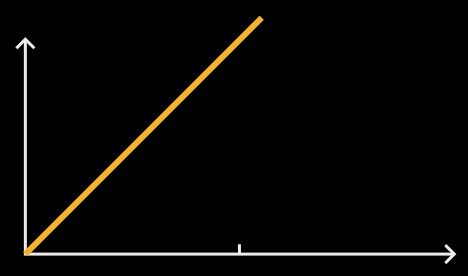
Timing Functions

a reference when creating animations in JavaScript

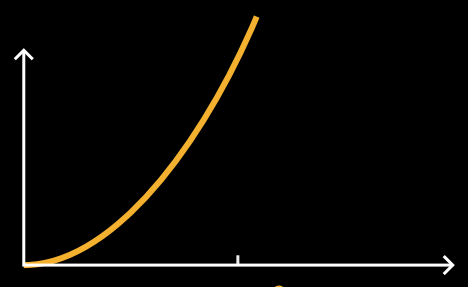
Linear



$f(t)=t$

```
function linear(t) {
  return t;
}
```

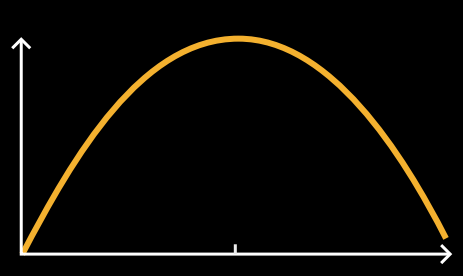
Ease-In
(Qudratic)



$f(t)=t^2$

```
function easeInQuad(t) {
  return t*t;
}
```

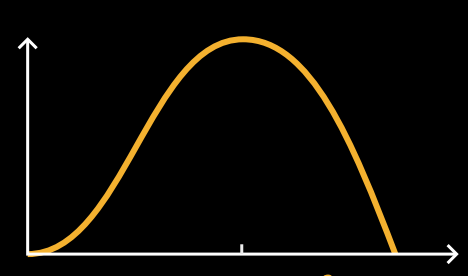
Ease-Out
(Qudratic)



$f(t)=t(2-t)$

```
function easeOutQuad(t) {
  return t*(2 - t);
}
```

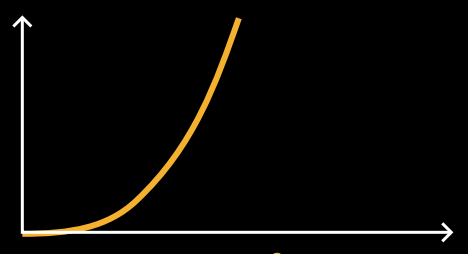
Ease-In-Out
(Qudratic)



For $t < 0.5: f(t) = 2t^2$
For $t \geq 0.5: f(t) = -1+(4-2t)t$

```
function easeInOutQuad(t) {
  return t < .5 ? 2*t*t : -1 + (4 -2*t)*t;
}
```

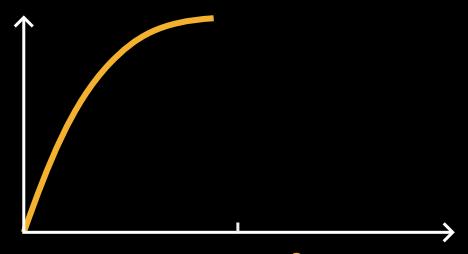
Ease-In
(Cubic)



$f(t)=t^3$

```
function easeInCubic(t) {
  return t*t*t;
}
```

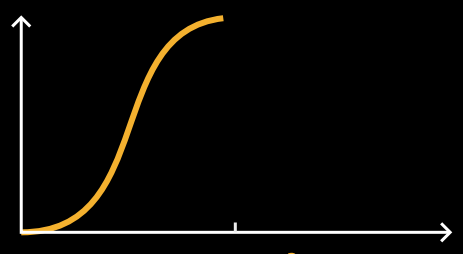
Ease-Out
(Cubic)



$f(t)=(t - 1)^3+1$

```
function easeOutCubic(t) {
  return (--t)*t*t+1;
}
```

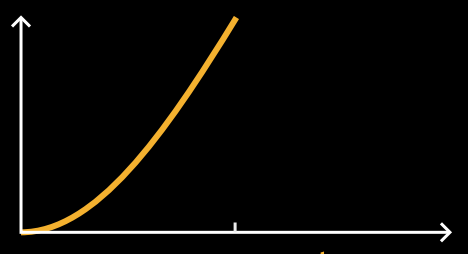
Ease-In-Out
(Cubic)



For $t < .5: f(t)=4t^3$
For $t \geq .5: f(t)=(t-1)(2t-2)^2+1$

```
function easeInOutCubic(t) {
  return t < .5 ? 4*t*t*t :
         (t-1)*(2*t-2)*(2*t-2)+1;
}
```

Ease-In
(Sine)



$f(t)=1-\cos(\frac{\pi t}{2})$

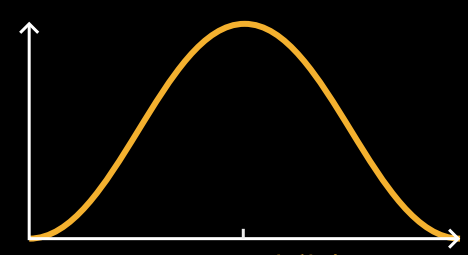
```
function easeInSine(t) {
  return 1 - Math.cos(t*Math.PI/2);
}
```

Ease-Out
(Sine)

$f(t)=\sin(\frac{\pi t}{2})$

```
function easeOutSine(t) {
  return Math.sin(t*Math.PI / 2);
}
```

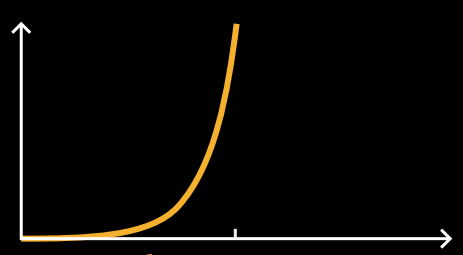
Ease-In-Out
(Sine)



$f(t)=-\frac{\cos(\pi t)-1}{2}$

```
function easeInOutSine(t) {
  return -(Math.cos(Math.PI*t)-1) / 2;
}
```

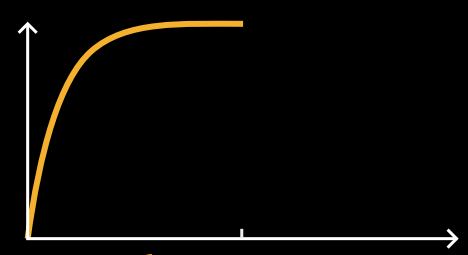
Ease-In
(Exponential)



$f(t)=\begin{cases}0 & \text{if } t = 0\\ 2^{10(t-1)} & \text{otherwise}\end{cases}$

```
function easeInExpo(t) {
  return t === 0 ?
  0 : Math.pow(2, 10 * (t - 1));
}
```

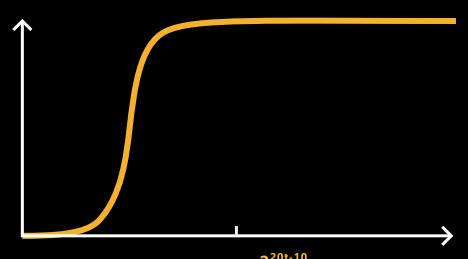
Ease-Out
(Exponential)



$f(t)=\begin{cases}1 & \text{if } t = 1\\ 1-2^{-10t} & \text{otherwise}\end{cases}$

```
function easeOutExpo(t) {
  return t === 1 ?
  1 : 1 - Math.pow(2, -10 * t);
}
```

Ease-In-Out
(Exponential)



For $t < .5: f(t)=\frac{2^{20t-10}}{2}$
For $t \geq .5: f(t)=\frac{2-2^{-20t+10}}{2}$

```
function easeInOutExpo(t) {
  if (t===0) return 0; if (t===1) return 1;
  return t<.5 ? Math.pow(2, 20*t - 10) / 2 :
  (2 - Math.pow(2, -20*t+10)) / 2
}
```

HE.DOT.DESIGN