API > @angular/core

EventEmitter **CLASS** FINAL

Use in components with the `@Output` directive to emit custom events synchronously or asynchronously, and register handlers for those events by subscribing to an instance.

```
class EventEmitter<T> extends Subject<T> {
  constructor(isAsync?: boolean): EventEmitter<T>
  emit(value?: T): void
  subscribe(next?: (value: T) => void, error?: (error:
any) => void, complete?: () => void): Subscription
}
```

See also

- Observables in Angular

Constructor

Creates an instance of this class that can deliver events synchronously or asynchronously.

This class is "final" and should not be extended. See the public API notes.

```
constructor(isAsync?: boolean): EventEmitter<T>
```

Parameters

isAsync	boolean	When true, deliver events asynchronously. Optional. Default is `false`.

Returns

`EventEmitter<T>`

Methods

emit()

Emits an event containing a given value.

```
emit(value?: T): void
```

Parameters

value	T	The value to emit. Optional. Default is `undefined`.

Returns

`void`

subscribe()

Registers handlers for events emitted by this instance.

```
subscribe(next?: (value: T) => void, error?: (error: any)
=> void, complete?: () => void): Subscription
```

Parameters

next	`(value: T) => void`	When supplied, a custom handler for emitted events. Optional. Default is `undefined`.
error	`(error: any) => void`	When supplied, a custom handler for an error notification from this emitter. Optional. Default is `undefined`.
complete	`() => void`	When supplied, a custom handler for a completion notification from this emitter. Optional. Default is `undefined`.

Returns

`Subscription`

subscribe()

```
subscribe(observerOrNext?: any, error?: any, complete?: any): Subscription
```

Parameters

observerOrNext	any	When supplied, a custom handler for emitted events, or an observer object. Optional. Default is `undefined`.
error	any	When supplied, a custom handler for an error notification from this emitter. Optional. Default is `undefined`.
complete	any	When supplied, a custom handler for a completion notification from this emitter. Optional. Default is `undefined`.

Returns

`Subscription`

Usage notes

Extends RxJS `Subject` for Angular by adding the `emit()` method.

In the following example, a component defines two output properties that create event emitters. When the title is clicked, the emitter emits an open

or close event to toggle the current visibility state.

```
@Component({
  selector: 'zippy',
  template: `
  <div class="zippy">
    <div (click)="toggle()">Toggle</div>
    <div [hidden]="!visible">
      <ng-content></ng-content>
    </div>
 </div>`})
export class Zippy {
  visible: boolean = true;
  @Output() open: EventEmitter<any> = new
EventEmitter();
  @Output() close: EventEmitter<any> = new
EventEmitter();

  toggle() {
    this.visible = !this.visible;
    if (this.visible) {
      this.open.emit(null);
    } else {
      this.close.emit(null);
    }
  }
}
```

Access the event object with the `$event` argument passed to the output event handler:

```
<zippy (open)="onOpen($event)"
(close)="onClose($event)"></zippy>
```