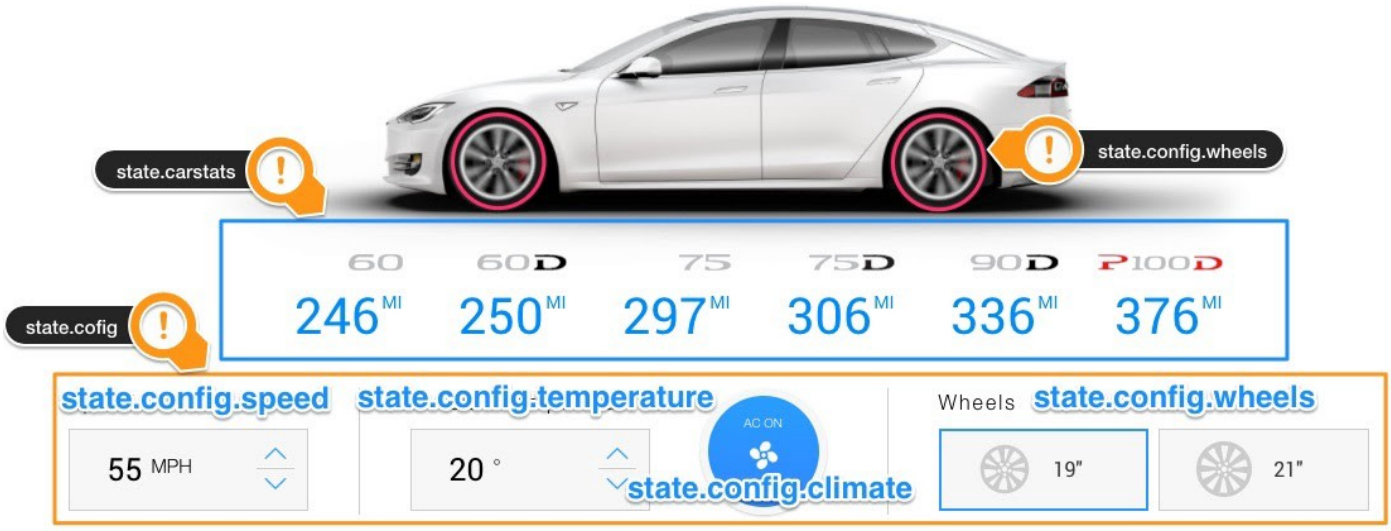
1.9 State of Application

We need to think about what `state` is required to be managed in our app. If you look at the final app GIF image at the top of this article, the state values are:

- **carstats (object array)**: An array of battery numerical value objects by car model according to the currently selected condition value (speed, temperature, climate, wheel)
- **config (object)**: Currently selected conditions object (speed: 55, temperature: 20, climate: aricon on, wheel: 19)



That is the single source of truth for our app. Now we will add the constructor method to the `TeslaBattery` container and set the initial value so that we can manage this state value and pass it to the subcomponent. The TeslaCarcomponent accepts the `wheelsize` input through `props` and renders the Tesla car image and spins the wheels.

> *Both the parent component and the child component do not know whether a particular component is stateful or stateless and do not care whether it is defined as a `function` or a `class`. This is why the state is often called local or encapsulated. This state can not be accessed by components other than the component that owns and sets the state. So this state value can be passed to the*

> *sub-component as `props`. This is commonly referred to as a "top-down" or "unidirectional" data flow. Every state is always owned by a particular*
>
> *component, and any data or UI derived from that state only affects the "downward" component of the tree.*

```
...
class TeslaBattery extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      carstats: [],
      config: {
        speed: 55,
        temperature: 20,
        climate: true,
        wheels: 19
      }
    }
  }

  render() {
    // ES6 Object destructuring Syntax,
    // takes out required values and create references to them
    const { config } = this.state;
    return (
      <form className="tesla-battery">
        <h1>Range Per Charge</h1>
        <TeslaCar wheelsize={config.wheels}/>
        <TeslaNotice />
      </form>
    )
  }
}
...
```

In `render()`, the code in the form `const {a, b} = c` is `ES6 Object Destructuring`. It takes the required value out of the object and makes a reference to it.

> *Conceptually, the React component is like a JavaScript function and receives an arbitrary input called '**props**' and returns a React element that describes what should be shown.*

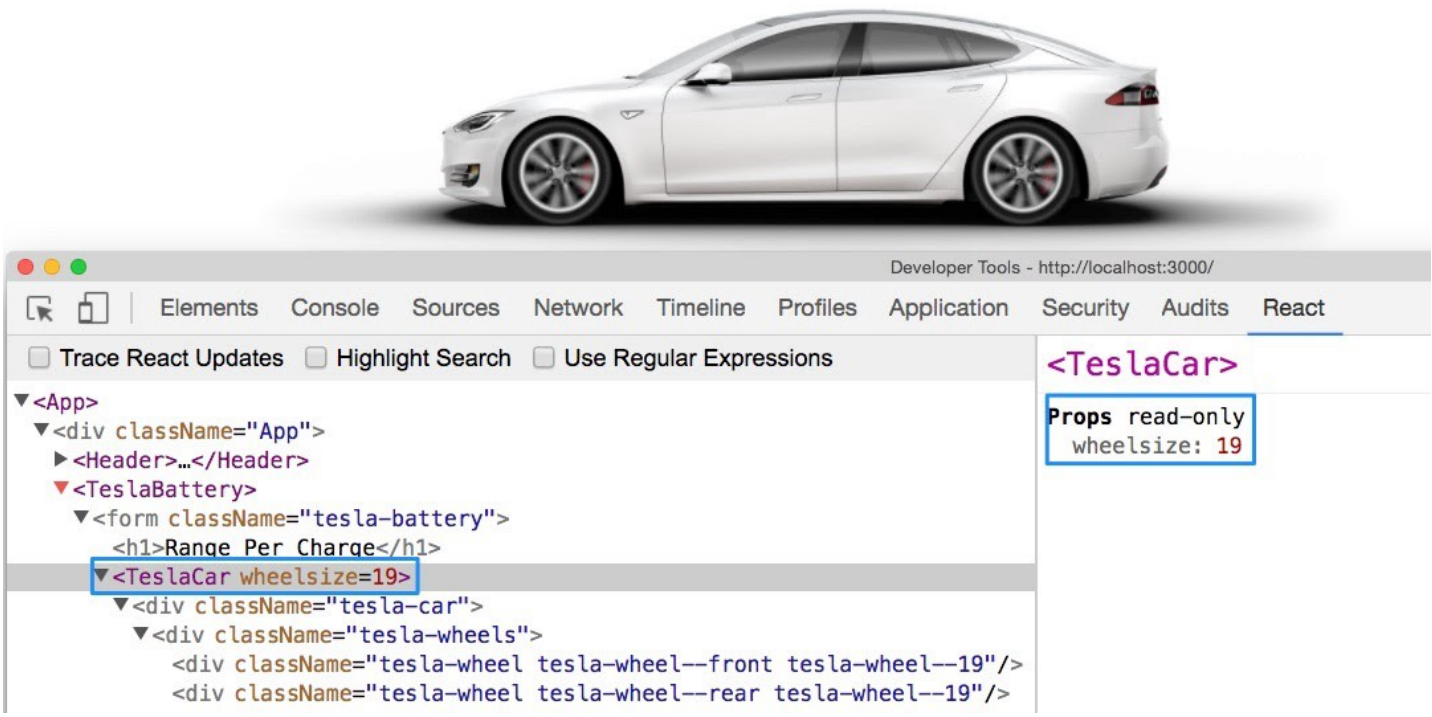
In a word, this concept can be expressed by the following formula.

> ***fn(d) = V***

A function that receives data as input and returns a view

A function that receives data as input and returns a view.

If you save files, you can see that the rendered Tesla car and wheel animation work well on the updated screen. You can also see that `props` is passed well in the component tree.



> *Some functions are called "pure" in the sense that they always return the same output value if they have the same input value without changing the input value. (`Pure function`) One important React strict rule here is that all React components should behave like pure functions with respect to props. `props` must be read-only.*

```
<?xml version="1.0" ?>
<svg width="1600px" height="208px" version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xm
<g>
        <path fill="#FFFFFF" d="M0,0c0.3,1.1,1.3,2.3,2.6,2.6h4.1l0.2,0.1V13h2.5V2.7l0.2-0.1h4.1c1.4
        <path fill="#FFFFFF" d="M77.8,13c1.3-0.5,2-1.5,2.2-2.6H68.7l0-10.5l-2.5,0v13H77.8z"/>
        <path fill="#FFFFFF" d="M47.3,2.6h9C57.6,2.2,58.8,1.2,59,0L44.8,0v7.7h11.6v2.7l-9.1,0c-1.4
                H47.3V2.6z"/>
        <polygon fill="#FFFFFF" points="85.4,5.2 85.4,13 88,13 88,7.8 97.1,7.8 97.1,13 99.7,13 99.
        <path fill="#FFFFFF" d="M25.2,2.6h9.7c1.3-0.3,2.4-1.5,2.6-2.6h-15C22.9,1.2,23.9,2.3,25.2,2
        <path fill="#FFFFFF" d="M25.2,7.8h9.7c1.3-0.3,2.4-1.5,2.6-2.6h-15C22.9,6.3,23.9,7.5,25.2,7
        <path fill="#FFFFFF" d="M25.2,13h9.7c1.3-0.3,2.4-1.5,2.6-2.6h-15C22.9,11.6,23.9,12.8,25.2,1
        <path fill="#FFFFFF" d="M87.7,2.6h9.7c1.3-0.3,2.4-1.5,2.6-2.6H85C85.3,1.2,86.3,2.4,87.7,2.(
</g>
</svg>
```