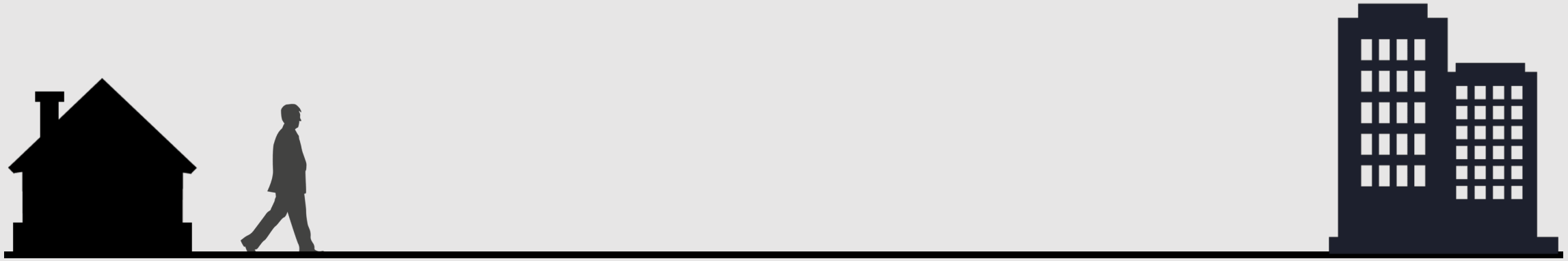
Basic Animations

- Animations: Overview
- Basic Animations: A CSS Approach
- Keyframing Animations



3 Properties define an animation

- Change in property (Δ)
- Time (t)
- Interpolation (I)

If two animation cycles share the same 3 properties, they are the same

Change in Property (Δ)

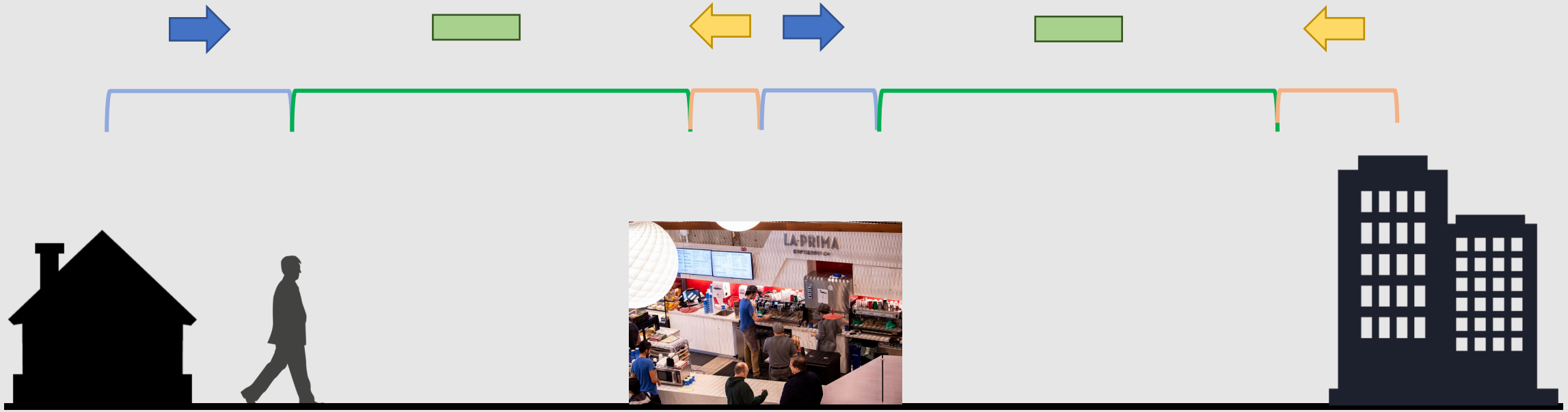
x-position: 0m;

x-position: 1000m;

Time (t)

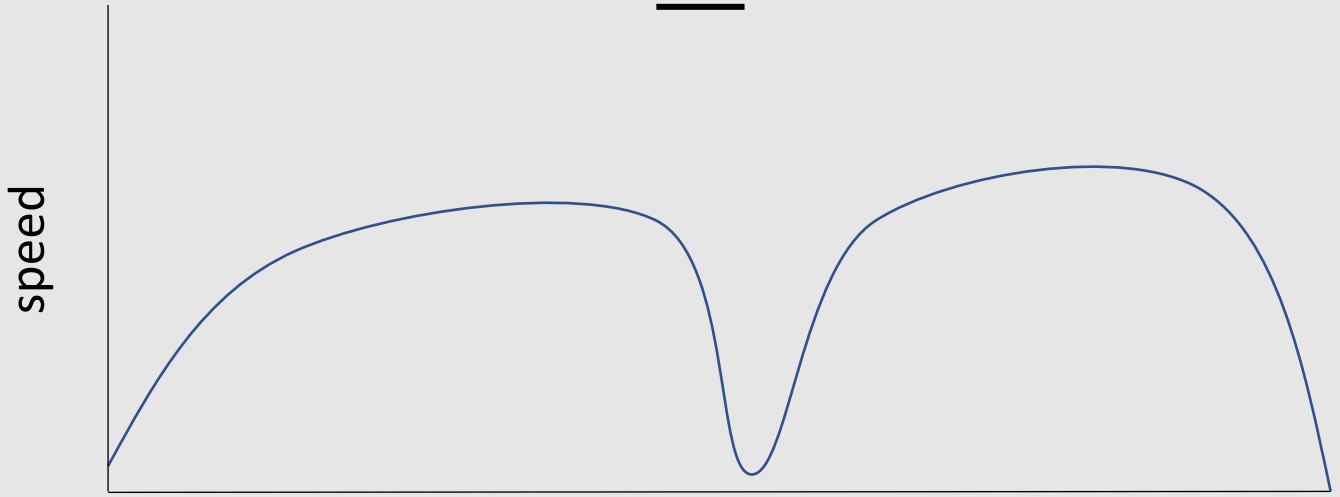
Takes 10 minutes to walk to work

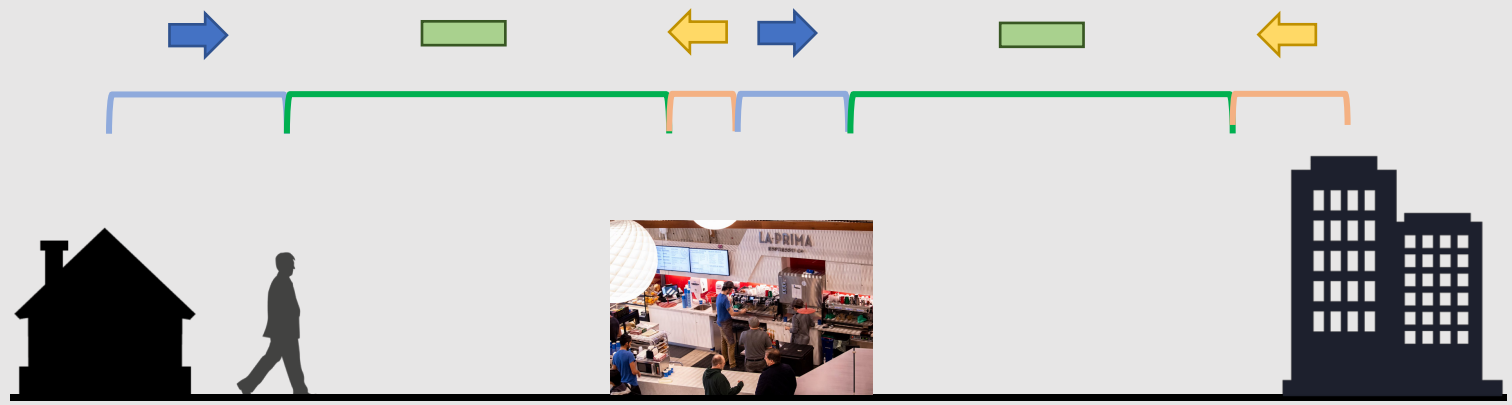
Interpolation (I)



Stops for a cup of coffee

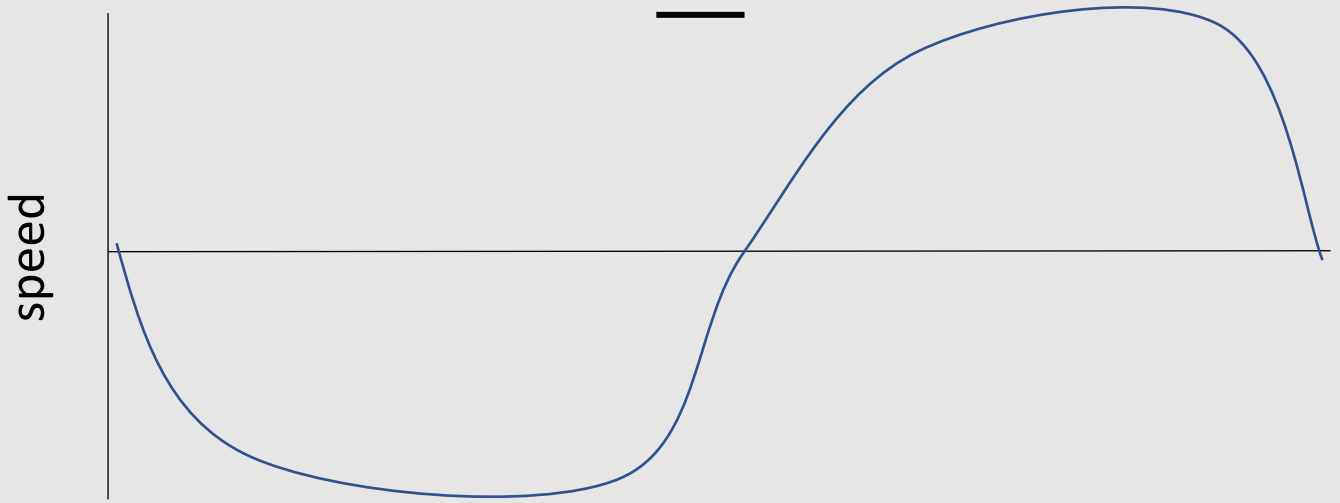
Interpolation (I)

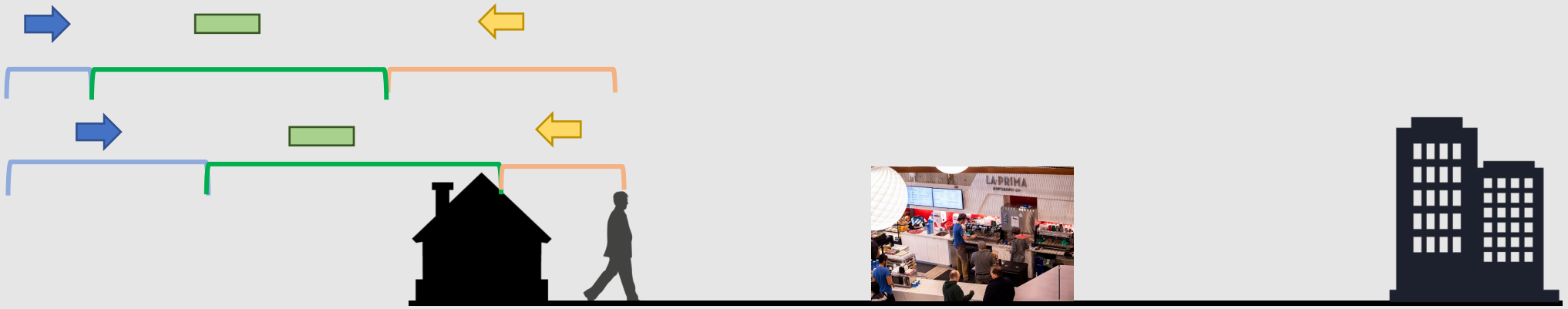




Interpolation tells us what the man does during the range of time **t**

Interpolation (I)

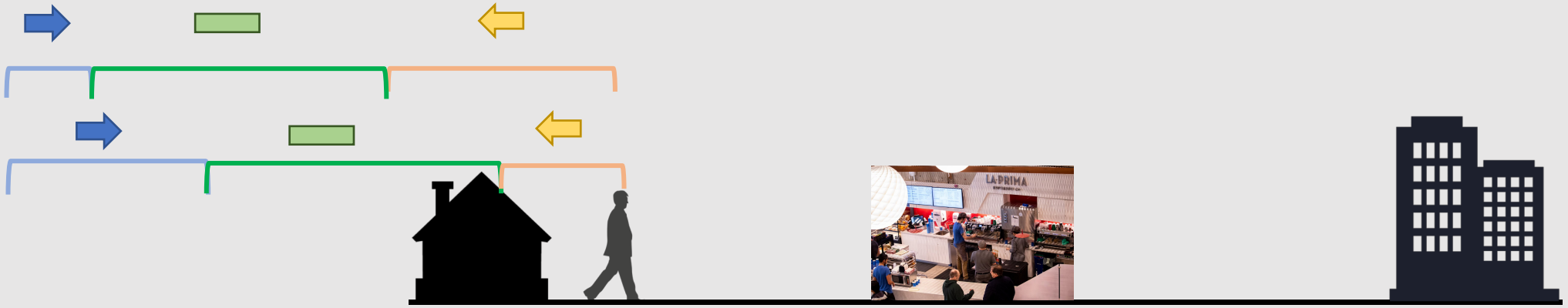




This is also possible too.

Interpolation (I)

Change in property (Δ)	None (We start and end in the same place)
Time (t)	10 mins
Interpolation (I)	sin(2*pi * t / 10)



- ~~Animations: Overview~~
- Basic Animations: A CSS Approach
- Keyframing Animations

Transition Property

```
.css-transition {
    width: 50%; }
```

```
.css-transition:hover {
    width: 100%;
    transition-property: width;
    transition-duration: 2s;
    transition-timing-function: linear;
    transition-delay: 1s; }
```

Change in property (Δ)	width += 50%
Time (t)	2 sec
Interpolation (I)	(50 / 2) (%/sec)

When we hover, the width increases linearly by 50% over 2 seconds with a 1 second delay.
When we unhover, the width decreases linearly by 50% over 2 seconds with a 1 second delay.

Transition Property

These two are equivalent.

```
.css-transition:hover {
    width: 100%;
    transition: width 2s linear 1s;
}
```

```
.css-transition:hover {
    width: 100%;
    transition-property: width;
    transition-duration: 2s;
    transition-timing-function: linear;
    transition-delay: 1s; }
```

Transition Property

We can have separate animation cycles for different values.

```
.css-transition {
    width: 50%;
    height: 50%;
    opacity: 0.5;
}
```

```
.css-transition:hover {
    width: 100%;
    height: 100%;
    opacity: 1;
    transition: width 2s linear 1s,
                height 2s linear 1s;
                opacity 3s cubic 0s;
}
```

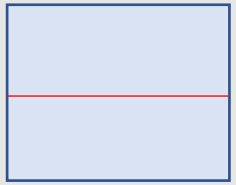
Or we can have all properties share the same interpolation.

```
.css-transition {
    width: 50%;
    height: 50%;
    opacity: 0.5;
}
```

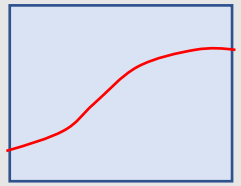
```
.css-transition:hover {
    width: 100%;
    height: 100%;
    opacity: 1;
    transition: all 2s linear 0s, }
```

Transition Timing Functions

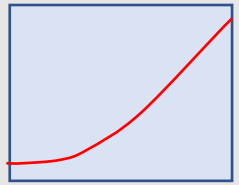
linear



ease

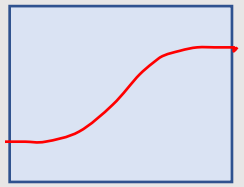


ease-in



ease-out

ease-in-out



Transition Timing Functions

Easing Demo

https://www.w3schools.com/css/tryit.asp?filename=trycss3_transition_speed

Transition Timing Functions

Can also specify custom easing properties

```
.css-transition:hover {
        width: 100%;
        transition: width 2s cubic-Bezier(.17,.67,.83,.67) 1s;
}
```

Easy Interface (Highly Recommend Bookmarking)

http://cubic-bezier.com/#.17,.67,.83,.67

- ~~Animations: Overview~~
- ~~Basic Animations: A CSS Approach~~
- Keyframing Animations

Keyframing Animations

Animation keyframes obey the following format:

```
@keyframes name {
    from {property: value;}
    to {property: value;) }
```

For multiple keyframes:

```
@keyframes name {
    0% {property: value;}
    25% {property: value;}
    50% {property: value;}
    75% {property: value;}
    100% {property: value;) }
```

Keyframing Animations

To bind keyframes to an event:

```
animation-name: name;
```

To set the duration of an animation:

```
animation-duration: 5s;
```

To set the interpolation of an animation:

```
animation-timing-function: linear;
```

To set the interpolation of an animation:

```
animation-delay: 1s;
```

Looping Animations

Define the direction the animation plays:

```
animation-direction: normal;            /* animation plays forwards */
animation-direction: reverse;           /* animation plays backwards */
animation-direction: alternate;         /* animation plays forwards then forwards */
animation-direction: alternate-reverse; /* animation plays backwards then forwards */
```

Define the number of iterations the animation plays:

```
animation-iteration-count: 5;           /* animation loops 5 times */
animation-iteration-count: infinite;    /* animation loops forever */
```

Setting Animations

These two are equivalent.

```
.css-transition:hover {
        animation-name: name;
        animation-duration: 5s;
        animation-timing-function: linear;
        animation-delay: 1s;
        animation-iteration-count: infinite;
        animation-direction: alternate; }
```

```
.css-transition:hover {
        animation: name 5s linear 1s infinite alternate; }
```

Setting Animations

What's the issue with this?

```
@keyframes cycle {
    0% {left: 10px;}
    50% {left: 70px;}
    100% {left: 50px;} }

.css-transition {
    left: 0px; }

.css-transition:hover {
    animation: cycle 2s linear 0s 1 alternate; }
```

Animation starts by snapping right 10px and ends snapping left 10px.

Animation Fill Mode

We can modify how animation cycles start and end:

```
animation-fill-mode: none;          /* animation reverts back to initial values */
animation-fill-mode: forwards;      /* animation holds on last keyframe */
animation-fill-mode: backwards;     /* animation reverts and holds on first keyframe */
```

Execution varies depending on animation direction.

Homework Ideas

- ❑ Play around with the Cubic Bezier site for custom interpolations.
- ❑ Use the transition property to create custom animations for elements.
- ❑ Use keyframe animations for more advanced interpolations.

Live Demo