Internals of Vue

On Twitter, Github, and Instagram as

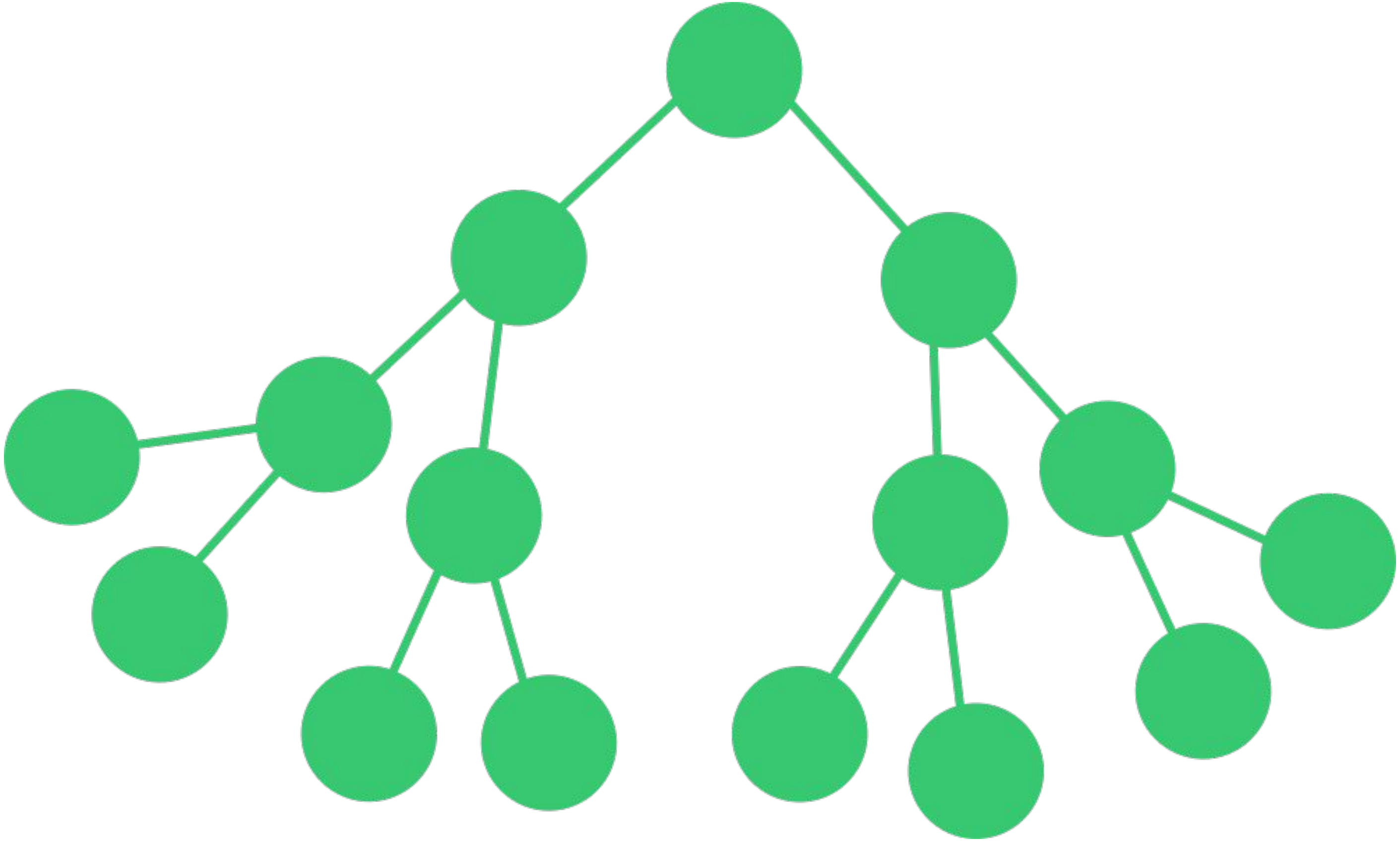
@nshntarora

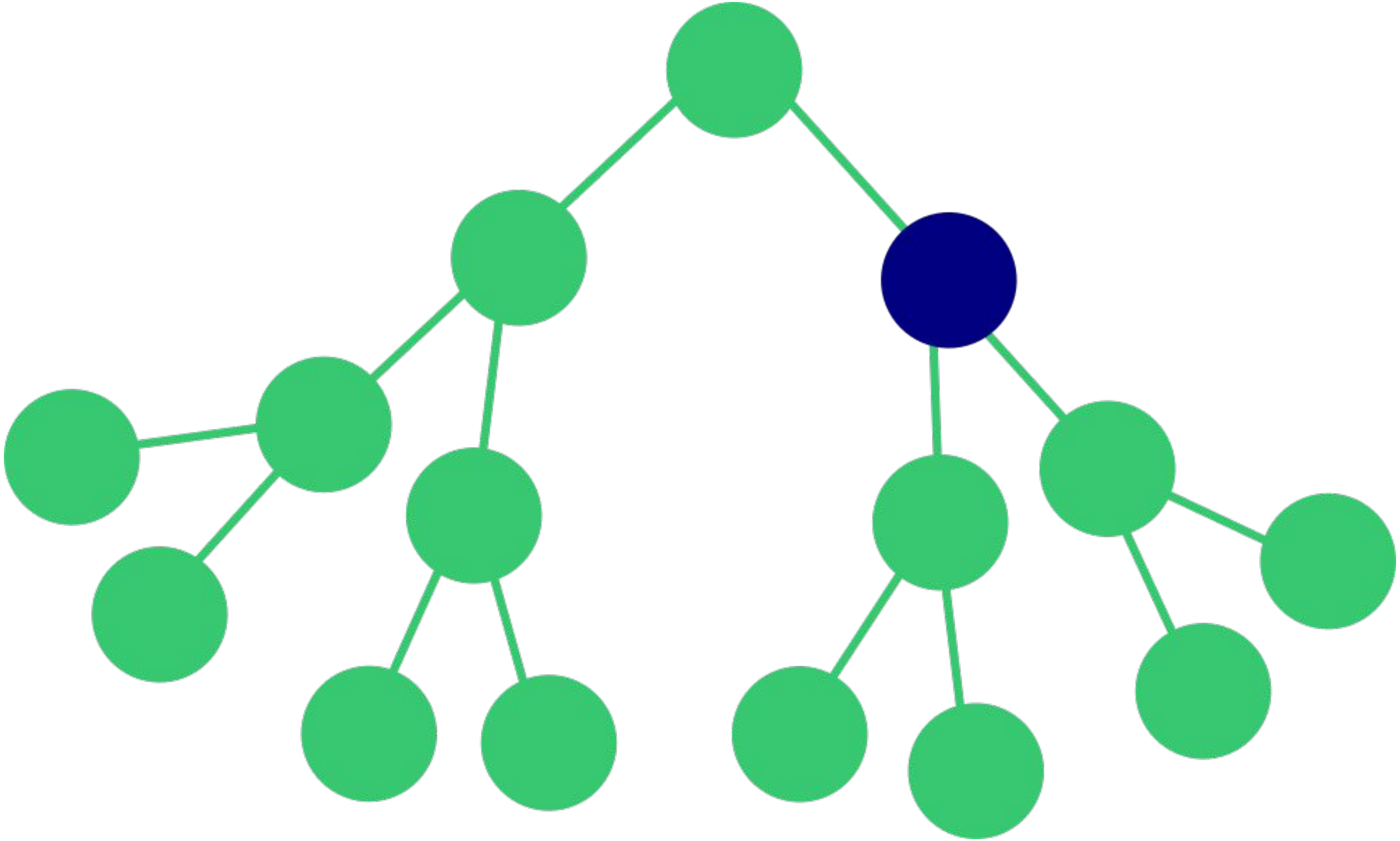
The obligatory introduction to Reactivity

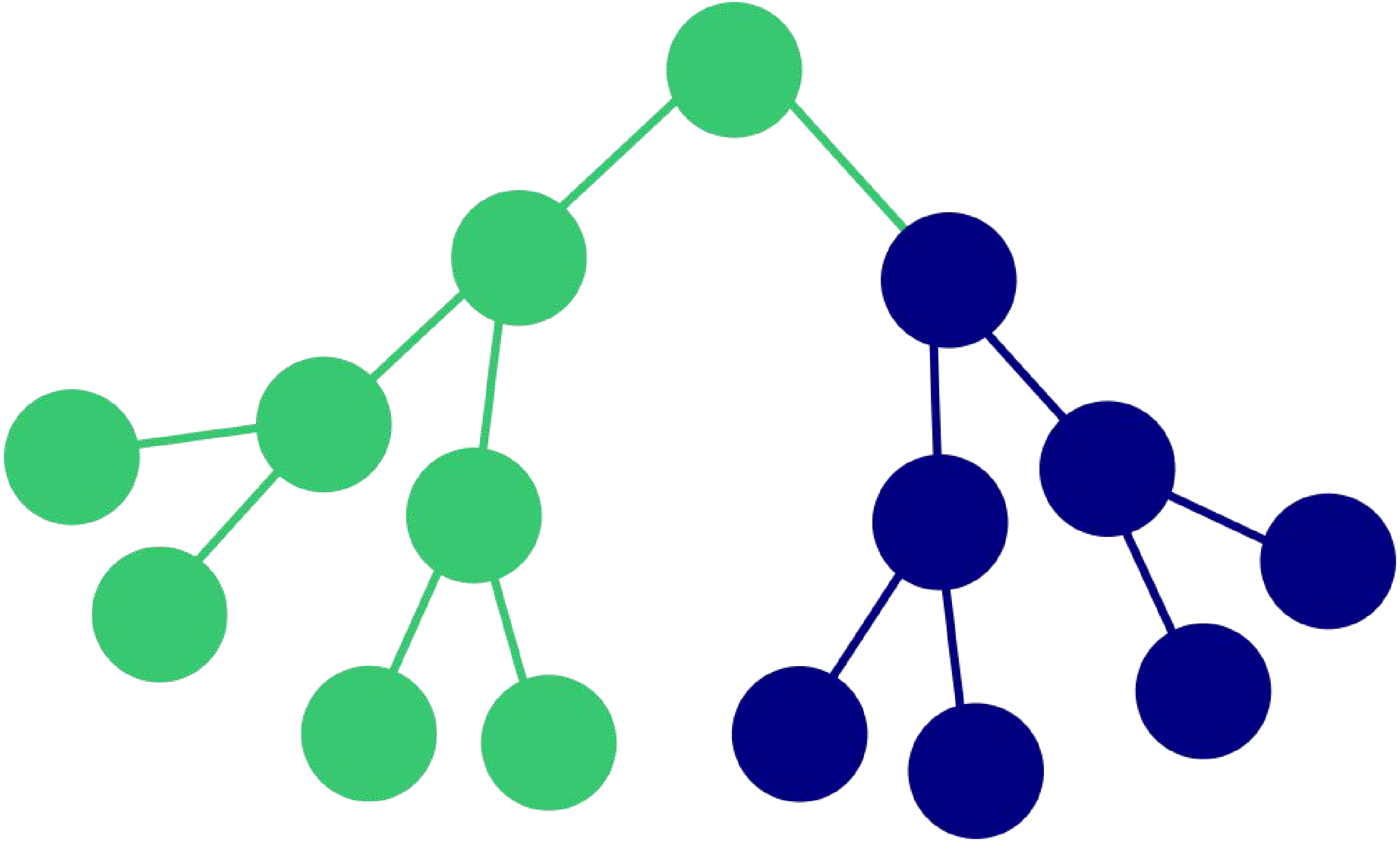
Pull-based Reactivity

Pull-based Reactivity

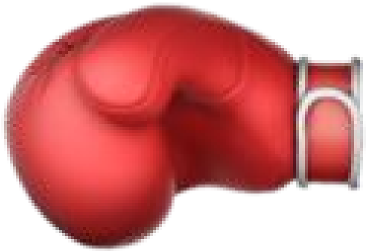
```
setState({ a: 1 })


// takes care of re-rendering. Probably calls render()
handleChanges()
```





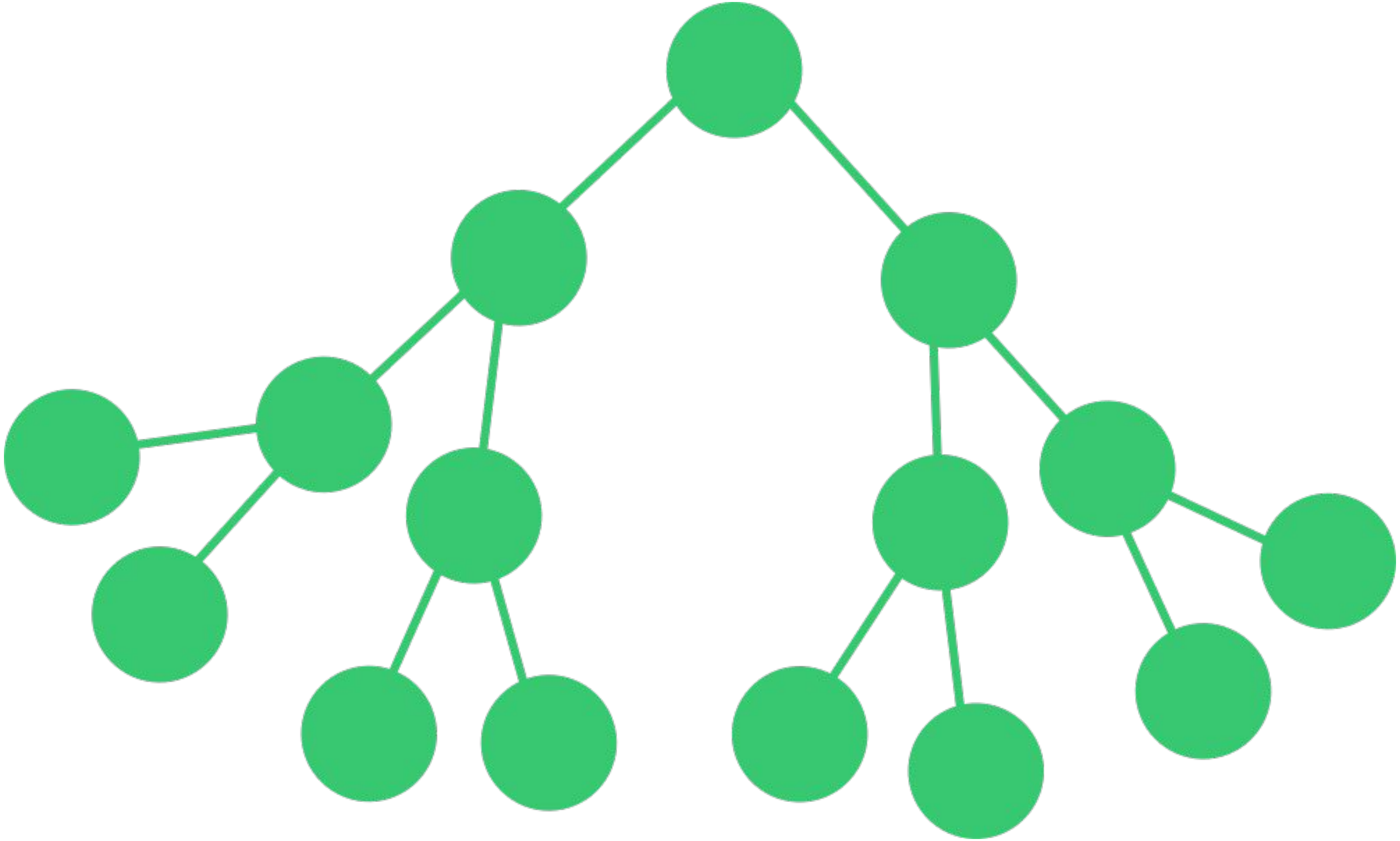


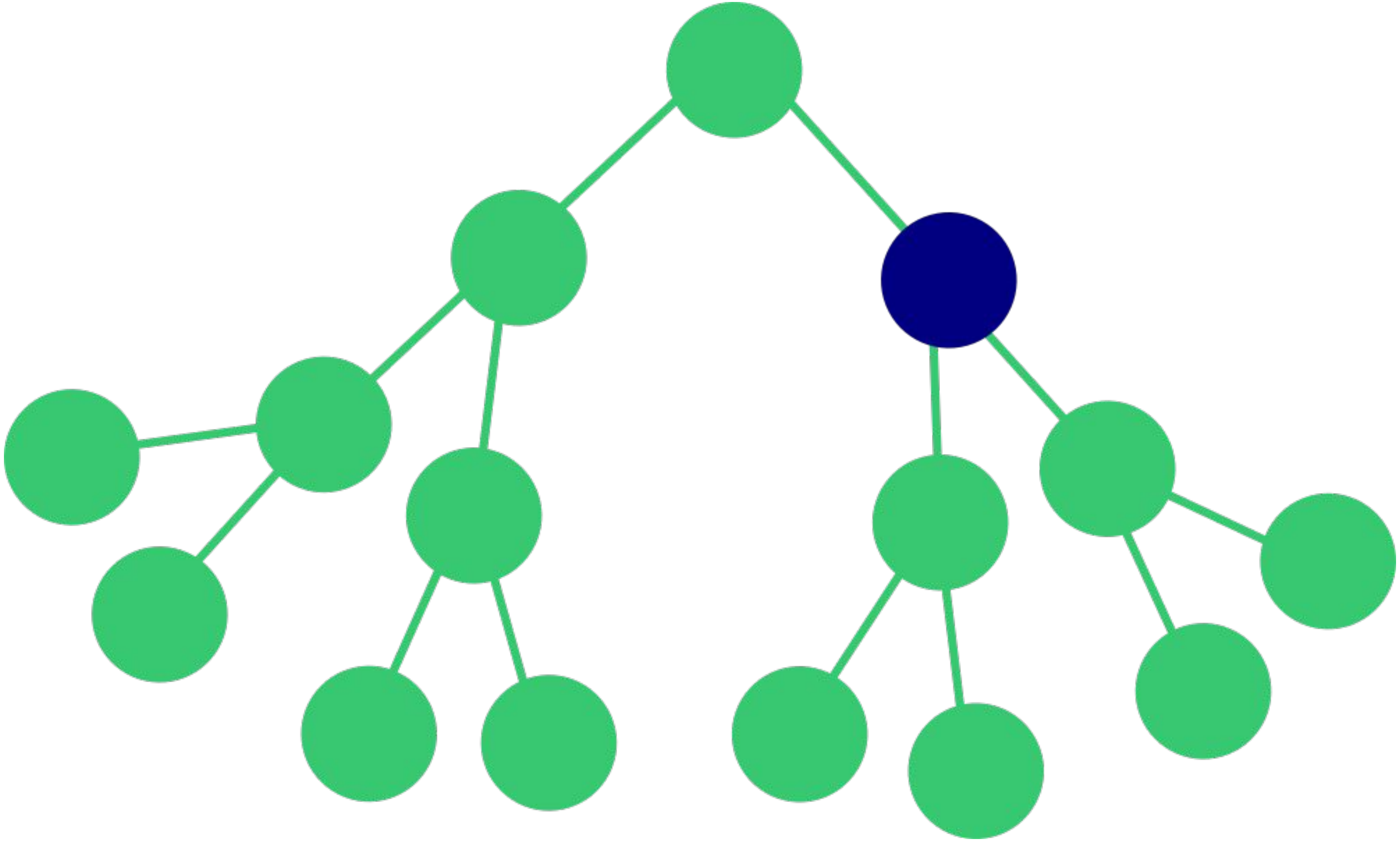
Push-based Reactivity

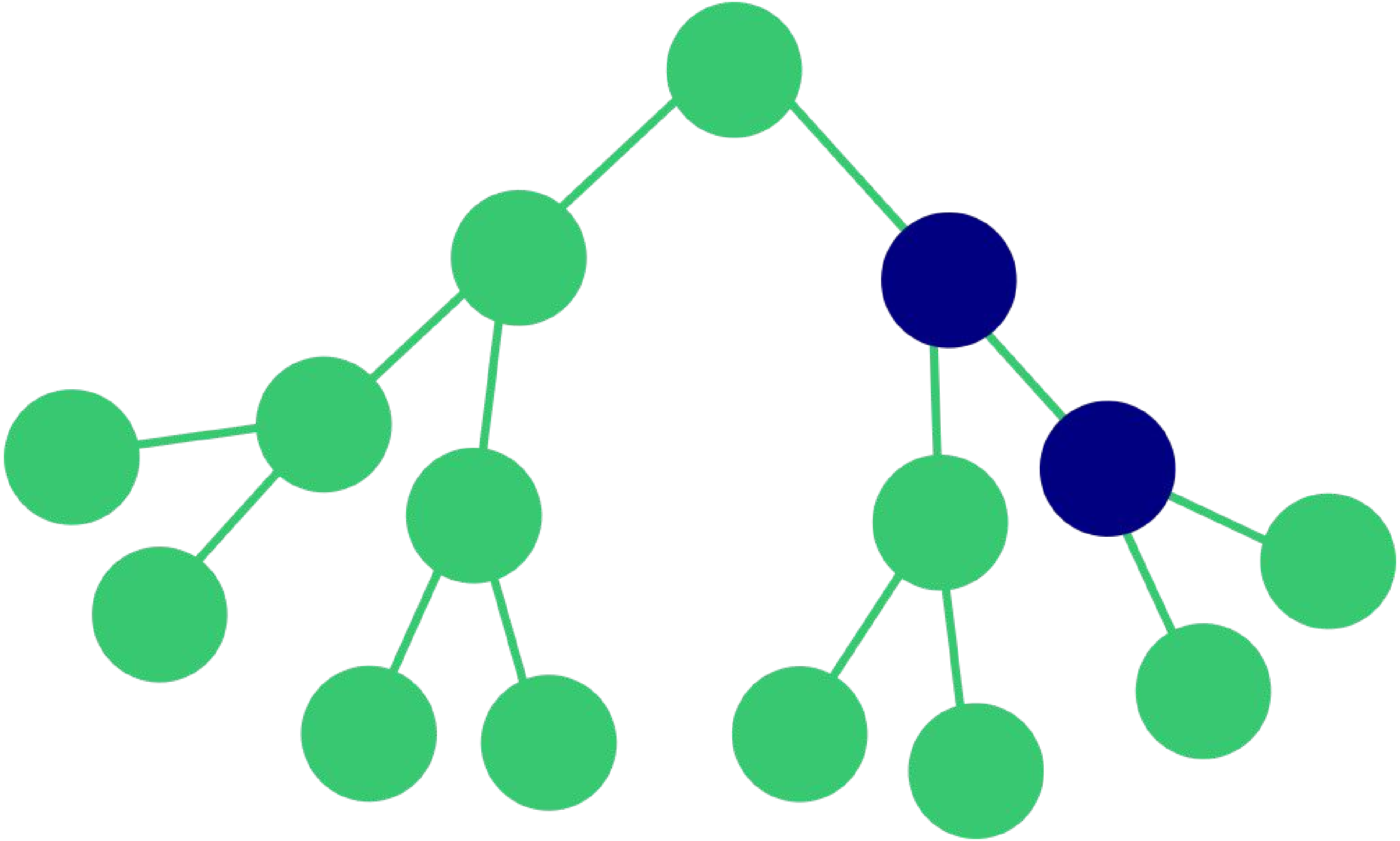


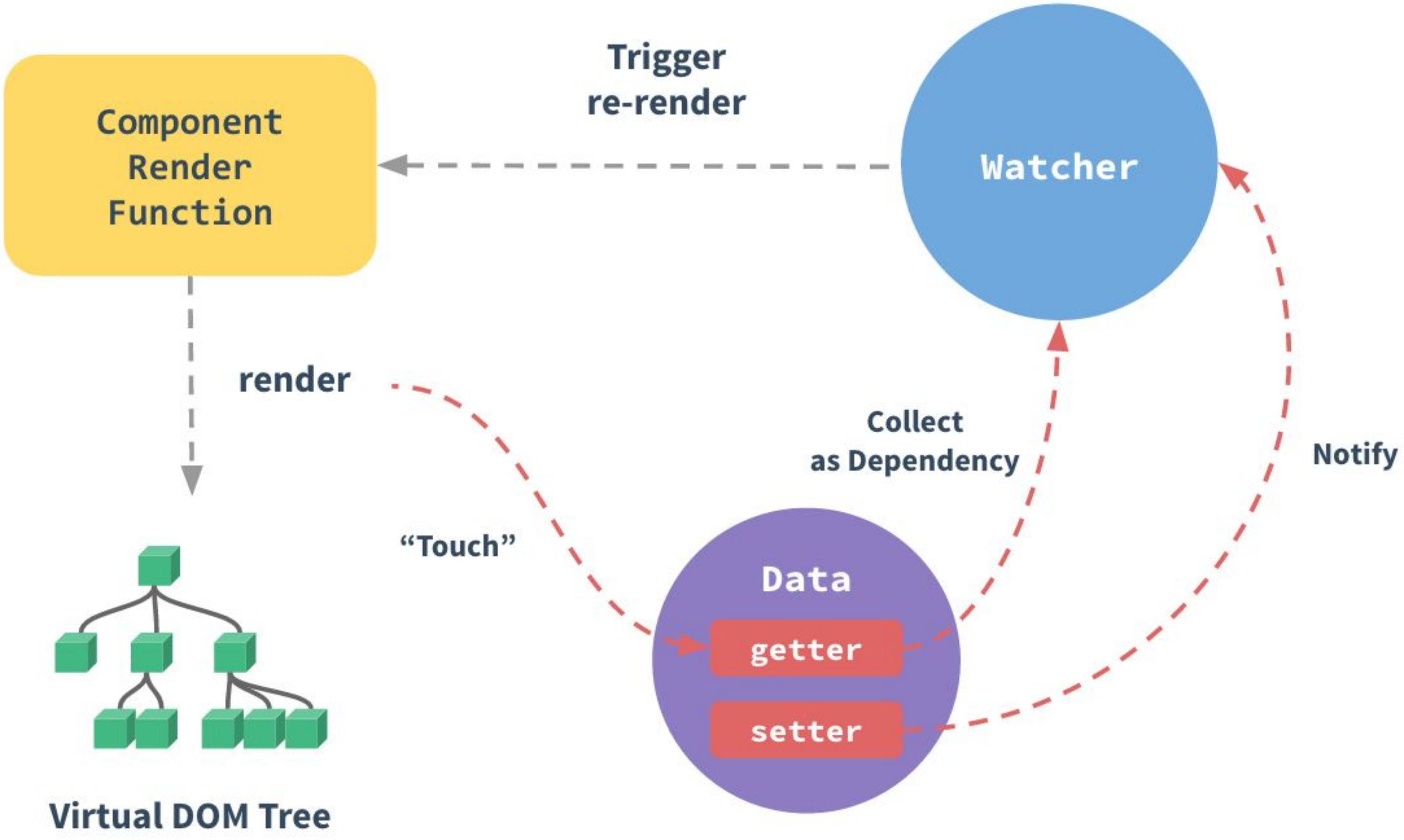
Push-based Reactivity

```
state.a = “Hi VueBLR”;

updateAllDependents();
```









Shamelessly downloaded from vuejs.org

Component State

Component State

1. What the component renders?
2. How it behaves?

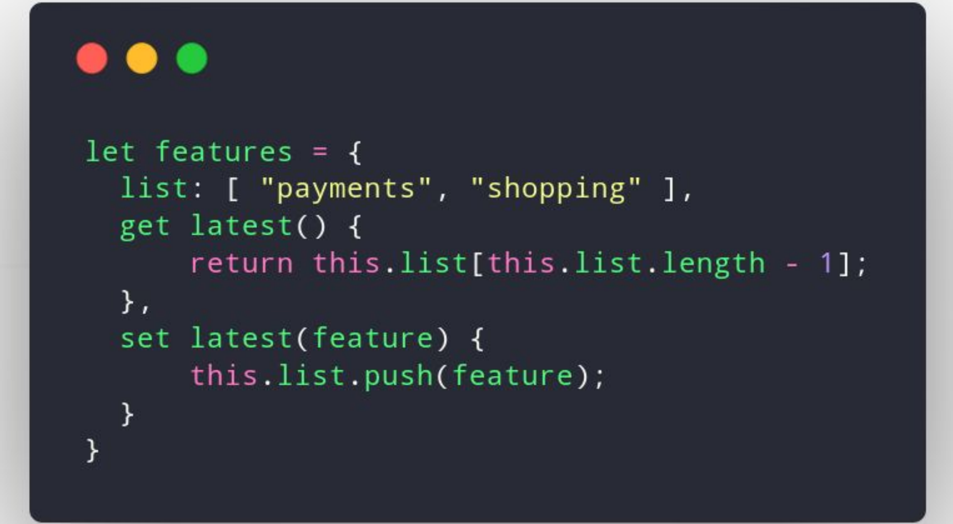
The `data` property

1. Component Level State.
2. Triggers a re-render when changed.

new **Vue**({…});

1. Looks up the data property.
2. Makes it reactive.

Getters and Setters



```
let features = {
  list: [ "payments", "shopping" ],
  get latest() {
      return this.list[this.list.length - 1];
  },
  set latest(feature) {
      this.list.push(feature);
  }
}
```

The watcher

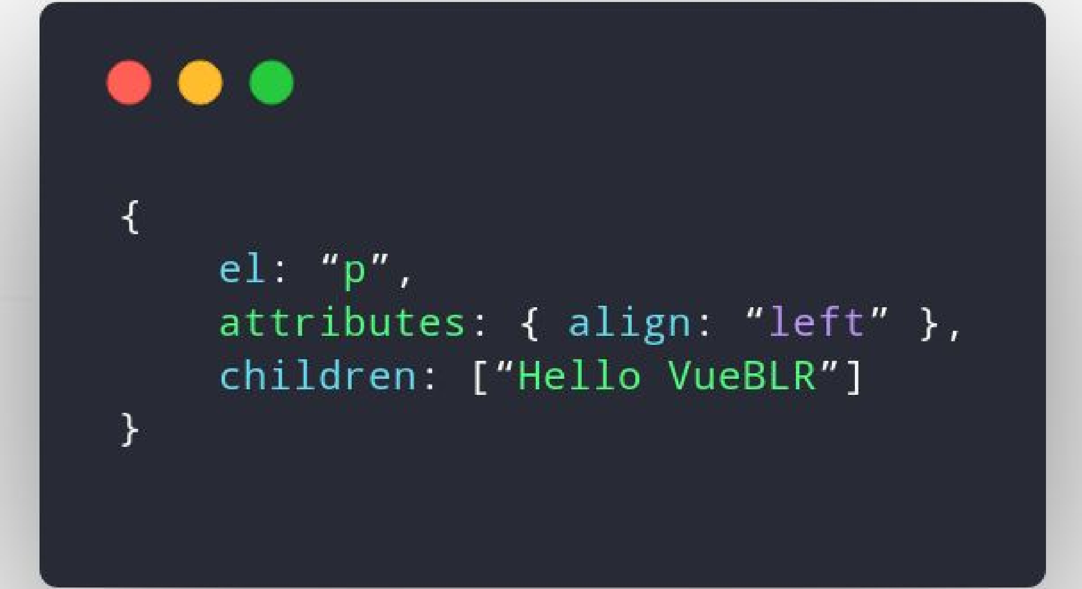
The watcher

1. Collects dependencies.
2. Fires callback when value changes.

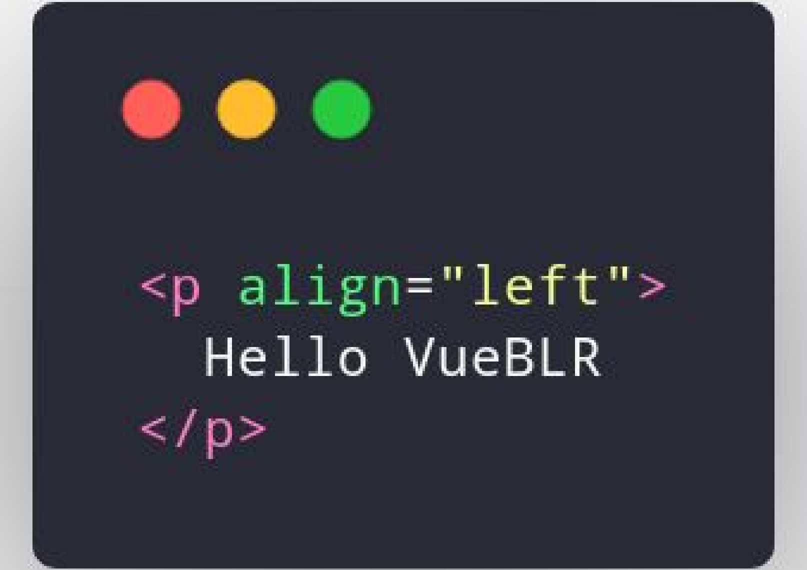
The Virtual DOM

The Virtual DOM

1. Representation of the DOM tree.
2. Inexpensive to update.



```
{
    el: “p”,
    attributes: { align: “left” },
    children: [“Hello VueBLR”]
}
```



```
<p align="left">
  Hello VueBLR
</p>
```