Ready, fire, aim

Letting UI respond to failure

Getting closure through closure actions

Andy Boughton / Center for Open Science

May 23, 2017

Actions: example

```
actions: {
    doSomething() {
        console.log('Hi!');
    }
}
```

```
<button onclick={{action 'doSomething'}}>Click me</button>
```

More than just functions

Actions are good for templates/ user interaction:

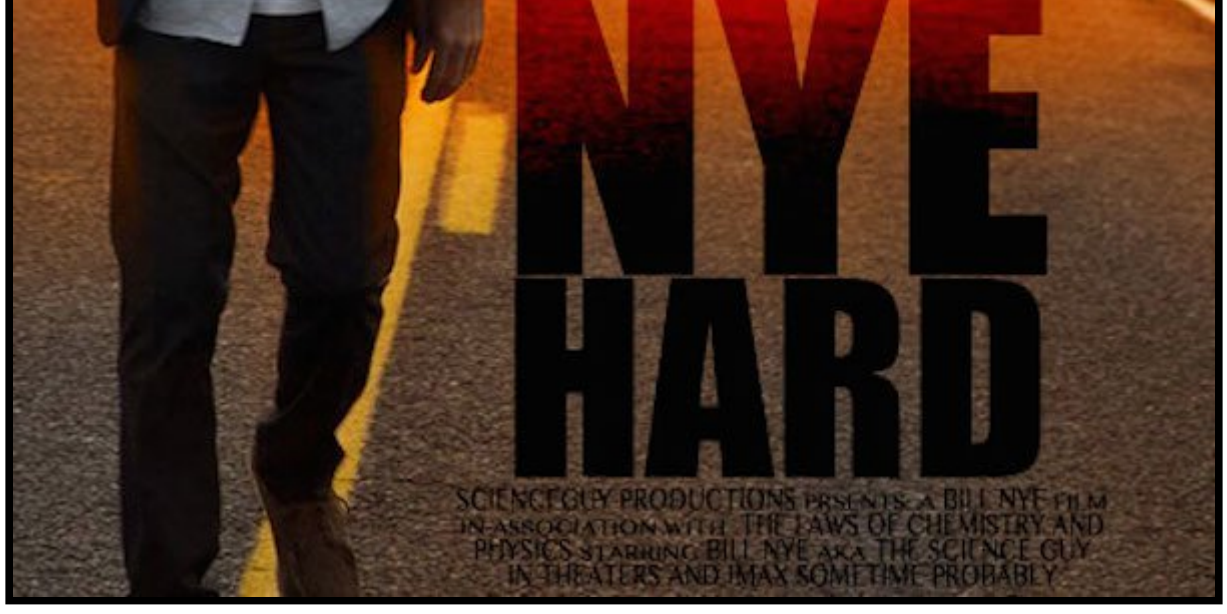
- Preserve a reference to their parent scope
- Can pass shared logic to child components
- Can pass params via templates
- Support advanced behaviors like currying

Your app will often need a way to let users interact with controls that change application state.

Actions: Don’t Look Back

Traditionally passed Ember actions are "fire and forget". For a save button, this can be very dangerous!!

```
this.send('goTalkToAServer');
```


NYE
HARD
SCIENCEGUY PRODUCTIONS PRESENTS A BILL NYE FILM
IN ASSOCIATION WITH THE LAWS OF CHEMISTRY AND
PHYSICS STARRING BILL NYE AKA THE SCIENCE GUY
IN THEATERS AND IMAX SOMETIME PROBABLY

ISP: A problem and a solution

Save error / The fix

Closure actions have return values!

The key is that *closure actions can return a promise*
The component calls the action in a special way that consumes the return value.
This is most useful for passed-in actions.

```
/**
* Logic that we passed from a parent.
* Expect an action with call signature method(textValue)
* @property
*/
passedInActionHandle: null,
actions: {
  clickAButtonAndSaveTheModel() {
    this.attrs.passedInActionHandle().catch(this.showErrorZOMG);
  }
}
```

Don’t mix and match

Old style Ember actions worked differently- returning a truthy value triggered *action bubbling*. Avoid mixing old and new style actions.

If the controller does not implement a method with the same name as the action in its actions object, the action will be sent to the router, where the currently active leaf route will be given a chance to handle the action... To continue bubbling the action, you must return true from the handler... This allows you to create a button that has different behavior based on where you are in the application.

Bonus: currying

The template action helper supports *currying*- a way to build partial functions that bake in some arguments

Taggable mixin / Usage example

An example without currying (NodeActions)

Bonus: no middle actions

Without bubbling, you don't need `this.send` boilerplate in every intermediate component. This makes layering components nicer- just keep passing the action handle down

References

- Intro to closure actions (in practice)
- Intro to closure actions (good conceptual review)
- Closure actions have return values (async example)
- Bubbling vs closure actions
- Benefits of closure actions