

GSAP: Искусство Анимации

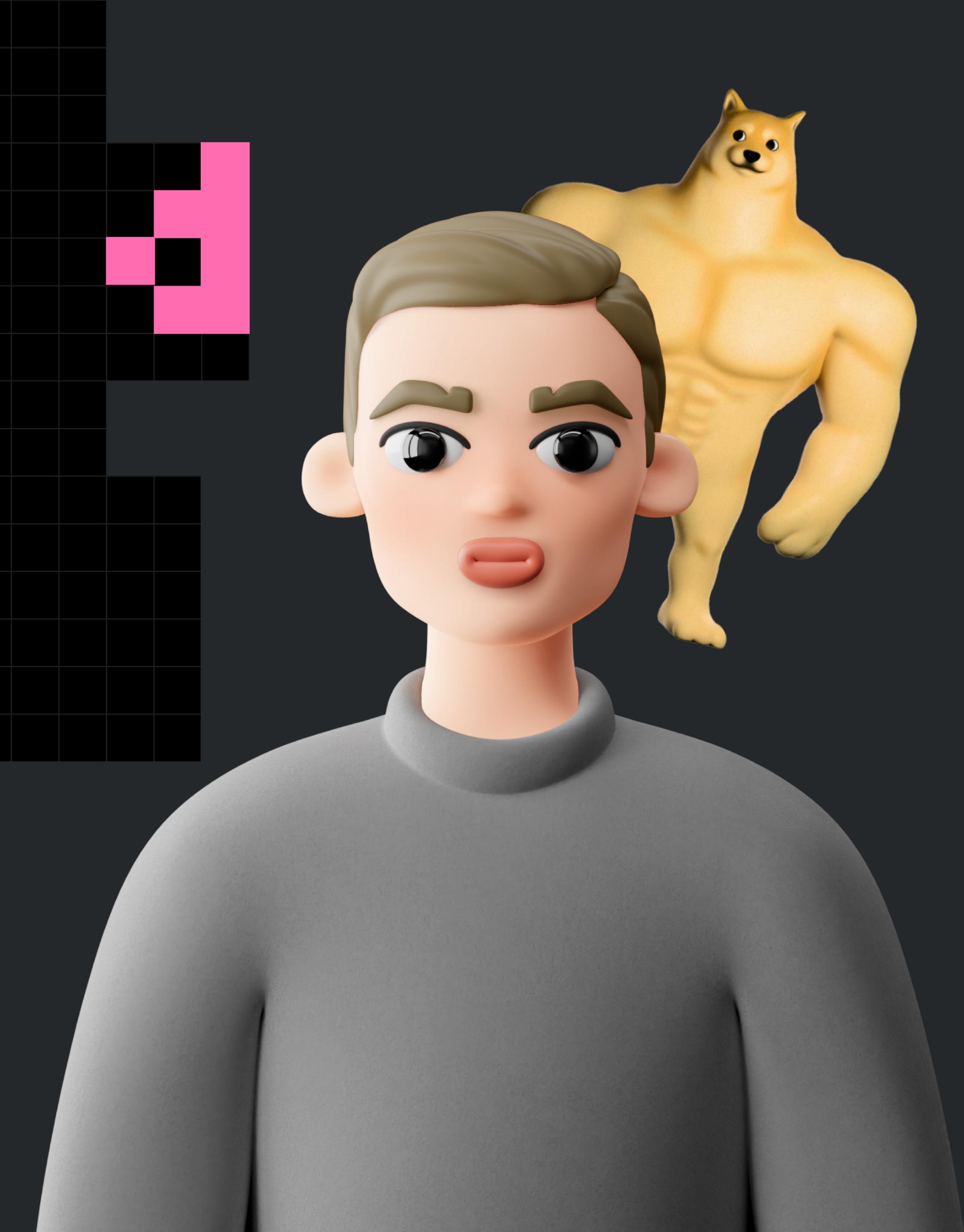
Возможности, примеры
и практическое руководство



Евгений
Дорошкевич
Frontend Developer

Who am I?

— Более 3-ёх лет во Frontend



Who am I?

- Более 3-ёх лет во Frontend
- 1.5+ года работаю с GSAP

GSAP



Kodix: чем мы занимаемся?

- Проектная разработка

— Крупнейшие
компании авто-отрасли



Kodix: чем мы занимаемся?

- Проектная разработка
- Крупнейшие компании авто-отрасли
- 100 сотрудников в штате



Kodix: чем мы занимаемся?

- Проектная разработка
- Крупнейшие компании авто-отрасли
- 100 сотрудников в штате
- 20 проектов в год



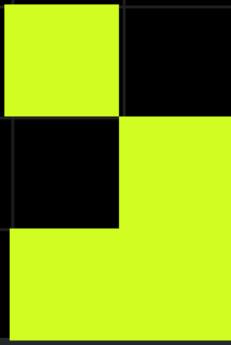
Kodix: чем мы занимаемся?

- Проектная разработка
- Крупнейшие компании авто-отрасли
- 100 сотрудников в штате
- 20 проектов в год
- 16 лет в индустрии



GSAP

Какую проблему решает?



Упрощение работы с анимацией
и предоставление интуитивно понятного
API и широкого набора функций



GSAP поддерживает анимацию практически всего:

- CSS-свойств (включая трансформации и фильтры)

GSAP поддерживает анимацию практически всего:

- CSS-свойств (включая трансформации и фильтры)
- SVG-графики и её свойств (масштаб, заливка, обводка и т. д.)

GSAP поддерживает анимацию практически всего:

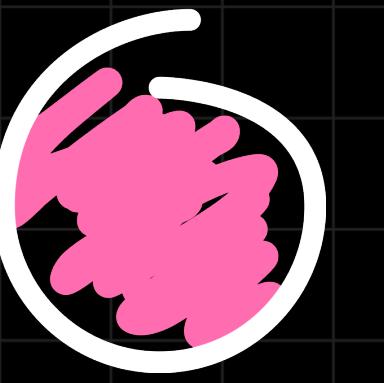
- CSS-свойств (включая трансформации и фильтры)
- SVG-графики и её свойств (масштаб, заливка, обводка и т. д.)
- даже произвольных JavaScript-значений

План

- 1** Анимация CSS свойств
- 2** timeline
- 3** ScrollTrigger
- 4** SplitText
- 5** Draggable
- 6** MorphSVG
- 7** ScrambleText

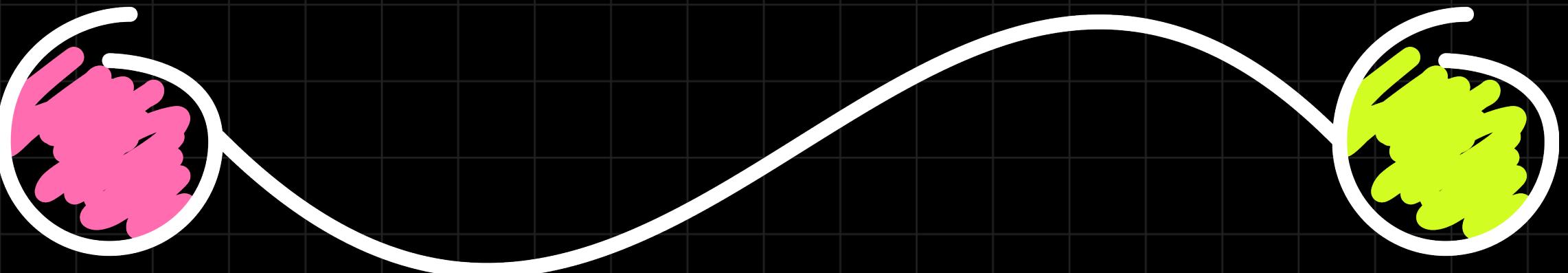
Как создается любая анимация?

1. Обозначаем состояние в начальной точке



Как создается любая анимация?

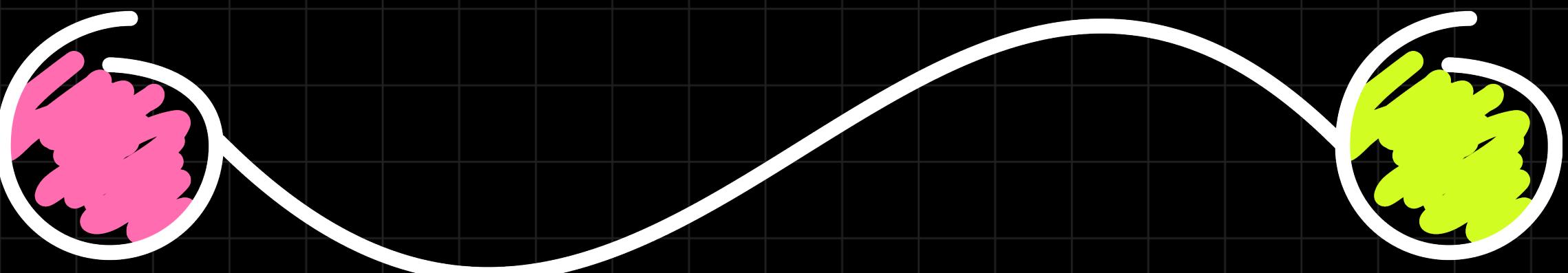
1. Обозначаем состояние в начальной точке
2. Обозначаем состояние в конечной точке



Как создается любая анимация?

1. Обозначаем состояние в начальной точке
2. Обозначаем состояние в конечной точке

Совокупность этих двух состояний
называется **такт анимации**



Анимация CSS свойств

Пример анимации opacity

Пример анимации translate

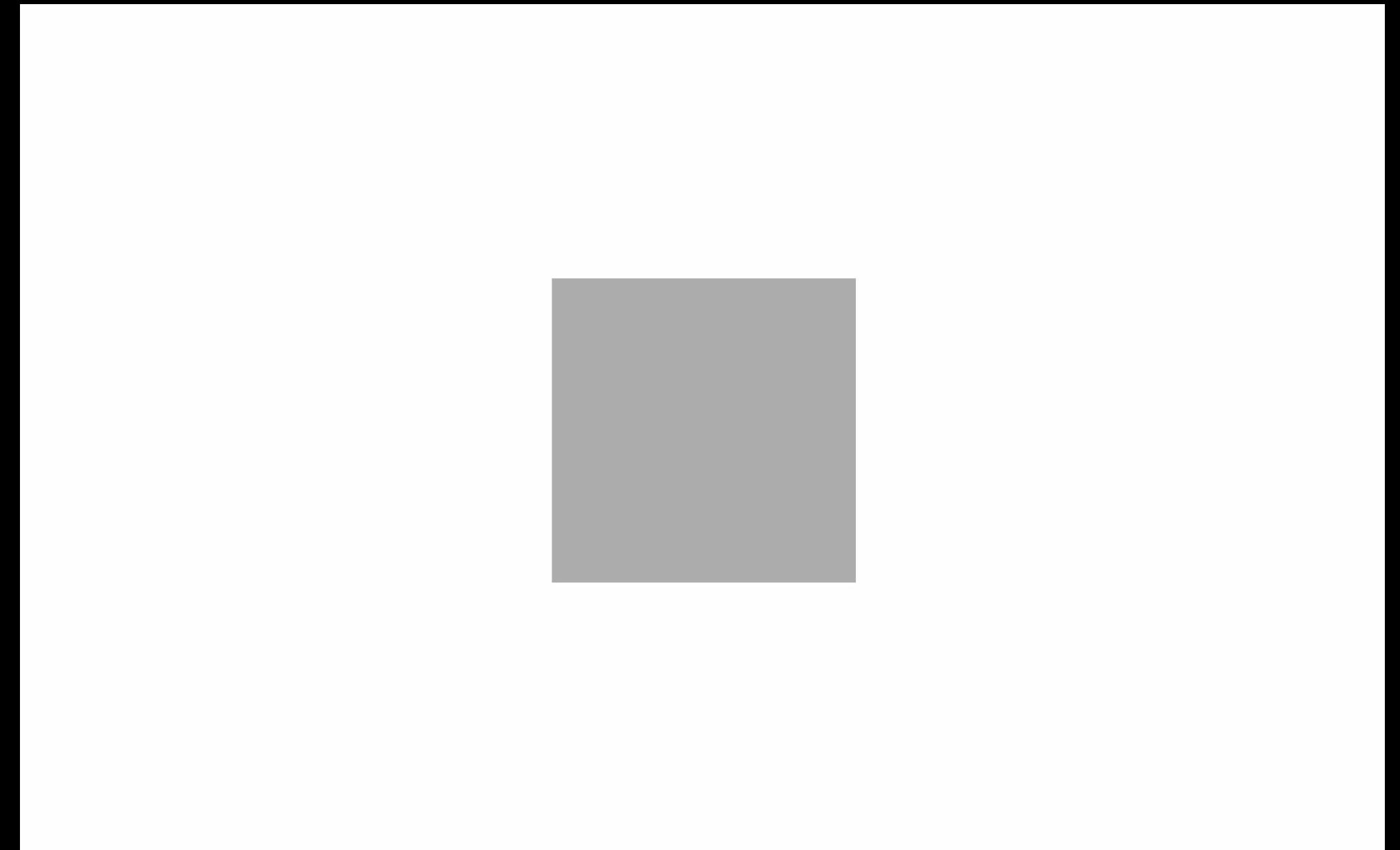
Пример анимации rotate

Анимация CSS свойств — opacity

```
gsap.from('#rect', {  
  opacity: 0,  
  repeat: -1,  
  duration: 2,  
  yoyo: true, // анимация проигрывается маятником  
})
```

Анимация CSS свойств — opacity

```
gsap.from('#rect', {  
  opacity: 0,  
  repeat: -1,  
  duration: 2,  
  yoyo: true, // анимация проигрывается маятником  
})
```

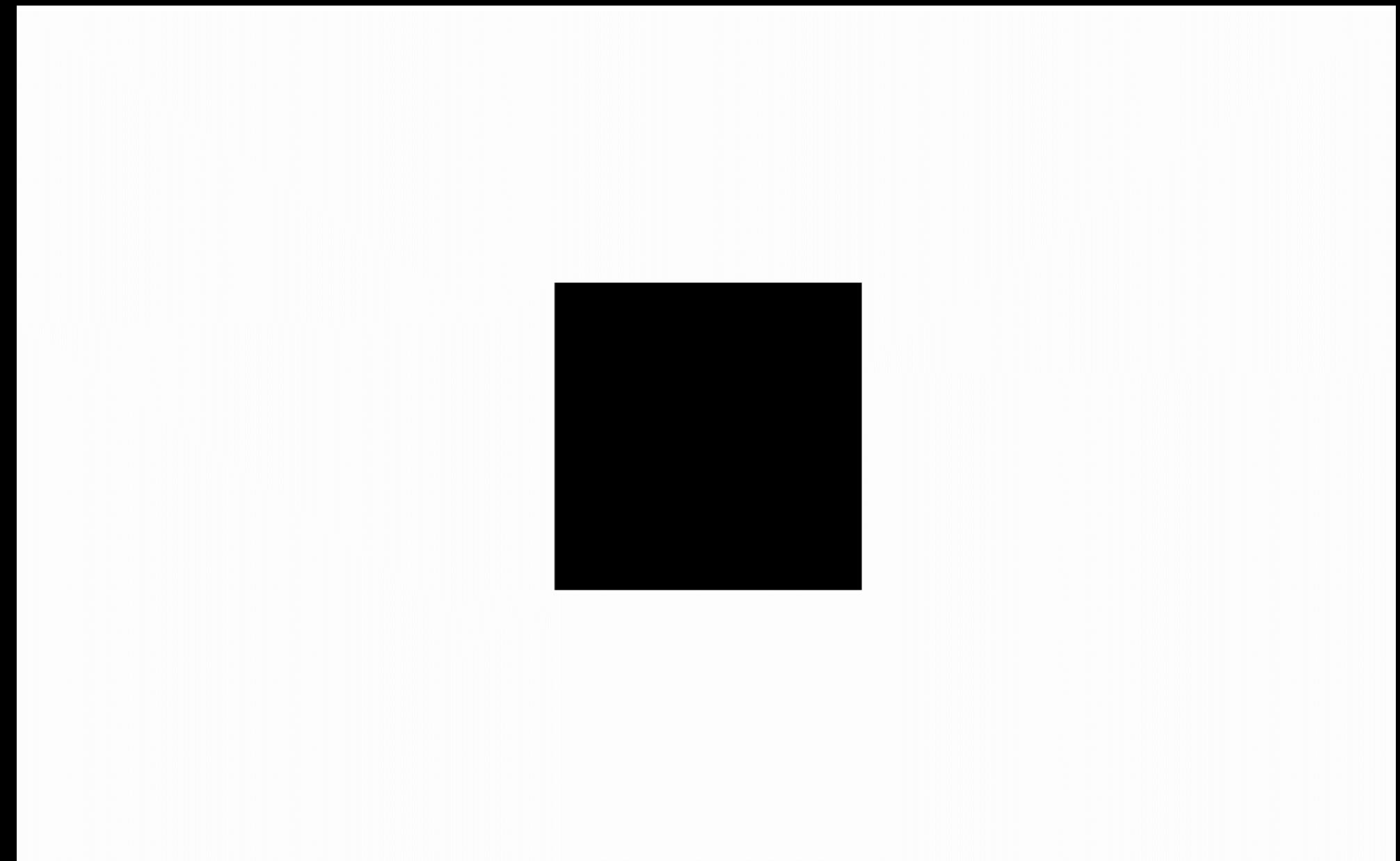


Анимация CSS свойств — translate

```
gsap.from('#rect', {  
    xPercent: 100,  
    repeat: -1,  
    duration: 2,  
    yoyo: true  
})
```

Анимация CSS свойств — translate

```
gsap.from('#rect', {  
  xPercent: 100,  
  repeat: -1,  
  duration: 2,  
  yoyo: true  
})
```

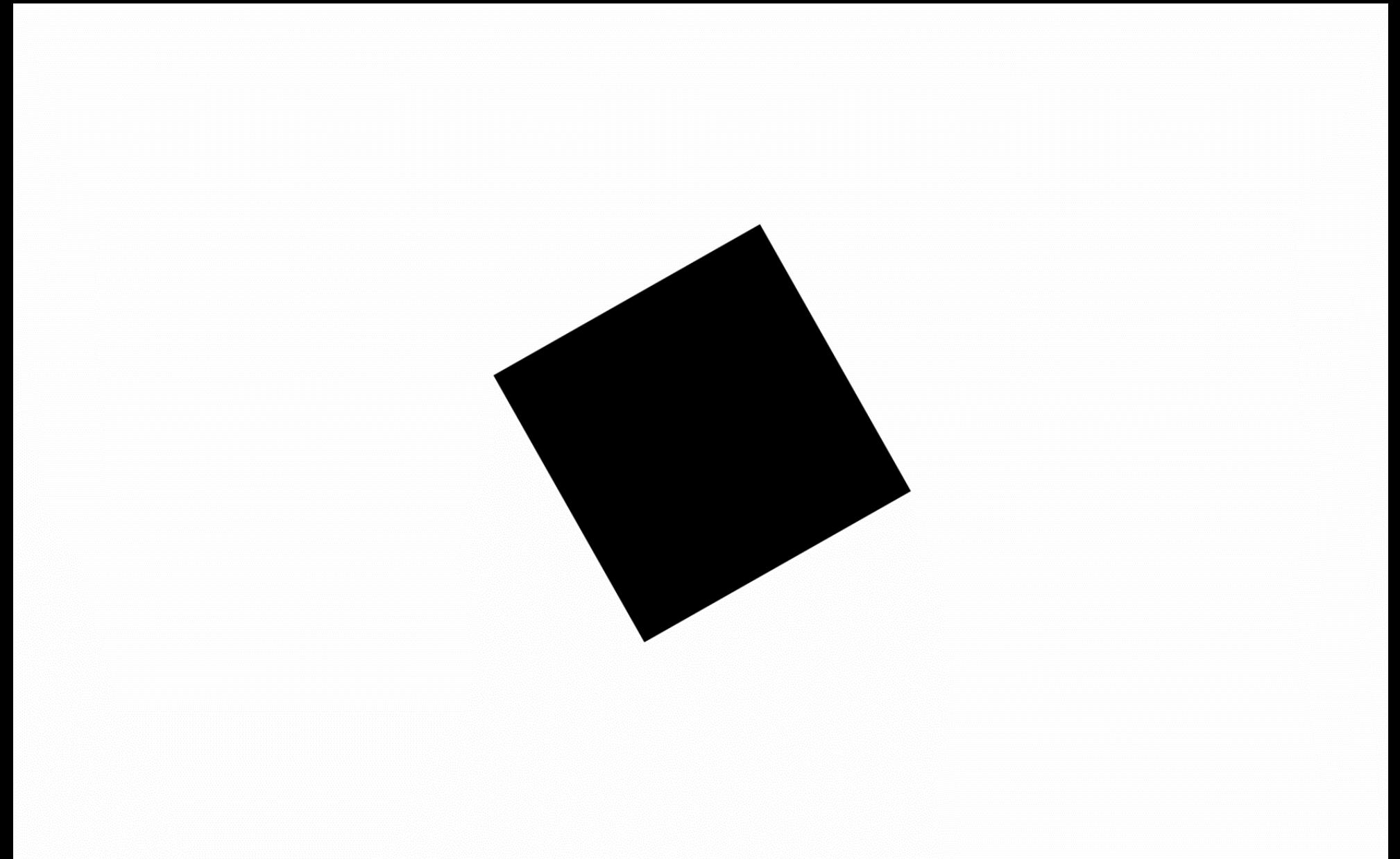


Анимация CSS свойств — rotate

```
gsap.from('#rect', {  
  rotate: 360,  
  repeat: -1,  
  ease: "none",  
  duration: 2,  
})
```

Анимация CSS свойств — rotate

```
gsap.from('#rect', {  
  rotate: 360,  
  repeat: -1,  
  ease: "none",  
  duration: 2,  
})
```





timeline

Что из себя представляет?

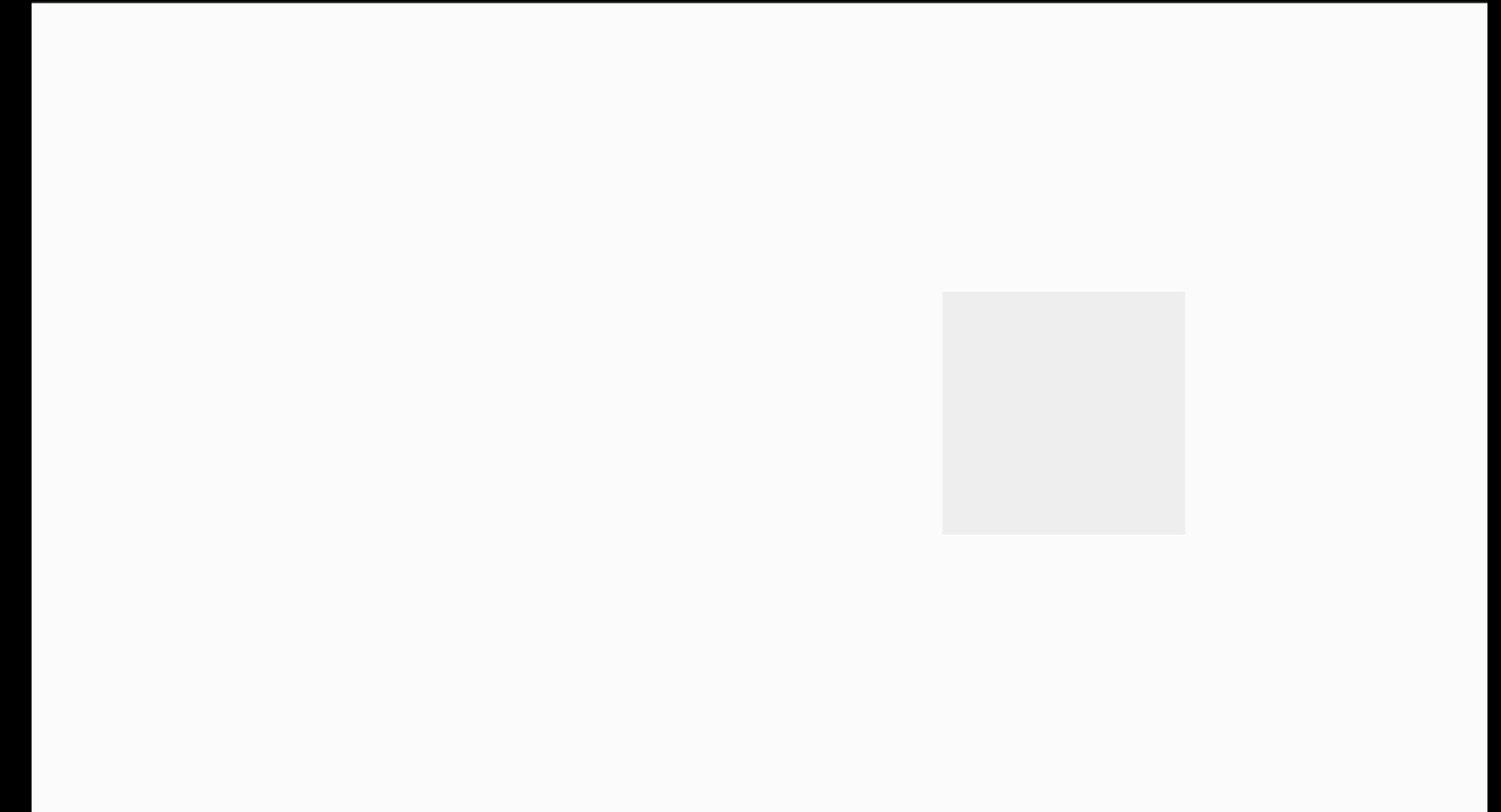
Позволяет создавать последовательность анимаций

```
const tl = gsap.timeline()
tl
  .from('#rect', {
    opacity: 0,
    duration: 1
  })
  .from('#rect', {
    translateX: 100,
    duration: 1
  }, "<")
  .from('#rect', {
    rotate: 180,
    duration: 2
  })
})
```

Что из себя представляет?

Позволяет создавать последовательность анимаций

```
const tl = gsap.timeline()
tl
  .from('#rect', {
    opacity: 0,
    duration: 1
  })
  .from('#rect', {
    translateX: 100,
    duration: 1
  })
  , "<")
  .from('#rect', {
    rotate: 180,
    duration: 2
  })
)
```



Что мы можем делать с timeline?

pause

play

resume

reverse

Запускать несколько тактов одновременно
или с задержкой используя:

«<» — запускает анимацию вместе с предыдущим тактом

«<+1» — запускает анимацию через секунду после начала предыдущего такта

«>-1» — запускает анимацию за секунду до окончания предыдущего такта

Что из себя представляет?

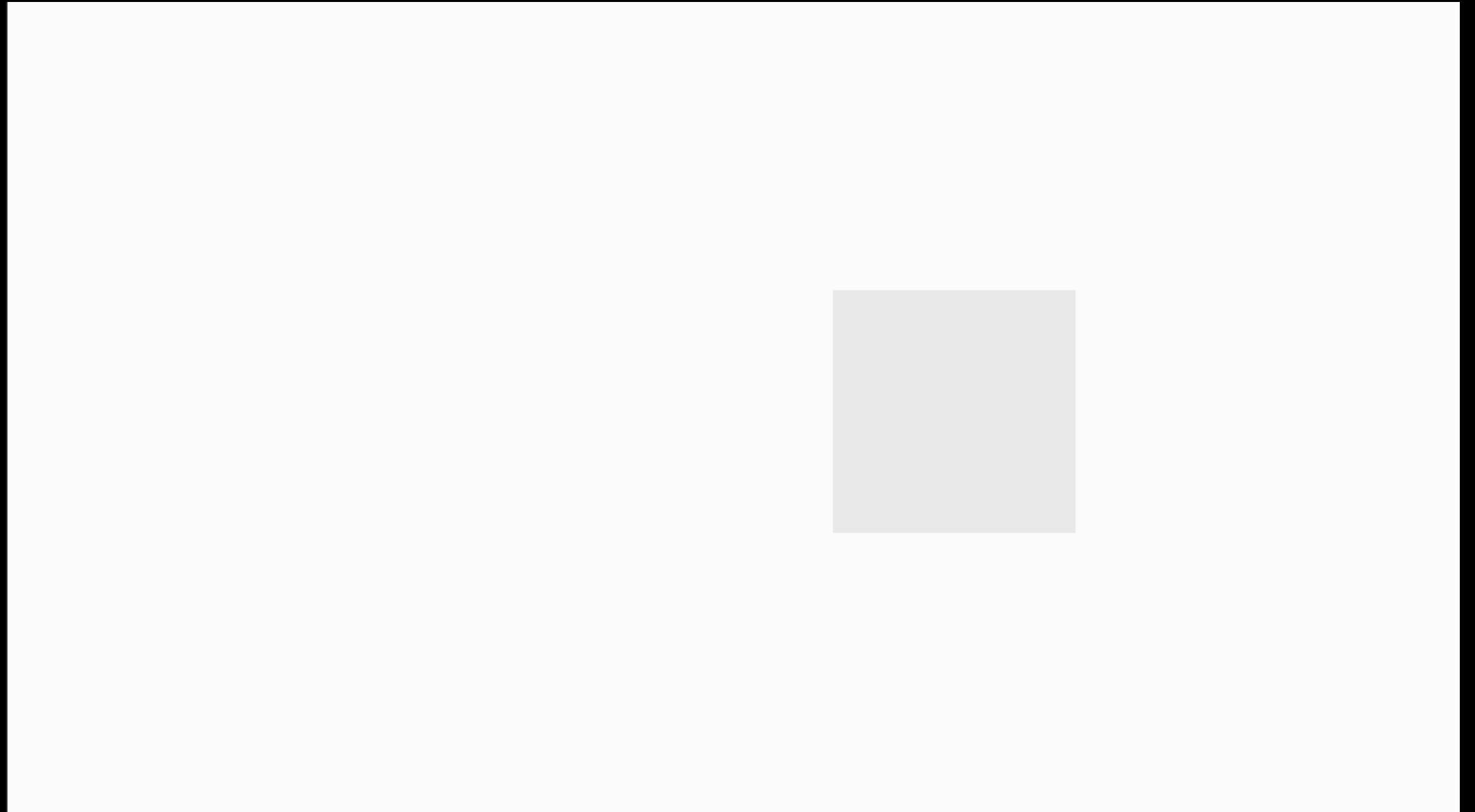
Позволяет создавать последовательность анимаций

```
const tl = gsap.timeline()
tl
  .from('#rect', {
    opacity: 0,
    duration: 1
  })
  .from('#rect', {
    translateX: 100,
    duration: 1
  }, "<")
  .from('#rect', {
    rotate: 180,
    duration: 2
  })
})
```

Что из себя представляет?

Позволяет создавать последовательность анимаций

```
const tl = gsap.timeline()
tl
  .from('#rect', {
    opacity: 0,
    duration: 1
  })
  .from('#rect', {
    translateX: 100,
    duration: 1
  })
  , "<")
  .from('#rect', {
    rotate: 180,
    duration: 2
  })
})
```



ScrollTrigger

Что из себя представляет?

Плагин для привязки анимации к позиции скролла

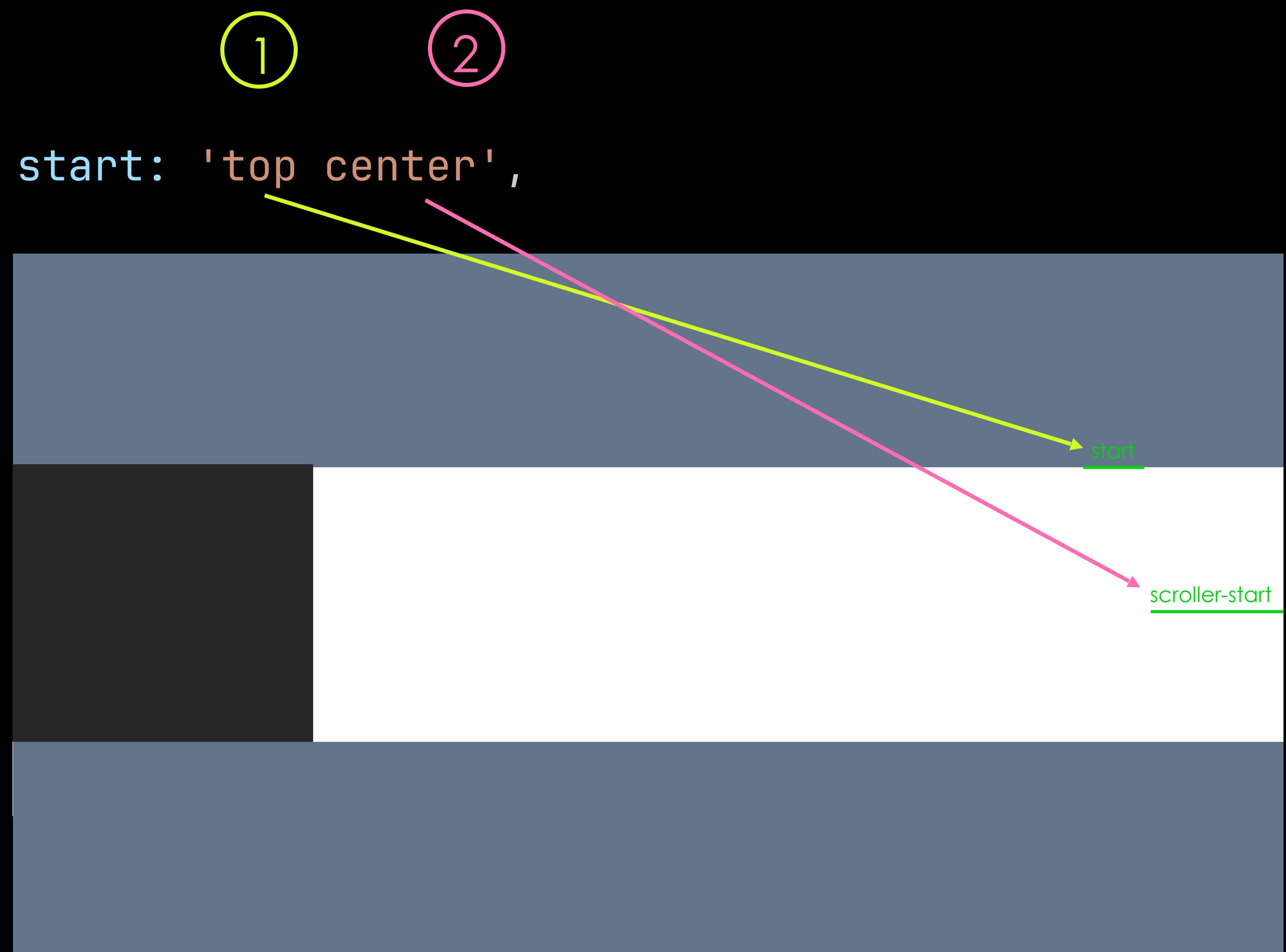
1. Устанавливаем блок-триггер
2. Создаем маркеры границ начала и конца анимации
3. Для лучшей отладки работы анимации используем **markers: true**

Как работает?

```
const tl = gsap.timeline({  
    scrollTrigger: {  
        trigger: '#rect',  
        start: 'top center',  
    },  
})
```

ScrollTrigger

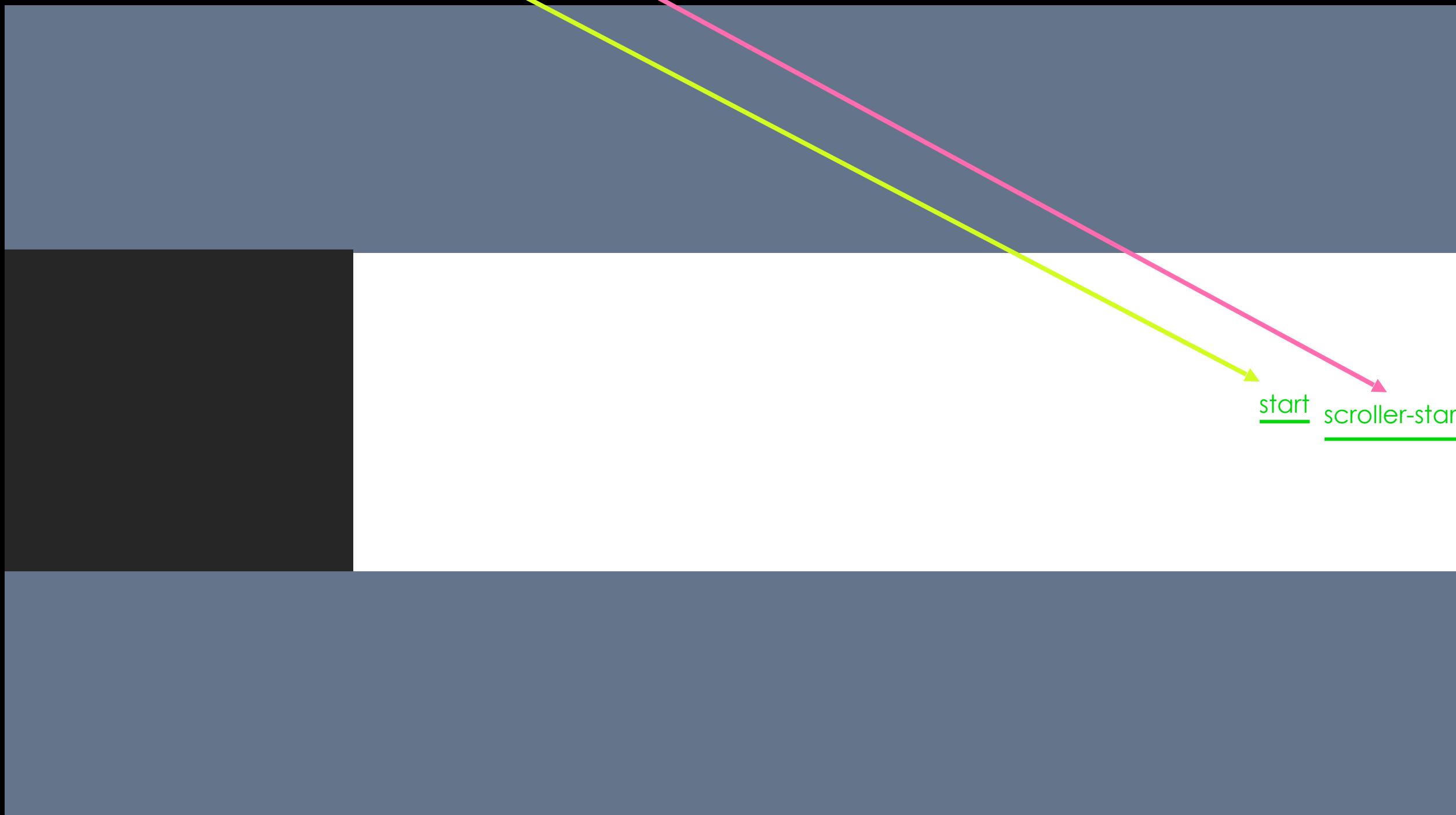
Первый маркер указывает позицию на блоке-тригере, второй указывает позицию на viewport



ScrollTrigger

- 1
- 2

start: 'center center',

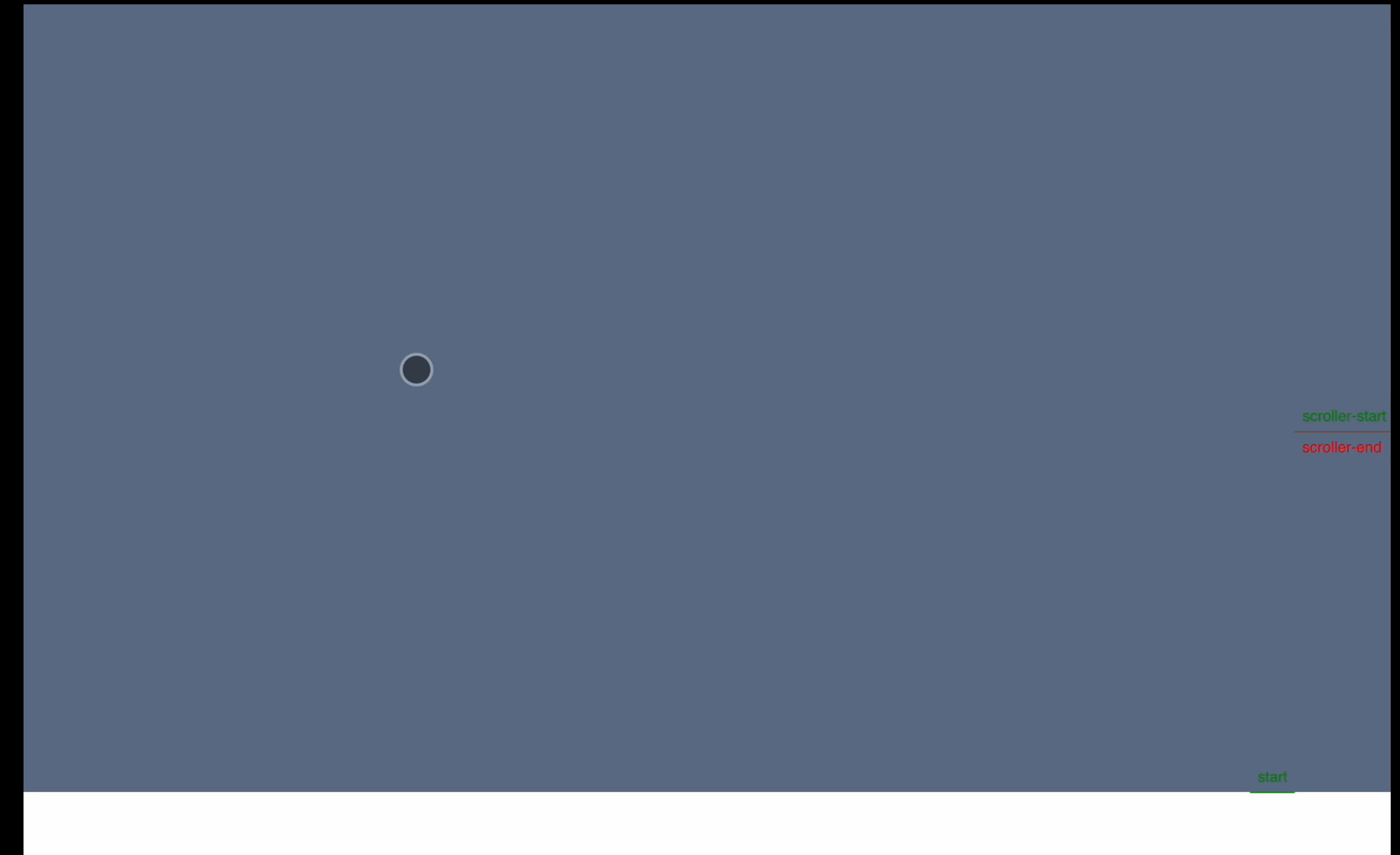


ScrollTrigger

```
tl.from('#rect', {  
  opacity: 0,  
  duration: 2  
})
```

ScrollTrigger

```
tl.from('#rect', {  
  opacity: 0,  
  duration: 2  
})
```



pin: true

Закрепляет элемент в то время, когда ScrollTrigger активен

```
const tl = gsap.timeline({
  scrollTrigger: {
    markers: true,
    trigger: '#rect',
    pin: true,
    start: 'top center',
    end: 'bottom center',
  },
})
tl.from('#rect', {
  opacity: 0,
  duration: 2
})
```

pin: true

Закрепляет элемент в то время, когда ScrollTrigger активен

```
const tl = gsap.timeline({
  scrollTrigger: {
    markers: true,
    trigger: '#rect',
    pin: true,
    start: 'top center',
    end: 'bottom center',
  },
})
tl.from('#rect', {
  opacity: 0,
  duration: 2
})
```



scrub: true

Связывает ход анимации непосредственно с полосой прокрутки, поэтому она действует как ползунок

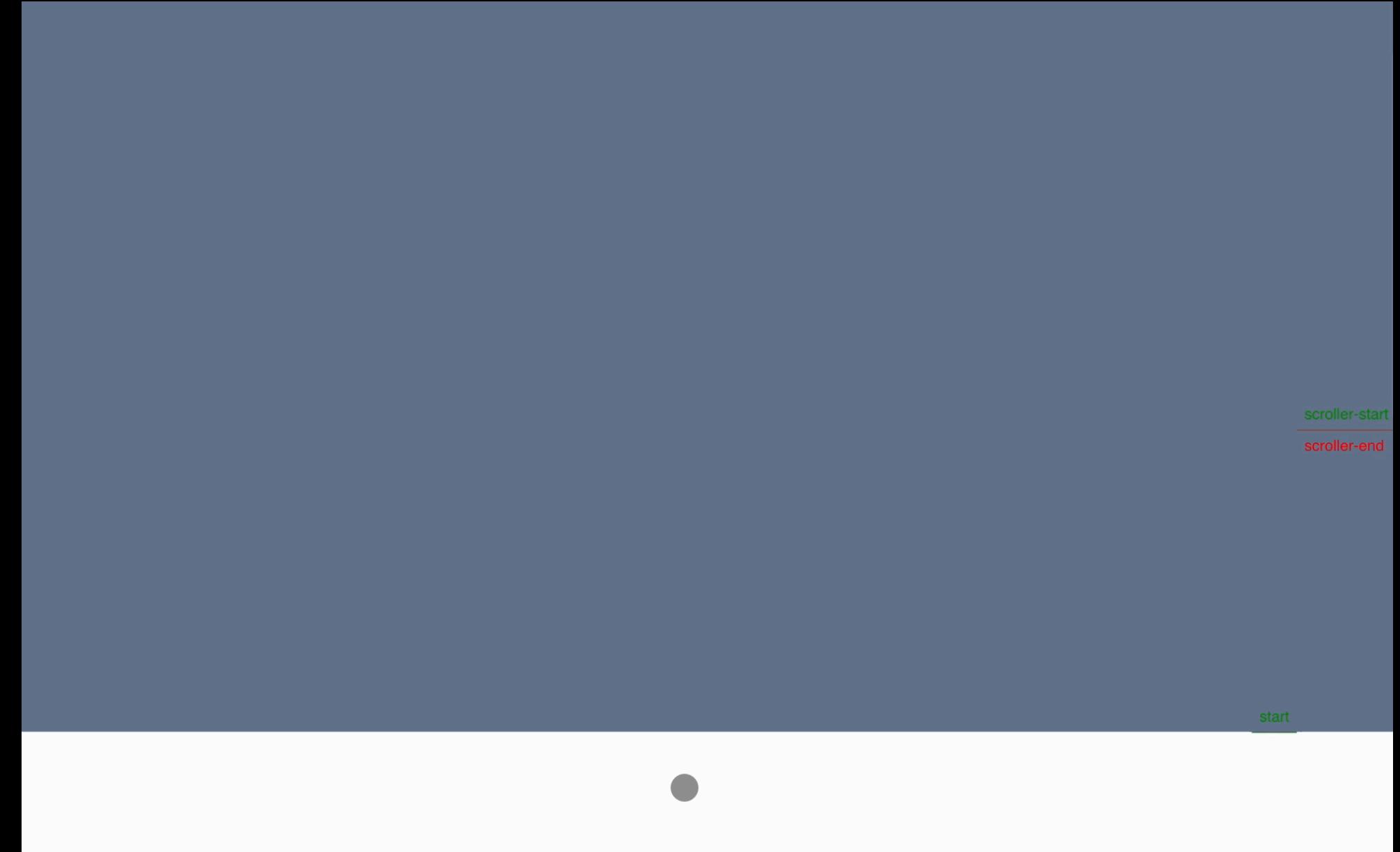
```
const tl = gsap.timeline({
  scrollTrigger: {
    markers: true,
    trigger: '#rect',
    scrub: true,
    start: 'top center',
    end: 'bottom center',
  },
})
tl.from('#rect', {
  opacity: 0,
  duration: 2
})
```

Vide

scrub: true

Связывает ход анимации непосредственно с полосой прокрутки, поэтому она действует как ползунок

```
const tl = gsap.timeline({
  scrollTrigger: {
    markers: true,
    trigger: '#rect',
    Scrub: true,
    start: 'top center',
    end: 'bottom center',
  },
})
tl.from('#rect', {
  opacity: 0,
  duration: 2
})
```



Stagger

Что из себя представляет?

Позволяет создавать ступенчатые анимации

```
gsap.to("#box", {  
    duration: 0.5,  
    opacity: 0,  
    y: -100,  
    stagger: {  
        from: from,  
        each: 0.2,  
    },  
    ease: "back.in",  
});
```



SplitText

Что из себя представляет?

Плагин для текстовой анимации

```
mySplitText = new SplitText("#animatedText"),
chars = mySplitText.elements; //разбиваем текст по элементам

gsap.from(chars, {
  duration: 0.2,
  opacity: 0,
  scale: 0,
  repeat: -1,
  repeatDelay: 5,
  stagger: 1, //добавляет ступенчатость анимации, каждый новый элемент
появляется через 1 секунду
});
```



Эй ты



Draggable

Что из себя представляет?

Плагин для взаимодействия с элементами

Что из себя представляет?

Плагин для взаимодействия с элементами

Основные возможности плагина

Drag (перемещение)

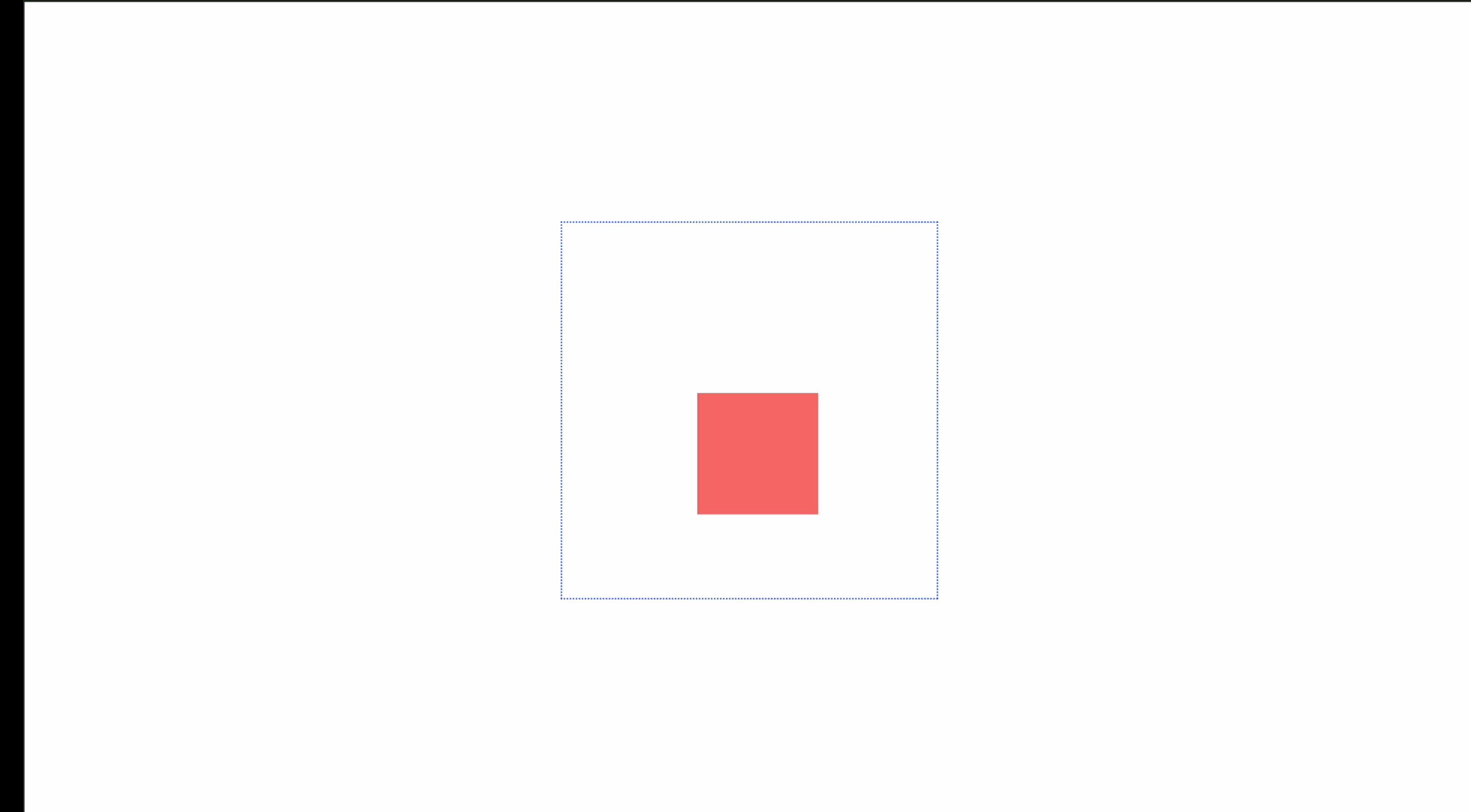
Spin (поворот)

Drag

```
Draggable.create(".rect_drag", {  
  type: "x,y",  
  bounds: '.bounds-container'  
});
```

Drag

```
Draggable.create(".rect_drag", {  
  type: "x,y",  
  bounds: '.bounds-container'  
});
```



API

Позволяет легко получить позицию элемента

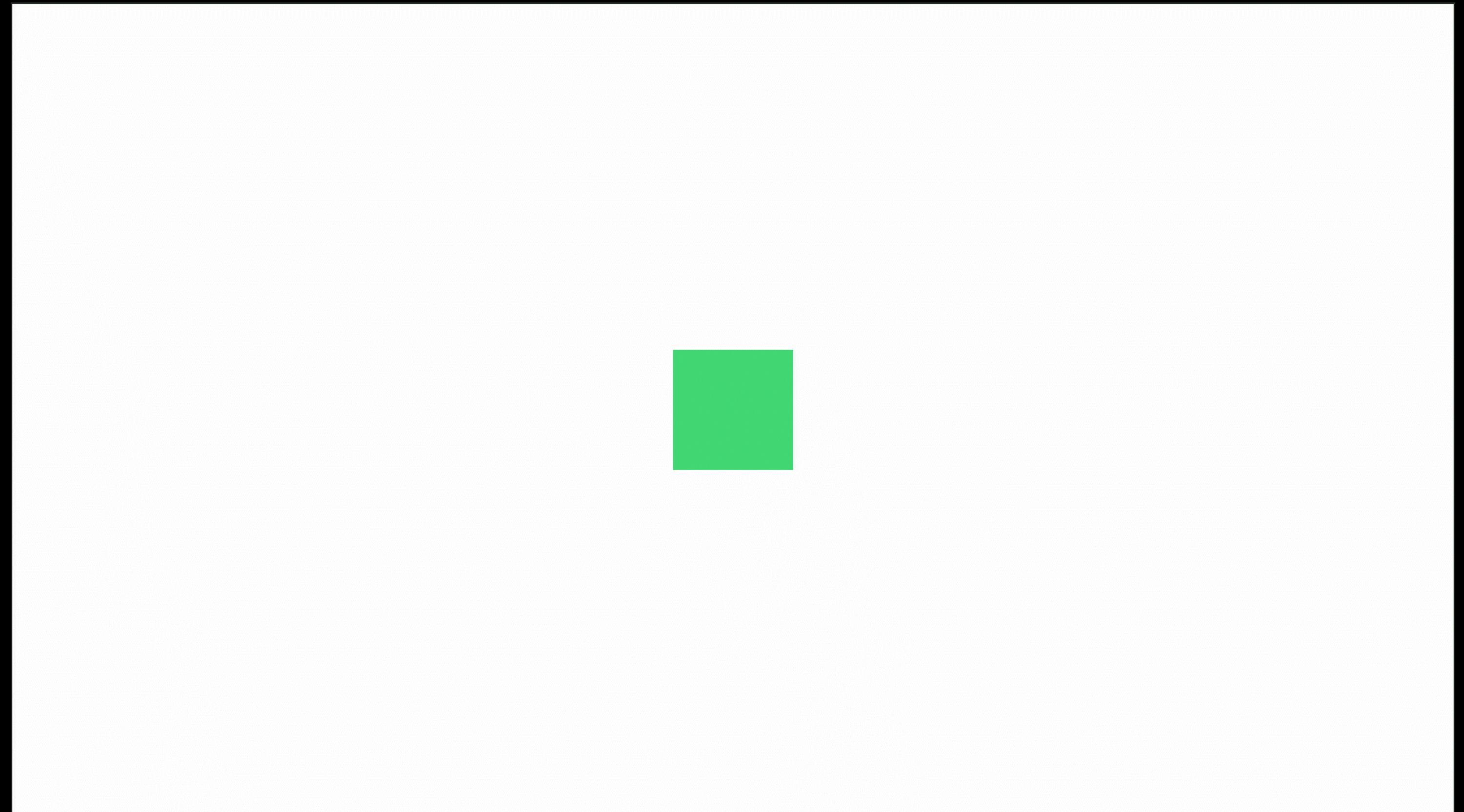
```
Draggable.create(".rect_drag", {  
  type: "x,y",  
  onDragEnd: function () {  
    const x = this.x; // Позиция по оси X  
    const y = this.y; // Позиция по оси Y  
    console.log(x, y)  
  },  
  bounds: '.bounds-container'  
});
```

Spin

```
Draggable.create("#rect_rotate", {  
  type: "rotation",  
  inertia: true  
});
```

Spin

```
Draggable.create("#rect_rotate", {  
  type: "rotation",  
  inertia: true  
});
```



Пример с сайта компании Kodix

```
useGSAP(() => {
  gsap.registerPlugin(Draggable)

  const startAutoRotate = () => {
    return gsap.to('.GLOBUS', {
      rotation: '+=360', // поворачивать на 30 градусов каждую секунду
      duration: 20, // длительность анимации - 1 секунда
      ease: 'linear',
      repeat: -1,
    });
  }

  let rotateAnimation = startAutoRotate();

  Draggable.create(".GLOBUS", {
    type: "rotation",
    inertia: true,
    onPress: () => {
      // Остановка анимации при начале взаимодействия
      rotateAnimation.pause();
    },
    onRelease: () => {
      // Возобновление анимации после взаимодействия
      rotateAnimation = startAutoRotate();
    },
  })
}, [])
```

Video

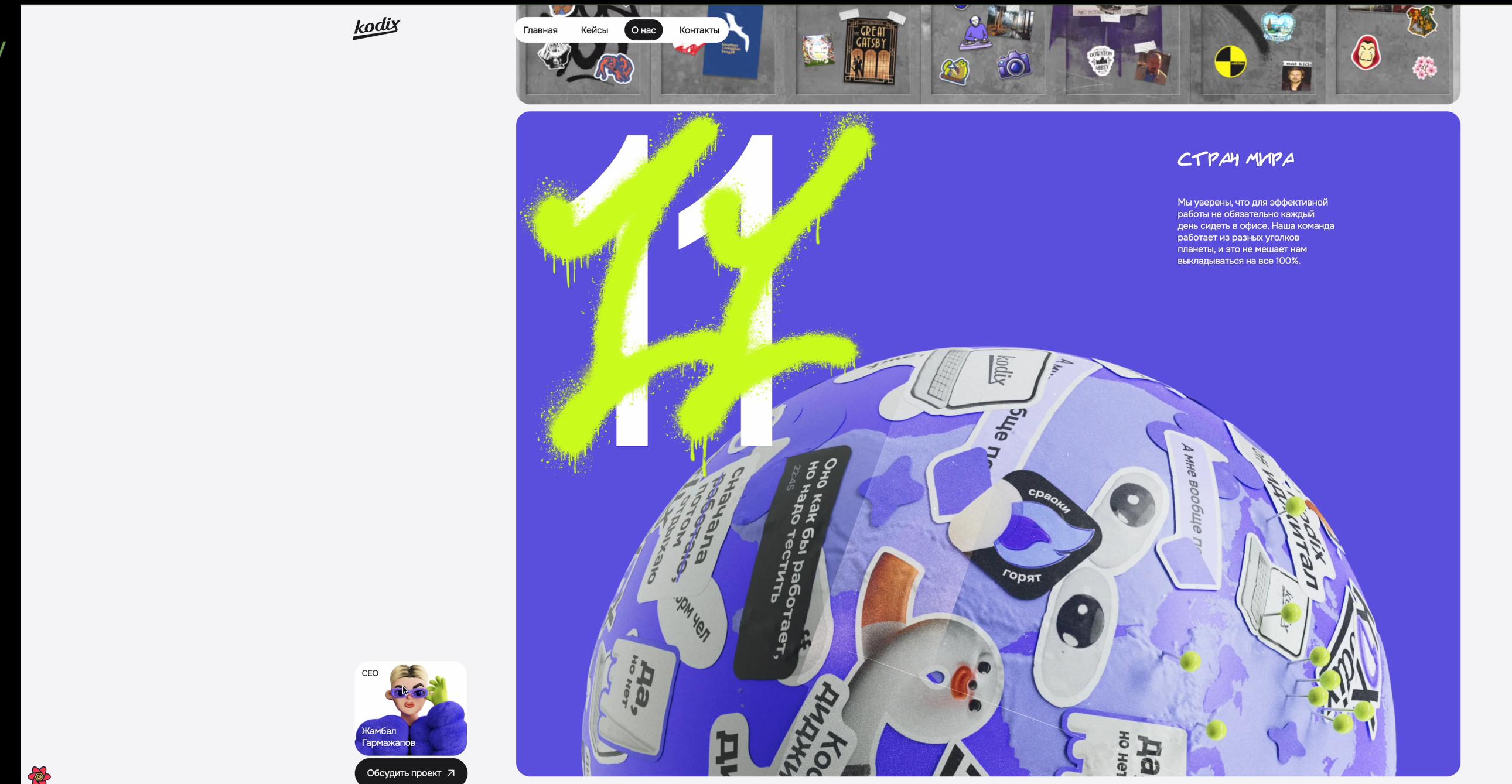
Пример с сайта компании Kodix

```
useGSAP(() => {
  gsap.registerPlugin(Draggable)

  const startAutoRotate = () => {
    return gsap.to('.GLOBUS', {
      rotation: '+=360', // поворачивать на 30 градусов каждую секунду
      duration: 20, // длительность анимации - 1 секунда
      ease: 'linear',
      repeat: -1,
    });
  }

  let rotateAnimation = startAutoRotate();

  Draggable.create(".GLOBUS", {
    type: "rotation",
    inertia: true,
    onPress: () => {
      // Остановка анимации при начале взаимодействия
      rotateAnimation.pause();
    },
    onRelease: () => {
      // Возобновление анимации после взаимодействия
      rotateAnimation = startAutoRotate();
    },
  })
}, [])
```





MorphSVG

Что из себя представляет?

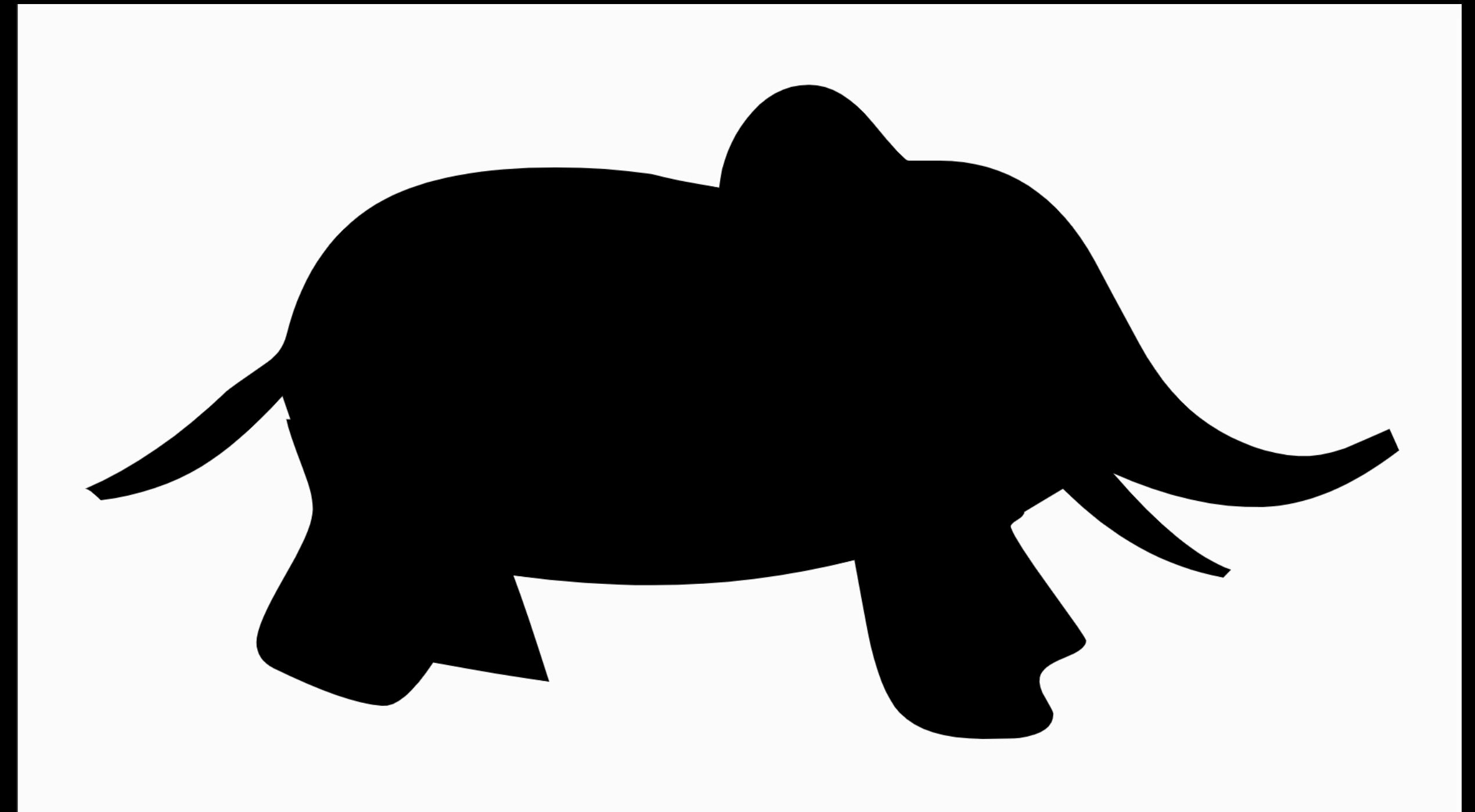
Переход одной svg в другую

```
gsap.to('#svg1', {  
  morphSVG: '#svg2'  
})
```

Что из себя представляет?

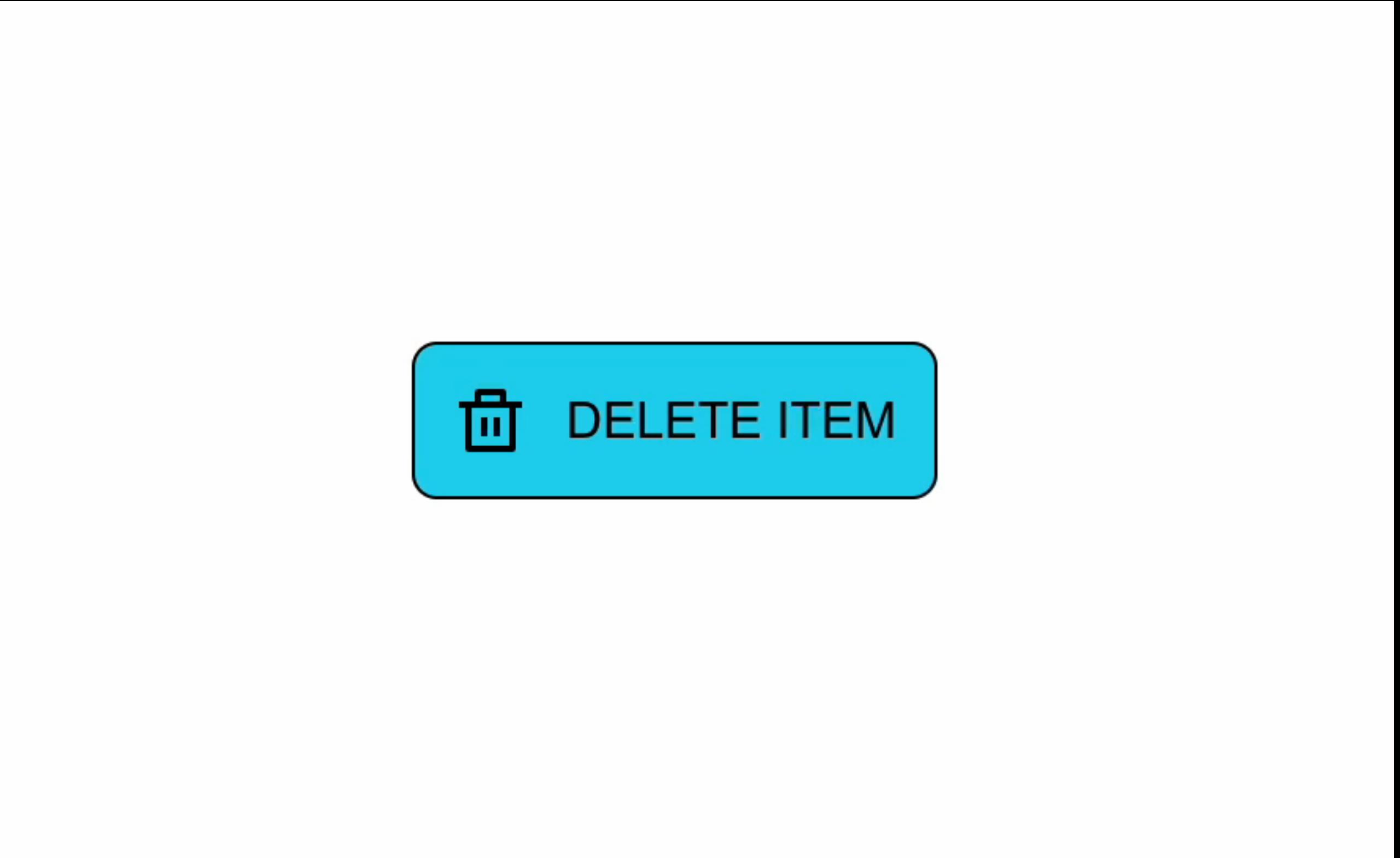
Переход одной svg в другую

```
gsap.to('#svg1', {  
  morphSVG: '#svg2'  
})
```



```
const tl = gsap.timeline()
tl.to('#trash_init', {
  morphSVG: '#trash_last'
})
  .to('#svg', {
    x: button.clientWidth / 2 - svg.clientWidth,
    scale: 1.5
  })
  .to("#span", {
    opacity: 0
  }, "<")
  .to('#paper', {
    opacity: 1,
  })
  .to('#paper', {
    y: 100,
  })
  .to("#paperSVG", {
    morphSVG: "#check"
  }, "<")
  .to('#paper', {
    opacity: 0,
  })
  .to('#trash_init', {
    morphSVG: '#trash_init',
  })
  .to('#svg', {
    x: 0,
    scale: 1
  }, "<")
  .to("#span", {
    opacity: 1
  }, "<")
  .to('#paper', {
    opacity: 0,
  })
  .to('#paper', {
    y: 0,
  })
  .to("#paperSVG", {
    morphSVG: "#paperSVG"
})

```



Продакшен кейсы

Solaris auto



Kodix Tournament CS2 / Dota 2



GSAP

Плюсы

- Гибкость и мощность
- Удобный API
- Совместимость с браузерами
- Расширяемость с помощью плагинов

Минусы

- Избыточность для простых задач
- Увеличение веса бандла

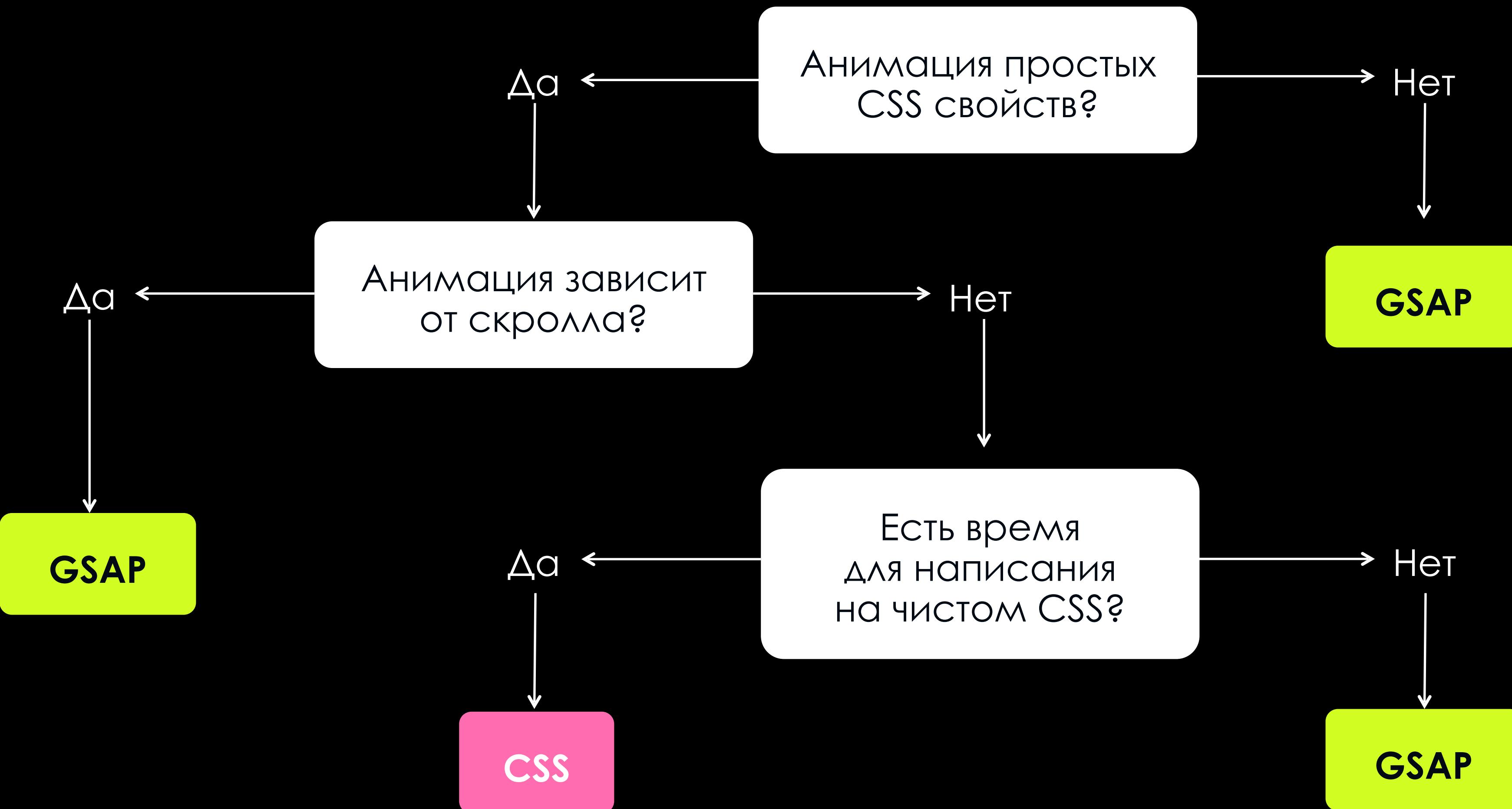
26.5 kB

MINIFIED + GZIPPED

GSAP vs CSS?

Критерий	GSAP	CSS
Сложность	Подходит для сложных анимаций	Подходит для простых эффектов
Контроль	Полный контроль (пауза, отмена, обратное воспроизведение)	Меньше контроля
Производительность	Оптимизация для сложных сцен	Высокая для простых анимаций
SVG, Canvas, WebGL	Полная поддержка	Ограниченнная поддержка
Кроссбраузерность	Обеспечивается библиотекой	Иногда нужно решать проблемы вручную
Скорость разработки	Быстрее для сложных сценариев	Быстрее для базовых эффектов
Зависимость от JS	Нужен JavaScript	Не требует JS

Как выбрать GSAP или CSS?



Итого

1

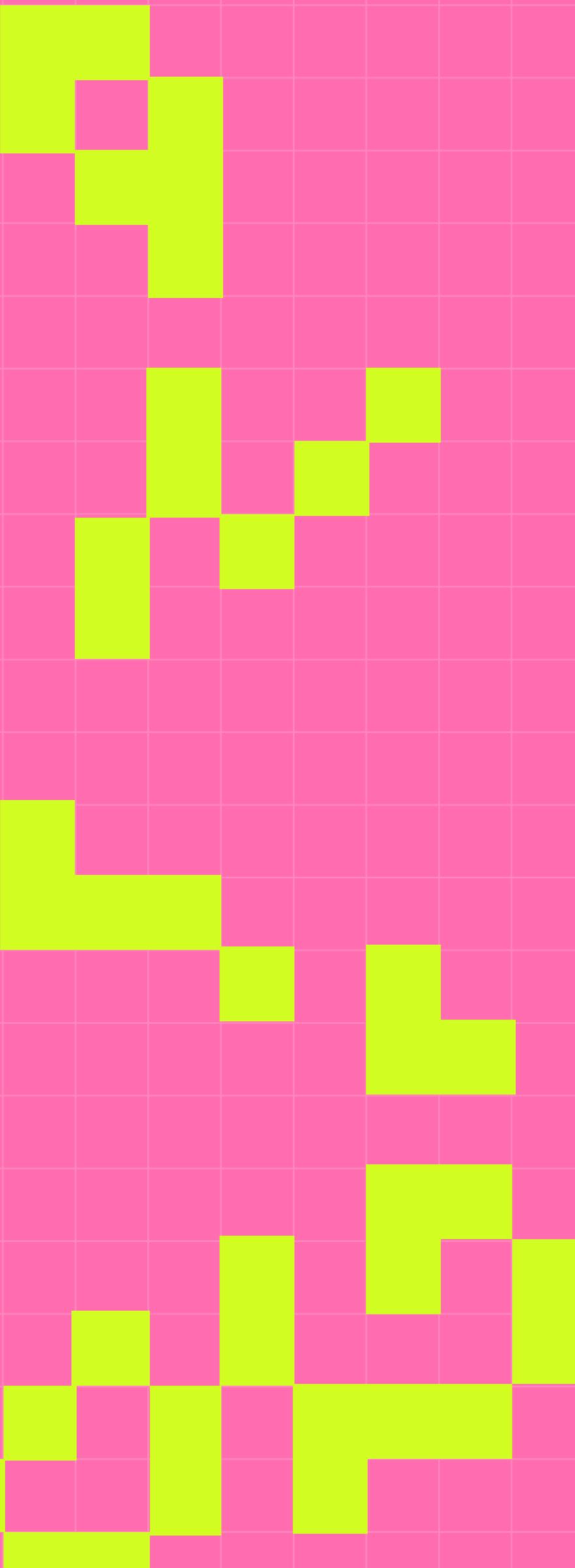
Анимации это просто , если знать правильный подход

2

GSAP экономит не только время, но и нервы

3

Используйте то, что уже написано до вас



Мой TG:

@Doroshkeviches



Мой Linkedin:

<https://clck.ru/3FimF4>



Сайт Kodix Agency:

<https://kodix.ru>



ТГ Kodix Agency:

@kodixxx



По вопросам **Dota 2/ CS2 турнира**
обращаться к HRD Kodix
Елене Ганзиной:

