React

React are components that handle **state**, **props**, **context** and whenever these changes – components are **re-executed**/**re-evaluated**!

Re-Evaluating Components !== Re-Rendering the DOM

**Real DOM** is changed based on difference between previous state of a component and current state after state props or context change.

**ReactDOM** will not re render the entire DOM, it would only **insert** the difference between the html elements.

Closer look at Child Component Re-Evaluation

**React.memo()** re-render a component **only if** props are changed by **cost** of compare new props value and previous props value

**useCallback()** passes function to be comparable with previous initialization and not letting the component to be re-executed. If variables from outside are used inside -> they should be inserted into the array of dependencies! Otherwise the old (first loaded) value of this variable is used because of **closure storing**.

**useMemo()** memorizes result of **heavy sorting** came from props and to not do the sort again.

It is **recommended** to use the function form for updating state depend on previous state.

**Updating State**:

React takes **all the state updates** produced by function and **batch them together** into **one state update**, only if not using any callbacks or promises.