Web Components

Tomek Sułkowski

Technicalities

workshop plan

- Your trainer
  - Tomek Sułkowski
  - IT Trainer
  - FrontEnd Engineer @ StackBlitz

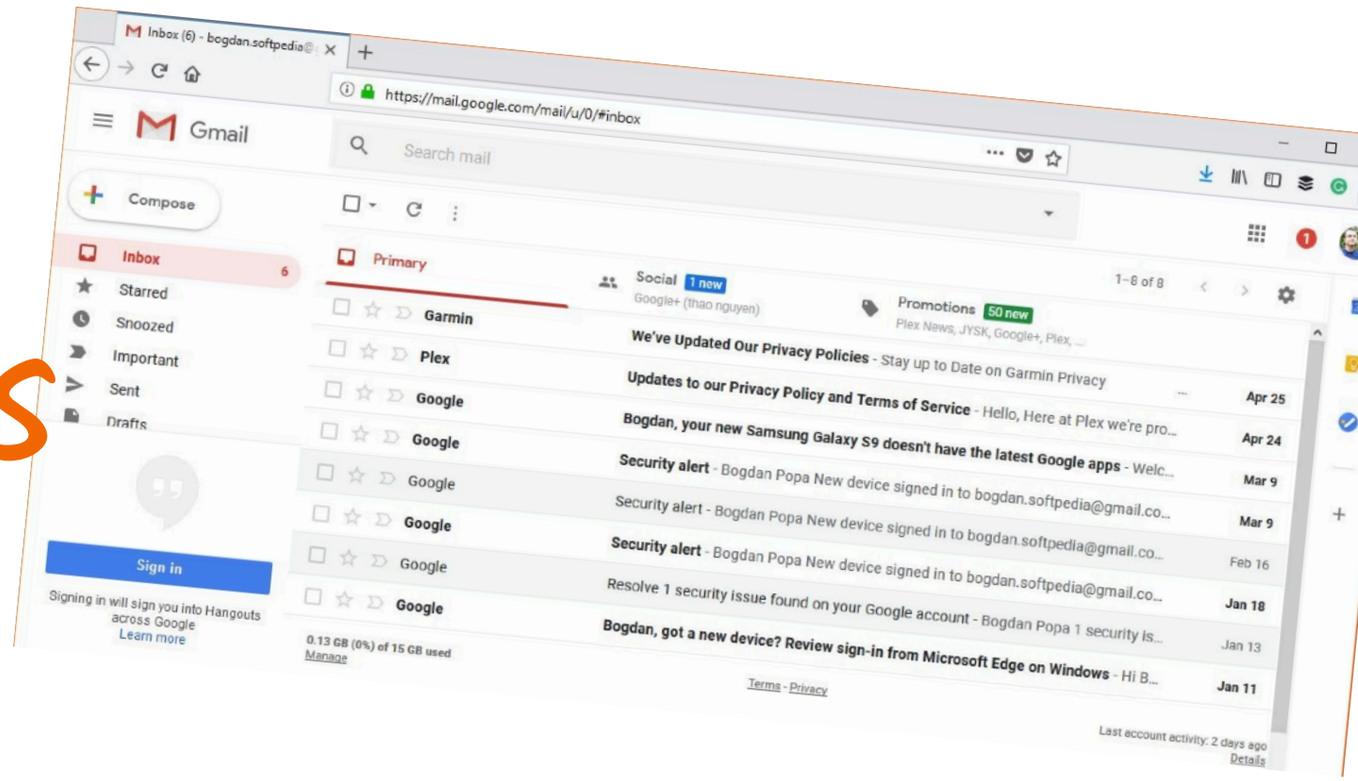
- Training objectives & plan
- Questions, discussions
- Issues flexibility

Tooling

Why Components?

Why components?

VS



What's a Web Component?

"A suite of different technologies allowing you to create reusable custom elements — with their functionality encapsulated away from the rest of your code — and utilize them in your web apps."

Limitations?

- no binding
- no change detection
- no state management
- no routing
- no forms support
- …

Limitations?

Custom Elements (V1) - LS

Method of defining new HTML tags.

Usage: Global — % of all users: 72.31% + 19.13% = 91.44%

Current aligned | Usage relative | Date relative | Apply filters | Show all

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baic Brow
		2-49			10-40								4		
		2 1 50-58	4-53										1 5.4		
	12-18	3 1 59-62	1 54-66	3.1-10	1 41-63	3.2-10.2									
6-10	79-80	63-75	67-80	1 10.1-13	64-67	1 10.3-13.3		2.1-4.							
11	81	76	81	1 13.1	68	1 13.4	all	81							
		77-78	83-85	1 TP											

HTML templates - LS

Method of declaring a portion of reusable markup that is parsed but not rendered until cloned.

Usage: Global

Current aligned | Usage relative | Date relative | Apply filters | Show all

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome for Android	Firefox for Android	UC Brows for Andro
				3.1-6								
				3 6.1								
	12		4-25	7	10-12.1	3.2-7.1						
	1 2 13-14	2-21	1 26-34	1 2 7.1-8	1 15-21	1 2 8-8.4		2.1-4.3				
6-10	15-80	22-75	35-80	9-13	22-67	9-13.3		4.4-4.4.4	12-12.1			
11	81	76	81	13.1	68	13.4	all	81	46	81	68	12.12
		77-78	83-85	TP								

Shadow DOM (V1) - WD

Method of establishing and maintaining functional boundaries between DOM trees and how these trees interact with each other within a document, thus enabling better functional encapsulation within the DOM & CSS.

Current aligned | Usage relative | Date relative | Apply filters | Show all

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baic Brow
		2-57													
		2 58											4		
	12-18	3 59-62	4-52	3.1-9.1	10-39	3.2-9.3							5.4		
6-10	79-80	63-75	53-80	10-13	40-67	1 10-13.3		2.1-4.4.4	12-12.1				6.2-10.1		
11	81	76	81	13.1	68	1 13.4	all	81	46	81	68	12.12	11.1	1.2	7.1
		77-78	83-85	TP											

Benefits?

Browser support	CHROME	OPERA	SAFARI	FIREFOX	EDGE
HTML TEMPLATES	STABLE	STABLE	STABLE	STABLE	STABLE
CUSTOM ELEMENTS	STABLE	STABLE	STABLE	STABLE	STABLE
SHADOW DOM	STABLE	STABLE	STABLE	STABLE	STABLE
ES MODULES	STABLE	STABLE	STABLE	STABLE	STABLE

Benefits?

- Lightweight
- Flexible
- Framework independent

What is a Web Component?

Web Components' building blocks

- ES Modules
- HTML Templates
- Custom Elements
- Shadow DOM

~~ECMAScript 6~~

ECMAScript 2015

ECMAScript 2015

- JavaScript standard following ECMAScript 5 version
- The biggest single development of the JS language yet
- Since 2015 new versions comes in yearly release schedule
- ECMAScript 2016 is a much smaller change
- 5 stages for each proposal development

  Strawman (0) –> Proposal (1) –> Draft (2) –> Candidate (3) –> Finished (4)

ES2015 - arrow functions

```
// Implicit returns
var odds = myArr.map(v => v + 1);
var nums = myArr.map((v, i) => v + i);
var pairs = myArr.map(v => ({even: v, odd: v + 1}));

// Declarations stay in brackets
nums.forEach(v => {
  if (v % 5 === 0) fives.push(v);
});

// Lexical this
var bob = {
  _name: "Bob",
  _friends: [],
  printFriends() {
    this._friends.forEach(f =>
      console.log(this._name + " knows " + f));
  }
}
```

ES2015 - classes

```
class Student extends Person {
  constructor(name, age, major) {
    super(name, age);
    this.major = major;
  }

  greet(greeting) {
    console.log(greeting, this.name);
  }

  get year() {
    return (new Date()).getFullYear() - this.age;
  }

  set age(age) {
    this.age = parseInt(age);
  }

  static defaultAge() {
    return 18;
  }
```

ES2015 - template literals

```
tag`Welcome to ${myApplicationName}, where
strings are quite powerful… finally `
```

ES2015 - modules

```
// lib/math.js
export function sum(x, y) {
  return x + y;
}
export var pi = 3.141593;
export default var name = "Math module";
```

```
// app.js
import * as math from "lib/math";
alert("2π = " + math.sum(math.pi, math.pi));
```

```
// otherApp.js
import anyName from "lib/math";
import {sum, pi} from "lib/math";
alert("2π = " + sum(pi, pi));
```

ES2015 - [resources]

Learn more about:

- maps & weakmaps
- sets & weaksets
- proxies
- reflection
- symbols
- …

https://ponyfoo.com/articles/es6

Web Components API

Web Components: create

```
class Hello extends HTMLElement {
    constructor() {
        super()
    }
}
```

Web Components: register

```
window.customElements.define(...)
```

Defines a new custom element.

Two types of custom elements are:

- Autonomous custom element: Standalone elements. Don't inherit from built-in HTML elements.

- Customized built-in element: Inherit from — and extend — built-in HTML elements.

Web Components: shadows

`Element.`**`attachShadow(`**`{mode:'open'}`**`)`**

Attaches a shadow DOM tree to the specified element and returns a reference to its ShadowRoot

`Element.`**`shadowRoot`**

Read-only property represents the shadow root hosted by the element

Web Components: shadows

`Node.`**`getRootNode(`**`{composed: Bool}?)`**`)`**

Returns the context object's root
#document or *shadowRoot*

`Element.`**`isConnected`**

true is the Node is **connected**,
false otherwise

Web Components: lifecycle

`connected``Callback`
`disconnected``Callback`

custom element is first connected (/disconnected) to the document's DOM

`attributeChanged``Callback`

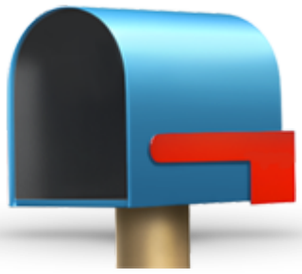
when one of the custom element's attributes is

- added,
- removed,
- changed

Template element

```
<template id="tmpl">
    <style>div {color: green}</style>
    <div>Hello template</div>
</template>
```

Holds HTML that is not to be rendered immediately when a page is loaded but may be instantiated using JavaScript



Slot element

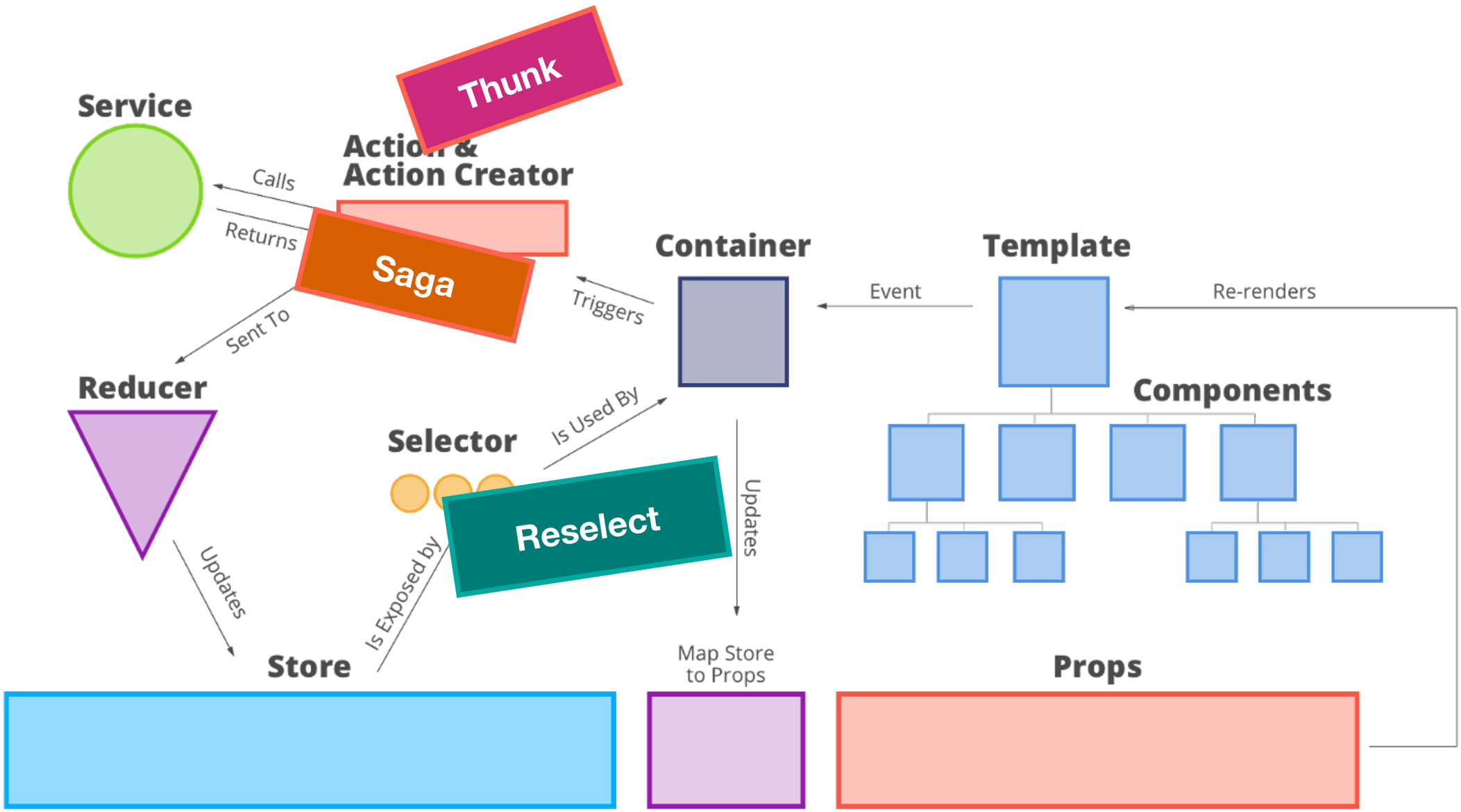
```
<slot name="something">
```

A placeholder inside a web component that you can fill with your own markup:

```
<my-component>
  <span slot="something">Hello!</span>
</my-component>
```

State Management

Redux?

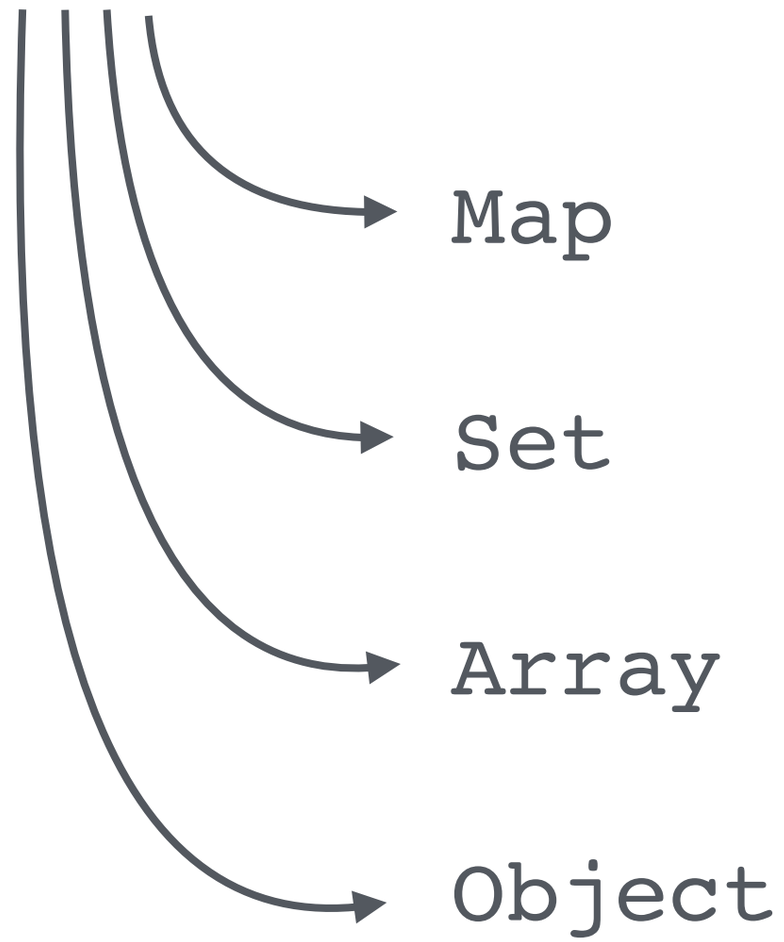

Service
Thunk
Action &
Action Creator
Calls
Returns
Saga
Container
Template
Event
Re-renders
Triggers
Components
Reducer
Selector
Is Used By
Reselect
Updates
Updates
Is Exposed by
Store
Map Store
to Props
Props

MobX

`observable` `computed` `action` `autorun`

MobX: observable

```
observable(value)
```



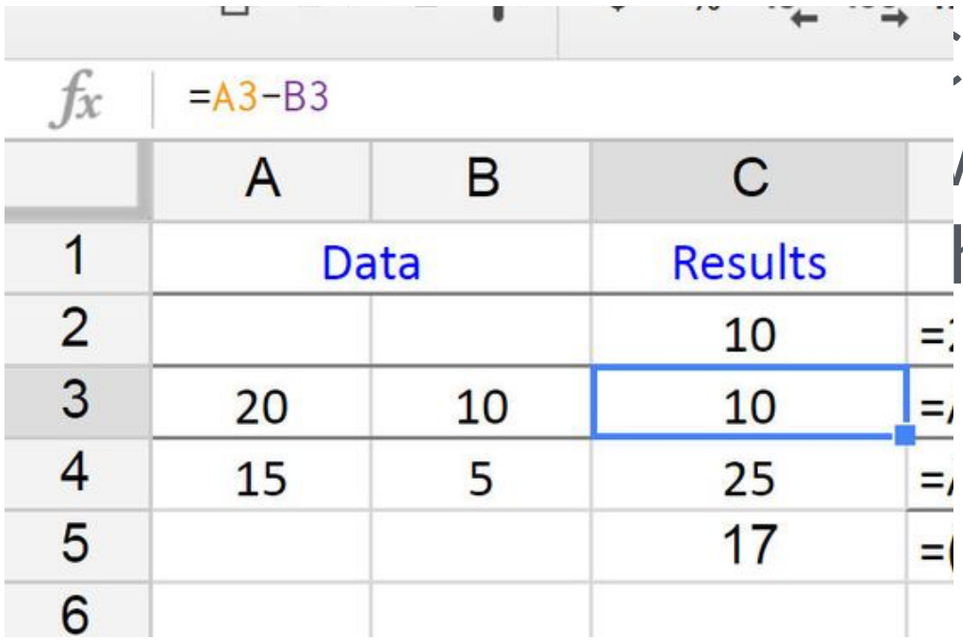
MobX: autorun

```
autorun(() => console.log(value))
```

Run it whenever observable value changes

MobX: autorun

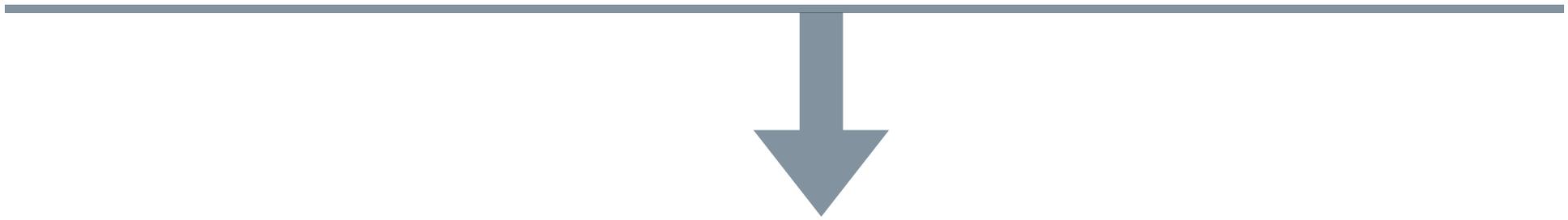
```
const currentFib = computed(() => fib(value))
```



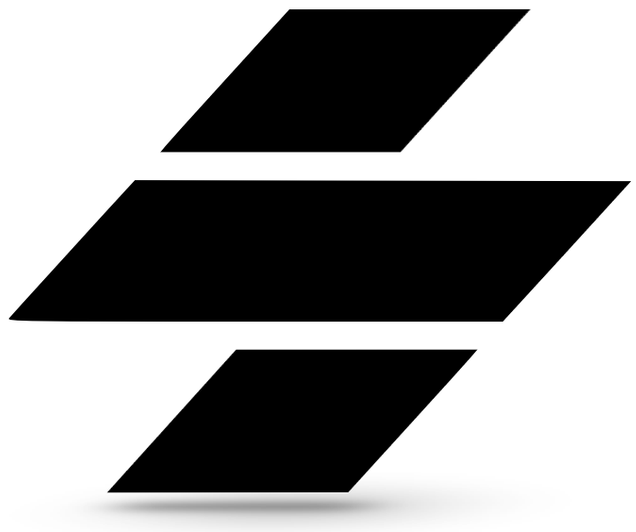
Creates a new observable that changes whenever the used observable value changes

MobX: autorun

```
action(() ⇒ value.myStatus++)
```



A piece of code that changes the state
(*implicitly* or *explicitly)*



Stencil

Stencil component

```
@Component({
   tag: 'my-color'
})
export class MyColorComponent {
   render() {
      return (
         <p>My favorite color is {this.color}</p>
      );
   }
}
```

Stencil component's data

```
export class MyColorComponent {

  @Prop() color: string = 'blue';

  @State() open = false;

}
```

external

internal

Stencil lifecycle methods

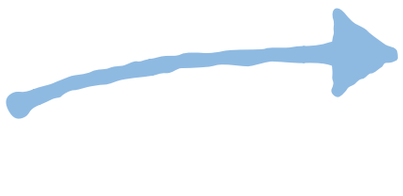
```
connectedCallback()
disconnectedCallback()

componentWillLoad()
componentDidLoad()

componentWillRender()
componentDidRender()

componentShouldUpdate()
componentWillUpdate()
componentDidUpdate()
```

Stencil events

from outside

```
@Listen('click')
handleClick(e) {
  console.log('What a click!', e);
}
```

to outside



```
@Event() complete: EventEmitter

/* ... */

this.complete.emit()
```