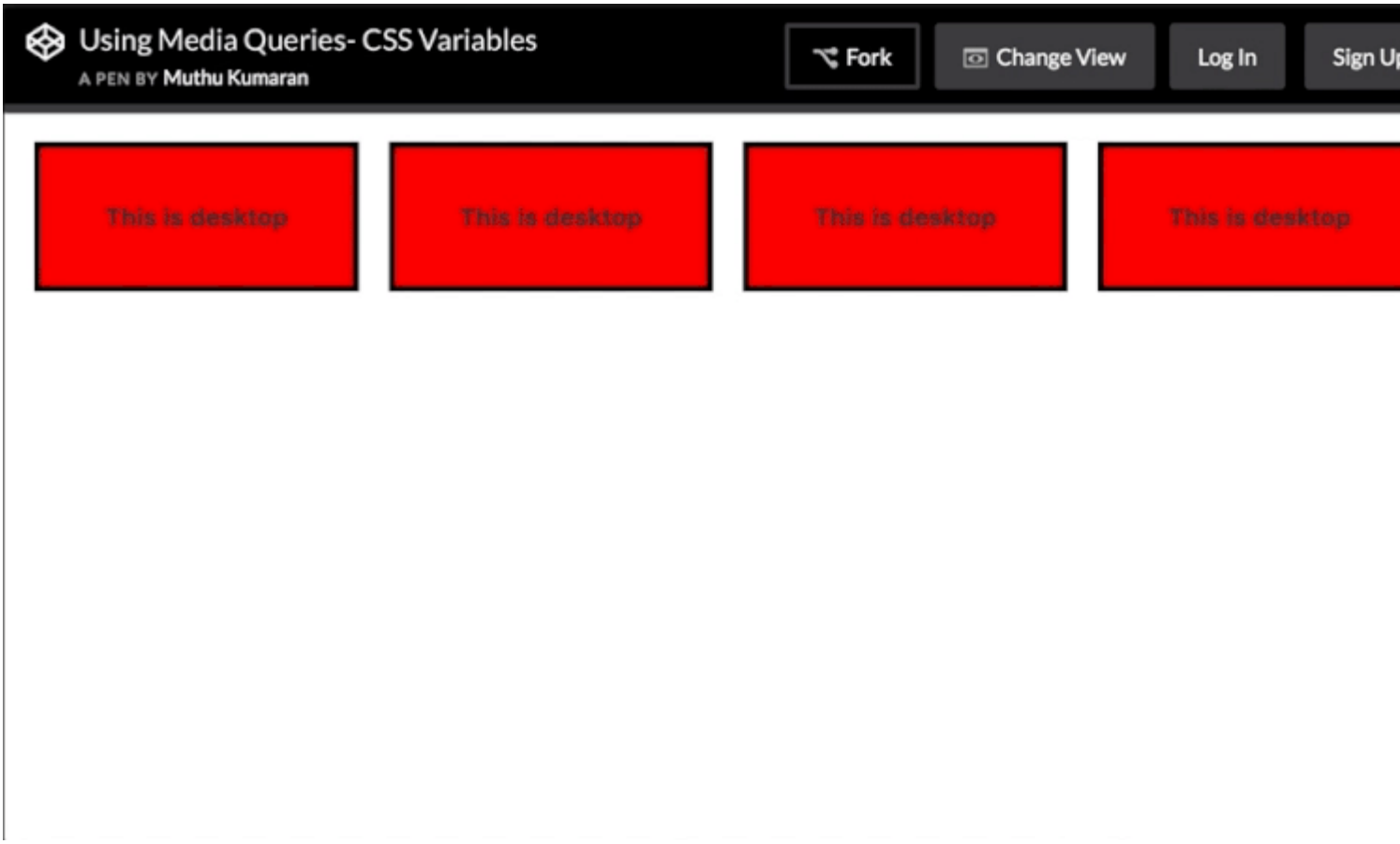
```
<div></div>
<div></div>
<div></div>
<div></div>
```

CSS

```
:root{
    --width: 25%;
    --content: 'This is desktop';
}
@media only screen and (max-width: 767px){
    :root{
        --width:50%;
        --content: 'This is mobile';
    }
}
@media only screen and (max-width: 480px){
    :root{
        --width:100%;
    }
}

div{
    width: calc(var(--width) - 20px);
    height: 100px;
}
div:before{
    content: var(--content);
}

/* Other Styles */
body {
    padding: 10px;
}

div{
    display: flex;
    align-items: center;
    justify-content: center;
    font-weight:bold;
    float:left;
    margin: 10px;
    border: 4px solid black;
    background: red;
}
```

このCodePenデモでウィンドウのサイズをすることができます

のサイズのアニメーションされたスクリーンショットです



オンラインでカスタムプロパティをむ https://riptutorial.com/ja/css/topic/1755/カスタムプロパティ

--

17: グリッド

き

グリッドレイアウトは、Webページのコンテンツをにとにすることをにする、しいなCSSレイアウトシステムです。

CSSグリッドレイアウトモジュールレベル1は、201699、W3Cのです。テスト https://www.w3.org/Style/CSS/current-workにあるとみなされます。

201773、MicrosoftのInternet Explorer 10および11およびEdgeブラウザは、ベンダーのプレフィックスをしてバージョンののみをサポートしています。

Examples

な

プロパティ	な
	グリッド/インライングリッド

CSSグリッドはプロパティとしてされています。これは、とそののにのみされます。

のマークアップをえてみましょう。

```
<section class="container">
  <div class="item1">item1</div>
  <div class="item2">item2</div>
  <div class="item3">item3</div>
  <div class="item4">item4</div>
</section>
```

のマークアップをグリッドとしてするもなは、に`display`プロパティを`grid`すること`grid`。

```
.container {
  display: grid;
}
```

しかし、これをうと、ずすべてのがいになりうようになります。これは、たちがどのようにをグリッドにするかをらないためです。しかし、にえます。

に、グリッドの`.container`をするとのをえるがあります。グリッド`grid-columns`と`grid-rows`プロパティをしてこれをうことができますにしてください。

```
.container {
```

```
  display: grid;
  grid-columns: 50px 50px 50px;
  grid-rows: 50px 50px;
}
```

しかし、にをえるがあるので、それはまだたちをけません。これをうには、 `grid-row`にかれているをす`grid-row`と`grid-column`をします。

```
.container .item1 {
  grid-column: 1;
  grid-row: 1;
}
.container .item2 {
  grid-column: 2;
  grid-row: 1;
}
.container .item3 {
  grid-column: 1;
  grid-row: 2;
}
.container .item4 {
  grid-column: 2;
  grid-row: 2;
}
```

にとのをえることによって、コンテナののをします。

JSFiddleのをてください 。あなたはそれをさせるために、これらは、ベンダープレフィックスをつグリッドレイアウトサポートするのブラウザであるようIE10、IE11またはエッジでこれをするがあります`-ms-`によるとクローム、オペラとFirefoxでまたはフラグをcaniuseために、それらとにテストする。

オンラインでグリッドをむ https://riptutorial.com/ja/css/topic/2152/グリッド

18: クリッピングとマスキング

- クリッピング
- clip-path<クリップ> | [<> || <clip-geometry-box>] |し
- マスキング
- マスク[なし| <マスクリファレンス>]
- マスクモード[<mask-mode>]
- マスクリピート[<repeat-style]
- マスク[<position>]
- マスククリップ[<ジオメトリボックス> |クリップなし]
- マスク[<geometry-box>]
- マスクサイズ[<bg-size>]
- マスク[<compositing-operator>]
- mask[<mask-reference> <masking-mode> || <position> [/ <bg-size>] || <リピートスタイル> || <ジオメトリボックス> || [<ジオメトリボックス> |クリップなし] || <compositing-operator>]

パラメータ

パラメータ	
クリップソース	インラインSVGまたはクリップパスのをむファイルのSVGをすURL。
	`inset()` 、 `circle()` 、 `ellipse()`または`polygon()`いずれかを`polygon()`ます。これらのの1つをして、クリッピングパスがされます。これらのシェイプは、 のシェイプとまったくじようにします
クリップジオメトリボックス	これは、 `content-box` 、 `padding-box` 、 `border-box` 、 `margin-box` 、 `fill-box` 、 `stroke-box` 、 `view-box`をとしてつことができます。 <basic-shape>にをせずにこれをすると、するボックスのがクリッピングのパスとしてされます。 <basic-shape>とともにすると、これはシェイプのボックスとしてします。
マスク	これは、マスクイメージソースへのURLまたはイメージでなくて`none` 。
リピートスタイル	これは、XとYでマスクをどのようにりすかタイルするかをします。サポートされているは、 `repeat-x` 、 `repeat-y` 、 `repeat` 、 `space` 、 `round` 、 `no-repeat`です。
マスクモード	`alpha`または`luminance`または`auto`ことができ、マスクをアルファマスクまたはルミナンスマスクとしてうべきかどうかをします。がされておらず、マスクがダイレクト･イメージの、アルファ･マスクまたはとなされます。マスクがURLのはルミナンス･マスクとなされます。

パラメータ	
ポジション	これは、マスクレイヤーのをし、 `background-position`プロパティーとがています。は、1つの `top` 、 `10%` または2 `top right` 、 `50% 50%` でできます。
ジオメトリボックス	プロパティにじて、 マスクをクリッピングするボックス マスクりつぶし または マスクの マスクめ のとしてするボックスをします。なのリストは、 `content-box` 、 `padding-box` 、 `border-box` 、 `margin-box` 、 `fill-box` 、 `stroke-box` 、 `view-box`です。これらのがどのようにするかについてのなは、 W3Cにされています。
bg-size	これはマスクレイヤーのサイズをし、 `background-size`とじをっています。は、さまたはパーセンテージ、またはautoまたはcoverまたはcontainsです。さ、パーセンテージ、およびは、のとして、またはの1つとしてすることができます。
オペレータ	これは、レイヤーごとの`add` 、 `subtract` 、 `exclude` 、 `multiply`うちのいずれか1つで、このレイヤーでするのタイプをそのレイヤーでします。のについては、W3Cのをしてください 。

CSSのクリッピングとマスキングはにしいコンセプトなので、これらのプロパティのブラウザサポートはかなりいです。

マスク

716のように、Chrome、Safari、Operaはこれらのプロパティを`-webkit-`でサポートして`-webkit-`ます。

Firefoxはをとしませんが、SVG `mask`でするにのみマスクをサポートします。インラインSVG `mask`の、は`mask: url(#msk)`、SVGファイルの`mask`をする、は`mask: url('yourfilepath/yourfilename.svg#msk')`です。のの`#msk`は、されている`mask`の`id`をします。 このにされているように、のところ、Firefoxでは`mask`プロパティの`mask` `mask-reference`のパラメータはサポートされていません。

Internet ExplorerおよびEdgeはまだこのプロパティをサポートしていません。

`mask-mode`プロパティは、きまたはなしのブラウザではサポートされていません。

クリップパス

716のChromeSafari、Operaでは、なシェイプ `circle` 、 `polygon` またはインラインSVGの`url(#clipper)`をしてパスをするときに、 `clip-path`サポートしてい`clip-path`。SVGファイルのであるシェイプにづいてクリッピングをサポートしていません。また、 `-webkit`がです。

Firefoxは`clip-path`の`url()`のみをサポートしてい`clip-path`、Internet ExplorerおよびEdgeはサポートしていません。

Examples

クリッピングポリゴン

CSS

```
div{
  width:200px;
  height:200px;
  background:teal;
  clip-path: polygon(0 0, 0 100%, 100% 50%); /* refer remarks before usage */
}
```

HTML

```
<div></div>
```

のでは、200×200のをにクリップするためにのクリッピングパスがされています。シェイプはです。パスがまるつまり、のは`0 0` - ボックスのを`0 100%`にし`0 100%`これはボックスのです。に`100% 50%`になります。これはボックスのだけです。これらのパスはしますつまり、がになります。なはのになります。

これは、またはグラデーションをつでとしてすることもできます。

ビューの

クリッピングサークル

CSS

```
div{
  width: 200px;
  height: 200px;
  background: teal;
  clip-path: circle(30% at 50% 50%); /* refer remarks before usage */
}
```

HTML

```
<div></div>
```

このは、divをにクリップするをしています。は、ボックスのにをつボックスのにづいてが30のにクリップされます。ここでは、<clip-geometry-box>つまり、ボックスがされないため、の`border-box`がボックスとしてされます。

のはと`(x,y)`のをつがあります

```
circle(radius at x y)
```

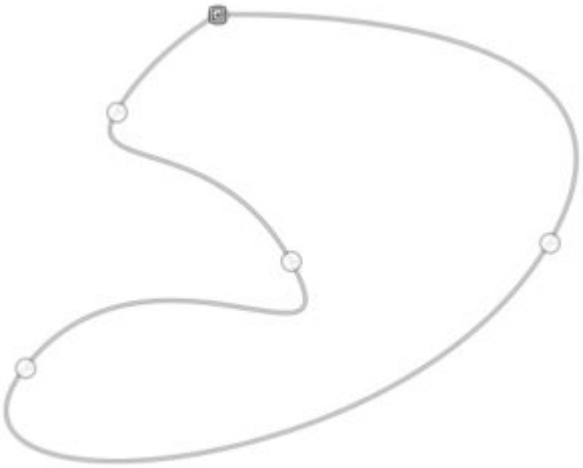
ビューの

クリッピングとマスキングと

クリッピングとマスキングでは、ののをまたはにすることができます。どちらもHTMLにできます。

クリッピング

クリップはベクトルパスです。このパスのでは、はになり、はになります。したがって、に`clip-path`プロパティをすることができます。 SVGにもするすべてのグラフィカルは、ここでパスをするとしてできます。は`circle()` 、 `polygon()`または`ellipse()`です。



```
clip-path: circle(100px at center);
```

は、のにされ、が100ピクセルののにのみされます。

マスキング

マスクはクリップとていますが、パスをするのではなく、のにどのレイヤーをマスクするかをします。このマスクは、に2つのとからなるとしてすることができます。

ルミナンスマスク はがでがであることをしますが、なのもあり、スムーズなトランジションがです。

Alpha Mask マスクのでのみがになります。

このは、えばマスクとしてして、をからへ、およびからににスムーズにさせることができます。

`mask`プロパティをすると、マスクタイプとレイヤーとしてされるイメージをできます。

```
mask: url(masks.svg#rectangle) luminance;
```

`masks.svg`されている`rectangle`は、のルミナンスマスクとしてされます。

イメージをからにフェードするなマスク

CSS

```
div {
  height: 200px;
  width: 200px;
  background: url(http://lorempixel.com/200/200/nature/1);
  mask-image: linear-gradient(to right, white, transparent);
}
```

HTML

```
<div></div>
```

のでは、としてイメージをつがあります。イメージにされたマスクCSSをは、からにフェードアウトしているかのようにえます。

マスキングは、からをマスクとする`linear-gradient`をすることによってされます。これはアルファマスクなので、マスクがなでははになります。

マスクなしの

マスクき

でべたように、のは、 `-webkit`をけてしたにのみChrome、Safari、Operaで`-webkit`ます。この

`linear-gradient` マスクは、Firefoxではまだサポートされていません。

マスクをしてののをカットする

CSS

```
div {
  width: 200px;
  height: 200px;
  background: url(http://lorempixel.com/200/200/abstract/6);
  mask-image: radial-gradient(circle farthest-side at center, transparent 49%, white 50%); /*
check remarks before using */
}
```

HTML

のでは、 `radial-gradient` をしてにをし、これをマスクとしてして、のからりられたのをします。

マスクなしの

マスクき

マスクをしてなのをする

CSS

```
div { /* check remarks before usage */
  height: 200px;
  width: 400px;
  background-image: url(http://lorempixel.com/400/200/nature/4);
  mask-image: linear-gradient(to top right, transparent 49.5%, white 50.5%), linear-
gradient(to top left, transparent 49.5%, white 50.5%), linear-gradient(white, white);
  mask-size: 75% 25%, 25% 25%, 100% 75%;
  mask-position: bottom left, bottom right, top left;
  mask-repeat: no-repeat;
}
```

HTML

```
<div></div>
```

のでは、3つの`linear-gradient`なにかれたときにコンテナのサイズの100x 100をカバーするがマスクとしてされ、のになのカットがされます。

マスクなしの

マスクき

オンラインでクリッピングとマスキングをむ https://riptutorial.com/ja/css/topic/3721/クリッピング

とマスキング

19: コメント

- / * Comment * /

- CSSのコメントはに`/*`まり、 `*/`
- コメントはネストできません

Examples

```
/* This is a CSS comment */
div {
   color: red; /* This is a CSS comment */
}
```

```
/*
    This
    is
    a
    CSS
    comment
*/
div {
   color: red;
}
```

オンラインでコメントをむ https://riptutorial.com/ja/css/topic/1625/コメント

20: コンテキストをブロックする

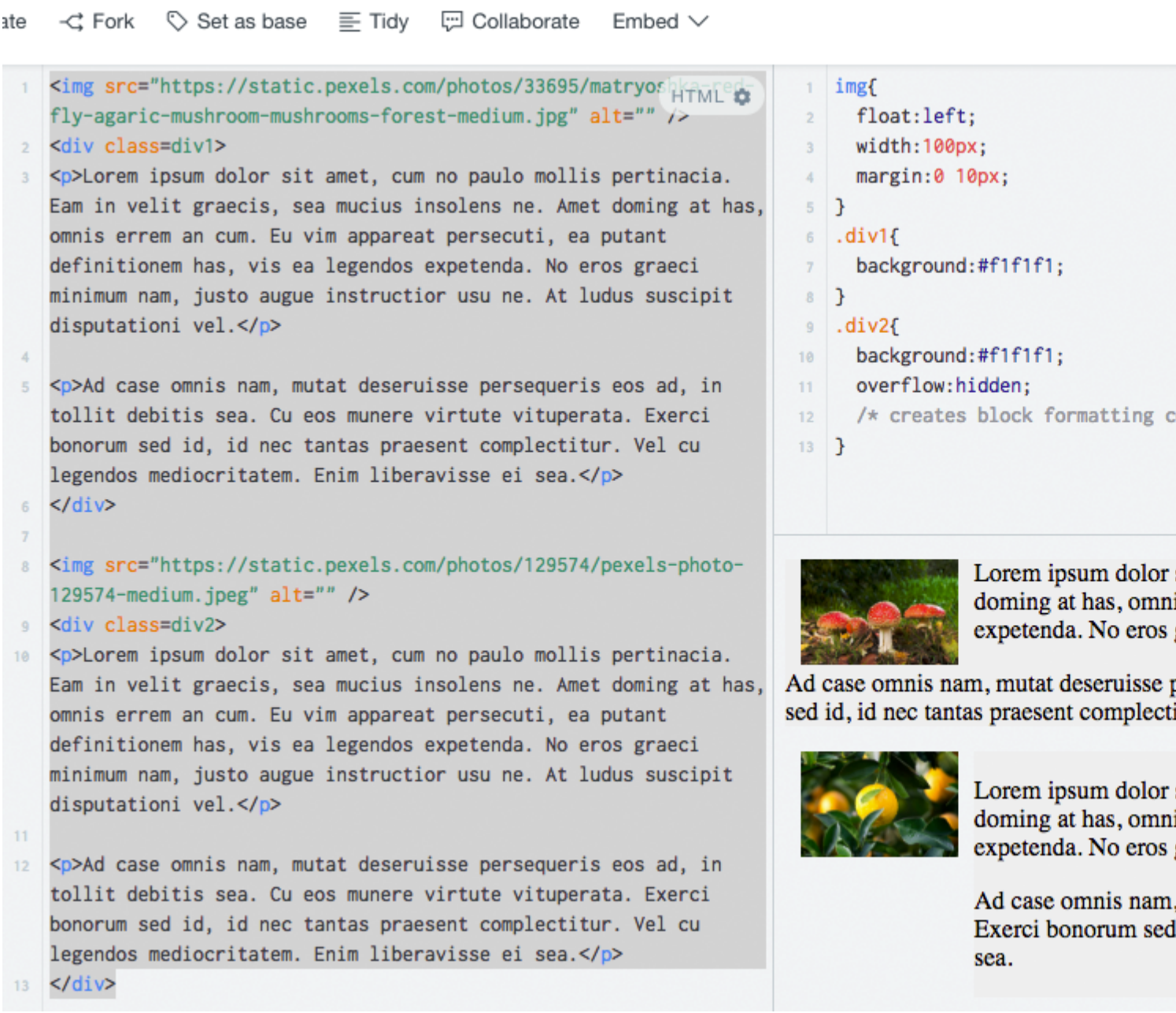
[ブロックコンテキストは、Webページのビジュアルcssレンダリングのです。ブロックボックスのレイアウトがし、フロートがするです。] [1]

[1] https : //developer.mozilla.org/en-US/docs/Web/Guide/CSS/Block_formatting_context MDN

Examples

オーバーフロープロパティをvisibleとなるでする

```
img{
  float:left;
  width:100px;
  margin:0 10px;
}
.div1{
  background:#f1f1f1;
  /* does not create block formatting context */
}
.div2{
  background:#f1f1f1;
  overflow:hidden;
  /* creates block formatting context */
}
```



https://jsfiddle.net/MadalinaTn/qkwwmu6m/2/

visibleデフォルトとなるをつoverflowプロパティをすると、しいブロックコンテキストがされます。これはにです。がスクロールとすると、にをラップします。

のがイメージとどのようにするかをすこのは 、 こののcss-tricks.comにています。

2 https : //developer.mozilla.org/en-US/docs/Web/CSS/overflow MDN

オンラインでコンテキストをブロックするをむ https://riptutorial.com/ja/css/topic/5069/コンテキストをブロックする

21: スタイルをリストする

- リストスタイルリストスタイル|リストスタイルの|リストスタイルのイメージ|の|する;

パラメーター

リストスタイルタイプ	リストマーカーのタイプ
リストスタイルの	マーカーをするをします。
リストスタイルイメージ	リストアイテムマーカーのタイプをする
	このプロパティをデフォルトにする
する	このプロパティをからします。

`list-style-type`はにはリストは`<li>` のみにされるプロパティですが、listタグ `<ol>`または`<ul>` にされることがよくあります。この、リストはプロパティをします。

Examples

きまたはの

けられていないリスト `<ul>` の`<li>`タグに

```
list-style: disc;                /* A filled circle (default) */
list-style: circle;              /* A hollow circle */
list-style: square;              /* A filled square */
list-style: '-';                 /* any string */
```

きリスト `<ol>` の`<li>`タグ

```
list-style: decimal;             /* Decimal numbers beginning with 1 (default) */
list-style: decimal-leading-zero;/* Decimal numbers padded by initial zeros (01, 02, 03, … 10)
*/
list-style: lower-roman;         /* Lowercase roman numerals (i., ii., iii., iv., ...) */
list-style: upper-roman;         /* Uppercase roman numerals (I., II., III., IV., ...) */
list-style-type: lower-greek;    /* Lowercase roman letters (α., β., γ., δ., ...) */
list-style-type: lower-alpha;    /* Lowercase letters (a., b., c., d., ...) */
list-style-type: lower-latin;    /* Lowercase letters (a., b., c., d., ...) */
list-style-type: upper-alpha;    /* Uppercase letters (A., B., C., D., ...) */
list-style-type: upper-latin;    /* Uppercase letters (A., B., C., D., ...) */
```

```
list-style: none;                /* No visible list marker */
list-style: inherit;             /* Inherits from parent */
```

の

リストは<li>まれる または のでされます。リストアイテムとコンテナのには、ドキュメントのリストアイテムコンテンツのなにをえるとパディングがあります。マージンとパディングのデフォルトは、ブラウザごとになるがあります。クロスブラウザとじレイアウトをるためには、これらをにするがあります。

リストは、マーカーマーカーをむ'マーカーボックス'をします。このボックスは、リストアイテムボックスのまたはにできます。

```
list-style-position: inside;
```

にきをし、コンテンツをにじてにします。

```
list-style-position: outside;
```

のにきをきます。のパディングになスペースがない、マーカボックスはページからはみしてもにびます。

insideとoutsideをする jsfiddle

き/の

によっては、リストにはきやはされません。そのは、マージンとパディングをずしてください。

```
<ul>
    <li>first item</li>
    <li>second item</li>
</ul>
```

CSS

```
ul {
    list-style-type: none;
}
li {
    margin: 0;
    padding: 0;
}
```

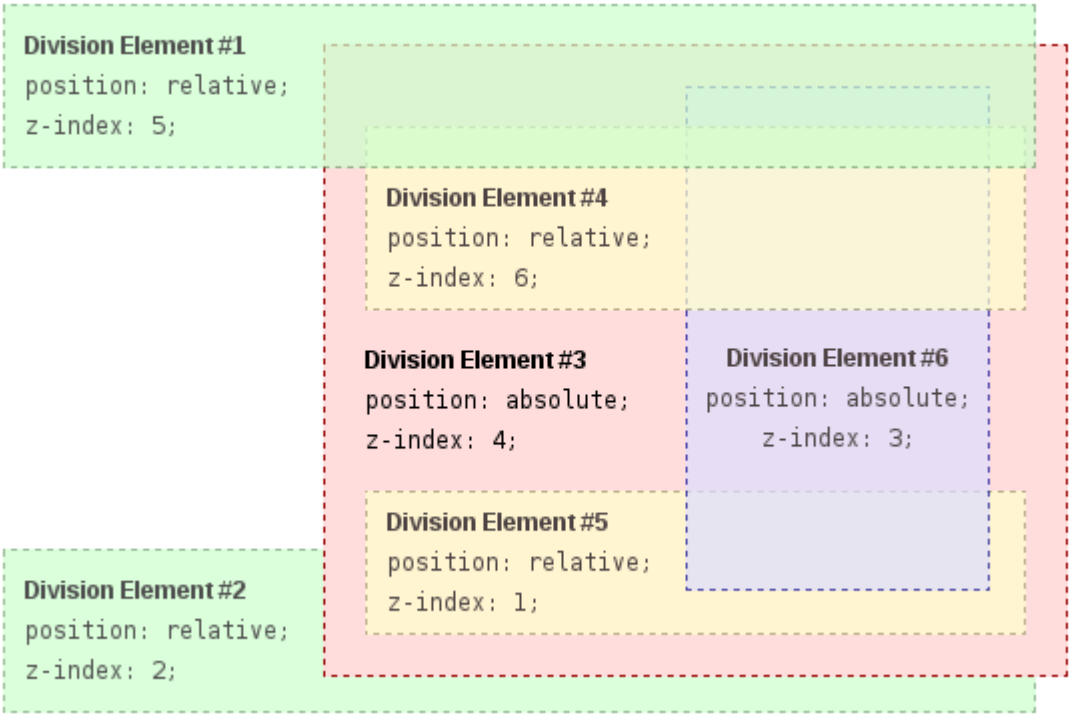
オンラインでスタイルをリストするをむ https://riptutorial.com/ja/css/topic/4215/スタイルをリストする

22: スタッキングコンテキスト

Examples

スタッキングコンテキスト

このでは、されたすべてのは、そのとZ-インデックスのために、のスタックコンテキストをします。スタッキングコンテキストのは、のようにされています。



- ルート
  - DIV1
  - DIV2
  - DIV3
  - DIV4
  - DIV5
  - DIV6

DIV4、DIV5、DIV6はDIV3のであるため、これらののみねはDIV3でにされることにすることがです。 DIV3のスタッキングおよびレンダリングがすると、DIV3が、そののDIVにしてルートにスタッキングするためにされる。

HTML

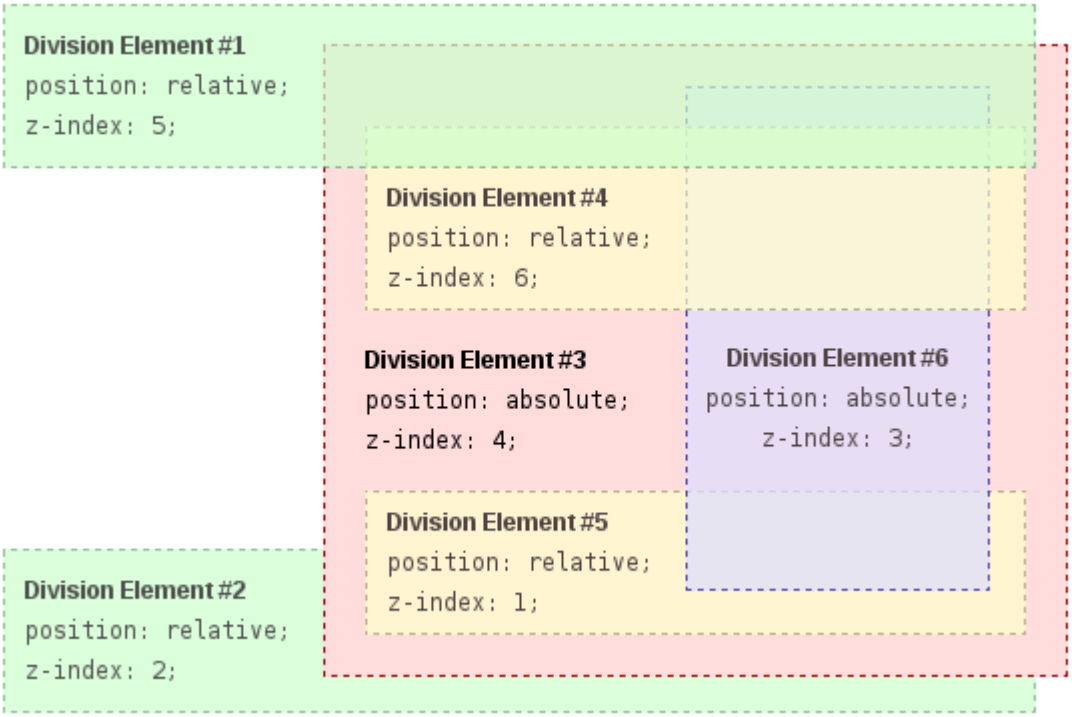
```
<div id="div1">
  <h1>Division Element #1</h1>
  <code>position: relative;<br/>
```

```
  z-index: 5;</code>
</div>
<div id="div2">
  <h1>Division Element #2</h1>
  <code>position: relative;<br/>
  z-index: 2;</code>
</div>
<div id="div3">
  <div id="div4">
    <h1>Division Element #4</h1>
    <code>position: relative;<br/>
    z-index: 6;</code>
  </div>
  <h1>Division Element #3</h1>
  <code>position: absolute;<br/>
  z-index: 4;</code>
  <div id="div5">
    <h1>Division Element #5</h1>
    <code>position: relative;<br/>
    z-index: 1;</code>
  </div>
  <div id="div6">
    <h1>Division Element #6</h1>
    <code>position: absolute;<br/>
    z-index: 3;</code>
  </div>
</div>
```

CSS

```
* {
    margin: 0;
}
html {
    padding: 20px;
    font: 12px/20px Arial, sans-serif;
}
div {
    opacity: 0.7;
    position: relative;
}
h1 {
    font: inherit;
    font-weight: bold;
}
#div1,
#div2 {
    border: 1px dashed #696;
    padding: 10px;
    background-color: #cfc;
}
#div1 {
    z-index: 5;
    margin-bottom: 190px;
}
#div2 {
    z-index: 2;
}
#div3 {
    z-index: 4;
```

```
    opacity: 1;
    position: absolute;
    top: 40px;
    left: 180px;
    width: 330px;
    border: 1px dashed #900;
    background-color: #fdd;
    padding: 40px 20px 20px;
}
#div4,
#div5 {
    border: 1px dashed #996;
    background-color: #ffc;
}
#div4 {
    z-index: 6;
    margin-bottom: 15px;
    padding: 25px 10px 5px;
}
#div5 {
    z-index: 1;
    margin-top: 15px;
    padding: 5px 10px;
}
#div6 {
    z-index: 3;
    position: absolute;
    top: 20px;
    left: 180px;
    width: 150px;
    height: 125px;
    border: 1px dashed #009;
    padding-top: 125px;
    background-color: #ddf;
    text-align: center;
}
```



https : //developer.mozilla.org/en-US/docs/Web/CSS/CSS_Positioning/Understanding_z_index/The_stacking_context

オンラインでスタッキングコンテキストをむ https://riptutorial.com/ja/css/topic/5037/スタッキングコンテキスト

23: セレクタ

き

CSSセレクタは、のHTMLをCSSスタイルのターゲットとしてします。このトピックでは、CSSセレクタがHTMLをターゲットとするについてします。セレクタは、、クラス、ID、とクラス、パターンなど、CSSでされる50のメソッドをくします。

- *id*
- 。 クラス
- クラス
- :: *pseudo-elementname*
- [*attr*] / *には*attr*があります。 * /
- [*attr* = " *value* "] / *は*attr*をち、そのはに " *value* "です。 * /
- [*attr* = " *value* "] / *は*attr*をち、そのはでされたときに " *value* "をみます 。 * /
- [*attr* | = " *value* "] / *は*attr*をち、そのはに " *value* "であるか 、は " *value* - "でまります。 * /
- [*attr* ^ = " *value* "] / *は*attr*をち、そのは " *value* "でまります 。 * /
- [*attr* $ = " *value* "] / *は*attr*をち、そのは " *value* "でわります 。 * /
- [*attr* * = " *value* "] / *は*attr*をち、そのには " *value* "がまれています 。 * /
-
- *

- にはあなたは、のコロンされます`::` のわりに、1つだけ`:` 。これはクラスをからするです。
- Internet Explorer 8のようないブラウザでは、のコロンをサポート`:`をするため。
- クラスとはなり、セレクタごとに1つのしかできません。セレクタのをすなセレクタのシーケンスのにれなければなりません W3Cのバージョンではセレクタごとにの。

Examples

セレクタ

アトリビュートセレクタは、をそれにじてするさまざまなタイプのでできます。されたまたはのをしてをします。

セレクタ1	した	を...	CSSバージョン
`[attr]`	`<div attr>`	アトリビュート`attr`	2
`[attr='val']`	`<div attr="val">`	`attr`が`val`	2
`[attr~='val']`	`<div attr="val val2`	`val`が	2

セレクタ1	した	を...	CSSバージョン
	val3">	でられたattrリスト	
[attr^='val']		attrのがvalまるところ	3
[attr$='val']		attrのがvalわるところ	3
[attr*='val']		attrはどこでもvalがまれています	3
[attr|='val']		attrのがにval、 またはvalでまりすぐに - U + 002D	2
[attr='val' i]		attrにvalがある、 valのをして	4 [2]

ノート

1. は、またはでむことができます。はわないかもしれませんが、CSSによればではありませんので、おめしません。
2. のモジュールにされているため、されたのCSS4はありません。しかし、 "レベル4"モジュールがあります。 ブラウザのサポートをしてください 。

[attribute]

されたをつをします。

```
div[data-color] {
  color: red;
}
```

```
<div data-color="red">This will be red</div>
<div data-color="green">This will be red</div>
<div data-background="red">This will NOT be red</div>
```

JSBinのライブデモ

[attribute="value"]

されたとをつをします。

```
div[data-color="red"] {
  color: red;
```

```
}
```

```
<div data-color="red">This will be red</div>
<div data-color="green">This will NOT be red</div>
<div data-color="blue">This will NOT be red</div>
```

JSBinのライブデモ

[attribute*="value"]

されたおよびをつをします。されたには、されたがどこでもとしてまれます。

```
[class*="foo"] {
  color: red;
}
```

```
<div class="foo-123">This will be red</div>
<div class="foo123">This will be red</div>
<div class="bar123foo">This will be red</div>
<div class="barfooo123">This will be red</div>
<div class="barfo0">This will NOT be red</div>
```

JSBinのライブデモ

[attribute~="value"]

えられたがでられたリストにれる、えられたとをつをします。

```
[class~="color-red"] {
  color: red;
}
```

```
<div class="color-red foo-bar the-div">This will be red</div>
<div class="color-blue foo-bar the-div">This will NOT be red</div>
```

JSBinのライブデモ

[attribute^="value"]

されたがでまる、されたとをつをします。

```
[class^="foo-"] {
  color: red;
}
```

```
<div class="foo-123">This will be red</div>
<div class="foo-234">This will be red</div>
<div class="bar-123">This will NOT be red</div>
```

JSBinのライブデモ

[attribute$="value"]

されたがされたでわる、されたとをつをします。

```
[class$="file"] {
  color: red;
}
```

```
<div class="foobar-file">This will be red</div>
<div class="foobar-file">This will be red</div>
<div class="foobar-input">This will NOT be red</div>
```

JSBinのライブデモ

[attribute|="value"]

されたとをつをします。のがされたまたはにされたの、 - U + 002D

```
[lang|="EN"] {
  color: red;
}
```

```
<div lang="EN-us">This will be red</div>
<div lang="EN-gb">This will be red</div>
<div lang="PT-pt">This will NOT be red</div>
```

JSBinのライブデモ

[attribute="value" i]

えられたとをつをします。のは、 `Value` 、 `VALUE` 、 `vAlUe`またはそののとをしないとしてすことができます。

```
[lang="EN" i] {
  color: red;
}
```

```
<div lang="EN">This will be red</div>
<div lang="en">This will be red</div>
<div lang="PT">This will NOT be red</div>
```

JSBinのライブデモ

セレクタの

0-1-0

クラスセレクタおよびクラスとじです。

```
*[type=checkbox] // 0-1-0
```

これは、セレクタをして、 IDセレクタでされたよりもいでIDでをできることにしてください。 `[id="my-ID"]`は`#my-ID`はいがはい。

については、「 セクション 」をしてください。

コンビネーション

セレクタ	
`div span`	セレクタ `<span>`は`<div>`です
`div > span`	セレクタすべての`<span>`は`<div>`のです
`a ~ span`	なセレクタ `<a>`のであるすべての`<span>`
`a + span`	するセレクタすべての`<span>`は`<a>`にあります

> セレクタは、ソースドキュメントののろにあるをとします。 CSSはそのカスケード、またはをとすることはできません。ただし、flex `order`プロパティをすると、のセレクタをビジュアルメディアでシミュレートできます 。

Descendant Combinator `selector selector`

なくとも1つのスペースでされるコンビネータ は、されたのであるをします。このコンビネータは、のすべてのをのからします。

```
div p {
  color:red;
}
```

```
<div>
  <p>My text is red</p>
  <section>
    <p>My text is red</p>
  </section>
</div>

<p>My text is not red</p>
```

JSBinのライブデモ

のでは、の2つの`<p>`はとも`<div>`であるためされています。

コンバイナ `selector > selector`

> コンビネータは、されたの 、またはのであるを、するためにされます。

```
div > p {
  color:red;
}
```

```
<div>
  <p>My text is red</p>
  <section>
    <p>My text is not red</p>
  </section>
</div>
```

JSBinのライブデモ

のCSSは からしたのであるため、の<p>のみをします。

2の<p>は のではないためされません。

する `selector + selector`

する + は、されたのにあるをします。

```
p + p {
  color:red;
}
```

```
<p>My text is not red</p>
<p>My text is red</p>
<p>My text is red</p>
<hr>
<p>My text is not red</p>
```

JSBinのライブデモ

のでは、の<p>のにある<p>のみをしています。

な `selector ~ selector`

な ~ は、されたにくすべてのをします。

```
p ~ p {
```

```
  color:red;
}
```

```
<p>My text is not red</p>
<p>My text is red</p>
<hr>
<h1>And now a title</h1>
<p>My text is red</p>
```

JSBinのライブデモ

のでは、すぐにあるかどうかにかかわらず、の<p>でまるすべての<p>をしています。

クラスセレクタ

クラスセレクタは、のクラスをつすべてのをします。たとえば、クラス.warningはの をします。

```
<div class="warning">
    <p>This would be some warning copy.</p>
</div>
```

クラスとタ―ゲットをよりにみわせることもできます。のをに、よりなクラスをしましょう。

CSS

```
.important {
    color: orange;
}
.warning {
    color: blue;
}
.warning.important {
    color: red;
}
```

HTML

```
<div class="warning">
    <p>This would be some warning copy.</p>
</div>

<div class="important warning">
    <p class="important">This is some really important warning copy.</p>
</div>
```

このでは、とのすべての.warningクラスはのテキストのをすることになる、と.importantつクラスは、オレンジのテキストのをし、 っているての.importantと.warningクラスはのテキストをっています。

CSSでは、 .warning.importantに2つのクラスのにスペ―スがまれていなかったことにしてくださ

い。つまり、 `class`に`warning`と`important`のクラスをむのみがされます。これらのクラスは、ののでできます。

スペースはCSSのつのクラスのにまれていた、それだけで、のつをします`.warning`つクラスと`.important`クラスを。

IDセレクタ

IDセレクタは、ターゲットIDをつDOMをします。 CSSののIDでをするには、 `#`がされます。

たとえば、のHTML `div`...

```
<div id="exampleID">
    <p>Example</p>
</div>
```

...はCSSの`#exampleID`でのようにできます

```
#exampleID {
    width: 20px;
}
```

HTMLでは、じIDをつのをできません

クラス

クラスは、ツリーのにある、またはのセレクタやコンビネータではできないについてできるキーワードです。このは、の およびクラス、 およびターゲットクラス、 クラスまたは langクラスのにけることができます。としては、リンクがたどられた `:visited` か、マウスが `:hover` のにあるか、チェックボックスがチェックされた `:checked` かなどがまれ`:checked`。

```
selector:pseudo-class {
    property: value;
}
```

クラスのリスト

`:active`	ユーザーがアクティブつまりクリックしているにされます。
`:any`	するセレクタのセットをすることができます。 まれているアイテムはします。これは、セレクタをりすわりのです。
`:target`	アクティブな#newsをしますURLでクリックされます。 そのアンカーをむ

`:checked`	チェックされているラジオ、チェックボックス、またはオプションにされます または「オン」にトグルされる。
`:default`	グループののデフォルトであるのユーザーインターフェースをします。 の。
`:disabled`	なにあるすべてのUIにされます。
`:empty`	をたないにされます。
`:enabled`	なにあるすべてのUIにされます。
`:first`	`@page`ルールとみわせてすると、 された。
`:first-child`	ののであるをします。
`:first-of-type`	エレメントがされたエレメントタイプののエレメントであるにされます そののにある。これはのかもしれないし、そうでないかもしれません。
`:focus`	ユーザーのフォーカスがあるにされます。これは、 ユーザのキーボード、マウスイベント、またはのをむことができる。
`:focus-within`	セクションの1つのがフォーカスされている、セクションをするためにできます。これは、focusクラスがマッチする、またはがフォーカスをつにします。
`:full-screen`	フルスクリーンモードでされるにされます。スタックをします。 トップレベルのだけではなくの
`:hover`	ユーザーのポインティングデバイスによってホバリングされているにされますが、 されません。
`:indeterminate`	チェックされていないラジオまたはチェックボックスのUIをします。 チェックされていませんが、です。これは、 のまたはDOM。
`:in-range`	`:in-range` CSSクラスは、が こののされたののvalue これにより、ページはされているをフィードバックすることができます をするとのになります。
`:invalid`	のようにがな`<input>`にされます。 `type=`でされた`type=`。

`:lang`	ラップする`<body>`のににされます された`lang=`。クラスがであるためには、 な2または3のコードがまれています。
`:last-child`	ののであるをします。
`:last-of-type`	エレメントがされたエレメントタイプのであるにされます その。これはのかもしれないし、そうでないかもしれません。
`:left`	`@page`ルールとみわせてすると、これはすべての されたのページ。
`:link`	ユーザーがしていないリンクにされます。
`:not()`	されたとしないすべてのにされます `:not(p)`または`:not(.class-name)`などです。 で、1つのセレクタのみをむことができます。ただし、セレクタで`:not`の ものをすることができます。
`:nth-child`	がそののnのであるにされますn 、えば、 `n+3` またはキーワード `odd`または`even`。
`:nth-of-type`	がそののnのであるにされます。 じタイプ nは、 えば、 `n+3` またはキーワード`odd`または`even`。
`:only-child`	`:only-child` CSSのクラスはのをします これはののです。これは `:first-child:last-child`または`:nth-child(1):nth-last-child(1)`、 はい。
`:optional`	`:optional` CSSクラスはのをします ながされていません。これにより、 オプションのフィールドをにし、それにじてスタイルをするフォーム。
`:out-of-range`	`:out-of-range` CSSクラスは、に こののされたのの。 これは、ページをしてされている がのです。がであってもいません およびよりもさいかまたはきいである。
`:placeholder-shown`	。プレースホルダテキストをしているフォームにされます。
`:read-only`	ユーザーができないにされます。

:read-write	<input>など、ユーザーがなにされます。
:right	@pageルールとみわせてすると、 された。
:root	ドキュメントをすツリーのルートとします。
:scope	CSSクラスはであるにマッチします セレクタがするポイント。
:target	アクティブな#newsをしますURLでクリックされます。 そのアンカーをむ
:visited	ユーザーがしたリンクにされます。

:visited pseudoclassは、セキュリティホールであるため、くののブラウザではほとんどのスタイリングにはできません。までにこのリンクをしてください。

セレクタ

セレクタ	
*	ユニバーサルセレクタ
div	タグセレクタすべての
.blue	クラスセレクタクラスblueすべての
.blue.red	blue と redクラスをつすべてのCompoundセレクタの
#headline	IDセレクタ "id"がheadlineされている
:pseudo-class	クラスをつすべての
::pseudo-element	とする
:lang(en)	するlangえば、
div > p	セレクタ

IDのは、Webページでであるがあります。じツリーでIDのをすることは、 HTMLにしています 。

セレクタのなリストは、 CSSセレクタレベル3にあります。

をスタイルする

HTML

```
<input type="range"></input>
```

CSS

	セレクタ
	`input[type=range]::-webkit-slider-thumb, input[type=range]::-moz-range-thumb, input[type=range]::-ms-thumb`
トラック	`input[type=range]::-webkit-slider-runnable-track, input[type=range]::-moz-range-track, input[type=range]::-ms-track`
OnFocus	`input[type=range]:focus`
トラックの	`input[type=range]::-moz-range-progress, input[type=range]::-ms-fill-lower` WebKitブラウザでは - JSが

チェックボックスきグローバルブールチェックし、コンビネータ

セレクタをすると、JavaScriptをせずにグローバルにアクセスなブールをにできます。

booleanをチェックボックスとして

ドキュメントのに、の`id`と`hidden`セットをして、なブールをします。

```
<input type="checkbox" id="sidebarShown" hidden />
<input type="checkbox" id="darkThemeUsed" hidden />

<!-- here begins actual content, for example: -->
<div id="container">
    <div id="sidebar">
        <!-- Menu, Search, ... -->
    </div>

    <!-- Some more content ... -->
</div>

<div id="footer">
    <!-- ... -->
</div>
```

ブールをする

ブールをりえるには、 `for`をした`label`を`for`ます。

```
<label for="sidebarShown">Show/Hide the sidebar!</label>
```

CSSでブールにアクセスする

のセレクタ `.color-red` は、デフォルトのプロパティをします。 `true` / `false`セレクタにうことでオーバーライドできます

```
/* true: */
<checkbox>:checked ~ [sibling of checkbox & parent of target] <target>

/* false: */
<checkbox>:not(:checked) ~ [sibling of checkbox & parent of target] <target>
```

`<checkbox>` 、 `[sibling ...]` 、および`<target>`はなセレクタにきえてください。 `[sibling ...]`あなたはけがい、に、のセレクタ*ターゲットがすでにチェックボックスのであるまたはも。

のHTMLのはのとおりです。

```
#sidebarShown:checked ~ #container #sidebar {
    margin-left: 300px;
}

#darkThemeUsed:checked ~ #container,
#darkThemeUsed:checked ~ #footer {
    background: #333;
}
```

これらのグローバルブールのについては、 このフィドルをしてください。

CSS3セレクタの

```
<style>
input:in-range {
    border: 1px solid blue;
}
</style>



<input type="number" min="10" max="20" value="15">
<p>The border for this value will be blue</p>
```

`:in-range` CSSクラスは、がそののされたののvalueをつにします。これにより、をしてされているがのにあるというフィードバックがページにされます。 [1]

クラス

> "nth-childan + bCSSクラスは、ドキュメントツリーのに+ b-1のをつをn"ののまたは0のにマッチさせます。 - MDNnth-child

セレクタ	1	2	3	4	5	6	7	8	9	10
`:first-child`	✔									
`:nth-child(3)`			✔							
`:nth-child(n+3)`			✔	✔	✔	✔	✔	✔	✔	✔
`:nth-child(3n)`			✔			✔			✔	
`:nth-child(3n+1)`	✔			✔			✔			✔
`:nth-child(-n+3)`	✔	✔	✔							
`:nth-child(odd)`	✔		✔		✔		✔		✔	
`:nth-child(even)`		✔		✔		✔		✔		✔
`:last-child`										✔
`:nth-last-child(3)`								✔		

IDセレクタのいなしにIDをしてエレメントをする

このトリックは、IDセレクター—のいをけるために、セレクター—のとしてIDをしてエレメントをするのにちます。

HTML

```
<div id="element">...</div>
```

CSS

```
#element { ... } /* High specificity will override many selectors */

[id="element"] { ... } /* Low specificity, can be overridden easily */
```

A.クラスのではなくBCSSのクラスのフォ—カス

A.はのとおりです。

のセレクタは、すべてのする`<input>`されていないとクラスがありませんHTMLドキュメントの`.example`

HTML

```
<form>
    Phone: <input type="tel" class="example">
```

```
    E-mail: <input type="email" disabled="disabled">
    Password: <input type="password">
</form>
```

CSS

```
input:not([disabled]):not(.example){
   background-color: #ccc;
}
```

`:not()`クラスはセレクタレベル4でカンマりのセレクタもサポートします

CSS

```
input:not([disabled], .example){
   background-color: #ccc;
}
```

JSBinのライブデモ

ここでのをしてください。

B.フォーカスCSSのクラス

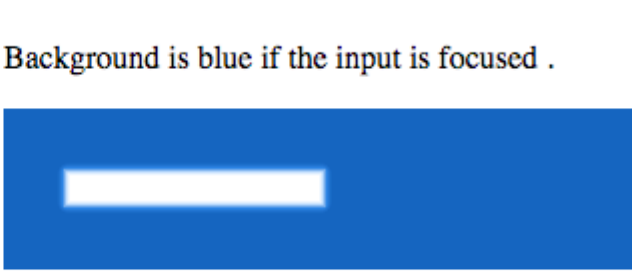
HTML

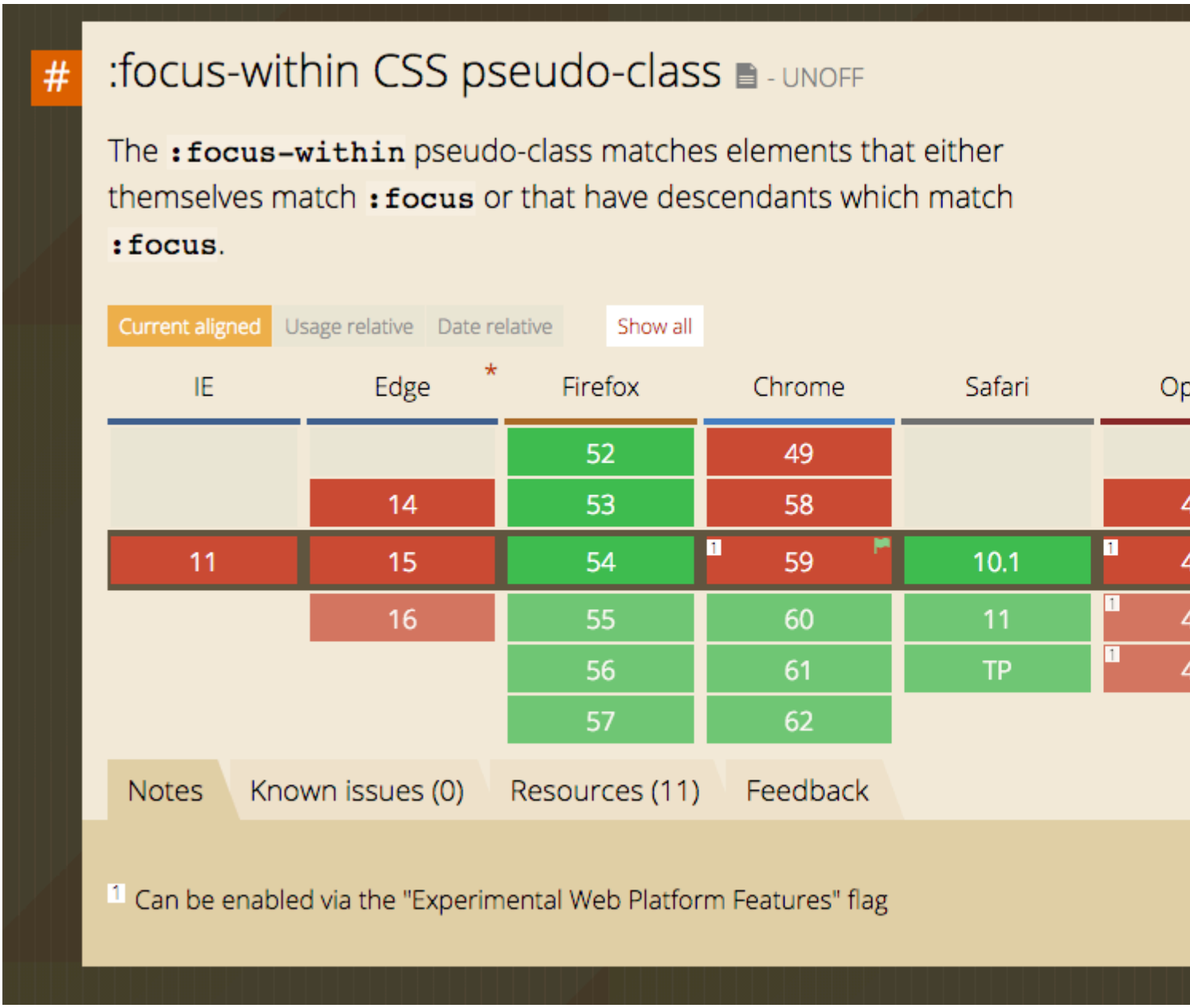
```
  <h3>Background is blue if the input is focused .</p>
  <div>
    <input type="text">
  </div>
```

CSS

```
div {
  height: 80px;
}
input{
  margin:30px;
}
div:focus-within {
  background-color: #1565C0;
}
```

```
div {
  height: 80px;
}
input{
  margin:30px;
}
div:focus-within {
  background-color: #1565C0;
}
```





Theonly-child クラスセレクタの

`:only-child` CSSクラスは、そののののであるをします。

HTML

```
<div>
  <p>This paragraph is the only child of the div, it will have the color blue</p>
</div>

<div>
  <p>This paragraph is one of the two children of the div</p>
  <p>This paragraph is one of the two children of its parent</p>
</div>
```

CSS

```
p:only-child {
```

```
  color: blue;
}
```

のでは、からのである`<p>`をしています。このは`<div>`です。

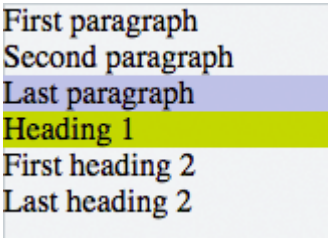
JSBinのライブデモ

ののセレクタ

`:last-of-type`は、そのののののののであるをします。のでは、cssはのとのし`h1`ます。

```
p:last-of-type {
  background: #C5CAE9;
}
h1:last-of-type {
  background: #CDDC39;
}
```

```
<div class="container">
    <p>First paragraph</p>
    <p>Second paragraph</p>
    <p>Last paragraph</p>
    <h1>Heading 1</h1>
    <h2>First heading 2</h2>
    <h2>Last heading 2</h2>
</div>
```



jsFiddle

オンラインでセレクタをむ https://riptutorial.com/ja/css/topic/611/セレクタ

24: センタリング

Examples

CSSトランスフォームの

CSSのはのサイズにづいているため、のさやがわからないは、なコンテナのとからに50をし、それを50またはにできますそれをおよびにセンタリングする。

このでは、がのピクセルでレンダリングされてぼやけてえることにしてください。については、このをしてください。

HTML

```
<div class="container">
  <div class="element"></div>
</div>
```

CSS

```
.container {
  position: relative;
}

.element {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
```

JSFiddleでのをる

CROSS BROWSER COMPATIBILITY

transformプロパティには、いブラウザでサポートされるがです。プレフィックスは、Chrome <= 35、Safari <= 8、Opera <= 22、Androidブラウザ<= 4.4.4、およびIE9でです。 CSSはIE8およびそれのバージョンではサポートされていません。

これはののなです

```
-webkit-transform: translate(-50%, -50%); /* Chrome, Safari, Opera, Android */
    -ms-transform: translate(-50%, -50%); /* IE 9 */
        transform: translate(-50%, -50%);
```

は、「 canluse 」をしてください。

しくは

- は、のな `position: relative` 、 `absolute` 、または`fixed` にってされています。このフィドルとこのドキュメントのトピックでしくべてください。
- のみのセンタリングでは、 `left: 50%` 、 `transform: translateX(-50%)`ます。じことがのみのセンタリングのが`top: 50%` 、 `transform: translateY(-50%)` です。
- このようなセンタリングでな/さのをすると、センタリングされたがりがってされることがあります。これはにテキストをむでし、 `margin-right: -50%; margin-bottom: -50%;` 。はこのフィドルをてください。

Flexboxの

HTML

```
<div class="container">
  <img src="http://lorempixel.com/400/200" />
</div>
```

CSS

```
html, body, .container {
  height: 100%;
}
.container {
  display: flex;
  justify-content: center; /* horizontal center */
}
img {
  align-self: center; /* vertical center */
}
```

をる

HTML

```
<img src="http://lorempixel.com/400/200" />
```

CSS

```
html, body {
  height: 100%;
}
body {
  display: flex;
  justify-content: center; /* horizontal center */
  align-items: center;     /* vertical center */
}
```

をる

flexboxのとスタイルのについては、 FlexboxのドキュメントのDynamic Vertical and Horizontal Centeringをしてください。

ブラウザのサポート

Flexboxは、 10よりのバージョンのIEをくすべてのなブラウザでサポートされています。

Safari 8やIE10などののブラウザのバージョンには、 ベンダープレフィックスがです。

プレフィックスをするためのなとして、サードパーティのツールであるAutoprefixerがあります。

いブラウザIE 8や9などでは、 Polyfillがです 。

フレックスボックスブラウザのサポートのについては、 このをしてください。

の

いブラウザでする*IE> = 8*

ゼロのとにマージン、 `left`び`right`は`top`と`bottom`オフセットは、ののののをにします。

をる

HTML

```
<div class="parent">
  <img class="center" src="http://lorempixel.com/400/200/" />
</div>
```

CSS

```
.parent {
  position: relative;
  height: 500px;
}

.center {
  position: absolute;
  margin: auto;
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
}
```

のようななとさをたないは、されたをとします。

そののリソース CSSのセンタリング

ゴーストMichałCzernowのハック

このは、コンテナのがなでもします。

100のさになるように、コンテナの "ゴースト"をし、そのに`vertical-align: middle`をし、にするのに`vertical-align: middle`。

CSS

```
/* This parent can be any width and height */
.block {
  text-align: center;

  /* May want to do this if there is risk the container may be narrower than the element
inside */
  white-space: nowrap;
}

/* The ghost element */
.block:before {
  content: '';
  display: inline-block;
  height: 100%;
  vertical-align: middle;

  /* There is a gap between ghost element and .centered,
  caused by space character rendered. Could be eliminated by
  nudging .centered (nudge distance depends on font family),
  or by zeroing font-size in .parent and resetting it back
  (probably to 1rem) in .centered. */
  margin-right: -0.25em;
}

/* The element to be centered, can also be of any width and height */
.centered {
  display: inline-block;
  vertical-align: middle;
  width: 300px;
  white-space: normal; /* Resetting inherited nowrap behavior */
}
```

HTML

```
<div class="block">
  <div class="centered"></div>
</div>
```

テキストアライン メントをする

もでもなタイプのセンタリングは、のテキストのセンタリングです。 CSSはこののために`text-align: center`ルールをってい`text-align: center`

HTML

```
<p>Lorem ipsum</p>
```

CSS

```
p {
    text-align: center;
}
```

これは、ブロックをセンタリングするためにはしません。 `text-align`は、ブロックのテキストのようなインラインコンテンツのアライメントのみをします。

タイポグラフィのセクションで`text-align`をしてください。

のアイテムとのけ

いのさについてコンテンツをどのようにするかをていきます。

IE8 +、のすべてののブラウザ。

HTML

```
<div class="content">
  <div class="position-container">
    <div class="thumb">
      <img src="http://lorempixel.com/400/200/">
    </div>
    <div class="details">
      <p class="banner-title">text 1</p>
      <p class="banner-text">content content content content content content content content
content content content content content content</p>
      <button class="btn">button</button>
    </div>
  </div>
</div>
```

CSS

```
.content * {
  box-sizing: border-box;
}
.content .position-container {
  display: table;
}
.content .details {
  display: table-cell;
  vertical-align: middle;
  width: 33.333333%;
  padding: 30px;
  font-size: 17px;
  text-align: center;
}
.content .thumb {
  width: 100%;
}
```

```
.content .thumb img {
  width: 100%;
}
```

JSFiddleへのリンク

なポイントは、3つの.thumb 、 .details 、 .position-containerコンテナです。

- .position-containerはdisplay: tableって.position-containerなければなりません。
- .detailsはののwidth: ....とdisplay: table-cell 、 vertical-align: middleっていなければなりません。
- .thumbはりのスペースをすべてるようにするにはwidth: 100%するがあり、 .detailsのをけます。
- .thumbにあるがあるのwidth: 100%にするがありますが、がしいはありません。

3のコードでかをにさせる

IE11 +でサポート

をる

これらの3をして、にすべてをにさせます。コードをするdiv / imageにさのあるがあることをしてください。

CSS

```
div.vertical {
  position: relative;
  top: 50%;
  transform: translateY(-50%);
}
```

HTML

```
<div class="vertical">Vertical aligned text!</div>
```

divのをにする

HTML

```
<div class="wrap">
    <img src="http://lorempixel.com/400/200/" />
</div>
```

CSS

```
.wrap {
    height: 50px;/* max image height */
    width: 100px;
    border: 1px solid blue;
    text-align: center;
}
.wrap:before {
  content:"";
  display: inline-block;
  height: 100%;
  vertical-align: middle;
  width: 1px;
}

img {
    vertical-align: middle;
}
```

テーブルレイアウトをしたとのセンタリング

`table`プロパティをしてをににくことができます。

HTML

```
<div class="wrapper">
    <div class="parent">
        <div class="child"></div>
    </div>
</div>
```

CSS

```
.wrapper {
    display: table;
    vertical-align: center;
    width: 200px;
    height: 200px;
    background-color: #9e9e9e;
}
.parent {
    display: table-cell;
    vertical-align: middle;
    text-align: center;
}
.child {
    display: inline-block;
    vertical-align: middle;
    text-align: center;
    width: 100px;
    height: 100px;
    background-color: teal;
}
```

calcをすると、

calcは、ピクセル、パーセンテージなどのさまざまなをしてがめるサイズ/をできるにCSS3のしいのです。 - このをするときはいつでも、に2つの`calc(100% - 80px)`のスペースを`calc(100% - 80px)`ます。

CSS

```
.center {
    position: absolute;
    height: 50px;
    width: 50px;
    background: red;
    top: calc(50% - 50px / 2); /* height divided by 2*/
    left: calc(50% - 50px / 2); /* width divided by 2*/
}
```

HTML

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

さをにする

にCSSをしても、ましいがられません。

- `vertical-align:middle` ブロックレベルに `vertical-align:middle` はされません
- `margin-top:auto`および`margin-bottom:auto` されるはゼロとしてされます
- `margin-top:-50%` パーセンテージベースの `margin-top:-50%` マージンは、ブロックのにしてされます

もいブラウザサポートのために、ヘルパーをした

HTML

```
<div class="vcenter--container">
  <div class="vcenter--helper">
    <div class="vcenter--content">
      <!--stuff-->
    </div>
  </div>
</div>
```

CSS

```
.vcenter--container {
  display: table;
  height: 100%;
  position: absolute;
  overflow: hidden;
  width: 100%;
}
.vcenter--helper {
  display: table-cell;
  vertical-align: middle;
```

```
}
.vcenter--content {
  margin: 0 auto;
  width: 200px;
}
```

のからjsfiddle 。このアプローチ

- なさのでする
- コンテンツのれをする
- のブラウザでサポートされています

のさをする

また、 `line-height`をして、コンテナの1のテキストをにセンタリングすることもできます。

CSS

```
div {
    height: 200px;
    line-height: 200px;
}
```

これはかなりいですが、 `<input />`のではです。 `line-height`プロパティは、えされるテキストが1にまたがるにのみします。テキストがのにりされた、のはえされません。

さやをにすることなくにセンタリング

のでは、コンテンツをHTMLにし、そのさやをにすることなく、とのにコンテンツをすることができます。

- `display: table;`があり`display: table;`
- `display: table-cell;`があり`display: table-cell;`
- `vertical-align: middle;`なければなりません`vertical-align: middle;`
- `text-align: center;`が`text-align: center;`

コンテンツボックス

- `display: inline-block;``display: inline-block;`
- のテキストのをするがあります。 `text-align: left;`または`text-align: right;`あなたがテキストをにえたいのでないり

デモ

HTML

```
<div class="outer-container">
   <div class="inner-container">
     <div class="centered-content">
        You can put anything here!
     </div>
   </div>
</div>
```

CSS

```
body {
    margin : 0;
}

.outer-container {
    position : absolute;
    display: table;
    width: 100%; /* This could be ANY width */
    height: 100%; /* This could be ANY height */
    background: #ccc;
}

.inner-container {
    display: table-cell;
    vertical-align: middle;
    text-align: center;
}

.centered-content {
    display: inline-block;
    text-align: left;
    background: #fff;
    padding: 20px;
    border: 1px solid #000;
}
```

このフィドルもてください

サイズのセンタリング

コンテンツのサイズがされているは、コンテンツのとさのをする`margin`でを50まででできます。

HTML

```
<div class="center">
    Center vertically and horizontally
</div>
```

CSS

```
.center {
    position: absolute;
    background: #ccc;

    left: 50%;
```

```
    width: 150px;
    margin-left: -75px;  /* width * -0.5 */

    top: 50%;
    height: 200px;
    margin-top: -100px;  /* height * -0.5 */
}
```

のみのセンタリング

コンテンツのさがからないでも、をにすることができます。

HTML

```
<div class="center">
    Center only horizontally
</div>
```

CSS

```
.center {
    position: absolute;
    background: #ccc;

    left: 50%;
    width: 150px;
    margin-left: -75px;  /* width * -0.5 */
}
```

さによるセンタリング

のさがわかっているは、をににできます。

HTML

```
<div class="center">
    Center only vertically
</div>
```

CSS

```
.center {
    position: absolute;
    background: #ccc;

    top: 50%;
    height: 200px;
    margin-top: -100px;  /* width * -0.5 */
}
```

をする**0;**

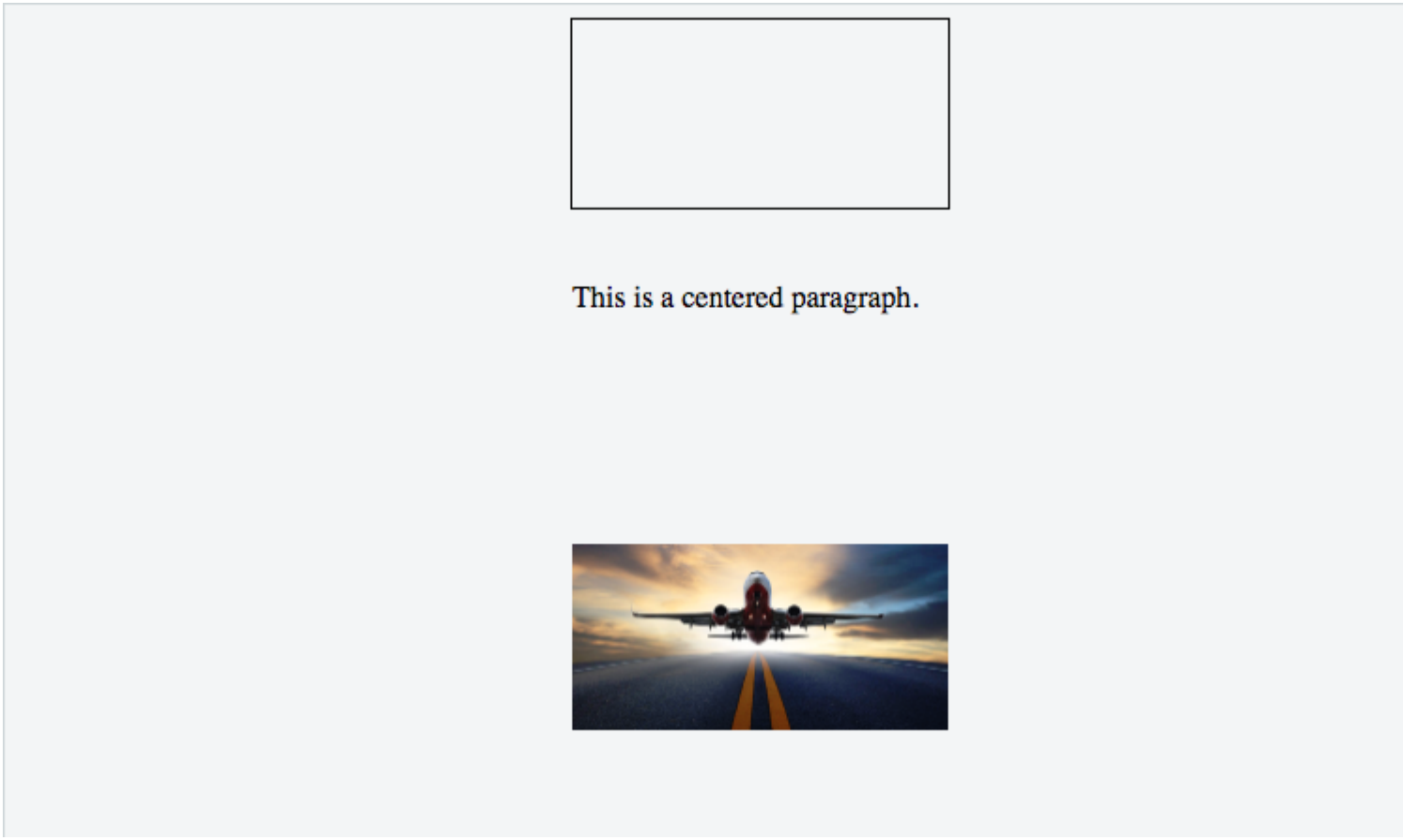
`margin: 0 auto;`をってオブジェクトをにすることができます`margin: 0 auto;`それらがブロックであり、されたをつ。

HTML

```
<div class="containerDiv">
    <div id="centeredDiv"></div>
</div>

<div class="containerDiv">
    <p id="centeredParagraph">This is a centered paragraph.</p>
</div>

<div class="containerDiv">
    <img id="centeredImage" src="https://i.kinja-img.com/gawker-media/image/upload/s--
c7Q9b4Eh--/c_scale,fl_progressive,q_80,w_800/qqyvc3bkpyl3mfhr8all.jpg" />
</div>
```

CSS

```
.containerDiv {
    width: 100%;
    height: 100px;
    padding-bottom: 40px;
}

#centeredDiv {
    margin: 0 auto;
    width: 200px;
    height: 100px;
    border: 1px solid #000;
}

#centeredParagraph {
    width: 200px;
    margin: 0 auto;
}

#centeredImage {
    display: block;
    width: 200px;
    margin: 0 auto;
}
```



JSFiddleの きのオブジェクトのセンタリング0 auto;

オンラインでセンタリングをむ https://riptutorial.com/ja/css/topic/299/センタリング

25: タイポグラフィ

- font *-style* [*font-style*] *[font-variant] [font-weight]* フォントサイズ *[/ line-height] font-family* ;
- font-style *font-style*
- font-variant *font-variant*
- font-weight *font-weight* ;
- font-size *font-size* ;
- line-height のさ 。
- font-family *font-family* ;
- カラー カラー ;
- なし||| ;
- font-stretch *font-stretch* ;
- text-align *text-align* ;
- テキストインデント さ|| ;
- テキストオーバーフロー クリップ|||| ;
- テキスト なし||||| ;
- text-shadow *h-shadow v-shadow blur-radius*カラー| *none* | *initial* | ;
- font-size-adjust *number* | *none* | *initial* |;
- フォントストレッチ |||||||||;
- ハイフン なし|マニュアル| ;
- tab-size *number* | *length* | *initial* |します。
- *normal* | *length* | *initial* | ;
- *normal* | *length* | *initial* | ;

パラメーター

パラメータ	
フォントスタイル	`italics`や`oblique`
フォントバリアント	`normal`または`small-caps`
フォントウェイト	`normal` 、 `bold`または100から900までのです。
フォントサイズ	`%` 、 `px` 、 `em` 、またはそののなCSSでされたフォントサイズ
のさ	`%` 、 `px` 、 `em` 、またはのなCSSでえられたのさ

パラメータ	
フォントファミリー	これは、のをするためのものです。
	`red` 、 `#00FF00` 、 `hsl(240, 100%, 50%)`などのなCSSカラー
フォントストレッチ	フォントからののまたはをするかどうか。なは、 `normal` 、 `ultra-condensed` `extra-condensed` 、 `condensed` 、 `semi-condensed` 、 `semi-expanded` 、 `expanded` 、 `ultra-expanded` `extra-expanded`または`ultra-expanded`
テキスト	`start` 、 `end` 、 `left` 、 `right` 、 `center` 、 `justify` 、 `match-parent`
テキスト	`none` 、 `underline` 、 `overline` 、 `line-through` 、 `initial` 、 `inherit` ;

- `text-shadow`プロパティは、Internet Explorerのバージョンが10ではサポートされていません。

Examples

フォントサイズ

HTML

```
<div id="element-one">Hello I am some text.</div>
<div id="element-two">Hello I am some smaller text.</div>
```

CSS

```
#element-one {
    font-size: 30px;
}

#element-two {
    font-size: 10px;
}
```

`#element-one`のテキストのサイズは`30px`ですが、 `#element-two`テキストのサイズは`10px`になります。

フォントショートカット

はのとおりです。

```
element {
```

```
    font: [font-style] [font-variant] [font-weight] [font-size/line-height] [font-family];
}
```

すべてのフォントスタイルを1つので`font`にすることができます。に`font`プロパティをして、しいでをしてください。

たとえば、すべての`p`を20 pxのフォントサイズでにし、Arialをフォントファミリとしてするは、、のようにコ―ドします。

```
p {
    font-weight: bold;
    font-size: 20px;
    font-family: Arial, sans-serif;
}
```

しかし、フォントのでは、のようにできます

```
p {
    font: bold 20px Arial, sans-serif;
}
```

`font-style`、 `font-variant`、 `font-weight`および`line-height`はオプションであるため、このでは3つをしています。ショ―トカットをすると、えられていないのがリセットされることにすることがです。のなは、フォントショ―トカットがするためにな2つのが`font-size`と`font-family`です。 がまれていない、ショ―トカットはされます。

プロパティの

- `font-style: normal;`
- `font-variant: normal;`
- `font-weight: normal;`
- `font-stretch: normal;`
- `font-size: medium;`
- `line-height: normal;`
- `font-family` - ユ―ザエ―ジェントにする

フォントスタック

```
font-family: 'Segoe UI', Tahoma, sans-serif;
```

ブラウザは、のプロパティのとなるのにフォント "Segoe UI"をしようとします。このフォントができない、またはフォントになのグリフがまれていない、ブラウザはTahomaにり、にじてユ―ザ―のコンピュ―タのサンセリフのフォントにります。 "Segoe UI"のようなのをつフォントは、またはをするがあることにしてください。

```
font-family: Consolas, 'Courier New', monospace;
```

ブラウザは、のプロパティのとなるのにフォントフェイス "Consolas"をしようとします。このフ

ォントができない、またはフォントになのグリフがまれていない、ブラウザは「Courier New」にり、にじてユーザーのコンピュータのモノスペースフォントにります。

```
h2 {
  /* adds a 1px space horizontally between each letter;
     also known as tracking */
  letter-spacing: 1px;
}
```

---

letter-spacingプロパティは、テキストののスペースをするためにされます。

はのもサポートします

```
p {
  letter-spacing: -1px;
}
```

リソース https : //developer.mozilla.org/en-US/docs/Web/CSS/letter-spacing

テキスト

`text-transform`プロパティをすると、テキストのをできます。なは、 `uppercase` 、 `capitalize` 、 `lowercase` 、 `initial` 、 `inherit` 、 および`none`

CSS

```
.example1 {
    text-transform: uppercase;
}
.example2 {
    text-transform: capitalize;
}
.example3 {
    text-transform: lowercase;
}
```

HTML

```
<p class="example1">
    all letters in uppercase  <!-- "ALL LETTERS IN UPPERCASE" -->
</p>
<p class="example2">
    all letters in capitalize <!-- "All Letters In Capitalize (Sentence Case)" -->
</p>
<p class="example3">
    all letters in lowercase  <!-- "all letters in lowercase" -->
</p>
```

テキスト インデント

```
p {
    text-indent: 50px;
}
```

`text-indent`プロパティは、のテキストコンテンツののののにするのスペ―スのをします。

リソ―ス

- のテキストののだけイ ンデントしますか
- https://www.w3.org/TR/CSS21/text.html#propdef-text-indent
- https://developer.mozilla.org/en-US/docs/Web/CSS/text-indent

テキスト

`text-decoration`プロパティは、テキストからをまたはするためにされます。

```
h1 { text-decoration: none; }
h2 { text-decoration: overline; }
h3 { text-decoration: line-through; }
h4 { text-decoration: underline; }
```

text-decoration-styleおよびtext-decoration-colorとみわせて、プロパティとしてtext-decorationをすることができます。

```
.title { text-decoration: underline dotted blue; }
```

これは、

```
.title {
  text-decoration-style: dotted;
  text-decoration-line: underline;
  text-decoration-color: blue;
}
```

のプロパティはFirefoxでのみサポ―トされていることにしてください

- テキスト
- テキスト
- テキストスタイル
- テキスト - スキップ

テキスト オ―バ―フロ―

`text-overflow`プロパティは、オ―バ―フロ―したコンテンツをユ―ザ―にするをいます。このでは、 `ellipsis`はクリップされたテキストをします。

```
.text {
  overflow: hidden;
```

```
  text-overflow: ellipsis;
}
```

ながら、 `text-overflow: ellipsis`は、1のテキストでしかしません。CSSののにをけるはありませんが、フレックスボックスのWebkitのみのなでできます。

```
.giveMeEllipsis {
   overflow: hidden;
   text-overflow: ellipsis;
   display: -webkit-box;
   -webkit-box-orient: vertical;
   -webkit-line-clamp: N; /* number of lines to show */
   line-height: X;        /* fallback */
   max-height: X*N;       /* fallback */
}
```

ChromeまたはSafariでく

http://jsfiddle.net/csYjC/1131/

リソース

https://www.w3.org/TR/2012/WD-css3-ui-20120117/#text-overflow0

ワード

word-spacingプロパティは、タグとののスペーシングのをします。

な

- またはのさ `em` `px` `vh` `cm`などをまたはパーセンテージ `%`を
- キーワード`normal`は、フォントのデフォルトのをします
- キーワード`inherit`はから`inherit`けります

CSS

```
.normal    { word-spacing: normal; }
.narrow    { word-spacing: -3px; }
.extensive { word-spacing: 10px; }
```

HTML

```
<p>
  <span class="normal">This is an example, showing the effect of "word-spacing".</span><br>
  <span class="narrow">This is an example, showing the effect of "word-spacing".</span><br>
  <span class="extensive">This is an example, showing the effect of "word-spacing".</span><br>
</p>
```

オンラインデモ

でしてみてください

- - MDN
- ワードスペース - w3.org

テキストの

```
div {
    direction: ltr; /* Default, text read read from left-to-right */
}
.ex {
    direction: rtl; /* text read from right-to-left */
}
.horizontal-tb {
    writing-mode: horizontal-tb; /* Default, text read from left-to-right and top-to-bottom.
*/
}
.vertical-rtl {
    writing-mode: vertical-rl; /* text read from right-to-left and top-to-bottom */
}
.vertical-ltr {
    writing-mode: vertical-rl; /* text read from left-to-right and top to bottom */
}
```

`direction`プロパティは、ののテキストをするためにされます。

```
direction: ltr | rtl | initial | inherit;
```

`writing-mode`プロパティは、テキストのをし、にじてから、またはからにむことができます。

```
direction: horizontal-tb | vertical-rl | vertical-lr;
```

フォントバリアント

フォントのデフォルト。

さな

にすべてのをしますが、はよりもサイズがさいのテキストからになります。

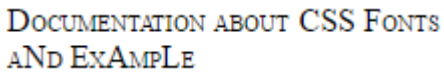
CSS

```
.smallcaps{
   font-variant: small-caps;
}
```

HTML

```
<p class="smallcaps">
    Documentation about CSS Fonts
    <br>
    aNd ExAmpLe
</p>
```

OUTPUT



font-variantプロパティは、font-variant-caps、font-variant-numeric、font-variant-alternates、font-variant-ligatures、およびfont-variant-east-asianのです。

`quotes`プロパティは、 `<q>`タグのおよびのをカスタマイズするためにされます。

```
q {
    quotes: "«" "»";
}
```

テキストシャドー

テキストにをするには、 `text-shadow`プロパティをし`text-shadow`。 はのとおりです。

```
text-shadow: horizontal-offset vertical-offset blur color;
```

ぼかしのない

```
h1 {
  text-shadow: 2px 2px #0000FF;
}
```

これにより、しのりにいのがされます

ぼかしのシャドウ

ぼかしをするには、オプションの`blur radius`をします

```
h1 {
  text-shadow: 2px 2px 10px #0000FF;
}
```

の

にのをけるには、それらをカンマでります

```
h1 {
    text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;
}
```

オンラインでタイポグラフィをむ https://riptutorial.com/ja/css/topic/427/タイポグラフィ

26: テーブル

- テーブルレイアウト |;
- |;
- border-spacing<さ> | <length> <length>;
- セル |す;
- キャプション *top* |;

これらのプロパティは、 `<table>`*と`display: table`として`display: table`されるHTMLのにされ`display: table`または`display: inline-table`

* `<table>`は、 `display: table`としてUA /ブラウザによってネイティブにスタイルされてい`display: table`

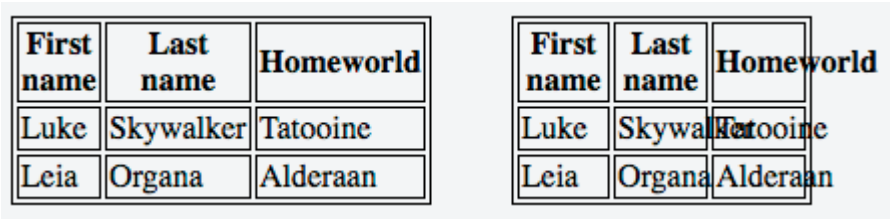
HTMLは、データにしてにです。レイアウトにテーブルをすることはされません。わりに、CSSをしてください。

Examples

テーブルレイアウト

`table-layout`プロパティは、テーブルのレイアウトにされるアルゴリズムをします。

のでは、2つのテーブルのを`width: 150px`



First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

の`table-layout: auto`は`table-layout: auto`をち、の`table-layout: fixed`は`table-layout: fixed`です。はされた150pxのわりに210pxよりもいですが、コンテンツはまります。は、がオーバーフローするかどうかにかかわらず、された150pxをとります。

オート	これがデフォルトです。それは、そのセルのによってされるテーブルのレイアウトをします。
	このは、テーブルにされたwidthプロパティによってされるテーブルレイアウトをします。セルのコンテンツがこのをえると、セルのサイズはされず、わりにコンテンツがオーバーフローします。

`border-collapse`プロパティは、テーブルのをまたはマージするがあるかどうかをします。

`border-collapse`プロパティのがなる2つのテーブルのをにします。

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

のは、っている`border-collapse: separate`の1がっていながら、 `border-collapse: collapse`。

々の	これがデフォルトです。それは、テーブルのをいにさせます。
	このは、のをするのではなく、するようにします。

ボーダー

`border-spacing`プロパティは、セルのをします。 `border-collapse`が`separate`されていないり、これはがありません。

`border-spacing`プロパティとなるをつ2つのテーブルのをにします。

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

のには`border-spacing: 2px` デフォルトがあり、に`border-spacing: 8px`ます。

<length>	これはデフォルトのですが、なはブラウザによってなるがあります。
<length> <length>	このでは、それぞれ々のとをできます。

セル

`empty-cells`プロパティは、のないセルをするかどうかをします。 `border-collapse`が`separate`されていないり、これはがありません。

なるをつ2つのテーブルが`empty-cells`プロパティにされているをにします。

First name	Last name	Homeworld
Luke		Tatooine
Leia	Organa	

First name	Last name	Homeworld
Luke		Tatooine
Leia	Organa	

のは、っている`empty-cells: show`の1がっていながら、 `empty-cells: hide` 。 はのセルをし、はのセルをします。

ショー	これがデフォルトです。それらがであってもセルをします。
す	セルにがない、このはセルをにします。

しくは

- https://www.w3.org/TR/CSS21/tables.html#empty-cells
- https://developer.mozilla.org/en-US/docs/Web/CSS/empty-cells
- http://codepen.io/SitePoint/pen/yfhtq
- https://css-tricks.com/almanac/properties/e/empty-cells/

`caption-side`プロパティは、テーブルの`<caption>`ののをします。このようながしない、これはがありません。

なるをつ2つのテーブルが`caption-side`プロパティにされているをにします。

Star Wars figures

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

Star Wars figures

のには`caption-side: top`あり、のには`caption-side: bottom`ます。

	これがデフォルトです。キャプションをテーブルのにきます。
	このは、キャプションをテーブルのにきます。

オンラインでテーブルをむ https://riptutorial.com/ja/css/topic/1074/テーブル

27: トランジション

- [プロパティ] [] [タイミング] [];

パラメーター

パラメータ	
	すべてのなプロパティをするがあるは、のがなのCSSプロパティを`all`または`all`するがあります。
	がこらなければならないまたは`s`またはミリ`ms`。
タイミング	ののをする。にされるは、`ease`、`ease-in`、`ease-out`、`ease-in-out`、`linear`、`cubic-bezier()` `steps()`さまざまなタイミングのについては、W3Cのをしてください。
	がするまでにしていなければならない。でることができます`s`またはミリ`ms`

いくつかのいブラウザは、ベンダーの`transition`プロパティのみをサポートしています。

- `-webkit` Chrome 25-、Safari 6-、SafariChrome for iOS 6.1-、Android 4.3-ブラウザ、Blackberry Browser 7-、UC Browser 9.9- Android。
- `-moz` Firefox 15-
- `-o` Opera 11.5-、Opera Mobile 12-。

```
-webkit-transition: all 1s;
-moz-transition: all 1s;
-o-transition: all 1s;
transition: all 1s;
```

Examples

トランジションの

CSS

```
div{
    width: 150px;
    height:150px;
    background-color: red;
    transition: background-color 1s;
```

```
}
div:hover{
    background-color: green;
}
```

HTML

```
<div></div>
```

このでは、divがされたときにがされます。のは1きます。

トランジションさ

CSS

```
div {
  height: 100px;
  width: 100px;
  border: 1px solid;
  transition-property: height, width;
  transition-duration: 1s, 500ms;
  transition-timing-function: linear;
  transition-delay: 0s, 1s;
}
div:hover {
  height: 200px;
  width: 200px;
}
```

HTML

```
<div></div>
```

- **transition-property** トランジションエフェクトがされるCSSプロパティをします。この、divはとのでされます。
- **transition-duration** がするまでのをします。のでは、さとのにはそれぞれ1と500ミリがかかります。
- **transition-timing-function** トランジションのカーブをします。 リニアは、トランジションのからまでのスピードがじであることをします。
- **transition-delay** トランジションエフェクトがするまでにするをします。この、さはすぐにをしますが、は1します。

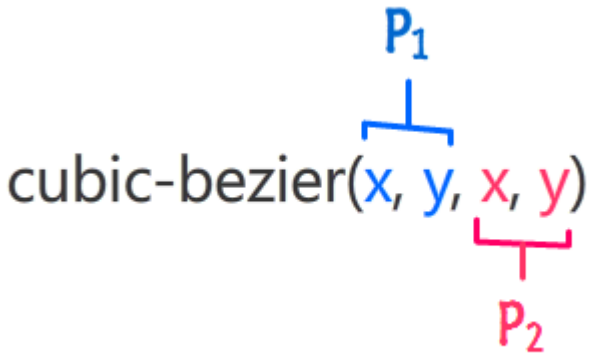
ベジェ

`cubic-bezier`は、カスタムおよびスムーズなによくされるタイミングです。

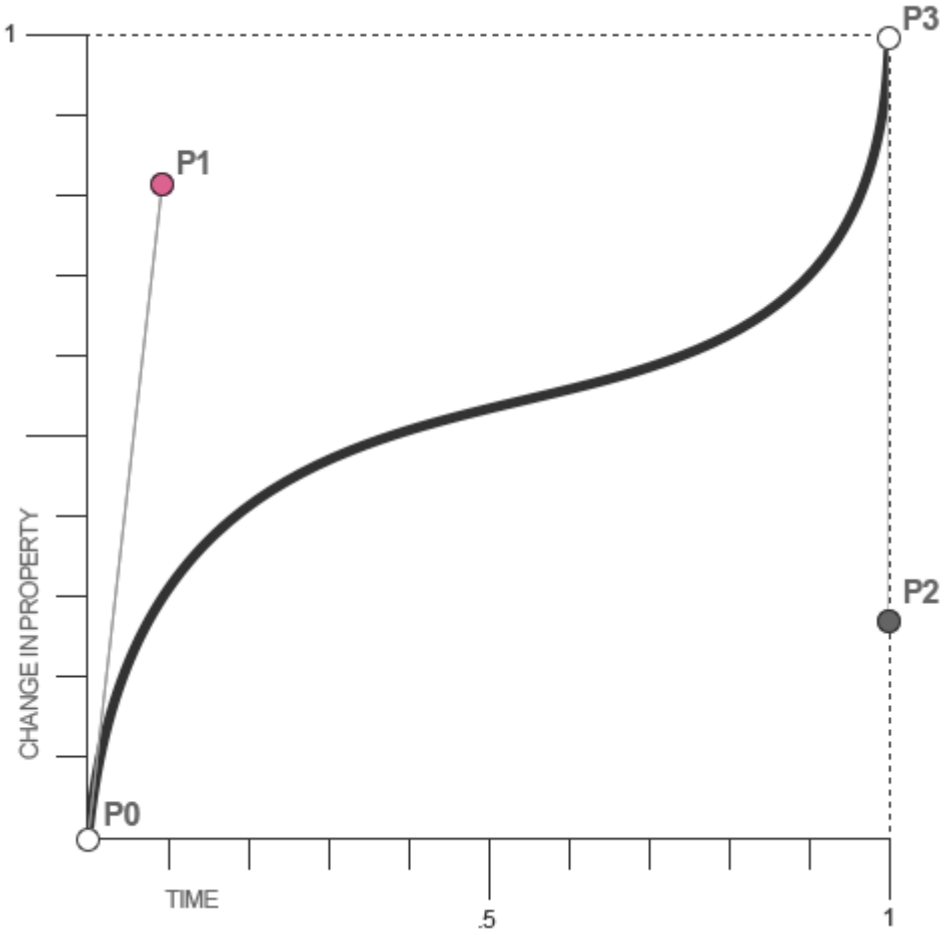
```
transition-timing-function: cubic-bezier(0.1, 0.7, 1.0, 0.1);
```

は4つのパラメータをとります

```
cubic-bezier(P1_x, P1_y, P2_x, P2_y)
```



これらのパラメータは、 ベジエのであるにマップされます。



CSSベジエの、P0とP3はにじにあります。 P0は0,0にあり、P3は1,1にあり、キュービックベジェにされるパラメータは0と1のにしかない。

このにないパラメータをすと、はデフォルトで`linear`になります。

ベジエはCSSでもなであるため、のすべてのタイミングを3ベジエにすることができます。

```
linear cubic-bezier(0,0,1,1)
```

`cubic-bezier(0.42, 0.0, 1.0, 1.0)` `ease-in` `cubic-bezier(0.42, 0.0, 1.0, 1.0)`

`ease-out` `cubic-bezier(0.0, 0.0, 0.58, 1.0)`

`cubic-bezier(0.42, 0.0, 0.58, 1.0)` `ease-in-out` `cubic-bezier(0.42, 0.0, 0.58, 1.0)`

オンラインでトランジションをむ https://riptutorial.com/ja/css/topic/751/トランジション

28: パディング

- パディング *length* | initial || unset;
- padding-top *length* | initial || unset;
- padding-right *length* | initial || unset;
- padding-bottom *length* | initial || unset;
- padding-left *length* | initial || unset;

  paddingプロパティは、のすべてのにパディングスペースをします。パディングは、のとそののののスペースです。 のはされません 。

1 https : //developer.mozilla.org/en/docs/Web/CSS/padding MDN

このもしてください。なぜ 、 CSSはなめみをサポートしていませんかとのえ。

パッディングをするは、 The Box Modelもしてください。ボックスサイズのにじて、エレメントのパディングは、エレメントのにされたさ/にすることも、しないこともできます。

プロパティ

マージン

インラインのめみは、インラインののプロパティのために、のにのみされ、はされません。

Examples

されたのパディング

paddingプロパティは、のすべてのにパディングスペースをします。パディングは、のとそののののスペースです。のはされません。

サイドをにすることができます

- `padding-top`
- `padding-right`
- `padding-bottom`
- `padding-left`

のコードでは、divのに5pxパディングがされます。

```
<style>
.myClass {
    padding-top: 5px;
}
</style>

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

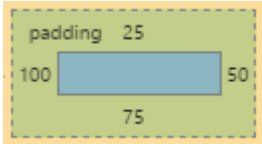
パディングショートカット

paddingプロパティは、のすべてのにパディングスペースをします。パディングは、のとそののののスペースです。のはされません。

パディングをににするには `padding-top` 、 `padding-left` をして、のようにしてくことができます

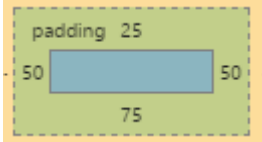
4つの

```
<style>
    .myDiv {
        padding: 25px 50px 75px 100px; /* top right bottom left; */
    }
</style>
<div class="myDiv"></div>
```



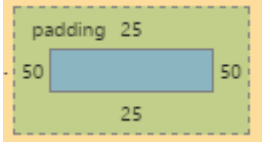
3つの

```
<style>
    .myDiv {
        padding: 25px 50px 75px; /* top left/right bottom */
    }
</style>
<div class="myDiv"></div>
```



2つの

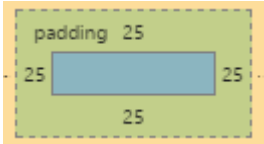
```
<style>
    .myDiv {
        padding: 25px 50px; /* top/bottom left/right */
    }
</style>
<div class="myDiv"></div>
```



1つの

```
<style>
```

```
    .myDiv {
        padding: 25px; /* top/right/bottom/left */
    }
</style>
<div class="myDiv"></div>
```



オンラインでパディングをむ https://riptutorial.com/ja/css/topic/1255/パディング

29: パフォーマンス

Examples

とをしてトリガーレイアウトをする

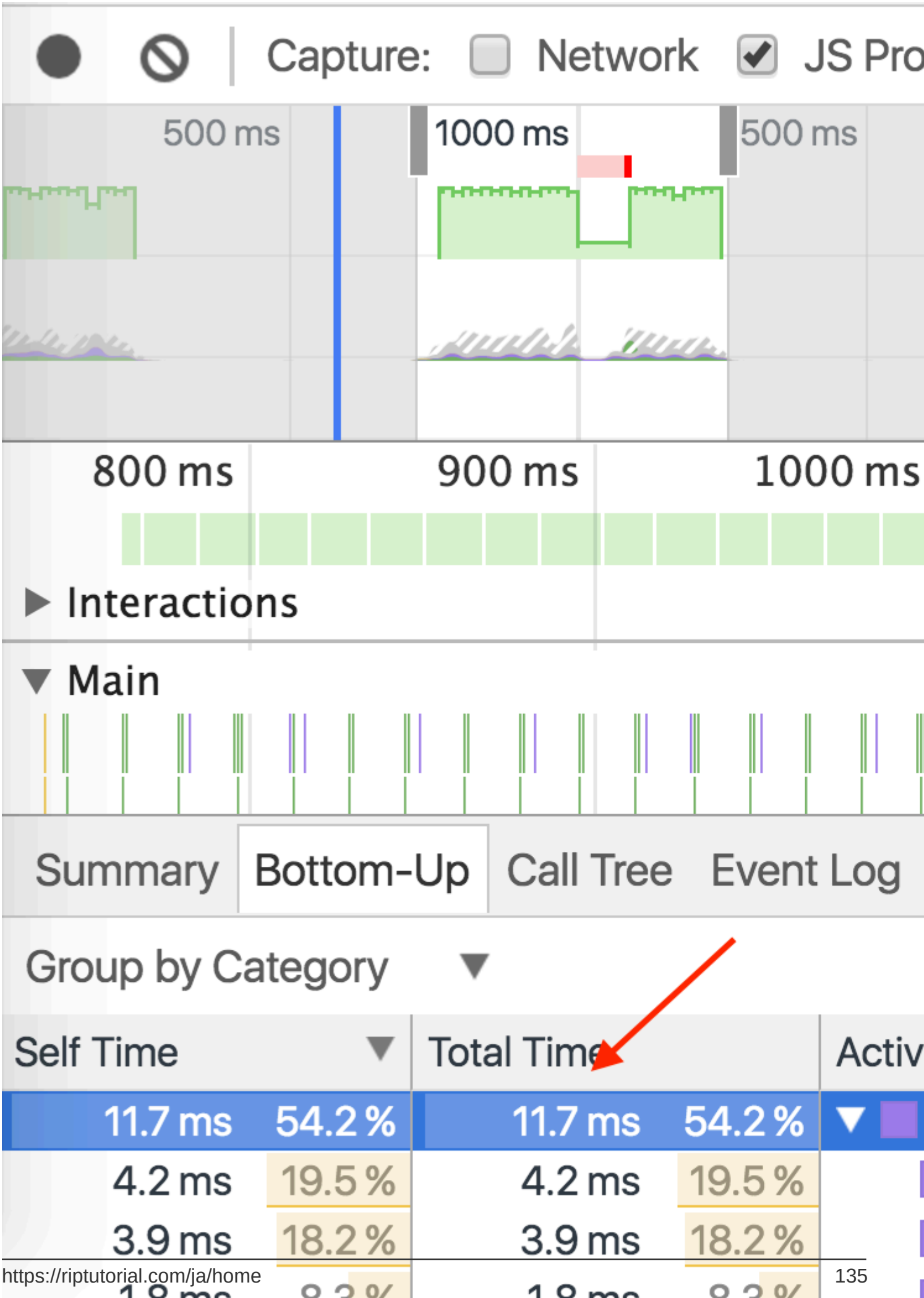
のCSSをするとブラウザがトリガされ、スタイルとレイアウトがにされます。これは60fpsでアニメーションをするがあるにはいことです。

しないでください

`left`と`top`トリガーレイアウトでアニメーションをします。

```
#box {
  left: 0;
  top: 0;
  transition: left 0.5s, top 0.5s;
  position: absolute;
  width: 50px;
  height: 50px;
  background-color: gray;
}

#box.active {
  left: 100px;
  top: 100px;
}
```

デモでは 、のための**9.8ms**をレンダリングするための**11.7ms**をりました


Capture:
Network
JS Pro
500 ms
1000 ms
500 ms
800 ms
900 ms
1000 ms
Interactions
Main
Summary
Bottom-Up
Call Tree
Event Log
Group by Category
Self Time
Total Time
Activ
11.7 ms 54.2 %
11.7 ms 54.2 %
4.2 ms 19.5 %
4.2 ms 19.5 %
3.9 ms 18.2 %
3.9 ms 18.2 %
1.8 ms 8.3 %
1.8 ms 8.3 %

30: フィーチャクエリ

- @supports [] {/ *するCSSルール* /}

パラメーター

パラメータ	
`(property: value)`	ブラウザがCSSルールをできるはtrueをします。ルールのりのかっこがです。
`and`	のとののがであるにのみをします。
`not`	のをします。
`or`	のまたはのがであればをします。
`(...)`	グループ

`@supports`をしたのは、Edge、Chrome、Firefox、Opera、Safari 9 `@supports`でサポートされています。

Examples

な@サポートの

```
@supports (display: flex) {
  /* Flexbox is available, so use it */
  .my-container {
    display: flex;
  }
}
```

にして、 `@supports`は`@media`とにていますが、のサイズときをするわりに、 `@supports`はブラウザがのCSSルールをできるかどうかをします。

`@supports (flex)`ようなものではなく、ルールが`@supports (display: flex)`ことにしてください。

の

にのフィーチャをするには、 `and`をします。

```
@supports (transform: translateZ(1px)) and (transform-style: preserve-3d) and (perspective:
1px) {
  /* Probably do some fancy 3d stuff here */
```

```
}
```

また、 `or`と`not`もあります。

```
@supports (display: flex) or (display: table-cell) {
  /* Will be used if the browser supports flexbox or display: table-cell */
}
@supports not (-webkit-transform: translate(0, 0, 0)) {
  /* Will *not* be used if the browser supports -webkit-transform: translate(...) */
}
```

の`@supports`エクスペリエンスをるには、でをグループしてみてください。

```
@supports ((display: block) and (zoom: 1)) or ((display: flex) and (not (display: table-
cell))) or (transform: translateX(1px)) {
  /* ... */
}
```

これは、ブラウザ

1. `display: block`をサポート`display: block` AND `zoom: 1` 、または
2. `display: flex`をサポートし`display: flex` AND NOT `display: table-cell` 、または
3. `transform: translateX(1px)`をサポートしています。

オンラインでフィーチャクエリをむ https://riptutorial.com/ja/css/topic/5024/フィーチャクエリ

31: フィルタプロパティ

- フィルタなしデフォルト
- filterinitialデフォルトはnone;
- フィルタデフォルトはの;
- フィルタblurpx
- フィルター—るさ|
- フィルター—コントラスト|
- フィルタドロップシャドウシャド—-pxシャドウ-pxシャドウ - ブラー—-pxシャドウ - スプレッドカラー—
- フィルタグレースケール|
- フィルタhue-rotatedeg
- フィルタ|
- フィルタ|
- フィルタ|
- フィルター—セピア|

パラメーター

ぼかしx	をxピクセルだけぼかします。
るさx	1.0または100をえるでをるくします。そのに、がくなります。
コントラストx	1.0または100をえるのでにコントラストをえます。それでは、のがくなります。
ドロップシャドウh、v、x、y、z	イメージにドロップシャドウをえます。 hとvはのをつことができます。 x、y、zはオプションです。
グレースケールx	をグレースケールでします。は1.0または100です。
- x	にをします。
x	のを1.0または100でします。
x	の/を1.0または100でします。
するx	1.0または100をえるのでイメージをさせます。そのに、はしめます。
セピアx	を1.0または100のでセピアにします。

1. フィルタはななので、-webkitというをするがあります。やがわるかもしれませんが、はおそらくさくなるでしょう。
2. ブラウザのいバージョンではサポートされていないがあります。モバイルブラウザではにサポートされていないがあります。
3. られたサポートのため、 `filter: drop-shadow()`わりに`box-shadow`をしてみてください。 `filter: opacity()`わりに`opacity`をしてください。
4. Javascript / jQueryをしてアニメーションすることができます。 Javascriptのは、 `object.style.WebkitFilter`し`object.style.WebkitFilter`。
5. については 、 W3SchoolsまたはMDNをしてください。
6. W3Schoolsには、さまざまなのフィルタのデモページもあります。

Examples

ドロップシャドウであればボックスシャドウを

HTML

```
<p>My shadow always follows me.</p>
```

CSS

```
p {
    -webkit-filter: drop-shadow(10px 10px 1px green);
    filter: drop-shadow(10px 10px 1px green);
}
```

のフィルタ

のフィルターをするには、をスペースでります。

HTML

```
<img src='donald-duck.png' alt='Donald Duck' title='Donald Duck' />
```

CSS

```
img {
    -webkit-filter: brightness(200%) grayscale(100%) sepia(100%) invert(100%);
    filter: brightness(200%) grayscale(100%) sepia(100%) invert(100%);
```

```
}
```

HTML

```
<img src='donald-duck.png' alt='Donald Duck' title='Donald Duck' />
```

CSS

```
img {
    -webkit-filter: hue-rotate(120deg);
    filter: hue-rotate(120deg);
}
```

HTML

```
<div></div>
```

CSS

```
div {
```

```
    width: 100px;
    height: 100px;
    background-color: white;
    -webkit-filter: invert(100%);
    filter: invert(100%);
}
```

からにわります。

ぼかし

HTML

```
<img src='donald-duck.png' alt='Donald Duck' title='Donald Duck' />
```

CSS

```
img {
    -webkit-filter: blur(1px);
    filter: blur(1px);
}
```

あなたのをこすります。

オンラインでフィルタプロパティをむ https://riptutorial.com/ja/css/topic/1567/フィルタプロパティ

32: ブラウザスタイルをする

き

ブラウザには、をするためにするCSSスタイルのデフォルトセットがあります。がで、スタイルがのためにがったり、ブラウザがされたにわなかったり、ブラウザがしいHTMLのデフォルトスタイルをたないことがあるため、これらのデフォルトスタイルはブラウザでしないがあります。その、できるだけくのブラウザでデフォルトスタイルをしたいとうかもしれません。

Meyer's Resetはですが、ほぼすべてのにじをいます。これには、りしされるスタイルがじWebブラウザのインスペクタウィンドウがりしされ、ブラウザにくのがされますさらにくのにされるルールがえます。、ノーマライズは、はるかにがられており、いブラシはほとんどありません。これにより、ブラウザのがされ、ブラウザツールのがなくなります。

Examples

normalize.css

ブラウザには、のにするデフォルトのCSSスタイルがあります。これらのスタイルのには、ブラウザのをして、デフォルトのフォントやサイズのをするなど、カスタマイズすることもできます。スタイルには、ブロックレベルまたはインラインであるとされるのがまれます。

これらのデフォルトスタイルにはによってがあるため、またブラウザがにしくわないためブラウザごとになるがあるためです。

これはnormalize.cssがするところです。これはもなをにし、のバグをします。

それはをするためのものか

- くのCSSリセットとはなり、なデフォルトをします。
- いのスタイルをします。
- バグやなブラウザのをします。
- なでいをさせます。
- なコメントをしてコードがをしているかをします。

したがって、プロジェクトにnormalize.cssをめると、デザインがよりていて、なるブラウザでしてえます。

reset.cssとのい

あなたは`reset.css`についていたことがあり`reset.css`。のいはですか

normalize.cssはなるプロパティをされたデフォルトにすることによってをしますが、reset.cssはブラウザができるすべてのなスタイルをすることでをします。これははいアイデアのようにこえるかもしれませんが、にはすべてのルールをでしなければならないということです。

アプローチと

CSSのリセットはブラウザのデフォルトにするのアプローチをとります。 Eric MeyerのReset CSSはしばらくありました。のアプローチは、すぐにをきこすことがられているくのブラウザをにします。は、のバージョンv2.0 | 20110126からのCSSリセットです。

```
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
}
```

Eric MeyerさんのリセットCSS

CSSをのものでし、これらのくをにします。は、コードのバージョンv4.2.0のサンプルです。

```
/**
 * 1. Change the default font family in all browsers (opinionated).
 * 2. Correct the line height in all browsers.
 * 3. Prevent adjustments of font size after orientation changes in IE and iOS.
 */

/* Document
   ========================================================================== */

html {
  font-family: sans-serif; /* 1 */
  line-height: 1.15; /* 2 */
  -ms-text-size-adjust: 100%; /* 3 */
  -webkit-text-size-adjust: 100%; /* 3 */
}

/* Sections
   ========================================================================== */

/**
```

```
 * Remove the margin in all browsers (opinionated).
 */

body {
  margin: 0;
}

/**
 * Add the correct display in IE 9-.
 */

article,
aside,
footer,
header,
nav,
section {
  display: block;
}

/**
 * Correct the font size and margin on `h1` elements within `section` and
 * `article` contexts in Chrome, Firefox, and Safari.
 */

h1 {
  font-size: 2em;
  margin: 0.67em 0;
}
```

CSSをする

オンラインでブラウザスタイルをするをむ https://riptutorial.com/ja/css/topic/1211/ブラウザスタイルをする

33: ブラウザのサポートと

パラメーター

	ブラウザ
`-webkit-`	Google Chrome、Safari、Opera 12のバージョン、Android、Blackberry、UCブラウザ
`-moz-`	Mozilla Firefox
`-ms-`	Internet Explorer、Edge
`-o-`、`-xv-`	バージョン12までのOpera
`-khtml-`	Konquerer

ベンダープレフィックスは、がまだでされていないしいCSSのプレビューサポートをにするためにされます。

> では、ベンダーをしないことをおめします。これらのは、まだされていないしいをテストするためにし、はしないものです。にプレフィックスをしても、いブラウザのブラウザサポートはされません。なるをするためにがってもがされているとはらず、サポートしているいブラウザではがわれるがあります。
>
> いブラウザをサポートすることがなは、わりにJavaScriptやそののソリューションをしてをし、いブラウザのサポートをにするがあります。

ブラウザはをし、していないプロパティはします。

プレフィックスは、でなしののにずれます。それのは、きのプロパティできされます。これはにのになります。

ブラウザがプレフィックスのないプレフィックスきのバージョンののプロパティをサポートしている、されるもしいプロパティがされます。

Examples

トランジション

```
div {
    -webkit-transition: all 4s ease;
       -moz-transition: all 4s ease;
```

```
         -o-transition: all 4s ease;
            transition: all 4s ease;
}
```

```
div {
    -webkit-transform: rotate(45deg);
       -moz-transform: rotate(45deg);
        -ms-transform: rotate(45deg);
         -o-transform: rotate(45deg);
            transform: rotate(45deg);
}
```

オンラインでブラウザのサポートとをむ https://riptutorial.com/ja/css/topic/1138/ブラウザのサポートと

34: ポジショニング

- positionstatic | absolute | fixed | relative | sticky | initial || unset;
- Zインデックス| *number* | initial |します。

パラメーター

パラメーター	
	デフォルト。フローは、ドキュメントフローにされるでレンダリングされます。top、right、bottom、left、z-indexのプロパティはされません。
	はにしてにされているため、 `left:20px` 20ピクセルがのLEFTに20ピクセルをします
	はブラウザウィンドウにしてされます
の	は、にされたではない
	このプロパティをデフォルトにします。
する	からこのプロパティをします。
の	な。えられたオフセット·スレッショルドにするまで、そののでposition: staticようにし、 position: fixedとしてします。
されていない	とのみわせ。はこちら 。

ノーマルフローは、エレメントのがであるのエレメントのフローです。

1. によってはののをぐため、 をすることはです。

Examples

をとしてすると、ドキュメントフローからをし、そのをブラウザウィンドウにしてにすることができます。 1つのなは、いページのにスクロールするときにかがえるようにしたいときです。

```
#stickyDiv {
    position:fixed;
    top:10px;
    left:10px;
}
```

のスタックのをするには `position`プロパティを`relative` 、 `absolute`または`fixed`します、 `z-index`プロパティをします。

z-インデックスがいほど、zのみねコンテクストのにされます。

のでは、z-indexのが3のはがに、z-indexのが2のはが、z-indexのは1がでされます。

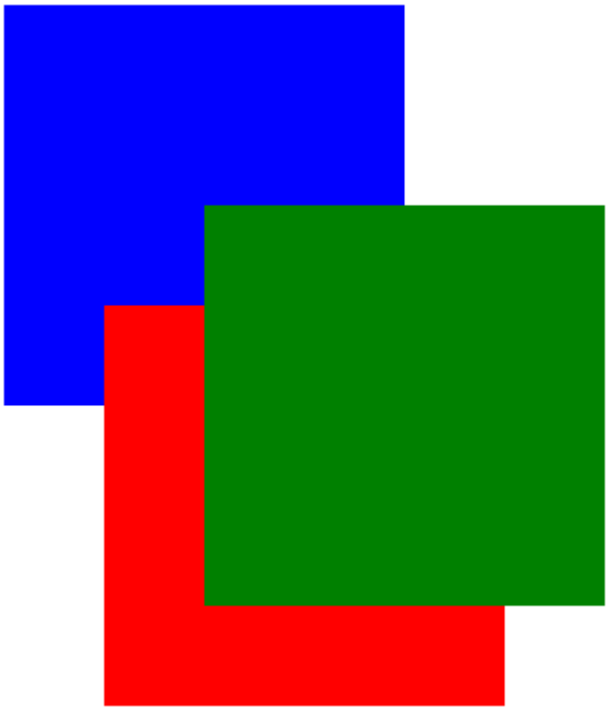
HTML

```
<div id="div1"></div>
<div id="div2"></div>
<div id="div3"></div>
```

CSS

```
div {
    position: absolute;
    height: 200px;
    width: 200px;
}
div#div1 {
    z-index: 1;
    left: 0px;
    top: 0px;
    background-color: blue;
}
div#div2 {
    z-index: 3;
    left: 100px;
    top: 100px;
    background-color: green;
}
div#div3 {
    z-index: 2;
    left: 50px;
    top: 150px;
    background-color: red;
}
```

これにより、のようながられます。



JSFiddleのをしてください。

```
z-index: [ number ] | auto;
```

パラメータ	
number	。 z-indexスタックではがきくなります。 0がデフォルトです。のもできます。
auto	にそのとじスタッキングコンテキストをえます。 デフォルト

すべてのは、 z-indexのz-indexプロパティでされたをむCSSの3Dにされます。 z-indexは、されたにしてのみしますsee なぜz-indexは、するにはされたがですか 。されるのは、 staticデフォルトです。

レイヤードプレゼンテーションとMozilla Developer Networkで、 CSSのz-indexプロパティとStacking Contextについてむことができます 。

なは、 のフローであったにしてをします。オフセットのプロパティ

1.
2.
3.
4.

がなフローのどこからしたかをすためにされます。

```
.relpos{
    position:relative;
    top:20px;
    left:30px;
}
```

このコードでは、class = "relpos"のをつをむボックスを20pxにし、30pxをのフロ―のどこからにします。

な

をすると、ののボックスがフロ―からりされ、ペ―ジののののにをえなくなります。オフセットプロパティ

1.
2.
3.
4.

そのがのでないとのでれるようにします。

```
.abspos{
    position:absolute;
    top:0px;
    left:500px;
}
```

このコードでは、`class="abspos"`をむボックスを、そのをむ0pxと500pxにします。

なめ

のデフォルトは`static`です。 MDNをするには

> このキ―ワ―ドは、がのをできるようにします。つまり、フロ―ののにされます。
> top、right、bottom、left、z-indexのプロパティはされません。

```
.element{
  position:static;
}
```

オンラインでポジショニングをむ https://riptutorial.com/ja/css/topic/935/ポジショニング

35: ボックスシャドウ

- box-shadownone | h-shadow v-shadowぼかし| inset | initial | inherit;

パラメーター

パラメーター	
インセット	デフォルトでは、はとしてわれます。 insetキーワードはフレーム/にをします。
オフセット-x	
オフセット-y	
ぼかし	デフォルトでは0です。はであってはなりません。がきければきいほど、はきくなります。
スプレッド	デフォルトでは0です。のをするとがします。のをするとがします。
	カラーキーワード、16、 `rgb()` 、 `rgba()` 、 `hsl()` 、 `hsla()` などのさまざまなをできます `rgb()`

ブラウザサポート

- Chrome 10.0
- IE 9.0
- Firefox 4.0 3.5 -moz
- Safari 5.1 3.1 - ウェブキット -
- Opera 10.5

Examples

をとす

JSFiddle https ://jsfiddle.net/UnsungHero97/80qod7aL/

HTML

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

CSS

```
.box_shadow {
  -webkit-box-shadow: 0px 0px 10px -1px #444444;
     -moz-box-shadow: 0px 0px 10px -1px #444444;
          box-shadow: 0px 0px 10px -1px #444444;
}
```

の

HTML

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

CSS

```
.box_shadow {
  background-color: #1C90F3;
  width: 200px;
  height: 100px;
  margin: 50px;
  -webkit-box-shadow: inset 0px 0px 10px 0px #444444;
     -moz-box-shadow: inset 0px 0px 10px 0px #444444;
          box-shadow: inset 0px 0px 10px 0px #444444;
}
```

JSFiddle https ://jsfiddle.net/UnsungHero97/80qod7aL/1/

をしたボトムドロップシャドウ

JSFiddle https ://jsfiddle.net/UnsungHero97/80qod7aL/2/

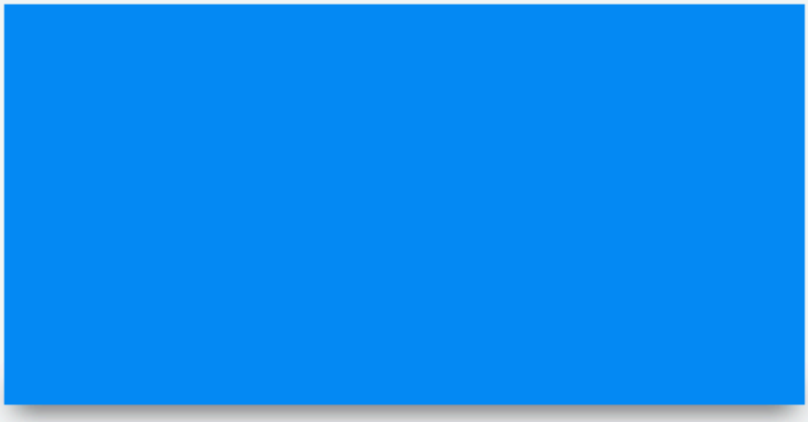
HTML

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

CSS

```
.box_shadow {
  background-color: #1C90F3;
  width: 200px;
  height: 100px;
```

```
  margin: 50px;
}

.box_shadow:after {
  content: "";
  width: 190px;
  height: 1px;
  margin-top: 98px;
  margin-left: 5px;
  display: block;
  position: absolute;
  z-index: -1;
  -webkit-box-shadow: 0px 0px 8px 2px #444444;
     -moz-box-shadow: 0px 0px 8px 2px #444444;
          box-shadow: 0px 0px 8px 2px #444444;
}
```



JSFiddle https ://jsfiddle.net/UnsungHero97/80qod7aL/5/

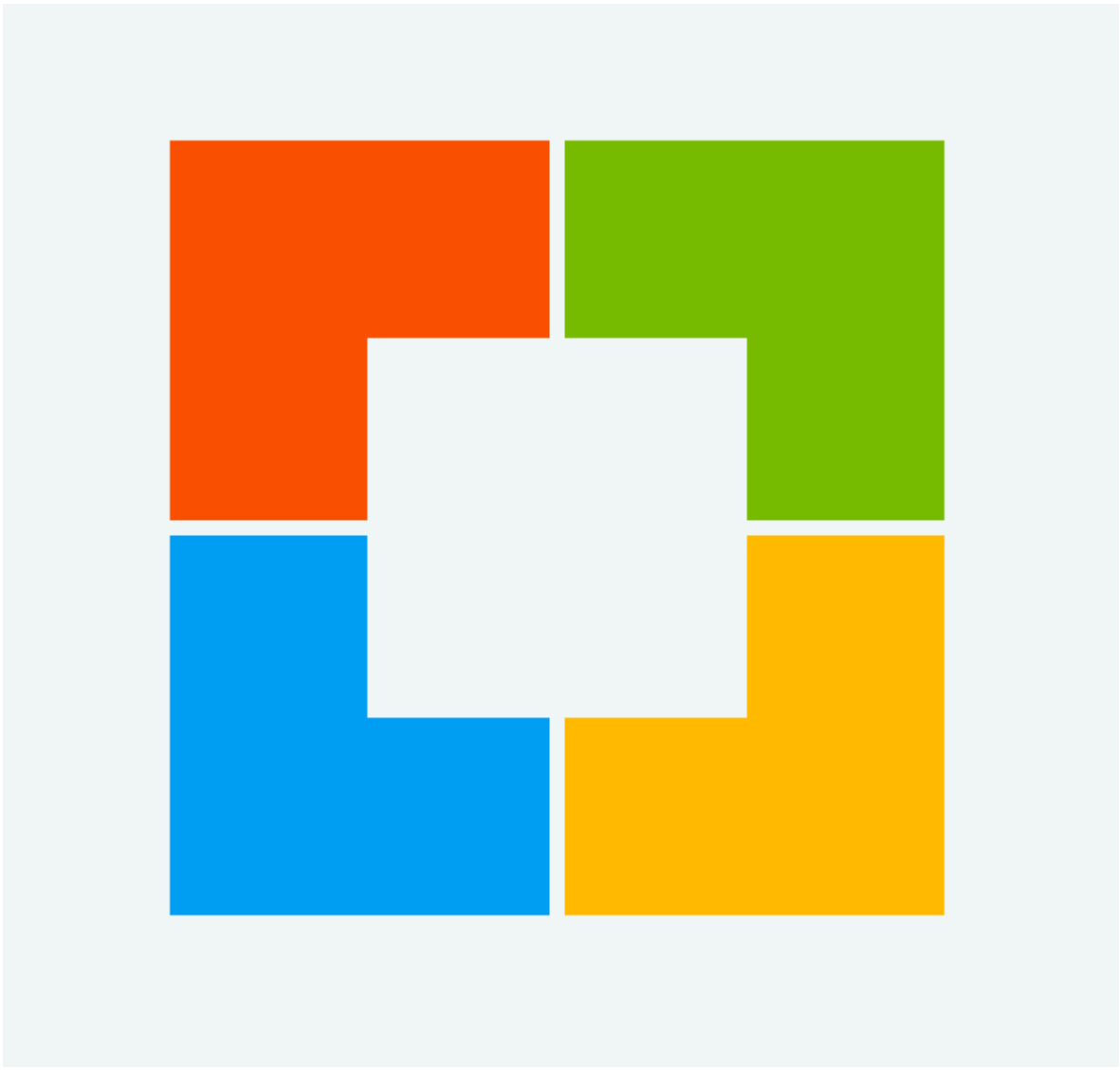
HTML

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

CSS

```
.box_shadow {
  width: 100px;
  height: 100px;
  margin: 100px;
  box-shadow:
    -52px -52px 0px 0px #f65314,
    52px -52px 0px 0px #7cbb00,
```

```
    -52px 52px 0px 0px #00a1f1,
    52px 52px 0px 0px #ffbb00;
}
```



オンラインでボックスシャドウをむ https://riptutorial.com/ja/css/topic/1746/ボックスシャドウ

36: ボックスモデル

- box-sizing パラメータ 。

パラメータ

パラメータ	
`content-box`	のときにはコンテンツのみがまれます。
`padding-box`	のときにはとパディングがまれます。
`border-box`	のときには、コンテンツ、パディング、がまれます。
`initial`	ボックスモデルをデフォルトにします。
`inherit`	のボックスモデルをします。

パディングボックスについて

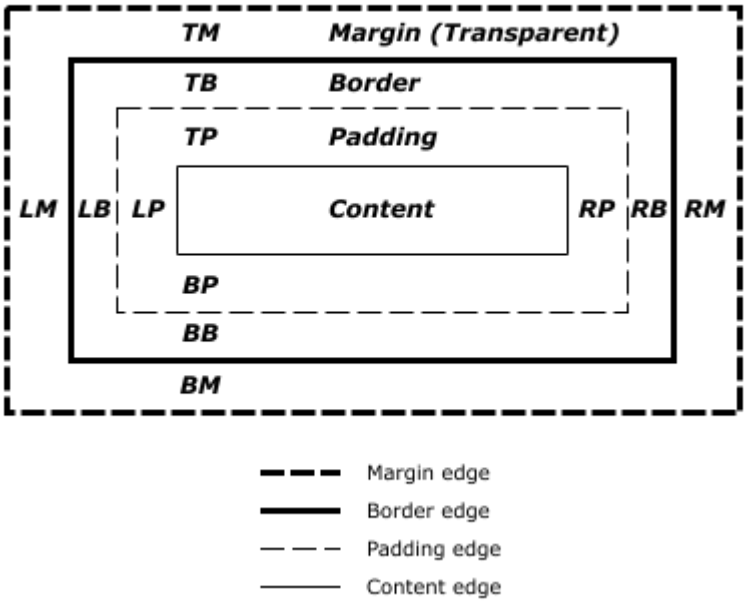
このは**Firefox**でのみされているため、しないでください。 Firefoxのバージョン50.0ではされました。

Examples

ボックスモデルとはですか

エッジ

ブラウザは、HTMLドキュメントののをします。ボックスモデルは、パディング、ボーダー、およびマージンをコンテンツにしてこのをするをします。



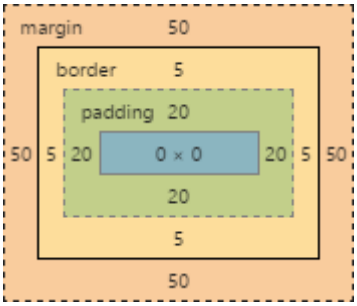
CSS2.2からの

4つののそれぞれのはエッジとばれます 。はボックスをします。

- のはコンテンツボックスです。これのとさは、のレンダリングされたテキスト、イメージ、およびそれがつのあるによってなります。
- に、 padding プロパティでされている、 パディングボックスです。 padding がされていない、パディングエッジはコンテンツエッジにしくなります。
- に、 border プロパティでされている**border**ボックスが border ます。 border がされていない、エッジはパディングエッジにしくなります。
- ものは margin プロパティでされている**margin**ボックスです。 margin がされていない、マージンエッジはボーダーエッジにしくなります。

```
div {
    border: 5px solid red;
    margin: 50px;
    padding: 20px;
}
```

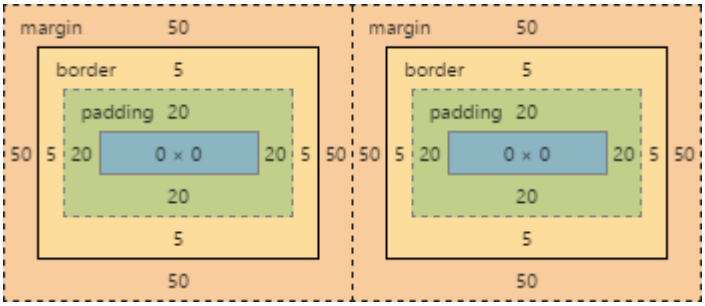
このCSSは、すべての div のスタイルを 5px の、、、のに 5px します。 50px の。 20px 、、、のパディングがあり 20px 。コンテンツをすると、されるボックスはのようになります



*Google Chrome*のスタイルパネルのスクリーンショット

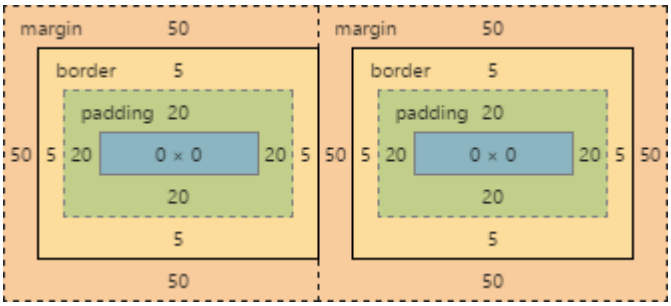
- コンテンツがないため、コンテンツのいボックスにはさやがありません0px x 0px。
- デフォルトでは、パディングボックスはコンテンツボックスとじサイズですが、 `padding`プロパティ40ピクセル×40ピクセルででしている4つのエッジの20ピクセルにえて、
- ボックスは、パディングボックスとじサイズに、 `border`プロパティ50px x 50pxででした5pxをえたものです。
- に、マージンボックスは、ボーダーボックスとじサイズと、 `margin`プロパティででしている50ピクセル150ピクセル×150ピクセルになります。

さて、たちのにじスタイルのをえましょう。ブラウザは、ののBox Modelをべて、ののコンテンツとのでしいをするをします。



のは150ピクセルのでられていますが、2つののボックスはいにしています。

のをマージンをたないようにすると、マージンエッジはボーダーエッジとじになり、2つのはのようになります。



ボックスサイズ

デフォルトのボックスモデル `content-box` は、ではありません。エレメントの`width` / `height`は、エレメントに`padding`と`border`スタイルをするとすぐにののやさをしません。

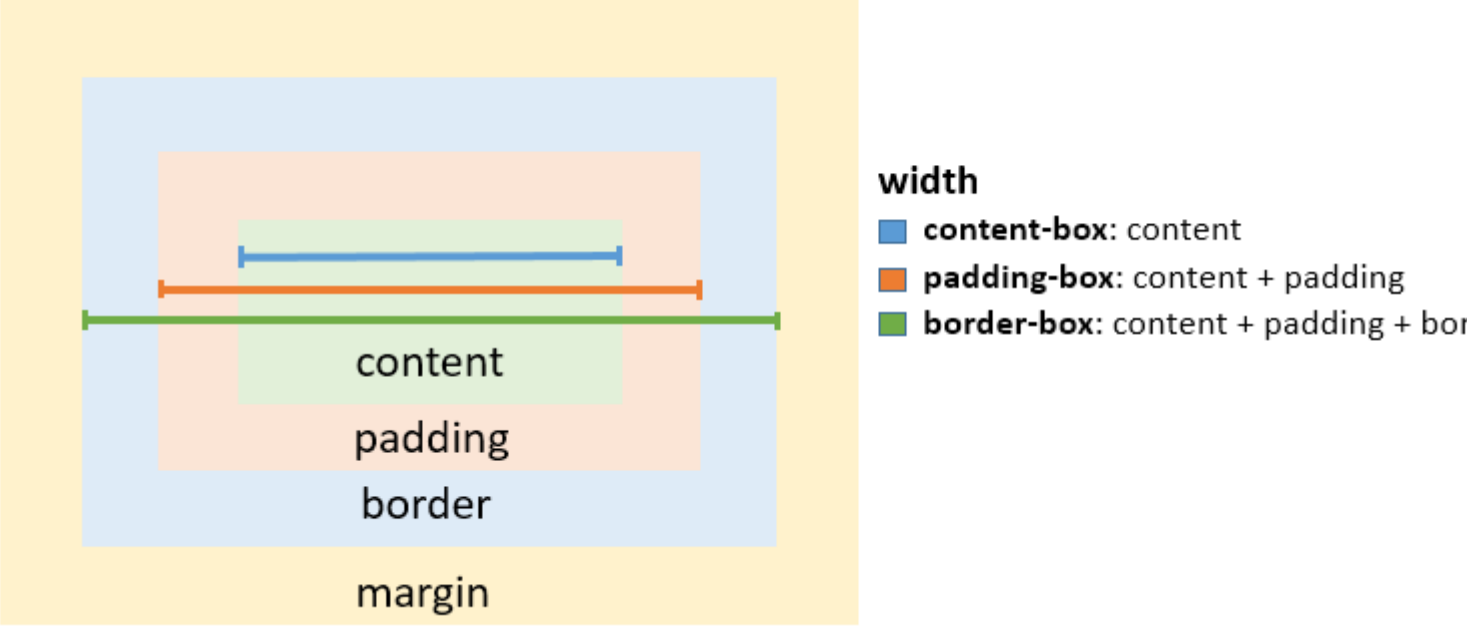
のは、 `content-box`このなをしてい`content-box`。

```
textarea {
    width: 100%;
    padding: 3px;
    box-sizing: content-box; /* default value */
}
```

パディングはテキストエリアのにされるので、のは100よりいテキストエリアになります。

いにも、CSSは、の`box-sizing`プロパティでボックスモデルをすることをにします。なプロパティには3つのなるがあります。

- `content-box` ボックスモデル - とさにはコンテンツのみがまれ、パディングやはまれません。
- `padding-box` とさにコンテンツとパディングがまれますが、はまれません。
- `border-box` とさには、め、がまれます



の`textarea`をするには、 `box-sizing`プロパティを`padding-box`または`border-box`するだけです。`border-box`がもにされます。

```
textarea {
    width: 100%;
    padding: 3px;
    box-sizing: border-box;
}
```

ページのすべてのにのボックスモデルをするには、のスニペットをします。

```
html {
    box-sizing: border-box;
}

*, *:before, *:after {
    box-sizing: inherit;
}
```

このコーディング`box-sizing:border-box;*`はされないため、々のにしてこのプロパティをにきすることができます。

オンラインでボックスモデルをむ https://riptutorial.com/ja/css/topic/646/ボックスモデル

37: マージン

- マージン <> ;
- マージン *<top>* 、 *<leftright>* 、 *<bottom>*
- マージン <> 、 <> ;
- マージン *<top>* 、 *<right>* 、 *<bottom>* 、 *<left>* ;
- margin-top *<top>* ;
- margin-right *<right>* ;
- margin-bottom *<bottom>* ;
- margin-left *<left>* ;

パラ メーター

パラ メータ	
0	マージンをnoneにする
オート	のをにすることでセンタリングにされます
px	なユニットのリストについては、 ユニットのパラメータセクションをしてください
する	からマージンをする
	にす

「りたたみマージン」のはこちら 。

Examples

のにをする

のプロパティ

CSSでは、マージンをするをすることができます。こののためにされる4つのプロパティはのとおりです。

- `margin-left`
- `margin-right`
- `margin-top`
- `margin-bottom`

のコードは、したdivのに30ピクセルのマージンをします。 をる

HTML

```
<div id="myDiv"></div>
```

CSS

```
#myDiv {
    margin-left: 30px;
    height: 40px;
    width: 40px;
    background-color: red;
}
```

パラメータ	
マージン	マージンをする。
30ピクセル	マージンの。

プロパティをしたの

`margin`プロパティをして、したのになるをすることができます。これをうためのはのとおりです。

```
margin: <top> <right> <bottom> <left>;
```

のでは、divのにゼロのマージンをします。には10ピクセルのマージン、には50ピクセルのマージン、には100ピクセルのマージンをします。 をる

HTML

```
<div id="myDiv"></div>
```

CSS

```
#myDiv {
    margin: 0 10px 50px 100px;
    height: 40px;
    width: 40px;
    background-color: red;
}
```

マージンりたたみ

2つのマージンがにおいにすると、それらはりたたまれます。 2つのマージンがにすると、それら

はしません。

マージンの

のスタイルとマークアップをえてみましょう。

```
div{
    margin: 10px;
}
```

```
<div>
    some content
</div>
<div>
    some more content
</div>
```

マージンが1つまたは2つするため、10ピクセルれています。 は2つのマージンのではありません。

の

のスタイルとマークアップをえてみましょう。

```
span{
    margin: 10px;
}
```

```
<span>some</span><span>content</span>
```

のマージンが1つまたは2つすることはないので、20ピクセルれています。 は2つのマージンのになります。

なるサイズのオーバーラップ

```
.top{
    margin: 10px;
}
.bottom{
    margin: 15px;
}
```

```
<div class="top">
    some content
</div>
<div class="bottom">
    some more content
</div>
```

これらのはに15ピクセルれてされます。マージンはなりしますが、マージンがきいほどエレメントのがまります。

マ―ジン

```
.outer-top{
    margin: 10px;
}
.inner-top{
    margin: 15px;
}
.outer-bottom{
    margin: 20px;
}
.inner-bottom{
    margin: 25px;
}
```

```
<div class="outer-top">
    <div class="inner-top">
        some content
    </div>
</div>
<div class="outer-bottom">
    <div class="inner-bottom">
        some more content
    </div>
</div>
```

2つのテキストのはどうなりますか ホバ―してえをる

> は25ピクセルです。 4つのマ―ジンはすべていにしているため、4つのマ―ジンのうちのマ―ジンをしてします。

のマ―クアップにいくつかのをするとどうなりますか

```
div{
    border: 1px solid red;
}
```

2つのテキストのはどうなりますか ホバ―してえをる

> は59pxになりますとのマ―ジンだけがいにし、のマ―ジンです。りのはでられています。だから々は1px + 10px + 1px + ~~15ピクセル~~ + 20px + 1px + 25px + 1px 1ピクセルはです...

とののマ―ジンのりたたみ

HTML

```
<h1>Title</h1>
<div>
  <p>Paragraph</p>
</div>
```

CSS

```
h1 {
  margin: 0;
  background: #cff;
}
div {
  margin: 50px 0 0 0;
  background: #cfc;
}
p {
  margin: 25px 0 0 0;
  background: #cf9;
}
```

のでは、のマ―ジンのみがされます。がh1から60pxにすることがされるかもしれませんdivのは40px、pのは20pxです。マ―ジンが1つのマ―ジンをするためににするため、これはこりません。

マ―ジンをしてペ―ジのをにえする

がブロックであり、 にされたのをつり、マ―ジンをして、ペ―ジのブロックをにセンタリングすることができます。

ウィンドウのよりもさいのをし、marginのautoプロパティはりのスペ―スをにします。

```
#myDiv {
 width:80%;
 margin:0 auto;
}
```

のでは、margin をしてに0 をとのマ―ジンにしますこれはのになります。にauto をして、ブラウザがスペ―スをにのマ―ジンにりてるようにします。

のでは、#myDivは80のにされ、りの20をしています。ブラウザはこのをりのにします

10080/ 2 = 10

マ―ジンプロパティの

```
p {
    margin:1px;                /* 1px margin in all directions */

    /*equals to:*/

    margin:1px 1px;

    /*equals to:*/

    margin:1px 1px 1px;

    /*equals to:*/

    margin:1px 1px 1px 1px;
}
```

もうつのexapmle

```
p{
    margin:10px 15px;        /* 10px margin-top & bottom And 15px margin-right & left*/

    /*equals to:*/

    margin:10px 15px 10px 15px;

    /*equals to:*/

    margin:10px 15px 10px;
    /* margin left will be calculated from the margin right value (=15px) */
}
```

のマ―ジン

マ―ジンは、のにできるいくつかのCSSプロパティの1つです。このプロパティをすると、 なしでをオ―バ―ラップさせることができます。

```
div{
  display: inline;
}

#over{
  margin-left: -20px;
}

<div>Base div</div>
<div id="over">Overlapping div</div>
```

1

`margin-left`と`margin-right`にする`margin-left`パ―セン テ―ジは、そのにしてであるとすることはらかである。

```
.parent {
    width : 500px;
    height: 300px;
}

.child {
    width : 100px;
    height: 100px;
    margin-left: 10%;  /* (parentWidth * 10/100) => 50px */
}
```

しかし、 `margin-top`と`margin-bottom`には、そうではありません。これらのプロパティはとも、コンテナのさにではなく、コンテナのにです。

そう、

```
.parent {
```

```
    width : 500px;
    height: 300px;
}

.child {
    width : 100px;
    height: 100px;
    margin-left: 10%;  /* (parentWidth * 10/100) => 50px  */
    margin-top: 20%;   /* (parentWidth * 20/100) => 100px */
}
```

オンラインでマージンをむ https://riptutorial.com/ja/css/topic/305/マージン

38: メディアクエリ

- @media [not | only] mediatypeとメディア{/ *するCSSルール* /}

パラメータ

パラメータ	
mediatype	オプションこれはメディアのタイプです。 screenのallのにあるものであるがall screen。
not	オプションこののメディアタイプにCSSをせず、のすべてにします。
media feature	CSSのユースケースをするロジック。にするオプション。
メディア	
aspect-ratio	デバイスのターゲットのアスペクトについてします。
color	デバイスのあたりのビットをします。デバイスがカラーデバイスでない、このはゼロです。
color-index	デバイスのカラールックアップテーブルのエントリをします。
grid	デバイスがグリッドデバイスであるかビットマップデバイスであるかをします。
height	さメディアフィーチャーは、デバイスのレンダリングサーフェスのさをします。
max-width	CSSは、したよりもいスクリーンではされません。
min-width	CSSは、されたよりいにはされません。
max-height	CSSはされたさよりものさにはされません。
min-height	CSSは、されたよりもいのさにはされません。
monochrome	モノクログレースケールデバイスのピクセルあたりのビットをします。
orientation	デバイスがされたをしているのみ、CSSがされます。はをしてください。
resolution	デバイスのピクセルをします。
scan	テレビデバイスのスキャンについてします。

パラメータ	
`width`	のメディアフィーチャは、デバイスのレンダリングサーフェスのドキュメントウィンドウの、またはプリンタのページボックスのなどをします。
されない	
`device-aspect-ratio`	**Deprecated**されないCSSは、さ/のがされたとするデバイスでのみされます。これは`deprecated`であり、することはされていません。
`max-device-width`	**Deprecated**れない`max-width`じですが、ブラウザのではなく、なをします。
`min-device-width`	**Deprecated** `min-width`じですが、ブラウザのではなくなをします。
`max-device-height`	**Deprecated**れない`max-height`じですが、ブラウザーのではなくなをします。
`min-device-height`	**Deprecated** `min-height`じですが、ブラウザのではなくなをします。

メディアクエリは、Chrome、Firefox、Opera、Internet Explorer 9ののブラウザでサポートされています。

`orientation`メディアはモバイルデバイスにされないことにすることがです。ビューポートのとさにづいていますウィンドウまたはデバイスではありません。

モードは、ビューポートのがビューポートのさよりもきいです。

ポートレートモードは、ビューポートのさがビューポートのよりもきいです。

これは、、デスクトップモニタがモードであることをしますが、にはポートレートであるもあります。

ほとんどの、モバイルデバイスは、ピクセルによってなるのあるのピクセルサイズではなく、をします。のピクセルサイズをするようにするには、 `head`タグのにのようにします。

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Examples

な

```
@media screen and (min-width: 720px) {
    body {
        background-color: skyblue;
    }
```

```
}
```

のメディアクエリは、2つのをします。

1. ページはのされたページ、プロジェクタなどではないでするがあります。
2. ユーザーのポートのは720ピクセルでなければなりません。

これらのがたされている、メディアクエリのスタイルがアクティブになり、ページのはになります。

メディアクエリはにされます。ページのみみでメディアクエリでされたがたされたは、CSSがされますが、がたされなくなったはすぐににになります。に、がにたされない、されたがたされるまでCSSはされません。

このでは、にユーザーのビューポートが720ピクセルをえていてもユーザーがブラウザのをすると、ユーザーがビューポートのサイズを720ピクセルにするとがになります。

リンクタグでする

```
<link rel="stylesheet" media="min-width: 600px" href="example.css" />
```

このスタイルシートはまだダウンロードされていますが、が600ピクセルをえるデバイスでのみされます。

メディアタイプ

メディアクエリにはオプションの`mediatype`パラメータがあります。このパラメータは`@media` `@media mediatype` のにかれます。たとえば、のようになります。

```
@media print {
    html {
        background-color: white;
    }
}
```

のCSSコードは、にDOM `HTML`にのをえ`HTML`。

`mediatype`パラメータはオプションであり`not`か`only`、それぞれ、されたMEDIATYPE のみされたメディアタイプをくすべてにスタイルをするを。たとえば、のコードは、 `print`をくすべてのメディアタイプにスタイルをし`print`。

```
@media not print {
    html {
        background-color: green;
    }
}
```

そしてじように、にのみするために、これをうことができます

```
@media only screen {
    .fadeInEffects {
        display: block;
    }
}
```

mediatypeのリストは、のでよりよくできます。

メディアタイプ	
all	すべてのデバイスに
screen	デフォルトのコンピュータ
print	にプリンタ。ウェブサイトのをスタイルするためにされます
handheld	PDA、、のハンドヘルド
projection	されたのために、えばプロジェクタ
aural	システム
braille	デバイス
embossed	ページきプリンタ
tv	テレビ
tty	ピッチグリッドをえたデバイス。ターミナル、ポータブル

メディアクエリをしてなるサイズをターゲットにする

くの、レスポンシブウェブデザインは、がたされたにのみされるCSSブロックであるメディアクエリをみます。メディアクエリをして、Webサイトのモバイルとデスクトップのなるスタイルをできるため、のいWebデザインにちます。

```
@media only screen and (min-width: 300px) and (max-width: 767px) {
    .site-title {
        font-size: 80%;
    }

    /* Styles in this block are only applied if the screen size is atleast 300px wide, but no
more than 767px */
}

@media only screen and (min-width: 768px) and (max-width: 1023px) {
    .site-title {
        font-size: 90%;
    }

    /* Styles in this block are only applied if the screen size is atleast 768px wide, but no
```

```
more than 1023px */
}

@media only screen and (min-width: 1024px) {
    .site-title {
        font-size: 120%;
    }

    /* Styles in this block are only applied if the screen size is over 1024px wide. */
}
```

とビュ―ポ―ト

メディアクエリで「」をする、メタタグをしくすることがです。なメタタグはこのようにえ、<head>タグのにれるがあります。

```
<meta name="viewport" content="width=device-width,initial-scale=1">
```

なぜこれがなのですか

MDNのについて「」は

> のメディアフィ―チャは、デバイスのレンダリングサ―フェスのドキュメントウィンドウの、またはプリンタのペ―ジボックスのなどをします。

どういうですか

View-portはデバイスのです。が1280 x 720のであれば、ビュ―ポ―トのは「1280ピクセル」になります。

くの、くのデバイスがなるピクセルをりてて1つのピクセルをします。たとえば、iPhone 6 Plusには1242 x 2208のがあります。のビュ―ポ―トとビュ―ポ―トさは414 x 736です。つまり、1ピクセルをするために3ピクセルがされます。

しかし、 metaしくしなかったは、のでウェブペ―ジをしようとします。その、されたビュ―さなテキストやがされます。

およびのメディアクエリ

これはWebKitベ―スのブラウザでのみしますが、これはにちます

```
/* ----------- Non-Retina Screens ----------- */
@media screen
  and (min-width: 1200px)
  and (max-width: 1600px)
  and (-webkit-min-device-pixel-ratio: 1) {
}

/* ----------- Retina Screens ----------- */
@media screen
  and (min-width: 1200px)
```

```
  and (max-width: 1600px)
  and (-webkit-min-device-pixel-ratio: 2)
  and (min-resolution: 192dpi) {
}
```

ディスプレイには2のピクセルがあります。 1つはピクセルで、もう1つはピクセルです。ほとんどの、ピクセルは、すべてのディスプレイデバイスでじなので、にじままです。ピクセルは、よりのピクセルをするためのデバイスのについてする。デバイスピクセルは、ピクセルとピクセルのです。たとえば、ながの2であるため、MacBook Pro Retina、iPhone 4ではデバイスのピクセルは2です。

これがWebKitベースのブラウザでのみするは、のによるものです。

- ルールのにベンダープレフィックス`-webkit-`。
- これは、WebKitとBlinkのエンジンではされていません。

と

メディアクエリは 、 メディアタイプとばれるデバイス/メディアスクリーン、プリント、ハンドヘルドなどのタイプについてCSSルールをすることをにし、デバイスののは、カラーまたはビューポートディメンションのなどのメディアでされます 。

メディアクエリのな

```
@media [...] {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

メディアタイプをむメディアクエリ

```
@media print {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

メディアタイプとメディアをむメディアクエリ

```
@media screen and (max-width: 600px) {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

メディアフィーチャおよびのメディアタイプ「

すべて」をむメディアクエリ

```
@media (orientation: portrait) {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

メディアクエリとIE8

IE8では、 メディアクエリはまったくサポートされていません。

Javascriptベースの

IE8のサポートをするには、いくつかのJSソリューションの1つをできます。たとえば、 レスポンスをして、IE8のメディアクエリサポートをするには、のコードをします。

```
<!--[if lt IE 9]>
<script
   src="respond.min.js">
</script>
<![endif]-->
```

CSS Mediaqueriesはじことをするのライブラリです。そのライブラリをHTMLにするコードはじです

```
<!--[if lt IE 9]>
<script
   src="css3-mediaqueries.js">
</script>
<![endif]-->
```

JSベースのソリューションがにらないは、IE <9のみのスタイルシートをすることをしてください。そのためには、のHTMLをコードにするがあります。

```
<!--[if lt IE 9]>
<link rel="stylesheet" type="text/css" media="all" href="style-ielt9.css"/>
<![endif]-->
```

にはもう1つのがあります **CSS**ハックをってIE <9をターゲットにします。 IE <9スタイルシートとじがありますが、それにはのスタイルシートはありません。しかし、はこのオプションをしません。なぜなら、それらがなCSSコードをするからですそれは、のCSSハックのがではないの1つです。

オンラインでメディアクエリをむ https://riptutorial.com/ja/css/topic/317/メディアクエリ

39: レイアウト

- なし|インライン|ブロック|リスト||インラインリストアイテム||インラインブロック|インラインテーブル|テーブル|テーブルセル|||||||フレックス|インラインフレックス|グリッド|インライングリッド|ランイン|ルビー|ルビーベース|ルビーテキスト|ルビーベース|ルビーテキストコンテナ|

パラメーター

none	をにし、スペースをしないようにします。
block	をブロックし、なの100をめ、のにブレークします。
inline	インライン。をたず、のにブレークをれません。
inline-block	インラインとブロックのからなプロパティをしますが、ブレークはありませんが、をつことができます。
inline-flex	をインラインレベルのフレックスコンテナとしてします。
inline-table	このは、インラインレベルのとしてされます。
grid	ブロックのようにし、グリッドモデルにってコンテンツをレイアウトします。
flex	ブロックのようにし、flexboxモデルにってコンテンツをレイアウトします。
inherit	からをします。
initial	を、HTMLにされているまたはブラウザ/ユーザのデフォルトスタイルシートからしたデフォルトにリセットします。
table	HTMLの`table`のようにします。
table-cell	が`<td>`のようにるうようにする
table-column	を`<col>`のようにさせる
table-row	が`<tr>`のようにるうようにする
list-item	が`<li>`のようにするようにします。

Examples

display CSSプロパティは、HTMLドキュメントのレイアウトとフローをするためのなです。ほとんどのは、 blockまたはinlineデフォルトdisplayをちますただし、のにはのデフォルトがあります。

をなして

inlineはなだけをします。じタイプののとにみねられ、のインラインをむことはできません。

```
<span>This is some <b>bolded</b> text!</span>
```

This is some **bolded** text!

でしたように、2つのinlineと<b>はインラインしたがってであり、テキストのれをさない。

ブロック

blockは、その "のなをめます。それはしいからまり、 inlineとはに、それがむかもしれないのをしません。

```
<div>Hello world!</div><div>This is an example!</div>
```

Hello world!
This is an example!

divはデフォルトではブロックレベルであり、にしたように、2つのblockがにみねられ、 inlineとはなり、テキストのれがれます。

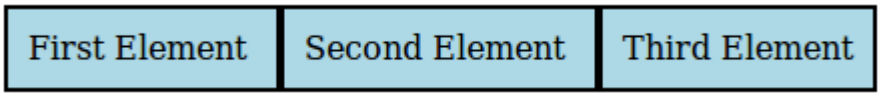
インラインブロック

inline-blockは、 inlineににえるをぼさないpadding 、 margin 、 heightなどのプロパティをできるようにしながら、エレメントをテキストのフローとブレンドして、ののベストをします。

このをつは、のテキストのようにし、その、 text-alignなどのテキストのれをするのをけtext-align 。 デフォルトでは、コンテンツをめるためにサイズにされます。

```
<!--Inline: unordered list-->
<style>
li {
    display : inline;
    background : lightblue;
    padding:10px;
```

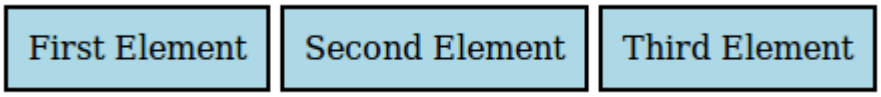
```
    border-width:2px;
    border-color:black;
    border-style:solid;
    }
</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```




```
<!--block: unordered list-->
<style>
li {
    display : block;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }
</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```


```
<!--Inline-block: unordered list-->
<style>
li {
    display : inline-block;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }

</style>
```

```
<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```



し

displayプロパティにnoneをしたは、まったくされません。

たとえば、 `myDiv`というIDをつdivをしましょう。

```
<div id="myDiv"></div>
```

これで、のCSSルールではされないとマークできます。

```
#myDiv {
    display: none;
}
```

が`display:none;`されている`display:none;`ブラウザはそのの `position`と`float`ののすべてのレイアウトプロパティをします。そのにしてボックスはレンダリングされず、htmlでのそのはののにしません。

これは`visibility`プロパティを`hidden`するのとはなることにしてください。 `visibility: hidden;` `visibility: hidden;`がページにをすることはないが、はレンダリングプロセスのを、あたかもであるかのようにすることになる。したがって、のがページにどのようにされるかにします。

displayプロパティの`none`は、JavaScriptをしてにをまたはにするためににされるため、にをしてするはありません。

divをしていテーブルをするには

これはのHTMLテーブルです

```
<style>
    table {
        width: 100%;
    }
</style>

<table>
  <tr>
    <td>
```

```
        I'm a table
    </td>
  </tr>
</table>
```

このようにじをうことができます

```
<style>
    .table-div {
        display: table;
    }
    .table-row-div {
        display: table-row;
    }
    .table-cell-div {
        display: table-cell;
    }
</style>

<div class="table-div">
  <div class="table-row-div">
    <div class="table-cell-div">
      I behave like a table now
    </div>
  </div>
</div>
```

オンラインでレイアウトをむ https://riptutorial.com/ja/css/topic/1473/レイアウト

40:

- *に0と1の||の|されていない。

をしたくないは、わりにこれをうことができます

rgba255、255、255、0.6;

リソース

- MDN https : //developer.mozilla.org/en/docs/Web/CSS/opacity ;
- W3C「」プロパティ https ://www.w3.org/TR/css3-color/#transparency
- ブラウザのサポート http : //caniuse.com/#feat=css-opacity

Examples

プロパティ

のは、 `opacity`プロパティをしてできます。は、 `0.0` から`1.0` までののにすることができます。

```
<div style="opacity:0.8;">
    This is a partially transparent element
</div>
```

プロパティ	
`opacity: 1.0;`	
`opacity: 0.75;`	2575
`opacity: 0.5;`	5050
`opacity: 0.25;`	7525
`opacity: 0.0;`	トランスペアレント

`opacity`にするIEの

IEのすべてのバージョンで`opacity`をするはのとおりです。

```
.transparent-element {
  /* for IE 8 & 9 */
  -ms-filter:"progid:DXImageTransform.Microsoft.Alpha(Opacity=60)"; // IE8
  /* works in IE 8 & 9 too, but also 5, 6, 7 */
  filter: alpha(opacity=60); // IE 5-7
  /* Modern Browsers */
  opacity: 0.6;
```

```
}
```

オンラインでをむ https://riptutorial.com/ja/css/topic/2864/

41:

- カウントauto | number || initial | unset;
- |さ;
- [] | []。
- スパンなし|すべて|||。
- ギャップnormal | length || initial | unset;
- りつぶしauto | balance || intial | unset;
- column-rule-colorcolor | inherit | initial | unset;
- column-rule-stylenone | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset || initial | unset;
- ルール||||||;
- ルール[ルール] | [ルールスタイル] | [ルール]。
- break-afterauto | always | left | right | recto | verso | page | column | region |||||;
- break-beforeauto | always | left | right | recto | verso | page | column | region || - ページ| - |;
- break-insideauto | avoid | - ページ| - |;

Examples

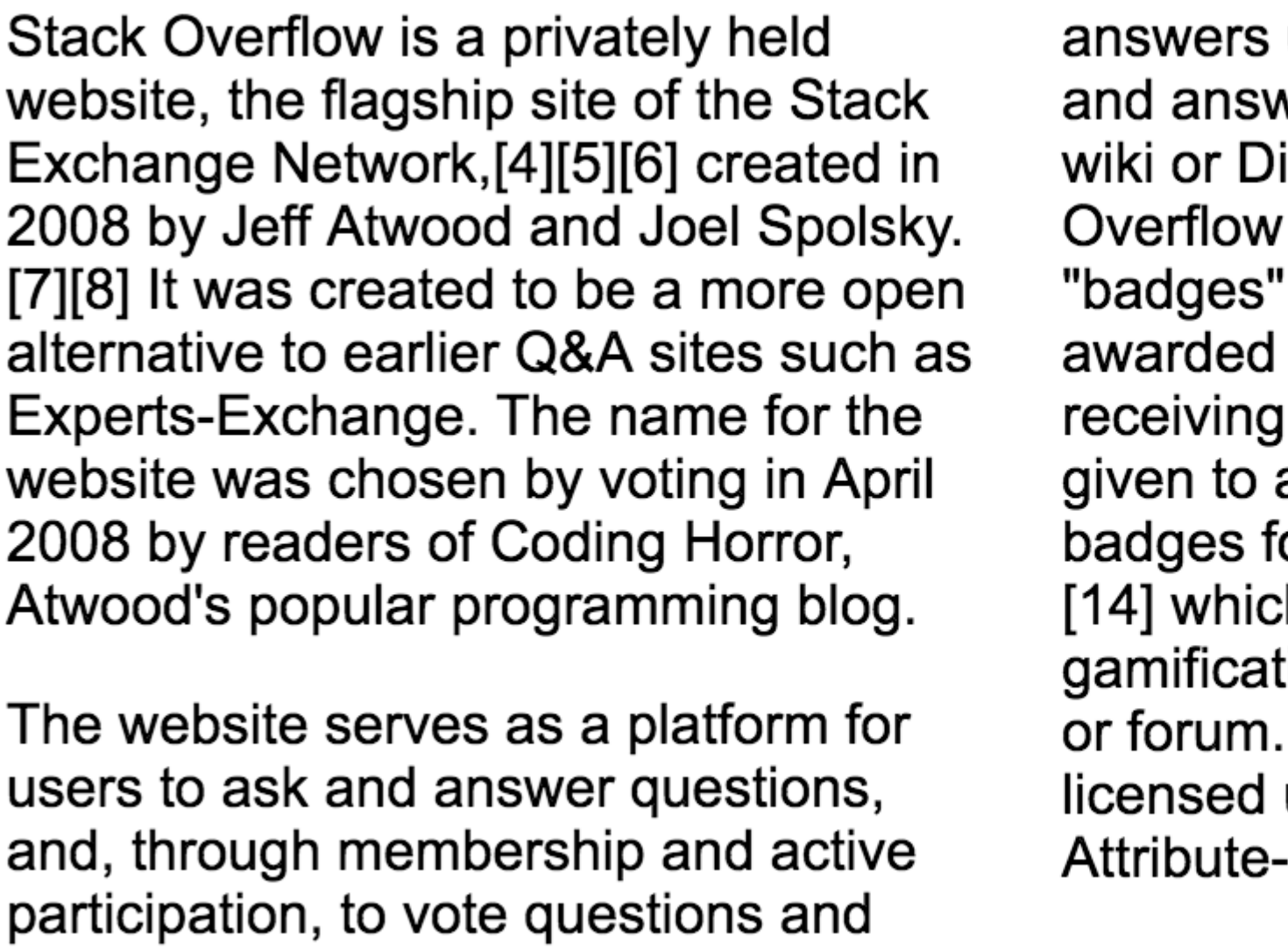
なカウント

CSSののレイアウトにより、のテキストをにできます。

コード

```
<div id="multi-columns">Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis
nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure
dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim
id est laborum</div>
```

```
.multi-columns {
  -moz-column-count: 2;
  -webkit-column-count: 2;
  column-count: 2;
}
```



`column-width`プロパティは、のをします。 `column-count`がされていない、ブラウザはなにまるのをします。

コーード

```
<div id="multi-columns">
    Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum
</div>
```

```
.multi-columns {
  -moz-column-width: 100px;
  -webkit-column-width: 100px;
  column-width: 100px;
}
```

Stack Overflow is a privately held website, the flagship site of the Stack Exchange Network,[4][5] [6] created in 2008 by Jeff Atwood and Joel Spolsky. [7][8] It was created to be a more open alternative to earlier Q&A sites such as Experts-	Exchange. The name for the website was chosen by voting in April 2008 by readers of Coding Horror, Atwood's popular programming blog. The website serves as a platform for users to ask and answer	questions, and, through membership and active participation, to vote questions and answers up or down and edit questions and answers in a fashion similar to a wiki or Digg.[13] Users of Stack Overflow can earn reputation	points and "badges"; for example, a person is awarded 10 reputation points for receiving an "up" vote on an answer given to a question, and can receive badges for their valued contributions, [14] which represents a kind of	gamification of the traditional Q&A site or forum. All user-generated content is licensed under a Creative Commons Attribute-ShareAlike license.

オンラインでをむ https://riptutorial.com/ja/css/topic/3042/

42: シェイプ

Examples

をするには、とさのでをします。のでは、 `width`と`height`がそれぞれ100ピクセルのがあります。

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

```
.square {
    width: 100px;
    height: 100px;
    background: rgb(246, 156, 85);
}
```

CSSのをするには、とさが0ピクセルのをします。のプロパティをしてがされます。さとが0のの、4つのボーダー、、、はそれぞれをします。ここにはさ/が0のとのついた4つのがあります。

いくつかのをに、のをにすることによって、さまざまなをできます。たとえば、では、のをのにし、のをにします。ここでは、がどのようにされているかをすために、とのがしをつけたをします。

のきさは、なるのをすることによってできます - のいもの、いもの、のものなど。のはすべて50

x50ピクセルのをしています。

- ポインティングアップ

```
<div class="triangle-up"></div>
```

```
.triangle-up {
  width: 0;
  height: 0;
  border-left: 25px solid transparent;
  border-right: 25px solid transparent;
  border-bottom: 50px solid rgb(246, 156, 85);
}
```

- をす

```
<div class="triangle-down"></div>
```

```
.triangle-down {
  width: 0;
  height: 0;
  border-left: 25px solid transparent;
  border-right: 25px solid transparent;
  border-top: 50px solid rgb(246, 156, 85);
}
```

- をす

```
<div class="triangle-right"></div>
```

```
.triangle-right {
```

```
  width: 0;
  height: 0;
  border-top: 25px solid transparent;
  border-bottom: 25px solid transparent;
  border-left: 50px solid rgb(246, 156, 85);
}
```

- をす

```
<div class="triangle-left"></div>
```

```
.triangle-left {
  width: 0;
  height: 0;
  border-top: 25px solid transparent;
  border-bottom: 25px solid transparent;
  border-right: 50px solid rgb(246, 156, 85);
}
```

- き/き

```
<div class="triangle-up-right"></div>
```

```
.triangle-up-right {
  width: 0;
  height: 0;
  border-top: 50px solid rgb(246, 156, 85);
  border-left: 50px solid transparent;
}
```

- き/き

```
<div class="triangle-up-left"></div>
```

```
.triangle-up-left {
  width: 0;
  height: 0;
  border-top: 50px solid rgb(246, 156, 85);
  border-right: 50px solid transparent;
}
```

- /をする

```
<div class="triangle-down-right"></div>
```

```
.triangle-down-right {
  width: 0;
  height: 0;
  border-bottom: 50px solid rgb(246, 156, 85);
  border-left: 50px solid transparent;
}
```

- /をす

```
<div class="triangle-down-left"></div>
```

```
.triangle-down-left {
  width: 0;
  height: 0;
  border-bottom: 50px solid rgb(246, 156, 85);
  border-right: 50px solid transparent;
}
```

バースト

バーストはにていますが、ポイントはかられています。バーストのをにして、ししたのがになっているとえてください。

のは、 `::before`および`::after`をしてされます。

8ポイントバースト

8バーストは、2つののである。のはであり、のは`:before`をしてされます。 を20°させ、のを135°させます。

```
<div class="burst-8"></div>

.burst-8 {
  background: rgb(246, 156, 85);
  width: 40px;
  height: 40px;
  position: relative;
  text-align: center;
  -ms-transform: rotate(20deg);
  transform: rotate(20eg);
}

.burst-8::before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  height: 40px;
  width: 40px;
  background: rgb(246, 156, 85);
  -ms-transform: rotate(135deg);
  transform: rotate(135deg);
}
```

12ポイントバースト

12ポイントバーストは3つのレイヤードスクエアです。のはであり、のは`:before`と`:after`をしてされます。 を0°させ、のを30°させ、を60°させます。

```
<div class="burst-12"></div>

.burst-12 {
  width: 40px;
  height: 40px;
  position: relative;
  text-align: center;
  background: rgb(246, 156, 85);
```

```
}

.burst-12::before, .burst-12::after {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  height: 40px;
  width: 40px;
  background: rgb(246, 156, 85);
}

.burst-12::before {
  -ms-transform: rotate(30deg);
  transform: rotate(30deg);
}

.burst-12::after {
  -ms-transform: rotate(60deg);
  transform: rotate(60deg);
}
```

と

サークル

をするには、 `width`と`height`しい をし、このの`border-radius`プロパティを`50%`ます。

HTML

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

CSS

```
.circle {
   width: 50px;
   height: 50px;
   background: rgb(246, 156, 85);
   border-radius: 50%;
}
```

はにていますが、 `width`と`height`がなります。

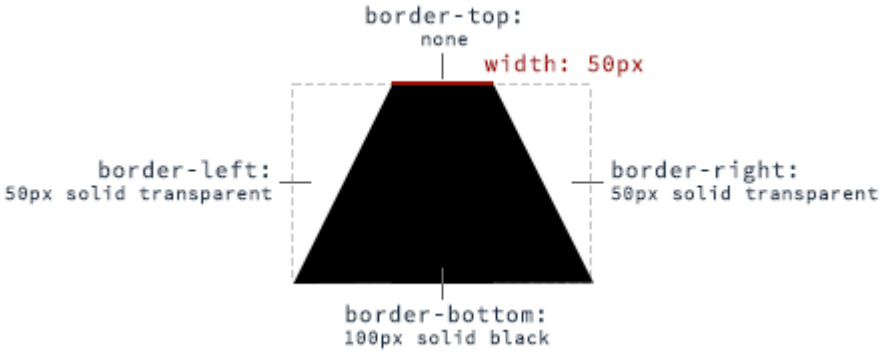
HTML

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

CSS

```
.oval {
  width: 50px;
  height: 80px;
  background: rgb(246, 156, 85);
  border-radius: 50%;
}
```

は、さゼロさ`0px`、がゼロよりきく、をいてであるブロックでることができます



HTML

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

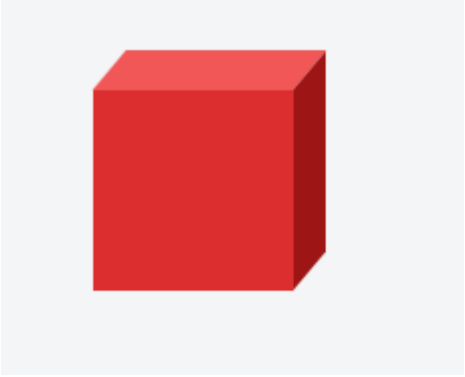
CSS

```
.trapezoid {
    width: 50px;
    height: 0;
    border-left: 50px solid transparent;
    border-right: 50px solid transparent;
    border-bottom: 100px solid black;
}
```

ボーダーサイドをすることで、のきをすることができます。

キューブ

このは、にして2Dメソッド`skewX()`および`skewY()`をしてキューブをするをしています。



HTML

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

CSS

```
.cube {
  background: #dc2e2e;
  width: 100px;
  height: 100px;
  position: relative;
  margin: 50px;
}

.cube::before {
  content: '';
  display: inline-block;
  background: #f15757;
  width: 100px;
  height: 20px;
  transform: skewX(-40deg);
  position: absolute;
  top: -20px;
  left: 8px;
}

.cube::after {
  content: '';
  display: inline-block;
  background: #9e1515;
  width: 16px;
  height: 100px;
  transform: skewY(-50deg);
  position: absolute;
  top: -10px;
  left: 100%;
}
```

デモをる

ピラミッド

このは、にしてと2Dメソッド`skewY()`と`rotate()`をしてピラミッドをするをしています。

HTML

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

CSS

```
.pyramid {
  width: 100px;
  height: 200px;
  position: relative;
  margin: 50px;
}

.pyramid::before,.pyramid::after {
  content: '';
  display: inline-block;
  width: 0;
  height: 0;
  border: 50px solid;
  position: absolute;
}

.pyramid::before {
  border-color: transparent transparent #ff5656 transparent;
  transform: scaleY(2) skewY(-40deg) rotate(45deg);
}

.pyramid::after {
  border-color: transparent transparent #d64444 transparent;
  transform: scaleY(2) skewY(40deg) rotate(-45deg);
}
```

オンラインでシェイプをむ https://riptutorial.com/ja/css/topic/2862/シェイプ

43: センタリング

これは、の `width`と`height` がまたはでないにされます。

Flexboxは、ユーザーインターフェイスのにされているため、のすべてのオプションよりもすることをおめします。

Examples

ディスプレイのセンタリングテーブル

HTML

```
<div class="wrapper">
    <div class="outer">
        <div class="inner">
            centered
        </div>
    </div>
</div>
```

CSS

```
.wrapper {
  height: 600px;
  text-align: center;
}
.outer {
  display: table;
  height: 100%;
  width: 100%;
}
.outer .inner {
  display: table-cell;
  text-align: center;
  vertical-align: middle;
}
```

トランスフォームによるセンタリング

HTML

```
<div class="wrapper">
    <div class="centered">
        centered
    </div>
</div>
```

CSS

```
.wrapper {
  position: relative;
  height: 600px;
}
.centered {
  position: absolute;
  z-index: 999;
  transform: translate(-50%, -50%);
  top: 50%;
  left: 50%;
}
```

Flexboxによるセンタリング

HTML

```
<div class="container">
    <div class="child"></div>
</div>
```

CSS

```
.container {
  height: 500px;
  width: 500px;
  display: flex;              // Use Flexbox
  align-items: center;        // This centers children vertically in the parent.
  justify-content: center;    // This centers children horizontally.
  background: white;
}

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

のさによるテキストのセンタリング

HTML

```
<div class="container">
    <span>vertically centered</span>
</div>
```

CSS

```
.container{
    height: 50px;           /* set height */
    line-height: 50px;      /* set line-height equal to the height */
    vertical-align: middle; /* works without this rule, but it is good having it explicitly
set */
}
```

このメソッドは、 *1*のテキストをにのみします 。ブロックがしくされることはなく、しいにされたは2つのにいのテキストがされます。

ポジションでセンタリング

HTML

```
<div class="wrapper">
  <img src="http://cdn.sstatic.net/Sites/stackoverflow/company/img/logos/so/so-
icon.png?v=c78bd457575a">
</div>
```

CSS

```
.wrapper{
   position:relative;
   height: 600px;
}
.wrapper img {
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  margin: auto;
}
```

のをにしたいは、そののさとをするがあります。

HTML

```
<div class="wrapper">
  <div class="child">
     make me center
  </div>
</div>
```

CSS

```
.wrapper{
   position:relative;
   height: 600px;
}
.wrapper .child {
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  margin: auto;
  width: 200px;
  height: 30px;
  border: 1px solid #f00;
}
```

によるセンタリング

HTML

```
<div class="wrapper">
  <div class="content"></div>
</div>
```

CSS

```
.wrapper{
   min-height: 600px;
}

.wrapper:before{
  content: "";
  display: inline-block;
  height: 100%;
  vertical-align: middle;
}

.content {
  display: inline-block;
  height: 80px;
  vertical-align: middle;
}
```

このメソッドは、さまざまなさの`.content`が`.wrapper`に`.wrapper`されているにもよくされます。`.content`のさが`.wrapper`のmin-heightをえたときに`.wrapper`のさがするようにし`.content`。

オンラインでセンタリングをむ https://riptutorial.com/ja/css/topic/5070/センタリング

44:

-
- borderborder-border border-style border-colorの|する;
- border-topborder-width border-style border-color |の|する;
- border-bottomborder-width border-style border-color |の|する;
- border-leftborder-width border-styleボーダー - カラー|の|する;
- border-rightborder-width border-style border-color |の|する;
- ボーダースタイル
- border-style1-4なし|された|する| |ソリッド|ダブル|グルーブ| |インセット|まり|の|する;
- ボーダー
- border-radius1-4 length | / 1-4さ| |の|する;
- border-top-left-radiusさ| [さ|] |の|する;
- border-top-right-radiusさ| [さ|] |の|する;
- border-bottom-left-radiusさ| [さ|] |の|する;
- border-bottom-right-radiusさ| [さ|] |の|する;
- ボーダーイメージ
- border-image-borderborder-image-repeat [border-image-width [border-image-outset]] border-image-repeat
- border-image-sourcenone |;
- border-image-slice1-4|パーセント[りつぶし]
- border-image-repeat1-2ストレッチ|りし|ラウンド|スペース
-
- | |の|する

プロパティ

-

- ボ—ダ—ボトム
- ボ—ダ—ボトムカラ—
- border-bottom-left-radius
- border-bottom-right-radius
- ボ—ダ—ボトムスタイル
- border-bottom-width
- ボ—ダの
- ボ—ダ—イメ—ジ
- り
- border-image-repeat
- ボ—ダ—イメ—ジスライス
- border-image-source
- border-image-width
- ボ—ダ—
- ボ—ダ—- -カラ—
- ボ—ダ—- -スタイル
- border-left-width
- ボ—ダ—
- ボ—ダ—
- ボ—ダ—- -カラ—
- ボ—ダ—- -スタイル
- border-right-width
- ボ—ダ—スタイル
- ボ—ダ—トップ
- ボ—ダ—トップカラ—
- border-top-left-radius

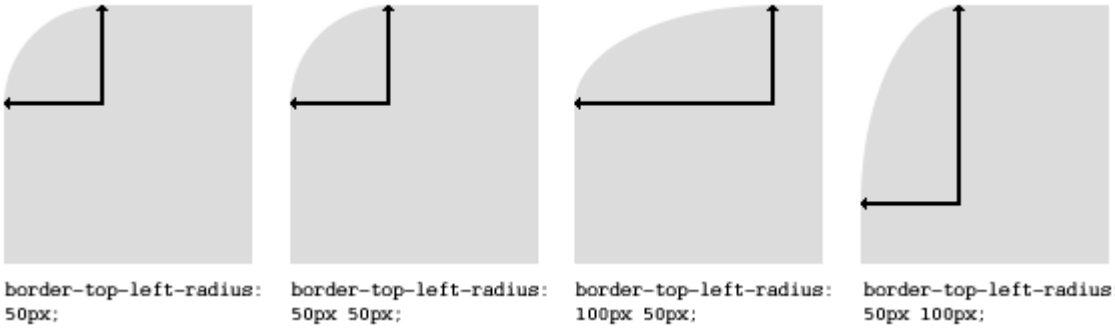
- ボーダー - -
- ボーダートップスタイル
- ボーダートップ
- ボーダー

Examples

ボーダー

border-radiusプロパティでは、ボックスモデルのをできます。

のコーナーには、そのコーナーのと8にして2つのをできます。



のセットはをします。オプションの2のセットは、 '/'のにをします。のセットが1つだけされたは、とのにされます。

```
border-radius: 10px 5% / 20px 25em 30px 35em;
```

10 `10px`は、とのです。そして、 `5%`はとののです。 '/'のの4つのは、、、およびのです。

くのCSSプロパティとに、のまたはすべてのなにしてをできます。したがって、18のをできます。のでは、すべてのコーナーのおよびをじにできます。

HTML

```
<div class='box'></div>
```

CSS

```
.box {
    width: 250px;
    height: 250px;
    background-color: black;
    border-radius: 10px;
}
```

Border-radiusは、ボックスをにするためにもにされます。 border-radiusをのさのにすると、がされます。

```
.circle {
    width: 200px;
    height: 200px;
    border-radius: 100px;
}
```

border-radiusはパーセンテージをけれるので、border-radiusをでするのをけるために50をするのがです。

```
.circle {
    width: 150px;
    height: 150px;
    border-radius: 50%;
}
```

widthプロパティとheightプロパティがしくない、のシェイプはではなくなります。

ブラウザのborder-radiusの

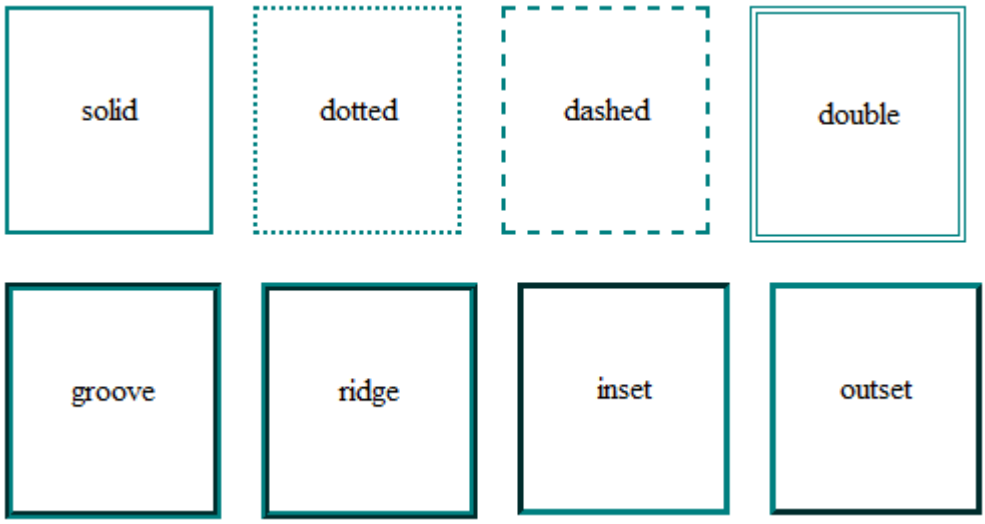
```
  -webkit-border-top-right-radius: 4px;
  -webkit-border-bottom-right-radius: 4px;
  -webkit-border-bottom-left-radius: 0;
  -webkit-border-top-left-radius: 0;
  -moz-border-radius-topright: 4px;
  -moz-border-radius-bottomright: 4px;
  -moz-border-radius-bottomleft: 0;
  -moz-border-radius-topleft: 0;
  border-top-right-radius: 4px;
  border-bottom-right-radius: 4px;
  border-bottom-left-radius: 0;
  border-top-left-radius: 0;
```

ボーダースタイル

`border-style`プロパティは、ののスタイルをします。このプロパティは14のをつことができますのすべてのに1つのがあります。

```
border-style: dotted;
```

```
border-style: dotted solid double dashed;
```



`border-style`は`none`と`hidden`をつこともできます。 `<table>`ののののための`hidden`をいて、じがあります。のをつ`<table>`では、がもい`none`はあり`none` がされ、がされます。 `hidden`ていることがもされますの、はされません。

ほとんどの、のすべてのにしていくつかののプロパティ `border-width` 、 `border-style` 、 `border-color` をするがあります。

くのではなく、

```
border-width: 1px;
border-style: solid;
border-color: #000;
```

あなたはにくことができます

```
border: 1px solid #000;
```

これらのは、のについてもできます `border-top` 、 `border-left` 、 `border-right` 、 `border-bottom` 。だからあなたはできる

```
border-top: 2px double #aaaaaa;
```

ボーダーイメージ

`border-image`プロパティでは、のスタイルのわりにイメージをするようにすることができます。

`border-image`に

- `border-image-source` するへのパス
- `border-image-slice` を**9**つの４ 、４ 、 にするためのオフセットをし`border-image-slice`。
- `border-image-repeat` のとのをどのようにスケーリングするかをします

のをえてみましょう。wheras border.pngは90x90ピクセルのです

```
border-image: url("border.png") 30 stretch;
```

は30x30ピクセルの9つのにされます。はのとしてされ、はのとしてされます。が30pxよりきい/い、のこのがびます。のはになっています。

ボーダー - [|||]

`border-[left|right|top|bottom]`プロパティは、ののにをするためにされます。

たとえば、のにをするは、のようにします。

```
#element {
    border-left: 1px solid black;
}
```

`border-collapse`プロパティは、 `table` および`display: table`または`inline-table`として`display: table`されるにのみされ、ののがのにりたたまれているのか、HTMLのようにりされているのかをします。

```
table {
  border-collapse: separate; /* default */
  border-spacing: 2px; /* Only works if border-collapse is separate */
}
```

「 - ドキュメント」のもしてください。

の

アウトラインの

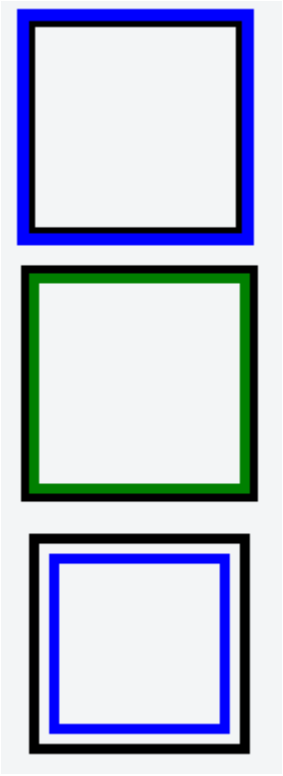
```
.div1{
  border: 3px solid black;
  outline: 6px solid blue;
  width: 100px;
  height: 100px;
  margin: 20px;
}
```

ボックスシャドウをう

```
.div2{
  border: 5px solid green;
  box-shadow: 0px 0px 0px 4px #000;
  width: 100px;
  height: 100px;
  margin: 20px;
}
```

をう

```
.div3 {
  position: relative;
  border: 5px solid #000;
  width: 100px;
  height: 100px;
  margin: 20px;
}
.div3:before {
  content: " ";
  position: absolute;
  border: 5px solid blue;
  z-index: -1;
  top: 5px;
  left: 5px;
  right: 5px;
  bottom: 5px;
}
```



http://jsfiddle.net/MadalinaTn/bvqpcohm/2/

border-imageをしてのをする

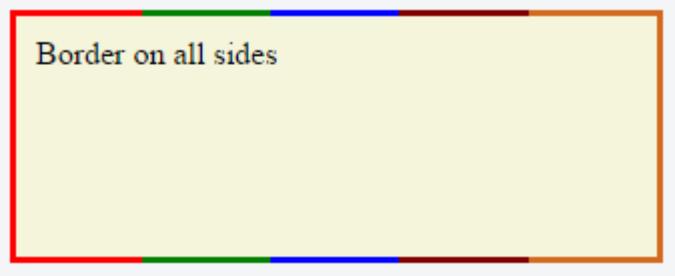
CSS

```
.bordered {
  border-image: linear-gradient(to right, red 20%, green 20%, green 40%, blue 40%, blue 60%,
maroon 60%, maroon 80%, chocolate 80%); /* gradient with required colors */
  border-image-slice: 1;
}
```

HTML

```
<div class='bordered'>Border on all sides</div>
```

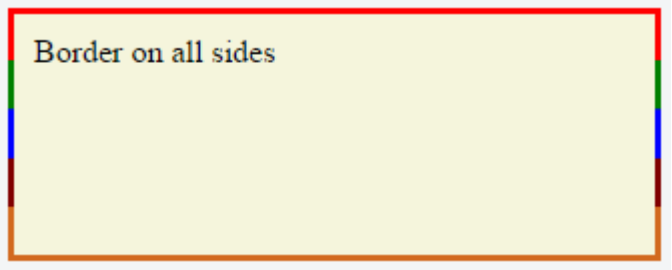
のでは、5つのなるからなるがされます。は`linear-gradient`されていますグラデーションにするはドキュメントにあります。 `border-image-slice`プロパティのについては、じページの`border-image`をしてください。



プレゼンテーションのでのプロパティがにされました。

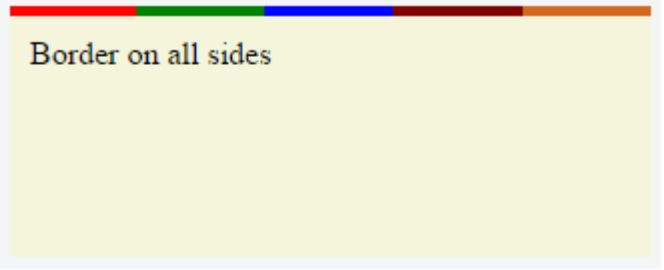
ボーダーにはのグラデーションのしかないのにし、ボーダーにはのグラデーションのわりのしかないことにづいたでしょう。これは、のイメージプロパティがするためです。まるでグラデーションがボックスにされ、パディングエリアとコンテンツエリアからがマスクされているようにえるので、だけがグラデーションをつようにえます。

どのがのをするかは、のにする。グラデーションが`to right`グラデーションの、のはグラデーションのになり、のはのになります。 `to bottom`グラデーションの、のボーダーはグラデーションのカラー、のボーダーはわりのカラーになります。は`to bottom` 5のカラーグラデーションのです。



がののにのみな、border `border-width`プロパティはののとにできます。たとえば、のコードをすると、のにのみがされます。

```
border-width: 5px 0px 0px 0px;
```



`border-image`プロパティをつは、 `border-radius` しないことにしてくださいボ―ダ―はカ―ブしません。これは、ののステ―トメントにづいています。

ボックスのはではなく、なにクリップされます「クリップ」によってされます。

オンラインでをむ https://riptutorial.com/ja/css/topic/2160/

45:

き

クラスはクラスとじようにCSSセレクタにされますが、なをするのではなく、htmlののをスコープしてスタイルすることができます。

たとえば、:: first-letterは、セレクタでされたブロックののののみをとします。

- selector :: pseudo-element {プロパティ}

パラメーター

`::after`	ののにをする
`::before`	ののにをする
`::first-letter`	ののをします。
`::first-line`	ののをします。
`::selection`	ユーザーがしたのとします。
`::backdrop`	レイヤーのスタックののとなるドキュメントをすをするためにされます
`::placeholder`	フォームのプレースホルダテキストをスタイルすることができます
`::marker`	のにリストスタイルのをする
`::spelling-error`	ブラウザがってられたテキストセグメントをします。
`::grammar-error`	ブラウザがにしくないとしたテキストセグメントをします。

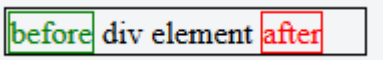
- にはあなたは、のコロンされます`::`のわりに、1つだけ`:`。これは、からクラスをするですが、Internet Explorer 8のようなのいブラウザでは、のコロンをサポートしています`:`のため。
- セレクタでは1つのしかできません。ステートメントのなセレクタのにするがあります。
- はDOMのではないため、`:hover`やそののユーザーイベントによってターゲティングすることはできません。

Examples

はセレクタにされますが、なをするのではなく、ドキュメントののをスタイルすることができます。

`content`は、のレンダリングにです。ただし、はのをつことができます `content: ""`。

```
div::after {
  content: 'after';
  color: red;
  border: 1px solid red;
}

div {
  color: black;
  border: 1px solid black;
  padding: 1px;
}

div::before {
  content: 'before';
  color: green;
  border: 1px solid green;
}
```



リストの

は、リストのをするためによくされますほとんどの、けられていないリスト、 `ul`。

のステップは、デフォルトリストのきをすることです

```
ul {
  list-style-type: none;
}
```

に、カスタムスタイリングをします。このでは、のグラデーションボックスをします。

```
li:before {
  content: "";
  display: inline-block;
  margin-right: 10px;
  height: 10px;
  width: 10px;
  background: linear-gradient(red, blue);
}
```

HTML

```
<ul>
   <li>Test I</li>
   <li>Test II</li>
</ul>
```

オンラインでをむ https://riptutorial.com/ja/css/topic/911/

46:

- ページり|いつも| |をける|||の|する;
- ページりauto |いつも| |をける|||の|する;
- ページりauto | |をける|の|する;

パラ メ—タ—

オ—ト	デフォルト。ペ—ジ
に	にペ—ジをする
ける	ペ—ジりをけるな
	ペ—ジをして、のペ—ジがペ—ジとしてフォ—マットされるようにする
	ペ—ジをして、のペ—ジがしいペ—ジとしてフォ—マットされるようにする
	このプロパティをデフォルトにします。
する	からこのプロパティをします。

CSSにペ—ジりのプロパティはありません。 3つのプロパティ **page-break-before** 、 **page-break-after** 、 **page-break-inside** のみ。

する `orphans` 、 `widows` 。

Examples

メディアペ—ジり

```
@media print {
  p {
    page-break-inside: avoid;
  }
  h1 {
    page-break-before: always;
  }
  h2 {
    page-break-after: avoid;
  }
}
```

このコ—ドは3つのことをいます

-

であれば、が2つのページにされることはありません。
- すべてのh1しにPage-Break-Beforeをします。つまり、h1がするにページがわれます。
- のh2のにページりをします

オンラインでをむ https://riptutorial.com/ja/css/topic/4316/

47: なボックスレイアウトFlexbox

き

Flexible Boxモジュールは、ユーザーインターフェイスにされたボックスモデルで、コンテナのアイテムのスペースをしてすることで、ページレイアウトがなるのサイズ。フレックスコンテナは、をしてなスペースをめるようにし、オーバーフローをぐためにします。

- ディスプレイフレックス;
- フレックス| - ||;
- フレックスラップnowrap |ラップ|りし;
- フレックスフロー<'フレックス'> || <'flex-wrap'>
- justify-contentフレックススタート|フレックスエンド|センター| |スペース·アラウンド;
- align-itemsフレックススタート|フレックスエンド|センター| |ストレッチ;
- align-contentフレックススタート|フレックスエンド|センター| |スペース - |ストレッチ;
- <>。
- flex-grow<>; / *デフォルト0 * /
- フレックスシュリンク<number>; / *デフォルト1 * /
- フレックスベース<length> |; / *デフォルトのauto * /
- フレックスなし| [<'flex-grow'> <'flex-shrink'> || <'flex-basis'>]
- align-selfauto |フレックススタート|フレックスエンド|センター| |ストレッチ;

ベンダーの

- -webkit-box; / * Chrome <20 * /
- -webkit-flex; / * Chrome 20+ * /
- -moz-box; / * Firefox * /
- ディスプレイ-ms-flexbox; / * IE * /
- ディスプレイフレックス; / *のブラウザ* /

リソース

- Flexboxのなガイド
- Flexboxによってされる
- Flexboxって
- 5でFlexbox
- フレックスバグ

Examples

スティッキーフッター

このコードは、フッターをします。コンテンツがビューポートのにしていない、フッターはビューポートのにりけられます。コンテンツがビューポートのをすると、フッターもビューポートからしされます。 をる

HTML

```
<div class="header">
  <h2>Header</h2>
</div>

<div class="content">
  <h1>Content</h1>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent
libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.
Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa.
Vestibulum lacinia arcu eget nulla. Class aptent taciti sociosqu ad litora torquent per
conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. </p>
</div>

<div class="footer">
  <h4>Footer</h4>
</div>
```

CSS

```
html, body {
  height: 100%;
}

body {
  display: flex;
  flex-direction: column;
}

.content {
  /* Include `0 auto` for best browser compatibility. */
  flex: 1 0 auto;
}

.header, .footer {
  background-color: grey;
  color: white;
  flex: none;
}
```

Flexboxをったレイアウト

レイアウトは、さのヘッダーとフッター、および3のセンターをつレイアウトです。 3つのには、のsidenav、の、などののコンテンツのがまれますのがマークアップのにされます。になマークアップでこれをするためにCSS Flexboxをうことができます

HTMLマークアップ

```
<div class="container">
  <header class="header">Header</header>
  <div class="content-body">
    <main class="content">Content</main>
    <nav class="sidenav">Nav</nav>
    <aside class="ads">Ads</aside>
  </div>
  <footer class="footer">Footer</footer>
</div>
```

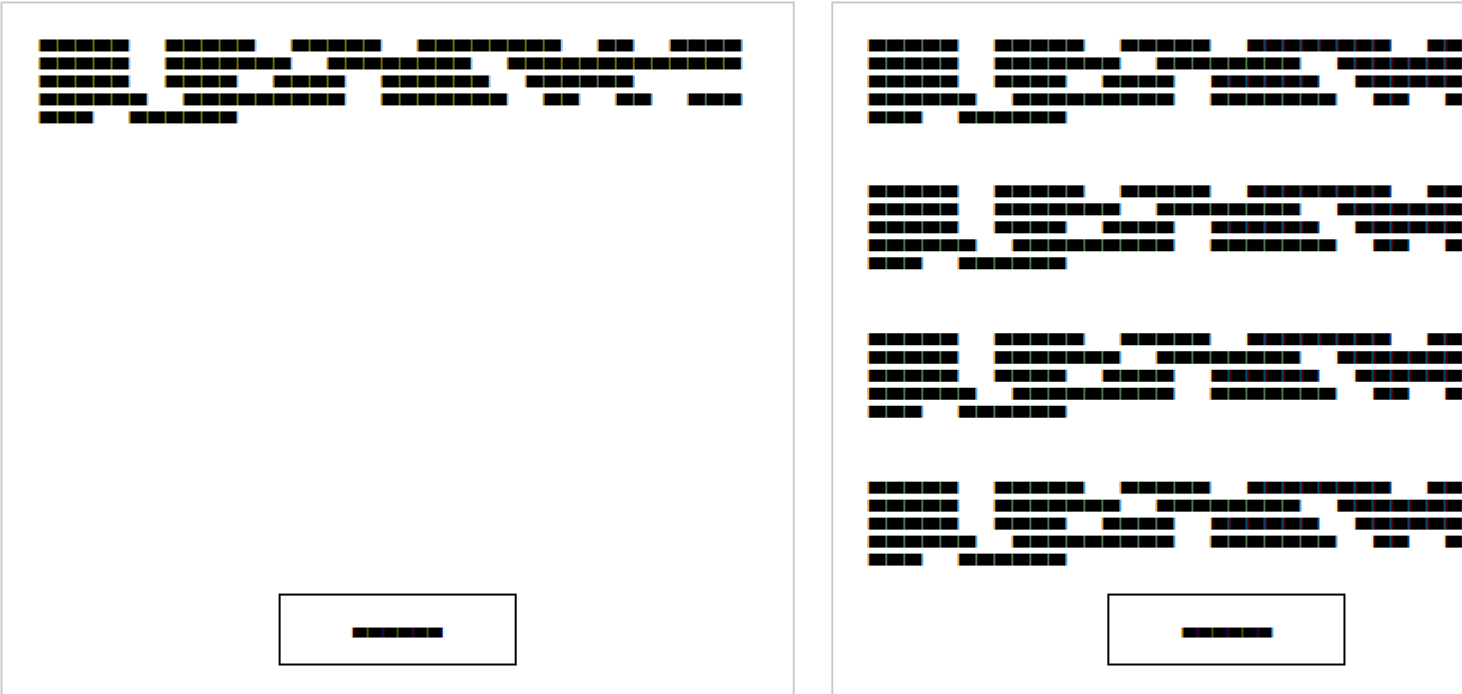
CSS

```
body {
  margin: 0;
  padding: 0;
}

.container {
  display: flex;
  flex-direction: column;
  height: 100vh;
}

.header {
  flex: 0 0 50px;
}

.content-body {
  flex: 1 1 auto;

  display: flex;
  flex-direction: row;
}

.content-body .content {
  flex: 1 1 auto;
  overflow: auto;
}

.content-body .sidenav {
  order: -1;
  flex: 0 0 100px;
  overflow: auto;
}

.content-body .ads {
  flex: 0 0 100px;
  overflow: auto;
}

.footer {
  flex: 0 0 50px;
}
```

デモ

Flexboxをってカ―ドのボタンをにわせ

これは、このようなカードののアクションのびしをにさせるために、のデザインではなパターンです



これは、 `flexbox`なトリックをしてすることができます

HTML

```
<div class="cards">
  <div class="card">
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p><button>Action</button></p>
  </div>
  <div class="card">
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p><button>Action</button></p>
  </div>
</div>
```

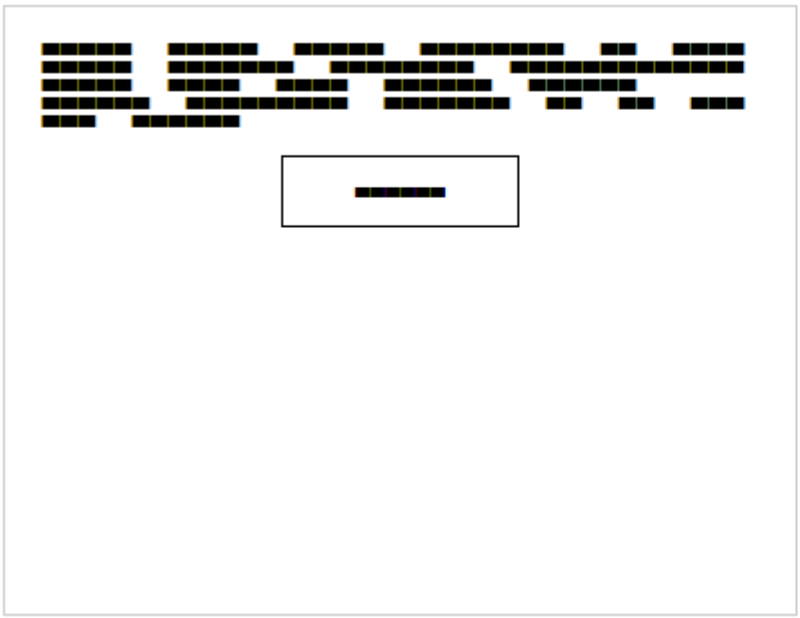
まずに、CSSをって`display: flex;`をし`display: flex;`にす。これにより、ににれるコンテンツとじさの2つのがされます

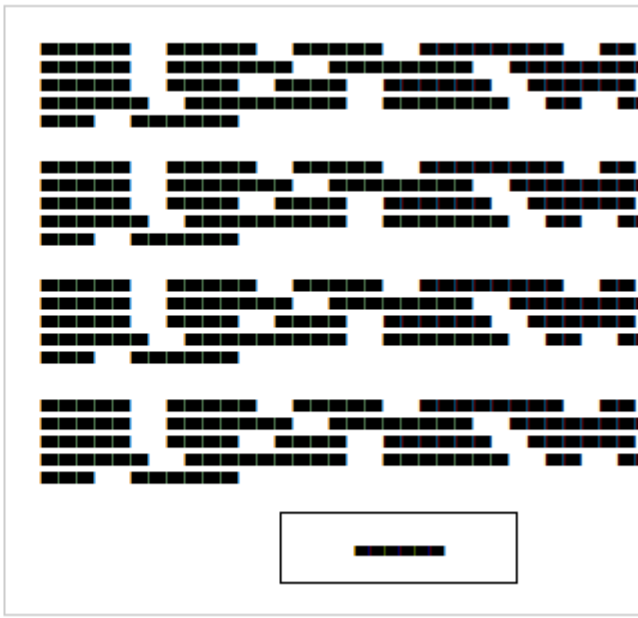
CSS

```
.cards {
```

```
    display: flex;
}
.card {
    border: 1px solid #ccc;
    margin: 10px 10px;
    padding: 0 20px;
}
button {
    height: 40px;
    background: #fff;
    padding: 0 40px;
    border: 1px solid #000;
}
p:last-child {
    text-align: center;
}
```

レイアウトがわり、のようになります





ボタンをブロックのにするには、 `display: flex;`をするがあります`display: flex;`を「 `column`してカーードにりけ`column`。 その、カーードののをし、 `margin-top`を`auto`するがあります。のをカーードのまでして、なをます。

なCSS

```
.cards {
    display: flex;
}
.card {
    border: 1px solid #ccc;
    margin: 10px 10px;
    padding: 0 20px;
    display: flex;
    flex-direction: column;
}
button {
```

```
    height: 40px;
    background: #fff;
    padding: 0 40px;
    border: 1px solid #000;
}
p:last-child {
    text-align: center;
    margin-top: auto;
}
```

なとのセンタリング**align-items**、justify-content

なをにする

HTML

```
<div class="aligner">
  <div class="aligner-item">…</div>
</div>
```

CSS

```
.aligner {
  display: flex;
  align-items: center;
  justify-content: center;
}

.aligner-item {
  max-width: 50%; /*for demo. Use actual width instead.*/
}
```

ここにデモがあります。

プロパティ		
`align-items`	`center`	これは、 `flex-direction`でされたもののにってをにきます。つまり、フレックスボックスのセンタリングと`flex-direction`ボックスのセンタリングです。
`justify-content`	`center`	これにより、 `flex-direction`されたにってが`flex-direction`されます。すなわち、 `flex-direction: row` の`flex-direction: row`ボックスの、これはにし、`flex-direction: column`ボックス `flex-direction: column` `flex-direction: column`ボックスの、これはにします

々のプロパティの

のスタイルはすべてこのなレイアウトにされます

```
<div id="container">
  <div></div>
  <div></div>
  <div></div>
</div>
```

`#container`は`flex-box`です。

`justify-content: center`フレックスボックスの**`justify-content: center`**

CSS

```
div#container {
  display: flex;
  flex-direction: row;
  justify-content: center;
}
```

ここにデモがあります。

`justify-content: center`フレックスボックスの`justify-content: center`

CSS

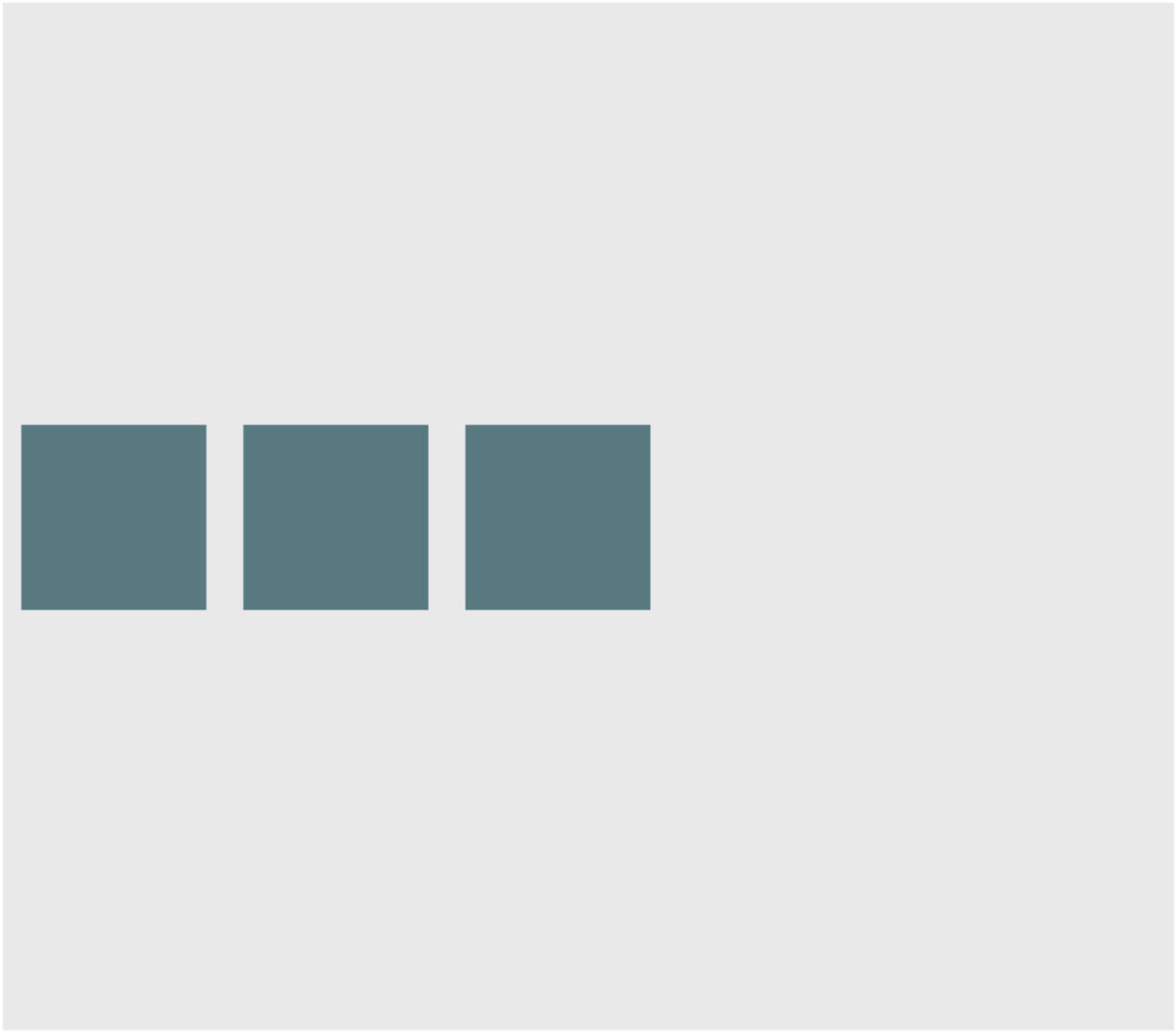
```
div#container {
  display: flex;
  flex-direction: column;
  justify-content: center;
}
```

ここにデモがあります。

`align-content: center`フレックスボックスの`align-content: center`

CSS

```
div#container {
  display: flex;
  flex-direction: row;
  align-items: center;
}
```



ここにデモがあります。

`align-content: center`フレックスボックスの`align-content: center`

CSS

```
div#container {
  display: flex;
  flex-direction: column;
  align-items: center;
}
```

ここにデモがあります。

フレックスボックスのをセンタリングするためのみわせ

```
div#container {
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
}
```

ここにデモがあります。

フレックスボックスのをセンタリングするためのみわせ

```
div#container {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
}
```

ここにデモがあります。

ネストされたコンテナのさ

このコードは、すべてのネストされたコンテナがにじさになるようにします。これは、すべてのネストされたがまれているparrent divとじさであることをすることによってわれます。 のをしてください https : //jsfiddle.net/3wwh7ewp/

このは、プロパティの`align-items`がデフォルトで`stretch`するようにされているためにされます。

HTML

```
<div class="container">
  <div style="background-color: red">
    Some <br />
    data <br />
    to make<br />
    a height <br />
 </div>
  <div style="background-color: blue">
```

```
    Fewer <br />
    lines <br />
 </div>
</div>
```

CSS

```
.container {
    display: flex;
    align-items: stretch; // Default value
}
```

10のIEバージョンではしません

をコンテナににフィット

flexboxのれたの1つは、コンテナをににさせることです。

ライブデモ

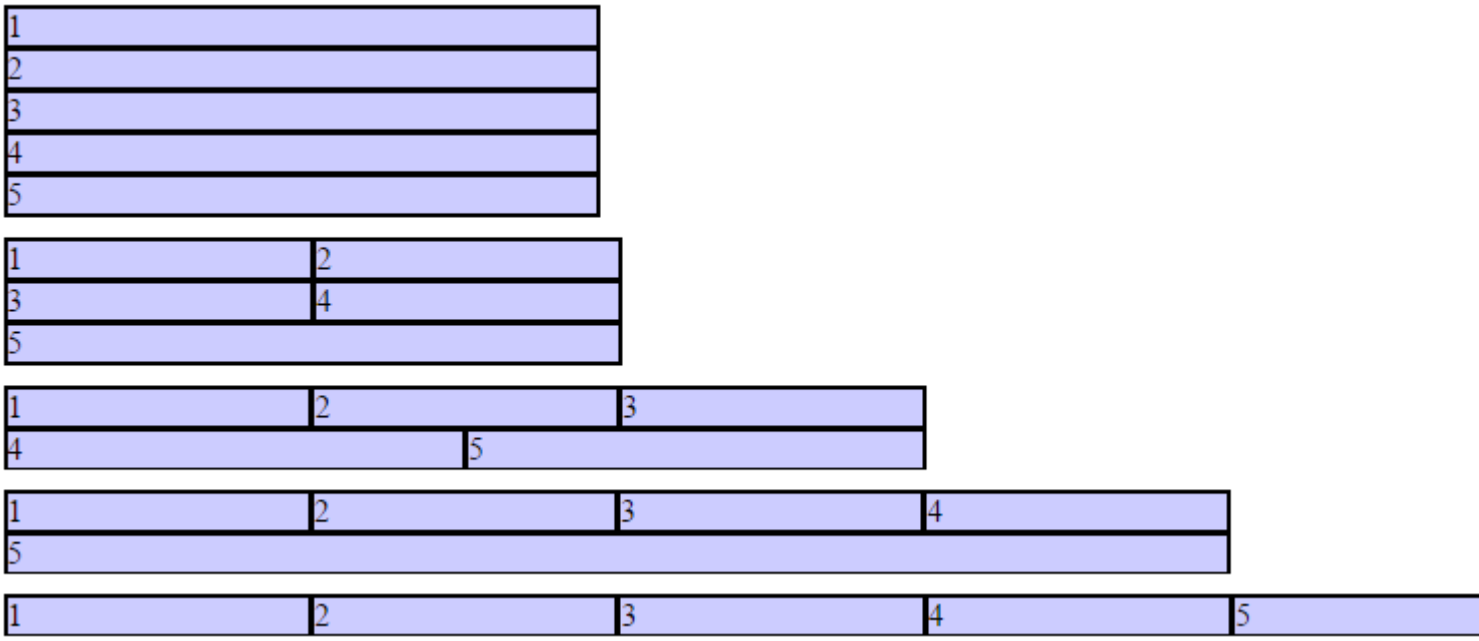
HTML

```
<div class="flex-container">
  <div class="flex-item">1</div>
  <div class="flex-item">2</div>
  <div class="flex-item">3</div>
  <div class="flex-item">4</div>
  <div class="flex-item">5</div>
</div>
```

CSS

```
.flex-container {
    background-color: #000;
    height: 100%;
    display:flex;
    flex-direction: row;
    flex-wrap: wrap;
    justify-content: flex-start;
    align-content: stretch;
    align-items: stretch;
}

.flex-item {
    background-color: #ccf;
    margin: 0.1em;
    flex-grow: 1;
    flex-shrink: 0;
    flex-basis: 200px; /* or % could be used to ensure a specific layout */
}
```

がリサイズされると、がします。



オンラインでなボックスレイアウトFlexboxをむ https://riptutorial.com/ja/css/topic/445/なボックスレイアウト-flexbox-

48: きの

- - なし| [<> || <shape-box>] | <image>
- shape-margin<さ> | <パーセンテージ>
- shape-image-threshold<number>

パラメーター

パラメータ	
し	noneのは、floatfloatのりにコンテンツをラップするためにされるがをけnoneことをします。これがデフォルト/です。
	inset()、 circle()、 ellipse()またはpolygon()いずれかをpolygon()ます。これらのとそののいずれかをして、がされます。
シェイプボックス	margin-box、 border-box、 padding-box、 content-box。 <shape-box>のみがされている<basic-shape>なし、このボックスはシェイプです。 <basic-shape>とともにすると、これはボックスとしてします。
	がとしてされると、されたのアルファチャンネルにづいてがされます。

このでCSSシェイプモジュールのブラウザサポートはにされています。

ブラウザ/ベンダープレフィックスなしでChrome v37 +およびOpera 24+でサポートされています。 Safariは、v7.1 +からそれをサポートしていますが、 -webkit-を-webkit-ます。

IE、Edge、Firefoxではまだサポートされていません。

Examples

のにある -

shape-outside CSSプロパティでは、のわりにシェイプをむように、エリアのシェイプをできます。

CSS

```
img:nth-of-type(1) {
  shape-outside: circle(80px at 50% 50%);
  float: left;
```

```
  width: 200px;
}
img:nth-of-type(2) {
  shape-outside: circle(80px at 50% 50%);
  float: right;
  width: 200px;
}
p {
  text-align: center;
  line-height: 30px; /* purely for demo */
}
```

HTML

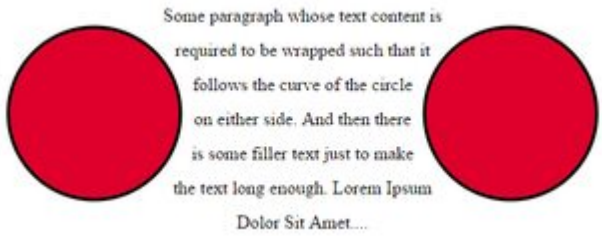
```
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<p>Some paragraph whose text content is required to be wrapped such that it follows the curve
of the circle on either side. And then there is some filler text just to make the text long
enough. Lorem Ipsum Dolor Sit Amet....</p>
```

のでは、はどちらもにはのであり、テキストが`shape-outside`プロパティなしでされると、どちらのものりをれません。のっているボックスのりをれます。 `shape-outside` 、はとしてされ、は`shape-outside`をしてされたこののりをれるようになります。

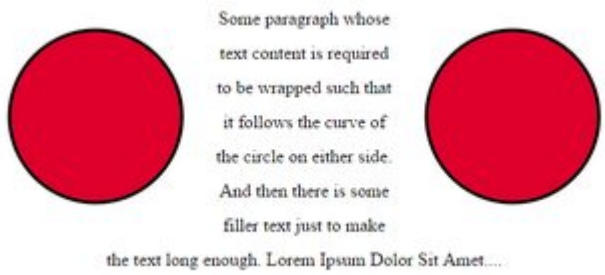
をするためにされるのは、のボックスの - からかれた80pxのです。

は、 `shape-outside`がされているときとされていないときに、コンテンツがどのようにラップされるかをすためのスクリーンショットです。

`shape-outside`



`shape-outside`なしの



マージン

`shape-margin` CSSプロパティは、 `shape-outside` マージンをします。

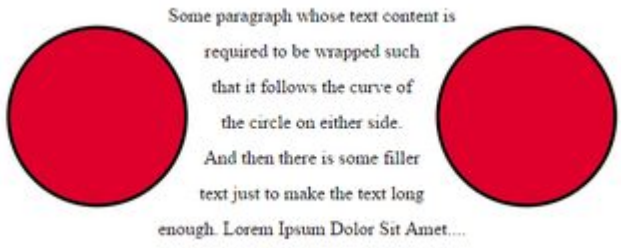
CSS

```
img:nth-of-type(1) {
  shape-outside: circle(80px at 50% 50%);
  shape-margin: 10px;
  float: left;
  width: 200px;
}
img:nth-of-type(2) {
  shape-outside: circle(80px at 50% 50%);
  shape-margin: 10px;
  float: right;
  width: 200px;
}
p {
  text-align: center;
  line-height: 30px; /* purely for demo */
}
```

HTML

```
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<p>Some paragraph whose text content is required to be wrapped such that it follows the curve
of the circle on either side. And then there is some filler text just to make the text long
enough. Lorem Ipsum Dolor Sit Amet....</p>
```

このでは、シェイプ `shape-margin`をして`shape-margin`に10ピクセルのがされています。これにより、をするのと、れっているのコンテンツとのに、よりくのスペースができます。



オンラインできのをむ https://riptutorial.com/ja/css/topic/2034/きの

49:

- クリアなし||||インラインスタート|インライン-エンド;
- ||なし|インラインスタート|インライン-エンド;

floatはブロックレイアウトのをするため、のをするもあります[1]

[1] https : //developer.mozilla.org/en-US/docs/Web/CSS/float MDN

Examples

をテキストにかべる

のもないは、のりにテキストラップをつことです。のコードは、2つのとイメージをし、2つのはイメージのりをれます。 floatのりをれるfloatのには、にがあることにしてください。

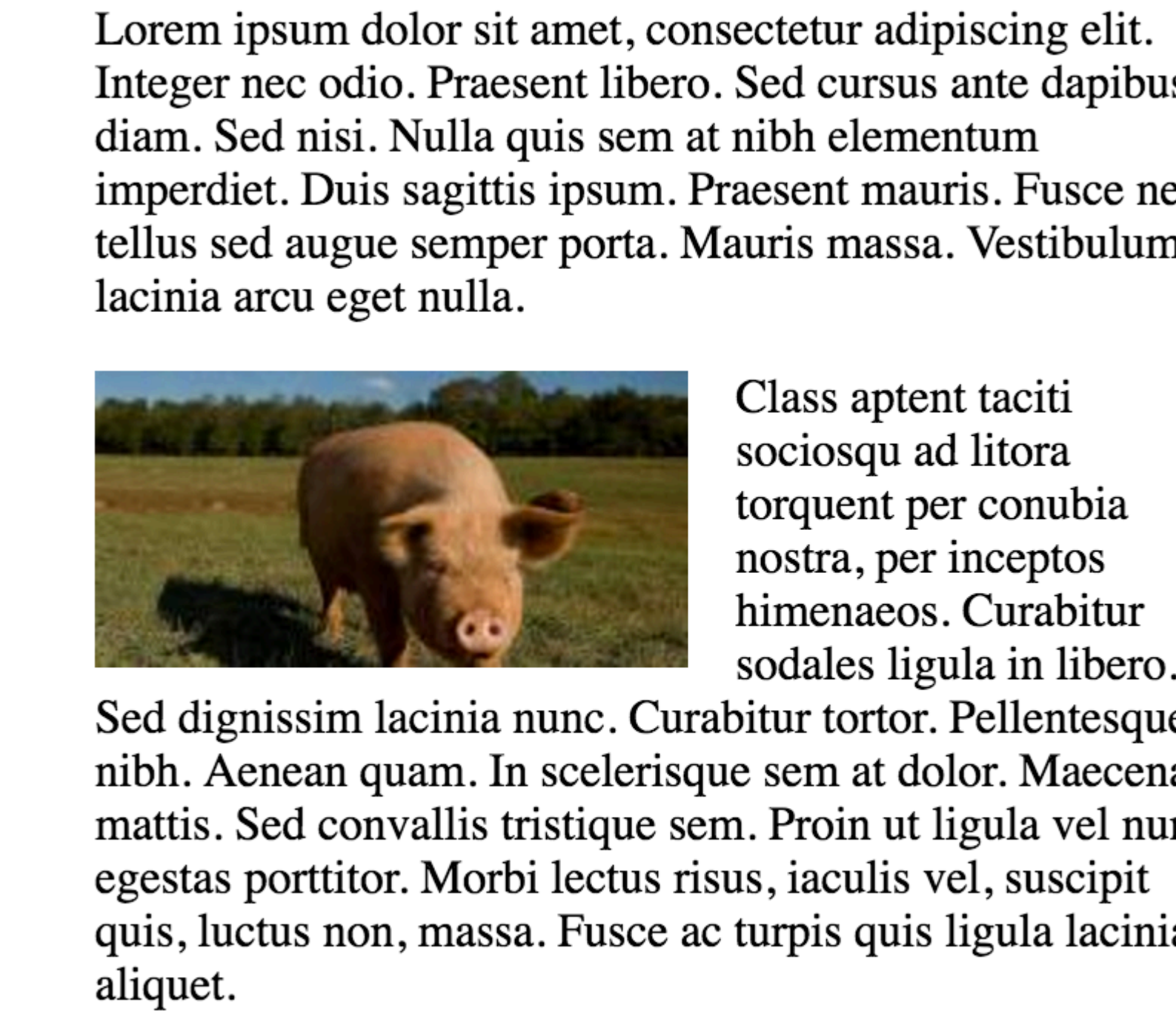
HTML

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet. Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa. Vestibulum lacinia arcu eget nulla. </p>

<img src="http://lorempixel.com/200/100/" />

<p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. </p>
```

CSS

```
img {
  float:left;
  margin-right:1rem;
}
```

これがになります



Codepen Link

な2つののレイアウト

な2のレイアウトは、2つののでされています。このでは、サイドバーとコンテンツのさがじではありません。これは、をするのレイアウトのしいの1つであり、のをじさにせるためのがです。

HTML

```
<div class="wrapper">
```

```
<div class="sidebar">
  <h2>Sidebar</h2>

  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio.</p>
</div>

<div class="content">
  <h1>Content</h1>

  <p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos
himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor.
Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis
tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel,
suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. </p>
</div>

</div>
```

CSS

```
.wrapper {
  width:600px;
  padding:20px;
  background-color:pink;

  /* Floated elements don't use any height. Adding "overflow:hidden;" forces the
     parent element to expand to contain its floated children. */
  overflow:hidden;
}

.sidebar {
  width:150px;
  float:left;
  background-color:blue;
}

.content {
  width:450px;
  float:right;
  background-color:yellow;
}
```

な3つののレ イアウト

HTML

```
<div class="wrapper">
  <div class="left-sidebar">
    <h1>Left Sidebar</h1>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. </p>
  </div>
  <div class="content">
    <h1>Content</h1>
    <p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos
himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor.
Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis
tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel,
```

```
suscipit quis, luctus non, massa. </p>
  </div>
  <div class="right-sidebar">
    <h1>Right Sidebar</h1>
    <p>Fusce ac turpis quis ligula lacinia aliquet.</p>
  </div>
</div>
```

CSS

```
.wrapper {
  width:600px;
  background-color:pink;
  padding:20px;

  /* Floated elements don't use any height. Adding "overflow:hidden;" forces the
     parent element to expand to contain its floated children. */
  overflow:hidden;
}

.left-sidebar {
  width:150px;
  background-color:blue;
  float:left;
}

.content {
  width:300px;
  background-color:yellow;
  float:left;
}

.right-sidebar {
  width:150px;
  background-color:green;
  float:right;
}
```

2レ イジ―/グリ―ディ―レイ アウト

このレイアウトでは、1つのをして、がされていない2のレイアウトをします。このでは、のサイドバ―はなだけのスペ―スをするというで「け」です。これをうのは、のサイドバ―が「シュリンクラップ」されていることです。しいのは、りのすべてのをめるというで「」です。

HTML

```
<div class="sidebar">
<h1>Sidebar</h1>
<img src="http://lorempixel.com/150/200/" />
</div>

<div class="content">
<h1>Content</h1>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero.
Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet. Duis
sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa.
```

```
Vestibulum lacinia arcu eget nulla. </p>
<p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos.
Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque
nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem.
Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis,
luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. Mauris ipsum. Nulla metus
metus, ullamcorper vel, tincidunt sed, euismod in, nibh. </p>
</div>
```

CSS

```
.sidebar {
  /* `display:table;` shrink-wraps the column */
  display:table;
  float:left;
  background-color:blue;
}

.content {
  /* `overflow:hidden;` prevents `.content` from flowing under `.sidebar` */
  overflow:hidden;
  background-color:yellow;
}
```

フィドル

な

clearプロパティはにしています。プロパティ

- none - デフォルト。のフロ―ティングをする
- left - にはできません
- - はにありません
- - またはのいずれのにもはできません
- initial - このプロパティをデフォルトにします。イニシャルについてむ
- inherit - このプロパティをからします。についてむ

```
<html>
<head>
<style>
img {
    float: left;
}

p.clear {
    clear: both;
}
</style>
</head>
<body>

<img src="https://static.pexels.com/photos/69372/pexels-photo-69372-medium.jpeg" width="100">
<p>Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum
Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum </p>
```

```
<p class="clear">Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum
Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum </p>

</body>
</html>
```

クリアフィックス

> clearfixハックは、をめるなですN. Gallagher aka @necolas

`clear`プロパティとしないように、clearfixはコンセプトです これはにもするため、のがあります。 をめるには、コンテナ に`.cf`または`.clearfix`クラスをし、このクラスのスタイルをいくつかのルールでします。

わずかになるをつ3つのバージョンソース しいマイクロclearfixハック N.ギャラガーによってリロードclearfix TJ Koblentzによって。

Clearfixまれているのがっている

```
.cf:after {
    content: "";
    display: table;
}

.cf:after {
    clear: both;
}
```

まれているフロートのマージンのをぐClearfix

```
/**
 * For modern browsers
 * 1. The space content is one way to avoid an Opera bug when the
 *    contenteditable attribute is included anywhere else in the document.
 *    Otherwise it causes space to appear at the top and bottom of elements
 *    that are clearfixed.
 * 2. The use of `table` rather than `block` is only necessary if using
 *    `:before` to contain the top-margins of child elements.
 */
.cf:before,
.cf:after {
    content: " "; /* 1 */
    display: table; /* 2 */
}

.cf:after {
    clear: both;
}
```

いブラウザIE6とIE7のサポートによるClearfix

```
.cf:before,
.cf:after {
    content: " ";
    display: table;
}

.cf:after {
    clear: both;
}

/**
 * For IE 6/7 only
 * Include this rule to trigger hasLayout and contain floats.
 */
.cf {
    *zoom: 1;
}
```

---

clearfixをすCodepen

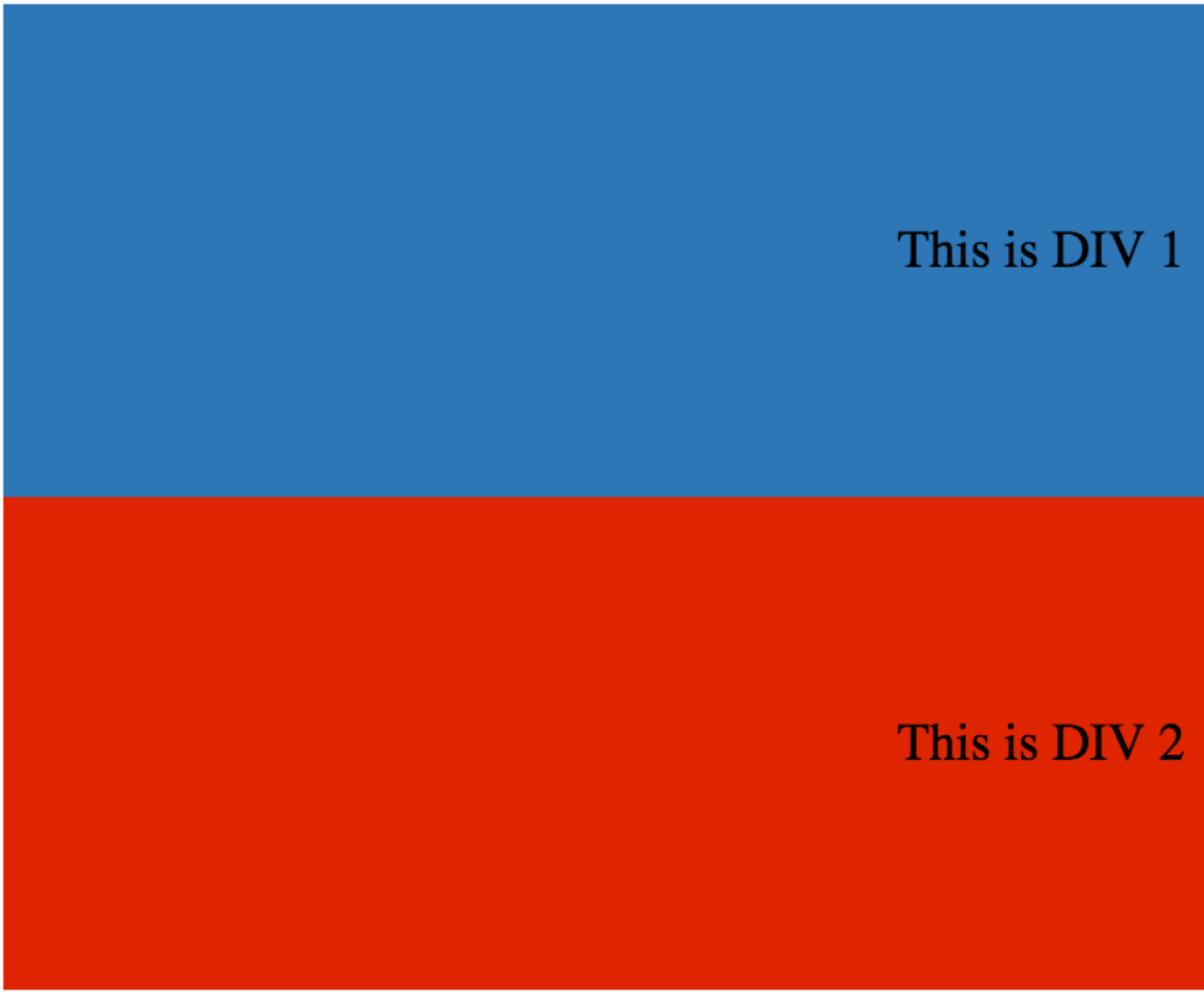
そののリソース clearfixについてっていることはすべてっています clearfixとBFC - Block Formatting Context、hasLayoutはくなったブラウザIE6にするかもしれません

フロートをしたインラインDIV

divはブロックレベルのです。つまり、ページのをめ、はになくにされます。

```
<div>
    <p>This is DIV 1</p>
</div>
<div>
    <p>This is DIV 2</p>
</div>
```

のコードのはのようになります。



`div` `float` cssプロパティをすることで、それらをインラインでることができます。

HTML

```
<div class="outer-div">
    <div class="inner-div1">
        <p>This is DIV 1</p>
    </div>
    <div class="inner-div2">
        <p>This is DIV 2</p>
    </div>
</div>
```

CSS

```
.inner-div1 {
    width: 50%;
    margin-right:0px;
    float:left;
```

```
    background : #337ab7;
    padding:50px 0px;
}

.inner-div2 {
    width: 50%;
    margin-right:0px;
    float:left;
    background : #dd2c00;
    padding:50px 0px;
}

p {
    text-align:center;
}
```

Codepen Link

フロートをクリアするためのオーバーフロープロパティの

`overflow`を`hidden`、 `auto`または`scroll`するようにすると、そののすべてのがクリアされます。

`overflow:scroll`をするとに`overflow:scroll`ボックスがされます

オンラインでをむ https://riptutorial.com/ja/css/topic/405/

50:

- プロパティ。

Examples

のいくつかのプロパティのされたがそのにされるCSSのなメカニズムです。これは、マ―クアップのすべてのにのプロパティをするのではなく、グロ―バルスタイルをにするににです。

にされるなプロパティは、 `font` 、 `color` 、 `text-align` 、 `line-height` です。

のスタイルシ―トをします。

```
#myContainer {
  color: red;
  padding: 5px;
}
```

これは`<div>`だけでなく、 `<h3>` `<p>`と`<p>`にも`color: red`をします。ただし、 `padding`の、そのはそのにされません 。

```
<div id="myContainer">
  <h3>Some header</h3>
  <p>Some paragraph</p>
</div>
```

な

のプロパティは、からそのににされません。これは、これらのプロパティが、プロパティがされるまたはのにしてであることがましいためです。なこのようなプロパティは、 `margin` 、 `padding` 、 `background` 、 `display`などです。

しかし、とにかくがまれることもあります。これをするために、 `inherit`があるプロパティに`inherit`をできます。 `inherit`は、 の CSSプロパティとの HTMLにできます。

のスタイルシ―トをします。

```
#myContainer {
  color: red;
  padding: 5px;
}
#myContainer p {
  padding: inherit;
}
```

これは、 `color`プロパティののために、 `<h3>` `<p>`と`<p>`のに`color: red`をします。ただし、 `<p>`はされているため、そのからの`padding`もします。

```
<div id="myContainer">
  <h3>Some header</h3>
  <p>Some paragraph</p>
</div>
```

オンラインでをむ https://riptutorial.com/ja/css/topic/3586/

51:

き

CSSをすると、、グラデーション、およびをのとしてできます。

、、のさまざまなみわせをし、これらのサイズ、、りしなどをすることができます。

- |な|の|する;
- URL |なし|の|する;
- バックグラウンド;
- サイズ<bg-size> [<bg-size>]
- <bg-size>auto |さ|カバー| |まれる|の|する;
- バックグラウンドリピートリピート|リピート-x |リピート-y |ノーリピート|の|する;
- background-originパディングボックス|ボーダーボックス|コンテンツボックス|の|する;
- - クリップborder-box |パディングボックス|コンテンツボックス|の|する;
- background-attachmentスクロール||ローカル|の|する;
- bg-color bg-image/ bg-size bg-repeat bg-origin bg-クリップbg-attachment initial |する;

- CSS3グラデーションは、Internet Explorerのバージョンが10ではしません。

Examples

`background-color`プロパティは、カラーを`inherit`か、または`transparent` 、 `inherit`または`initial`などのキーワードをして、のをします。

- **transparent**は、があることをします。これはデフォルトです。
- し 、このプロパティをからします。
- **initial**は 、このプロパティをデフォルトにします。

これはすべてのと`::first-letter` / `::first-line` にできます。

CSSのは、 さまざまなでできます。

の

CSS

```
div {
  background-color: red;  /* red */
}
```

HTML

```
<div>This will have a red background</div>
```

- のは、CSSがのをしなければならないいくつかのの1つです。

16のカラーコード

16コードは、16の16ののRGBコンポーネントをすためにされます。たとえば、ff0000はるいで、のは256ビットffで、するとのは000です。

3つのRGBペアR、G、Bののがじ、カラーコードは3ペアリングののにできます。 `#ff0000`にすることが`#f00`、び`#ffffff`にすることができる`#fff`。

16ではとをしません。

```
body {
  background-color: #de1205; /* red */
}

.main {
  background-color: #00f; /* blue */
}
```

RGB / RGBa

をするのは、RGBまたはRGBaをすることです。

RGBは、、ので、、、それぞれののののにする、ブラケットのにある0255の3つの々のがです。

RGBaでは、0.01.0のにアルファパラメータをしてをできます。

```
header {
  background-color: rgb(0, 0, 0); /* black */
}

footer {
  background-color: rgba(0, 0, 0, 0.5); /* black with 50% opacity */
}
```

HSL / HSLa

をするのは、HSLまたはHSLaをすることです。これはRGBおよびRGBaにています。

HSLは、、ので、HLSともばれます。

-

はカラーホイールのいです0360。
- は0100のです。
- も0から100ののパーセンテージです。

HSLaでは0.01.0のにアルファパラメータをしてをできます。

```
li a {
  background-color: hsl(120, 100%, 50%); /* green */
}

#p1 {
  background-color: hsla(120, 100%, 50%, .3); /* green with 30% opacity */
}
```

との

のステートメントはすべてです。

```
body {
  background: red;
  background-image: url(partiallytransparentimage.png);
}

body {
  background-color: red;
  background-image: url(partiallytransparentimage.png);
}

body {
  background-image: url(partiallytransparentimage.png);
  background-color: red;
}

body {
  background: red url(partiallytransparentimage.png);
}
```

それらはすべて、いがのにされ、のがであるか、がされないおそらく`background-repeat`として。

はではないことにしてください。

```
body {
  background-image: url(partiallytransparentimage.png);
  background: red;
}
```

ここでは、 `background`のが`background` `background-image`よりもされます。

`background`プロパティのについては、 `background` グラウンドのをしてください。

`background-image`プロパティは、するすべてのにされるイメージをするためにされます。デフォル

トでは、このはマージンをいてをカバーするようにべられています。

```
.myClass {
  background-image: url('/path/to/image.jpg');
}
```

のイメージを`background-image`イメージとしてするには、カンマりの`url()`

```
.myClass {
  background-image: url('/path/to/image.jpg'),
                    url('/path/to/image2.jpg');
}
```

は、にされたをののにねてにスタックします。

`url('/path/to/image.jpg')`	イメージのパスまたはデータURIスキーマでされたイメージリソースをするアポストロフィは、のコンマでる
`none`	なし
`initial`	デフォルト
`inherit`	のをきぐ

のためのよりくの**CSS**

これにくはにでほとんどです。

```
background-size:     xpx ypx | x% y%;
background-repeat:   no-repeat | repeat | repeat-x | repeat-y;
background-position: left offset (px/%) right offset (px/%) | center center | left top | right
bottom;
```

グラデーション

グラデーションはしいイメージタイプで、CSS3でされました。イメージとして、グラデーションは`background-image`プロパティ、または`background`でされます。

には、との2があります。タイプには、りしとりしがあります。

- `linear-gradient()`
- `repeating-linear-gradient()`
- `radial-gradient()`
- `repeating-radial-gradient()`

`linear-gradient`はのとおりです

```
background: linear-gradient( <direction>?, <color-stop-1>, <color-stop-2>, ...);
```

`<direction>`	`to top` 、 `to bottom` 、 `to right`または`to left`ようなかもしれません。または `0deg` 、 `90deg` ...のようなです。はからにかってりにします。 deg 、 grad 、 rad 、 turnでできます。された、はからにれる
`<color-stop-list>`	のリスト。にじてパーセントまたはさでそれぞれをしてします。たとえば、 `yellow 10%` 、 `rgba(0,0,0,.5) 40px` 、 `#fff 100%` ...

たとえば、これはからしてからへのをします

```
.linear-gradient {
  background: linear-gradient(to left, red, blue); /* you can also use 270deg */
}
```

およびののをすることによって、 `diagonal`をすることができます。

```
.diagonal-linear-gradient {
  background: linear-gradient(to left top, red, yellow 10%);
}
```

グラデーションにののカラーストップをカンマでってすることができます。のでは、8つのカラーストップでグラデーションをします

```
.linear-gradient-rainbow {
  background: linear-gradient(to left, red, orange, yellow, green, blue, indigo, violet)
}
```

```
.radial-gradient-simple {
  background: radial-gradient(red, blue);
}

.radial-gradient {
  background: radial-gradient(circle farthest-corner at top left, red, blue);
}
```

`circle`	グラデーションの。は`circle`または`ellipse` 、デフォルトは`ellipse`です。
`farthest-corner`	のきさをすキーワード。は、 `closest-side` 、 `farthest-side` 、 `closest-corner` 、 `farthest-corner`
`top left`	グラデーションのを、 `background-position`とじでし`background-position` 。

をりすことは、のとじをとりますが、のにをきます。

```
.bullseye {
  background: repeating-radial-gradient(red, red 10%, white 10%, white 20%);
}
.warning {
  background: repeating-linear-gradient(-45deg, yellow, yellow 10%, black 10%, black 20% );
}
```

`-45deg`	。はからにかってりにします。 deg 、 grad 、 rad 、 turnでできます。
`to left`	グラデ―ションの、デフォルトは`to bottom`です。 `to [y-axis(top OR bottom)] [x-axis(left OR right)]`すなわち`to top right`
`yellow 10%`	をし、オプションでパ―セントまたはさをけてします。 2りした。

カラ―のわりに、HEX、RGB、RGBa、HSL、およびHSLaのカラ―コ―ドをできます。のためにのをしました。また、のはよりはるかにであり、されたバ―ジョンがここにされていることにもしてください。なとについては、 MDN Docsをしてください。

`background`プロパティは、1つまたはのプロパティをするためにできます。

		CSS Ver。
`background-image`	するイメ―ジ	1+
`background-color`	する	1+
`background-position`	の	1+
`background-size`	のサイズ	3+
`background-repeat`	ィメ―ジをりす	1+
`background-origin`	がどのようにされているか `background-attachment`が`fixed`されているはさ `fixed`	3+
`background-clip`	`content-box` 、 `border-box` 、または`padding-box` `border-box`にバックグラウンドをりつぶす	3+
`background-attachment`	が、そのブロックとにスクロ―ルするか、またはビュ―ポ―トのにあるかにかかわらず、どのようにするか	1+
`initial`	プロパティをデフォルトにします。	3+

		CSS Ver。
`inherit`	からプロパティをする	2+

のはではなく、すべてのはオプションです

のはのとおりです。

```
background: [<background-image>] [<background-color>]  [<background-position>]/[<background-size>] [<background-repeat>] [<background-origin>] [<background-clip>] [<background-attachment>] [<initial|inherit>];
```

```
background: red;
```

に`red`で`background-color`をします。

```
background: border-box red;
```

`background-clip`をボーダーボックスにし、 `background-clip` `background-color`をにします。

```
background: no-repeat center url("somepng.jpg");
```

`background-repeat`からノーリピート、 `background-origin`をに、 `background-image`を`background-image`にします。

```
background: url('pattern.png') green;
```

このでは、の`background-color`は`green`にされ、なは`pattern.png`がにオーバーレイされ、をたすためになだけにりされます。 `pattern.png`にがまれていると、 `green`がそのにされます。

```
background: #000000 url("picture.png") top left / 600px auto no-repeat;
```

このでは、に 'picture.png'をむいがあり、はどちらのでもりされず、にされます。 `/`のには、このようにされたのサイズをむことができることである`600px`さのと。このは、にフェードするでうまくいくかもしれません。

> のバックグラウンドプロパティをすると、がされていなくても、にされたバックグラウンドプロパティがすべてリセットされます。にしたバックグラウンドのプロパティをしたいは、わりにlonghandプロパティをします。

の

`background-position`プロパティは、またはグラデーションのをするためにされます

```
.myClass {
  background-image: url('path/to/image.jpg');
  background-position: 50% 50%;
}
```

はXとYをしてされ、CSSでされているいずれかのをしてされます。

	オフセットのパーセンテージは、 ののの - のにするなです。 オフセットのパーセンテージは、 ののののさ - のさ イメージのサイズは、 `background-size`された`background-size`です。
px px	バックグラウンドを、バックグラウンドのをにしたピクセルのさだけオフセットします。

CSSのは、さまざまなでできます ここを。

ロング・バックグラウンドのプロパティ

ののプロパティにえて、ののプロパティ`background-position-x`と`background-position-y`することもできます。これにより、xまたはyのを々にすることができます。

> これは、Firefoxバージョン31-48 2をくすべてのブラウザでサポートされています。20149にリリースされるFirefox 49は、これらのプロパティをサポートします。それまでは、 Stack OverflowのえにFirefoxのハックがあります。

アタッチメント

background-attachmentプロパティは、イメージがされているか、ページのりのでスクロールされているかをします。

```
body {
  background-image: url('img.jpg');
  background-attachment: fixed;
}
```

スクロール	はとともにスクロールします。これはデフォルトです。
	はビューポートにしてされています。
	はのとともにスクロールします。
	このプロパティをデフォルトにします。

する	からこのプロパティをします。

アタッチメントスクロール

デフォルトのでは、ボディをスクロールするとがスクロールします。

```
body {
  background-image: url('image.jpg');
  background-attachment: scroll;
}
```

はされ、ボディがスクロールされるとしません。

```
body {
  background-image: url('image.jpg');
  background-attachment: fixed;
}
```

ローカル

divのがスクロールされると、divのイメージがスクロールします。

```
div {
  background-image: url('image.jpg');
  background-attachment: local;
}
```

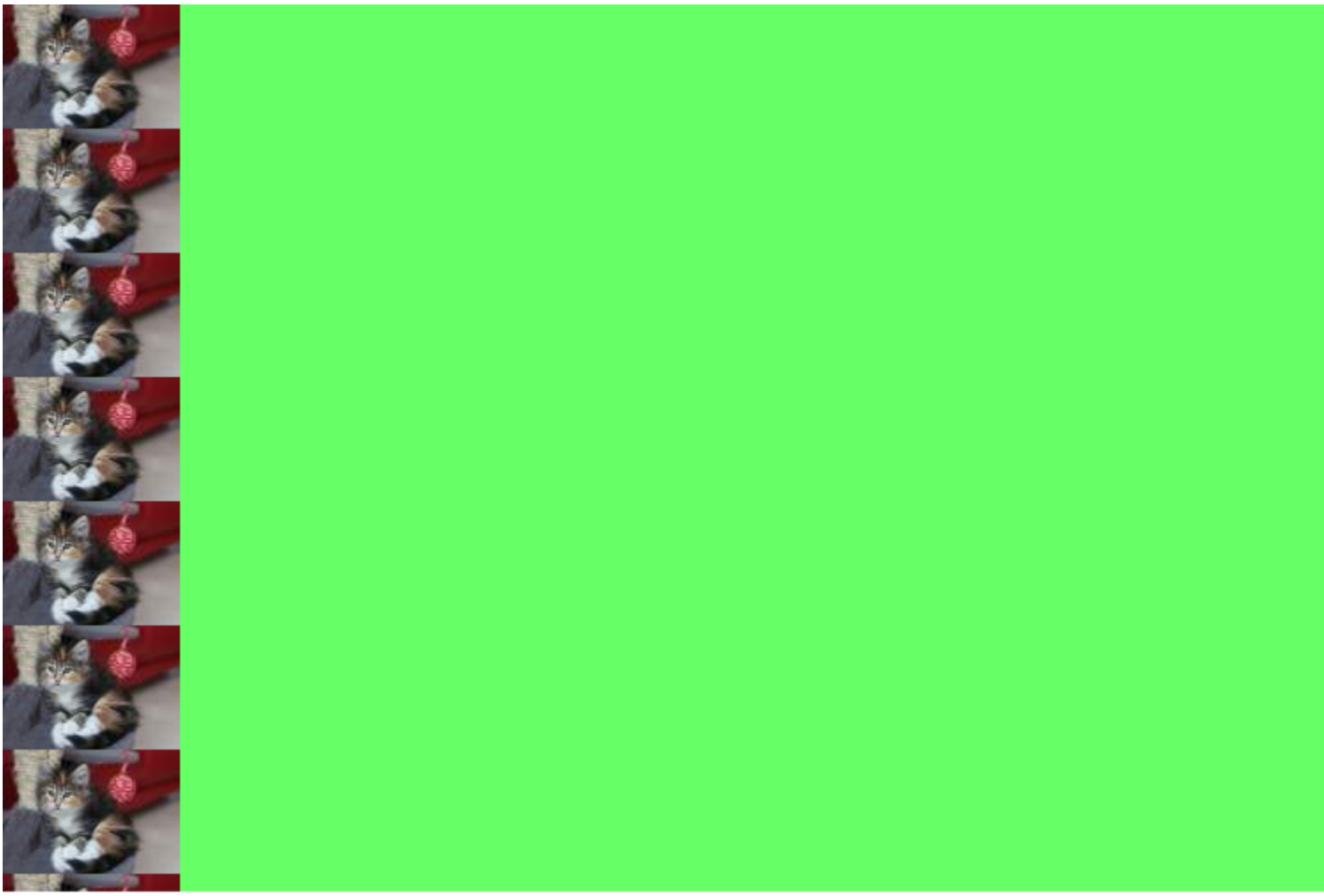
のりし

background-repeatプロパティは、イメージがりされるかどうかを/します。

デフォルトでは、はとのでりされます。

```
div {
  background-image: url("img.jpg");
  background-repeat: repeat-y;
}
```

`background-repeat: repeat-y`ようになります



のある

にopacityをすると、そのすべてにします。のだけにをするには、RGBAカラーをするがあります。のでは、0.6のをついがされます。

```
/* Fallback for web browsers that don't support RGBa */
background-color: rgb(0, 0, 0);

/* RGBa with 0.6 opacity */
background-color: rgba(0, 0, 0, 0.6);

/* For IE 5.5 - 7*/
filter: progid:DXImageTransform.Microsoft.gradient(startColorstr=#99000000,
endColorstr=#99000000);

/* For IE 8*/
-ms-filter: "progid:DXImageTransform.Microsoft.gradient(startColorstr=#99000000,
endColorstr=#99000000)";
```

の

CSS3では、じにのをねることができます。

```
#mydiv {
```

```
  background-image: url(img_1.png), /* top image */
                    url(img_2.png), /* middle image */
                    url(img_3.png); /* bottom image */
  background-position: right bottom,
                       left top,
                       right top;
  background-repeat: no-repeat,
                     repeat,
                     no-repeat;
}
```

は、のがに、のがろにねてねられます。 `img_1`がにされ、 `img_2`と`img_3`がにされます。

これにバックグラウンドプロパティをすることもできます

```
#mydiv {
  background: url(img_1.png) right bottom no-repeat,
              url(img_2.png) left top repeat,
              url(img_3.png) right top no-repeat;
}
```

とグラデーションをみねることもできます

```
#mydiv {
  background: url(image.png) right bottom no-repeat,
              linear-gradient(to bottom, #fff 0%,#000 100%);
}
```

- デモ

プロパティ

background-originプロパティは、イメージのをします。

`background-attachment`プロパティが`fixed`にされている`fixed` 、このプロパティはです。

デフォルト `padding-box`

な

- `padding-box` - はパディングボックスをにしています
- `border-box` - はボックスにです
- `content-box` - はコンテンツボックスにです
- `initial`
- `inherit`

CSS

```
.example {
  width: 300px;
  border: 20px solid black;
  padding: 50px;
```

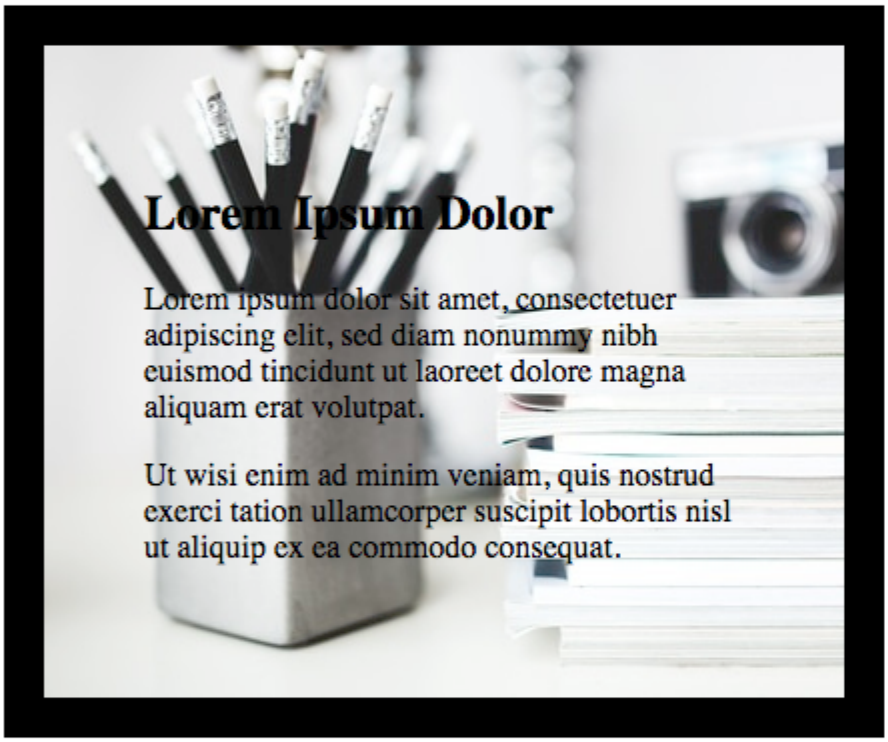
```
  background: url(https://static.pexels.com/photos/6440/magazines-desk-work-workspace-
medium.jpg);
  background-repeat: no-repeat;
}

.example1 {}

.example2 { background-origin: border-box; }

.example3 { background-origin: content-box; }
```

HTML

```
<p>No background-origin (padding-box is default):</p>

<div class="example example1">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: border-box:</p>
<div class="example example2">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: content-box:</p>
<div class="example example3">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>
```

No background-origin (padding-box is default):

background-origin: border-box:



background-origin: content-box:

- `padding-box`は、のパディングののにをクリップし、それがにびないようにします。
- `content-box`は、コンテンツボックスのにをクリップする。
- `inherit`は、したにのをします。

CSS

```
.example {
  width: 300px;
  border: 20px solid black;
  padding: 50px;
  background: url(https://static.pexels.com/photos/6440/magazines-desk-work-workspace-
medium.jpg);
  background-repeat: no-repeat;
}

.example1 {}

.example2 { background-origin: border-box; }

.example3 { background-origin: content-box; }
```

HTML

```
<p>No background-origin (padding-box is default):</p>

<div class="example example1">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: border-box:</p>
<div class="example example2">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: content-box:</p>
<div class="example example3">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>
```

バックグラウンドサイズ

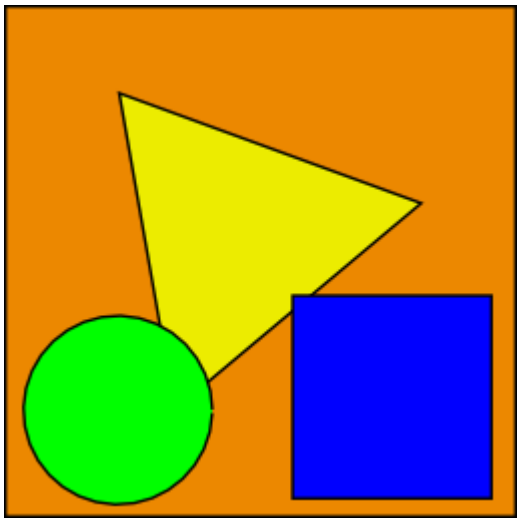
`background-size`プロパティは、 `background-image`スケーリングをすることをにする。それは、およ

びののスケール/サイズをする2つのをります。プロパティがないは、 `width`と`height`が`auto`とみなされます。

`auto`をすると、のアスペクトがされます。さはオプションで、 `auto`となすことができます。したがって、256ピクセル×256ピクセルのでは、の`background-size`ですべてさとが50ピクセルのが`background-size`ます。

```
background-size: 50px;
background-size: 50px auto; /* same as above */
background-size: auto 50px;
background-size: 50px 50px;
```

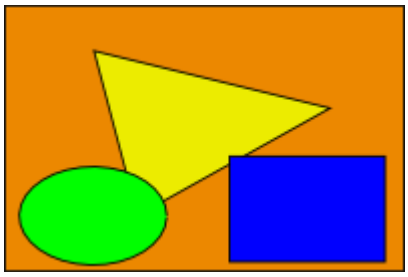
したがって、ののサイズが256ピクセル×256ピクセルからめた、



たちはのにまれるユーザーのに50ピクセル×50ピクセルでします。

パーセンテージをして、にしてを/することもできます。のでは、200ピクセル×133ピクセルのがられます。

```
#withbackground {
    background-image: url(to/some/background.png);

    background-size: 100% 66%;

    width: 200px;
    height: 200px;

    padding: 0;
    margin: 0;
}
```



そのるいは`background-origin`する。

アスペクトをする

previosセクションののでは、のアスペクトがわれました。はになり、はになり、はのになります。

さまたはパ―セントのアプロ―チは、にアスペクトをするのになほどではありません。のどのがきくなるかわからないがあるので、 `auto`はにちません。しかし、のをでにいまたはしいアスペクト、しいアスペクトのをにに`contain` 、 、 `contain`および`cover`がをします。

`contain`と`cover`ためのの

しありませんが、たちはデビュ―のためにBiswarup Gangulyによってのをするつもりです。これはあなたので、のはえるのにあるといます。デモンストレ―ションのために、々は16×9のをします。

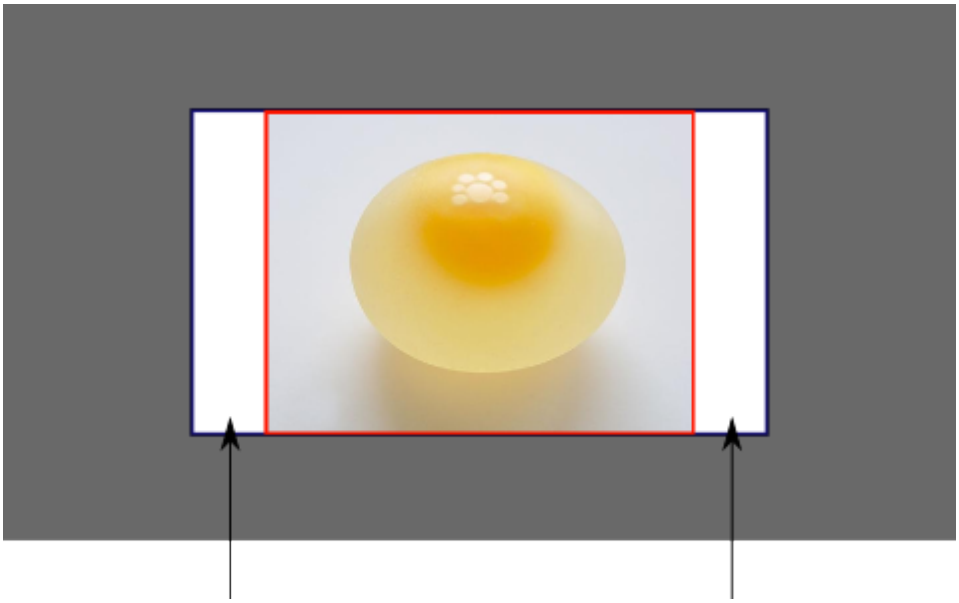
ののをバックグラウンドとしてしたいとえています。しかし、らかのでを4×3にりった。`background-size`プロパティをにすることもできますが、 `contain`と`cover`をて`contain`。 は、 `body`やさをえていないともしています。

`contain`

```
contain
```

とさのがバックグラウンドのめにまるように、そののアスペクトするをのサイズにしながら、を/します。

これにより、バックグラウンドイメ―ジがにバックグラウンドのめににまれていることがされますが、このは`background-color`でりつぶされたがあります

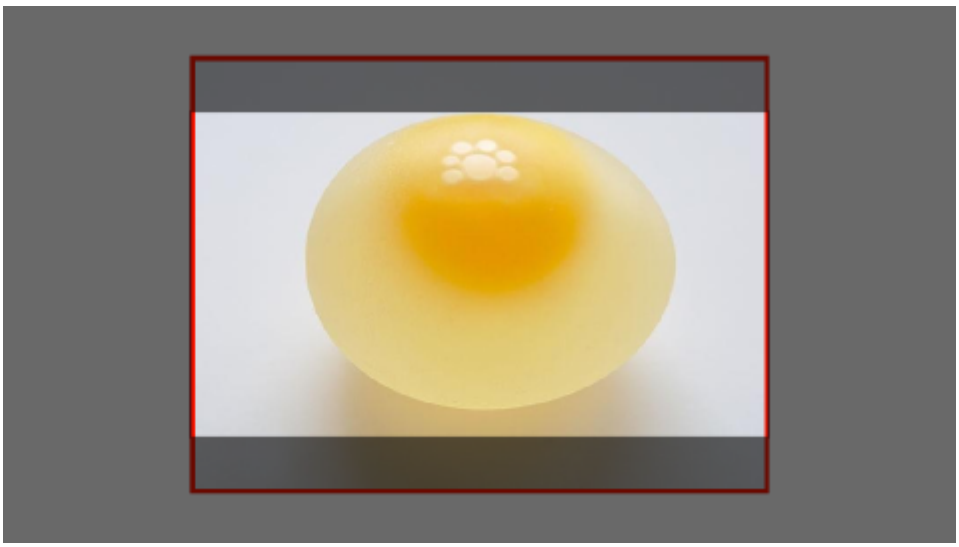


cover

```
cover
```

のとさのがバックグラウンドのめをにうように、そののアスペクトするをサイズにしながらを/します。

これは、イメ―ジがすべてをカバ―していることをします。にえる`background-color`はありませんが、のによってはのがれることがあります



のコ―ドによるデモンストレ―ション

```
div > div {
  background-image: url(http://i.stack.imgur.com/r5CAq.jpg);
  background-repeat: no-repeat;
  background-position: center center;
  background-color: #ccc;
  border: 1px solid;
  width: 20em;
  height: 10em;
}
div.contain {
  background-size: contain;
}
div.cover {
  background-size: cover;
}
/*****************************************
 Additional styles for the explanation boxes
*****************************************/

div > div {
  margin: 0 1ex 1ex 0;
  float: left;
}
div + div {
  clear: both;
  border-top: 1px dashed silver;
  padding-top:1ex;
}
div > div::after {
  background-color: #000;
  color: #fefefe;
  margin: 1ex;
  padding: 1ex;
  opacity: 0.8;
  display: block;
  width: 10ex;
  font-size: 0.7em;
  content: attr(class);
}
```

```
<div>
  <div class="contain"></div>
  <p>Note the grey background. The image does not cover the whole region, but it's fully
<em>contained</em>.
  </p>
</div>
<div>
  <div class="cover"></div>
  <p>Note the ducks/geese at the bottom of the image. Most of the water is cut, as well as a
part of the sky. You don't see the complete image anymore, but neither do you see any
background color; the image <em>covers</em> all of the <code>&lt;div&gt;</code>.</p>
</div>
```

Note the grey background. The image does not cover the whole region, but it's fully *contained*.

Note the ducks/geese at the bottom of the image. Most of the water is cut, as well as a part of the sky. You don't see the complete image anymore, but neither do you see any background color; the image *covers* all of the `<div>`.

background-blend-modeプロパティ

```
.my-div {
    width: 300px;
    height: 200px;
    background-size: 100%;
    background-repeat: no-repeat;
    background-image: linear-gradient(to right, black 0%,white 100%),
url('https://static.pexels.com/photos/54624/strawberry-fruit-red-sweet-54624-medium.jpeg');
    background-blend-mode:saturation;
}
```

```
<div class="my-div">Lorem ipsum</div>
```

はこちら https : //jsfiddle.net/MadalinaTn/y69d28Lb/

CSSシンタックスbackground-blend-modenormal ||スクリーン|オーバーレイ|ダークン| |カラードッジ||カラー|;

オンラインでをむ https://riptutorial.com/ja/css/topic/296/

52:

- #rgb
- #rrggbb
- colorrgb [a]<red>、<green>、<blue> [、<alpha>]
- hsl [a]<>、<>、<> [、<α>]
- colorkeyword / *、、、オレンジ、、..etc * /

Examples

カラーキーワード

ほとんどのブラウザでは、のキーワードをしてをできます。たとえば、の`color`をにするには、`blue`キーワードをします。

```
.some-class {
    color: blue;
}
```

CSSキーワードはsensitive-をされていない`blue`、 `Blue`と`BLUE`のすべてのでしょう`#0000FF`。

カラーキーワード

の	16	RGB	
AliceBlue	F0F8FF	rgb240,248,255	
アンティークホワイト	FAEBD7	rgb250,235,215	
アクア	00FFFF	rgb0,255,255	
アクアマリン	7FFFD4	rgb127,255,212	
アズール	F0FFFF	rgb240,255,255	
ベージュ	F5F5DC	rgb245,245,220	
ビスク	FFE4C4	rgb255,228,196	
ブラック	000000	rgb0,0,0	

の	16	RGB	
BlanchedAlmond	#FFEBCD	rgb255,235,205	
	0000FF	rgb0,0,255	
	8A2BE2	rgb138,43,226	
	A52A2A	rgb165,42,42	
BurlyWood	DEB887	rgb222,184,135	
CadetBlue	5F9EA0	rgb95,158,160	
Chartreuse	7FFF00	rgb127,255,0	
チョコレート	D2691E	rgb210,105,30	
コーラル	FF7F50	rgb255,127,80	
コーンフラワーブルー	6495ED	rgb100,149,237	
トウモロコシの	FFF8DC	rgb255,248,220	
	DC143C	rgb220,20,60	
シアン	00FFFF	rgb0,255,255	
	00008B	rgb0,0,139	
ダークシアン	008B8B	rgb0,139,139	
DarkGoldenRod	B8860B	rgb184,134,11	
	A9A9A9	rgb169,169,169	
い	A9A9A9	rgb169,169,169	
い	006400	rgb0,100,0	
ダークカーキ	BDB76B	rgb189,183,107	

の	16	RGB	
ダークマジェンタ	8B008B	rgb139,0,139	
DarkOliveGreen	556B2F	rgb85,107,47	
ダークオーレンジ	FF8C00	rgb255,140,0	
DarkOrchid	9932CC	rgb153,50,204	
	8B0000	rgb139,0,0	
ダークサルモン	E9967A	rgb233,150,122	
DarkSeaGreen	8FBC8F	rgb143,188,143	
DarkSlateBlue	483D8B	rgb72,61,139	
DarkSlateGray	2F4F4F	rgb47,79,79	
DarkSlateGrey	2F4F4F	rgb47,79,79	
ダークターコイズ	00CED1	rgb0,206,209	
ダークバイオレット	9400D3	rgb148,0,211	
いピンク	FF1493	rgb255,20,147	
DeepSkyBlue	00BFFF	rgb0,191,255	
DimGray	696969	rgb105,105,105	
DimGrey	696969	rgb105,105,105	
DodgerBlue	1E90FF	rgb30,144,255	
FireBrick	B22222	rgb178,34,34	
FloralWhite	FFFAF0	rgb255,250,240	
の	228B22	rgb34,139,34	

の	16	RGB	
フクシア	FF00FF	rgb255,0,255	
ゲインズボロ	#DCDCDC	rgb220,220,220	
ゴーストホワイト	F8F8FF	rgb248,248,255	
ゴールド	FFD700	rgb255,215,0	
ゴールデンロード	DAA520	rgb218,165,32	
グレー	808080	rgb128,128,128	
グレー	808080	rgb128,128,128	
	008000	rgb0,128,0	
	ADFF2F	rgb173,255,47	
	F0FFF0	rgb240,255,240	
ホットピンク	FF69B4	rgb255,105,180	
IndianRed	CD5C5C	rgb205,92,92	
インジゴ	4B0082	rgb75,0,130	
	FFFFF0	rgb255,255,240	
カーキ	F0E68C	rgb240,230,140	
ラベンダー	E6E6FA	rgb230,230,250	
LavenderBlush	FFF0F5	rgb255,240,245	
	7CFC00	rgb124,252,0	
レモンシフォン	#FFFACD	rgb255,250,205	
ライトブルー	ADD8E6	rgb173,216,230	

の	16	RGB	
ライトコラル	F08080	rgb240,128,128	
LightCyan	E0FFFF	rgb224,255,255	
LightGoldenRodYellow	FAFAD2	rgb250,250,210	
ライトグレー	D3D3D3	rgb211,211,211	
ライトグレー	D3D3D3	rgb211,211,211	
ライトグリーン	90EE90	rgb144,238,144	
ライトピンク	FFB6C1	rgb255,182,193	
ライトサルモン	FFA07A	rgb255,160,122	
LightSeaGreen	20B2AA	rgb32,178,170	
LightSkyBlue	87CEFA	rgb135,206,250	
LightSlateGray	778899	rgb119,136,153	
LightSlateGrey	778899	rgb119,136,153	
LightSteelBlue	B0C4DE	rgb176,196,222	
ライトイエロー	FFFFE0	rgb255,255,224	
ライム	00FF00	rgb0,255,0	
ライムグリーン	32CD32	rgb50,205,50	
リネン	FAF0E6	rgb250,240,230	
マゼンタ	FF00FF	rgb255,0,255	
マルーン	800000	rgb128,0,0	
ミディアムアクアマリン	66CDAA	rgb102,205,170	

の	16	RGB	
ミディアムブルー	0000CD	rgb0,0,205	
MediumOrchid	BA55D3	rgb186,85,211	
ミディアムパープル	9370DB	rgb147,112,219	
ミディアムシーグリーン	3CB371	rgb60,179,113	
MediumSlateBlue	7B68EE	rgb123,104,238	
MediumSpringGreen	00FA9A	rgb0,250,154	
ミディアムターコイズ	48D1CC	rgb72,209,204	
MediumVioletRed	C71585	rgb199,21,133	
ミッドナイトブルー	191970	rgb25,25,112	
ミントクリーム	F5FFFA	rgb245,255,250	
ミスティローズ	FFE4E1	rgb255,228,225	
モカシン	FFE4B5	rgb255,228,181	
ナバホワホワイト	#FFDEAD	rgb255,222,173	
	000080	rgb0,0,128	
オールドレース	FDF5E6	rgb253,245,230	
オリーブ	808000	rgb128,128,0	
OliveDrab	6B8E23	rgb107,142,35	
オレンジ	FFA500	rgb255,165,0	
オレンジレッド	FF4500	rgb255,69,0	
	DA70D6	rgb218,112,214	

の	16	RGB	
PaleGoldenRod	EEE8AA	rgb238,232,170	
	98FB98	rgb152,251,152	
ペールターコイズ	#AFEEEE	rgb175,238,238	
PaleVioletRed	DB7093	rgb219,112,147	
パパイヤウィップ	FFEFD5	rgb255,239,213	
PeachPuff	FFDAB9	rgb255,218,185	
ペルー	CD853F	rgb205,133,63	
ピンク	FFC0CB	rgb255,192,203	
	DDA0DD	rgb221,160,221	
パウダーブルー	B0E0E6	rgb176,224,230	
の	800080	rgb128,0,128	
レベッカパープル	663399	rgb102,51,153	
	FF0000	rgb255,0,0	
RosyBrown	BC8F8F	rgb188,143,143	
ロイヤルブルー	4169E1	rgb65,105,225	
サドルブラウン	8B4513	rgb139,69,19	
サーモン	FA8072	rgb250,128,114	
SandyBrown	F4A460	rgb244,164,96	
	2E8B57	rgb46,139,87	
シーシェル	FFF5EE	rgb255,245,238	

の	16	RGB	
シエナ	A0522D	rgb160,82,45	
	C0C0C0	rgb192,192,192	
	87CEEB	rgb135,206,235	
SlateBlue	6A5ACD	rgb106,90,205	
SlateGray	708090	rgb112,128,144	
SlateGrey	708090	rgb112,128,144	
	#FFFAFA	rgb255,250,250	
SpringGreen	00FF7F	rgb0,255,127	
スチールブルー	4682B4	rgb70,130,180	
タン	D2B48C	rgb210,180,140	
ティール	008080	rgb0,128,128	
アザミ	D8BFD8	rgb216,191,216	
トマト	FF6347	rgb255,99,71	
ターコイズ	40E0D0	rgb64,224,208	
バイオレット	EE82EE	rgb238,130,238	
	F5DEB3	rgb245,222,179	
	#FFFFFF	rgb255,255,255	
い	F5F5F5	rgb245,245,245	
	FFFF00	rgb255,255,0	
YellowGreen	9ACD32	rgb154,205,50	

されたにえて、なをすキーワード`transparent`もあります`rgba(0,0,0,0)`

16

バックグラウンド

CSSのは、の、、のをすヘックスのつとしてすることもできます。これらののそれぞれののをし`00`に`FF`、または`0`に`255`で。またはのHexidecimalをできますつまり、`#3fc` = `#3FC` = `#33ffCC`。ブラウザは`#369`を`#336699`します。それがしたものではなく`#306090`をむなら、にするがあります。

16でできるのは、256 ^ 3または16,777,216です。

```
color: #rrggbb;
color: #rgb
```

`rr`	`00` - のの`FF`
`gg`	`00` - `FF`のの
`bb`	`00` - ののFF

```
.some-class {
    /* This is equivalent to using the color keyword 'blue' */
    color: #0000FF;
}

.also-blue {
    /* If you want to specify each range value with a single number, you can!
       This is equivalent to '#0000FF' (and 'blue') */
    color: #00F;
}
```

16は、 W3Cの「カラー」ごとにRGBカラーフォーマットでカラーをするためにされます。

インターネットでは、16またはに16のカラーをべるためのツールがたくさんあります。

おにりのWebブラウザで「 **16**カラーパレット 」または「 **16**カラーピッカー 」をして、さまざまなオプションをつけましょう

16はにシャープでまり、6ので、とをしません。つまり、をしません。`#FFC125`と`#ffc125`はじです。

rgb

RGBは、、、ののとしてのをすなモデルである。に、RGBは、16の10にします。 16では、のは00FFで、10で0255、パーセントで0-100にします。

```
.some-class {
    /* Scalar RGB, equivalent to 'blue'*/
    color: rgb(0, 0, 255);
}

.also-blue {
    /* Percentile RGB values*/
    color: rgb(0%, 0%, 100%);
}
```

```
rgb(<red>, <green>, <blue>)
```

`<red>`	0255のまたは0100のパーセンテージ
`<green>`	0255のまたは0100のパーセンテージ
`<blue>`	0255のまたは0100のパーセンテージ

hsl

HSLは、「どの」、「どのくらいの」、「どれくらいの」をします。

は0°360°なしのでされ、とはパーセンテージでされます。

```
p {
    color: hsl(240, 100%, 50%); /* Blue */
}
```

```
color: hsl(<hue>, <saturation>%, <lightness>%);
```

`<hue>`	0は、60は、120は、180はシアン、240は、300はマゼンタ、360はです。
`<saturation>`	0がにグレースケールで100がやかなであるパーセンテージでされます。
`<lightness>`	でします.0はにで、100はにです

ノート

- 0のはにグレースケールのをします。をしてもはありません。
- 0のはにをし、100はにをします。またはをしてもはありません。

currentColor

currentColorは、ののされたカラーをします。

じでする

currentColorは、 colorプロパティがredされているため、にされred。

```
div {
   color: red;
   border: 5px solid currentColor;
   box-shadow: 0 0 5px currentColor;
}
```

この、ボーダーにcurrentColorをすると、それをするとじがされるため、がくなります。 よりのセレクタのためにきされるは、じのborderプロパティのでのみcurrentColorをしてください 。

これはされたなので、のではがになります。これは、2のルールが1のルールをきするためです。

```
div {
   color: blue;
   border: 3px solid currentColor;
   color: green;
}
```

からされます。

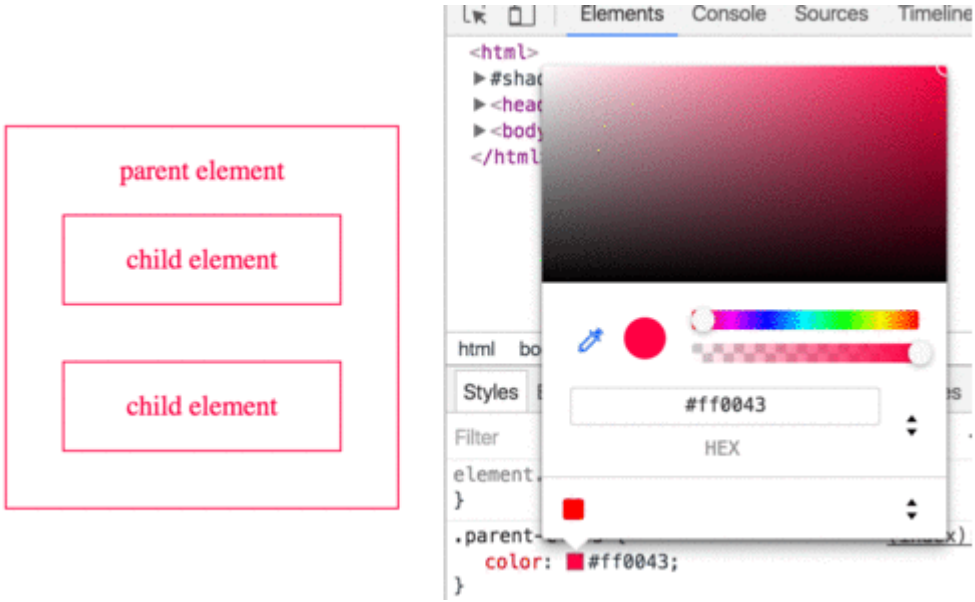
のがされます。ここではcurrentColorは「」とされ、ののがになります。

```
.parent-class {
    color: blue;
}

.parent-class .child-class {
    border-color: currentColor;
}
```

currentColorは、background-colorなど、colorプロパティからしないのルールでもできます。のでは、としてされているをとしてしているをしています。

```
.parent-class {
    color: blue;
}

.parent-class .child-class {
    background-color: currentColor;
}
```

えられる



rgba

rgbとていますが、のアルファがいています。

```
.red {
    /* Opaque red */
    color: rgba(255, 0, 0, 1);
}

.red-50p {
    /* Half-translucent red. */
    color: rgba(255, 0, 0, .5);
}
```

```
rgba(<red>, <green>, <blue>, <alpha>);
```

`<red>`	0255のまたは0100のパーセンテージ
`<green>`	0255のまたは0100のパーセンテージ
`<blue>`	0255のまたは0100のパーセンテージ
`<alpha>`	01の。0.0はに、1.0はにです。

hsla

hslにていますが、アルファがされています。

```
hsla(240, 100%, 50%, 0)      /* transparent */
hsla(240, 100%, 50%, 0.5)    /* half-translucent blue */
hsla(240, 100%, 50%, 1)      /* fully opaque blue */
```

```
hsla(<hue>, <saturation>%, <lightness>%, <alpha>);
```

`<hue>`	0は、60は、120は、180はシアン、240は、300はマゼンタ、360はです。
`<saturation>`	0がにグレースケールで100がにしているやかな
`<lightness>`	0がにで100がにである
`<alpha>`	0はに、1はにです。

オンラインでをむ https://riptutorial.com/ja/css/topic/644/

53: の

き

CSSでは、のをのにラップし、それらのにギャップやをれてすることができます。

CSSマルチカラムレイアウトモジュールレベル1は、2011412、W3Cのです。それ、しのがえられました 。 したにあるとえられます。

201773、MicrosoftのInternet Explorer 10および11およびEdgeブラウザは、ベンダーのプレフィックスをしてバージョンののみをサポートしています。

Examples

な

のHTMLマークアップをえてみましょう。

```
<section>
  <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum.</p>
  <p> Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.</p>
  <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.</p>
</section>
```

のCSSをすると、コンテンツは2つのピクセルののルールでられた3つのにされます。

```
section {
  columns: 3;
  column-gap: 40px;
  column-rule: 2px solid gray;
}
```

JSFiddleのライブ·サンプルをください。

のをする

```
<div class="content">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim
```

```
ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl
ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in
hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu
feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui
blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla
facilisi. Nam liber tempor cum soluta nobis eleifend option congue nihil
imperdiet doming id quod mazim placerat facer possim assum.
</div>
```

Css

```
.content {
-webkit-column-count: 3; /* Chrome, Safari, Opera */
-moz-column-count: 3; /* Firefox */
column-count: 3;
}
```

オンラインでのをむ https://riptutorial.com/ja/css/topic/10688/の

54:

- アウトラインアウトライン - アウトライン - スタイルアウトライン - |の|する;
- アウトライン - ミディアム|い|い|さ|の|する;
- アウトラインスタイルなし|された|する| |ソリッド|ダブル|グルーブ| |インセット|まり|の|する;

パラメーター

パラメータ	
	の
	の
	の
ダブル	
	3Dの、ののにする
リッジ	3Dリッジきアウトライン、アウトラインカラーに
インセット	3Dインセットアウトライン、アウトラインカラーに
	3Dアウトライン、アウトラインカラーに
し	アウトラインなし
された	し

`outline`はCSSモジューールレベル3のUI これはすでにREC CSS2.1でされています

Outlineプロパティは`:focus`ののブラウザでデフォルトでされています`:focus`。
しないでください。 http : //outlinenone.comをしてください。

> アウトラインプロパティはをしますか
>
> TABキーまたはこれにするものをしてWebドキュメントをナビゲートするときに、「フォーカス」をつリンクになフィードバックをします。これは、マウスをできないやにとってにです。アウトラインをすると、これらの々のためにサイトにアクセスできなくなります。 ...

スタックオーバーフローにするい

- テキストののハイライトをする
- リンクだけでなく、Firefoxののをするには

Examples

アウトラインは、のでをむです。 `border`とはに、アウトラインはボックスモデルにスペースをとらない。したがって、にアウトラインをしても、またはののにはしません。

さらに、アウトラインは、のブラウザではにすることができます。これは、 `font-size`プロパティがなるテキストをむ`span`に`outline`がされているにします。とはなり、はをくすることはできません。

`outline`のなは`outline` `outline-color` 、 `outline-style` 、 `outline-width`です。

アウトラインのは、のとじです。

> アウトラインはのりのです。ののにされます。ただし、ボーダープロパティとはなります。

```
outline: 1px solid black;
```

アウトラインスタイル

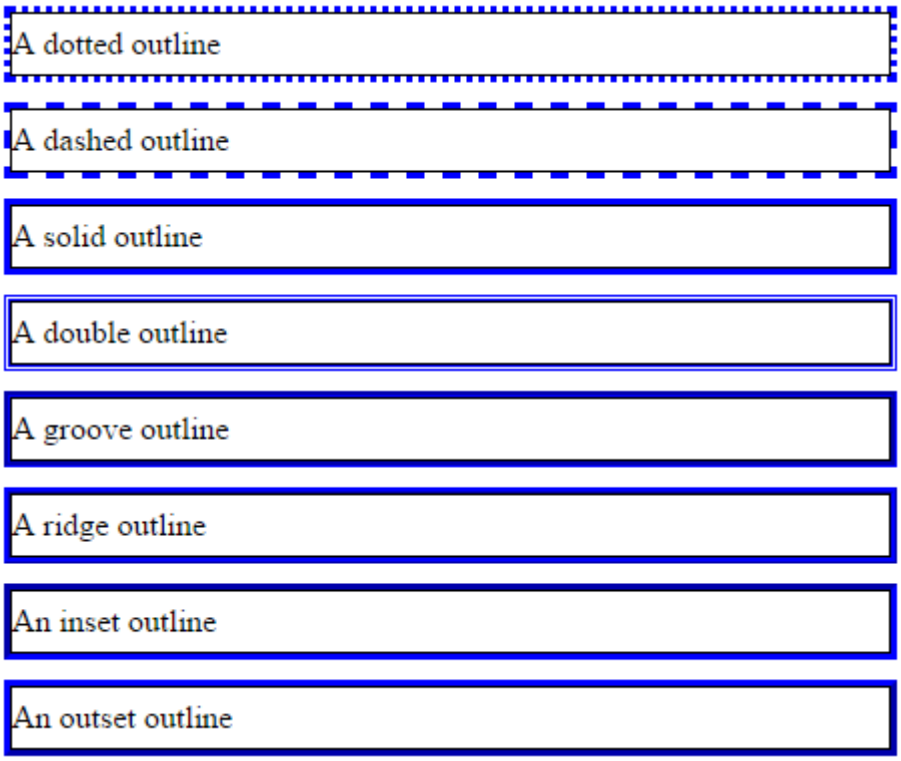
`outline-style`プロパティは、のアウトラインのスタイルをするためにされます。

```
p {
  border: 1px solid black;
  outline-color:blue;
  line-height:30px;
}
.p1{
  outline-style: dotted;
}
.p2{
  outline-style: dashed;
}
.p3{
  outline-style: solid;
}
.p4{
  outline-style: double;
}
.p5{
  outline-style: groove;
}
.p6{
  outline-style: ridge;
}
.p7{
  outline-style: inset;
}
.p8{
```

```
  outline-style: outset;
}
```

HTML

```
<p class="p1">A dotted outline</p>
<p class="p2">A dashed outline</p>
<p class="p3">A solid outline</p>
<p class="p4">A double outline</p>
<p class="p5">A groove outline</p>
<p class="p6">A ridge outline</p>
<p class="p7">An inset outline</p>
<p class="p8">An outset outline</p>
```



オンラインでをむ https://riptutorial.com/ja/css/topic/4258/

55: さ

き

CSSは、にさpx、em、pc、in、...がくです。

CSSはいくつかのさをサポートしています。らはまたはです。

-
- 1em

パラメーター

	オブジェクトまたはプロパティにするのオブジェクトにしてサイズをする
それら	のフォントサイズ2emはのフォントのサイズの2をしますにして、
レム	ルートのfont-sizeに
VW	ビューポートのの1にして*
vh	ビューポートのさの1
vmin	ビューポートの*さいのの1
vmax	ビューポートの*きいのの1
CM	センチメートル
mm	ミリメートル
に	インチ1インチ= 96ピクセル= 2.54cm
px	ピクセル1ピクセル= 1/96の1インチ
pt	ポイント1pt = 1/72 = 1in
pc	ピカ1= 12
s	アニメーションとトランジションに
ミズ	ミリアニメーションとトランジションに
	のフォントのx-さに

ch	ゼロ0のについて
fr	ユニットCSSグリッドレイアウトで

- とユニットのにをれることはできません。ただし、が0のはすることができます。
- のCSSプロパティでは、のさをできます。

Examples

remのフォントサイズ

CSS3では、「root em」をすremユニットをむいくつかのしいユニットがされています。 remどのようにくかてみましょう。

まず、 emとremいをてみましょう。

- **em** のフォントサイズにします。これはをきこす
- **rem** ルートまたは<html>のフォントサイズに<html>ます。これは、htmlのためののフォントサイズをし、すべてすることがですremそのようにを。

フォントサイジングのためにremをするのなは、がいにくいことです。のは、のサイズが16pxであるとして、 remでされるなフォントサイズのです。

- 10ピクセル= 0.625rem
- 12px = 0.75rem
- 14px = 0.875rem
- 16px = 1remベース
- 18px = 1.125rem
- 20px = 1.25rem
- 24px = 1.5rem
- 30px = 1.875rem
- 32px = 2rem

コード

3

```
html {
  font-size: 16px;
}

h1 {
  font-size: 2rem;          /* 32px */
}

p {
  font-size: 1rem;          /* 16px */
```

```
}

li {
  font-size: 1.5em;         /* 24px */
}
```

remsとemをってスケーラブルなをする

3

あなたはすることができ`rem`によってされ`font-size`、あなたの`html`らのすることで、スタイルにタグを`font-size`のに`rem`と`em`グローバルにをするために`font-size`。

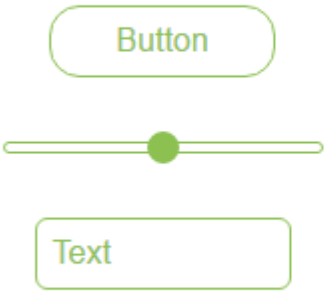
HTML

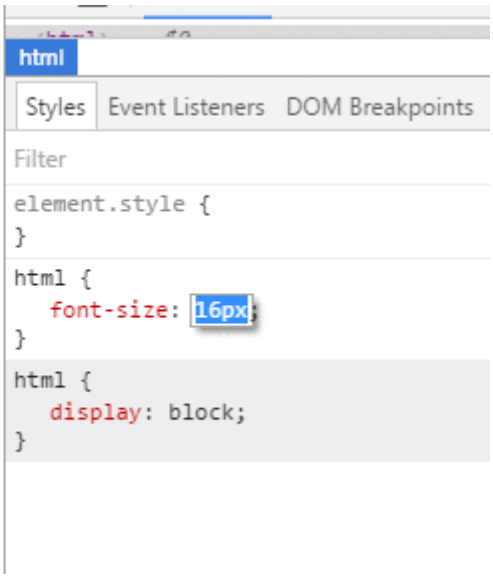
```
<input type="button" value="Button">
<input type="range">
<input type="text" value="Text">
```

する**CSS**

```
html {
  font-size: 16px;
}

input[type="button"] {
  font-size: 1rem;
  padding: 0.5em 2em;
}

input[type="range"] {
  font-size: 1rem;
  width: 10em;
}

input[type=text] {
  font-size: 1rem;
  padding: 0.5em;
}
```

えられる





vhおよびvw

CSS3は、サイズをするために2つのユニットをしました。

- `viewport height`をす`vh`は、 `viewport height` 1にです
- `viewport width`をす`vw`は、 `viewport width` 1にです

3

```
div {
  width: 20vw;
  height: 20vh;
}
```

の、divのサイズはビュ―ポ―トのとさの20をめます

vminおよびvmax

- **vmin** ビュ―ポ―トのよりさいの1に
- **vmax** ビュ―ポ―トのきいのの1に

すれば、1 `vmin` *1つのVHおよび1のVW*のさいにしいです。

1 `vmax`は*1 vh*と*1 vw*のきいにしい

`vmax`はでサポ―トされていません 。

- のバ―ジョンのInternet Explorer
- バ―ジョン6.1よりのSafari

パ―セント

のあるアプリケ―ションをするにつユニットの1つ。

そのサイズはコンテナによってなります。

コンテナの*パーセンテージ=

えば

は **50**をしているは **100px**にをしています。

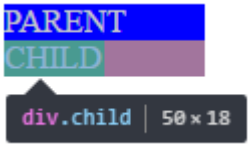
では、 のは50のの50になります 。

HTML

```
<div class="parent">
   PARENT
   <div class="child">
     CHILD
   </div>
</div>
```

CSS

```
<style>

*{
  color: #CCC;
}

.parent{
  background-color: blue;
  width: 100px;
}

.child{
  background-color: green;
  width: 50%;
}

</style>
```



オンラインでさをむ https://riptutorial.com/ja/css/topic/864/さ

56:

- `<calc()> = calc( <calc-sum> )`
- `<calc-sum> = <calc-product> [ [ '+' | '-' ] <calc-product> ]*`
- `<calc-product> = <calc-value> [ '*' <calc-value> | '/' <number> ]*`
- `<calc-value> = <number> | <dimension> | <percentage> | ( <calc-sum> )`

`calc()`、 "`-`"と "`+`"ではがですが、 "`*`"または "`/`"ではです。

すべてのユニットはじタイプでなければなりません。えば、のさでさをけることはです。

Examples

calc

をけれてをします。

これは、のをするために、なるタイプのをするたとえば、パーセントからpxをするなどににです。

`+`、 `-`、 `/`、 `*`をすべてすることができ、にじてのをするためにをできます。

`calc()`をしてdivのをします。

```
#div1 {
    position: absolute;
    left: 50px;
    width: calc(100% - 100px);
    border: 1px solid black;
    background-color: yellow;
    padding: 5px;
    text-align: center;
}
```

`calc()`をしてのをする

```
background-position: calc(50% + 17px) calc(50% + 10px), 50% 50%;
```

のさを`calc()`するには、 `calc()`をします。

```
height: calc(100% - 20px);
```

attr

したののをします。

は、このをしてCSSができる`data-*`にをむblockquoteです `::before`および`::after` 。

```
<blockquote data-mark='"'></blockquote>
```

のCSSブロックでは、のテキストのにがされます。

```
blockquote[data-mark]::before,
blockquote[data-mark]::after {
    content: attr(data-mark);
}
```

カラーのグラデーションをすイメージをします。

```
linear-gradient( 0deg, red, yellow 50%, blue);
```

これにより、からにかってグラデーションがされ、でし、にで50でまり、でします。

radial-gradient

グラデーションのからするのグラデーションをすイメージをします。

```
radial-gradient(red, orange, yellow) /*A gradient coming out from the middle of the
gradient, red at the center, then orange, until it is finally yellow at the edges*/
```

var

varは、CSSへのアクセスをします。

```
/* set a variable */
:root {
    --primary-color: blue;
}

/* access variable */
selector {
    color: var(--primary-color);
}
```

このはです。 caniuse.comでのブラウザサポートをしてください。

オンラインでをむ https://riptutorial.com/ja/css/topic/2214/

クレジット

S. No		Contributors
1	CSSをいめる	adamboro, animuson, Ashwin Ramaswami, awe, Boysenb3rry, Chris, Community, csx.cc, darrylyeo, FelipeAls, Gabriel R., Garconis, Gerardas, GoatsWearHats, G-Wiz, Harish Gyanani, Heri Hehe Setiawan, J Atkin, Jmh2013, joe_young, Jose Gomez , Just a student, Lambda Ninja, Marjorie Pickard, Nathan Arthur, patelarpan, RamenChef, Rocket Risa, Saroj Sasmal, ScientiaEtVeritas, selvassn, Sverri M. Olsen, Teo Dragovic, Todd, TylerH, Vivek Ghaisas, Xinyang Li, ZaneDickens
2	2D	Charlie H, Christiaan Maks, Harry, Hors Sujet, John Slegers, Luke Taylor, Madalina Taina, mnoronha, PaMaDo, Praveen Kumar, RamenChef, web-tiki, zer00ne
3	3D	Harry, Luka Kerr, Madalina Taina, mnoronha, Mr. Meeseeks, Nhan, RamenChef, web-tiki
4	CSSイメージスプライト	Elegant.Scripting, Jmh2013, RamenChef, Ted Goas, Ulrich Schwarz
5	CSSオブジェクトモデルCSSOM	Alohci, animuson, feeela, Paul Sweatte, RamenChef, rishabh dev
6	CSSデザインパターン	John Slegers
7	CSSルールのと	Alon Eitan, darrylyeo, Marjorie Pickard
8	Internet Explorerのハック	Aeolingamenfel, animuson, Elizaveta Revyakina, feeela, John Slegers, LiLacTac, Praveen Kumar, RamenChef, ScientiaEtVeritas, Timothy Miller, TylerH
9	アニメーション	Aeolingamenfel, apaul, Dex Star, Jasmin Solanki, Nathan Arthur , RamenChef, Richard Hamilton, TylerH
10	インラインブロックレイアウト	Marten Koetsier, RamenChef
11	オーバーフロー	Andrew, bdkopen, jgh, Madalina Taina, Miles, Qaz, RamenChef , ScientiaEtVeritas, Ted Goas, Zac
12	オブジェクトのフィ	4444, Miles, RamenChef

	ットと	
13	カーソルのスタイル	cone56, Madalina Taina, ScientiaEtVeritas, Squazz
14	カウンター	Harry, RamenChef
15	カスケーディングと	amflare, Arjan Einbu, brandaemon, DarkAjax, dippas, dmkerr, geeksal, Grant Palin, G-Wiz, James Donnelly, jehna1, John Slegers, kingcobra1986, Matas Vaitkevicius, Nathan Arthur, Nhan, Ortomala Lokni, RamenChef, ScientiaEtVeritas, Sunnyok , Ze Rubeus
16	カスタムプロパティ	animuson, Brett DeWoody, Community, Daniel Käfer, Muthu Kumaran, Obsidian, RamenChef, RedRiderX, TylerH
17	グリッド	Chris Spittles, FelipeAls, Jonathan Zúñiga, Mike McCaughan, Nhan, Praveen Kumar, RamenChef, Sebastian Zartner
18	クリッピングとマスキング	Andre Lopes, Harry, RamenChef, ScientiaEtVeritas, web-tiki
19	コメント	animuson, bdkopen, coderfin, Madalina Taina, Nick
20	コンテキストをブロックする	Madalina Taina, Milan Laslop, RamenChef
21	スタイルをリストする	animuson, Madalina Taina, Marten Koetsier, RamenChef, Ted Goas
22	スタッキングコンテキスト	Nemanja Trifunovic, RamenChef
23	セレクタ	75th Trombone, A.J, Aaron, abaracedo, Ahmad Alfy, Alex Filatov, amflare, Anil, animuson, Araknid, Arjan Einbu, Ashwin Ramaswami, BoltClock, Cerbrus, Charlie H, Chris, Chris Nager, Clinton Yeboah, Community, CPHPython, darrylyeo, Dave Everitt, David Fullerton, Demeter Dimitri, designcise, Devid Farinelli, Devon Bernard, Dinidu, dippas, Erenor Paz, Felix Edelmann, Felix Schütz, flyingfisch, Forty, fracz, Frits, gandreadis, geeksal, George Bailey, George Grigorita, H. Pauwelyn, HansCz, henry, Hugo Buff, Hynes, J Atkin, J F, Jacob Gray, James Donnelly, James Taylor, Jasha, jehna1, Jmh2013, joejoe31b, Joël Bonet Rodríguez, John Slegers, Kurtis Beavers, Madalina Taina, Marc, Mark Perera, Matas Vaitkevicius, Matsemann, Michael_B, Milan Laslop, Naeem Shaikh, Nathan Arthur, Nick, Ortomala Lokni, Persijn, Praveen Kumar, RamenChef, rdans, Richard Hamilton, Rion Williams, Robert Koritnik, RockPaperLizard, RoToRa, Sbats,

		ScientiaEtVeritas, Shaggy, Siavas, Stewartside, sudo bangbang , Sumner Evans, Sunnyok, ThatWeirdo, theB, Thomas Gerot, TylerH, xpy, Yury Fedorov, Zac, Zaffy, Zaz, Ze Rubeus, Zze
24	センタリング	abaracedo, Alex Morales, Alohci, andreas, animuson, apaul, Christiaan Maks, Daniel Käfer, Devid Farinelli, Diego V, dippas, Eliran Malka, Emanuele Parisio, Euan Williams, F. Müller, Farzad YZ, Felix A J, geek1011, insertusernamehere, JedaiCoder, jehna1, JHS, John Slegers, Jonathan Argentiero, Kilian Stinson, Kyle Ratliff, Lambda Ninja, Lucia Bentivoglio, Luke Taylor, Madalina Taina, maho, Maxouhell, Michael_B, Mifeet, Mod Proxy, Mohammad Usman, Nathan Arthur, Nhan, o.v., Ortomala Lokni, paaacman, Paul Kozlovitch, Praveen Kumar, Sandeep Tuniki, ScientiaEtVeritas, Sergej, Siavas, smonff, Someone, Stewartside, Sunnyok, Taylor, TylerH, web-tiki, Ze Rubeus, zeel
25	タイポグラフィ	Alex Morales, Alohci, andreas, Arjan Einbu, ChaoticTwist, Evgeny, Felix A J, Goulven, Hynes, insertusernamehere, James Donnelly, joe_young, Jon Chan, Madalina Taina, Michael Moriarty, Nathan Arthur, Nhan, Praveen Kumar, RamenChef, Richard Hamilton, rmondesilva, Ryan, Sourav Ghosh, Suhaib Janjua, Ted Goas, Toby, ToniB, Trevor Clarke, user2622348, Vlusion, Volker E.
26	テーブル	Casper Spruit, Chris, FelipeAls, JHS, Madalina Taina
27	トランジション	Christiaan Maks, dippas, Harry, Praveen Kumar, RamenChef, Richard Hamilton, ScientiaEtVeritas, Sergey Denisov, web-tiki
28	パディング	Andy G, CalvT, Felix A J, Madalina Taina, Mehdi Dehghani, Nathan, Paul Sweatte, pixelbandito, RamenChef, StefanBob, Will DiFruscio
29	パフォーマンス	mnoronha, RamenChef, TrungDQ
30	フィーチャクエリ	Andrew Myers, RamenChef
31	フィルタプロパティ	Jeffery Tang, Nathan, RamenChef
32	ブラウザスタイルをする	andre mcgruder, Confused One, Grant Palin, MMachinegun, mnoronha, RamenChef, SeinopSys
33	ブラウザのサポートと	Andrew, animuson, Braiam, Nhan, Obsidian, RamenChef, ScientiaEtVeritas, Shaggy, TylerH
34	ポジショニング	Abhishek Singh, Alohci, animuson, Blunderfest, CalvT, Cassidy Williams, Chetan Joshi, GingerPlusPlus, Jacob Gray, Lambda

		Ninja, Madalina Taina, Matthew Beckman, Mattia Astorino, Rahul Nanwani, RamenChef, Sourav Ghosh, Stephen Leppik, Theodore K., TylerH
35	ボックスシャドウ	Hristo, Madalina Taina, RamenChef
36	ボックスモデル	Arjan Einbu, Ben Rhys-Lewis, Benolot, BiscuitBaker, FelipeAls, James Donnelly, Lambda Ninja, Nathan Arthur, Ortomala Lokni, RamenChef, ScientiaEtVeritas, Sergej, V-Kopio
37	マージン	Arjan Einbu, cdm, Chris Spittles, Community, J F, Madalina Taina, Mr_Green, Nathan Arthur, RamenChef, rejnev, Sun Qingyao, Sunnyok, Tot Zam, Trevor Clarke
38	メディアクエリ	amflare, Chathuranga Jayanath, darrylyeo, Demeter Dimitri, dodopok, James Donnelly, Jmh2013, joe_young, joejoe31b, John Slegers, Matas Vaitkevicius, Mattia Astorino, Maximillian Laumeister, Nathan Arthur, Praveen Kumar, RamenChef, ScientiaEtVeritas, srikarg, Teo Dragovic, Viktor
39	レイアウト	Arjun Iv, Chathuranga Jayanath, Chris, Jmh2013, Kevin Katzke, Kurtis Beavers, Madalina Taina, mnoronha, Niek Brouwer, RamenChef, Sander Koedood, ScientiaEtVeritas, SeinopSys
40		Andrew, animuson, Hillel Tech, Madalina Taina, RamenChef
41		Brett DeWoody, Madalina Taina, RamenChef
42	シェイプ	andreas, animuson, Brett DeWoody, Chiller, J F, Nick, RamenChef, ScientiaEtVeritas, TylerH
43	センタリング	animuson, AVAVT, bocanegra, Chiller, Chris, jaredsk, leo_ap, mmativ, patelarpan, Phil, Praveen Kumar, RamenChef
44		andreas, Cassidy Williams, doctorsherlock, FelipeAls, Gnietschow, Harry, jaredsk, Madalina Taina, Nobal Mohan, RamenChef, ScientiaEtVeritas, Trevor Clarke
45		4dgaurav, ankit, Chris, Community, dippas, dmnsgn, geek1011, geeksal, Gofilord, kevin vd, Marcatectura, Milche Patern, Nathan, Pat, Praveen Kumar, RamenChef, ScientiaEtVeritas, Shaggy, Stewartside, Sunnyok
46		animuson, dodopok, Madalina Taina, Milan Laslop, RamenChef
47	なボックスレイアウトFlexbox	Ahmad Alfy, Asim K T, FelipeAls, fzzylogic, James Donnelly, Jef , Lambda Ninja, Marc-Antoine Leclerc, Nathan Arthur, Nhan, Ori Marash, RamenChef, Randy, Squazz, takeradi, Ted Goas, Timothy Morris, TylerH

48	きの	animuson, Harry, RamenChef, ScientiaEtVeritas
49		demonofthemist, FelipeAls, Madalina Taina, Nathan Arthur, RamenChef, vishak
50		Chris, mnoronha, RamenChef
51		4444, Ahmad Alfy, animuson, Asim K T, Ben Rhys-Lewis, Boris, CalvT, cdm, Charlie H, CocoaBean, Dan Devine, Dan Eastwell, Daniel G. Blázquez, Daniel Stradowski, Darthstroke, designcise, Devid Farinelli, dippas, fcalderan, FelipeAls, Goose, Horst Jahns, Hynes, Jack, Jacob Gray, James Taylor, John Slegers, Jon Chan, Jonathan Zúñiga, Kevin Montrose, Louis St-Amour, Madalina Taina, Maximillian Laumeister, Michael Moriarty, Mr. Alien, mtb, Nate, Nathan Arthur, Nhan, Persijn, Praveen Kumar, RamenChef, Richard Hamilton, ScientiaEtVeritas, Sergey Denisov, Shaggy, Sourav Ghosh, Stewartside, Stratboy, think123, Timothy, Trevor Clarke, TylerH, Zac, Zeta, Zze
52		andreas, animuson, Arjan Einbu, Brett DeWoody, Community, cuervoo, darrylyeo, designcise, H. Pauwelyn, Jasmin Solanki, John Slegers, Kuhan, Marc, Michael Moriarty, Miro, Nathan Arthur, niyasc, RamenChef, Richard Hamilton, ScientiaEtVeritas , SeinopSys, Stewartside, user007, Wolfgang, X-27
53	の	bipon, Sebastian Zartner
54		Arif, Casey, Chathuranga Jayanath, Daniel Nugent, FelipeAls, Madalina Taina, RamenChef, ScientiaEtVeritas
55	さ	4dgaurav, A B, animuson, Epodax, geeksal, J F, Marc, Milche Patern, Ortomala Lokni, RamenChef, Richard Hamilton, rmondesilva, Robert Koritnik, Robotnicka, ScientiaEtVeritas, StefanBob, Stewartside, Thomas Altmann, Toby, user2622348, vladdobra, Zakaria Acharki, zer00ne
56		animuson, Brett DeWoody, cone56, dodopok, H. Pauwelyn, haim770, jaredsk, khawarPK, Kobi, RamenChef, SeinopSys, TheGenie OfTruth, TylerH