INFINUM

Keeping React on a tight leash with MobX

DARKO KUKOVEC

01

APP STATE

APP STATE

- Data
  - Which user is logged in?
  - Which todos did the user create?

- UI
  - Which todos is the user looking at (filter – all, complete, incomplete)

02

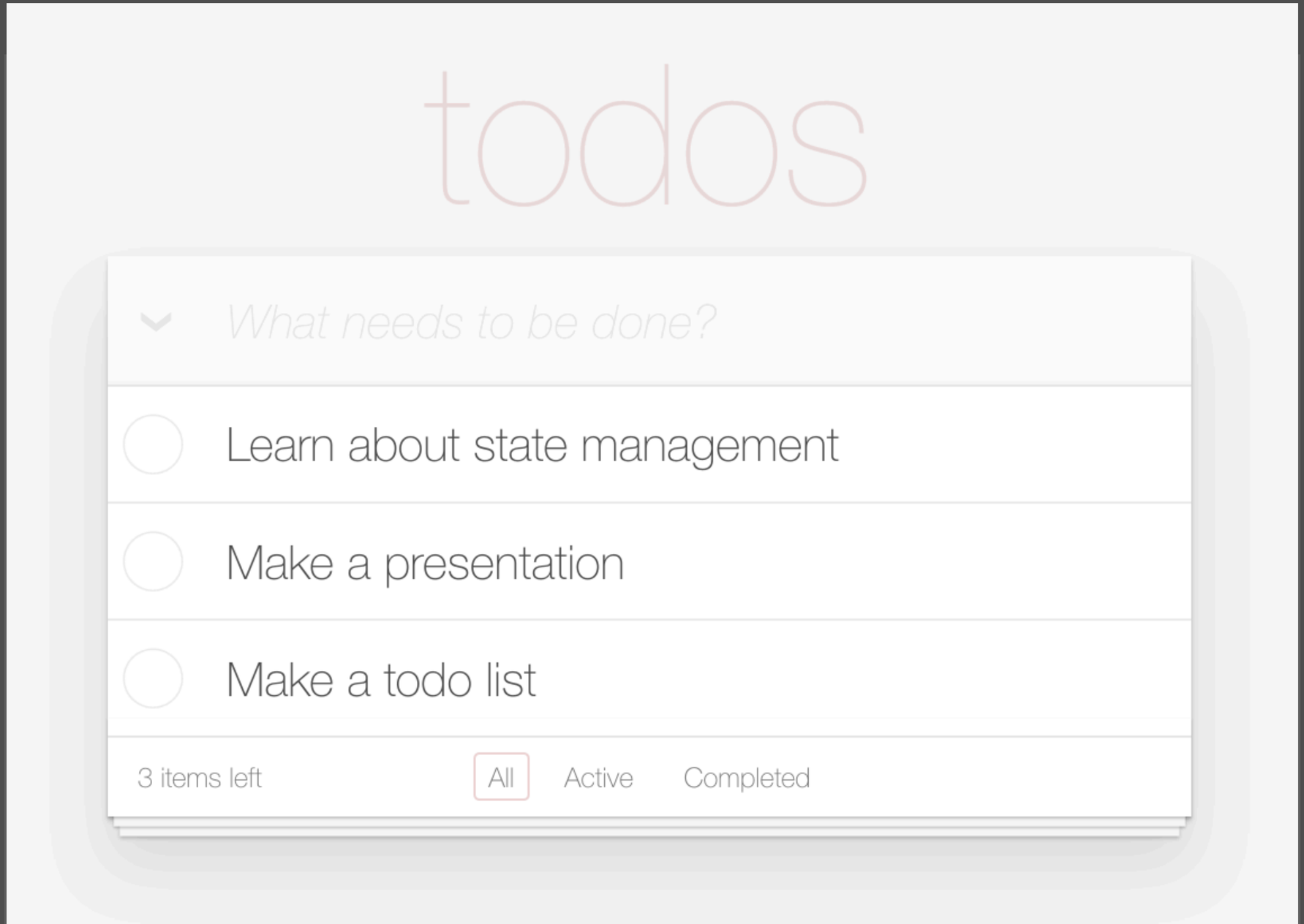
STATE MANAGEMENT

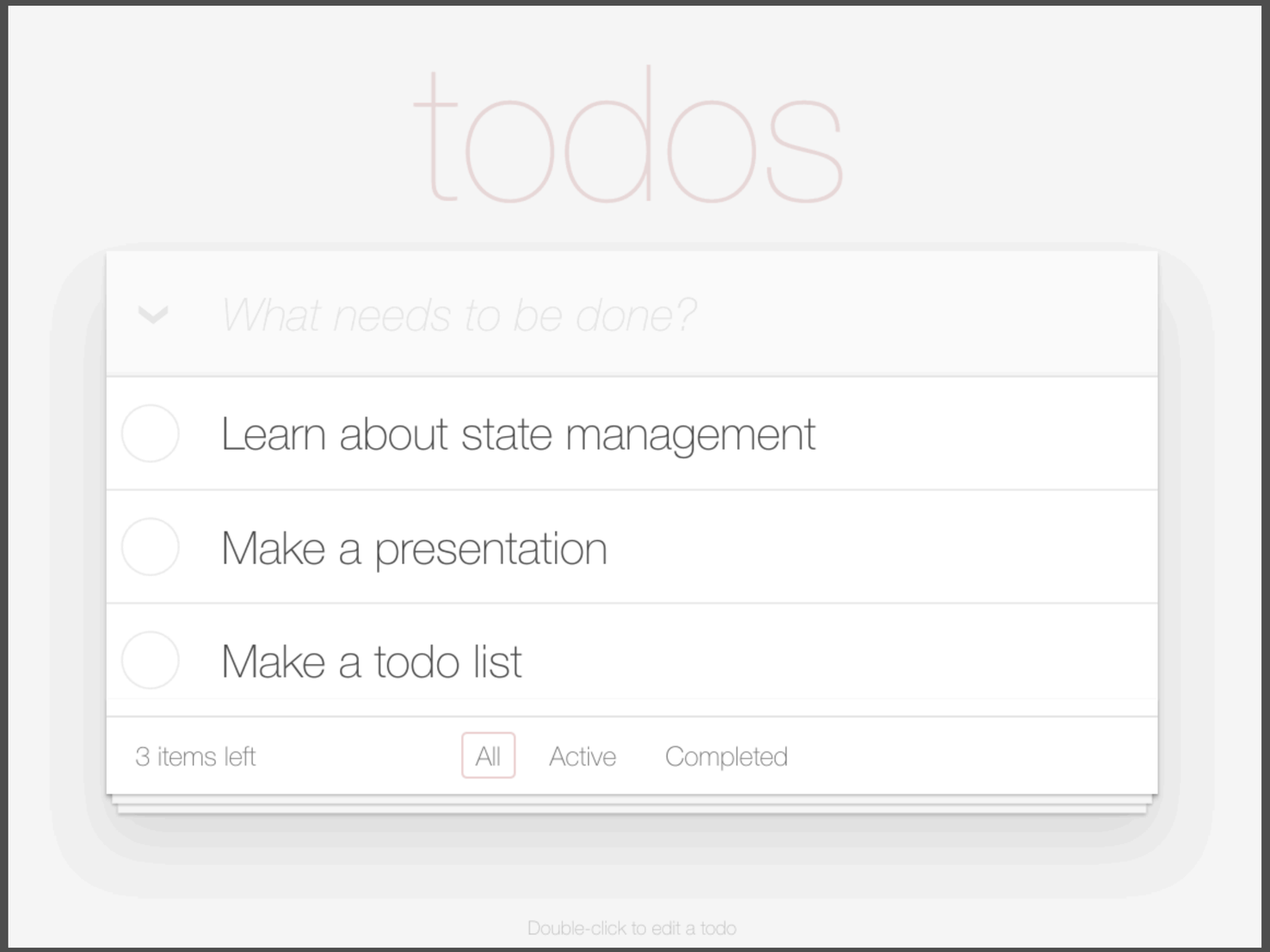
STATE MANAGEMENT

- What should happen when some data changes?
- Does the UI need to be updated
- Does some other data depend on it?
- Does the app need to make an action (e.g. API call)?

03

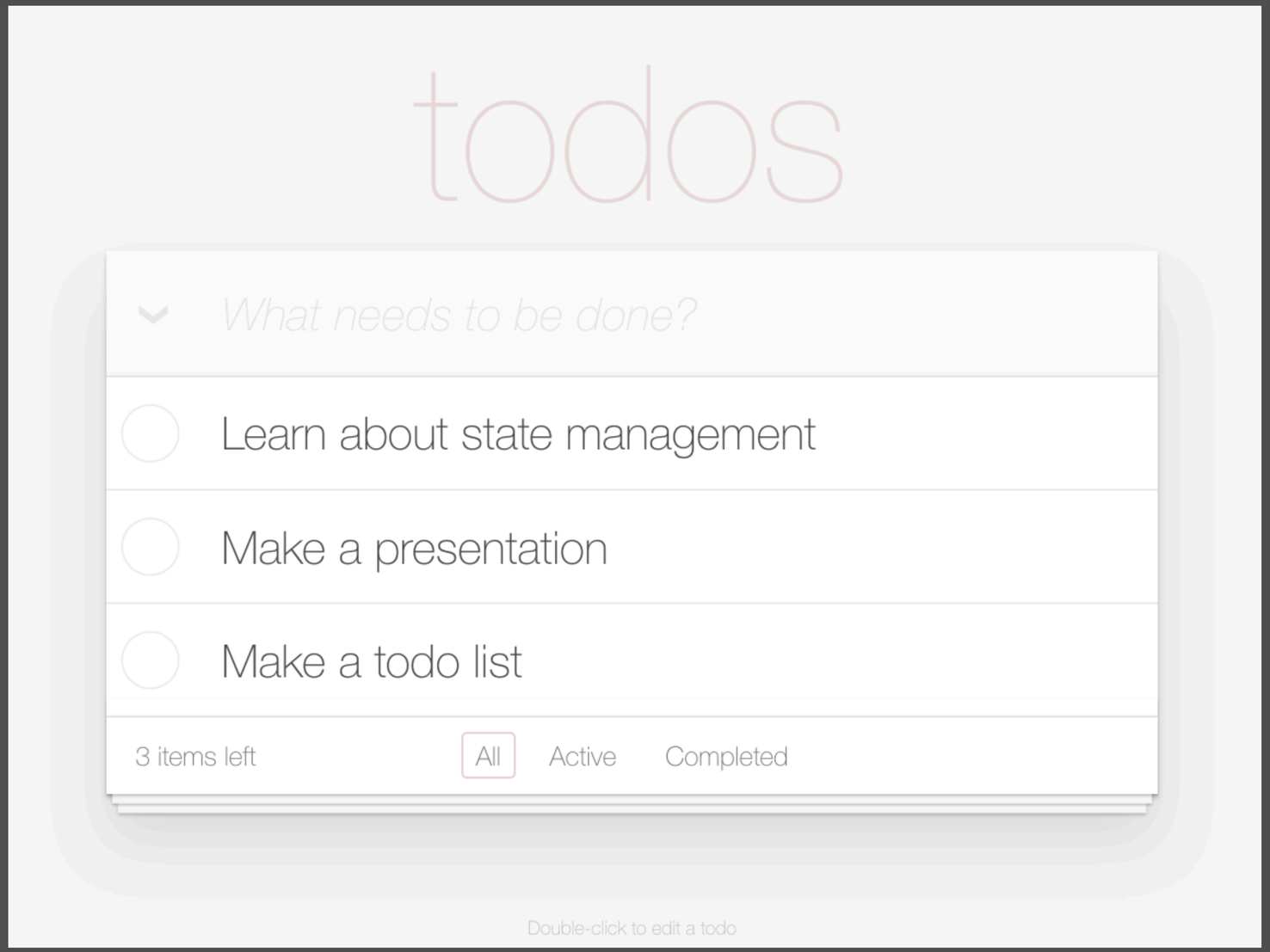
WHY DO WE NEED IT?


todos
What needs to be done?
Learn about state management
Make a presentation
Make a todo list
3 items left
All
Active
Completed

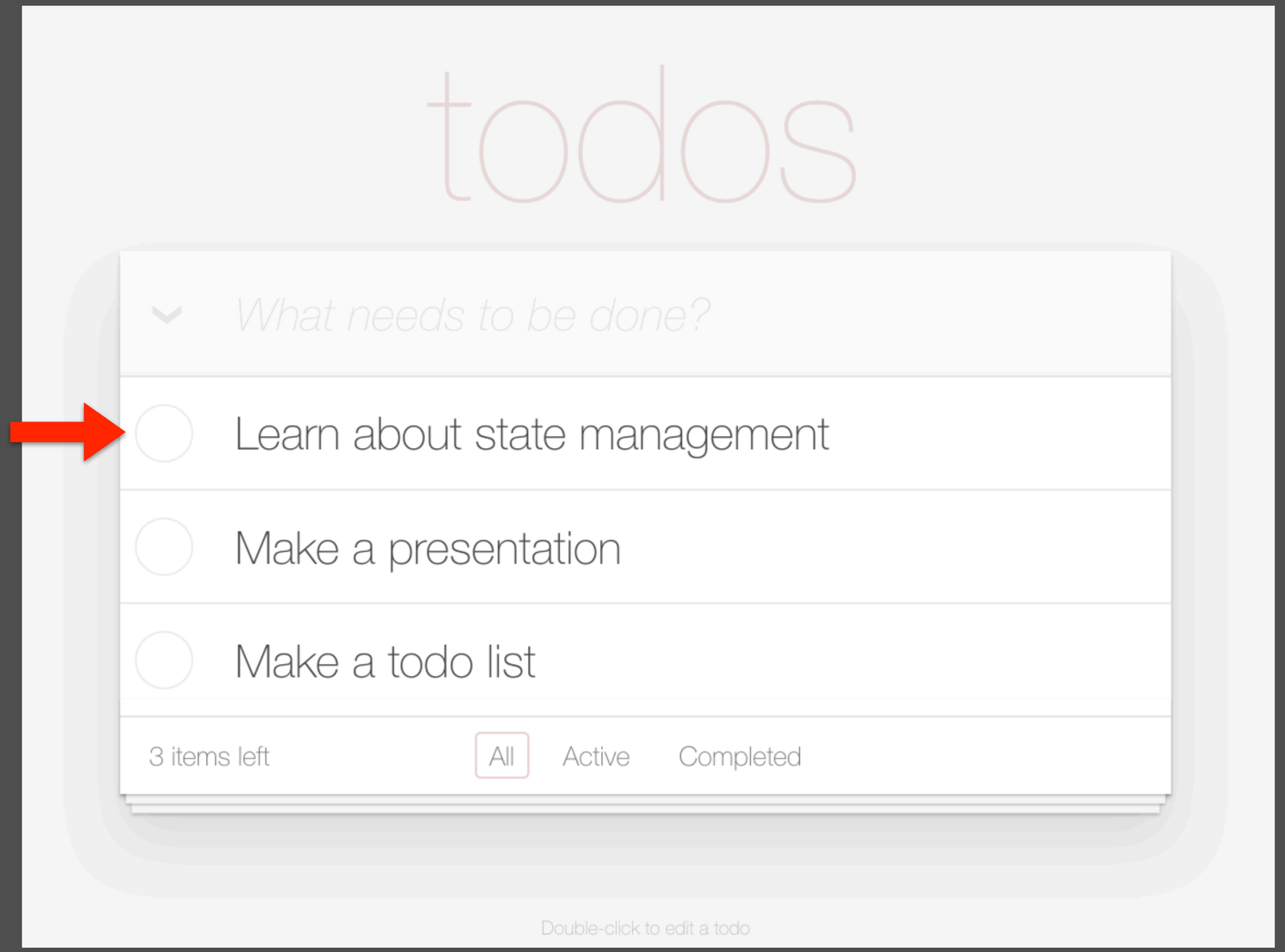


- Todos
  - Title
  - Status
- Active filter
- Number of todos

Handing an action in the app

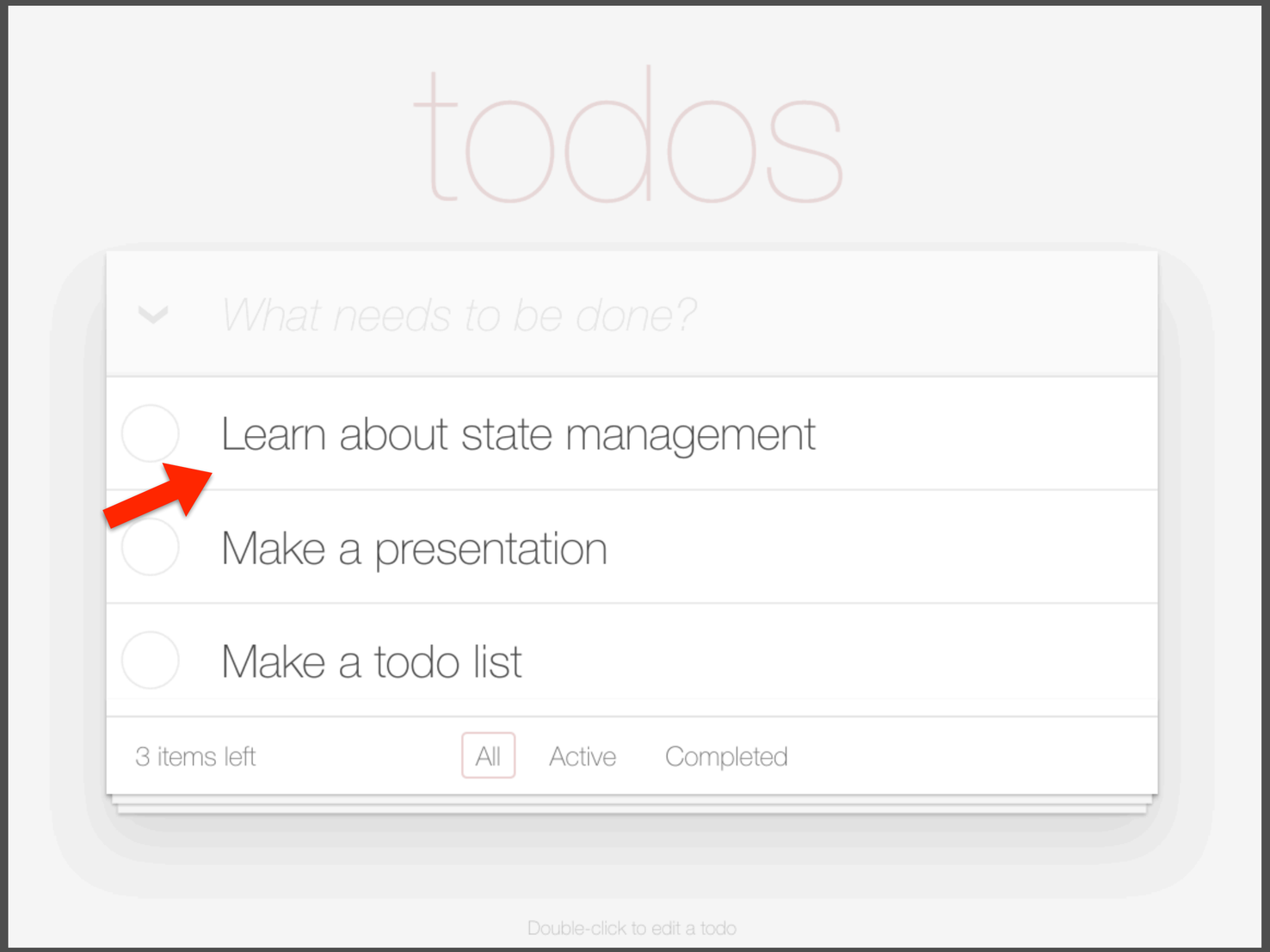


Handing an action in the app



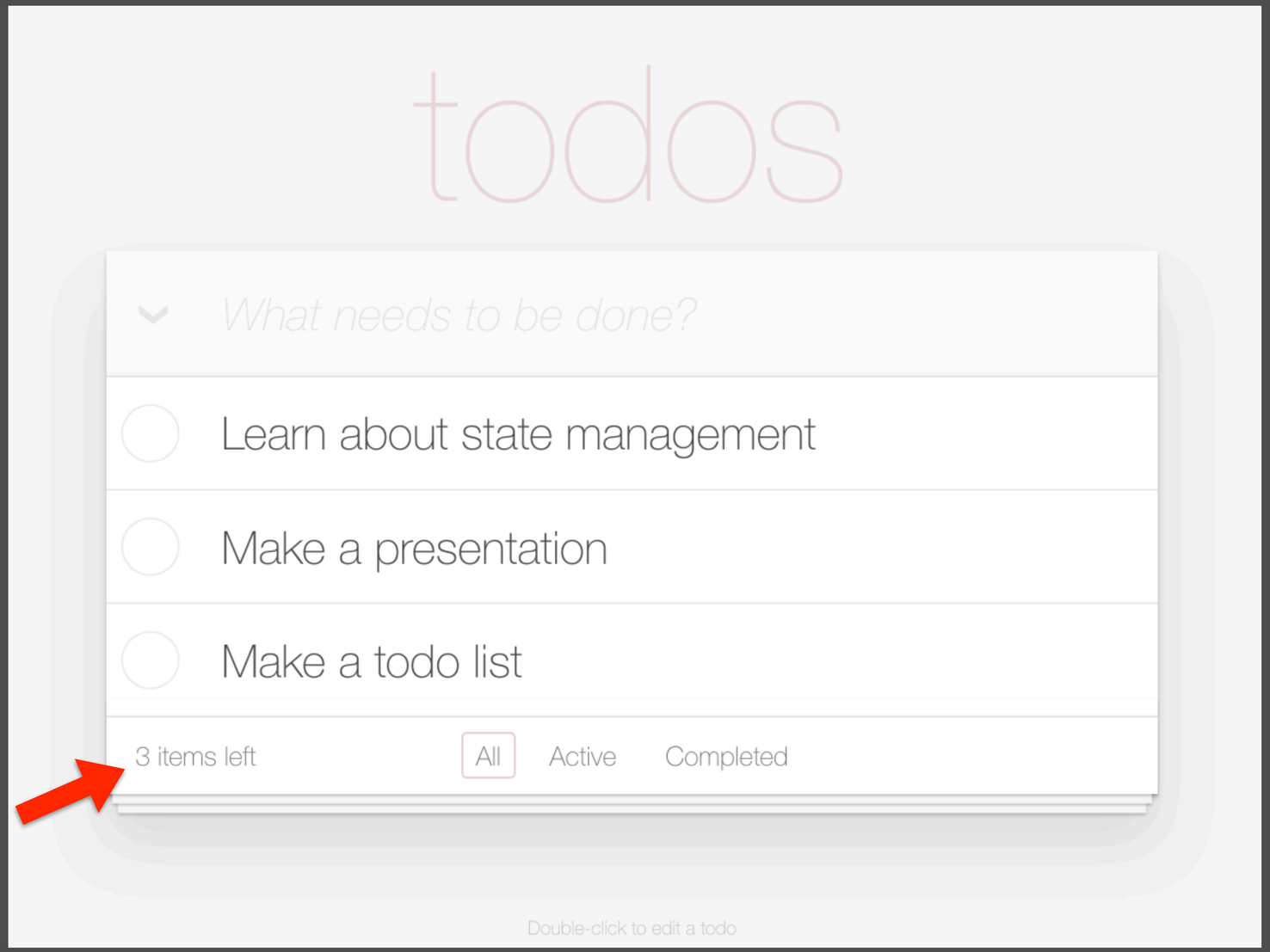
- The user checks a todo as complete

Handing an action in the app



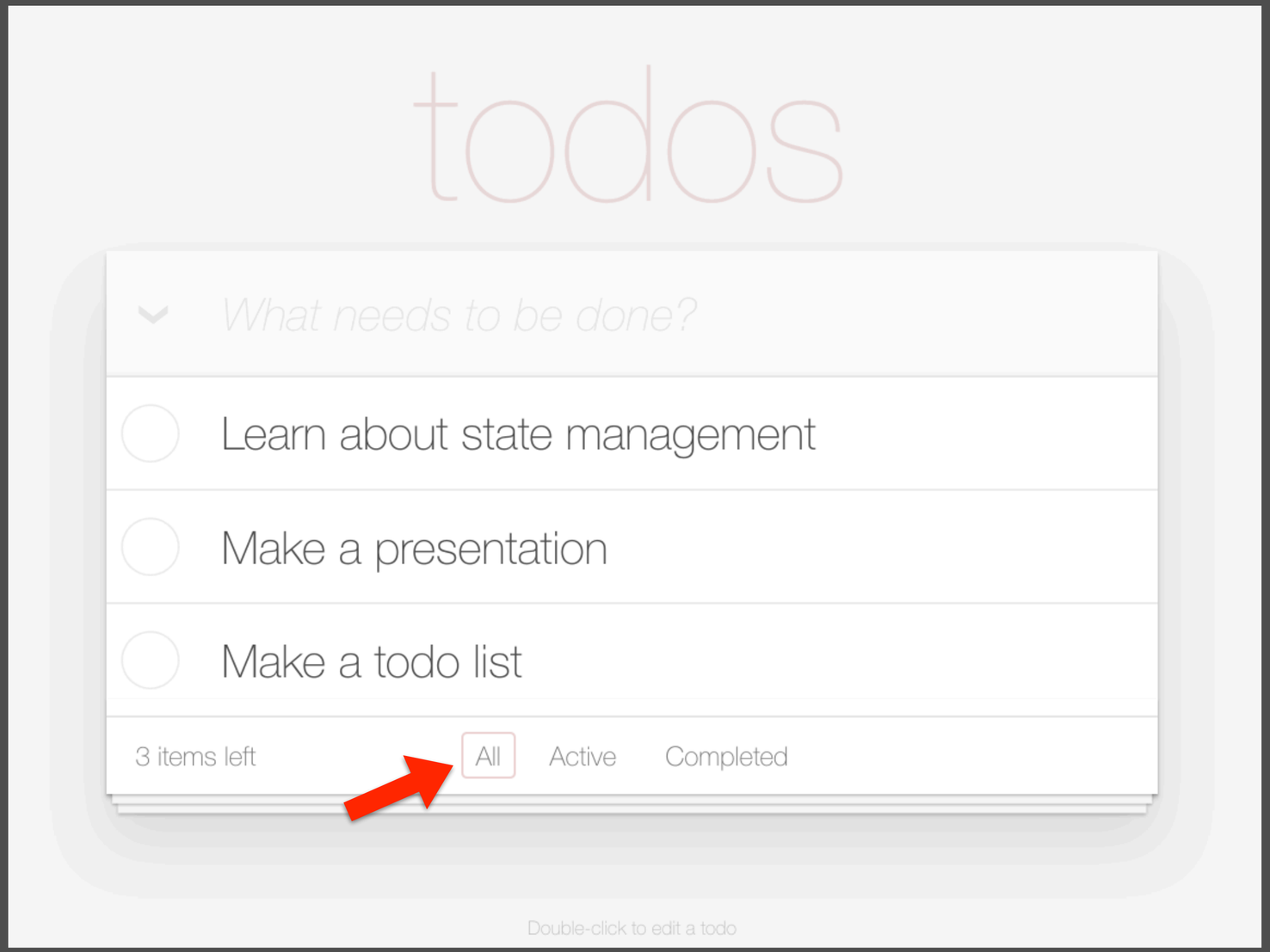
- The user checks a todo as complete
  - Mark the todo as complete

Handing an action in the app



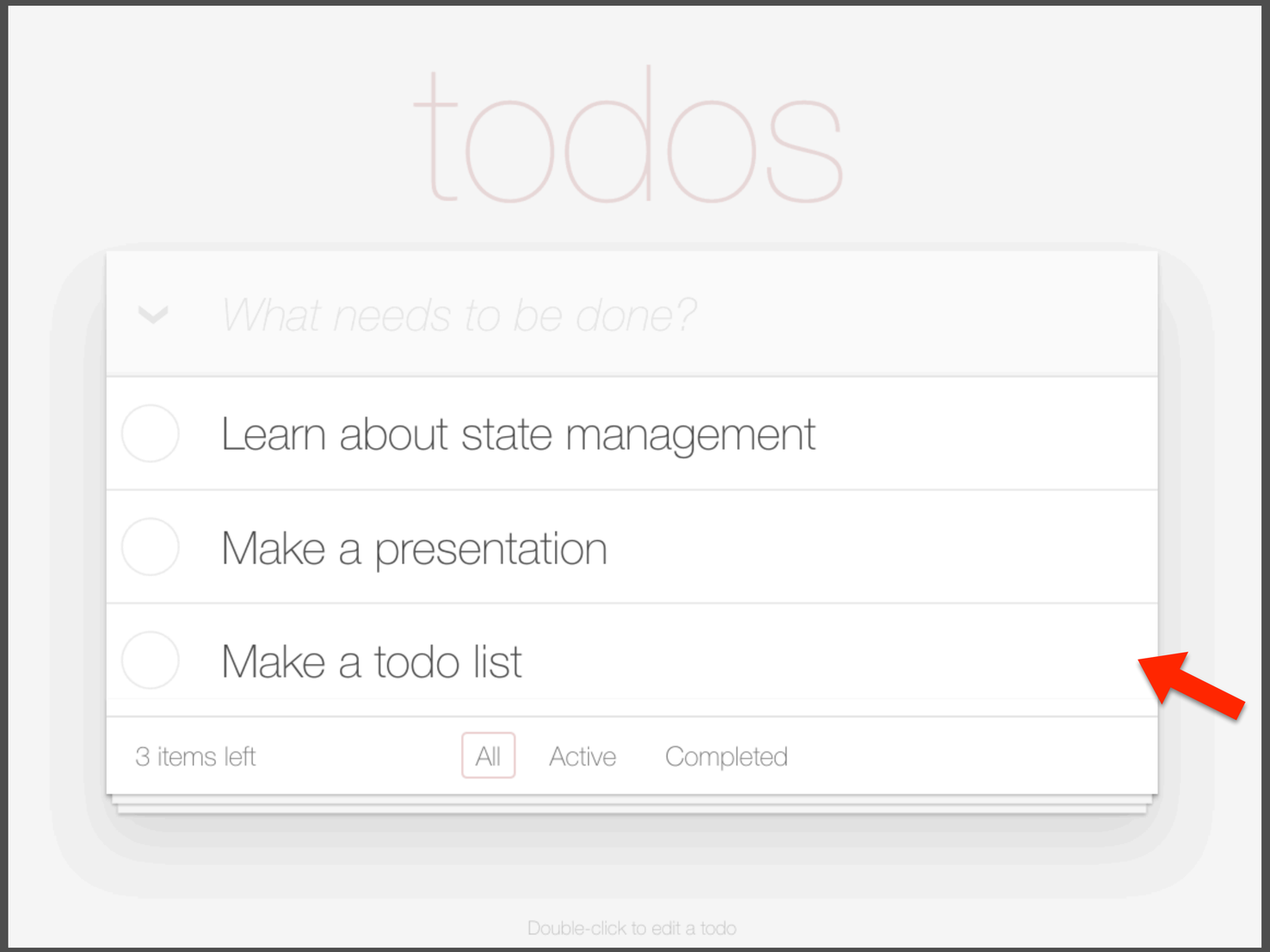
- The user checks a todo as complete
  - Mark the todo as complete
  - Update the incomplete items count

Handing an action in the app



- The user checks a todo as complete
  - Mark the todo as complete
  - Update the incomplete items count
  - What filter is active again?

Handing an action in the app



- The user checks a todo as complete
  - Mark the todo as complete
  - Update the incomplete items count
  - What filter is active again?
  - Do I need to update the list of todos?
    - Add/remove item?
    - Sort items?
  - Should I show an empty state?
  - Should I make an API call?
  - Should I save the change into localStorage?

You’re doing state management, you’re just not realising it

HOW TO DO STATE MANAGEMENT MORE EFFICIENTLY

MVC FRAMEWORKS

Models, services?

ANGULAR 2

RxJS – a way to work with the state in an async way

VUE.JS

Vuex

REACT

Redux

Mobx

Flux

Relay

05

MOBX VS REDUX

ADVANTAGES

MobX

- Faster
- Less boilerplate
- More flexible

Redux

- Easier to (unit) test
- Smaller
- Time travel
- Easier to debug?

USAGE

MobX

- mobx-react
- Not React dependent
- mobx-angular
- Python
- GWT

Redux

- react-redux
- Not React dependent

06

BASIC MOBX CONCEPTS

Anything that can be derived from the application state, should be derived. Automatically.

- The philosophy behind MobX

OBSERVABLE

Your state

e.g., list of TODOs

ACTION

A function that changes the state

e.g., function called after the user clicks on the completed checkbox

COMPUTED PROPERTY

A value derived from your state

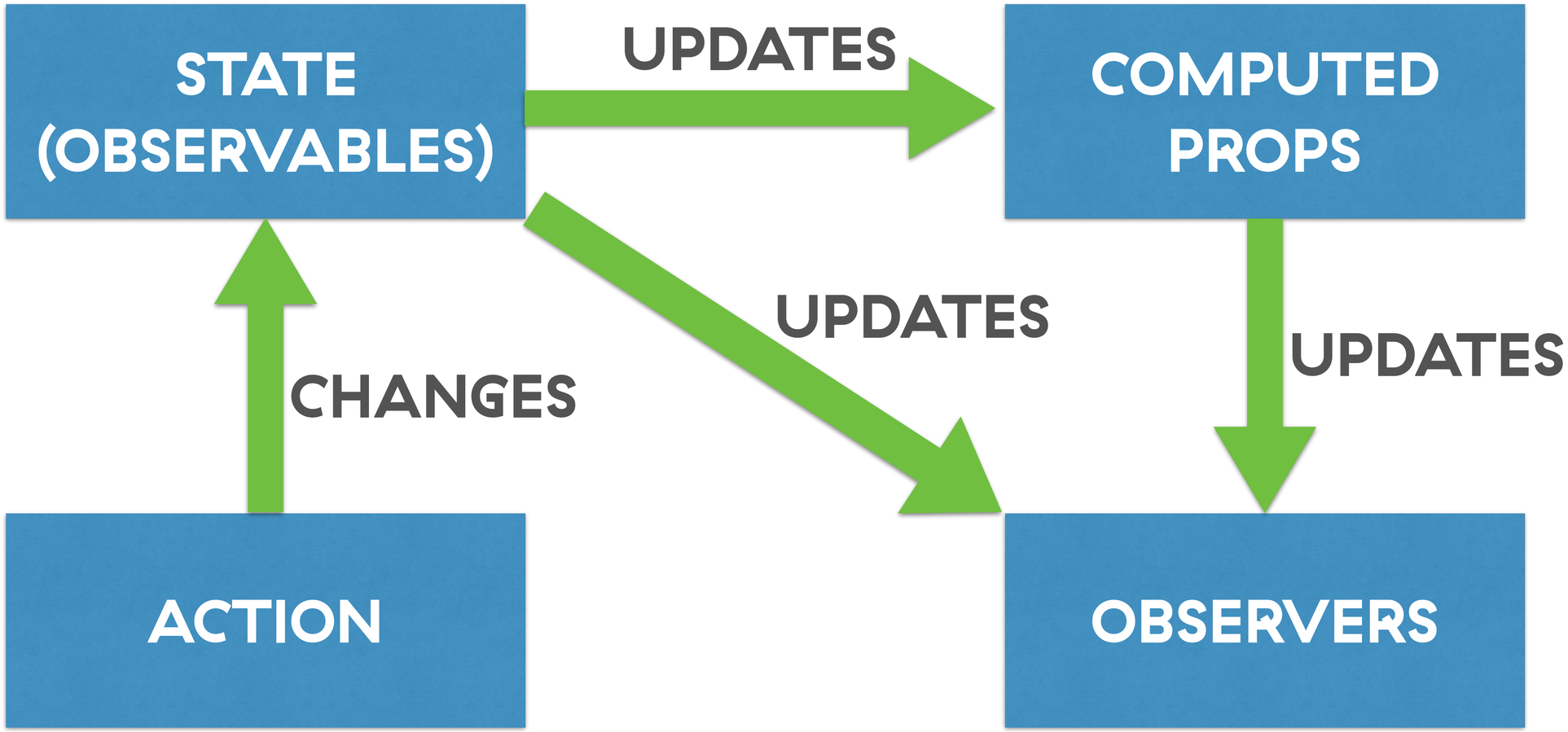
e.g., list of completed TODOs

OBSERVER

Function (component?) that needs to be updated

on state change

e.g., component showing a list of TODOs


STATE
(OBSERVABLES)
UPDATES
COMPUTED
PROPS
UPDATES
UPDATES
CHANGES
ACTION
OBSERVERS

DECORATORS

JS proposed feature (Babel plugin)

TypeScript feature

```
import {observable, computed} from 'mobx';

class Todos {
  @observable list = [];
  users = observable([]);

  @computed get complete() {
    return this.list.filter((todo) => todo.complete);
  }

  incomplete: computed(() => {
    return this.list.filter((todo) => !todo.complete);
  })
}
```

TAKING STATE MANAGEMENT TO THE LIMITS

Fantasy tabletop role-playing game

Fantasy tabletop role-playing game

CHARACTER SHEET

A lot of related data

A lot of computed properties

Dang it I forgot my char sheet!

Adran

Level 4

Paladin	Lawful Neutral	5950
Class	Alignment	Exp. Points
Sage	**Half-elf**	
Background	Race	Subrace

30
Speed

2 Proficiency bonus

Inspiration

13	14	14	16	16	18	16
+1	+2	+2	+3	+3	+4	
Strength	Dexterity	Constitution	Intelligence	Wisdom	Charisma	Passive per.

Skills (WIP)

Search

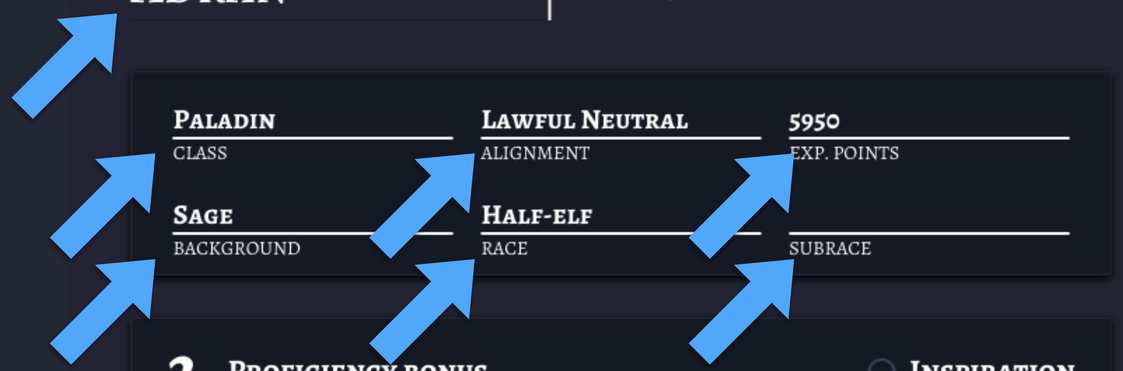
- +2 acrobatics
- +3 animalHandling
- +5 arcana
- +1 athletics
- +4 deception
- +5 history
- +3 insight
- +4 intimidation
- +3 investigation
- +3 medicine
- +3 nature
- +3 perception
- +4 performance
- +4 persuasion
- +3 religion
- +2 sleightOfHand
- +2 stealth
- +3 survival

Saving throws (WIP)

- +6 charisma
- +2 constitution
- +2 dexterity
- +3 intelligence
- +1 strength
- +5 wisdom

Dang it I forgot my char sheet!

Adran | Level 4



Paladin	Lawful Neutral	5950
Class	Alignment	Exp. Points
Sage	**Half-elf**	
Background	Race	Subrace

30
Speed

2 Proficiency bonus

Inspiration

13	14	14	16	16	18
+1	+2	+2	+3	+3	+4
Strength	Dexterity	Constitution	Intelligence	Wisdom	Charisma

16
Passive per.

Skills (WIP)

Search

+2 acrobatics	+3 insight	+4 performance
+3 animalHandling	+4 intimidation	+4 persuasion
+5 arcana	+3 investigation	+3 religion
+1 athletics	+3 medicine	+2 sleightOfHand
+4 deception	+3 nature	+2 stealth
+5 history	+3 perception	+3 survival

Saving throws (WIP)

- +6 charisma
- +2 constitution
- +2 dexterity
- +3 intelligence
- +1 strength
- +5 wisdom

Dang it I forgot my char sheet!

Adran

Level 4

Paladin	Lawful Neutral	5950
Class	Alignment	Exp. Points
Sage	Half-elf	
Background	Race	Subrace

30
Speed

2 Proficiency bonus

Inspiration

Strength	Dexterity	Constitution	Intelligence	Wisdom	Charisma	Passive Per.
13 (+1)	14 (+2)	14 (+2)	16 (+3)	16 (+3)	18 (+4)	16

Skills (WIP)

Search

- +2 acrobatics
- +3 animalHandling
- +5 arcana
- +1 athletics
- +4 deception
- +5 history
- +3 insight
- +4 intimidation
- +3 investigation
- +3 medicine
- +3 nature
- +3 perception
- +4 performance
- +4 persuasion
- +3 religion
- +2 sleightOfHand
- +2 stealth
- +3 survival

Saving throws (WIP)

- +6 charisma
- +2 constitution
- +2 dexterity
- +3 intelligence
- +1 strength
- +5 wisdom

Dang it I forgot my char sheet!

Adran

Level 4

Paladin	Lawful Neutral	5950
Class	Alignment	Exp. Points
Sage	**Half-elf**	
Background	Race	Subrace

30
Speed

2 Proficiency bonus

○ Inspiration

13	14	14	16	16	18	16
+1	+2	+2	+3	+3	+4	
Strength	Dexterity	Constitution	Intelligence	Wisdom	Charisma	Passive per.

Skills (WIP)

Search

○ +2 acrobatics	○ +3 insight	○ +4 performance
○ +3 animalHandling	○ +4 intimidation	○ +4 persuasion
● +5 arcana	○ +3 investigation	○ +3 religion
○ +1 athletics	○ +3 medicine	○ +2 sleightOfHand
○ +4 deception	○ +3 nature	○ +2 stealth
● +5 history	○ +3 perception	○ +3 survival

Saving throws (WIP)

● +6 charisma	○ +3 intelligence
○ +2 constitution	○ +1 strength
○ +2 dexterity	● +5 wisdom

Dang it I forgot my char sheet!

Adran | Level 4

Paladin	Lawful Neutral	5950
Class	Alignment	Exp. Points
Sage	**Half-elf**	
Background	Race	Subrace

30
Speed

2 Proficiency bonus

Inspiration

13	14	14	16	16	18
+1	+2	+2	+3	+3	+4
Strength	Dexterity	Constitution	Intelligence	Wisdom	Charisma

16
Passive Per.

Skills (WIP)

+2 acrobatics	+3 insight	+4 performance
+3 animalHandling	+4 intimidation	+4 persuasion
+5 arcana	+3 investigation	+3 religion
+1 athletics	+3 medicine	+2 sleightOfHand
+4 deception	+3 nature	+2 stealth
+5 history	+3 perception	+3 survival

Saving Throws (WIP)

- +6 charisma
- +2 constitution
- +2 dexterity
- +3 intelligence
- +1 strength
- +5 wisdom

Dang it I forgot my char sheet!

Adran | Level 4

Paladin	Lawful Neutral	5950
Class	Alignment	Exp. Points
Sage	**Half-elf**	
Background	Race	Subrace

2 Proficiency bonus

Inspiration

30

Speed

13	14	14	16	16	18	16
+1	+2	+2	+3	+3	+4	
Strength	Dexterity	Constitution	Intelligence	Wisdom	Charisma	Passive per.

Skills (WIP)

Search

- +2 acrobatics
- +3 animalHandling
- +5 arcana
- +1 athletics
- +4 deception
- +5 history
- +3 insight
- +4 intimidation
- +3 investigation
- +3 medicine
- +3 nature
- +3 perception
- +4 performance
- +4 persuasion
- +3 religion
- +2 sleightOfHand
- +2 stealth
- +3 survival

Saving throws (WIP)

- +6 charisma
- +2 constitution
- +2 dexterity
- +3 intelligence
- +1 strength
- +5 wisdom

Dang it I forgot my char sheet!

Adran

Level 4

Paladin	**Lawful Neutral**	**5950**
CLASS	ALIGNMENT	EXP. POINTS
Sage	**Half-elf**	
BACKGROUND	RACE	SUBRACE

30
Speed

2 **Proficiency bonus** ○ **Inspiration**

Strength	Dexterity	Constitution	Intelligence	Wisdom	Charisma	Passive Per.
13 (+1)	14 (+2)	14 (+2)	16 (+3)	16 (+3)	18 (+4)	16

Skills (WIP)

Search

○ +2 acrobatics	○ +3 insight	○ +4 performance
○ +3 animalHandling	○ +4 intimidation	○ +4 persuasion
● +5 arcana	○ +3 investigation	○ +3 religion
○ +1 athletics	○ +3 medicine	○ +2 sleightOfHand
○ +4 deception	○ +3 nature	○ +2 stealth
● +5 history	○ +3 perception	○ +3 survival

Saving throws (WIP)

- ● +6 charisma
- ○ +2 constitution
- ○ +2 dexterity
- ○ +3 intelligence
- ○ +1 strength
- ● +5 wisdom

THE STRUCTURE

mobx-collection-store

CHARACTER MODEL

Class

Alignment

Race

Skills

etc.

```
static defaults = {
  __subraceId: '',
  alignment: '',
  background: '',
  class: '',
  experience: 0,
  inspiration: false,
  name: '',
  playerName: '',
  race: '',
  stats: {
    charisma: 0,
    constitution: 0,
    dexterity: 0,
    intelligence: 0,
    strength: 0,
    wisdom: 0,
  }
};
```

```
@computed get modifiers() {
  return mapValues(this.stats, (value) => Math.floor((value - 10) / 2));
}

@computed get level() {
  const levels = this.__collection.level.filter((level) => level.exp <= this.experience);
  return last(levels); // lodash.last - get the last element in an array
}
```

WILD REACT APPEARS

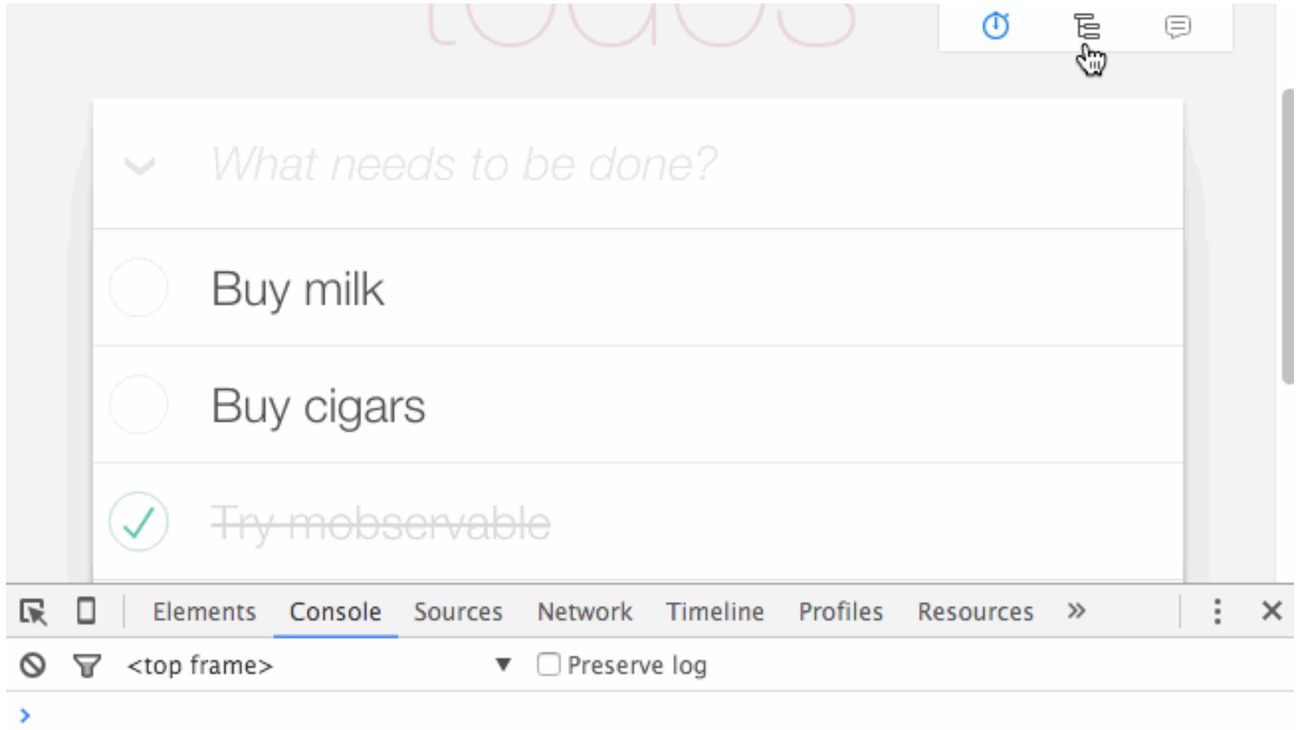
```
import {observer} from 'mobx-react';
import React from 'react';
import {map} from 'utils/helpers';

export const SavingThrows = observer({player}) => (
  <div>
    <h4>Saving throws (WIP)</h4>
    <div>
      {
        map(player.savingThrows, (value, key) => (
          <div key={key}>
            <input
              type='checkbox'
              checked={player.savingThrowProficiencies.includes(key)}
              disabled />
            <span>{value}</span>
            <span>{key}</span>
          </div>
        ))
      }
    </div>
  </div>
);
```

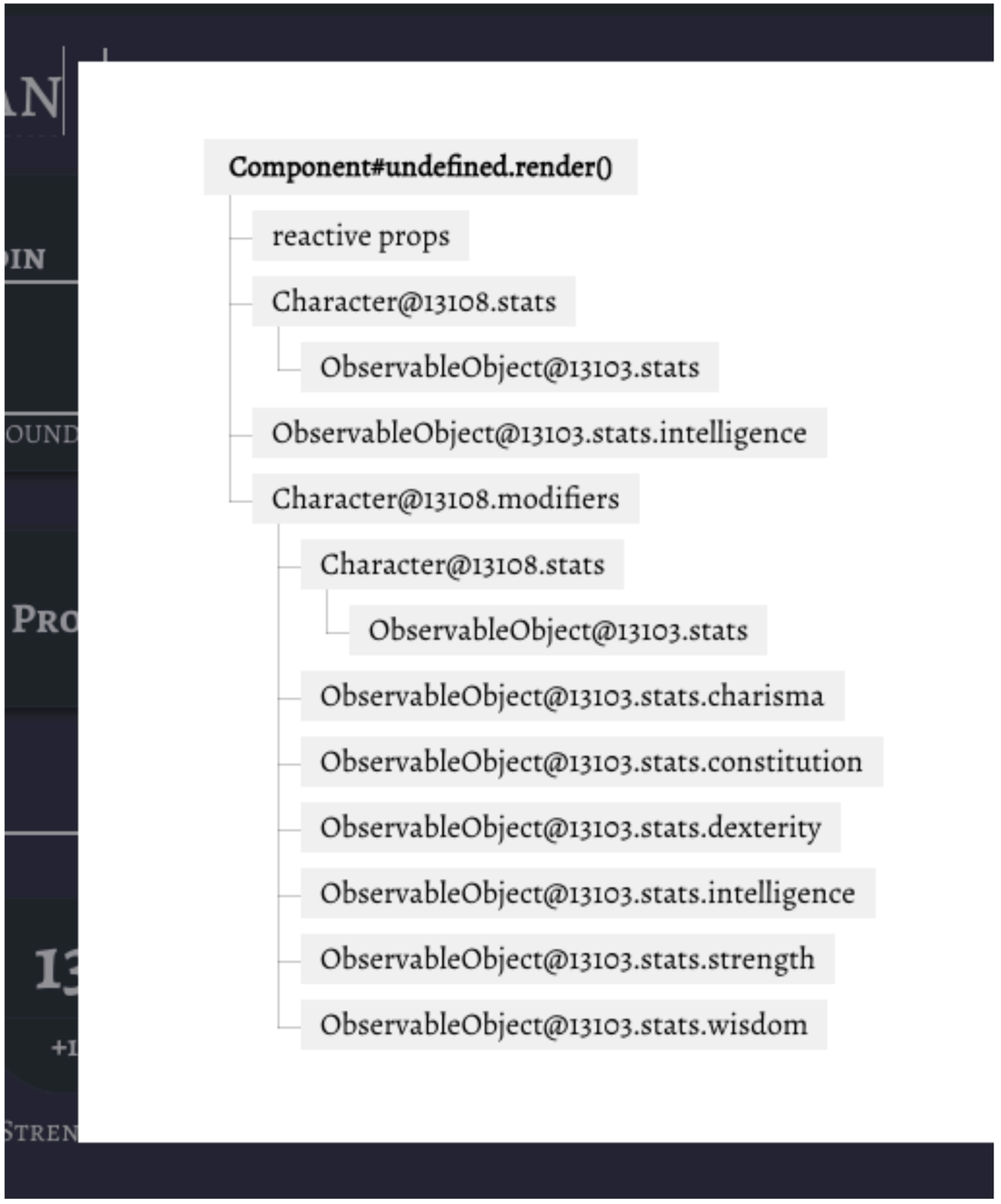
```
import {observer} from 'mobx-react';
import React from 'react';
import {map} from 'utils/helpers';

export const SavingThrows = observer({player}) => (
  <div>
    <h4>Saving throws (WIP)</h4>
    <div>
      {
        map(player.savingThrows, (value, key) => (
          <div key={key}>
            <input
              type='checkbox'
              checked={player.savingThrowProficiencies.includes(key)}
              disabled />
            <span>{value}</span>
            <span>{key}</span>
          </div>
        ))
      }
    </div>
  </div>
);
```

MOBX HELPER LIBS

MOBX-REACT-DEVTOOLS

- Log actions & reactions
- Check dependencies for a react component
- Show update times




Component#undefined.render()
reactive props
Character@13108.stats
ObservableObject@13103.stats
ObservableObject@13103.stats.intelligence
Character@13108.modifiers
Character@13108.stats
ObservableObject@13103.stats
ObservableObject@13103.stats.charisma
ObservableObject@13103.stats.constitution
ObservableObject@13103.stats.dexterity
ObservableObject@13103.stats.intelligence
ObservableObject@13103.stats.strength
ObservableObject@13103.stats.wisdom

MOBX-COLLECTION-STORE

- https://github.com/infinum/mobx-collection-store

- One collection, multiple model types
- Relationships between models

- mobx-jsonapi-store
- https://github.com/infinum/mobx-jsonapi-store

MOBX-STATE-TREE

- https://github.com/mobxjs/mobx-state-tree
- Made by MobX authors
- Opinionated
- Work in progress (not ready for use)
- Supports snapshots, replay, JSON patches, etc.

- **Supports time travel!**
- **Compatible with Redux DevTools!**

WHAT ABOUT REACT SETSTATE?

REACT SETSTATE

- Built in state management
- Component based
- Async!

REACT SETSTATE + MOBX

- You can still use it
- But, you can also do this...

```
import {observable} from 'mobx';
import {observer} from 'mobx-react';
import React, {Component} from 'react';

@observer
export default class Foo extends Component {
  @observable state = {
    clickCount = 0;
  };

  onClick: () => {
    this.state.clickCount++; // if you use setState it will stop being an observable!
  }

  render() {
    return (
      <button onClick={this.onClick}>
        {
          this.state.clickCount ? `Clicked ${this.state.clickCount} times!` : 'Click me!'
        }
      </button>
    );
  }
}
```

One more thing...

TYPESCRIPT

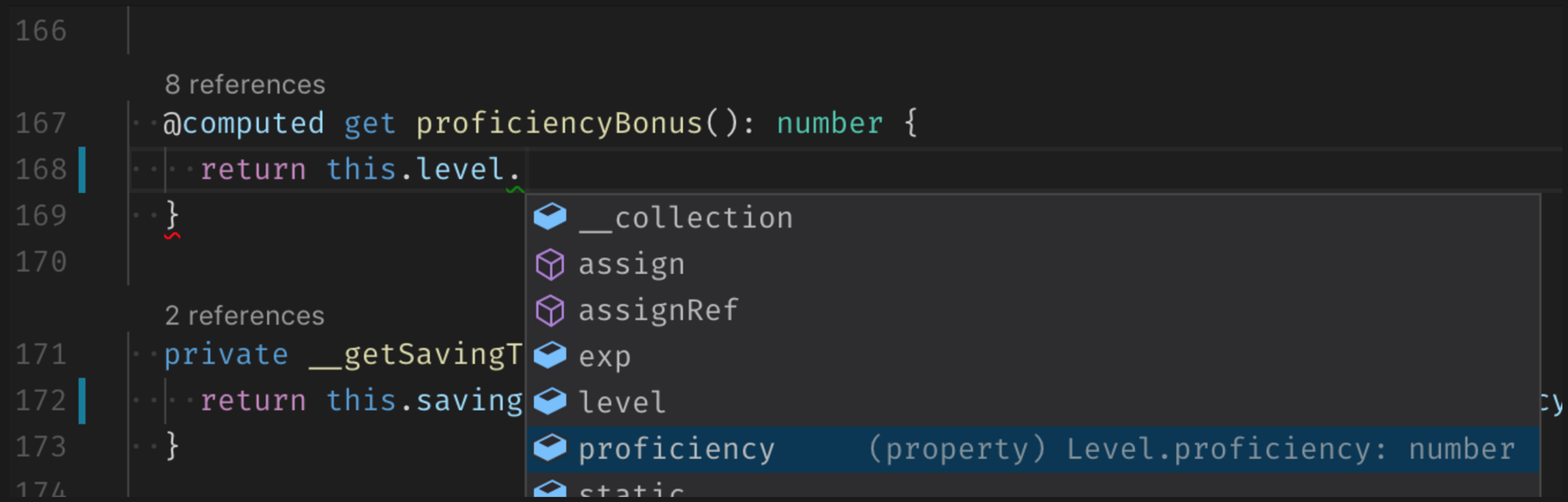
TYPESCRIPT

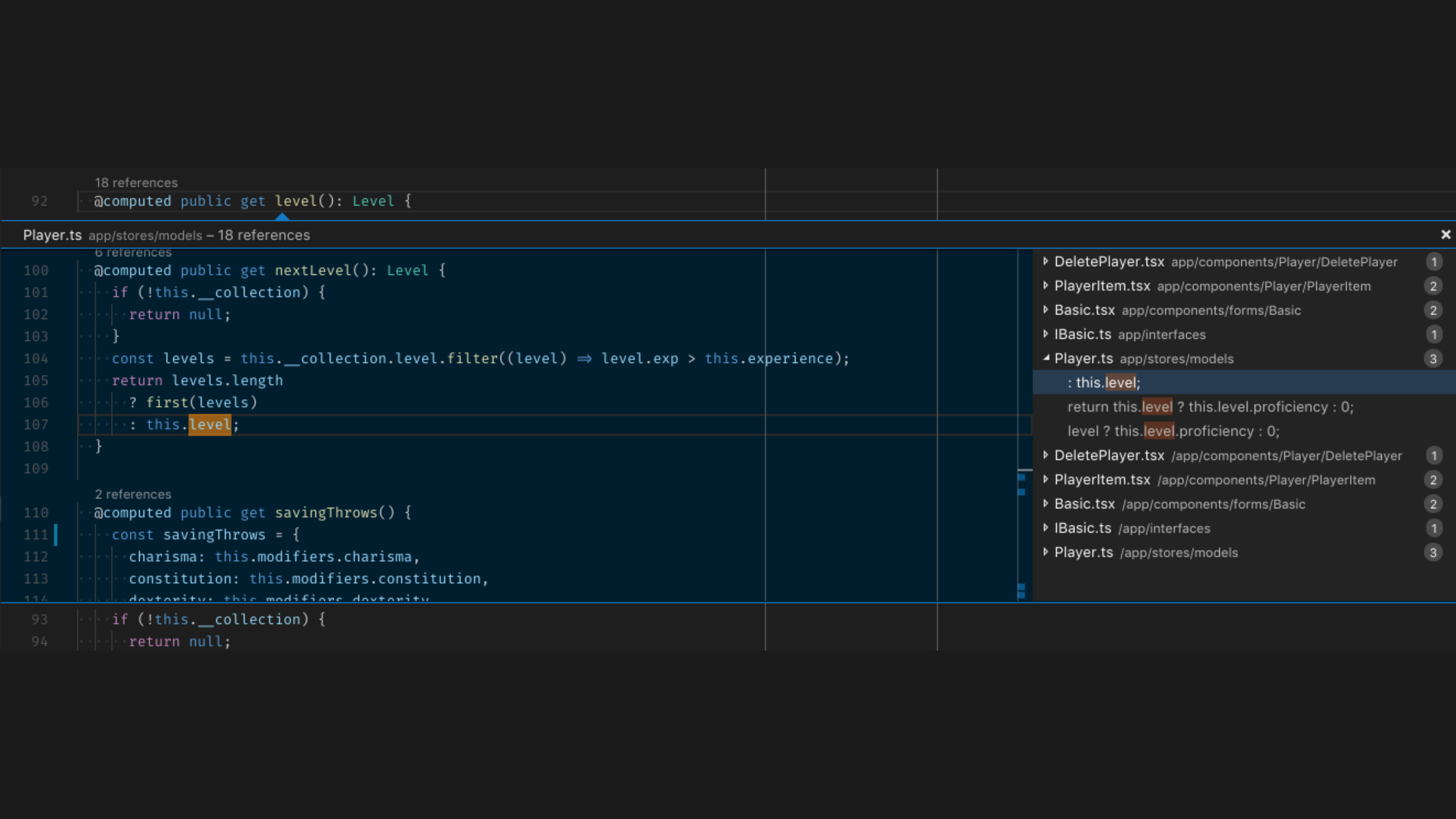
- A typed superset of JavaScript that compiles to plain JavaScript
- Basically, JavaScript with types
- Native support for decorators
- Much easier to refactor
- Much better editor support (especially with Visual Studio Code)

- Used by: Angular 2, MobX, Dungeons and Dragons :D

- More about it in the following months on JS Talks ;)

```
@computed public get level(): Level {
  const levels: Array<Level> = this.__collection.level.filter((level) => level.exp <= this.experience);
  return last(levels);
}
```




18 references
92 @computed public get level(): Level {
Player.ts app/stores/models – 18 references
6 references
100 @computed public get nextLevel(): Level {
101 if (!this.__collection) {
102 return null;
103 }
104 const levels = this.__collection.level.filter((level) ⇒ level.exp > this.experience);
105 return levels.length
106 ? first(levels)
107 : this.level;
108 }
109
2 references
110 @computed public get savingThrows() {
111 const savingThrows = {
112 charisma: this.modifiers.charisma,
113 constitution: this.modifiers.constitution,
DeletePlayer.tsx app/components/Player/DeletePlayer 1
PlayerItem.tsx app/components/Player/PlayerItem 2
Basic.tsx app/components/forms/Basic 2
IBasic.ts app/interfaces 1
Player.ts app/stores/models 3
: this.level;
return this.level ? this.level.proficiency : 0;
level ? this.level.proficiency : 0;
DeletePlayer.tsx /app/components/Player/DeletePlayer 1
PlayerItem.tsx /app/components/Player/PlayerItem 2
Basic.tsx /app/components/forms/Basic 2
IBasic.ts /app/interfaces 1
Player.ts /app/stores/models 3
93 if (!this.__collection) {
94 return null;

FOR THE END...

- https://mobx.js.org/

- https://github.com/infinum/dungeons-and-dragons
- https://dnd.byinfinum.co/

INFINUM

Thank you!

DARKO@INFINUM.CO
@DARKOKUKOVEC

Visit infinum.co or find us on social networks:

infinum.co infinumco infinumco infinum

INFINUM

Any questions?

DARKO@INFINUM.CO
@DARKOKUKOVEC

Visit infinum.co or find us on social networks:

infinum.co infinumco infinumco infinum