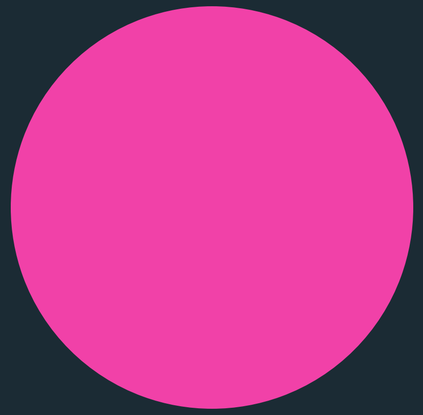


Macromedia Flash Professional 8 - [Untitled-1*]

File Edit View Insert Modify Text Commands Control Window Help

Tools

View

Colors

Options

Untitled-1*

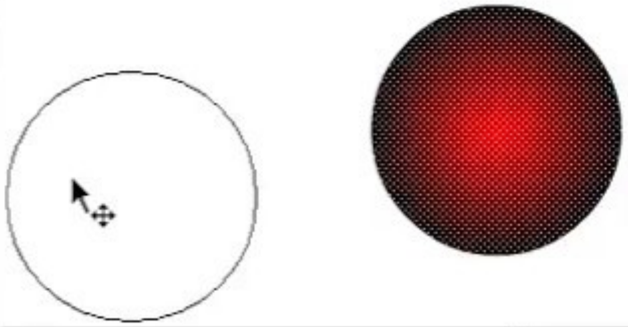
Timeline

Scene 1

100%

Layer 1

1 12.0 fps 0.0s



Color

Color Mixer Color Swatches

Library - Untitled-1

Untitled-1

One item in library

Name Type

Ball Graph

Actions

Properties Filters Parameters

Shape

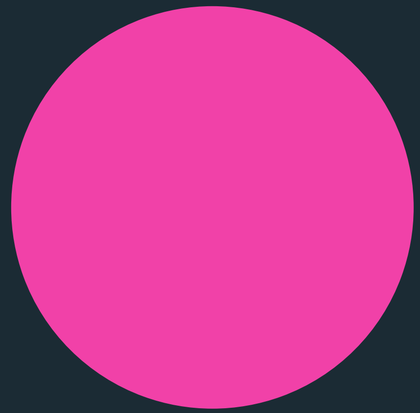
Custom... Cap:

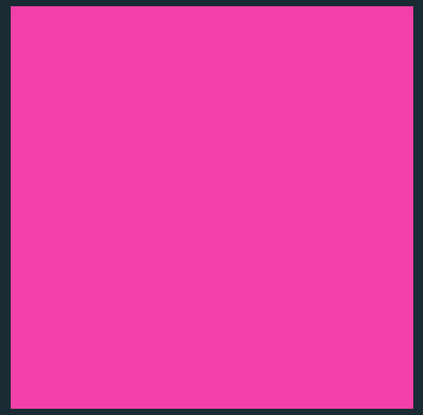
Stroke hinting Scale: Miter: 3 Join:

W: 132.5 X: 198.5

H: 132.5 Y: 172.9

EN 2:49 PM 14-Jun-18





Making A Splash With View Transitions

MIKE HARTINGTON
@MHARTINGTON

> THE VIEW TRANSITIONS API PROVIDES A MECHANISM FOR EASILY CREATING ANIMATED TRANSITIONS BETWEEN DIFFERENT DOM STATES WHILE ALSO UPDATING THE DOM CONTENTS IN A SINGLE STEP.

MDM WEB DOCS

```
<ul>
  <li>
    <p>Item Title</p>
    <i>Item Icon</i>
  </li>
  <li>
    <p>Item Title</p>
    <i>Item Icon</i>
  </li>
  <li>
    <p>Item Title</p>
    <i>Item Icon</i>
  </li>
  <li>
    <p>Item Title</p>
    <i>Item Icon</i>
  </li>
</ul>
```

- ITEM TITLE ⭐
- ITEM TITLE ⭐
- ITEM TITLE ⭐
- ITEM TITLE ⭐

```
<ul>
  <li>
    <p>Item Title</p>
    <i>Item Icon</i>
  </li>

  <li>
    <p>Item Title</p>
    <i>Item Icon</i>
  </li>
  <li>
    <p>Item Title</p>
    <i>Item Icon</i>
  </li>
</ul>
```

- ITEM TITLE ⭐
- ITEM TITLE ⭐
- ITEM TITLE ⭐

```
document.startViewTransition(() => {

  // Start changing the DOM

});
```

BASIC ANIMATIONS

```
document.startViewTransition(() => {

  // Start changing the DOM

});
```

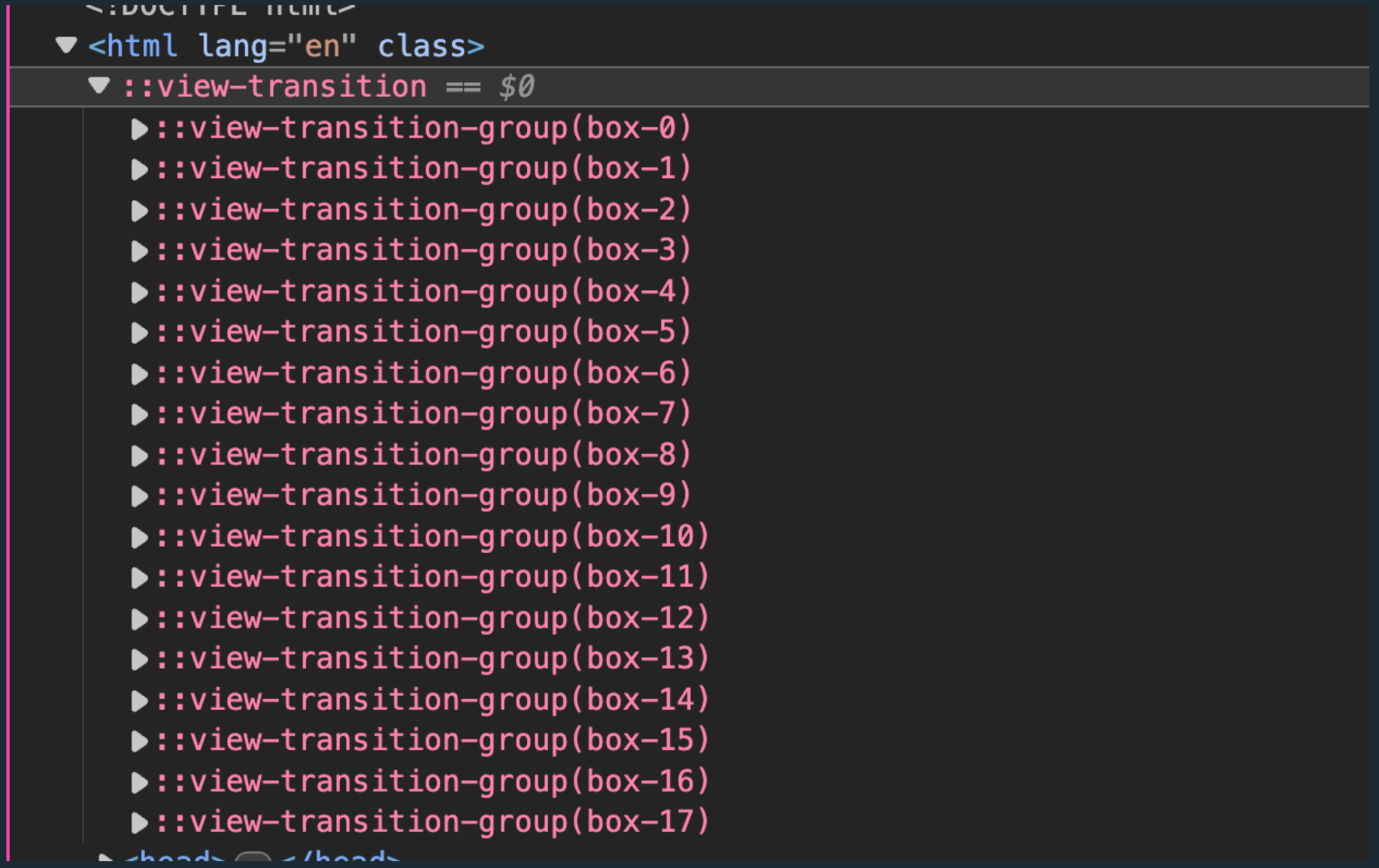
```
.box {
  view-transition-name: box;
}
```

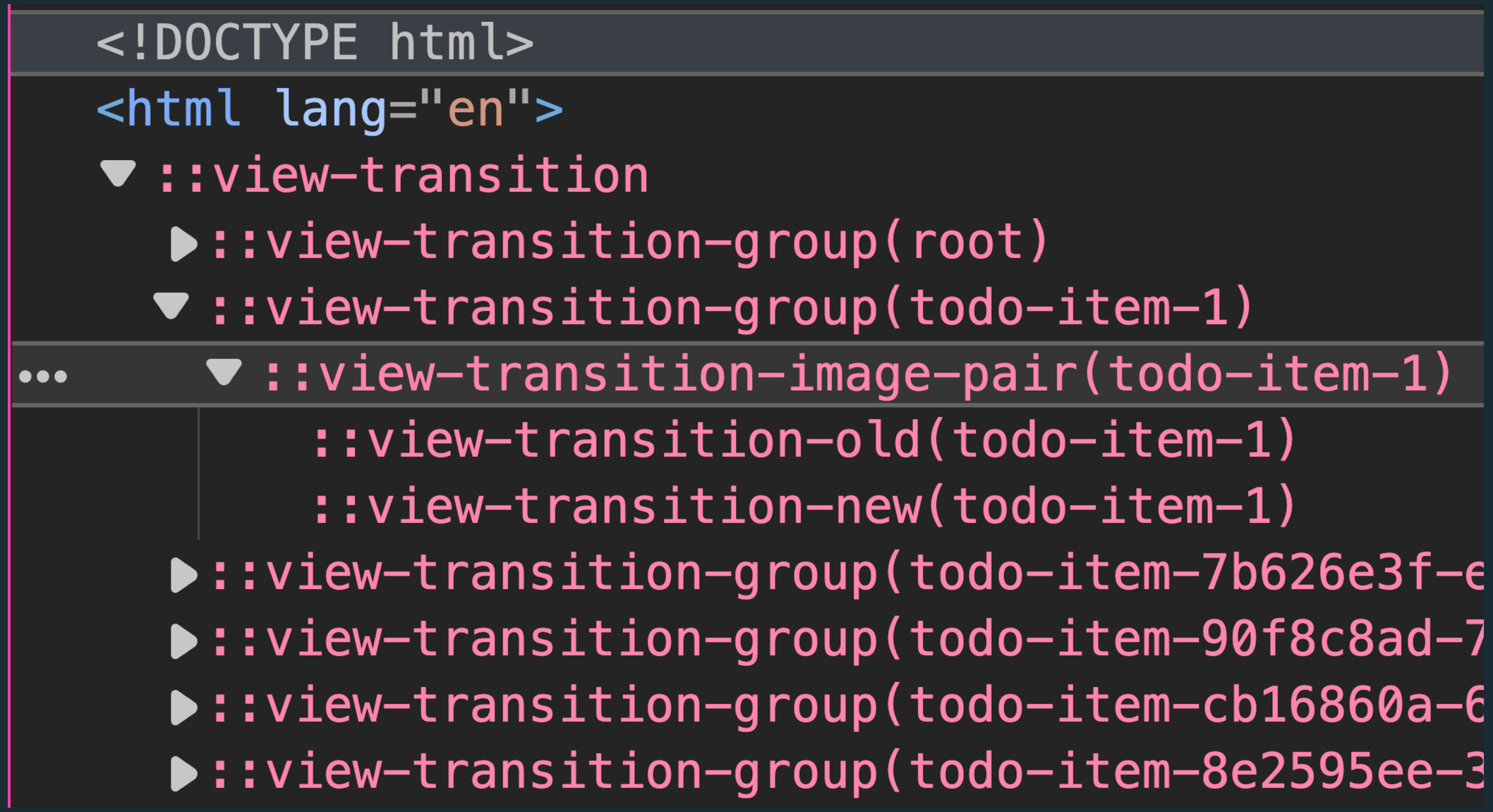
MAKING MOVES

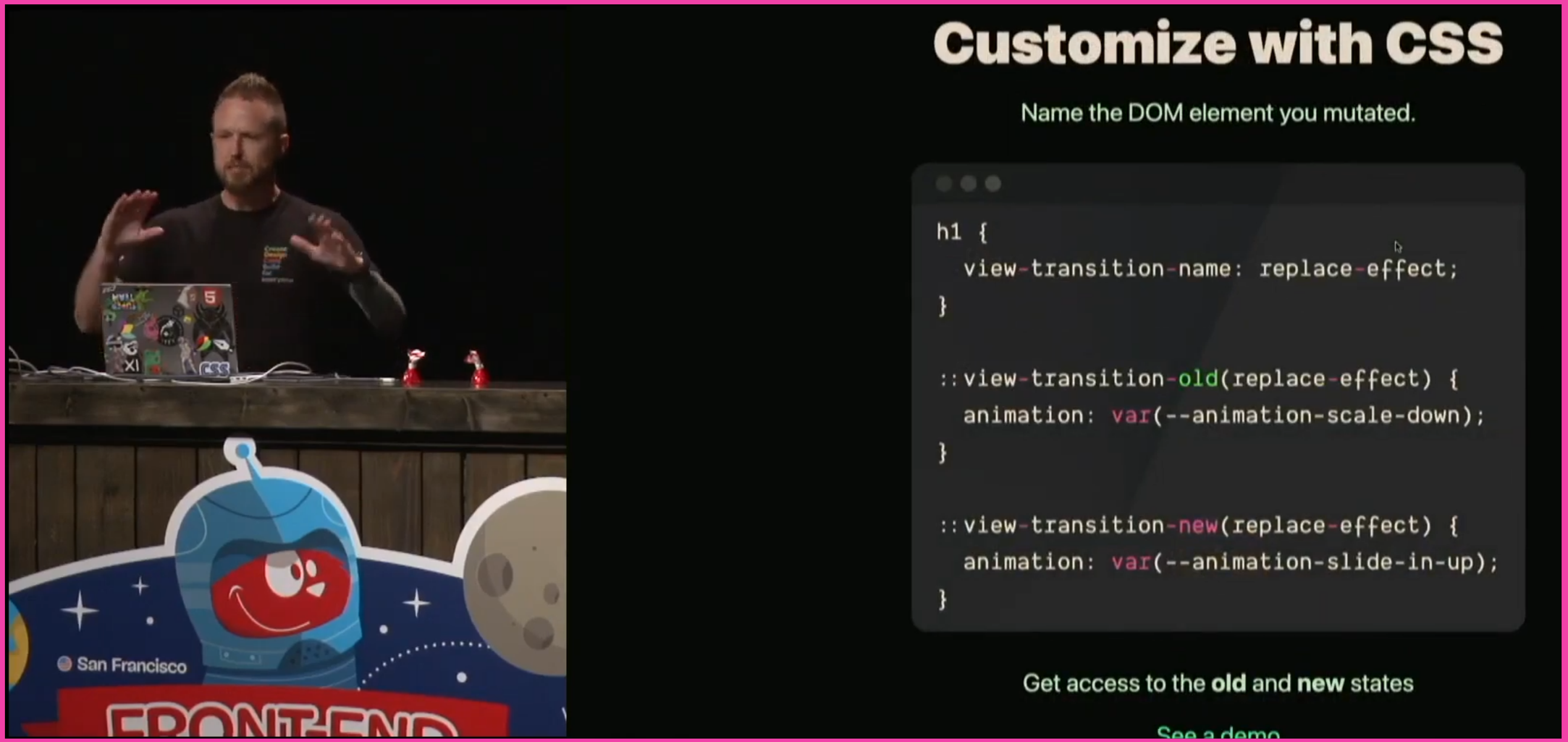
```
::view-transition-old(box-0):only-child {
  animation: scale-out 0.25s ease-out forwards;
}

::view-transition-new(box-0):only-child {
  opacity: 0;
  animation:
    fade-in 0.01s ease-in forwards,
    scale-out 0.4s ease-in reverse;
}
```

DISPLAY NONE


<!DOCTYPE html>
<html lang="en" class>
::view-transition == $0
::view-transition-group(box-0)
::view-transition-group(box-1)
::view-transition-group(box-2)
::view-transition-group(box-3)
::view-transition-group(box-4)
::view-transition-group(box-5)
::view-transition-group(box-6)
::view-transition-group(box-7)
::view-transition-group(box-8)
::view-transition-group(box-9)
::view-transition-group(box-10)
::view-transition-group(box-11)
::view-transition-group(box-12)
::view-transition-group(box-13)
::view-transition-group(box-14)
::view-transition-group(box-15)
::view-transition-group(box-16)
::view-transition-group(box-17)


<!DOCTYPE html>
<html lang="en">
::view-transition
::view-transition-group(root)
::view-transition-group(todo-item-1)
::view-transition-image-pair(todo-item-1)
::view-transition-old(todo-item-1)
::view-transition-new(todo-item-1)
::view-transition-group(todo-item-7b626e3f-e
::view-transition-group(todo-item-90f8c8ad-7
::view-transition-group(todo-item-cb16860a-6
::view-transition-group(todo-item-8e2595ee-3



ADAM ARGYLE'S TALK

Introduction to Angular animations

Animation provides the illusion of motion: HTML elements change styling over time. Well-designed animations can make your application more fun and straightforward to use, but they aren't just cosmetic. Animations can improve your application and user experience in a number of ways:

- Without animations, web page transitions can seem abrupt and jarring
- Motion greatly enhances the user experience, so animations give users a chance to detect the application's response to their actions
- Good animations intuitively call the user's attention to where it is needed

Typically, animations involve multiple style *transformations* over time. An HTML element can move, change color, grow or shrink, fade, or slide off the page. These changes can occur simultaneously or sequentially. You can control the timing of each transformation.

Angular's animation system is built on CSS functionality, which means you can animate any property that the browser considers animatable. This includes positions, sizes, transforms, colors, borders, and more. The W3C maintains a list of animatable properties on its CSS Transitions page.

```
<ul [@listAnimation]="store.todos().length">
   @for (todo of store.todos(); track todo.id) {
     <li>
       <p>{{ todo.name }}</p>
       <button (click)="removeTodo(todo)">🗑️</button>
     </li>
   }
 </ul>
```

```
<ul [@listAnimation]="store.todos().length">
   @for (todo of store.todos(); track todo.id) {
     <li>
       <p>{{ todo.name }}</p>
       <button (click)="removeTodo(todo)">🗑️</button>
     </li>
   }
 </ul>
```

```
@Component({
  animations: [
    trigger('listAnimation', [
      transition(':increment', [
        group([
          query('li:enter', [
            style({ transform: 'translate3d(0, calc(-100% + 1px), 0)', 'z-index': -1  }),
            animate( '300ms ease-out', style({ transform: 'translate3d(0, 0, 0)' })),
          ]),
        ]),
      ]),
      transition(':decrement', [
        group([
          query('li:leave', [animate('250ms ease-out', style({ opacity: 0 }))]),
          query('li:leave ~ li:not(:leave)', [
            animate( '300ms ease-out', style({ transform: 'translate3d(0, calc(-100% + 1px), 0)' })),
          ]),
        ]),
      ]),
    ]),
  ],
})
```

```
@Component({
  animations: [
    trigger('listAnimation', [
      transition(':increment', [
        group([
          query('li:enter', [
            style({ transform: 'translate3d(0, calc(-100% + 1px), 0)', 'z-index': -1  }),
            animate( '300ms ease-out', style({ transform: 'translate3d(0, 0, 0)' })),
          ]),
        ]),
      ]),
      transition(':decrement', [
        group([
          query('li:leave', [animate('250ms ease-out', style({ opacity: 0 }))]),
          query('li:leave ~ li:not(:leave)', [
            animate( '300ms ease-out', style({ transform: 'translate3d(0, calc(-100% + 1px), 0)' })),
          ]),
        ]),
      ]),
    ]),
  ],
})
```

```
@Component({
  animations: [
    trigger('listAnimation', [
      transition(':increment', [
        group([
          query('li:enter', [
            style({ transform: 'translate3d(0, calc(-100% + 1px), 0)', 'z-index': -1  }),
            animate( '300ms ease-out', style({ transform: 'translate3d(0, 0, 0)' })),
          ]),
        ]),
      ]),
      transition(':decrement', [
        group([
          query('li:leave', [animate('250ms ease-out', style({ opacity: 0 }))]),
          query('li:leave ~ li:not(:leave)', [
            animate( '300ms ease-out', style({ transform: 'translate3d(0, calc(-100% + 1px), 0)' })),
          ]),
        ]),
      ]),
    ]),
  ],
})
```

```
@Component({
  animations: [
    trigger('listAnimation', [
      transition(':increment', [
        group([
          query('li:enter', [
            style({ transform: 'translate3d(0, calc(-100% + 1px), 0)', 'z-index': -1  }),
            animate( '300ms ease-out', style({ transform: 'translate3d(0, 0, 0)' })),
          ]),
        ]),
      ]),
      transition(':decrement', [
        group([
          query('li:leave', [animate('250ms ease-out', style({ opacity: 0 }))]),
          query('li:leave ~ li:not(:leave)', [
            animate( '300ms ease-out', style({ transform: 'translate3d(0, calc(-100% + 1px), 0)' })),
          ]),
        ]),
      ]),
    ]),
  ],
})
```

Animation Approach

Angular Animations

View Transitions

Angular Animations

Learn CSS

+

```
<ul>
   @for (todo of store.todos(); track todo.id) {
     <li>
       <p>{{ todo.name }}</p>
       <button (click)="removeTodo(todo)">🗑️</button>
     </li>
   }
 </ul>
```

```
<ul>
   @for (todo of store.todos(); track todo.id) {
     <li [style.view-transition-name]="'todo-item-'+ todo.id">
       <p>{{ todo.name }}</p>
       <button (click)="removeTodo(todo)">🗑️</button>
     </li>
   }
 </ul>
```

```
export class ViewTransitions {
  store = inject(TodoStoreService);

  addTodo() {
    document.startViewTransition(() => {
      this.store.addTodo();
    });
  }
  removeTodo(todo: Todo) {
    document.startViewTransition(() => {
      this.store.remove(todo);
    });
  }
}
```

IT JUST WORKS*

DON'T FORGET CHANGE DETECTION

```ts
export class TodoStoreService {
  private ref = inject(ApplicationRef);
  todos = signal<Array<Todo>>([]);

  addTodo() {
    this.todos.update((val) =>[
      ...val,
      { name: `new-item-${Date.now()}`, id: crypto.randomUUID() },
    ]);
    this.ref.tick();
  }
  remove(todo: Todo) {
    this.todos.update((state) =>
      state.filter((todoToFind: Todo) =>todoToFind.id !==todo.id),
    );
    this.ref.tick();
  }
}
```

```css
::view-transition-group(*) {
  animation-timing-function: ease-out;
  animation-duration: 300ms;
  mix-blend-mode: normal;
  z-index: 1;
  overflow: hidden;
}

::view-transition-group:last-child {
  z-index: -1;
}

::view-transition-new(*):only-child {
  animation: var(--animation-slide-in-down);
}

::view-transition-old(*):only-child {
  animation: var(--animation-fade-out);
}
```

```
::view-transition-new(*):only-child {
  animation: var(--animation-slide-in-down);
}

::view-transition-old(*):only-child {
  animation: var(--animation-fade-out);
}
```

Animation Approach

Angular Animations

View Transitions

View Transitions

Learn CSS

+

WHAT ABOUT THE ROUTER?

feat(router): Add feature to support the View Transitions API #51314

Closed **atscott** wants to merge 2 commits into `angular:main` from `atscott:viewtransitionrouter`

Conversation 25 | Commits 2 | Checks 17 | Files changed 12

atscott commented on Aug 9, 2023 — Contributor

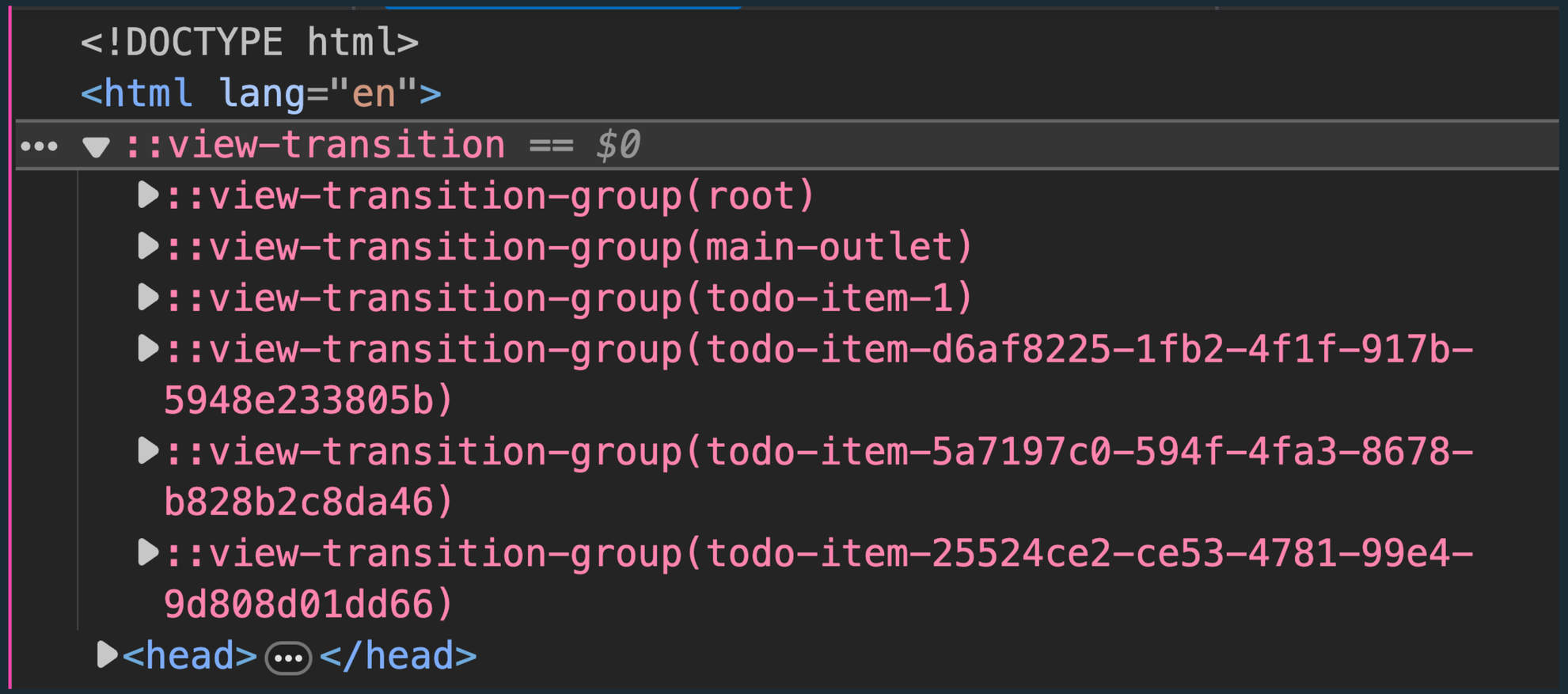
The View Transitions API enables easy animations when transitioning between different DOM states. This commit adds an opt-in feature to the Router which runs the component activation and deactivation logic in the `document.startViewTransition` callback. If the browser does not support this API, route activation and deactivation will happen synchronously.

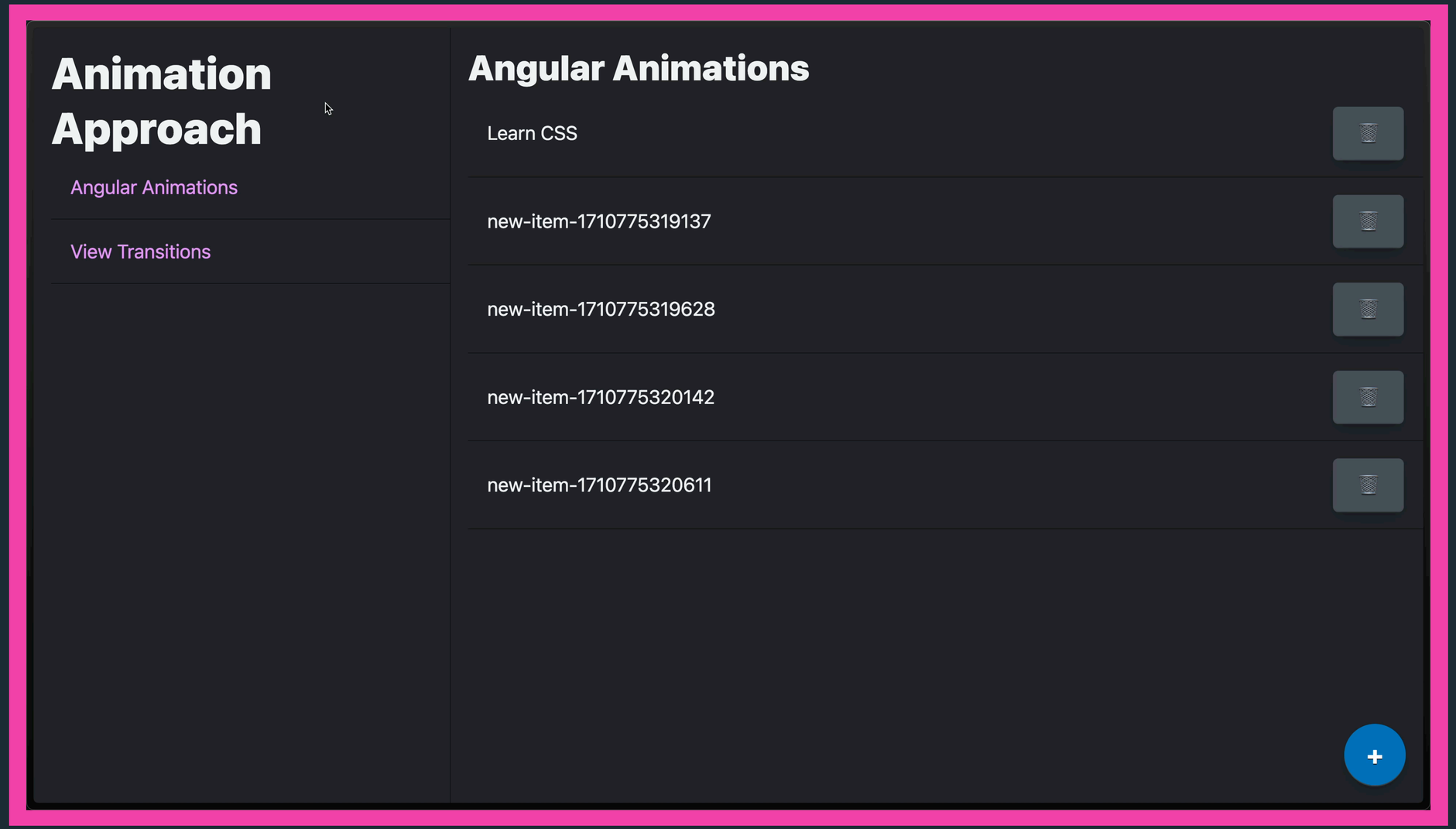
resolves #49401

👍 11 🎉 14 ❤️ 20 🚀 6

```ts
import { provideRouter, withViewTransitions } from "@angular/router";

export const appConfig: ApplicationConfig = {
  providers: [
    provideRouter(routes, withViewTransitions()),
  ],
};
```


<!DOCTYPE html>
<html lang="en">
::view-transition == $0
::view-transition-group(root)
::view-transition-group(main-outlet)
::view-transition-group(todo-item-1)
::view-transition-group(todo-item-d6af8225-1fb2-4f1f-917b-5948e233805b)
::view-transition-group(todo-item-5a7197c0-594f-4fa3-8678-b828b2c8da46)
::view-transition-group(todo-item-25524ce2-ce53-4781-99e4-9d808d01dd66)
<head> </head>


Animation Approach
Angular Animations
View Transitions
Angular Animations
Learn CSS
new-item-1710775319137
new-item-1710775319628
new-item-1710775320142
new-item-1710775320611
+

Scoped Transitions

This doc is a summary of early design explorations to scope transitions to a DOM sub-tree. It uses the pseudo-element tree to describe the UA generated tree to refer to the existing spec. A separate doc considers an alternate Shadow DOM implementation.

The following is a rough API sketch for how a developer would trigger the transition rooted on an element (other than the document's root element).

```
element.startViewTransition(() => {
  // Update the DOM somehow.
});
```

The element becomes the **scoped-transition-root** for the transition. It's the element that will host the pseudo-element tree and page-transition-container sub-trees will be created for descendants that have a `page-transition-tag`. For example, the following page:

```
<style>
  .outer {
    page-transition-tag: outer;
    contain: layout;
  }
  .inner {
    page-transition-tag: inner;
    contain: layout;
  }
```

SCOPED TRANSITIONS EXPLAINER

VIEW TRANSITIONS WILL BE THE FUTURE

ANGULAR ANIMATIONS ARE STILL VALUABLE

VIEW TRANSITIONS

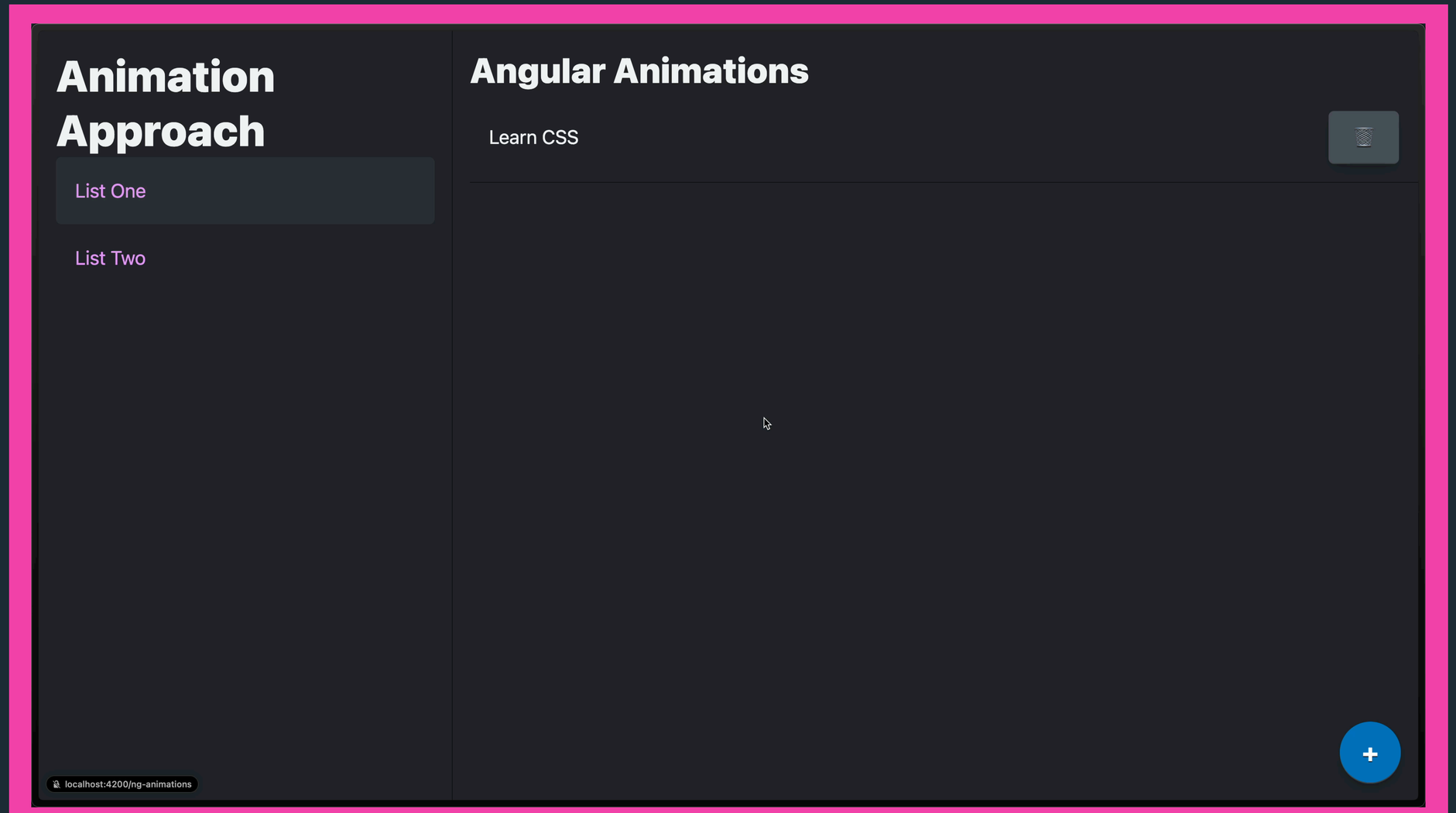
ANGULAR ANIMATIONS

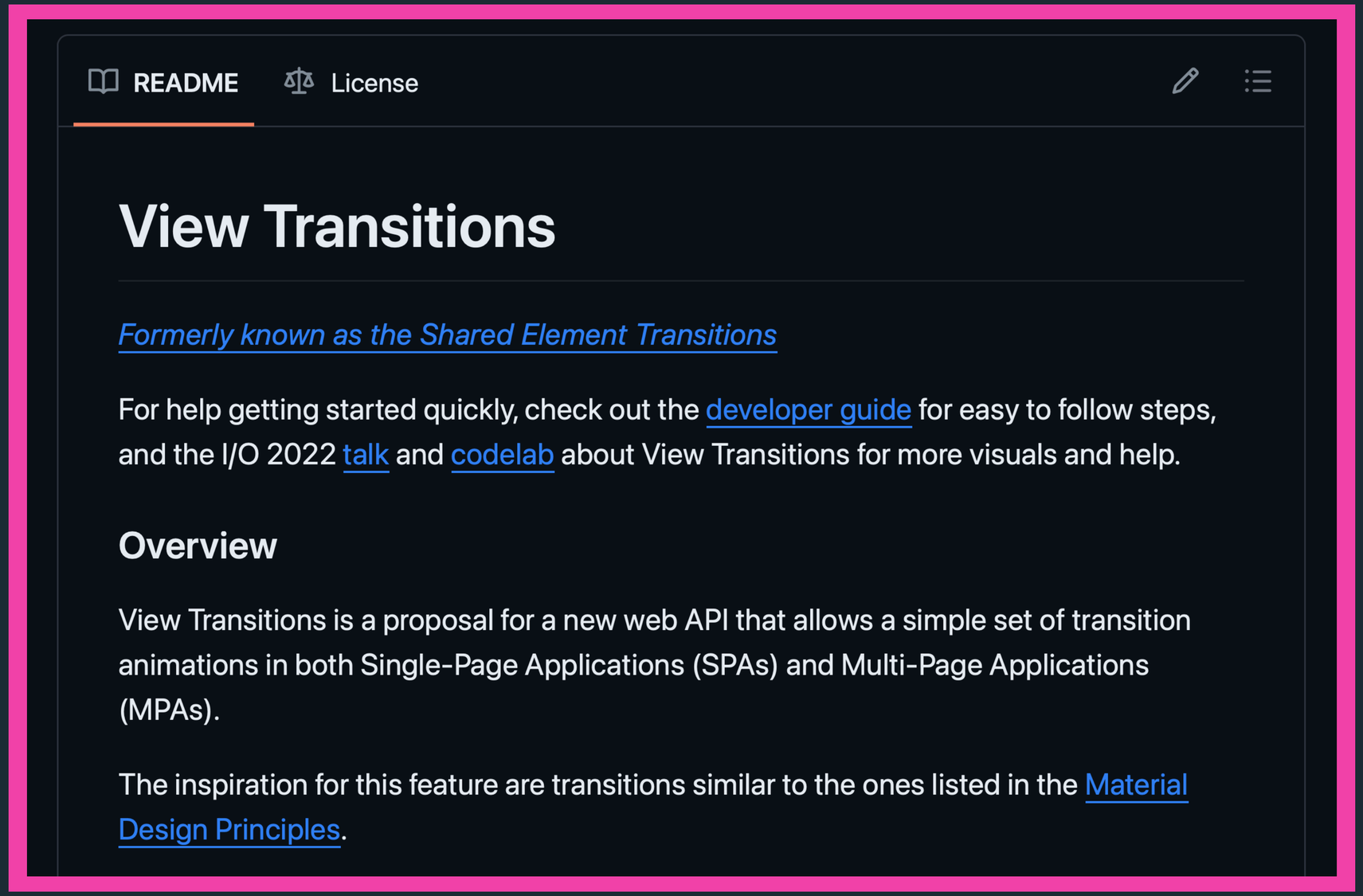
VIEW TRANSITIONS

TOP LEVEL CHANGES,
EFFECTS THE WHOLE DOM

ANGULAR ANIMATIONS

ISOLATED INTERACTIONS,
SCOPED TO A COMPONENT


Animation Approach
List One
List Two
Angular Animations
Learn CSS
+
localhost:4200/ng-animations



GITHUB.COM/WICG/VIEW-TRANSITIONS

Fin