# [Angular2 - Interaction between components using a service](https://stackoverflow.com/questions/41958836/angular2-interaction-between-components-using-a-service)

Shared service is a common way of communication between non-related components. Your components need to [use a single instance of the service](https://stackoverflow.com/questions/36198785/how-do-i-create-a-singleton-service-in-angular-2), so make sure it's provided at the root level.

An example using the[BehaviorSubject as a data delegate](https://stackoverflow.com/questions/34376854/delegation-eventemitter-or-observable-in-angular2):

Shared service:

@Injectable()export class SharedService {

isVisibleSource: BehaviorSubject<boolean> = new BehaviorSubject(false);

constructor() { }}

Component 1:

export class Component1 {

isVisible: boolean = false;

constructor(private sharedService: SharedService) { }

onClick(): void {

this.isVisible = !this.isVisible;

this.sharedService.isVisibleSource.next(this.isVisible);

}}

Component 2:

export class Component2 {

constructor(private sharedService: SharedService) { }

ngOnInit() {

this.sharedService.isVisibleSource.subscribe((isVisible: boolean) => {

console.log('isVisible: ', isVisible); // => true/false

});

}}

It is worth mentioning that BehaviorSubject upon a subscription returns the last value it holds, therefore the component from the example above will be updated with the most recent value immediately after the instantiation.

BehaviorSubject also allows to get its most recent value without even subscribing to it:

this.sharedService.isVisibleSource.getValue(); // => true/false