My name is Cory Rylan, Senior Front End Developer at Vintage Software and Angular Boot Camp instructor. I specialize in creating fast, scalable, and responsive web applications.

Follow @coryrylan

Listening to Angular Key Events with Host Listeners

Cory Rylan
Jan 6, 2017 - 3 min read

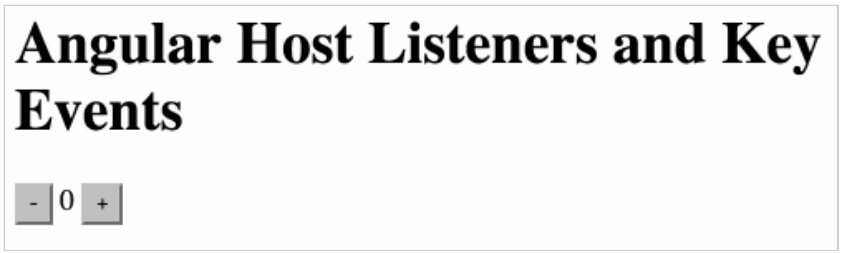
This article is for versions of Angular 2, Angular 4, Angular 5 and later.

A common pattern in many web applications is the ability to react to users via key board events or shortcuts. This great for user experience and accessibility. Typically we register window and document events to accomplish this. With Angular we try to avoid touching the DOM directly for certain rendering and performance reasons. There is a specific API within Angular we can use to listen to global window and document events like the `keyup` and `keydown` events.

Host Listeners

To listen to the `window` for events we will use the `HostListener` API. This API allows us to register a special listener for events in the browser and then call methods in our components

In our example we have a simple counter component that can increment and decrement a value on the view. Our counter component will have two buttons. We will listen to the `keyup` event to be able to use the keyboard arrow keys for the component. Here is our rendered output:



In our rendered counter component I can increment and decrement the value with the buttons and the keyboard arrow keys. Lets take a look at the component template first.

```
<h1>Angular Host Listeners and Key Events</h1>

<button (click)="decrement()">-</button>
{{value}}
<button (click)="increment()">+</button>
```

So in our template we show a single property `value` and then have two buttons that call the `increment()` and `decrement()` methods. Take notice that all of the code to listen to key events will be in the TypeScript side of the component unlike the local `click` events in the template.

```
import { Component, HostListener } from '@angular/core';

export enum KEY_CODE {
  RIGHT_ARROW = 39,
  LEFT_ARROW = 37
}

@Component({
```

```
})
export class AppComponent {
  value = 0;
  constructor() { }

  @HostListener('window:keyup', ['$event'])
  keyEvent(event: KeyboardEvent) {
    console.log(event);

    if (event.keyCode === KEY_CODE.RIGHT_ARROW) {
      this.increment();
    }

    if (event.keyCode === KEY_CODE.LEFT_ARROW) {
      this.decrement();
    }
  }

  increment() {
    this.value++;
  }

  decrement() {
    this.value--;
  }
}
```

The first thing in our component is to notice the `HostListener` decorator in import in out component. This special decorator is how we can listen to out host events. Out host is essentially the element or document our component is located in. We add the `@HostListener` to the `keyEvent()` method with a few important parameters.

```
@HostListener('window:keyup', ['$event'])
keyEvent(event: KeyboardEvent) {
```

```
    if (event.keyCode === KEY_CODE.RIGHT_ARROW) {
      this.increment();
    }

    if (event.keyCode === KEY_CODE.LEFT_ARROW) {
      this.decrement();
    }
  }
```

The `@HostListener` has two parameters. The first is the name of the host event we would like to listen to. For our use case it will be the `window:keyup` event. The second parameters takes a list of arguments returned by the event you are listening to. So for our `keyup` event Angular will pass us back a copy of the `keyup` event via the `$event` variable. This is similar to the `$event` that we commonly use in our templates to pass back other local events.

Now the the event is registered, every time the event is triggered in the DOM Angular will call our `keyEvent()` method passing in the event. Once we have the event we can check the `KeyboardEvent` for the key code. For our counter component we created a simple TypeScript interface to hold the key codes we care about. When the appropriate key code is returned we call the `increment()` or `decrement()` methods.

DEMO

nutondev • 12 days ago

How do you handle key events when there's 2+ instances of the component? How do you prevent all of them from responding and changing their values?

Reply • Share ›

chandramuralis • 6 months ago

Very well explained... thanks

Reply • Share ›

ALSO ON **CORYRYLAN.COM**

Enforcing Code Coverage in Angular CLI Projects

1 comment • 9 months ago

yfain — Hi Cory,I tried configuring thresholds, but for some reason ng test doesn't fail even if I specify high thresholds. Is there anything else

Angular Component Inheritance and Template Swapping

1 comment • 6 months ago

1antares1 — Excellent, man!.It works.Thank you.Regards.

Fast Offline Angular 2 Apps with Service Workers

23 comments • a year ago

Cory Rylan — Not yet but hopefully soon :)

Create your first Web Component with Stencil JS

1 comment • 3 months ago

Ben Adam — Is there a ts to tsx transcompiler available? Will all my code Angular code in Iconic work? Do you know why they decided to go with