OMI 响应式 Web Components

@ dntzhang

🔗 omijs.org

当耐特

dntzhang(张磊)

Creater of OMI, Cax, Pasition, AlloyFinger, PhyTouch, Westore, Qone, Netural, Curvejs

Tencent

Organizations

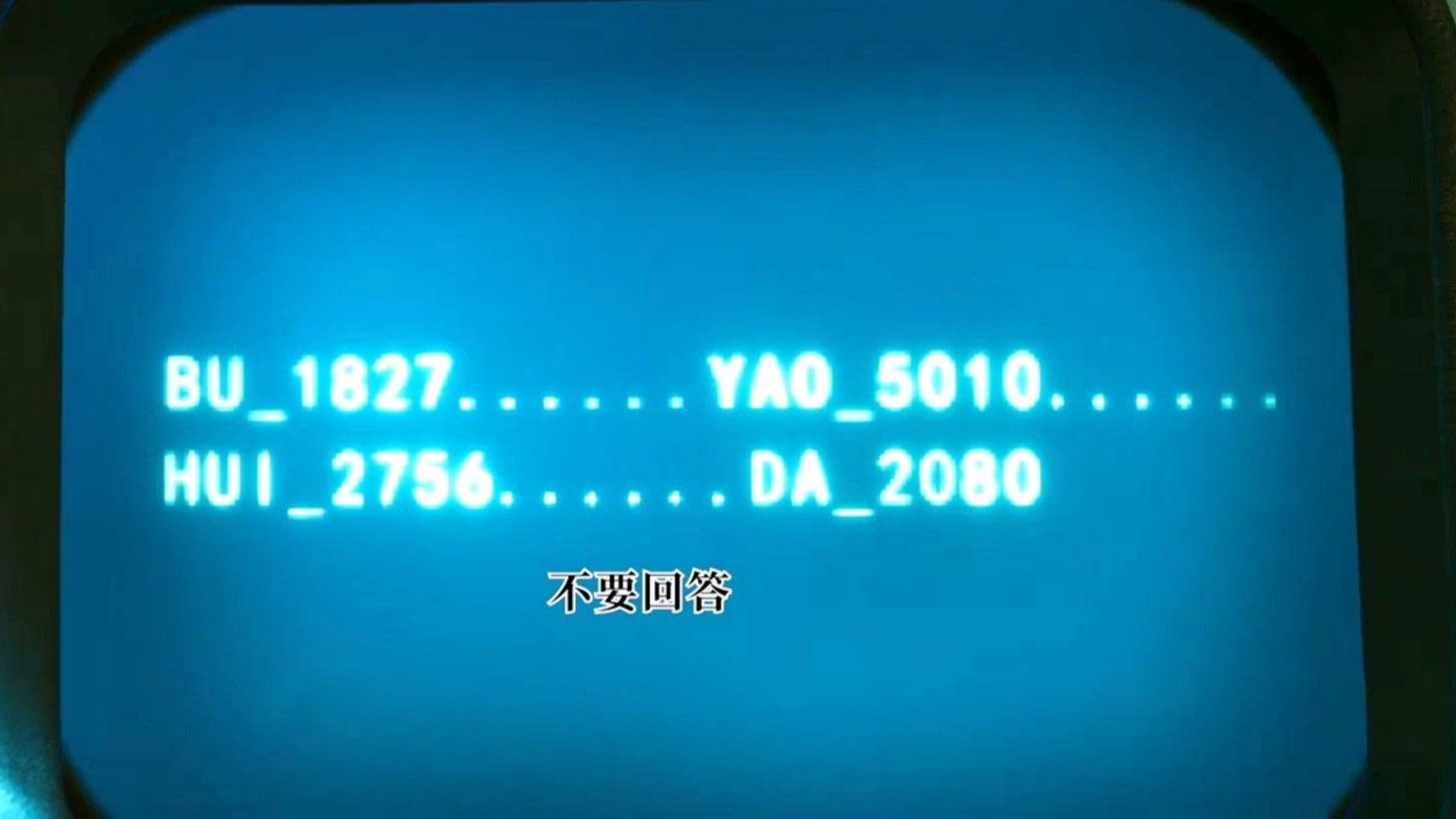
Part 1

信号

3BODY
三体
中国科幻的起源

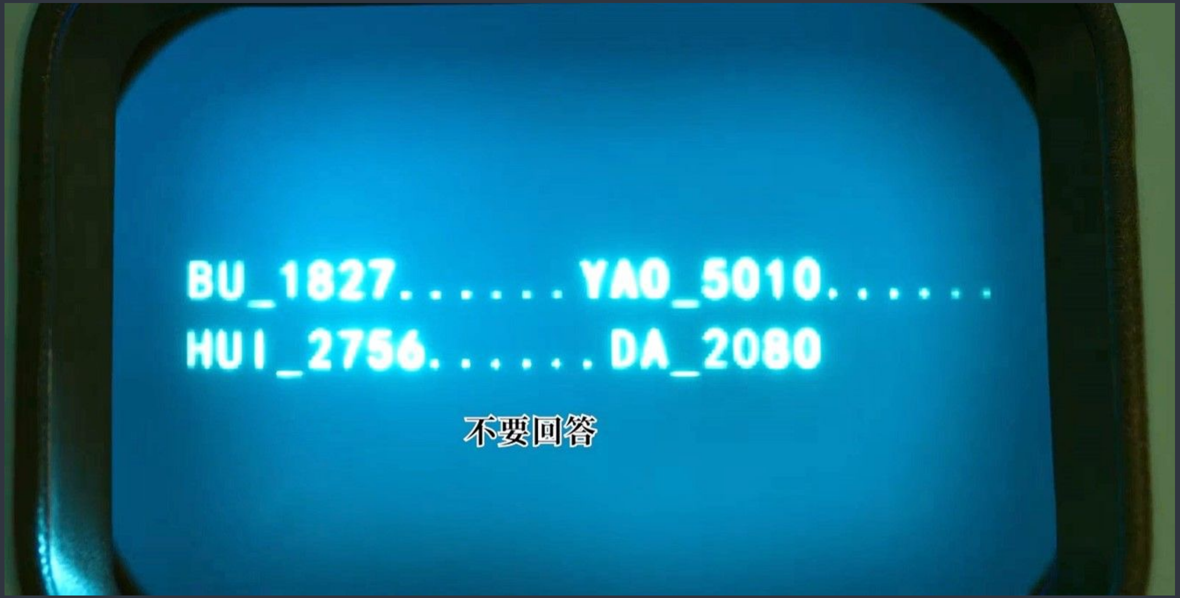
接收信号

阅读信号值



```
BU_1827.......YAO_5010.......
HUI_2756.......DA_2080
```

不要回答



```
BU_1827......YAO_5010......
HUI_2756......DA_2080
```

不要回答

信号

```
BU_1827......YAO_5010......
HUI_2756......DA_2080
```

不要回答

信号值

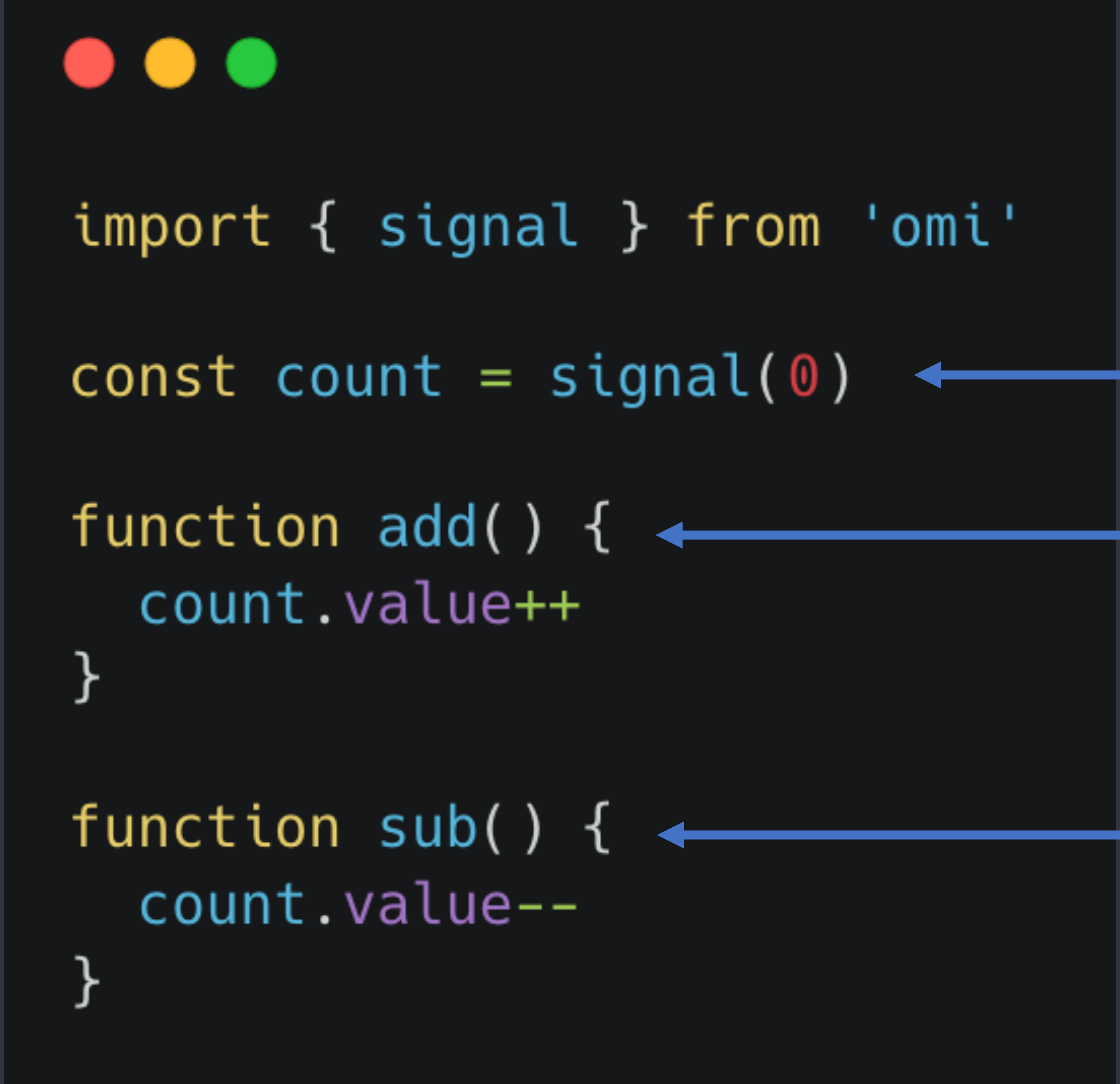
信号 === 信号

信号 === signal

信号值 === 信号值

信号值 === 信号.value

信号值 === signal.value

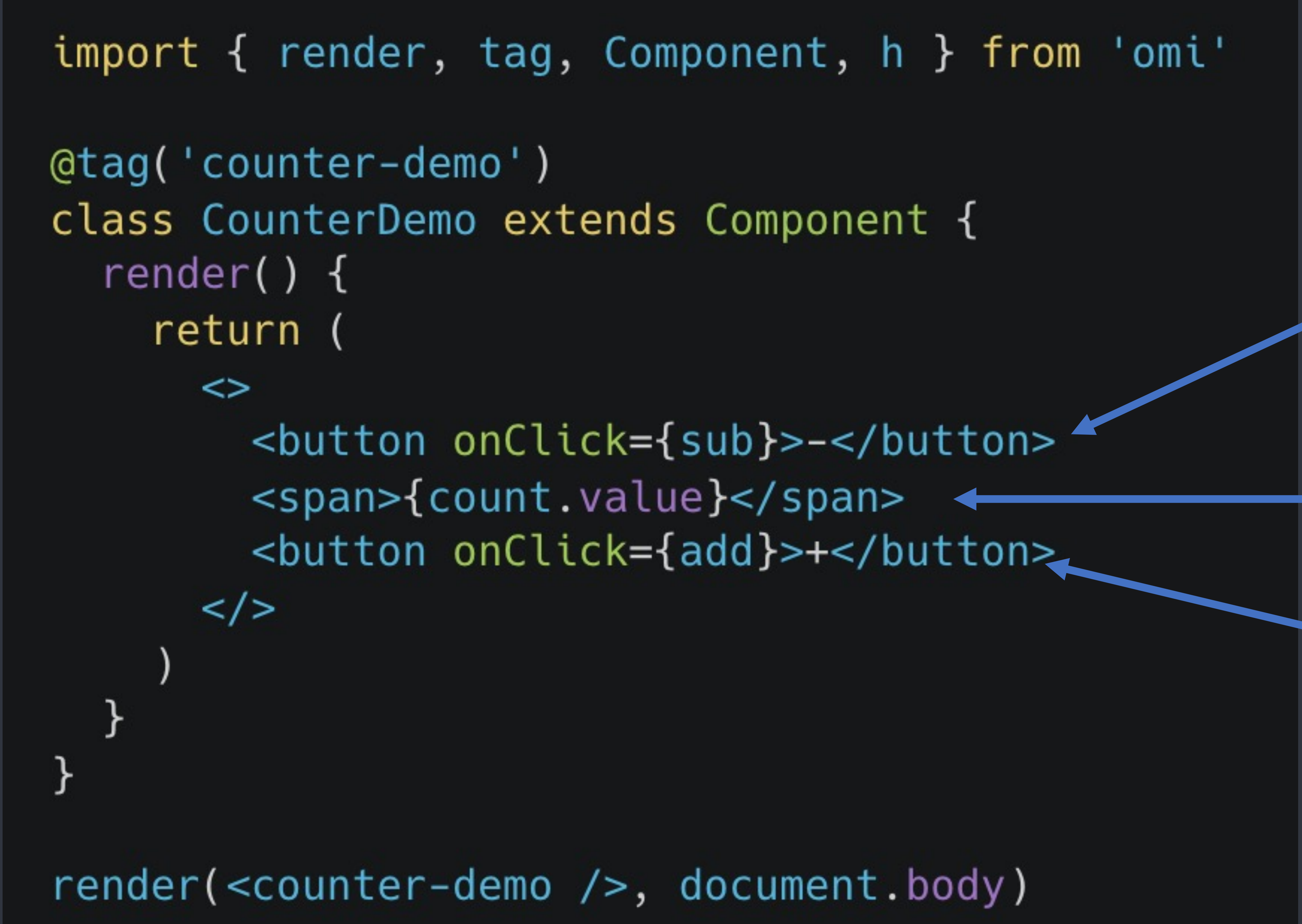


```
import { signal } from 'omi'

const count = signal(0)

function add() {
  count.value++
}

function sub() {
  count.value--
}
```

定义信号

定义操作信号值方法

定义操作信号值方法

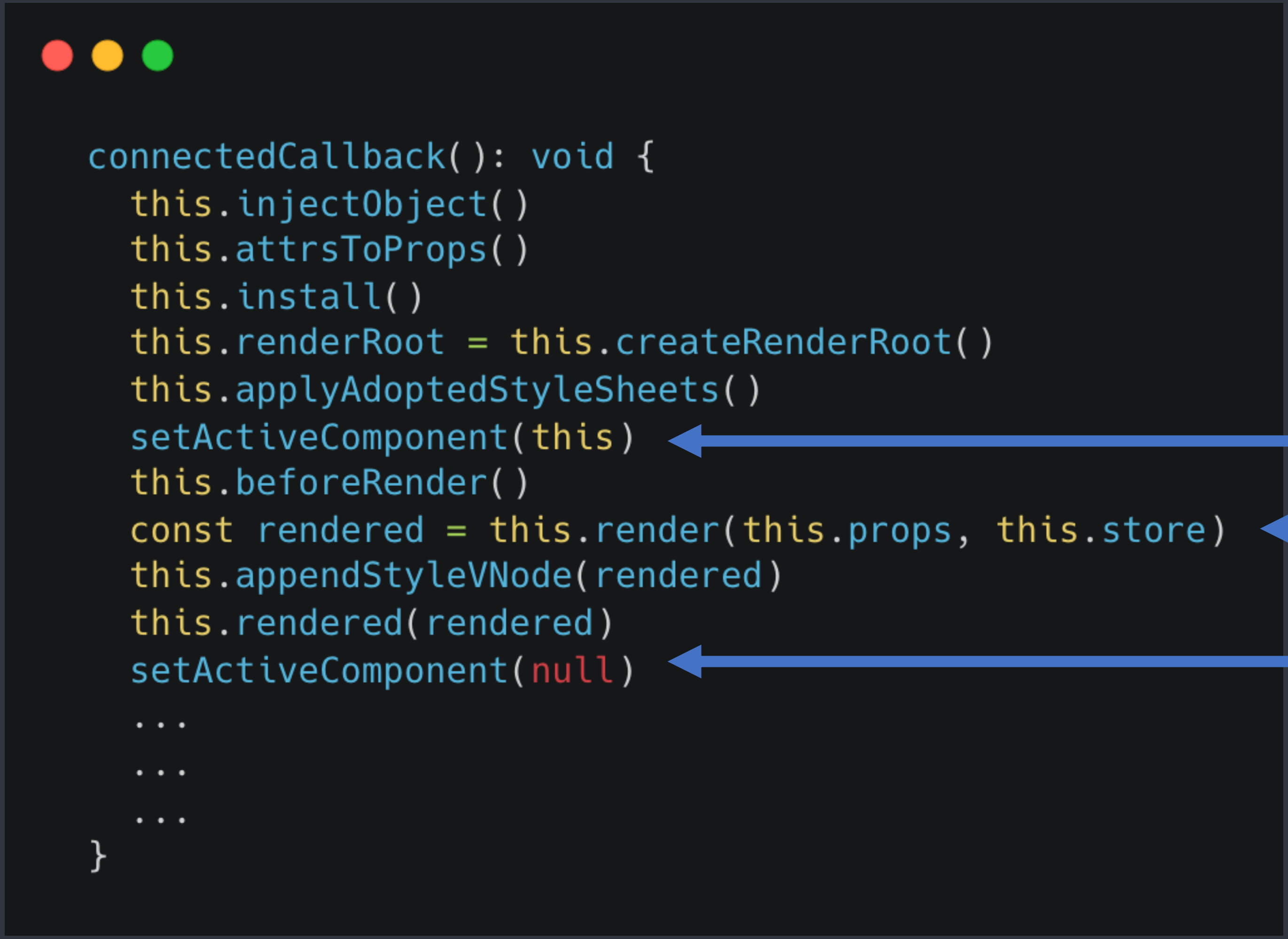


```
import { render, tag, Component, h } from 'omi'

@tag('counter-demo')
class CounterDemo extends Component {
  render() {
    return (
      <>
        <button onClick={sub}>-</button>
        <span>{count.value}</span>
        <button onClick={add}>+</button>
      </>
    )
  }
}

render(<counter-demo />, document.body)
```

操作信号值
自动更新UI

读取信号

操作信号值
自动更新UI



```
connectedCallback(): void {
  this.injectObject()
  this.attrsToProps()
  this.install()
  this.renderRoot = this.createRenderRoot()
  this.applyAdoptedStyleSheets()
  setActiveComponent(this)
  this.beforeRender()
  const rendered = this.render(this.props, this.store)
  this.appendStyleVNode(rendered)
  this.rendered(rendered)
  setActiveComponent(null)
  ...
  ...
  ...
}
```

读取 .value 前

这里会读取信号 .value

读取 .value 后

```
export function signal<T>(initialValue: T): SignalValue<T> {
  let value = initialValue
  const deps = new Set<EffectFn>()
  const depsComponents = new Set<Component>()

  return new Proxy({} as SignalValue<T>, {
    get(_, prop: keyof SignalValue<T>) {
      if (prop === 'value') {
        if (activeEffect) {
          deps.add(activeEffect)
        }
        const component = getActiveComponent()
        if (component) depsComponents.add(component)
        return value
      }
      if (prop === 'peek') return () => value
      if (prop === 'update')
        return () => {
          value = value
          deps.forEach((fn) => fn())
          depsComponents.forEach((component) => component.queuedUpdate())
        }
    },
    set(_, prop: keyof SignalValue<T>, newValue: T) {
      if (prop === 'value') {
        if (
          !isPrimitive(value) ||
          !isPrimitive(newValue) ||
          value !== newValue
        ) {
          value = newValue
          deps.forEach((fn) => fn())
          depsComponents.forEach((component) => component.queuedUpdate())
        }
        return true
      }
      return false
    },
  })
}
```

render里读取 value 进入 get

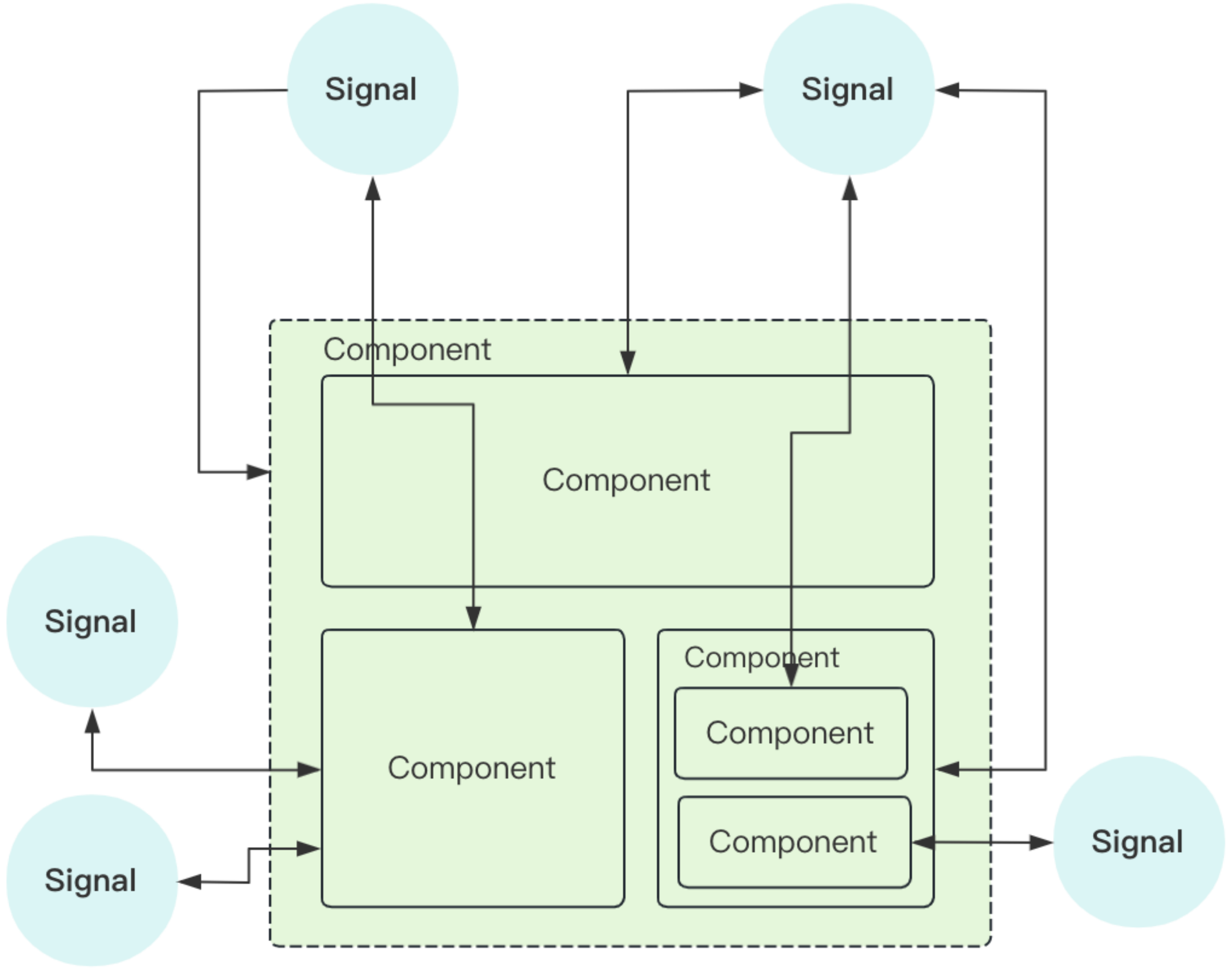
获取 acitve component 放入依赖

可 peek【偷窥】value，不产生依赖

非值类型可手动调用信号的 update 出发组件更新

判断是否是值类型，如果是，进行 value 对比

set 会自动调用依赖组件的更新方法



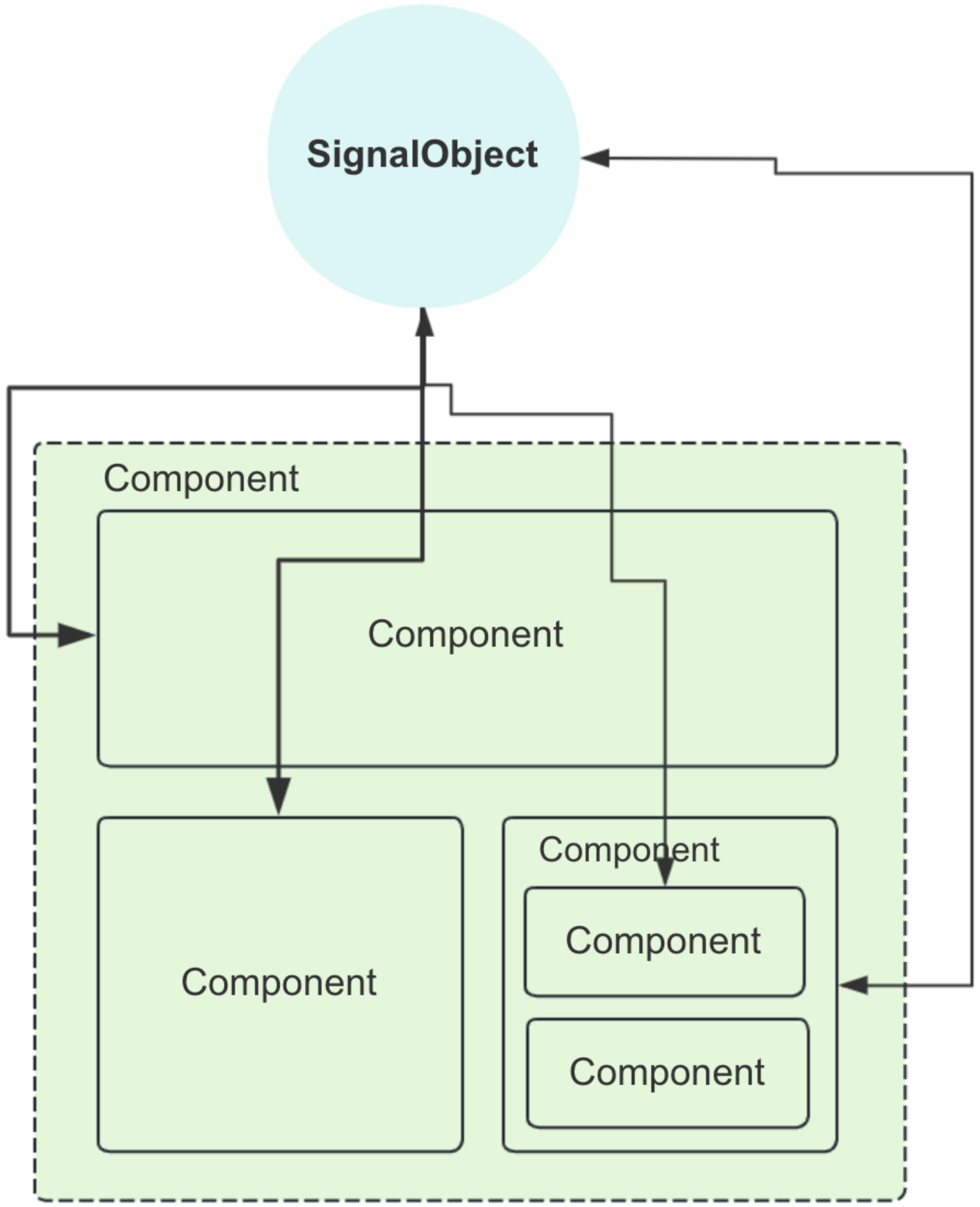
OMI Signal，按需自动化最小范围更新

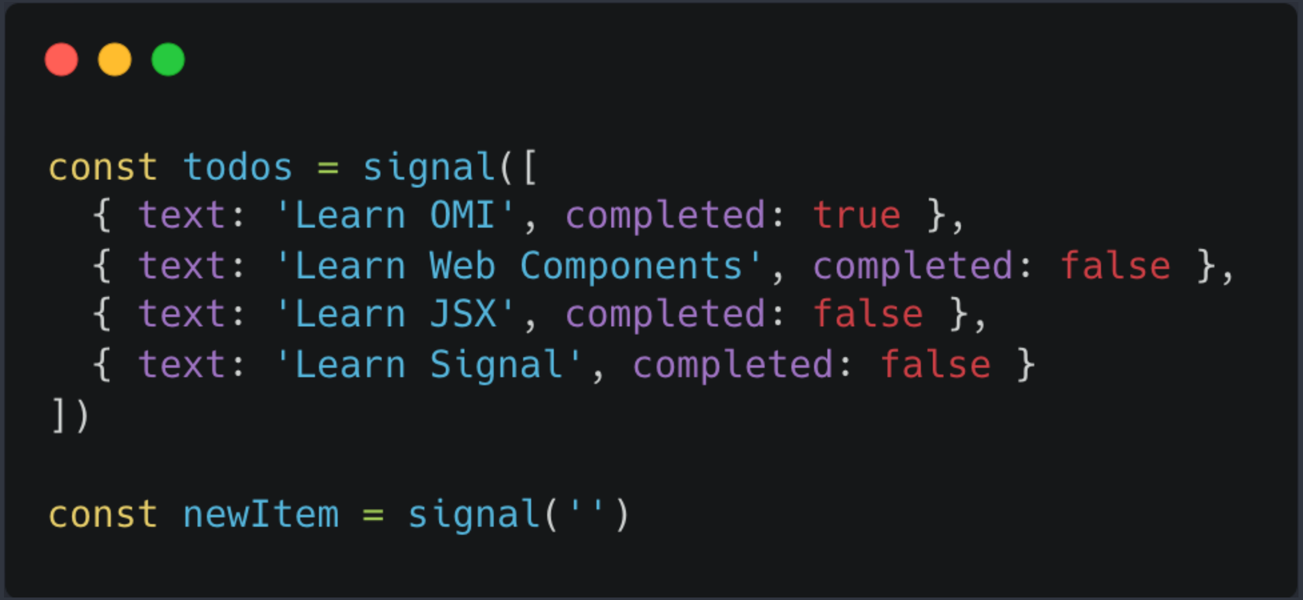
```
const todos = signal([
  { text: 'Learn OMI', completed: true },
  { text: 'Learn Web Components', completed: false },
  { text: 'Learn JSX', completed: false },
  { text: 'Learn Signal', completed: false }
])
```

```
function addTodo() {
  todos.value.push({
    text: newItem.value,
    completed: false
  })
  todos.update()
}
```

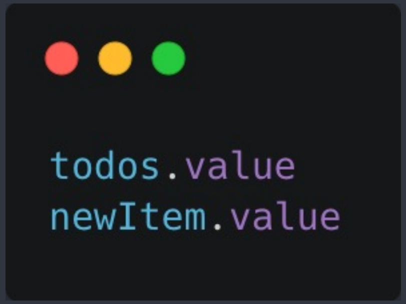
效果一样

```
function addTodo() {
  todos.value = [
    ...todos.value,
    {
      text: newItem.value,
      completed: false
    }
  ]
}
```


SignalObject
Component
Component
Component
Component
Component
Component

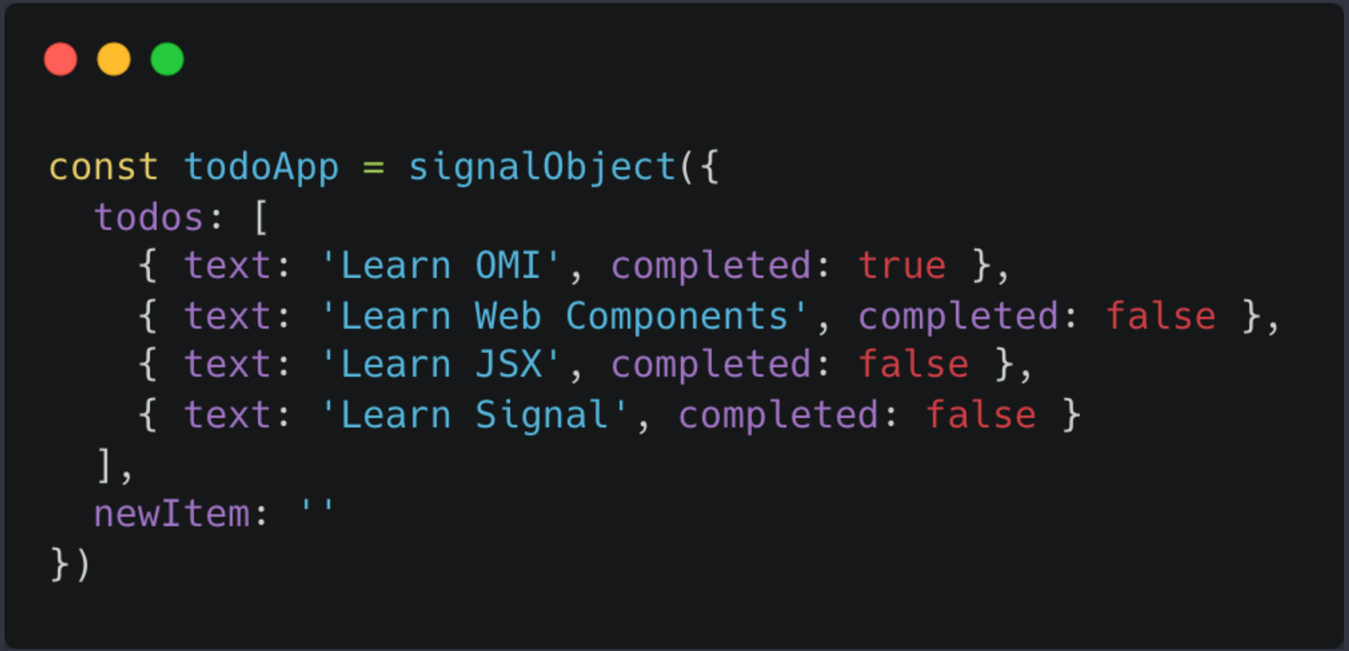


```
const todos = signal([
  { text: 'Learn OMI', completed: true },
  { text: 'Learn Web Components', completed: false },
  { text: 'Learn JSX', completed: false },
  { text: 'Learn Signal', completed: false }
])

const newItem = signal('')
```

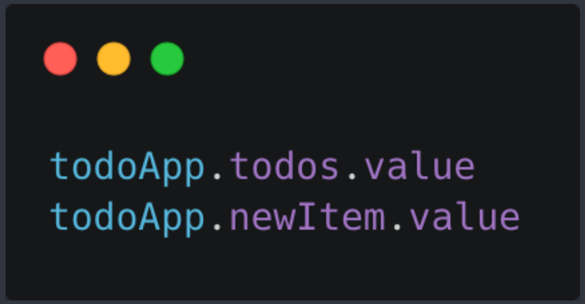


```
todos.value
newItem.value
```

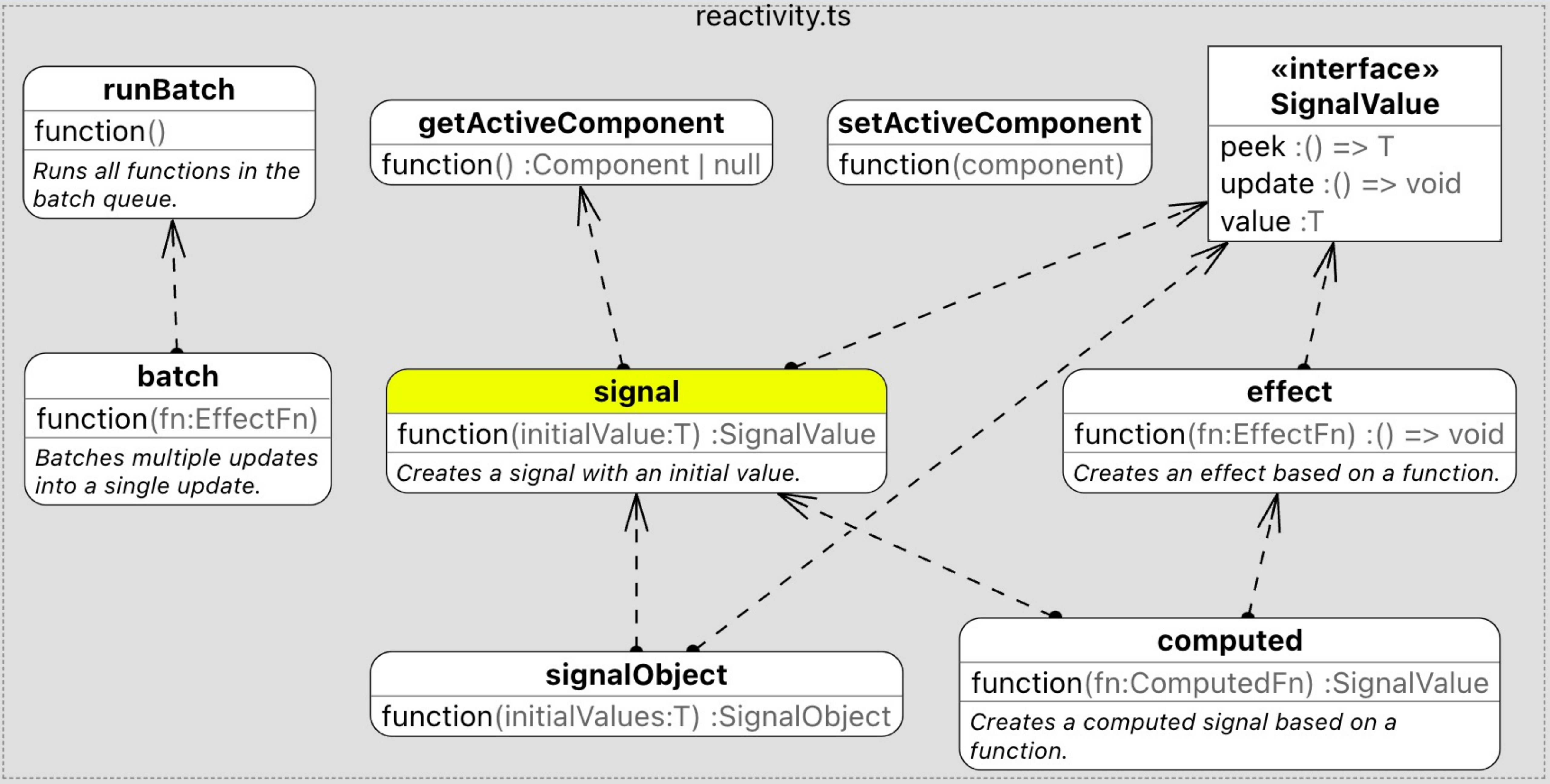
效果一样



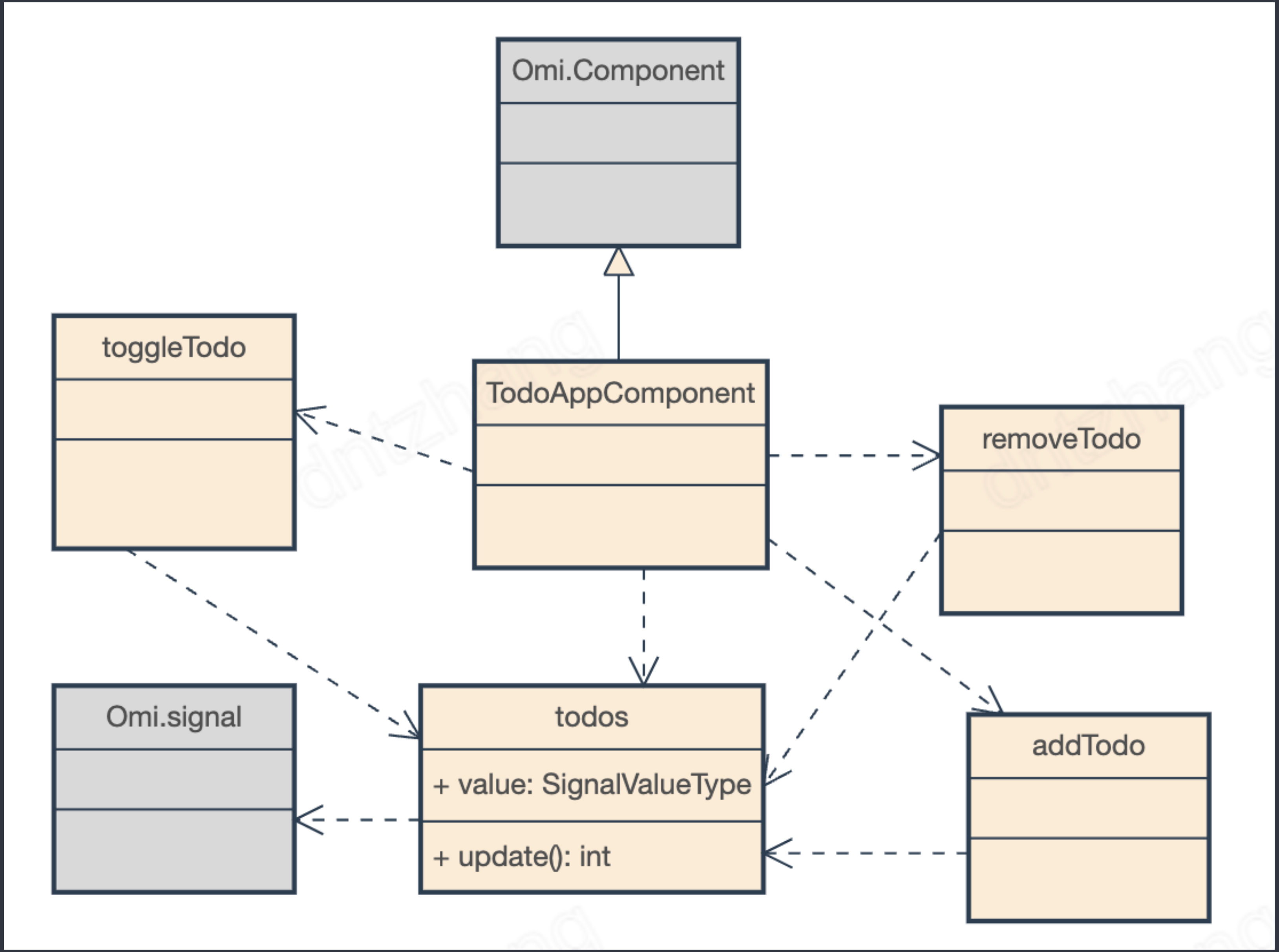
```
const todoApp = signalObject({
  todos: [
    { text: 'Learn OMI', completed: true },
    { text: 'Learn Web Components', completed: false },
    { text: 'Learn JSX', completed: false },
    { text: 'Learn Signal', completed: false }
  ],
  newItem: ''
})
```

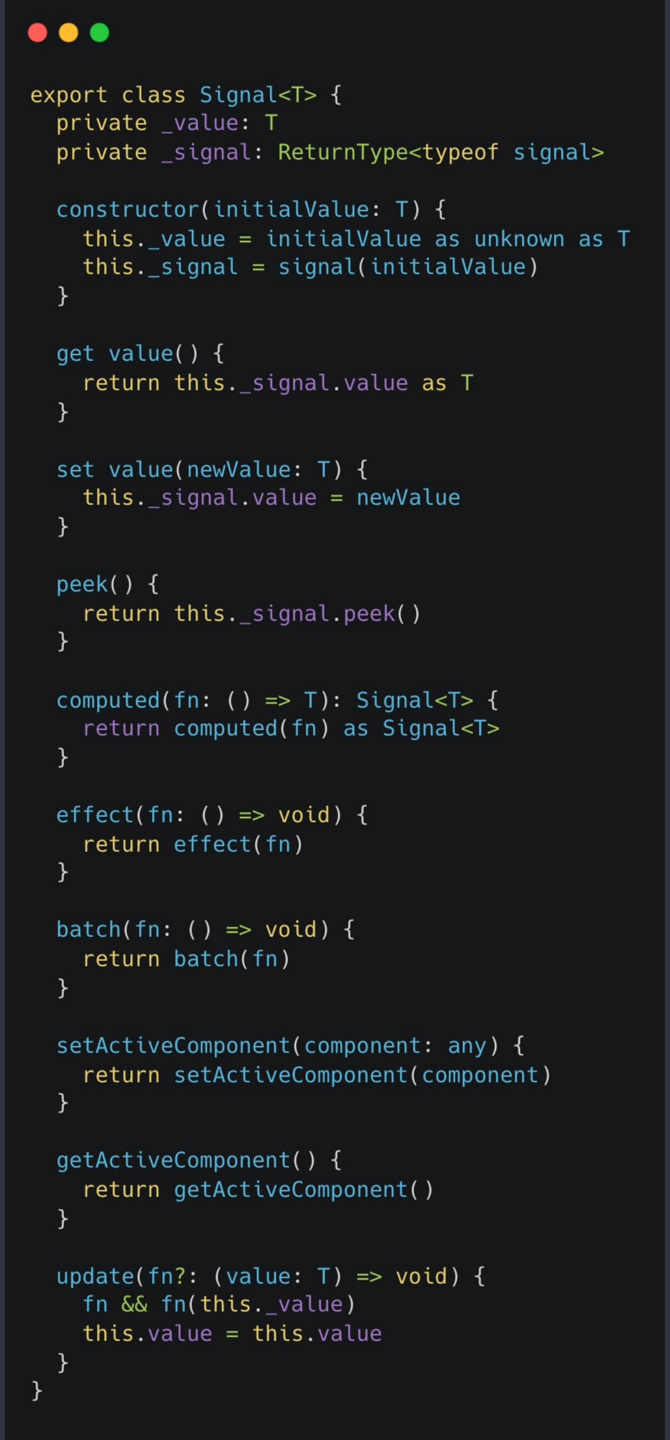


```
todoApp.todos.value
todoApp.newItem.value
```

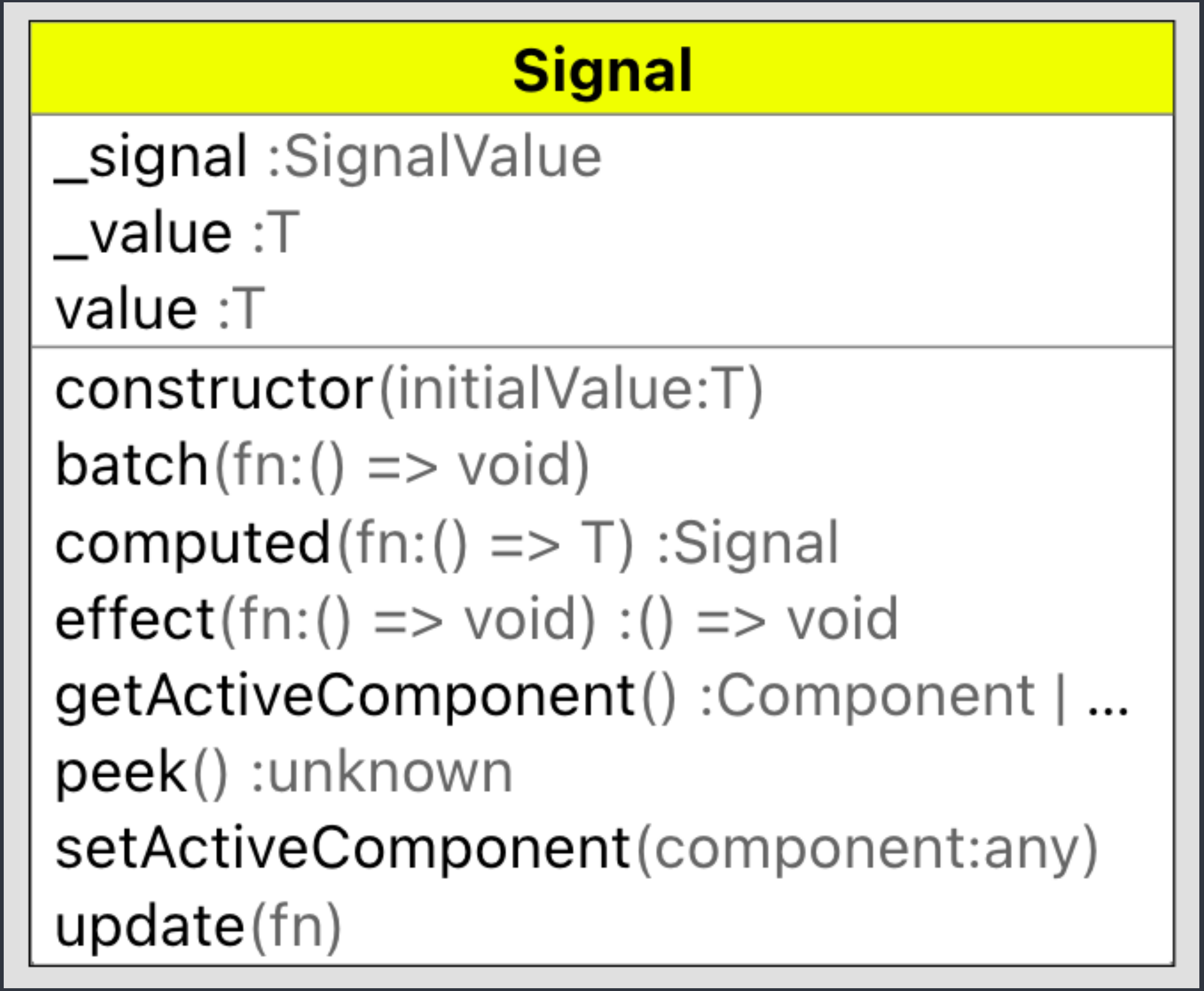


reactivity.ts
runBatch
function()
Runs all functions in the batch queue.
getActiveComponent
function() :Component | null
setActiveComponent
function(component)
«interface» SignalValue
peek :() => T
update :() => void
value :T
batch
function(fn:EffectFn)
Batches multiple updates into a single update.
signal
function(initialValue:T) :SignalValue
Creates a signal with an initial value.
effect
function(fn:EffectFn) :() => void
Creates an effect based on a function.
signalObject
function(initialValues:T) :SignalObject
computed
function(fn:ComputedFn) :SignalValue
Creates a computed signal based on a function.


Omi.Component
TodoAppComponent
toggleTodo
removeTodo
Omi.signal
todos
+ value: SignalValueType
+ update(): int
addTodo



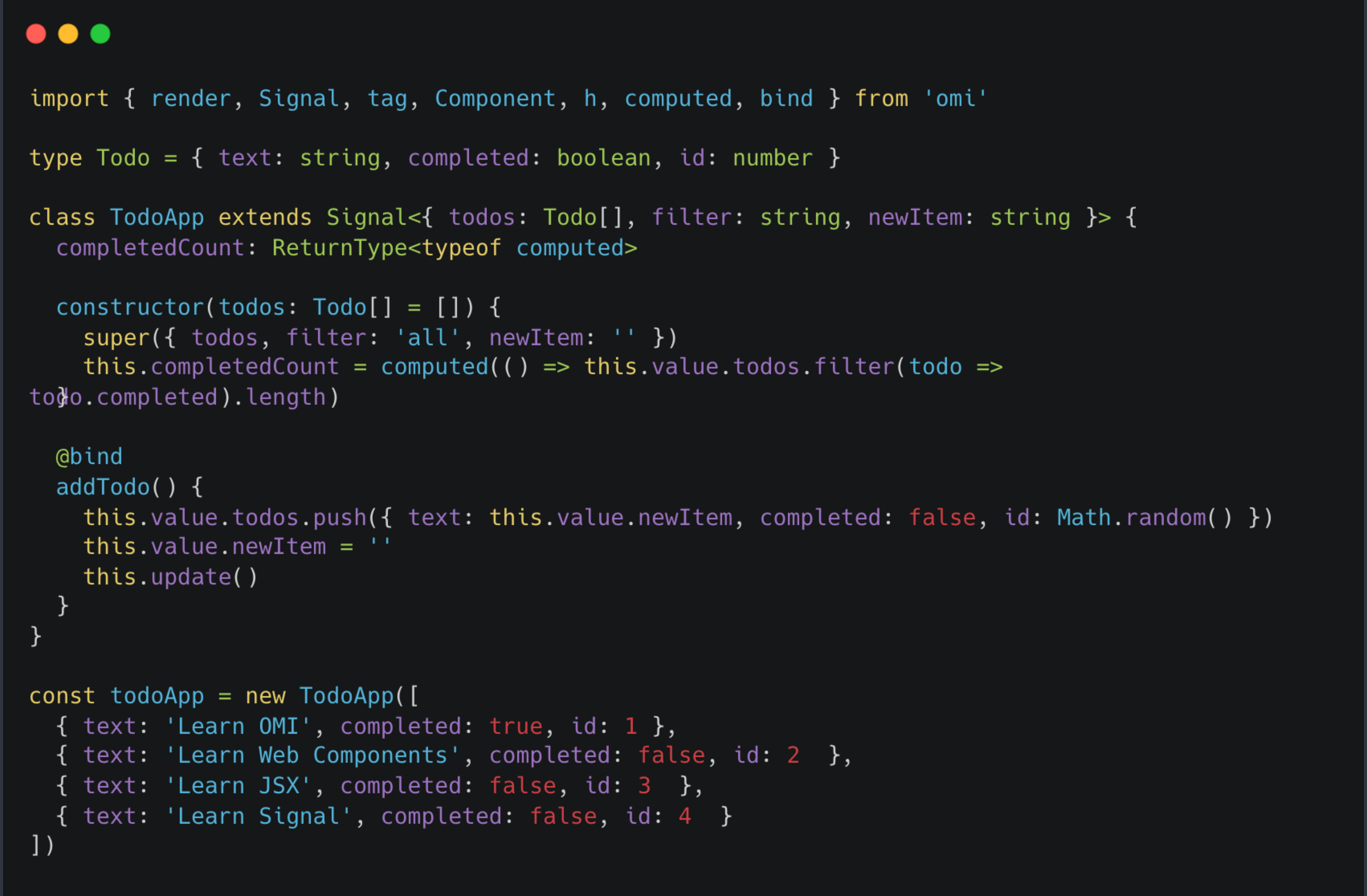
```typescript
export class Signal<T> {
  private _value: T
  private _signal: ReturnType<typeof signal>

  constructor(initialValue: T) {
    this._value = initialValue as unknown as T
    this._signal = signal(initialValue)
  }

  get value() {
    return this._signal.value as T
  }

  set value(newValue: T) {
    this._signal.value = newValue
  }

  peek() {
    return this._signal.peek()
  }

  computed(fn: () => T): Signal<T> {
    return computed(fn) as Signal<T>
  }

  effect(fn: () => void) {
    return effect(fn)
  }

  batch(fn: () => void) {
    return batch(fn)
  }

  setActiveComponent(component: any) {
    return setActiveComponent(component)
  }

  getActiveComponent() {
    return getActiveComponent()
  }

  update(fn?: (value: T) => void) {
    fn && fn(this._value)
    this.value = this.value
  }
}
```



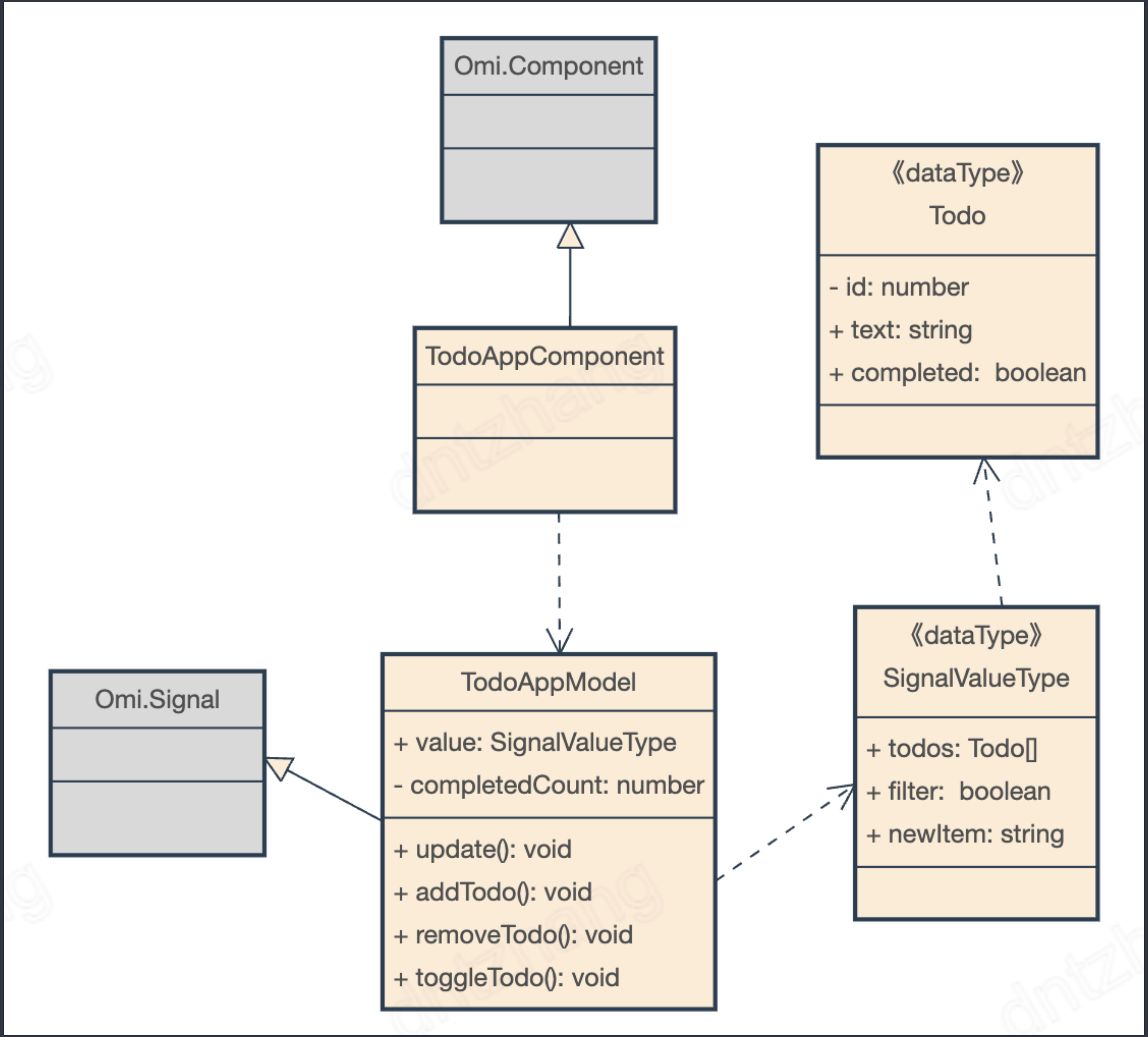
Signal

_signal :SignalValue
_value :T
value :T

constructor(initialValue:T)
batch(fn:() => void)
computed(fn:() => T) :Signal
effect(fn:() => void) :() => void
getActiveComponent() :Component | ...
peek() :unknown
setActiveComponent(component:any)
update(fn)

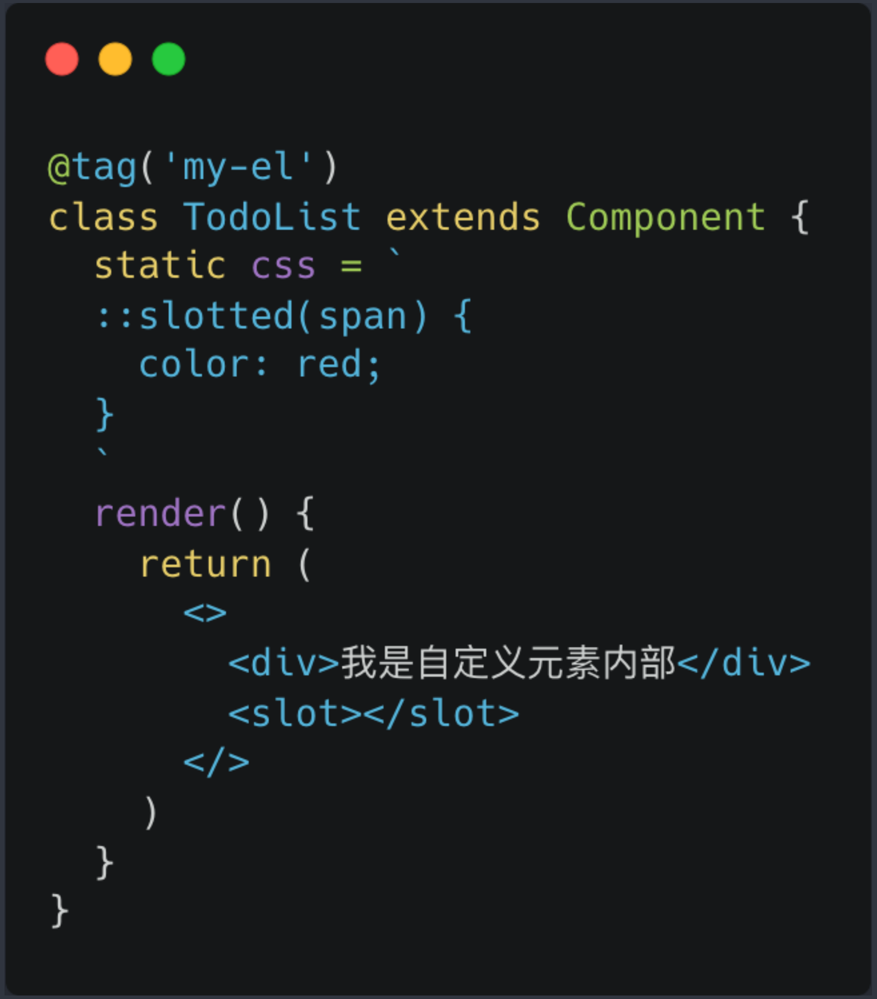


```
import { render, Signal, tag, Component, h, computed, bind } from 'omi'

type Todo = { text: string, completed: boolean, id: number }

class TodoApp extends Signal<{ todos: Todo[], filter: string, newItem: string }> {
  completedCount: ReturnType<typeof computed>

  constructor(todos: Todo[] = []) {
    super({ todos, filter: 'all', newItem: '' })
    this.completedCount = computed(() => this.value.todos.filter(todo =>
todo.completed).length)
  }

  @bind
  addTodo() {
    this.value.todos.push({ text: this.value.newItem, completed: false, id: Math.random() })
    this.value.newItem = ''
    this.update()
  }
}

const todoApp = new TodoApp([
  { text: 'Learn OMI', completed: true, id: 1 },
  { text: 'Learn Web Components', completed: false, id: 2  },
  { text: 'Learn JSX', completed: false, id: 3  },
  { text: 'Learn Signal', completed: false, id: 4  }
])
```

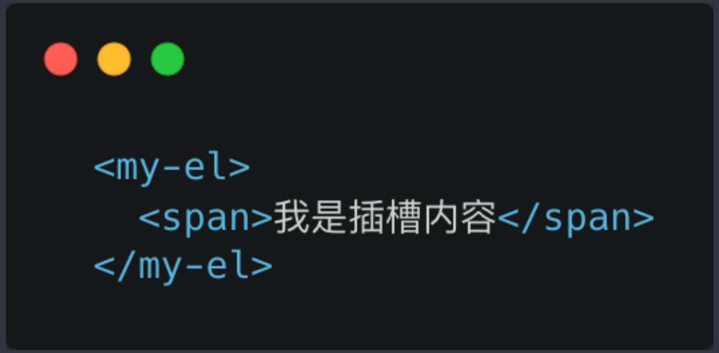

Omi.Component
TodoAppComponent
《dataType》
Todo
- id: number
+ text: string
+ completed: boolean
Omi.Signal
TodoAppModel
+ value: SignalValueType
- completedCount: number
+ update(): void
+ addTodo(): void
+ removeTodo(): void
+ toggleTodo(): void
《dataType》
SignalValueType
+ todos: Todo[]
+ filter: boolean
+ newItem: string

Part 2

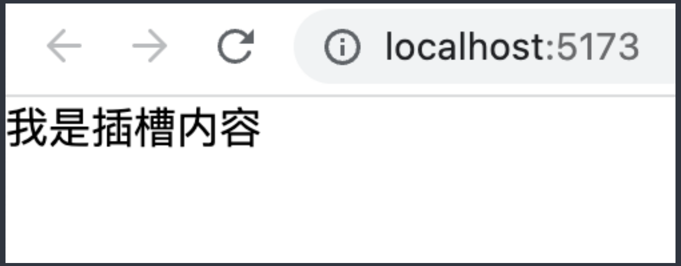
技巧



```
@tag('my-el')
class TodoList extends Component {
  static css = `
  ::slotted(span) {
    color: red;
  }
  `
  render() {
    return (
      <>
        <div>我是自定义元素内部</div>
        <slot></slot>
      </>
    )
  }
}
```



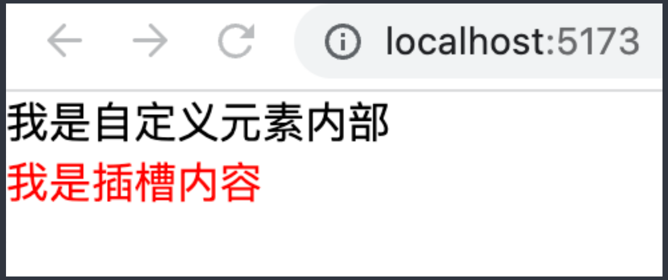
```
<my-el>
  <span>我是插槽内容</span>
</my-el>
```


localhost:5173

我是插槽内容

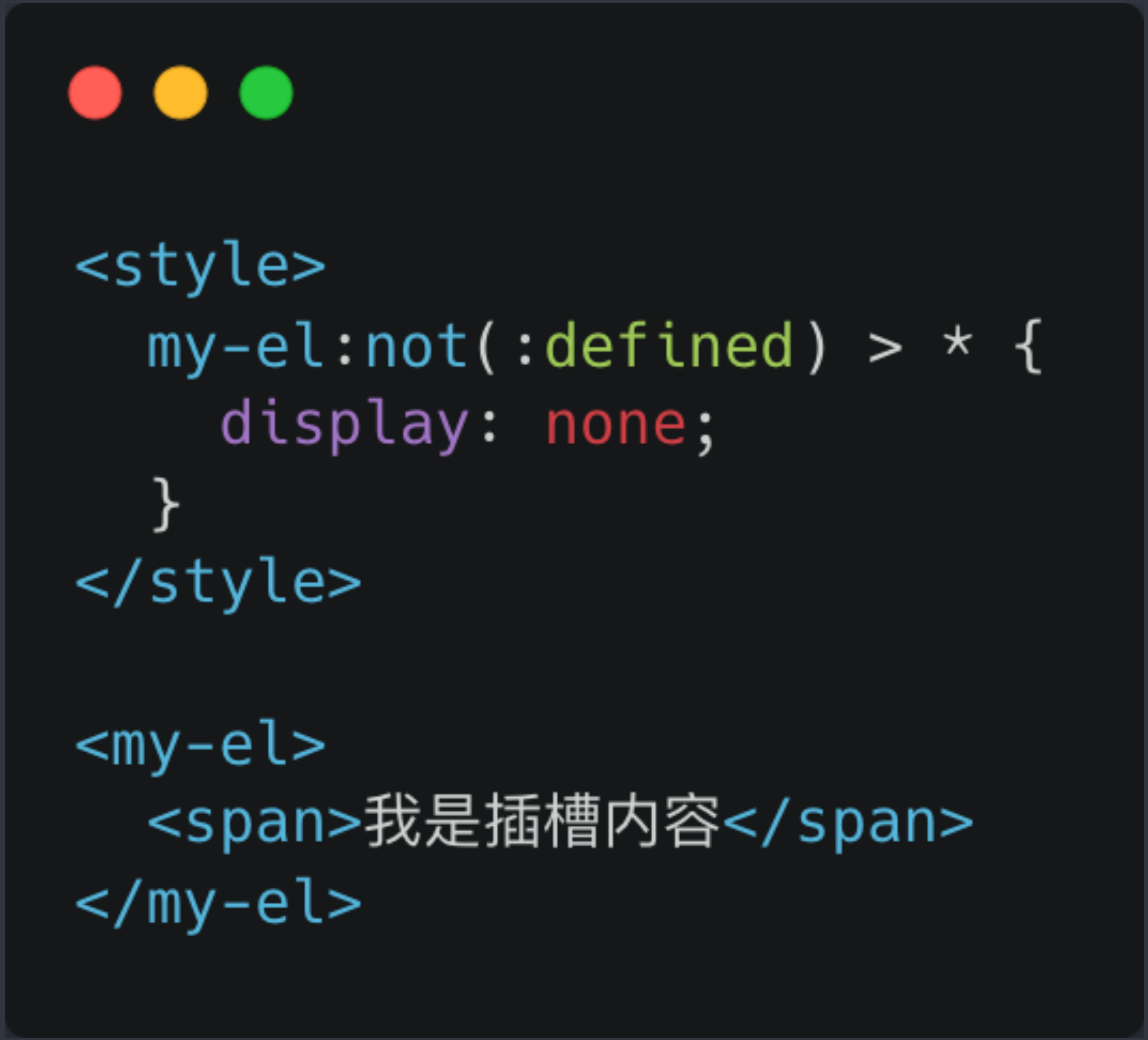
元素定义前

闪烁


localhost:5173

我是自定义元素内部
我是插槽内容

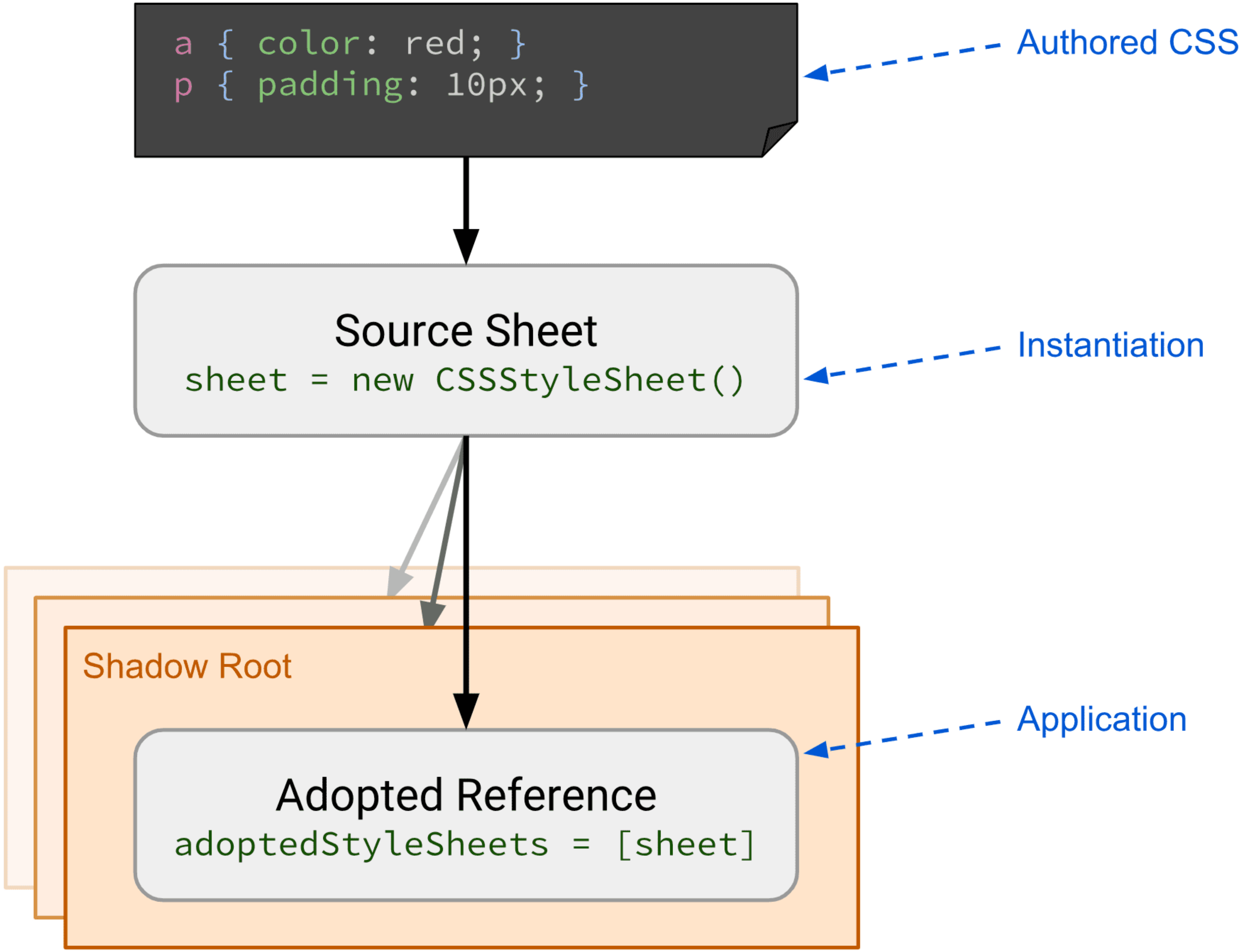
元素定义后



```
<style>
  my-el:not(:defined) > * {
    display: none;
  }
</style>

<my-el>
  <span>我是插槽内容</span>
</my-el>
```

不再闪烁

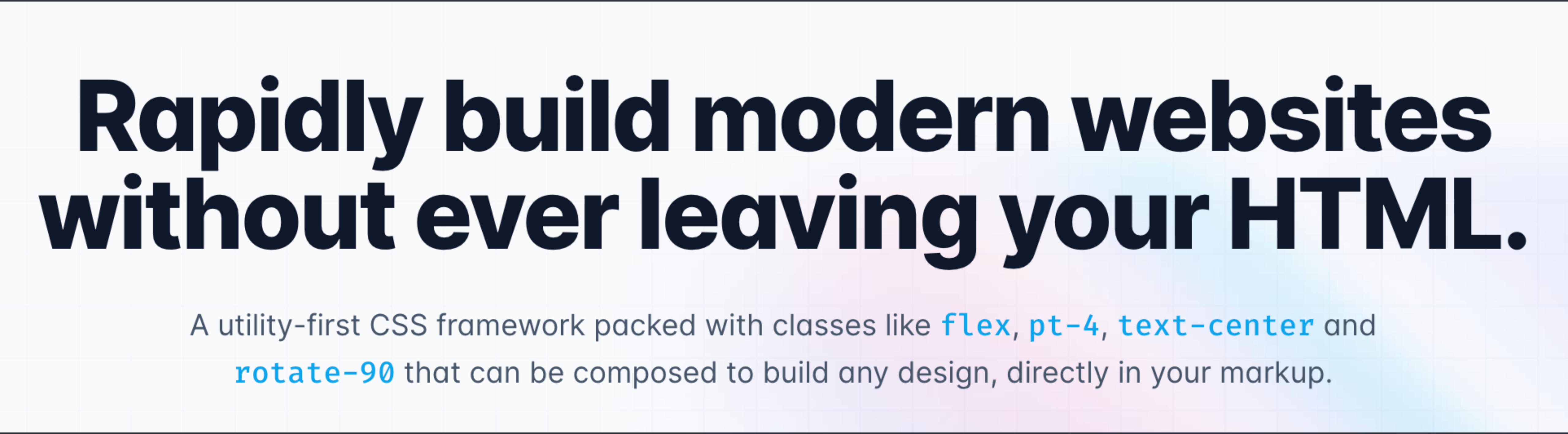
泡面和火腿的关系



```
a { color: red; }
p { padding: 10px; }
```

```
sheet = new CSSStyleSheet()
```

```
adoptedStyleSheets = [sheet]
```



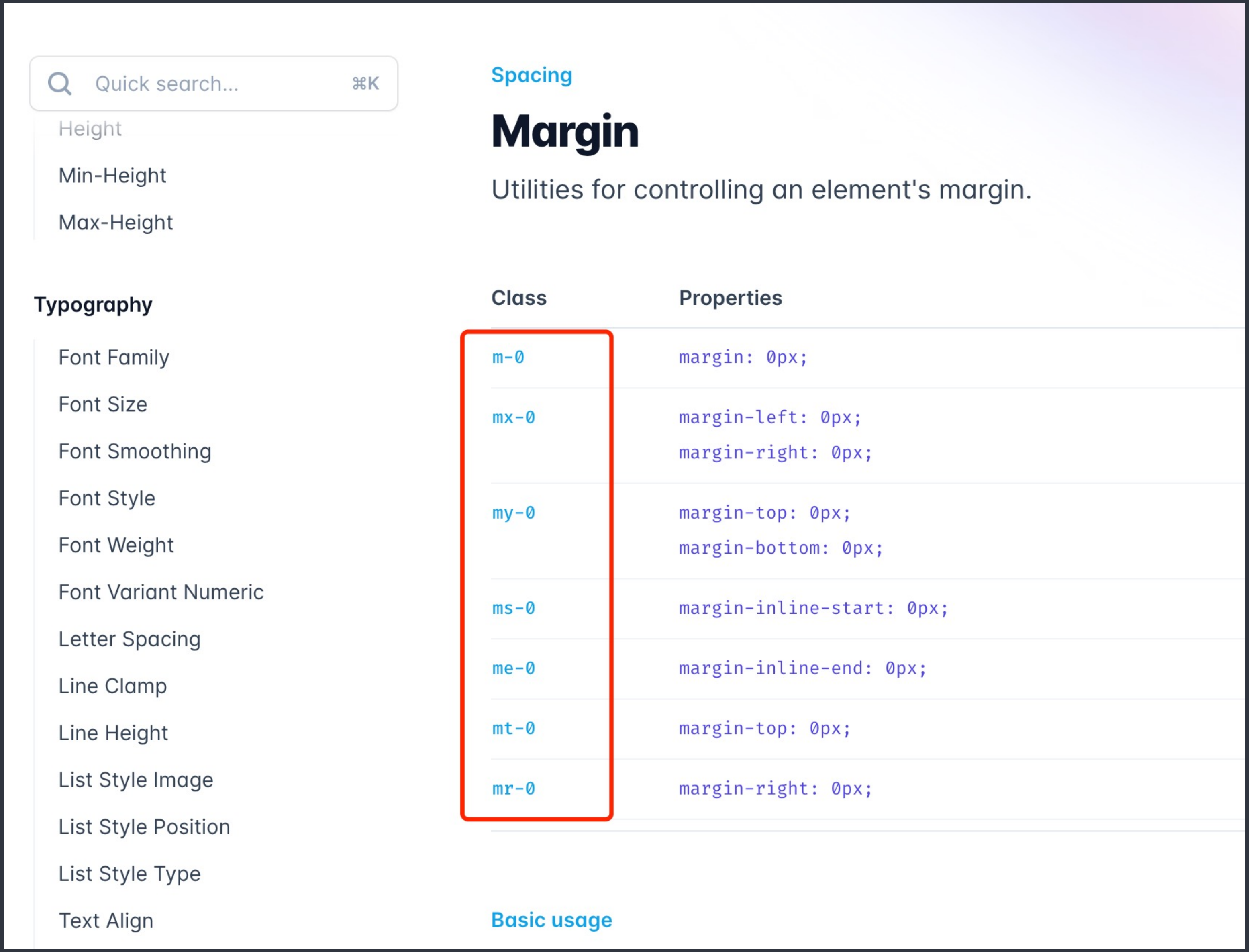
Rapidly build modern websites without ever leaving your HTML.

A utility-first CSS framework packed with classes like `flex`, `pt-4`, `text-center` and `rotate-90` that can be composed to build any design, directly in your markup.

梭哈 Tailwindcss , All in

- Tailwindcss 好处
  - **高度可定制，丰富预设的类**
  - **不用为取名称困干扰**
  - **不用担心改动副作用**
  - **支持响应式**
  - **支持 dark 模式**
  - **支持 @apply 组合**

困难1: 怎么在 web components 中使用 tailwindcss



Quick search... ⌘K

Height

Min-Height

Max-Height

Typography

Font Family

Font Size

Font Smoothing

Font Style

Font Weight

Font Variant Numeric

Letter Spacing

Line Clamp

Line Height

List Style Image

List Style Position

List Style Type

Text Align

Spacing

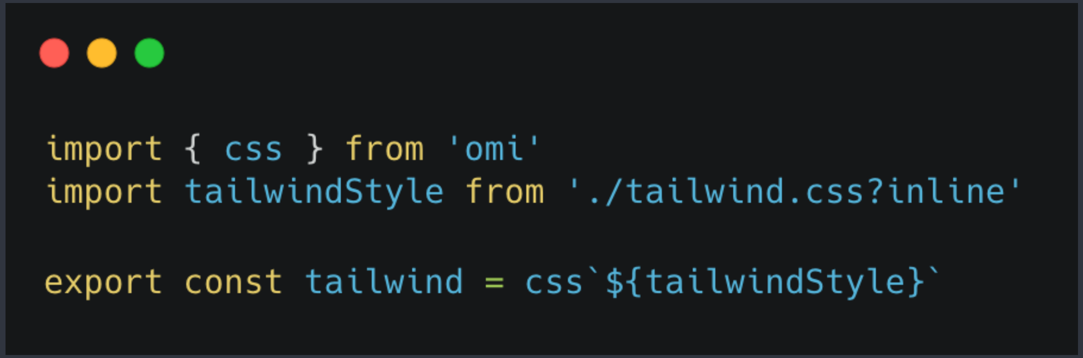
Margin

Utilities for controlling an element's margin.

Class	Properties
`m-0`	`margin: 0px;`
`mx-0`	`margin-left: 0px;` `margin-right: 0px;`
`my-0`	`margin-top: 0px;` `margin-bottom: 0px;`
`ms-0`	`margin-inline-start: 0px;`
`me-0`	`margin-inline-end: 0px;`
`mt-0`	`margin-top: 0px;`
`mr-0`	`margin-right: 0px;`

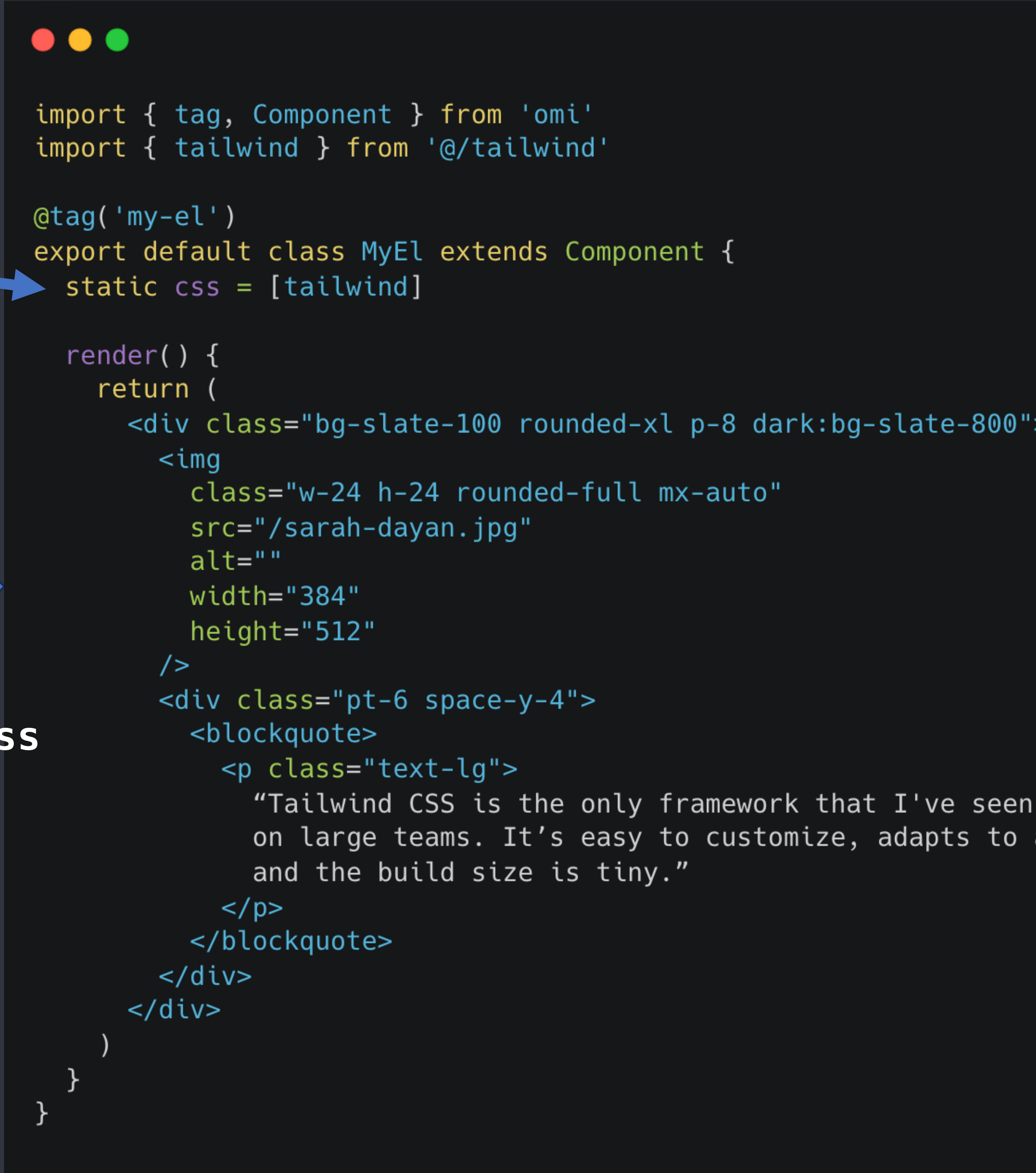
Basic usage

Constructed stylesheet



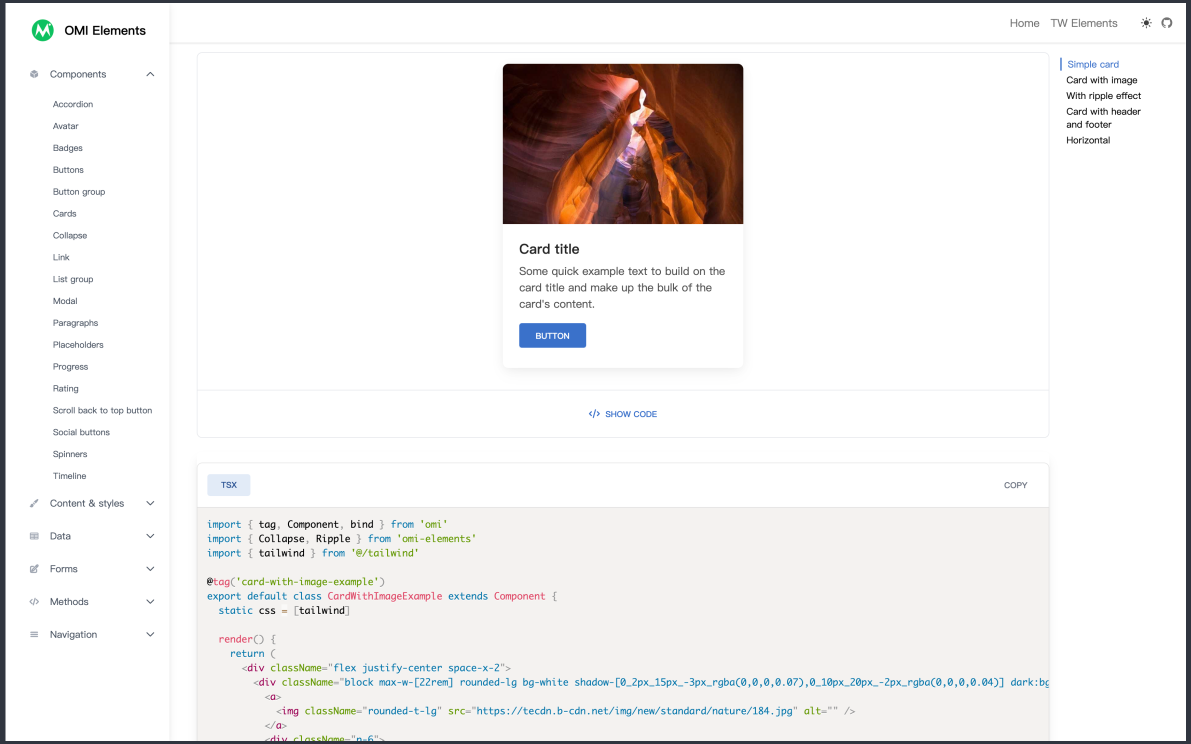
```
import { css } from 'omi'
import tailwindStyle from './tailwind.css?inline'

export const tailwind = css`${tailwindStyle}`
```

使用
Tailwindcss



```
import { tag, Component } from 'omi'
import { tailwind } from '@/tailwind'

@tag('my-el')
export default class MyEl extends Component {
  static css = [tailwind]

  render() {
    return (
      <div class="bg-slate-100 rounded-xl p-8 dark:bg-slate-800"
        <img
          class="w-24 h-24 rounded-full mx-auto"
          src="/sarah-dayan.jpg"
          alt=""
          width="384"
          height="512"
        />
        <div class="pt-6 space-y-4">
          <blockquote>
            <p class="text-lg">
              “Tailwind CSS is the only framework that I've seen
              on large teams. It’s easy to customize, adapts to
              and the build size is tiny.”
            </p>
          </blockquote>
        </div>
      </div>
    )
  }
}
```

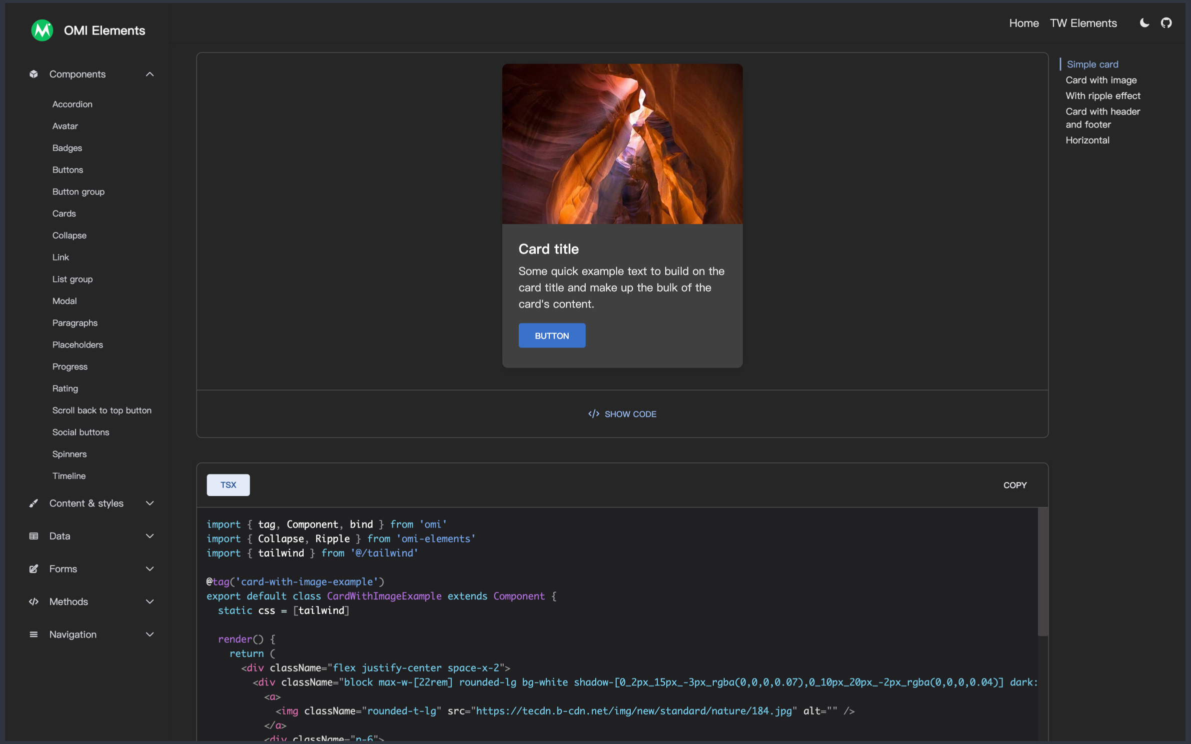
困难2：Shadow DOM 内怎么在使用到 HTML 的 dark class



Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

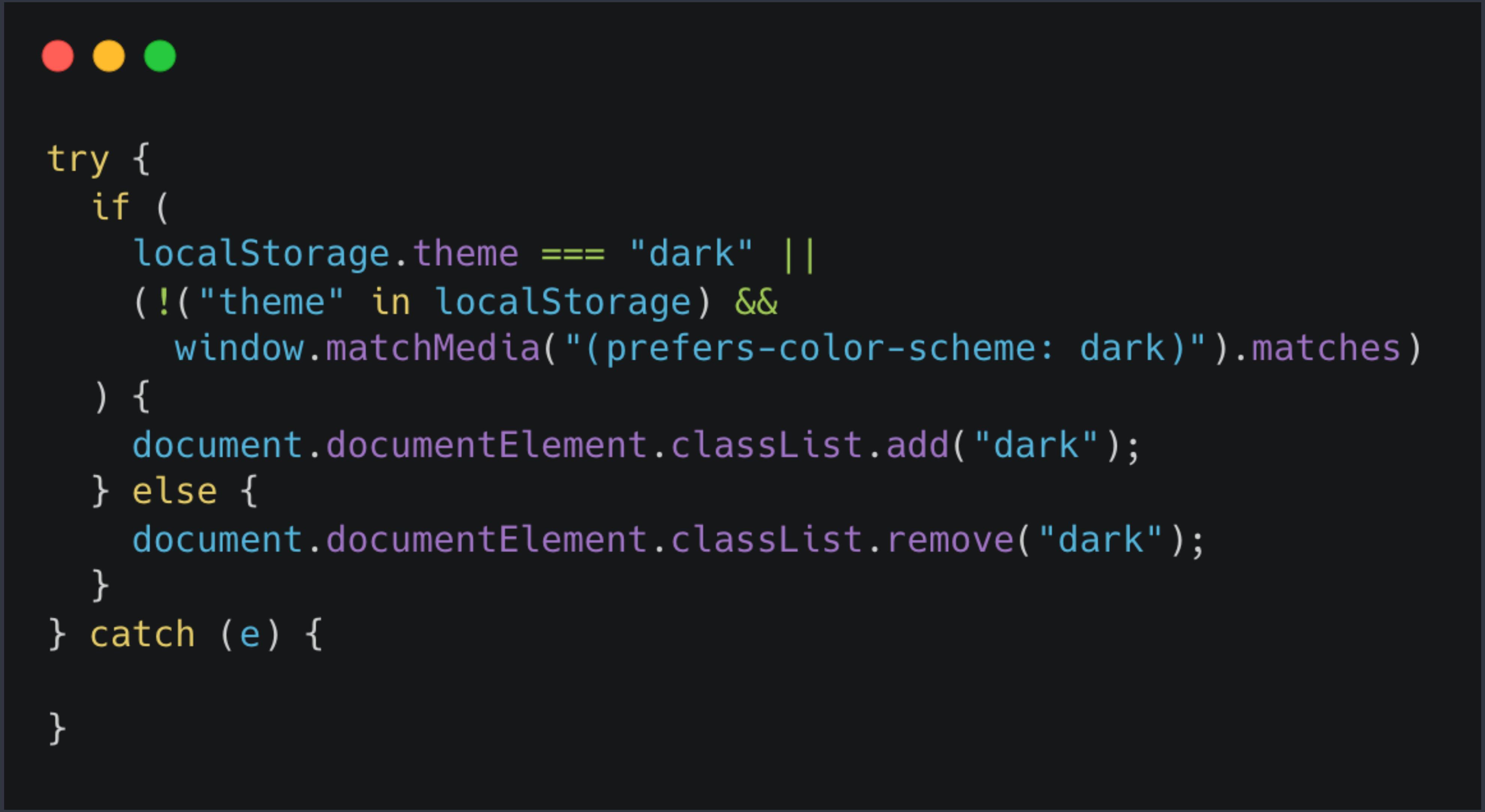
```
import { tag, Component, bind } from 'omi'
import { Collapse, Ripple } from 'omi-elements'
import { tailwind } from '@/tailwind'

@tag('card-with-image-example')
export default class CardWithImageExample extends Component {
  static css = [tailwind]

  render() {
    return (
      <div className="flex justify-center space-x-2">
        <div className="block max-w-[22rem] rounded-lg bg-white shadow-[0_2px_15px_-3px_rgba(0,0,0,0.07),0_10px_20px_-2px_rgba(0,0,0,0.04)] dark:bg
          <a>
            <img className="rounded-t-lg" src="https://tecdn.b-cdn.net/img/new/standard/nature/184.jpg" alt="" />
          </a>
          <div className="p-6">
```



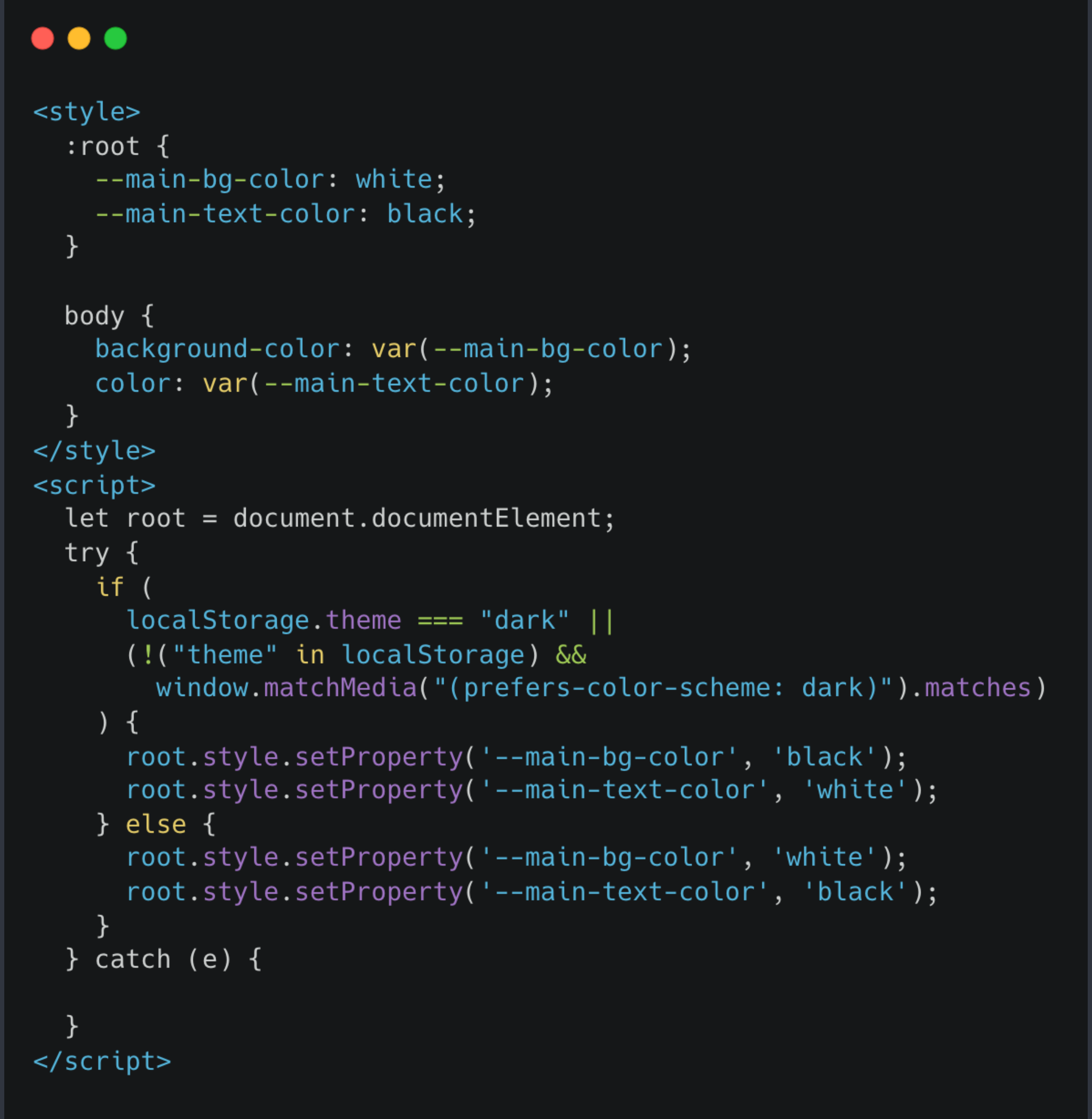
Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

```
import { tag, Component, bind } from 'omi'
import { Collapse, Ripple } from 'omi-elements'
import { tailwind } from '@/tailwind'

@tag('card-with-image-example')
export default class CardWithImageExample extends Component {
  static css = [tailwind]

  render() {
    return (
      <div className="flex justify-center space-x-2">
        <div className="block max-w-[22rem] rounded-lg bg-white shadow-[0_2px_15px_-3px_rgba(0,0,0,0.07),0_10px_20px_-2px_rgba(0,0,0,0.04)] dark:
          <a>
            <img className="rounded-t-lg" src="https://tecdn.b-cdn.net/img/new/standard/nature/184.jpg" alt="" />
          </a>
          <div className="p-6">
```



```javascript
try {
  if (
    localStorage.theme === "dark" ||
    (!("theme" in localStorage) &&
      window.matchMedia("(prefers-color-scheme: dark)").matches)
  ) {
    document.documentElement.classList.add("dark");
  } else {
    document.documentElement.classList.remove("dark");
  }
} catch (e) {

}
```

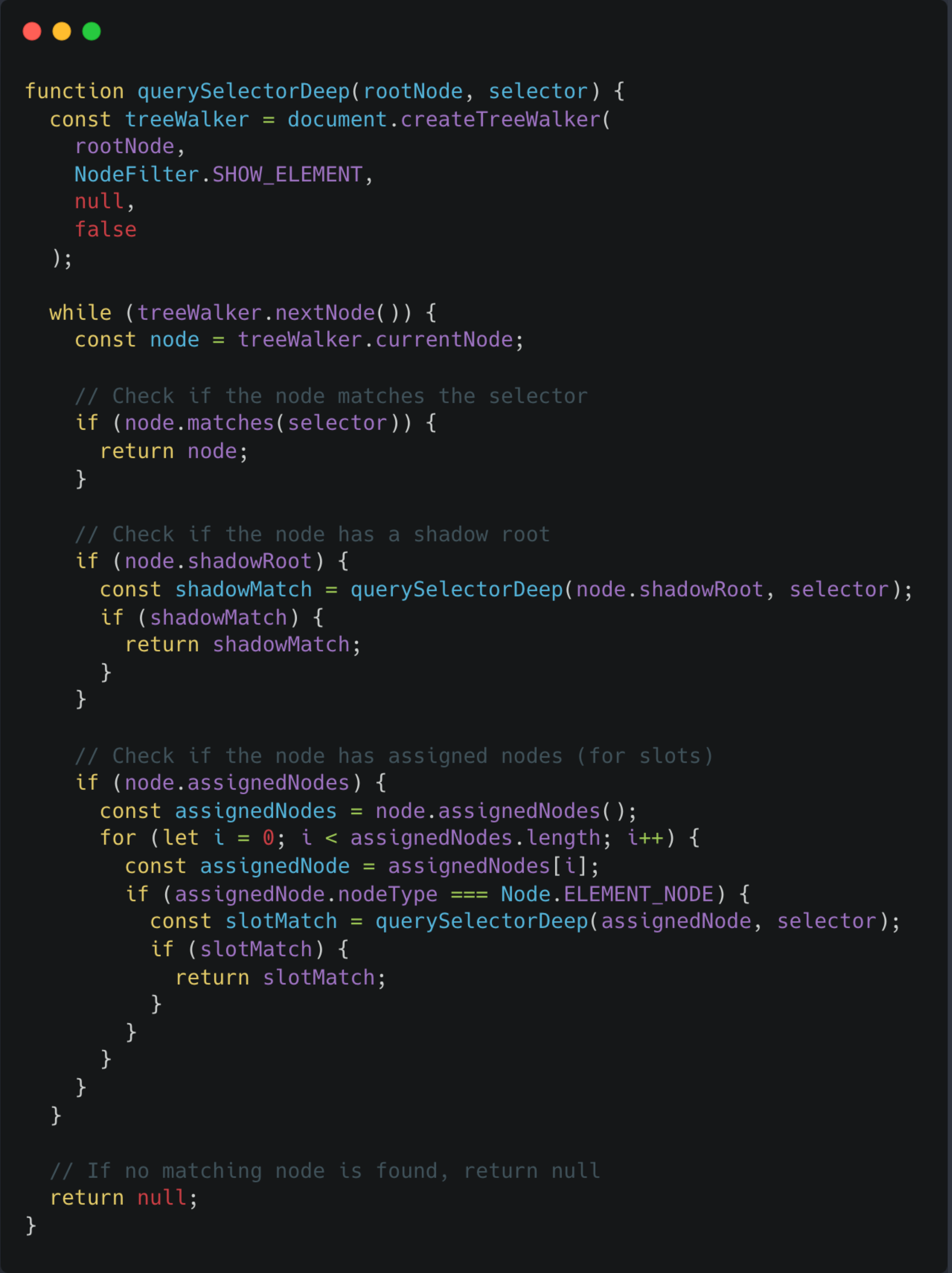


```html
<style>
  :root {
    --main-bg-color: white;
    --main-text-color: black;
  }

  body {
    background-color: var(--main-bg-color);
    color: var(--main-text-color);
  }
</style>
<script>
  let root = document.documentElement;
  try {
    if (
      localStorage.theme === "dark" ||
      (!("theme" in localStorage) &&
        window.matchMedia("(prefers-color-scheme: dark)").matches)
    ) {
      root.style.setProperty('--main-bg-color', 'black');
      root.style.setProperty('--main-text-color', 'white');
    } else {
      root.style.setProperty('--main-bg-color', 'white');
      root.style.setProperty('--main-text-color', 'black');
    }
  } catch (e) {

  }
</script>
```

```
document.querySelector('#my-el-id')
```

$$\downarrow$$

```
document.createTreeWalker
```

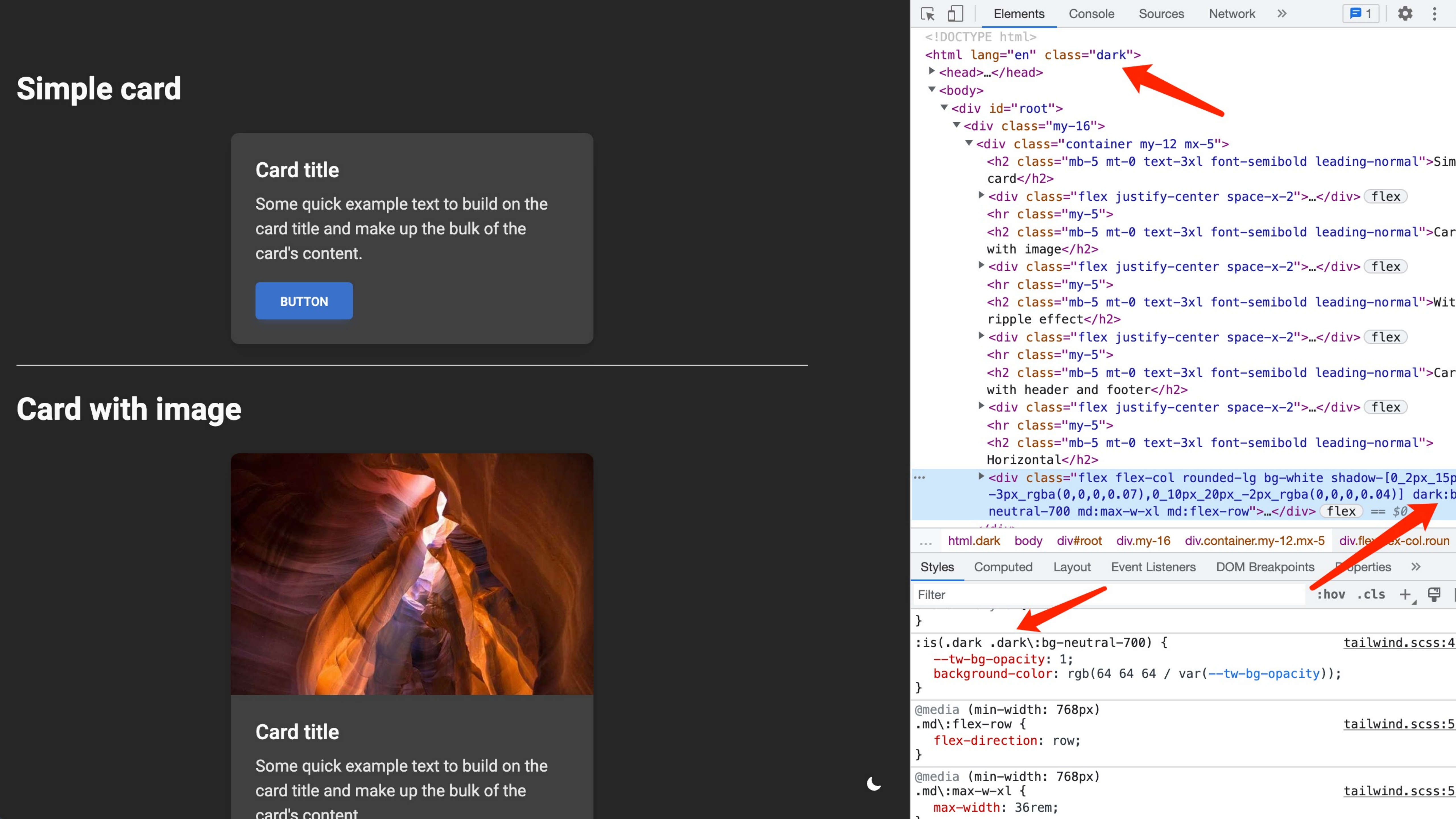


```
function querySelectorDeep(rootNode, selector) {
  const treeWalker = document.createTreeWalker(
    rootNode,
    NodeFilter.SHOW_ELEMENT,
    null,
    false
  );

  while (treeWalker.nextNode()) {
    const node = treeWalker.currentNode;

    // Check if the node matches the selector
    if (node.matches(selector)) {
      return node;
    }

    // Check if the node has a shadow root
    if (node.shadowRoot) {
      const shadowMatch = querySelectorDeep(node.shadowRoot, selector);
      if (shadowMatch) {
        return shadowMatch;
      }
    }

    // Check if the node has assigned nodes (for slots)
    if (node.assignedNodes) {
      const assignedNodes = node.assignedNodes();
      for (let i = 0; i < assignedNodes.length; i++) {
        const assignedNode = assignedNodes[i];
        if (assignedNode.nodeType === Node.ELEMENT_NODE) {
          const slotMatch = querySelectorDeep(assignedNode, selector);
          if (slotMatch) {
            return slotMatch;
          }
        }
      }
    }
  }

  // If no matching node is found, return null
  return null;
}
```

document.querySelector(‘#my-el-id’)

querySelectorDeep(document, ‘#my-el-id’)

深入到 shadow roots 和 slots 中查找



Simple card

Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

Card with image

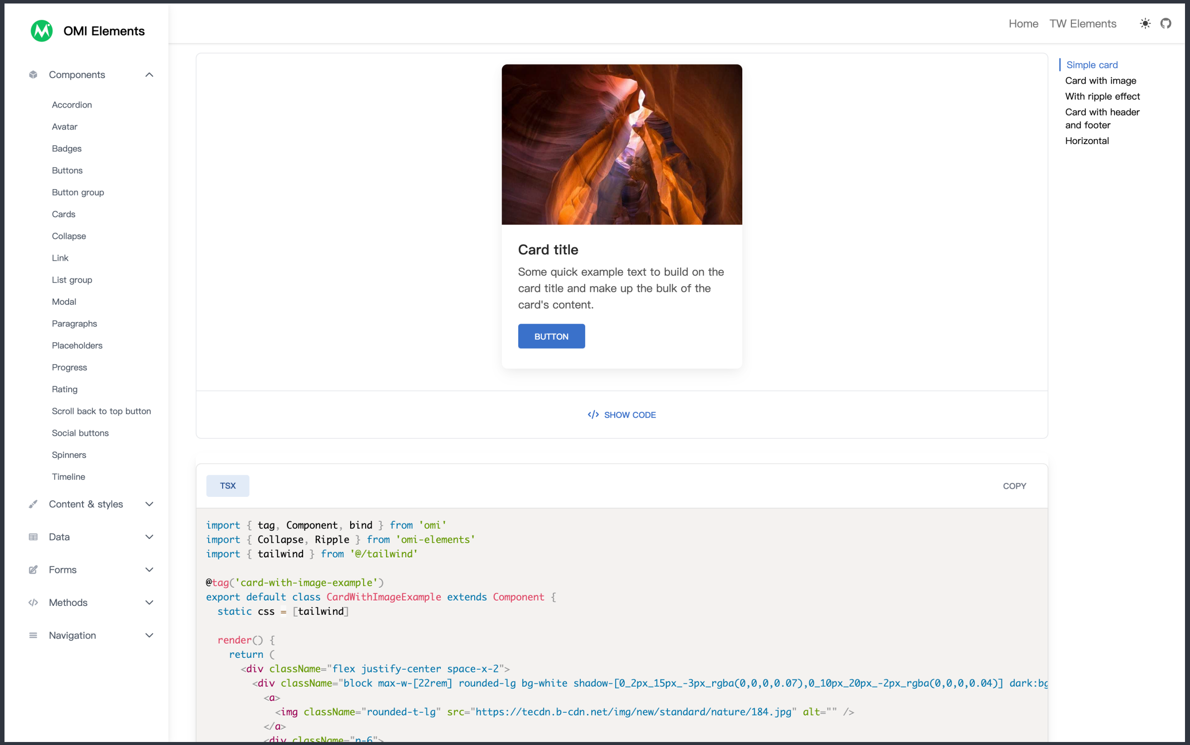
Card title

Some quick example text to build on the card title and make up the bulk of the card's content

```
<!DOCTYPE html>
<html lang="en" class="dark">
▶<head>…</head>
▼<body>
  ▼<div id="root">
    ▼<div class="my-16">
      ▼<div class="container my-12 mx-5">
         <h2 class="mb-5 mt-0 text-3xl font-semibold leading-normal">Sim
         card</h2>
        ▶<div class="flex justify-center space-x-2">…</div> flex
         <hr class="my-5">
         <h2 class="mb-5 mt-0 text-3xl font-semibold leading-normal">Car
         with image</h2>
        ▶<div class="flex justify-center space-x-2">…</div> flex
         <hr class="my-5">
         <h2 class="mb-5 mt-0 text-3xl font-semibold leading-normal">Wit
         ripple effect</h2>
        ▶<div class="flex justify-center space-x-2">…</div> flex
         <hr class="my-5">
         <h2 class="mb-5 mt-0 text-3xl font-semibold leading-normal">Car
         with header and footer</h2>
        ▶<div class="flex justify-center space-x-2">…</div> flex
         <hr class="my-5">
         <h2 class="mb-5 mt-0 text-3xl font-semibold leading-normal">
         Horizontal</h2>
...     ▶<div class="flex flex-col rounded-lg bg-white shadow-[0_2px_15p
         -3px_rgba(0,0,0,0.07),0_10px_20px_-2px_rgba(0,0,0,0.04)] dark:b
         neutral-700 md:max-w-xl md:flex-row">…</div> flex == $0
```

```
... html.dark body div#root div.my-16 div.container.my-12.mx-5 div.flex.flex-col.roun
```

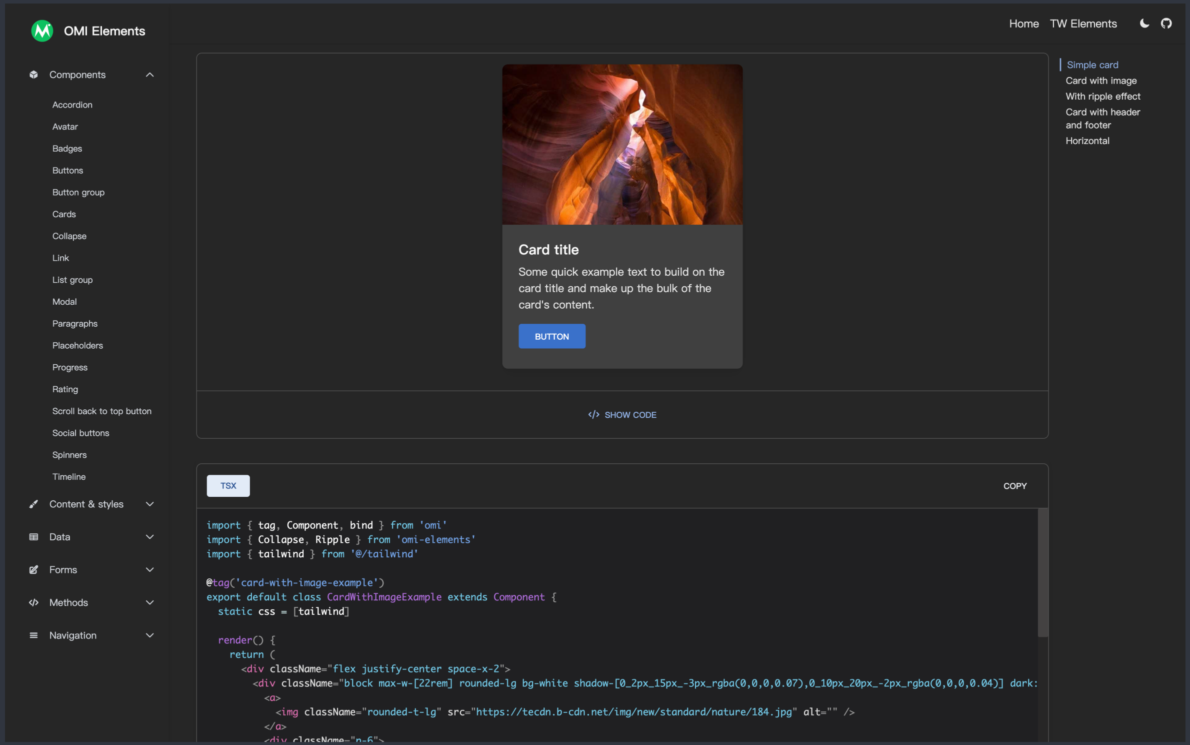
```
}
:is(.dark .dark\:bg-neutral-700) {                    tailwind.scss:4
   --tw-bg-opacity: 1;
   background-color: rgb(64 64 64 / var(--tw-bg-opacity));
}
@media (min-width: 768px)
.md\:flex-row {                                       tailwind.scss:5
   flex-direction: row;
}
@media (min-width: 768px)
.md\:max-w-xl {                                       tailwind.scss:5
   max-width: 36rem;
```



Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

```
import { tag, Component, bind } from 'omi'
import { Collapse, Ripple } from 'omi-elements'
import { tailwind } from '@/tailwind'

@tag('card-with-image-example')
export default class CardWithImageExample extends Component {
  static css = [tailwind]

  render() {
    return (
      <div className="flex justify-center space-x-2">
        <div className="block max-w-[22rem] rounded-lg bg-white shadow-[0_2px_15px_-3px_rgba(0,0,0,0.07),0_10px_20px_-2px_rgba(0,0,0,0.04)] dark:bg
          <a>
            <img className="rounded-t-lg" src="https://tecdn.b-cdn.net/img/new/standard/nature/184.jpg" alt="" />
          </a>
          <div className="p-6">
```

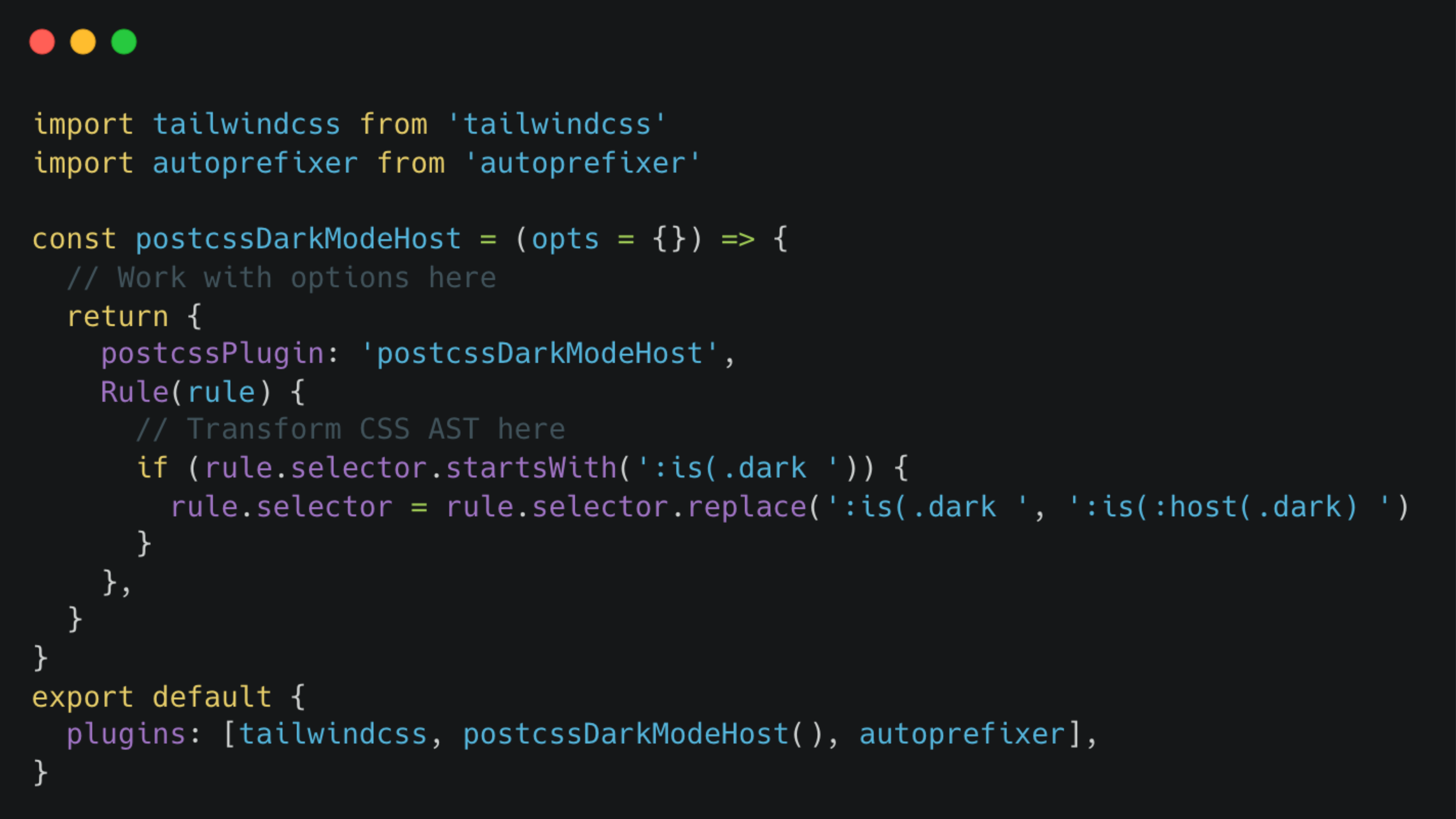


Card title

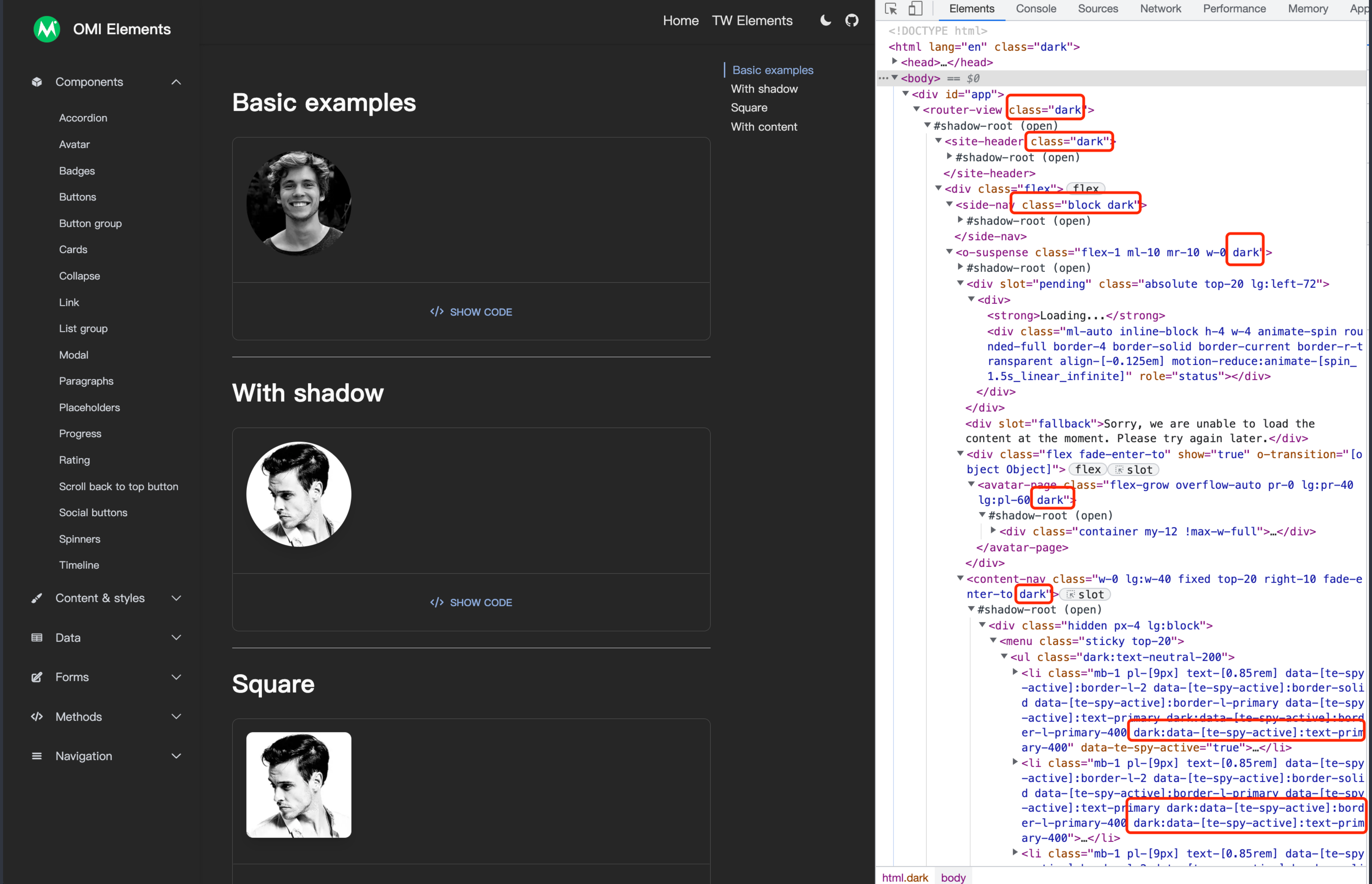
Some quick example text to build on the card title and make up the bulk of the card's content.

```
import { tag, Component, bind } from 'omi'
import { Collapse, Ripple } from 'omi-elements'
import { tailwind } from '@/tailwind'

@tag('card-with-image-example')
export default class CardWithImageExample extends Component {
  static css = [tailwind]

  render() {
    return (
      <div className="flex justify-center space-x-2">
        <div className="block max-w-[22rem] rounded-lg bg-white shadow-[0_2px_15px_-3px_rgba(0,0,0,0.07),0_10px_20px_-2px_rgba(0,0,0,0.04)] dark:
          <a>
            <img className="rounded-t-lg" src="https://tecdn.b-cdn.net/img/new/standard/nature/184.jpg" alt="" />
          </a>
          <div className="p-6">
```

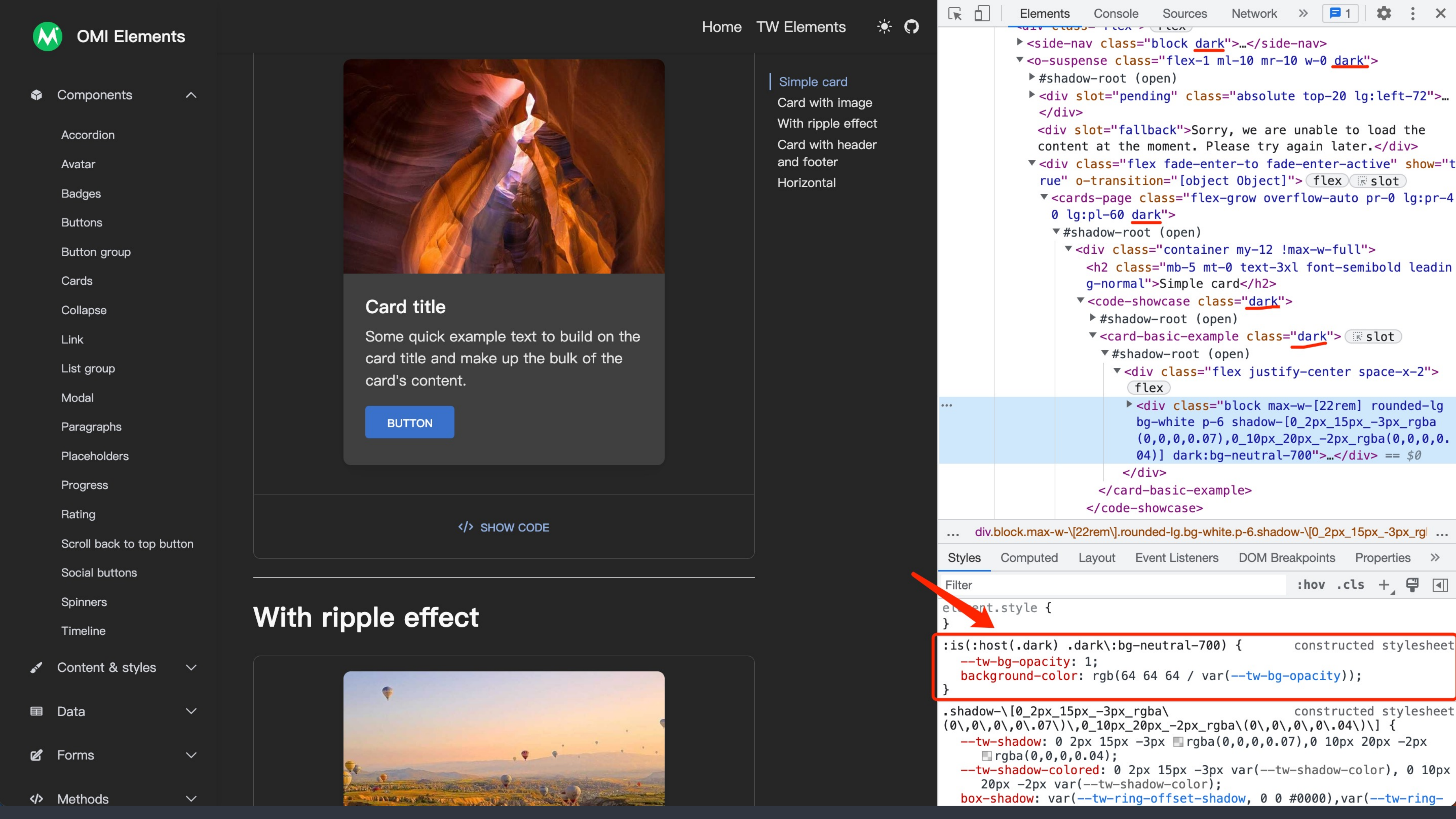
document.createTreeWalker 进行穿透



```
import tailwindcss from 'tailwindcss'
import autoprefixer from 'autoprefixer'

const postcssDarkModeHost = (opts = {}) => {
  // Work with options here
  return {
    postcssPlugin: 'postcssDarkModeHost',
    Rule(rule) {
      // Transform CSS AST here
      if (rule.selector.startsWith(':is(.dark ')) {
        rule.selector = rule.selector.replace(':is(.dark ', ':is(:host(.dark) ')
      }
    },
  }
}
export default {
  plugins: [tailwindcss, postcssDarkModeHost(), autoprefixer],
}
```



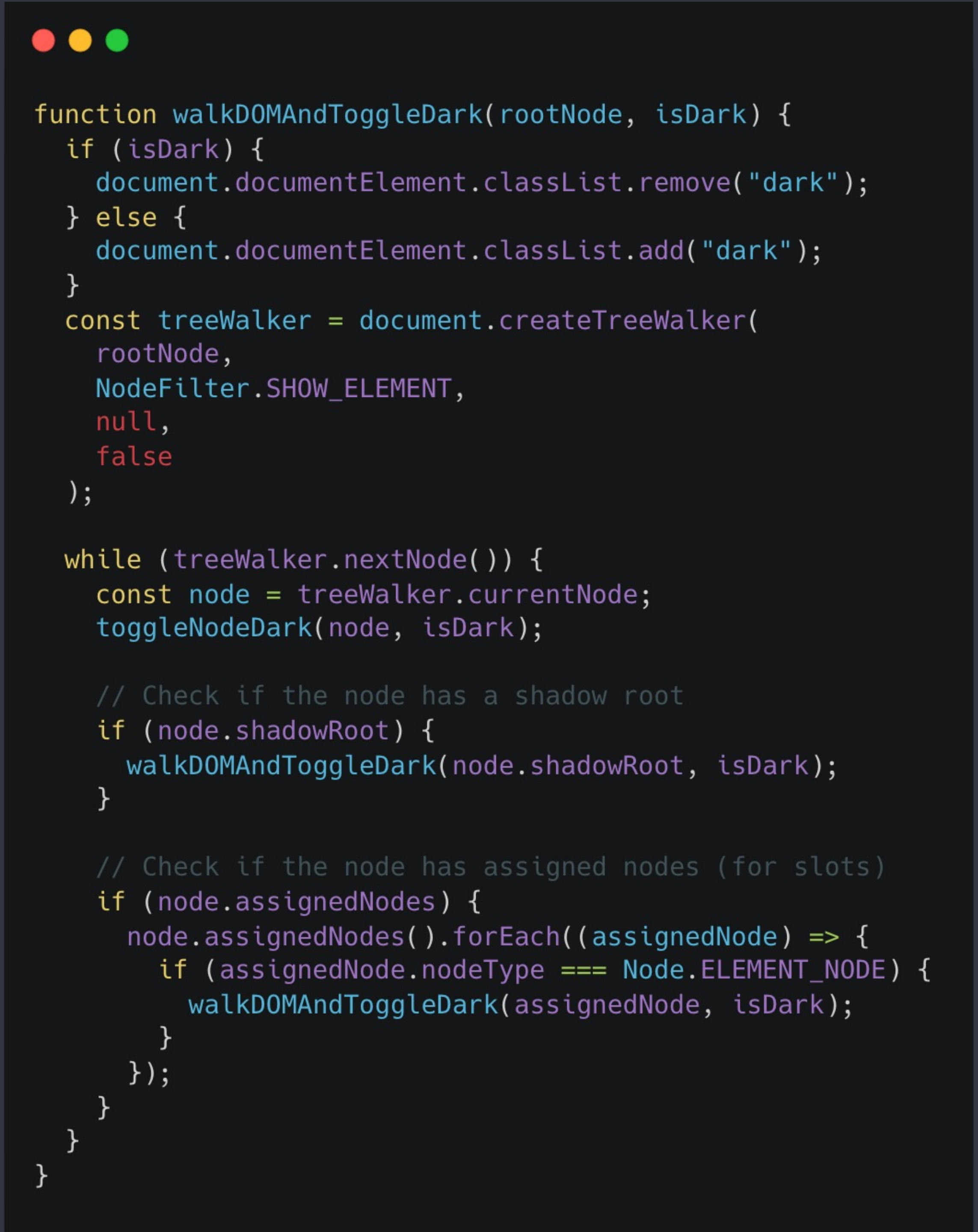
```
<!DOCTYPE html>
<html lang="en" class="dark">
 ▶<head>…</head>
…▼<body> == $0
  ▼<div id="app">
   ▼<router-view class="dark">
    ▼#shadow-root (open)
     ▼<site-header class="dark">
      ▶#shadow-root (open)
     </site-header>
     ▼<div class="flex"> flex
      ▼<side-nav class="block dark">
       ▶#shadow-root (open)
      </side-nav>
      ▼<o-suspense class="flex-1 ml-10 mr-10 w-0 dark">
       ▶#shadow-root (open)
       ▼<div slot="pending" class="absolute top-20 lg:left-72">
        ▼<div>
          <strong>Loading...</strong>
          <div class="ml-auto inline-block h-4 w-4 animate-spin rou
          nded-full border-4 border-solid border-current border-r-t
          ransparent align-[-0.125em] motion-reduce:animate-[spin_
          1.5s_linear_infinite]" role="status"></div>
         </div>
        </div>
        <div slot="fallback">Sorry, we are unable to load the
        content at the moment. Please try again later.</div>
       ▼<div class="flex fade-enter-to" show="true" o-transition="[o
        bject Object]"> flex  slot
        ▼<avatar-page class="flex-grow overflow-auto pr-0 lg:pr-40
          lg:pl-60 dark">
          ▼#shadow-root (open)
           ▶<div class="container my-12 !max-w-full">…</div>
         </avatar-page>
        </div>
       ▼<content-nav class="w-0 lg:w-40 fixed top-20 right-10 fade-e
        nter-to dark"> slot
        ▼#shadow-root (open)
         ▼<div class="hidden px-4 lg:block">
          ▼<menu class="sticky top-20">
           ▼<ul class="dark:text-neutral-200">
            ▶<li class="mb-1 pl-[9px] text-[0.85rem] data-[te-spy
              -active]:border-l-2 data-[te-spy-active]:border-soli
              d data-[te-spy-active]:border-l-primary data-[te-spy
              -active]:text-primary dark:data-[te-spy-active]:bord
              er-l-primary-400 dark:data-[te-spy-active]:text-prim
              ary-400" data-te-spy-active="true">…</li>
            ▶<li class="mb-1 pl-[9px] text-[0.85rem] data-[te-spy
              -active]:border-l-2 data-[te-spy-active]:border-soli
              d data-[te-spy-active]:border-l-primary data-[te-spy
              -active]:text-primary dark:data-[te-spy-active]:bord
              er-l-primary-400 dark:data-[te-spy-active]:text-prim
              ary-400">…</li>
            ▶<li class="mb-1 pl-[9px] text-[0.85rem] data-[te-spy
```



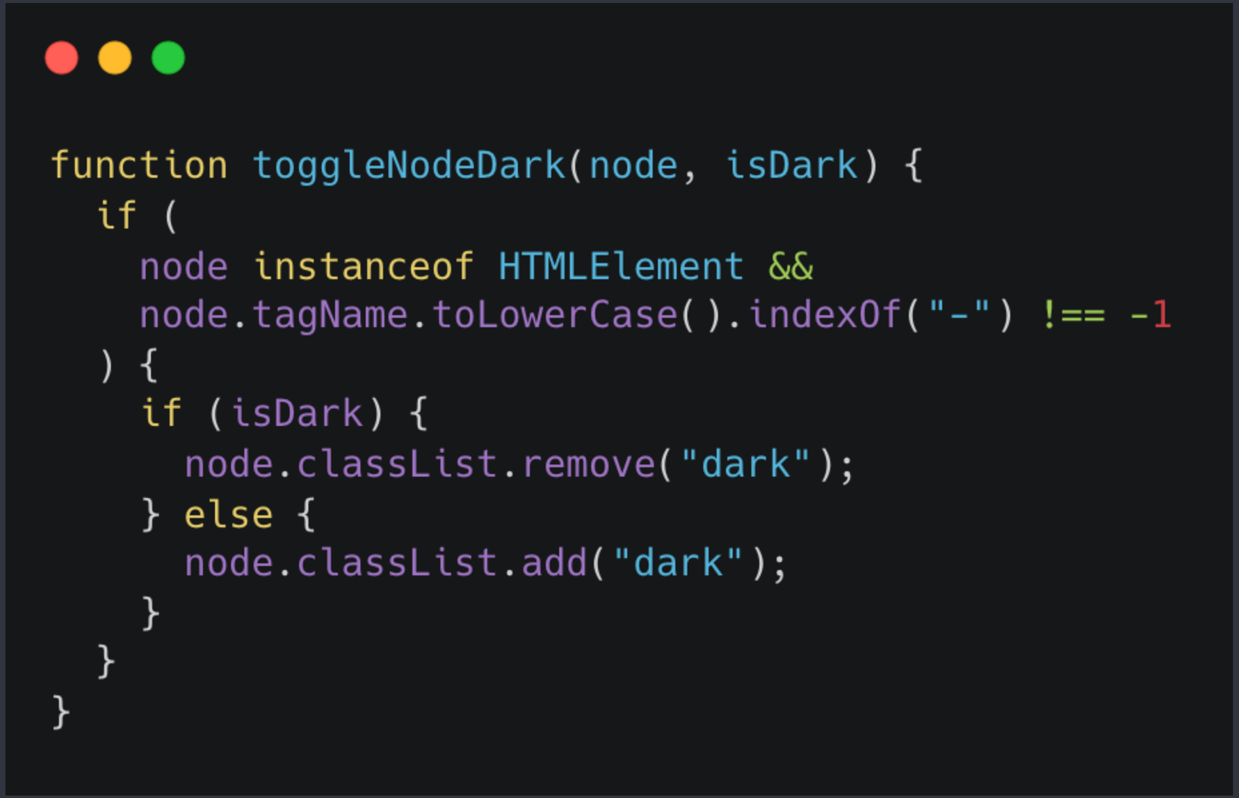
```
▶<side-nav class="block dark">…</side-nav>
▼<o-suspense class="flex-1 ml-10 mr-10 w-0 dark">
  ▶#shadow-root (open)
  ▶<div slot="pending" class="absolute top-20 lg:left-72">…
  </div>
  <div slot="fallback">Sorry, we are unable to load the
  content at the moment. Please try again later.</div>
  ▼<div class="flex fade-enter-to fade-enter-active" show="t
  rue" o-transition="[object Object]"> flex  slot
    ▼<cards-page class="flex-grow overflow-auto pr-0 lg:pr-4
    0 lg:pl-60 dark">
      ▼#shadow-root (open)
        ▼<div class="container my-12 !max-w-full">
          <h2 class="mb-5 mt-0 text-3xl font-semibold leadin
          g-normal">Simple card</h2>
          ▼<code-showcase class="dark">
            ▶#shadow-root (open)
            ▼<card-basic-example class="dark"> slot
              ▼#shadow-root (open)
                ▼<div class="flex justify-center space-x-2">
                  flex
...               ▶<div class="block max-w-[22rem] rounded-lg
                  bg-white p-6 shadow-[0_2px_15px_-3px_rgba
                  (0,0,0,0.07),0_10px_20px_-2px_rgba(0,0,0,0.
                  04)] dark:bg-neutral-700">…</div> == $0
                </div>
              </card-basic-example>
            </code-showcase>
```

```
... div.block.max-w-\[22rem\].rounded-lg.bg-white.p-6.shadow-\[0_2px_15px_-3px_rgl ...
```

```
element.style {
}
:is(:host(.dark) .dark\:bg-neutral-700) {        constructed stylesheet
  --tw-bg-opacity: 1;
  background-color: rgb(64 64 64 / var(--tw-bg-opacity));
}
.shadow-\[0_2px_15px_-3px_rgba\                  constructed stylesheet
(0\,0\,0\,0\.07\)\,0_10px_20px_-2px_rgba\(0\,0\,0\,0\.04\)\] {
  --tw-shadow: 0 2px 15px -3px rgba(0,0,0,0.07),0 10px 20px -2px
    rgba(0,0,0,0.04);
  --tw-shadow-colored: 0 2px 15px -3px var(--tw-shadow-color), 0 10px
    20px -2px var(--tw-shadow-color);
  box-shadow: var(--tw-ring-offset-shadow, 0 0 #0000),var(--tw-ring-
```

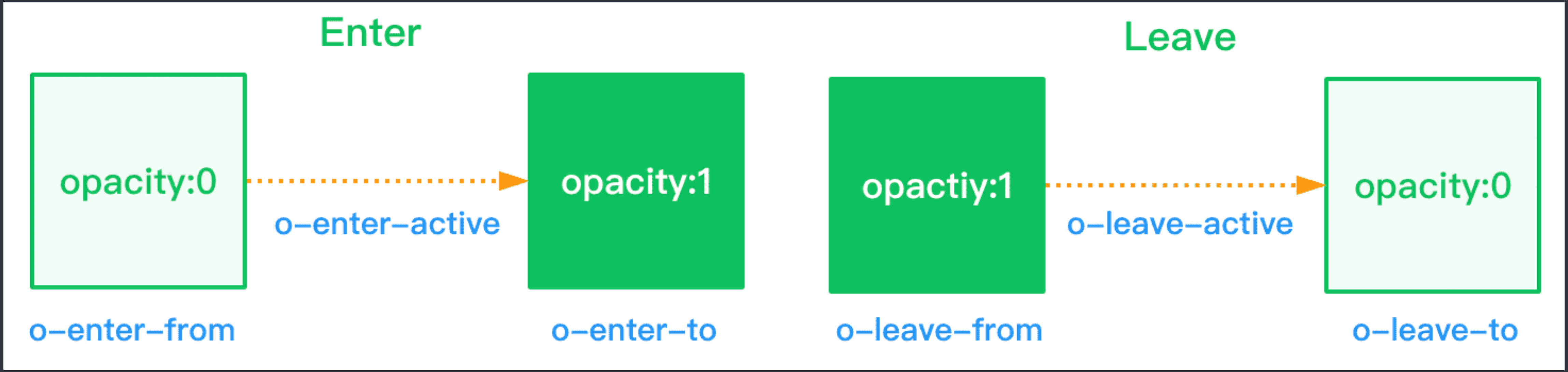


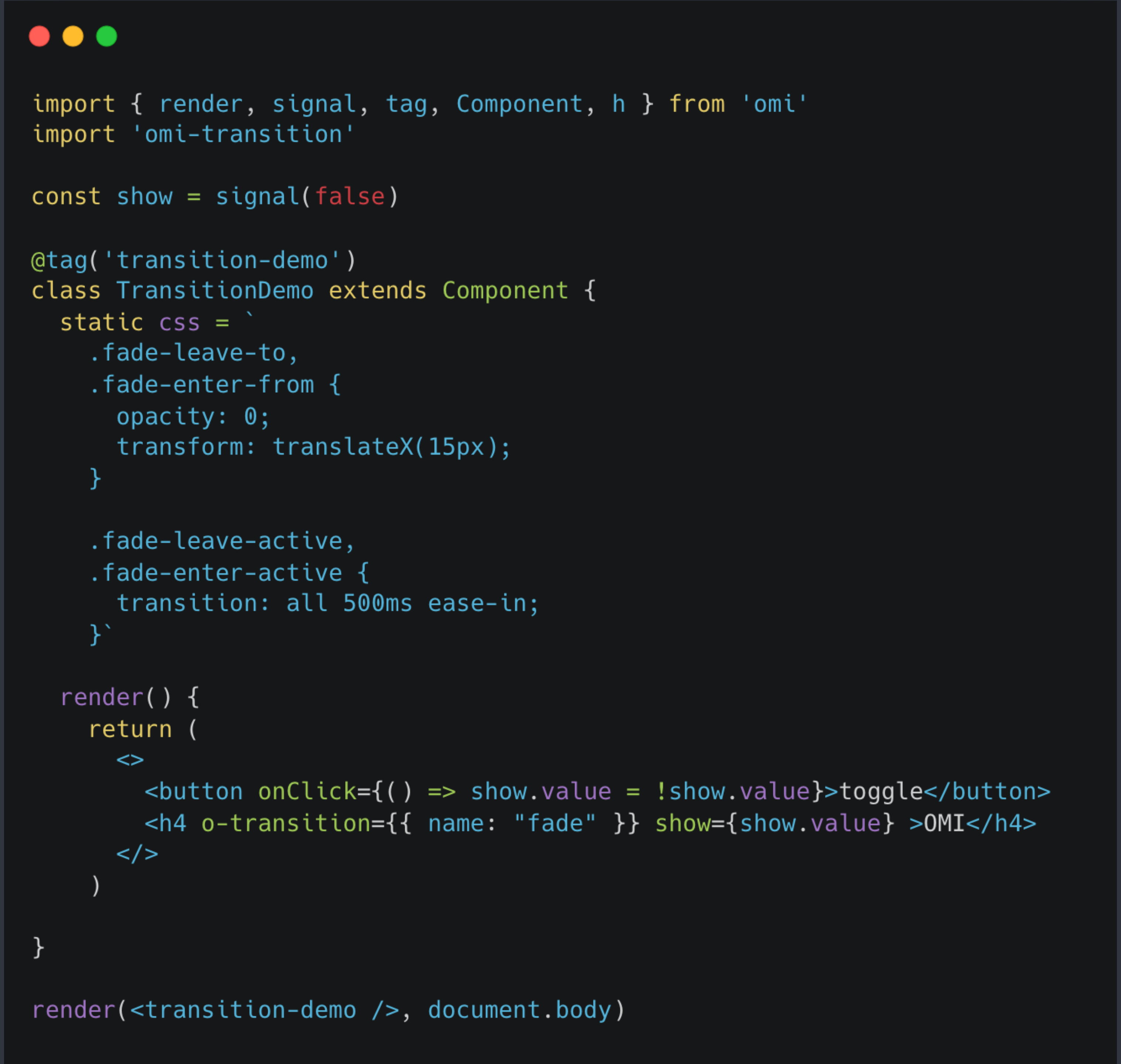
```
function walkDOMAndToggleDark(rootNode, isDark) {
  if (isDark) {
    document.documentElement.classList.remove("dark");
  } else {
    document.documentElement.classList.add("dark");
  }
  const treeWalker = document.createTreeWalker(
    rootNode,
    NodeFilter.SHOW_ELEMENT,
    null,
    false
  );

  while (treeWalker.nextNode()) {
    const node = treeWalker.currentNode;
    toggleNodeDark(node, isDark);

    // Check if the node has a shadow root
    if (node.shadowRoot) {
      walkDOMAndToggleDark(node.shadowRoot, isDark);
    }

    // Check if the node has assigned nodes (for slots)
    if (node.assignedNodes) {
      node.assignedNodes().forEach((assignedNode) => {
        if (assignedNode.nodeType === Node.ELEMENT_NODE) {
          walkDOMAndToggleDark(assignedNode, isDark);
        }
      });
    }
  }
}
```



```
function toggleNodeDark(node, isDark) {
  if (
    node instanceof HTMLElement &&
    node.tagName.toLowerCase().indexOf("-") !== -1
  ) {
    if (isDark) {
      node.classList.remove("dark");
    } else {
      node.classList.add("dark");
    }
  }
}
```

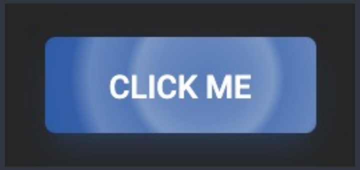
指令 - omi–transition

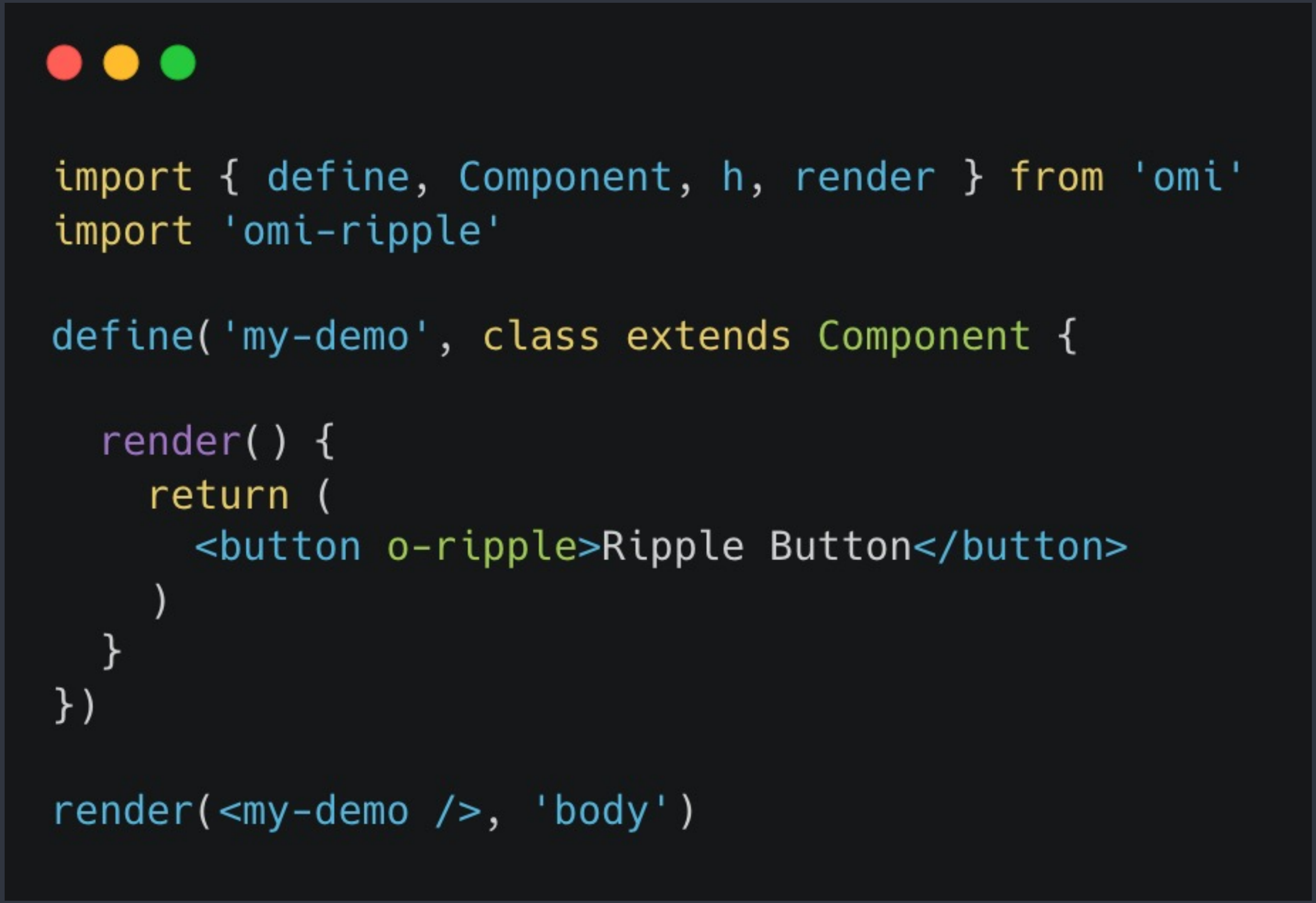




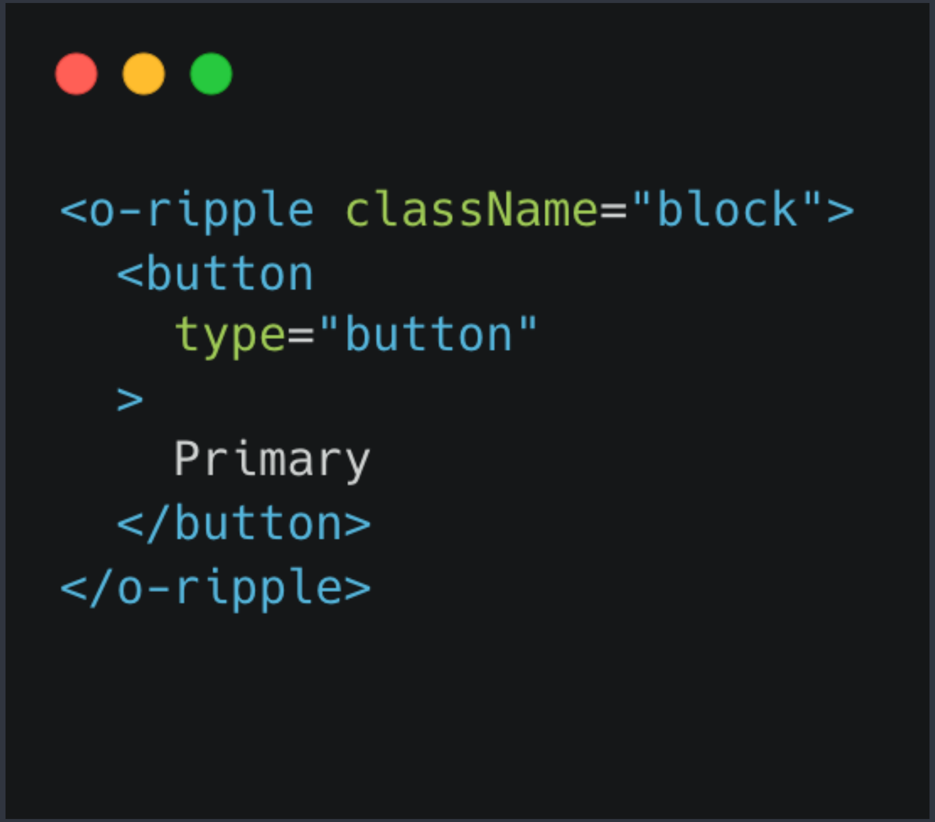
```
import { render, signal, tag, Component, h } from 'omi'
import 'omi-transition'

const show = signal(false)

@tag('transition-demo')
class TransitionDemo extends Component {
  static css = `
    .fade-leave-to,
    .fade-enter-from {
      opacity: 0;
      transform: translateX(15px);
    }

    .fade-leave-active,
    .fade-enter-active {
      transition: all 500ms ease-in;
    }`

  render() {
    return (
      <>
        <button onClick={() => show.value = !show.value}>toggle</button>
        <h4 o-transition={{ name: "fade" }} show={show.value} >OMI</h4>
      </>
    )

}

render(<transition-demo />, document.body)
```

指令 - omi-ripple

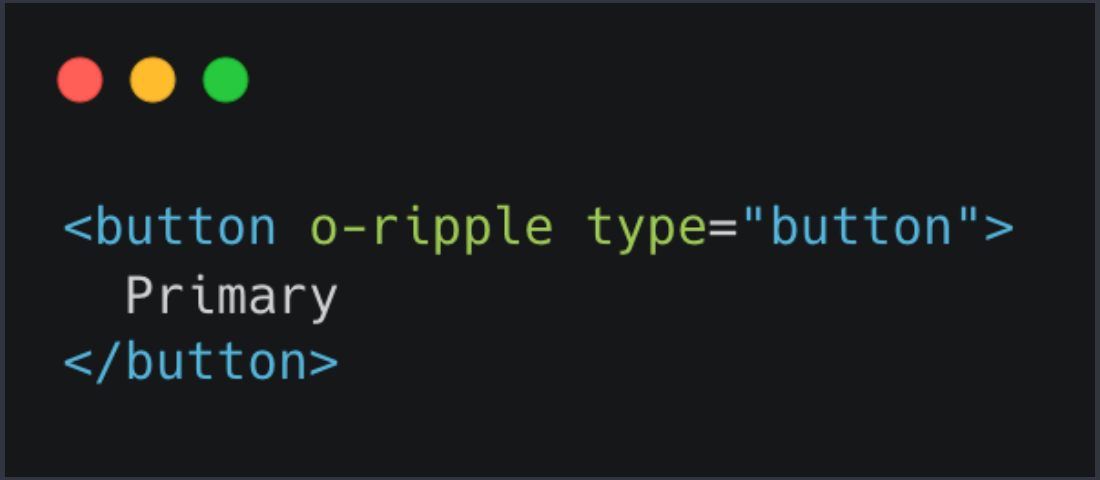




```
import { define, Component, h, render } from 'omi'
import 'omi-ripple'

define('my-demo', class extends Component {

  render() {
    return (
      <button o-ripple>Ripple Button</button>
    )
  }
})

render(<my-demo />, 'body')
```



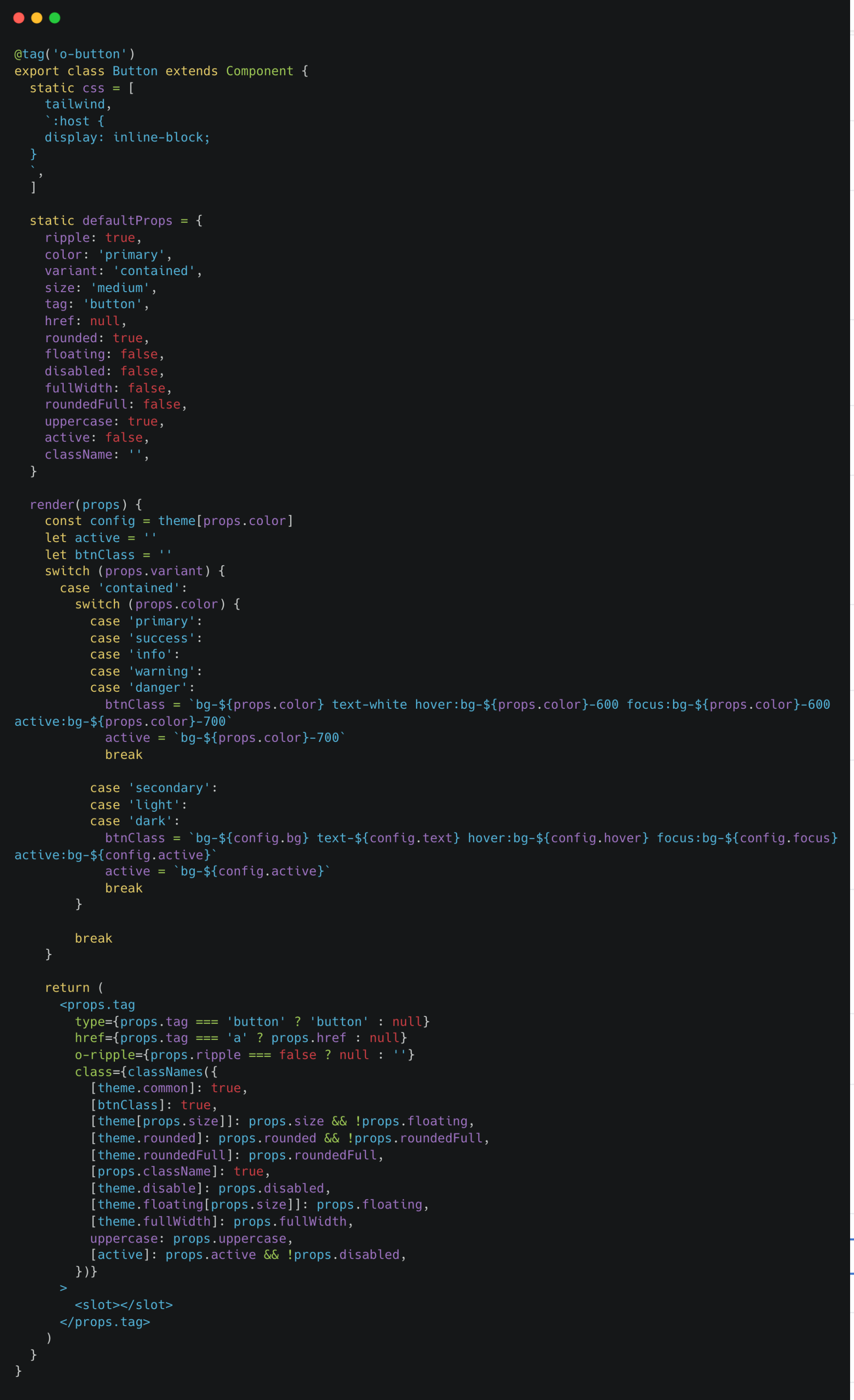
```
<o-ripple className="block">
  <button
    type="button"
  >
    Primary
  </button>
</o-ripple>
```



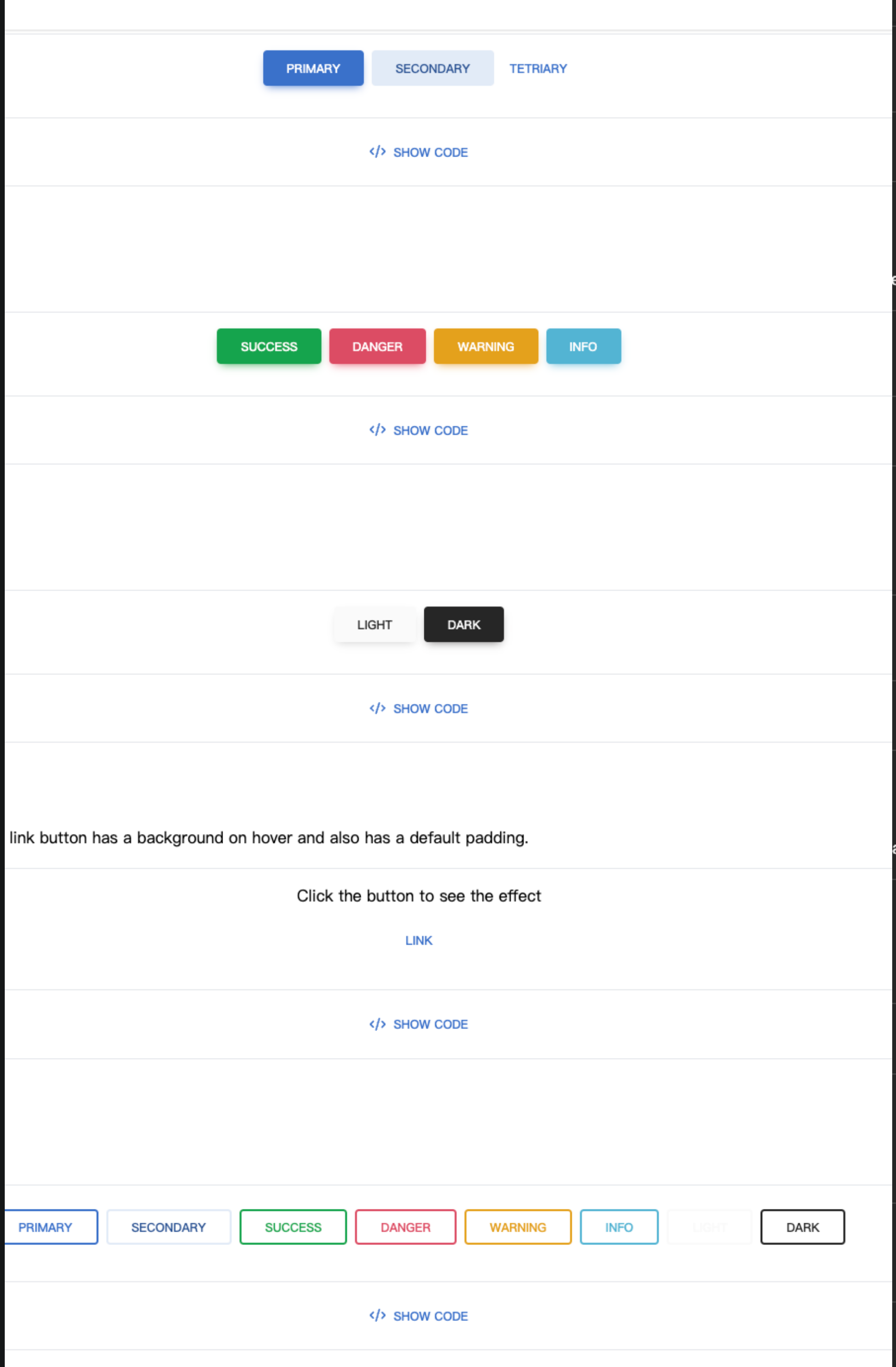
```
<button o-ripple type="button">
  Primary
</button>
```

:host 干扰布局

指令对布局无干扰

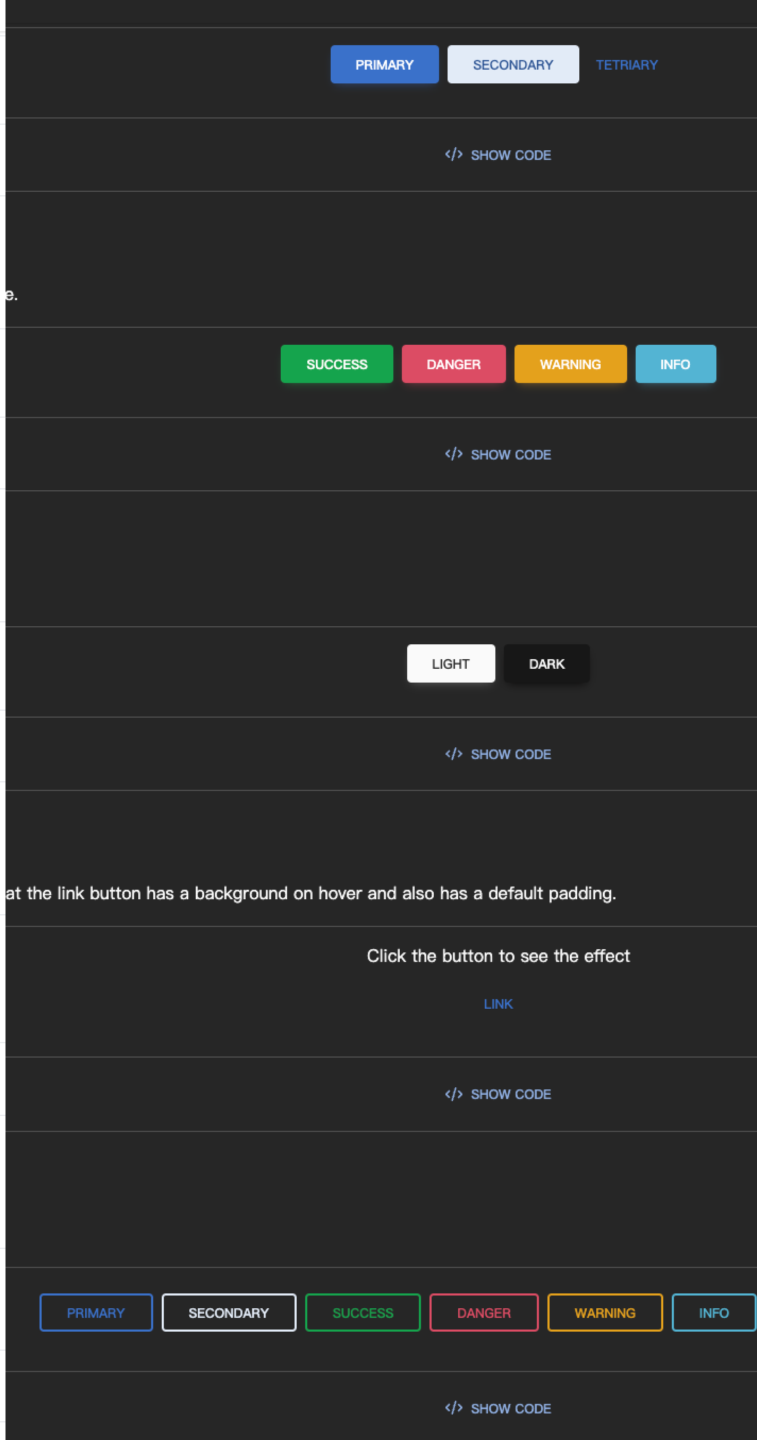


```
@tag('o-button')
export class Button extends Component {
  static css = [
    tailwind,
    `:host {
    display: inline-block;
  }
  `,
  ]

  static defaultProps = {
    ripple: true,
    color: 'primary',
    variant: 'contained',
    size: 'medium',
    tag: 'button',
    href: null,
    rounded: true,
    floating: false,
    disabled: false,
    fullWidth: false,
    roundedFull: false,
    uppercase: true,
    active: false,
    className: '',
  }

  render(props) {
    const config = theme[props.color]
    let active = ''
    let btnClass = ''
    switch (props.variant) {
      case 'contained':
        switch (props.color) {
          case 'primary':
          case 'success':
          case 'info':
          case 'warning':
          case 'danger':
            btnClass = `bg-${props.color} text-white hover:bg-${props.color}-600 focus:bg-${props.color}-600
active:bg-${props.color}-700`
            active = `bg-${props.color}-700`
            break

          case 'secondary':
          case 'light':
          case 'dark':
            btnClass = `bg-${config.bg} text-${config.text} hover:bg-${config.hover} focus:bg-${config.focus}
active:bg-${config.active}`
            active = `bg-${config.active}`
            break
        }

        break
    }

    return (
      <props.tag
        type={props.tag === 'button' ? 'button' : null}
        href={props.tag === 'a' ? props.href : null}
        o-ripple={props.ripple === false ? null : ''}
        class={classNames({
          [theme.common]: true,
          [btnClass]: true,
          [theme[props.size]]: props.size && !props.floating,
          [theme.rounded]: props.rounded && !props.roundedFull,
          [theme.roundedFull]: props.roundedFull,
          [props.className]: true,
          [theme.disable]: props.disabled,
          [theme.floating[props.size]]: props.floating,
          [theme.fullWidth]: props.fullWidth,
          uppercase: props.uppercase,
          [active]: props.active && !props.disabled,
        })}
      >
        <slot></slot>
      </props.tag>
    )
  }
}
```



link button has a background on hover and also has a default padding.

Click the button to see the effect



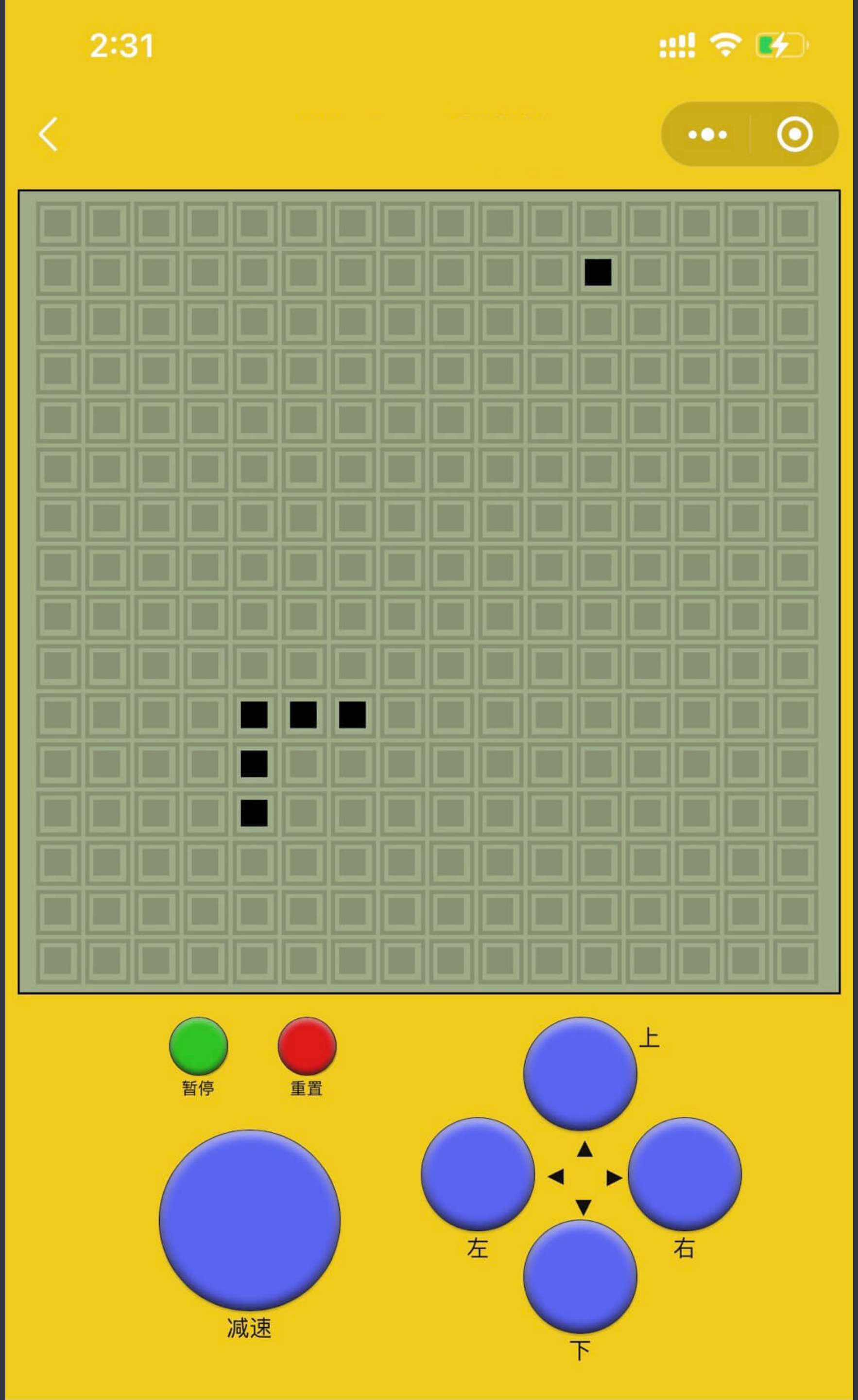
at the link button has a background on hover and also has a default padding.

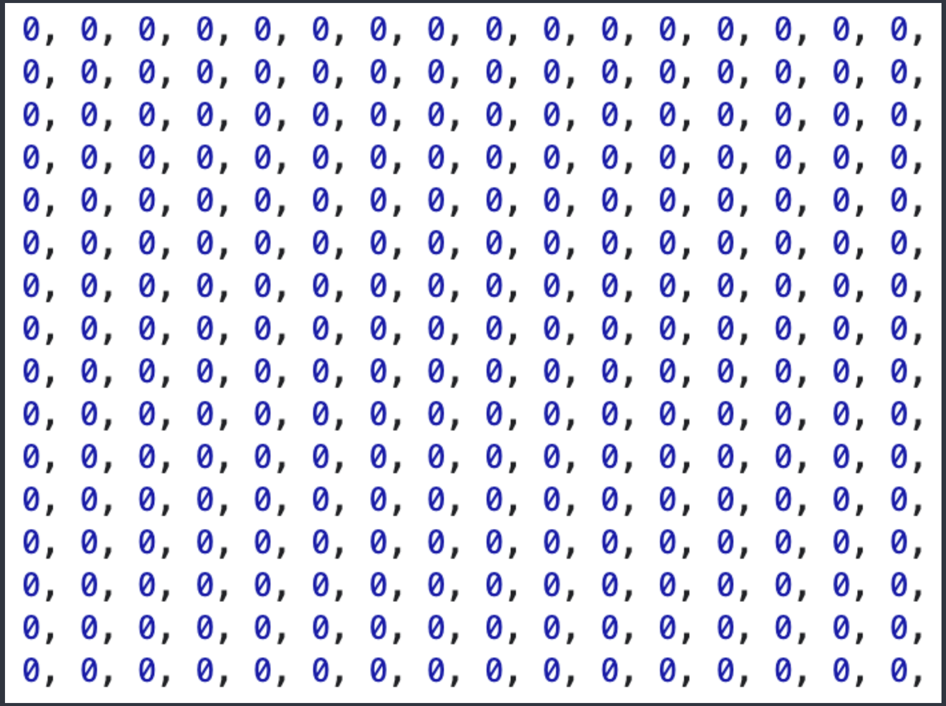
Click the button to see the effect

```
     case 'secondary':
     case 'light':
     case 'dark':
      btnClass = `bg-${config.bg} text-${config.text} hover:bg-${config.hover} focus:bg-${config.focus}
active:bg-${config.active}`
      active = `bg-${config.active}`
      break
    }

    break
  }

  return (
   <props.tag
    type={props.tag === 'button' ? 'button' : null}
    href={props.tag === 'a' ? props.href : null}
    o-ripple={props.ripple === false ? null : ''}
    class={classNames({
     [theme.common]: true,
     [btnClass]: true,
     [theme[props.size]]: props.size && !props.floating,
     [theme.rounded]: props.rounded && !props.roundedFull,
     [theme.roundedFull]: props.roundedFull,
     [props.className]: true,
     [theme.disable]: props.disabled,
     [theme.floating[props.size]]: props.floating,
     [theme.fullWidth]: props.fullWidth,
     uppercase: props.uppercase,
     [active]: props.active && !props.disabled,
    })}
   >
    <slot></slot>
   </props.tag>
  )
 }
}
```

Part 3

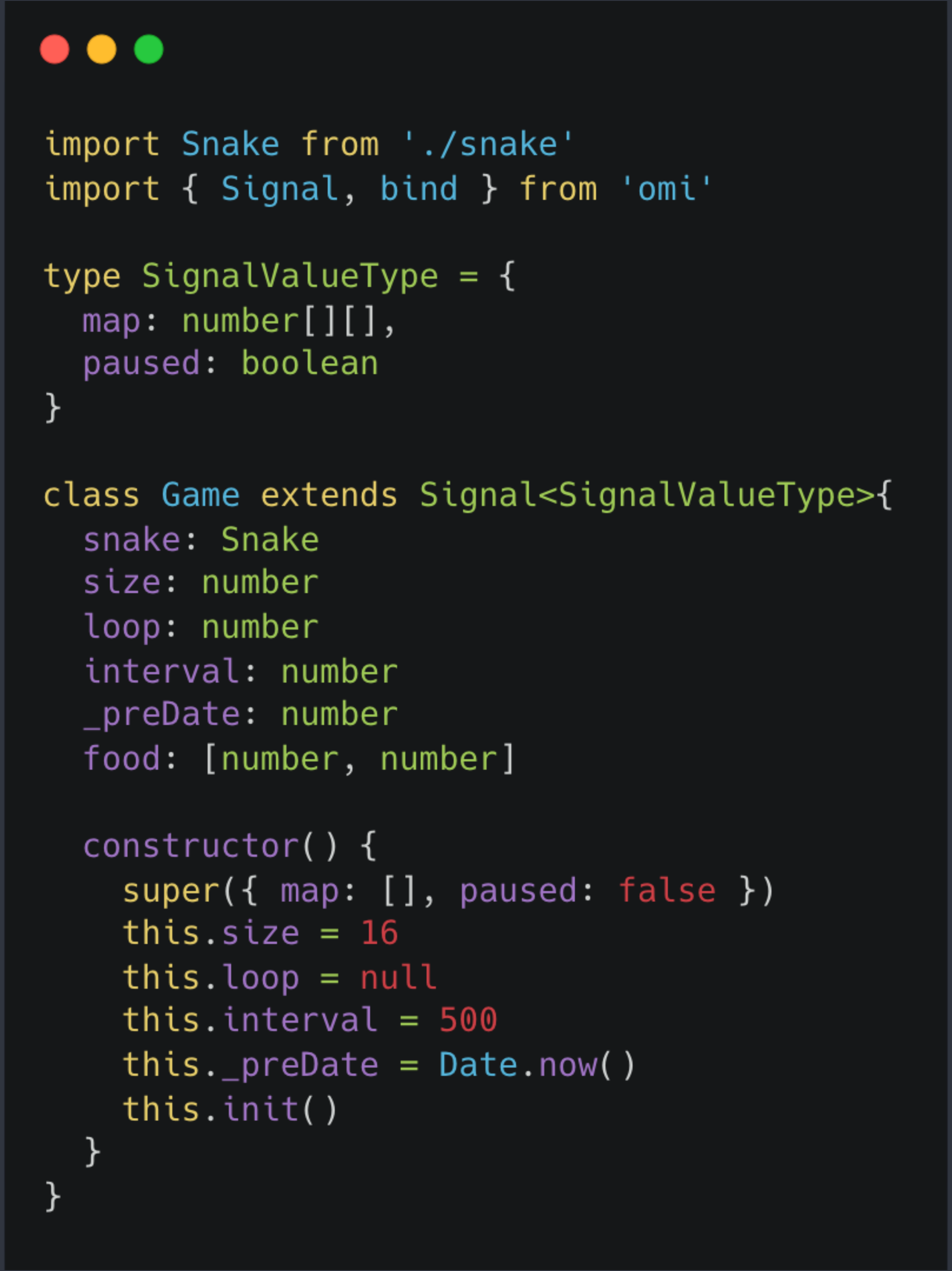
分层



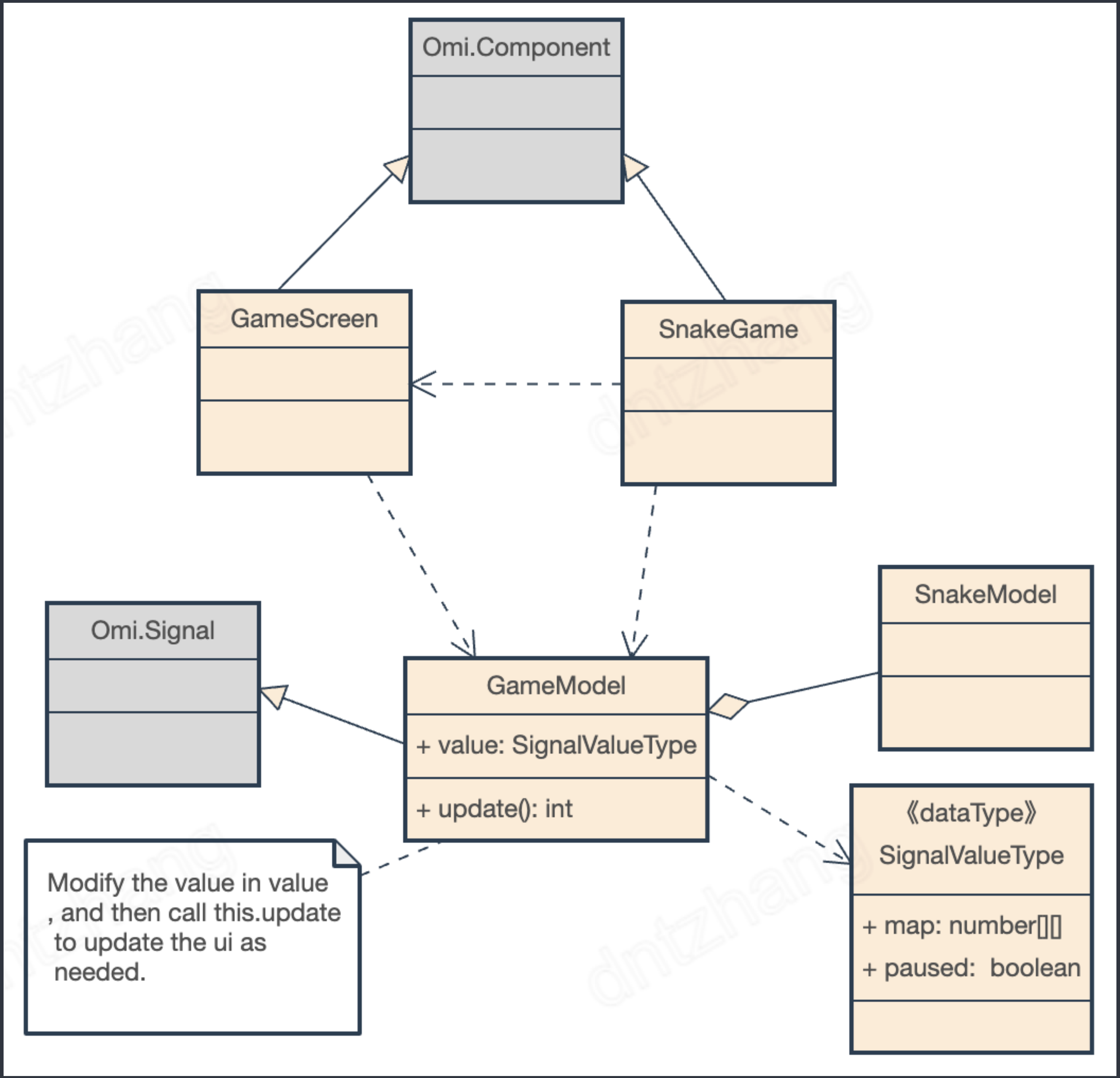


```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

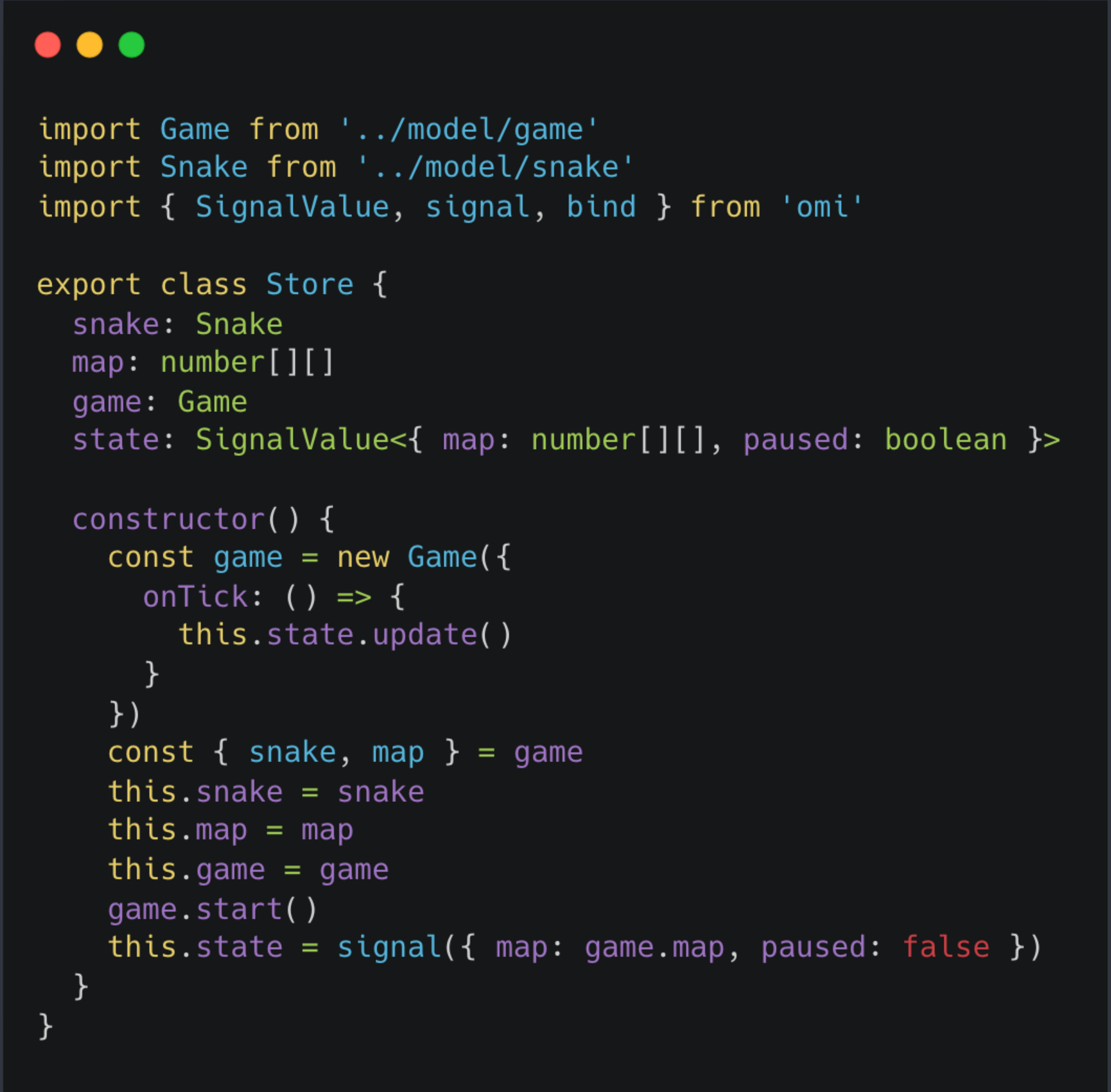
16*16个二维数组代表地图
食物和蛇的位置是1渲染成黑色



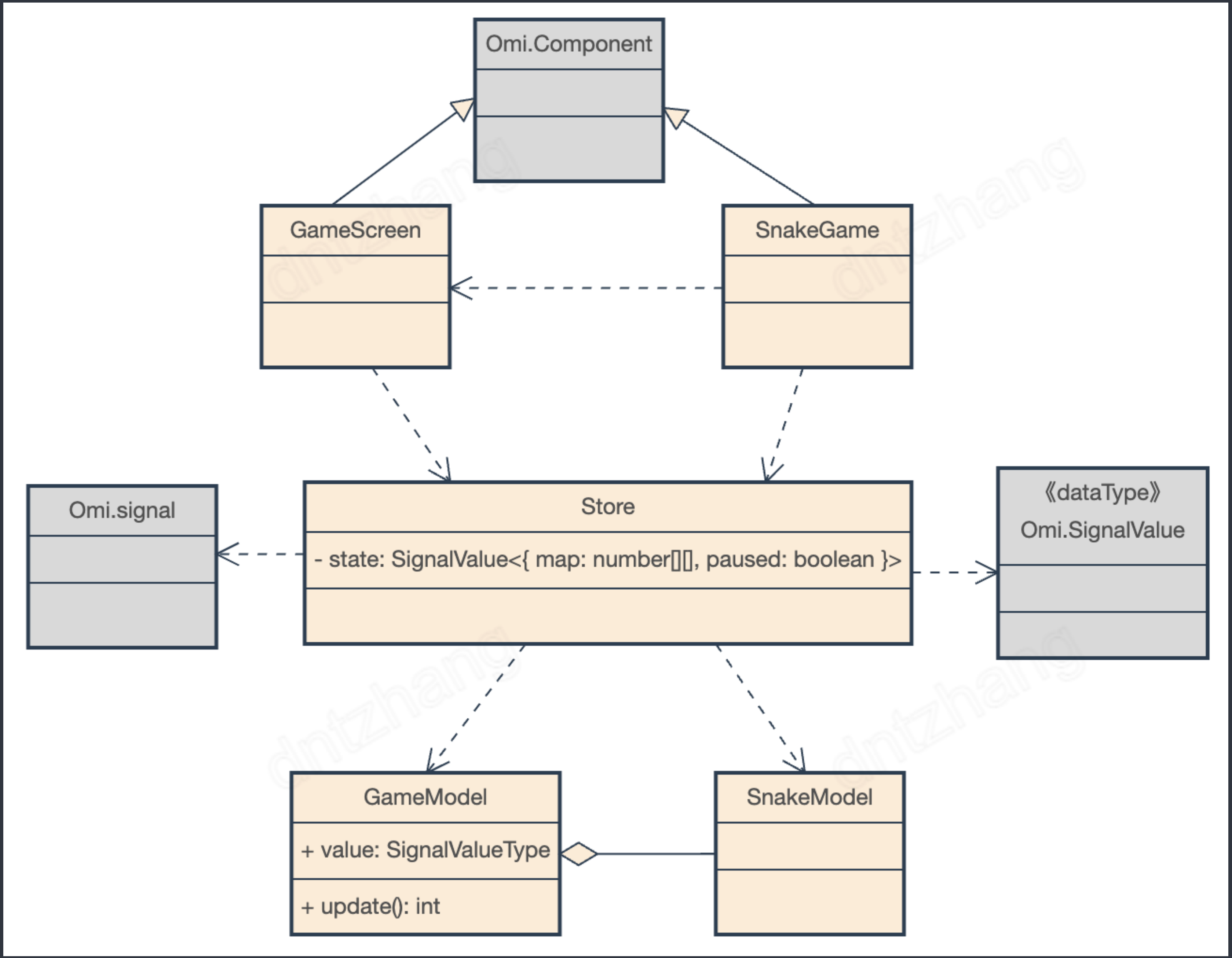
```
import Snake from './snake'
import { Signal, bind } from 'omi'

type SignalValueType = {
  map: number[][],
  paused: boolean
}

class Game extends Signal<SignalValueType>{
  snake: Snake
  size: number
  loop: number
  interval: number
  _preDate: number
  food: [number, number]

  constructor() {
    super({ map: [], paused: false })
    this.size = 16
    this.loop = null
    this.interval = 500
    this._preDate = Date.now()
    this.init()
  }
}
```

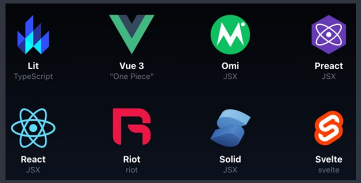


Modify the value in value
, and then call this.update
to update the ui as
needed.



```
import Game from '../model/game'
import Snake from '../model/snake'
import { SignalValue, signal, bind } from 'omi'

export class Store {
  snake: Snake
  map: number[][]
  game: Game
  state: SignalValue<{ map: number[][], paused: boolean }>

  constructor() {
    const game = new Game({
      onTick: () => {
        this.state.update()
      }
    })
    const { snake, map } = game
    this.snake = snake
    this.map = map
    this.game = game
    game.start()
    this.state = signal({ map: game.map, paused: false })
  }
}
```


Omi.Component
GameScreen
SnakeGame
Omi.signal
Store
- state: SignalValue<{ map: number[][], paused: boolean }>
《dataType》
Omi.SignalValue
GameModel
+ value: SignalValueType
+ update(): int
SnakeModel



UI 层

- 提供人机交互界面给 Model 层传输指令

中间层

- 负责把UI层的输入转成Model层需要的格式
- 负责把Model层的输出转成UI层需要的格式

Model 层

- 这个应用要做什么，都在这一层
  - 查询用户列表
  - 增加用户
  - 删除用户
  - 编辑用户

UI 层的实体

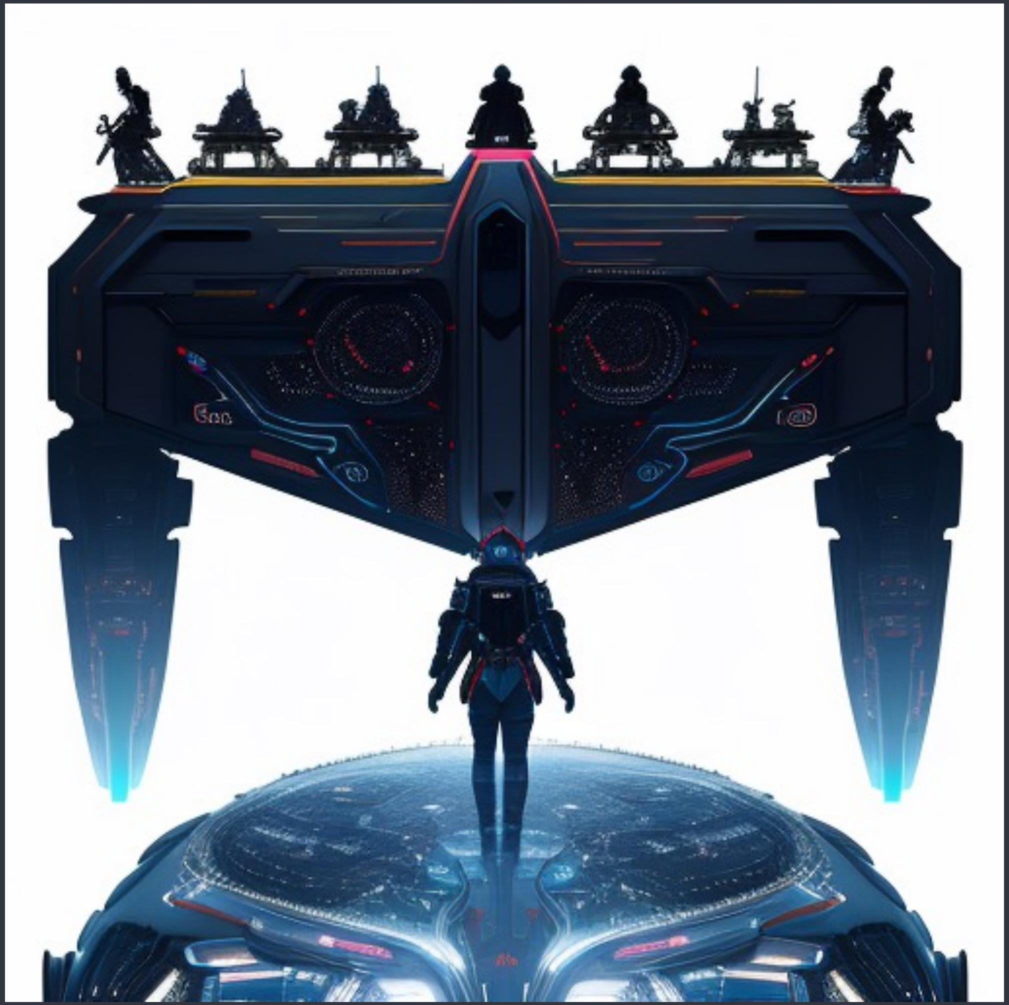
- 静态
  - 渲染引擎
- 交互
  - 框选
  - 拖拽
  - 对齐辅助
  - 快捷键
  - 手势
  - 缩放
- 其他
  - 撤销重做
  - 算法相关模型
  - 工具和辅助相关模型

中间层

Model 层

总结

1. 基于 proxy 的信号 signal 体系是前端响应式答案
2. 看好原子化 CSS，看好 tailwindcss，梭哈，ALL IN
3. 指令可以规避自定义元素 :host 对布局的影响
4. createTreeWalker 非常有用的 DOM API，自顶向下穿透
5. Web components 被定义前到已定义渲染闪烁可以通过 CSS 解决
6. PostCSS 插件可以对 tailwindcss 处理 .dark 来满足 web components 黑白切换
7. 基于原子化 CSS 的组件库是可行的，代码量少，书写便捷，扩展性强
8. UI 是 UI，UI 只是 UI，UI层 不包含 MV*
9. 前端分层开发可以提升项目维护性、扩展性和可测试性
10. 面向数据和面向对象不冲突，通过分层可使两种模式共存
11. 生成式 AI 势不可挡，它站在人类的肩膀上，我们只要站在他的肩膀上，就等于站在人类的肩膀上



Thanks

omijs.org