

In Angular (with RxJS), a BehaviorSubject holds a single value, so you can't directly "set multiple BehaviorSubjects at once" in one call — but you can **update multiple BehaviorSubjects in a single function** or **combine them into one BehaviorSubject that holds an object**.

Here are **two clean approaches**:

### 1 Update Multiple BehaviorSubjects in One Function

If you already have separate BehaviorSubjects, you can create a method that updates them together.

```
import { Injectable } from '@angular/core';
import { BehaviorSubject } from 'rxjs';

@Injectable({ providedIn: 'root' })
export class StateService {
  private userName$ = new BehaviorSubject<string>('');
  private userAge$ = new BehaviorSubject<number>(0);

  // Public observables
  userNameObs$ = this.userName$.asObservable();
  userAgeObs$ = this.userAge$.asObservable();

  // Update both at once
  setUserData(name: string, age: number): void {
    this.userName$.next(name);
    this.userAge$.next(age);
  }
}
```

**Usage in a component:**

```
constructor(private state: StateService) {}

ngOnInit() {
  this.state.setUserData('Alice', 30);
}
```

✅ **Pros:** Keeps BehaviorSubjects separate but still updates them in sync.

⚠️ **Cons:** You must remember to call the combined setter.

## 2 Use a Single BehaviorSubject with an Object

Instead of multiple BehaviorSubjects, store all related state in **one BehaviorSubject**.

```
interface UserState {  
  name: string;  
  age: number;  
}  
  
@Injectable({ providedIn: 'root' })  
export class StateService {  
  private userState$ = new BehaviorSubject<UserState>({ name: '',  
age: 0 });  
  
  // Public observable  
  userStateObs$ = this.userState$.asObservable();  
  
  // Update multiple fields at once  
  setUserData(partial: Partial<UserState>): void {  
    this.userState$.next({  
      ...this.userState$.value, // keep existing values  
      ...partial                // update only provided fields  
    });  
  }  
}
```

### Usage:

```
this.state.setUserData({ name: 'Bob', age: 25 });  
this.state.setUserData({ age: 26 }); // updates only age
```

✅ **Pros:**

- Only one subscription needed.

- Easy to update multiple fields at once.

**Cons:**

- All subscribers get notified even if only one field changes.

## Recommendation

If the values are **logically related** (e.g., user profile data), use **Approach 2** with a single BehaviorSubject holding an object.

If they are **independent pieces of state**, keep separate BehaviorSubjects but update them together via a method (Approach 1).

If you want, I can also show you **how to combine multiple BehaviorSubjects into one observable** so components can react to all changes in sync.

Do you want me to prepare that example?