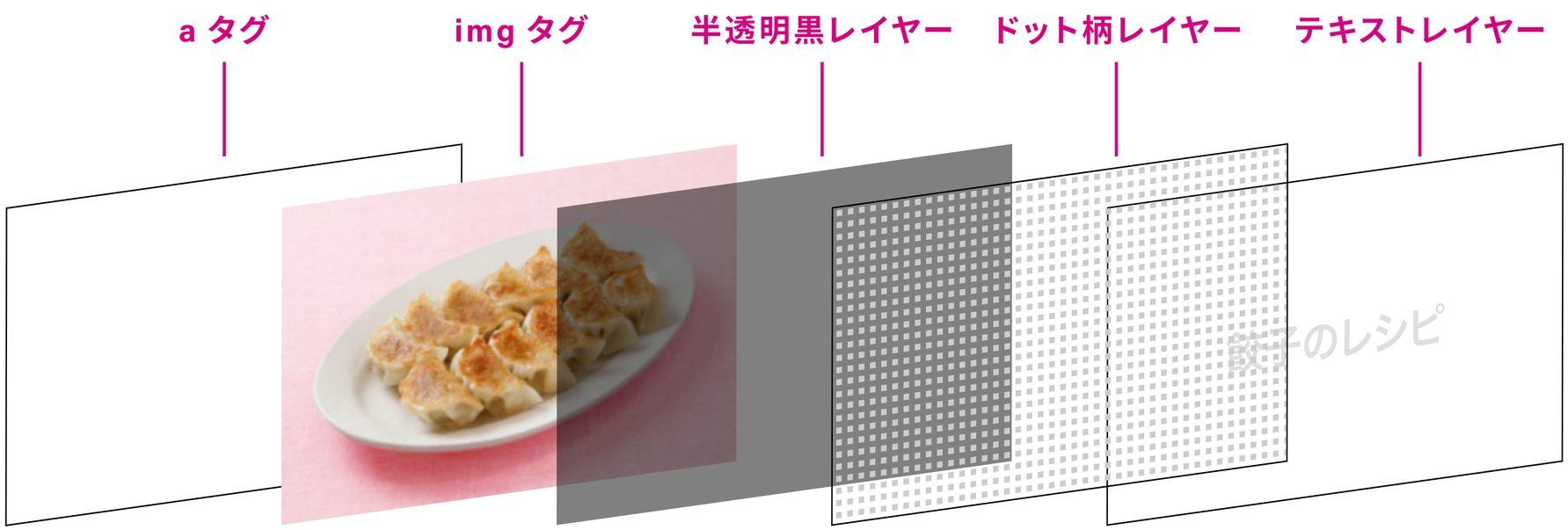
Web プログラミング

第十回：CSS(7)

CSS(7) - 「その他のレシピ」の実装

「その他のレシピ」の実装も、
パーツを分割し、レイアウト構成を考えましょう。
まずは、並んでいるボタンの１つだけを作ってみます。

CSS(7) - 「その他のレシピ」の実装



ボタンブロックの考え方は、ファーストビューの時とほぼ同じです。
a タグを display:block にして幅と高さを指定し、
その中に画像、半透明黒、ドット柄、テキストを重ね合わせる形でレイアウトします。

マウスオーバーで画像を動かすことを考慮し、
background-image ではなく、img タグを配置する形で実装します。

CSS(7) - 「その他のレシピ」の実装

ボタンブロックを作成できたら、
コピーして画像やテキスト内容を編集して複数作成します。
その後、全体を囲むブロックに display:flex を指定し、
レイアウトを調整して完了です。

CSS(7) -「擬似クラス」について

擬似クラスとは特殊な css のセレクタとして使えるクラスです。
「特定の条件にある時のみ有効になる」クラスとして使用できます。
セミコロン（:）の後に、擬似クラス名を書いて指定します。

CSS(7) -「擬似クラス」について

:hover 擬似クラスは
「マウスオーバーしている時」有効になる擬似クラスです。
これを使用すると、マウスオーバーした時に css を変化させたりできます。

リンク　　リンク

```
a{
    color: blue;
}
a:hover{
    color: red;
}
```

CSS(7) - 「擬似クラス」について

`a:hover{ ... }`	マウスオーバーした時に有効化
`a:visited{ ... }`	すでにクリックしたことのあるリンクのみに対して有効化
`input:focus{ ... }`	入力欄などをクリックしてアクティブにした時に有効化
`li:first-child{ ... }`	同じ階層にいる要素（兄弟要素）の中で、最初の一つ目に対して有効化
`li:last-child{ ... }`	同じ階層にいる要素（兄弟要素）の中で、最後の一つ目に対して有効化
`li:only-child{ ... }`	同じ階層にいる要素（兄弟要素）が無い時に有効化
`li:nth-child(odd){ ... }`	同じ階層にいる要素（兄弟要素）の中で、奇数番目の要素に対して有効化
`li:nth-child(even){ ... }`	同じ階層にいる要素（兄弟要素）の中で、偶数番目の要素に対して有効化

他にも色々あるので参考リンクなどから調べてみてください。

https://developer.mozilla.org/ja/docs/Web/CSS/Pseudo-classes

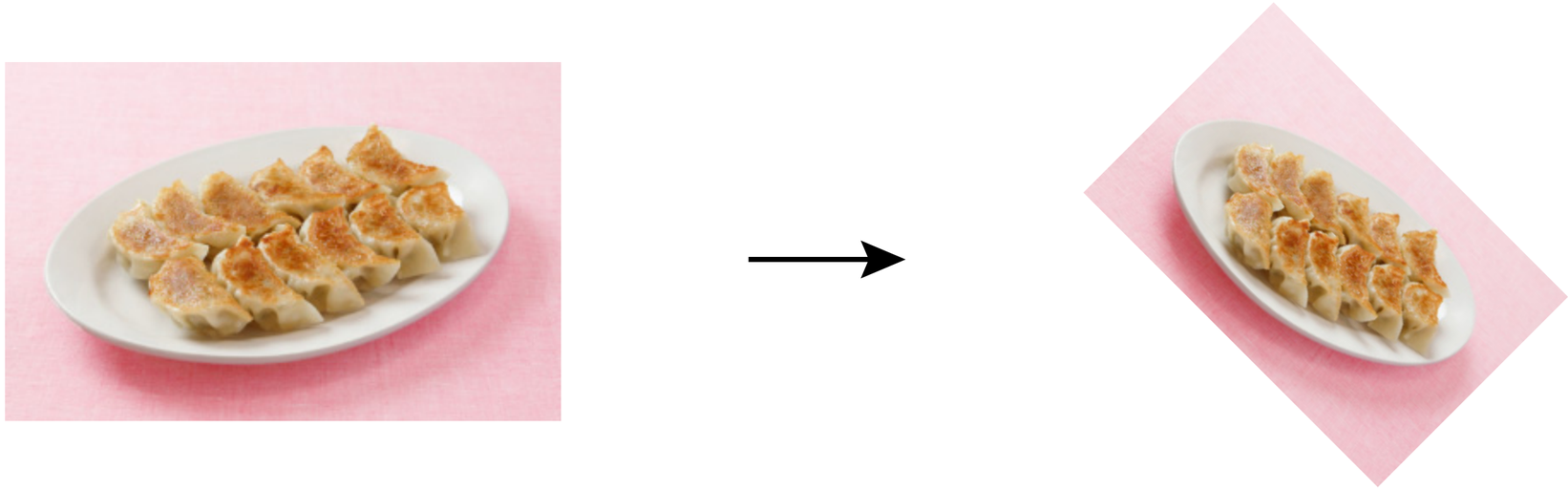
CSS(7) - 「擬似クラス」について

:hover 擬似クラスを使って、ボタンにマウスオーバーした時は
半透明黒レイヤーとドット柄レイヤーを透明（opacity:0）にしてみましょう


```css
.links_item:hover .links_blacklayer{
    opacity: 0;
}
.links_item:hover .links_dotlayer{
    opacity: 0;
}
```

CSS(7) - transform について

css の transform プロパティを使うと、
HTML 要素を様々な形に変形させることができます。



画像を 0.75 倍に縮小し、45 度回転させる例

```
img{
    transform: scale(0.75) rotate(45deg);
}
```

CSS(7) - transform について

transform プロパティは、様々な値を指定することができます。

`img{ transform: translate(10px, 10px) }`	移動（translate）- 右に 10px、下に 10px 移動させる
`img{ transform: scale(1.5, 1.5) }`	拡大（scale）- 横方向に 1.5 倍、縦方向に 1.5 倍のサイズに拡大する
`img{ transform: rotate(90deg) }`	回転（rotate）- 90 度回転させる
`img{ transform: skew(10deg, 10deg) }`	シアー変形（skew）- 水平方向に 10 度、垂直方向に 10 度歪ませる

平面的な変形だけでなく、立体的な変形をすることもできます。
非常に難しいので、講義では取り扱いません。
興味がある人は、以下の参考 URL などから調べてみてください。
https://developer.mozilla.org/ja/docs/Web/CSS/transform-function/translate3d
https://developer.mozilla.org/ja/docs/Web/CSS/transform-function/rotate3d
https://developer.mozilla.org/ja/docs/Web/CSS/transform-function/scale3d

CSS(7) - transform について

transform プロパティで変形するのは、見た目のみです。
レイアウト上のサイズは変化しないため、続きのフローレイアウトには影響しません。

餃子の画像餃子の画像餃子の画像餃子の画像餃子の画像

餃子の画像餃子の画像餃子の画像餃子の画像餃子の画像

餃子の画像餃子の画像餃子の画像餃子の画像餃子の画像

餃子の画像餃子の画像餃子の画像餃子の画像餃子の画像

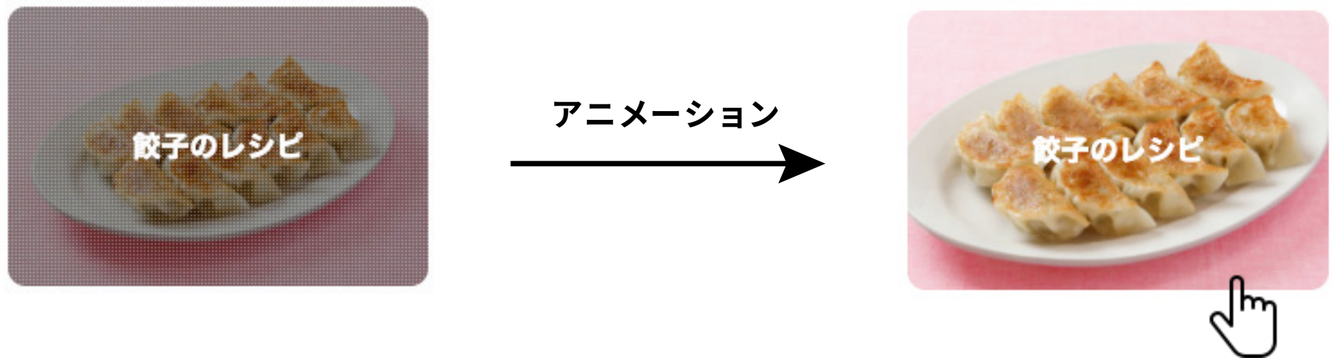
CSS(7) - transform について

:hover 擬似クラスを使って、マウスオーバーした時は
画像を少し拡大するようにしてみましょう

```
.links_item:hover .links_image{
    transform: scale(1.1);
}
```

CSS(7) - transition について

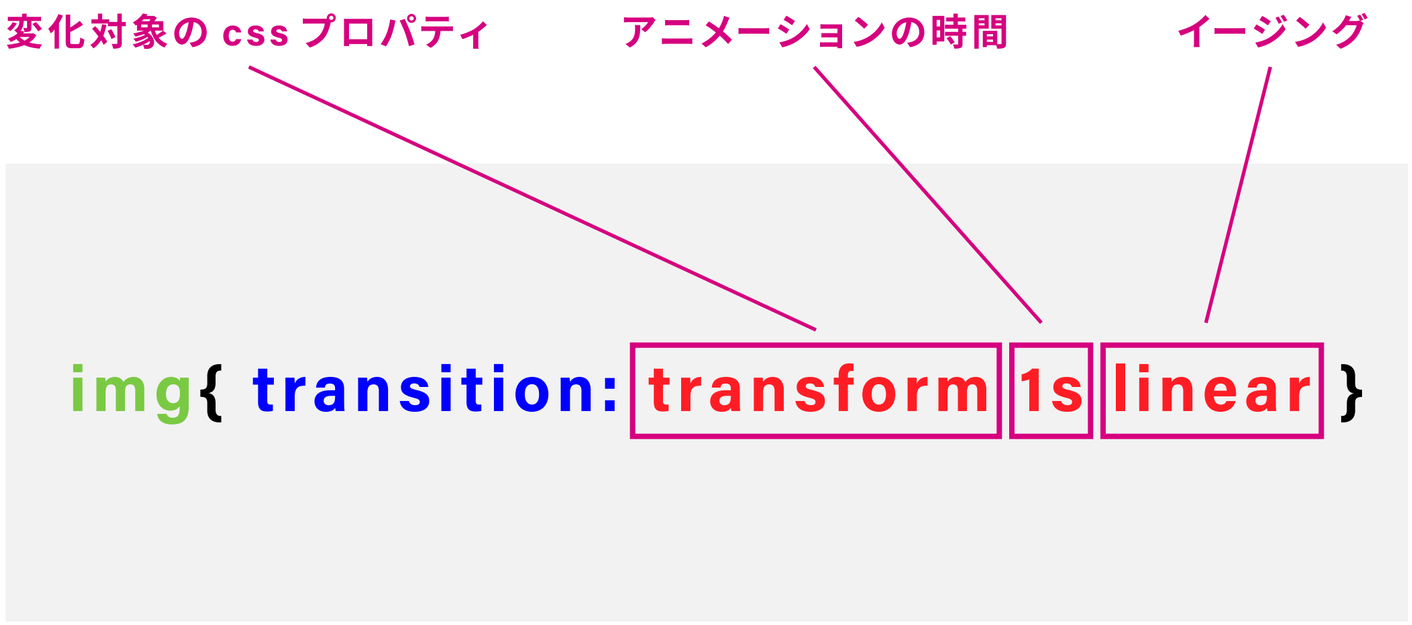
css の transition プロパティを使うと、
css の変化にアニメーションをつけることができます。
例えば「マウスオーバーしたら拡大する」など、css が変化する時、
その変わり方をアニメーションさせられるようになります。



transform と transition は名前が非常に似ているので注意しましょう。
変形させるのが「transform」
アニメーションをつけるのが「transition」です。

CSS(7) - transition について

transition プロパティは、値として
・対象の css プロパティ、
・アニメーション時間、
・イージング
の三つを指定します。



この場合、transform（変形）が、1 秒かけて、リニアにアニメーションする、という指定になる。

CSS(7) - transition について

値をカンマで区切って、複数の組み合わせを指定することができます。

```
img{
    transition: transform 1s linear, opacity .5s linear;
}
```

transform（変形）は、1 秒かけてリニアに、opacity（透明度）は 0.5 秒かけてリニアにアニメーションする

対象の css プロパティに「all」を指定すると、全ての css の変化にアニメーションがつきます。

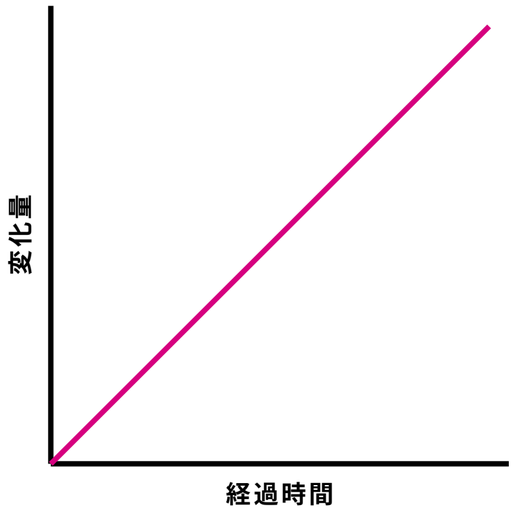
```
img{
    transition: all 0.5s linear;
}
```

全ての css プロパティが、0.5 秒かけてリニアにアニメーションする

CSS(7) - イージングについて

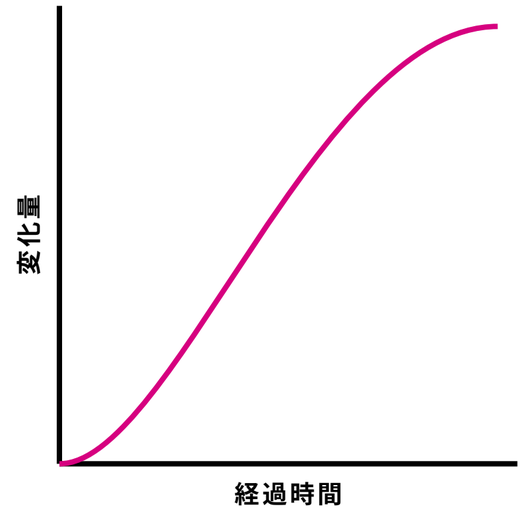
イージングとは、経過時間に対してどの程度値が変化するかを決めます。
「動きの加減速」のしかた、「動きのニュアンス」といった感じです。
linear, ease, ease-in, ease-out, ease-in-out の 5 種類のプリセットがあります。

linear



時間経過に比例して変化する
直線的な動き

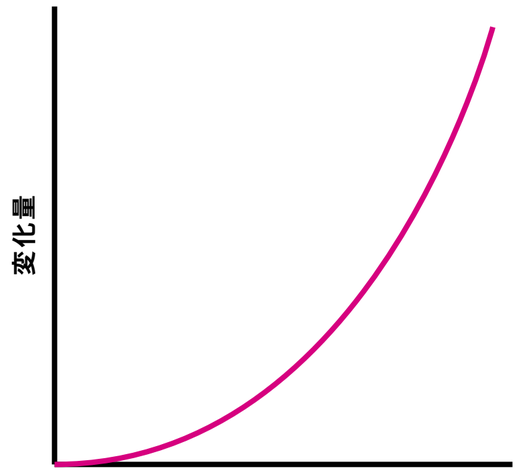
ease



ゆっくり始まり、
急激に加速し、
最後は徐々に遅くなること

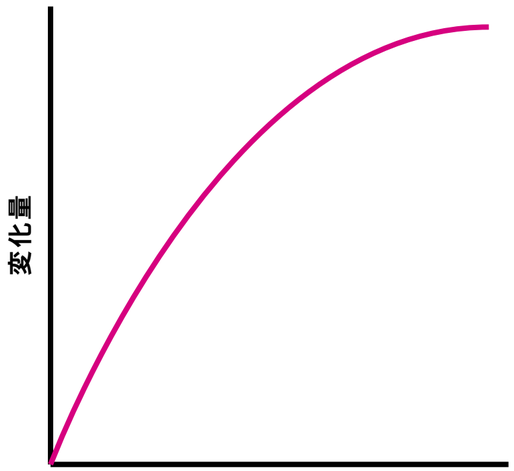
CSS(7) - イージングについて

ease-in



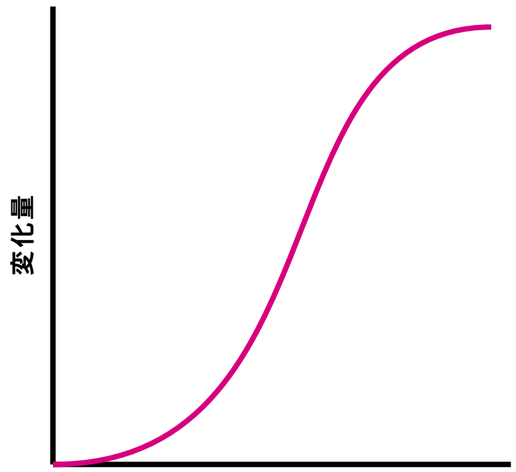
ゆっくり始まり、
次第に加速していき、
最後は急に止まる

ease-out



一気に始まり、
最後に近づくにつれて
徐々に遅くなる

ease-in-out



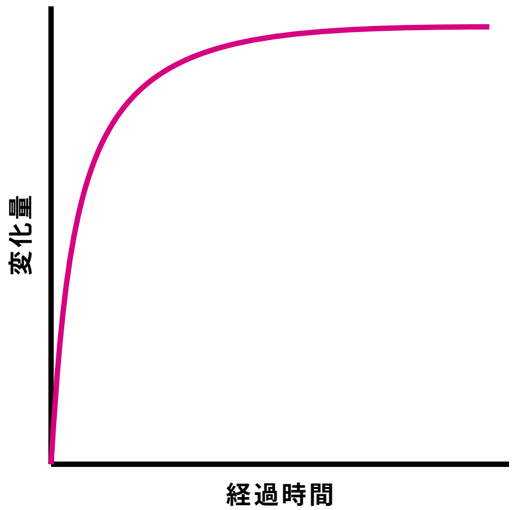
ゆっくり始まり、
加速し、
終わりに向かって減速する

マウスオーバーなど、インタラクションの動きをつける時は
ease-out がおすすめ。

CSS(7) - イージングについて

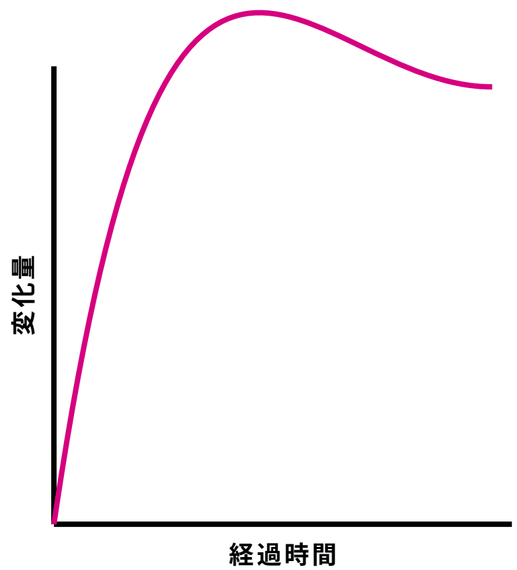
イージングの曲線は、「ベジエ曲線」で決められています。
cubic-bezier 関数を使って自ら定義することもできます。

cubic-bezier(0.16, 1, 0.3, 1)



開始後すぐに終点近くまで移動し、
最後はほぼ動かない

cubic-bezier(0.34, 1.56, 0.64, 1)

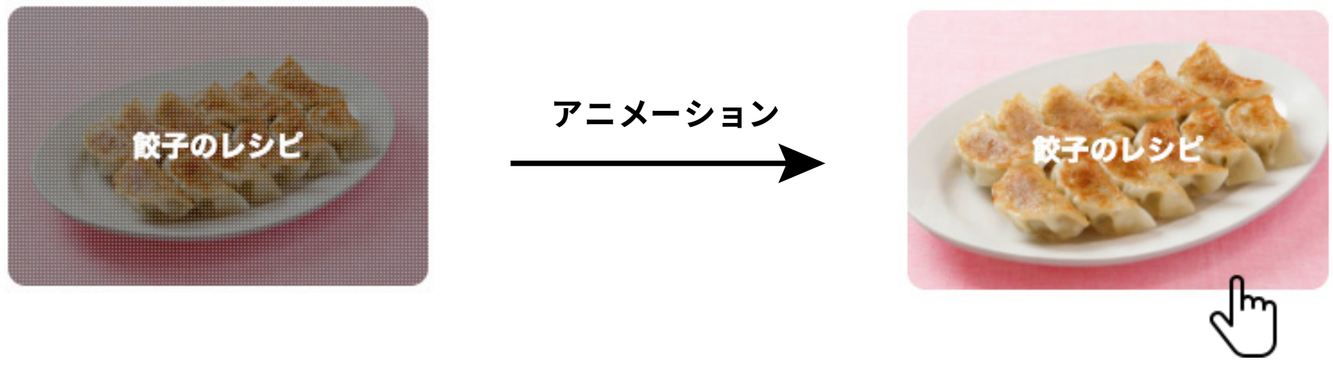


勢いよくスタートし、
はみ出すくらい進んだ後、
終点の位置にゆっくりもどる

参考：https://easings.net/ja

CSS(7) - transition について

:hover 擬似クラスで変化する要素に対して、
transition を設定して、アニメーションをつけましょう




```
.links_item .links_blacklayer{
    transition: opacity 0.1s linear;
}
.links_item .links_dotlayer{
    transition: opacity 0.1s linear;
}
.links_item .links_image{
    transition: transform 0.25s ease-out;
}
```

CSS(7) - 余談：animation

css の transition プロパティは、
「変化する css にアニメーションをつける」プロパティです。

それとは別に、アニメーションを定義する
「animation」という css プロパティがあります。

https://developer.mozilla.org/ja/docs/Web/CSS/animation

CSS(7) - 余談：animation

まず、keyframes を定義して動き方を決めます。

```
@keyframes slide-animation{
    0%   { left: 0%; }
    50%  { left: 100%; }
    100%{ left: 0%; }
}
```

最初（0%）は左にいて、
途中（50%）は左から 100% の位置（つまり右側）にいて、
最後は左にもどる、というアニメーションになります。

つまり、右側にスライドして左側に戻ってくる動きです。

CSS(7) - 余談：animation

次に、animation プロパティを使って
先ほど定義した動きをセレクタで選択した要素に適用します。

```
img.move-anime{
    animation: slide-animation 3s ease-in-out 0s infinite;
}
```

適用する keyframes 名
アニメーションの時間
イージング
ディレイ（開始を何秒遅らせるか）
繰り返し回数（infinite で無限）

これで、move-animation というクラス名がついた img 要素が、
3 秒間かけて左右に動くアニメーションを無限に繰り返すようになる