FRONT END FRAMEWORKS

As you work in front end JS, you find some common tasks

- (re)generating HTML based on variables
- Making service calls
- Reporting errors to the user
- Repeating portions of output
- Dealing with browser/version differences

COMMON LIBRARIES

Libraries package some of this common code in a useful interface

- jQuery (DOM searching/manpulation, service calls, browser diffs)
- Moustache, HandlebarsJS (templating)
- A bajillion others

FRAMEWORKS

A "Framework" does more than provide an interface to common code

It provides a restricted path of behavior, but grants convenient interaction with common needs

MODERN LIBRARIES/FRAMEWORKS

People will often argue over whether a package is a library or a framework

With passion

Others happily claim only one title

- Backbone
- Ember
- Meteor
- Angular (Angular vs AngularJS)
- React
- Vue

WHICH LIBRARY/FRAMEWORK IS THE BEST?

Depends on a lot - everything has strengths and weaknesses so there's not going to be one single "best".

Like asking "What is the best language?" - If there was one clear best there wouldn't be so many.

WHY REACT?

- React is scoped to View only - unopinionated about the rest
  - Which makes it more flexible
- React is Declarative
  - Makes it less complex to read/change
- React maps closely to HTML output
- React components map closely to JS functions
  - Means same best practices
  - Means non-React skills increase React skills
  - Means React skills are useful even without React
- Also very popular right now
  - Means helps with jobs