기초웹개발론

kakao

http://bit.ly/2EE4HuJ

Animation Part.2

kakao

Transition

- HTML 요소의 스타일 속성을 이용하여 부드럽게 바꾸는 속성(변이, 전이)
- click 이나 hover 같은 사용자 액션이 필요함.
- display: none 일경우 적용되지 않음(visibility 는 가능)

Transition Property

속성값	설명
transition-property	transition 을 적용해야 하는 CSS 속성 이름
transition-delay	transition 효과가 나타나기전까지의 지연 시간을 설정
transition-duration	transition 효과가 지속될 시간을 설정함
transition-timing-function	transition 효과의 시간당 속도를 설정함
transition	모든 transition 속성을 이용한 스타일을 한줄에 설정할 수 있음

transition-property

- transition-property: all | none | <속성이름>
- 효과를 적용할 속성을 지정

```
.box {
  transition-property: background-color;
}
```

kakao

transition-delay

- transition-delay: <시간>
- 효과에 대한 지연 시간 설정

```
.box {
   transition-delay: 3s;
}
```

kakao

transition-duration

- transition-duration: <시간>
- 효과에 대한 진행 시간 설정
- s, ms (1s == 1000ms)

```
.box {
  transition-duration: 3s;
}
```

kakao

transition-timing-function

- transition-timing-function: ease | linear | ease-in | ease-out | ease-in-out | cubic-bezier(a,b,c,d)
- 중간값을 계산하는 방법을 정의.
- ease : 천천히 시작하고 빨라지다가 천천히 끝남(기본값)
- linear : 시작부터 끝까지 동일한 속도로 실행
- ease-in : 시작을 느리게
- ease-out : 끝을 느리게
- ease-in-out : 느리게 시작하고 느리게 끝남
- cubic-bezier : 함수를 직접 정의해서 사용. (https://cubic-bezier.com/)

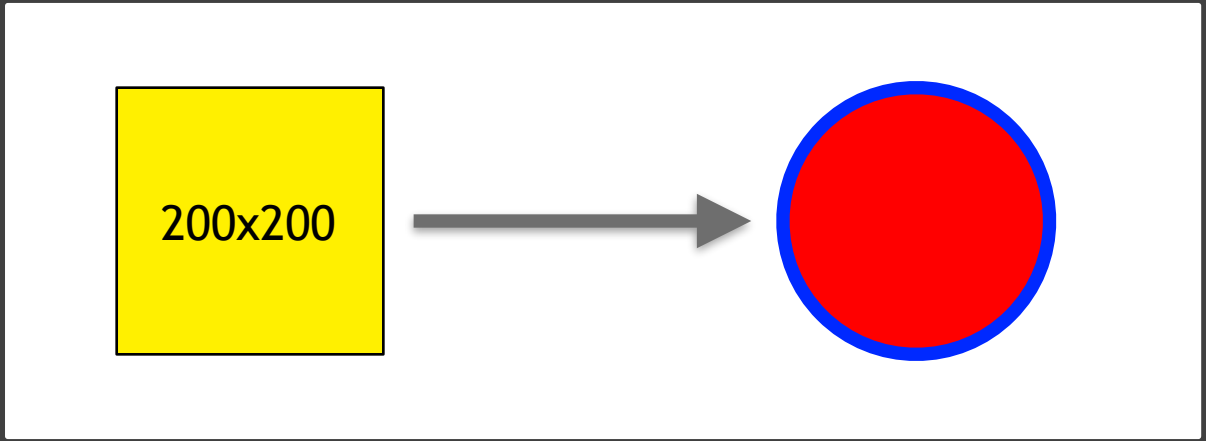
transition

- transition : <transition-property> | <transition-duration> | <transition-timing-function> | <transition-delay>

- transition 과 관련된 속성을 하나로 사용

```
.box {
  transition: all 3s ease 1s;
}
```

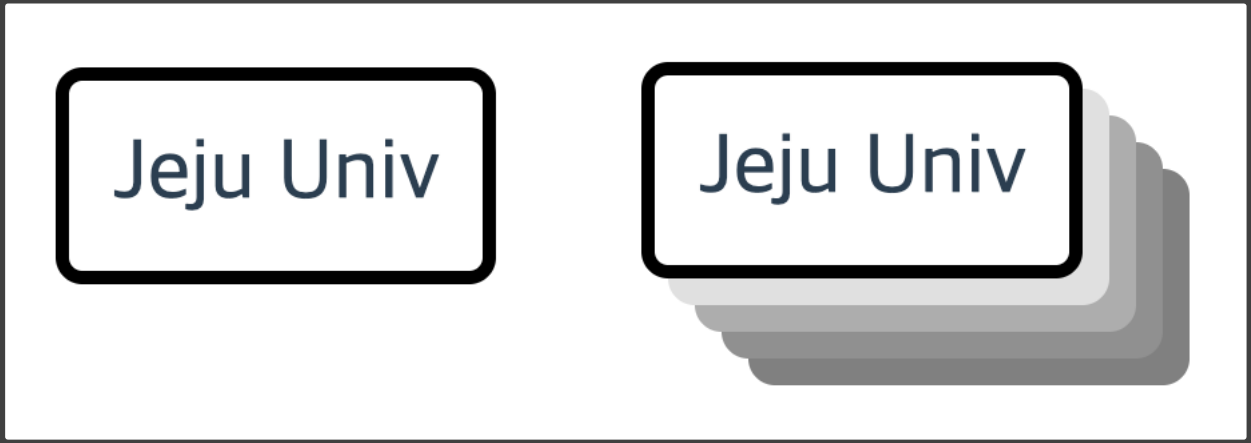
실습



실행시간 : 2초
배경색 : 노란색 -> 빨간색
테두리 : 1px 검정 실선 -> 5px 파랑 실선
timing-function : ease-in

kakao

실습



box : border 5px, radius 10px, padding 15px

실행시간 : 0.5초
box-shadow 이용 해서 그림자 4개 표현 (box-shadow 중첩)
timing-function : ease

kakao

Animation

- HTML 요소에 animation 을 적용.
- 별도의 사용자 액션이 필요하지 않으며 시작/정지/반복등을 제어할수 있음.
- @keyframes 을이용해 중간단계를 제어할수 있음.

kakao

@keyframes

- Animation 의 중간중간의 특정 지점을 거쳐 갈수 있는 키프레임.

```
@keyframes [name] {
   from {
      [styles]
   }
   to {
      [styles]
   }
}
```

```
@keyframes [name] {
   0% {
      [styles]
   }
   n% {
      [styles]
   }
   100% {
      [styles]
   }
}
```

animation-name

- animation-name : @keyframs

```
.container {
   animation-name: myAnimation
}

@keyframes myAnimation {
   from {
      [styles]
   }
   to {
      [styles]
   }
}
```

kakao

animation-duration

- animation-duration : @keyframs

```
.container {
   animation-name: myAnimation;
   animation-duration: 1s;
}

@keyframes myAnimation {
   from {
      [styles]
   }
   to {
      [styles]
   }
}
```

kakao

animation-timing-function

- animation-timing-function : linear | ease | ease-in | ease-out | cubic-bazier(n,n,n,n)

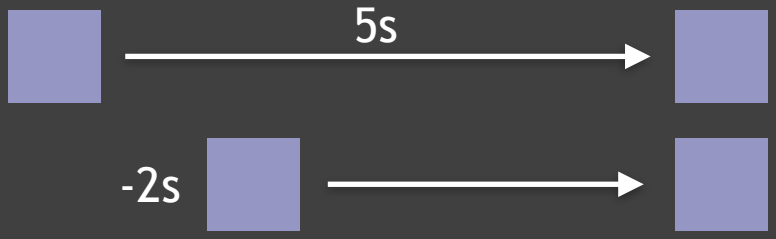
```
.container {
   animation-name: myAnimation;
   animation-duration: 1s;
   animation-timing-function: linear;
}

@keyframes myAnimation {
   …
}
```

animation-delay

- animation-delay : <time>

```
.container {
  animation-name: myAnimation;
  animation-duration: 1s;
  animation-timing-function: linear;
  animation-delay: 2s;
  /* animation-delay: -2s;*/
}
```



kakao

animation-iteration-count

- animation-iteration-count : <number> | infinite
- 기본값은 1
- infinite 는 무한반복

```
.container {
  animation-name: myAnimation;
  animation-duration: 1s;
  animation-timing-function: linear;
  animation-delay: 2s;
  animation-iteration-count: infinite;
}
```

kakao

animation-direction

- animation-direction : normal | reverse | alternate | alternate-reserve
- 반복되는 애니메이션의 방향
  - normal : 시작점에서 출발하여 끝점에서 끝나면 다시 시작점에서 출발(기본값)
  - reverse : 끝점에서 출발하여 시작점에서 끝나면 다시 끝점에서 출발
  - alternate : 시작점에서 출발하여 끝점에서 끝나면 끝점에서 다시 출발
  - alternate-reverse : 끝점에서 출발하여 시작점에서 끝나면 시작점에서 다시 출발

```
.container {
  animation-name: myAnimation;
  animation-duration: 1s;
  animation-timing-function: linear;
  animation-delay: 2s;
  animation-iteration-count: infinite;
  animation-direction: normal;
}
```

animation-fill-mode

- animation-fill-mode : none | forwards | backwards | both
- 반복이 아닌경우 애니메이션이 끝났을때의 위치
  - none : 시작 전까지 원래 위치에서 고정, 끝나면 그 위치로 다시 돌아감
  - forwards : 시작 전까지 원래 위치에서 고정, 요소가 애니메이션 끝나는 위치에서 멈춤(기본값)
  - backwards : 페이지가 로딩되면 곧장 애니메이션 시작 위치로 이동, 끝나면 원래 위치로 돌아감
  - both : 위의 두가지 효과 모두 적용

```
.container {
    animation-name: myAnimation;
    animation-duration: 1s;
    animation-timing-function: linear;
    animation-delay: 2s;
    animation-iteration-count: infinite;
    animation-direction: normal;
    animation-fill-mode: forwards;
}
```

animation-play-state

- animation-play-state : paused | running
- 애니메이션의 실행상태
  - paused : 정지
  - running : 진행(default)

```
.container {
    animation-name: myAnimation;
    animation-duration: 1s;
    animation-timing-function: linear;
    animation-delay: 2s;
    animation-iteration-count: infinite;
    animation-direction: normal;
    animation-fill-mode: forwards;
    animation-play-state: paused;
}
```

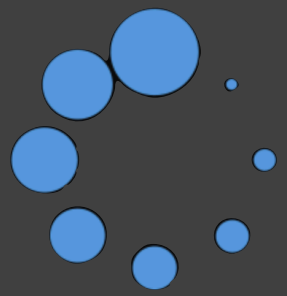
animation

- animation : name duration timing-function delay iteration-count direction fill-mode play-state
- 기본값 : none 0 ease 0 1 normal none running

```
.container {
    animation-name: myAnimation;
    animation-duration: 1s;
    animation-timing-function: linear;
    animation-delay: 2s;
    animation-iteration-count: infinite;
    animation-direction: normal;
    animation-fill-mode: forwards;
    animation-play-state: paused;

    animation: myAnimation 1s linear 2s infinite normal forwards paused;
}
```

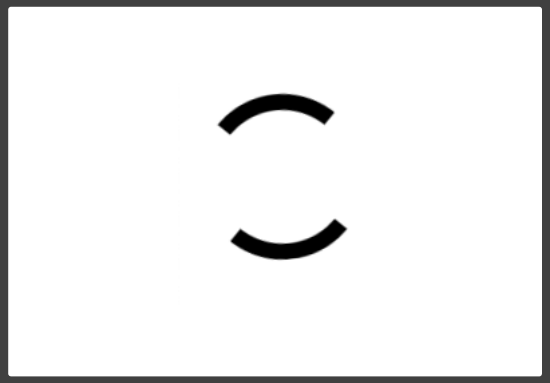
실습



spin.html
시간 : 1초
반복설정

kakao

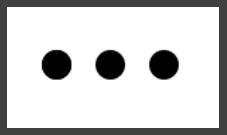
실습



spin2.html
시간 : 1초
반복설정

kakao

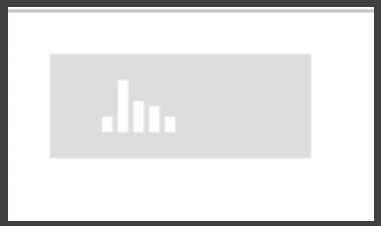
실습



spin3.html
시간 : 0.6초
left: 20px 이동
반복설정

kakao

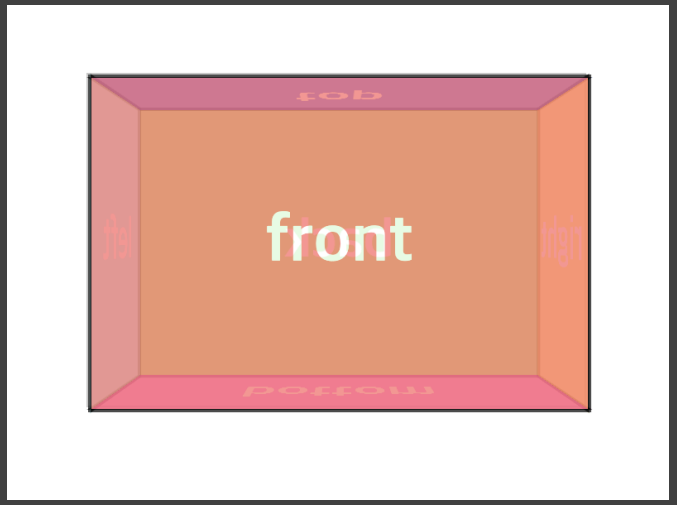
실습



equlize.html
시간 : 1초
반복설정

kakao

실습



box.html
시간 : 10초
X축으로 360
Y축으로 360
반복설정

kakao