React Principles:

**Components:** separate UI into different components, each component has separated state



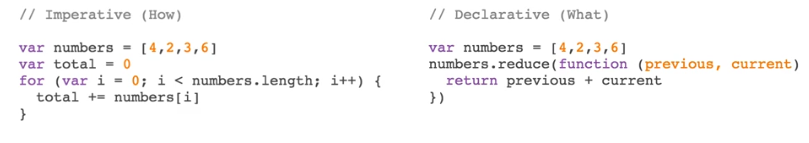
Note: Data in React either received from parent component or contain in component itself

if use any part of parent component data in the child component, pass that data to child as an attribute

HTML element in React JS file called JSX, and finally transformed into JS object🡪 React take these objects and form virtual DOM (in other words, virtual DOM is JS representation of actual DOM)

Process: signal to notify app some data change 🡪 Re-render virtual DOM 🡪 Diff prev virtual DOM with new virtual DOM 🡪 update real DOM with necessary changes.

**Declarative:** Imperative (specify to computer **how** thing happens, step by step) vs Declarative (tell **what** you want to actually happen, what you want it done)🡪 make simplified, reasonable statements). However, many (if not all) declarative approaches have some sort of underlying imperative abstraction.



Benefit of declarative:

* Reduce size effect
* Minimize mutability (change state or value of valuable because not sure its value at any point)
* More readable code
* Less Bugs.

React declarative most of the time:



**Unidirectinal Data Flow:** UI is the function of state, what we only care about in React application is to manage state changes, and UI will be updated accordingly.

**Explicit mutation:** whenever we want to update the state in our application, we have to explicitly call setState() to update the state 🡪 kick off re-render. So we not to make any listeners or do any dirty checking, instead we **explicitly** make the change

**Just Javascript:** small library based on javascript eco system, the better we are at React 🡪 better javascript programmer.

Support Tools

**NPM**: allow easily manage different packages (modules) and keep track which versions installed

**Webpack**??: code bundler. It takes codes, transforms and bundles it, then return brand new version 🡪 if we use any technology that require code conversion so that our code compliant with what browser uses (HTML,LESS SASS 🡪 CSS, Coffescript 🡪 Javascript), Webpack is able to carry out any transformation you need to make, and output bundle file full of those change (and minification)

What Webpack need to know ?

* Starting point of application, root JS file
* What transformation make on our code
* Which location to save new transformed code

**Babel.js**: tool for compiling Javascript 🡪 specific transformer for the code from JSX to JS or ES2015,16 to modern JS (that browser can understand)

React-dom : glue btw react and dom, decouple renderer from react itself (render to DOM or Android Native)