PRACTICE QUIZ

Quiz 33

3 questions

To Pass 100% or higher

Deadline June 18, 11:59 PM PDT

Start

Quiz 33

Practice Quiz, 3 questions

3/3 points (100%)

Congratulations! You passed!

Next Item

1 / 1 points

1. In implementing the Publish-Subscribe design pattern, what does Angular use for the "channel" portion of the pattern?

- Global object (window)
- Shared Services
- $scope

Correct

- $timeout

1 / 1 points

2. If your component is set up to listen for some event using the $rootScope, what must you ensure to do in such a setup? For example, what must be done (if ANYTHING!) when you see the following in your component:

```
$ctrl.$onInit = function () {
  $rootScope.$on('alert:turnOn', handler);

  function handler(event, data) {
    //...
  }
}
```

Quiz 33

Practice Quiz, 3 questions

3/3 points (100%)

- Aside from implementing the handler function, not much left to do. Angular takes care of the rest.
- The handler function must call the $emit method with the event name of 'alert:turnOff'.
- Weeeeeee! Ok, clearly not the right answer, but so much fun! 😀
- We must deregister the handler when this view is destroyed. We can do this by declaring a local to controller variable called 'cancelHandler' and changing this line 2 to this:

```
1  cancelHandler = $rootScope.$on('alert:turnOn', handler);
```

Then, add another lifecycle method to the component like this:

```
1  $ctrl.$onDestroy = function () {
2    cancelHandler();
3  };
```

Correct

3. Is it ever OK to call $rootScope.$emit(...)?

- No, never.

Quiz 33

Practice Quiz, 3 questions

3/3 points (100%)

```
2   cancelHandler();
3 };
```

Correct

---

1 / 1 points

3. Is it ever OK to call $rootScope.$emit(...)?

- No, never.
- Yes, calling $emit on the $rootScope would make ONLY the $rootScope in the application have a chance at catching that event.

Correct

While this is true, if you're finding yourself doing something like that, it probably means your architecture is messed up... USUALLY, you shouldn't be forced into such communications.