CSS Animation
パフォーマンスを考える原理

tomixy (@tetracalibers)

株式会社ピクセルグリッド所属

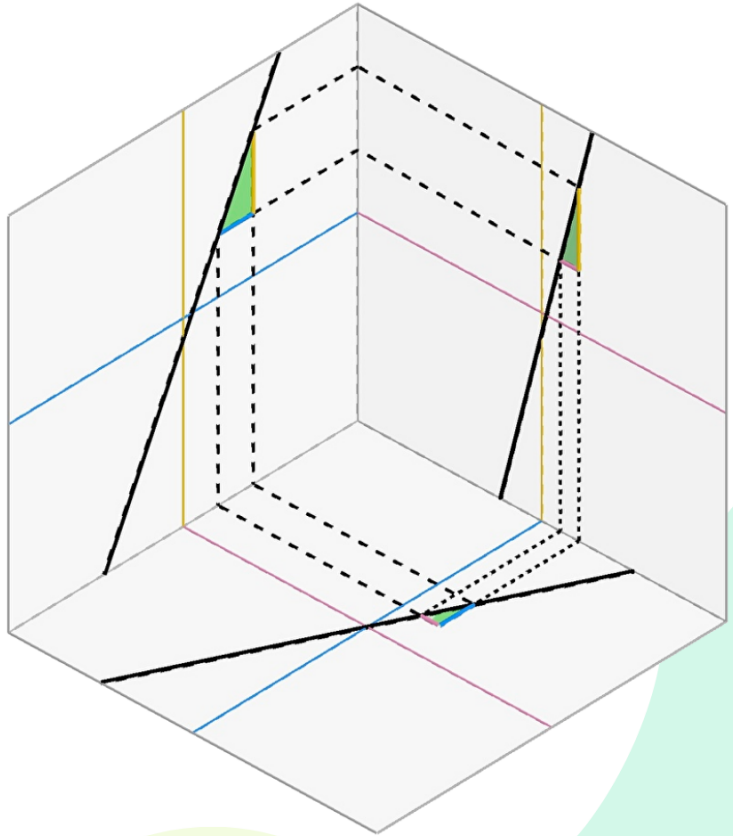
Biography

- 2002年生まれ
- tomixyに特に意味はない
- tetracalibersにも特に意味はない

最近の興味は

- SVG
- Web Assembly
- 画像処理

とか



Webページ ≒ デジタル絵画

アニメーションとは？

完成された絵画の一部を破壊し、

「壊れたところからやり直し」を短い時間で繰り返し行うこと

→ このプロパティ値を変化させたらどこまで絵画が壊れるのか？を想像できることが大事

そのためには、レンダリングエンジンがどのような手順で描画しているのかを知らなくてならない

CSSの描画手順

1. Layout

要素のボックスを並べる

ラフ画段階

2. Paint

各ボックス内を

飾りつける段階

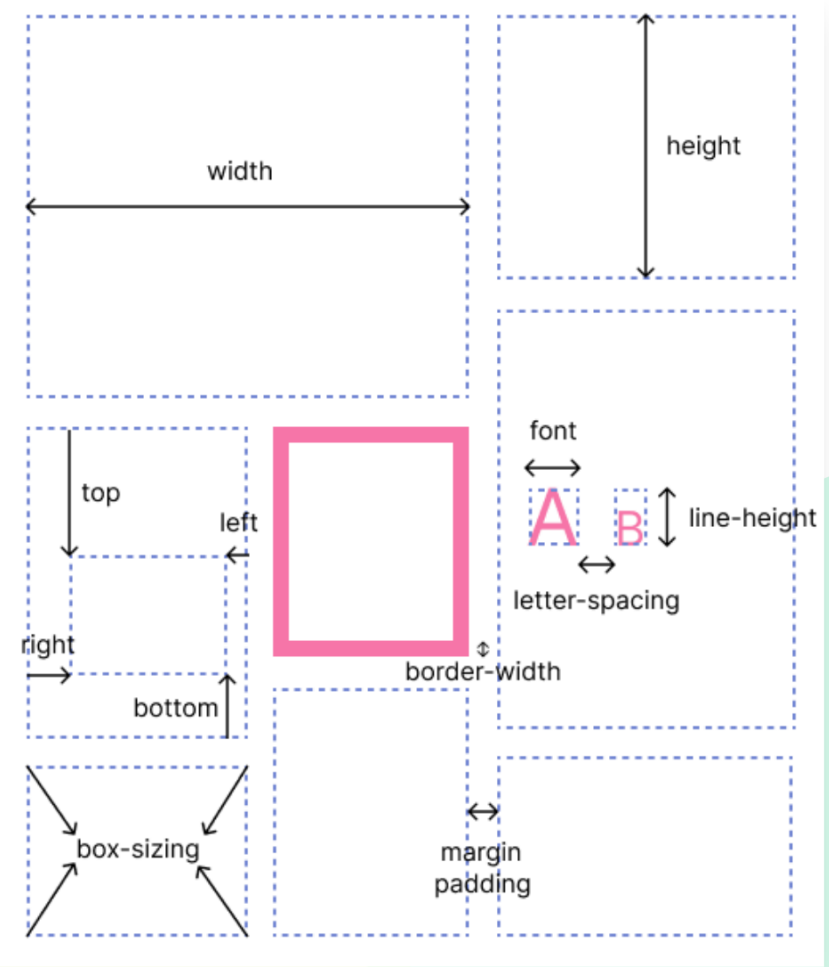
3. Composite

レイヤーを組み合わせる段階

1. Layout

要素のボックスを並べるラフ画段階

- size系プロパティ
  - `height` , `width` , `box-sizing`
- 余白系プロパティ
  - `padding` , `margin`
- 位置指定系プロパティ
  - `top` , `left` , `right` , `bottom`
- フォント系プロパティ
- レイアウト系プロパティ
  - `grid` , `flex` , `float` 関連
- 枠線の線種と太さ
  - `border-style` , `border-width` , `border-collapse`
  - `border-image-source`

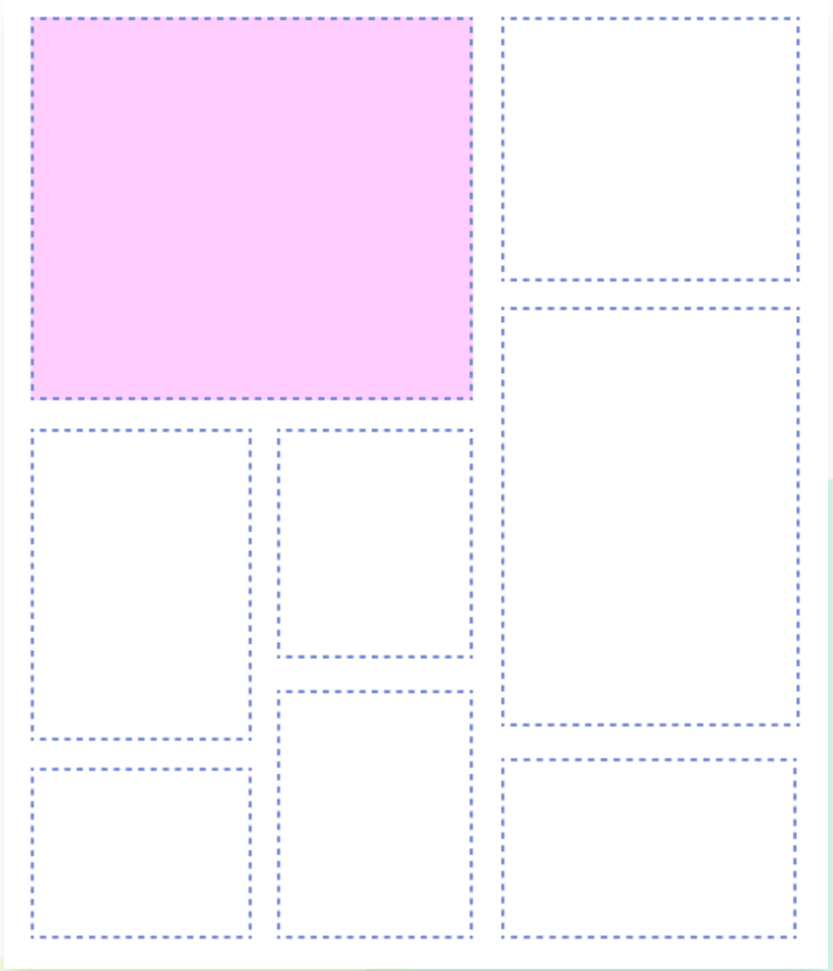


2. Paint

各ボックス内を飾り付けする段階

- color
- backface-visibility
- background-*
- border-color , border-image
- clip-path , border-radius
- box-shadow

etc.



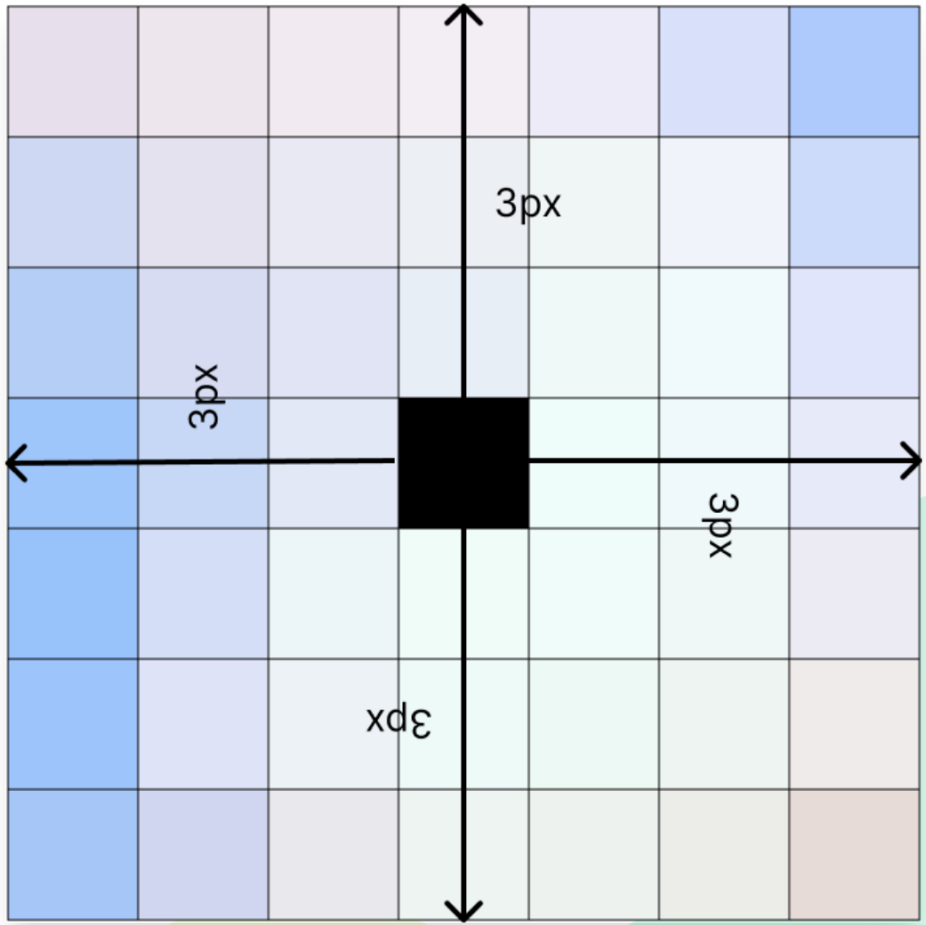
box-shadowには要注意

ぼかし半径によってパフォーマンスが変わる

ぼかし半径が3pxなら、
影を構成するピクセルごとに周囲3pxの色を混合する

- ぼかし半径が大きいほど重くなる
- `border-radius` と併用するとさらに重くなる
- SVGの `filter` や、CSSの `filter:blur` も同様

ぼかし半径を指定した影をアニメーションさせると、
処理がだんだん重くなってかくつく場合も
（特にSafari）



box-shadowのぼかし半径と角丸の影響

アニメーション中に発生した再レンダリング時間の合計

	Layout	Paint	Composite	TOTAL
box-shadow + border-radius	0	5669	1652	7321
box-shadow（ぼかし7px）	0	4536	2102	6638
box-shadow（ぼかし0px）	0	4290	1848	6138
opacity	0	148	640	788

解析スクリプトRepository

単位はすべてμs

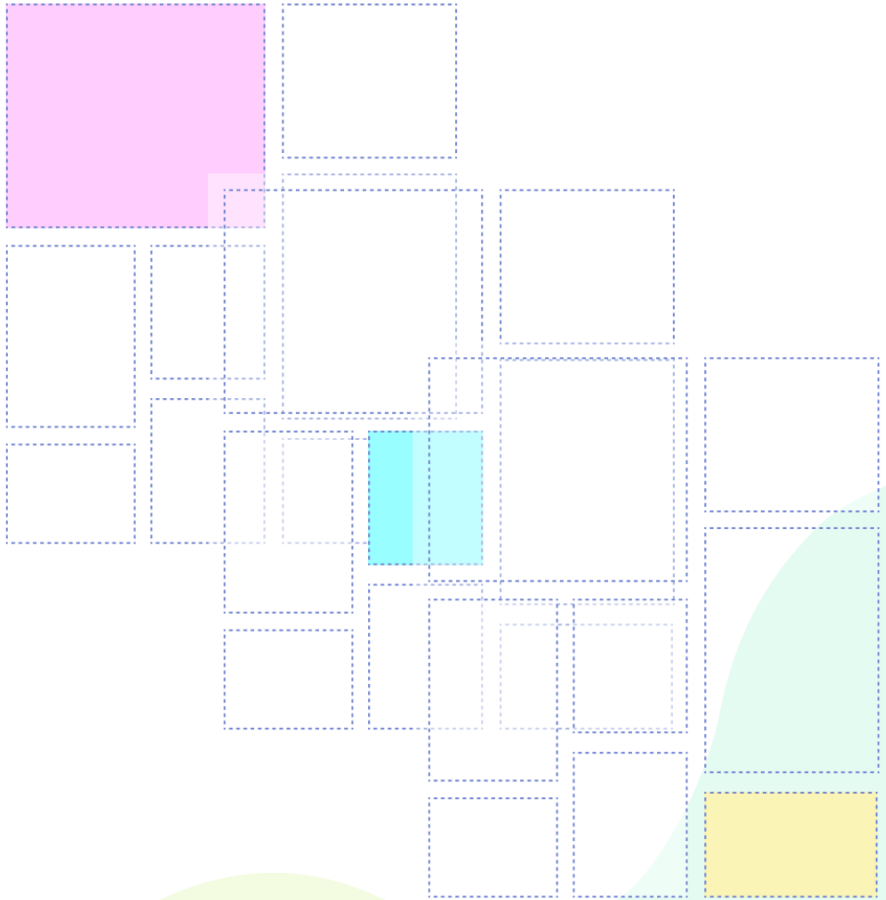
3. Composite

レイヤーを正しく重ねる

PaintLayerが生成される条件は

- `position` が明示されている
- `opacity` が1未満
- `filter` 指定あり
- `mask` 指定あり
- `mix-blend-mode` が `normal` 以外
- `transform` が `none` 以外
- `backface-visibility: hidden;`
- `overflow` が `visible` 以外

など（≒スタックコンテキストの生成条件）



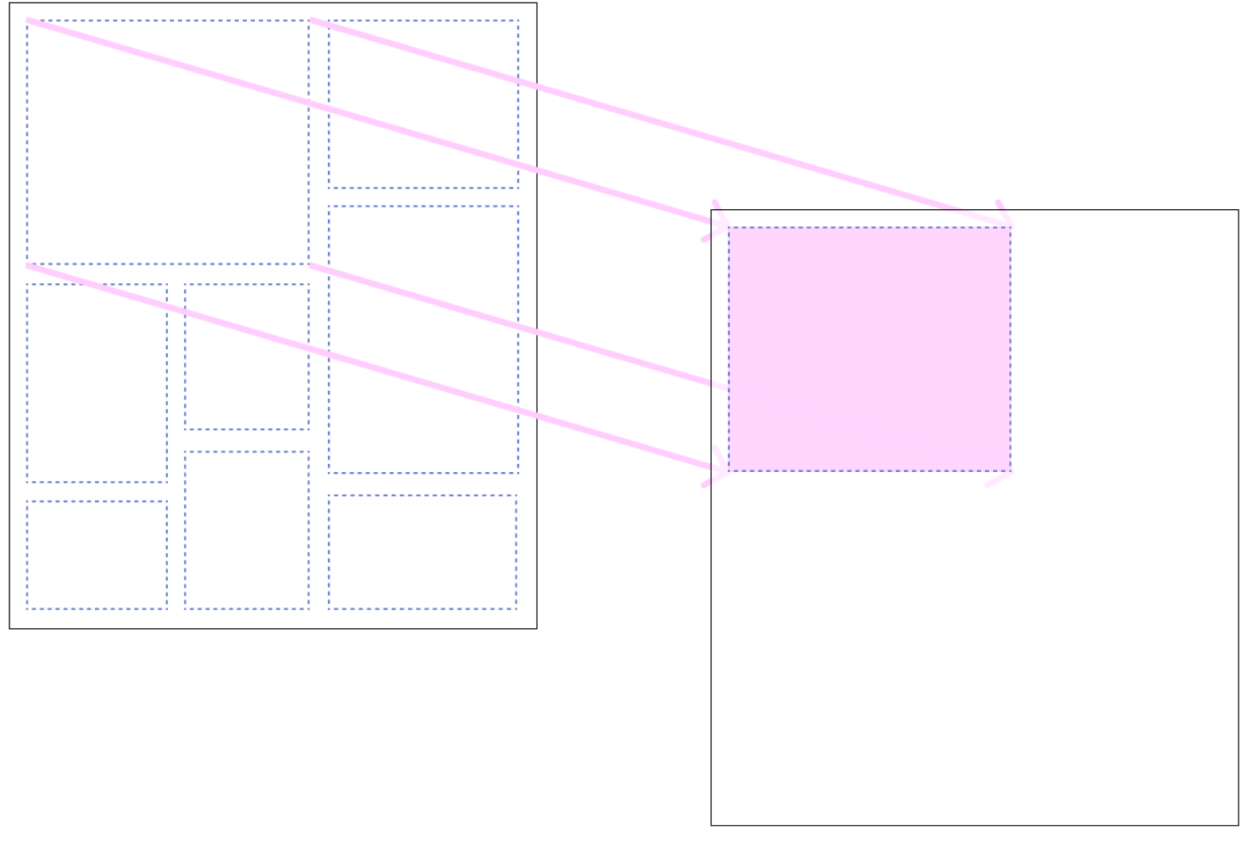
ハードウェアアクセラレーション

次のようなPaintLayerを**CompositeLayerに昇格させ、GPUに処理を委譲する**

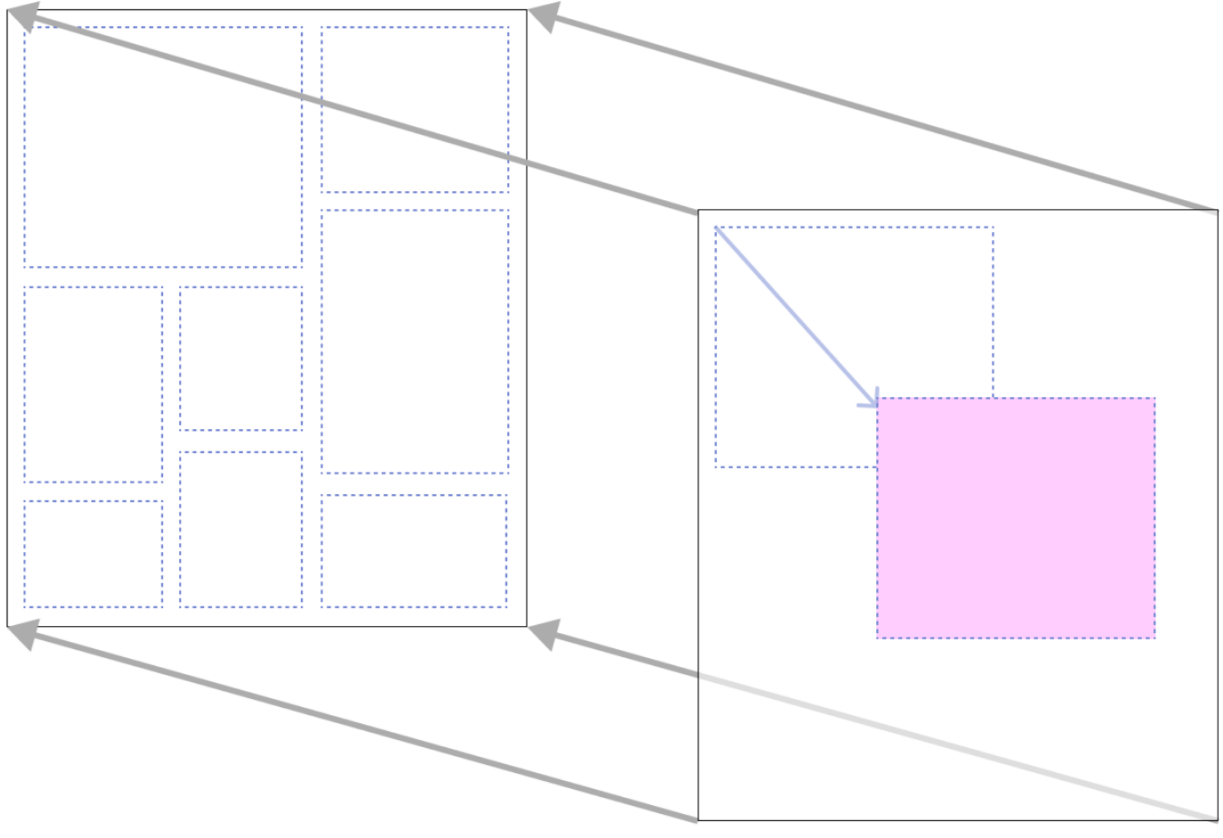
- 3Dの `transform` （Z指定、 `perspective` など）
- `backface-visibility: hidden;`
- `filter` , `backdrop-filter`
- `transform` 、 `opacity` （アニメーション中のみ）

etc.（ブラウザによって差異あり）

レイヤーに切り出して...

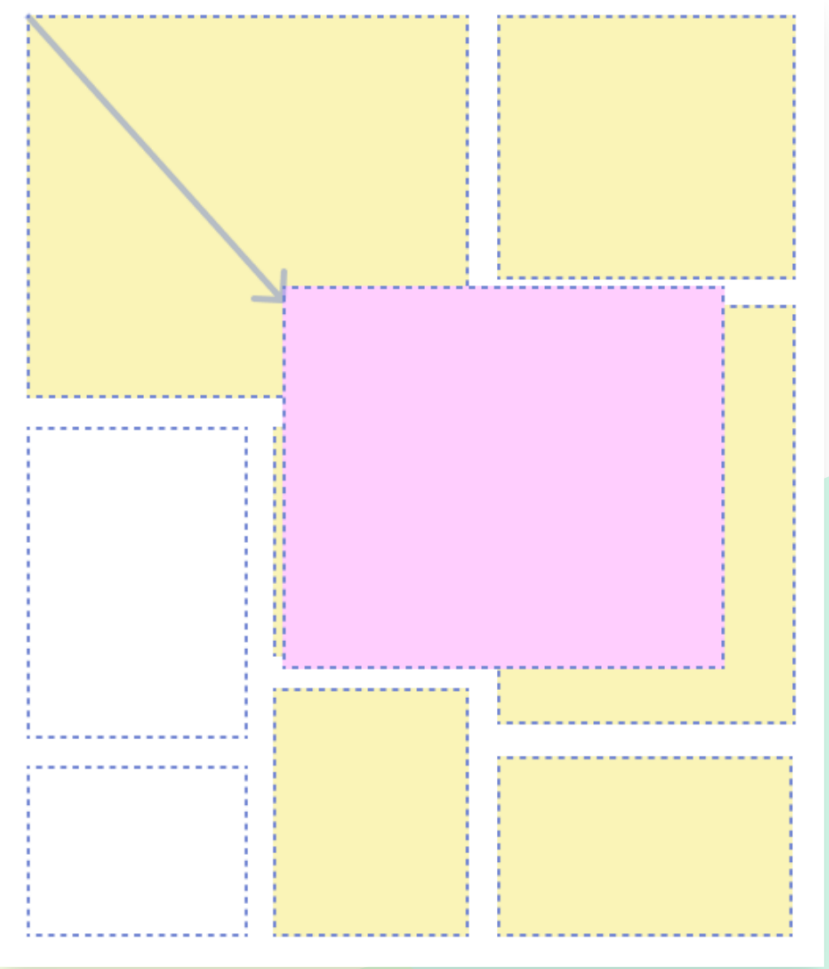


レイヤーをずらして合成する



もしLayoutを動かすと…

周囲の要素を壊してゴリゴリ進む



`translate` ではなく、 `top` や `left` を動かした場合…

- Layoutやり直し
- Paintやり直し（しかも周辺の要素も描き直し）
- 壊れた部分がGPUで管理している部分なら、描き直したレイヤーをGPUに再送信

ゴジラが建物をなぎ倒しながら進んでいくイメージ（？）

top / transform = 18.36...

アニメーション中に発生した再レンダリング時間の合計

	Layout	Paint	Composite	TOTAL
top	1327	2753	1080	5160
transform	0	107	174	281

解析スクリプトRepository

単位はすべてμs

レイヤーは画像

レイヤーの合成処理は軽いが、レイヤーの存在自体は重い

- 1ピクセルごとに、赤、緑、青、透明度の4バイトの情報を保持する
- レイヤーが増えるほど、その枚数分メモリを使う
- レイヤーに書き出された範囲が大きい（＝画像サイズが大きい）と、
  GPUに送信する処理も重くなる

iOS Chromeはむしろ**GPU処理で不安定になることもあった**（2023年1月に修正）

ハードウェアアクセラレーションの未来

> While the current CSS properties that are hardware-accelerated by default only include **opacity**, **filter**, and **transform** for now, **background-color** and **clip-path** will soon join the list.
>
> *Chromium - February 22, 2021*

clip-path

inset(<辺からの距離リスト> round <角丸リスト>)

none

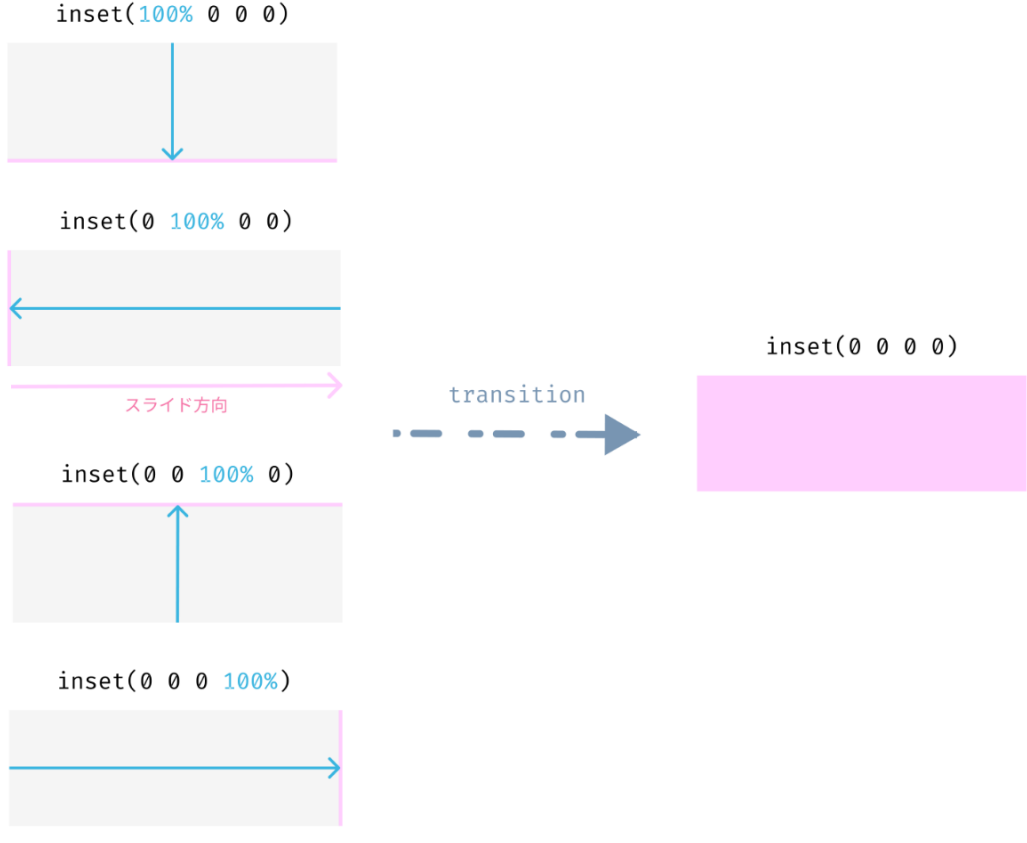
inset(1em)

inset(1em 1em 2em)

inset(1em 1em 2em round 0.75em)

inset(1em 1em 2em round 0.75em 0)

insetでスライド方向とは逆方向に押し潰しておく



スライドアニメーション（今はclip-pathは重め）

アニメーション中に発生した再レンダリング時間の合計

	Layout	Paint	Composite	TOTAL
width	3272	10563	3306	17141
clip-path	0	12127	2579	14706
background-position	0	5601	1768	7369
transform	0	0	1251	1251

解析スクリプトRepository

単位はすべてμs

border-radius: X;の代わりにinset(0 round X);

inset(0 round 0)

inset(0 round 100px)

filter:blurアニメーションと角丸

アニメーション中に発生した再レンダリング時間の合計

	Layout	Paint	Composite	TOTAL
blur(2px) + border-radius	0	701	4896	5597
blur(10px)	0	166	1926	2092
blur(2px) + clip-path角丸	0	163	1511	1674
blur(2px)	0	166	1310	1476

解析スクリプトRepository

単位はすべてμs

補足：border-radiusとclip-pathの違い

border-radius

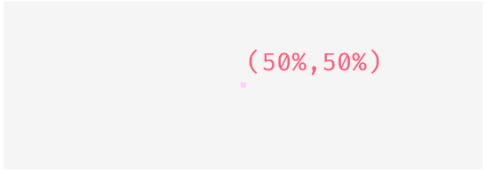
clip-path

polygon(x y, x y, ... , x y)

```css
img {
/* polygon(左上, 右上, 右下, 左下) */
  clip-path: polygon(
    1em 1em,
    calc(100% - 1em) 1em,
    calc(100% - 1em) 60%,
    1em calc(100% - 1em)
  );
}
```

polygonで点から塗りに遷移させる

polygon(50% 50%, 50% 50%, 50% 50%, 50% 50%)



transition

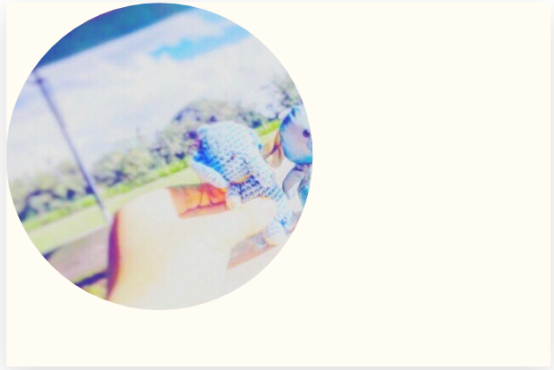
polygon(0 0, 100% 0, 100% 100%, 0 100%)

circle(<半径> at <中心点の位置>)

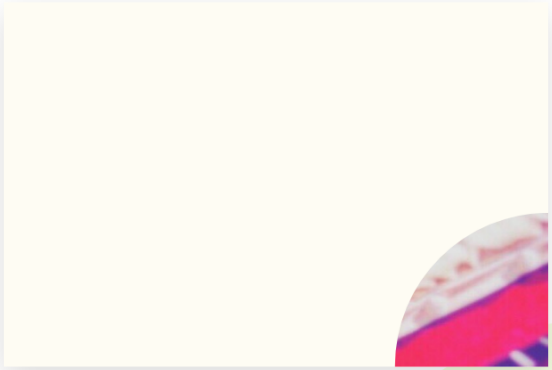
none

circle(4em at 50% 50%)

circle(4em at 0 0)



circle(4em at 4em 4em)



circle(4em at right bottom)

ellipse(<横の長さ> <縦の長さ> at <中心点の位置>)

none

ellipse(8em 6em at center)

clip-pathが苦手なアニメーション

要素自体の移動 translate

- clip-path は要素を内部で切り抜くだけなので、要素自体の位置は変えられない

回転 rotate

- 少しずつ頂点の位置をずらした polygon を多数 @keyframe に登録するしかない

剪断 skew

- 少しずつ頂点の位置をずらした polygon を多数 @keyframe に登録するしかない

子要素も含めて拡大縮小 scale

- inset を変えることで模倣はできるが、子要素は追従して拡大縮小されない

フェードイン/アウト opacity

- opacity でやるしかない

The future may be freer