1. State mirip dengan props, yaitu sebuah parameter yang dapat digunakan oleh Component tersebut.
2. Namun, state memiliki perbedaan dengan props, yaitu:
3. Nilai state dapat dirubah, apabila dirubah maka halaman website akan re-render dan merubah elemen-elemen yang dipengaruhi oleh state tersebut.
4. State dimiliki oleh class component, sedangkan props dimiliki oleh class maupun function component
5. State biasa digunakan untuk komunikasi internal di dalam class component tersebut, sedangkan props biasa digunakan untuk komunikasi antar elemen (parent ke child).
6. Bisakah memanipulasi state seeperti ini ?

this.state.counter = this.state.counter + 1;

mengubah dengan cara seperti itu hanya akan merubah nilai statenya tetapi tidak akan re-render, sehingga cara tersebut sangat tidak disarankan. Secara konsep, state hanya boleh berubah untuk menyesuaikan tampilan UI pada user agar lebih dinamis, sehingga harus sesuai dengan kebutuhan tampilan. Lalu bagaimana cara solvingnya? Bisa dengan cara seperti ini

()=>{this.setState((prevState)=>({counter:prevState.counter+1}))}

Jadi, digunakan setState untuk merubah nilai state yang kemudian akan re-render halaman menyesuaikan dengan perubahan nilai state. Pada method setState dapat digunakan parameter prevState untuk memperoleh nilai state saat ini, sehingga bisa diolah.

1. Sebelum membahas tentang virtual DOM, perlu diketahui definisi DOM itu sendiri. DOM (Document Object Model) adalah salah satu kegunaan javascript yang dapat memanipulasi halaman web (elemen-elemen HTML) dengan cara merepresentasikan elemen-elemen HTML sebagai sebuah objek. Pada javascript native (pure javascript), elemen HTML tersebut harus ditampilkan terlebih dahulu sebelum dapat dilakukan manipulasi DOM, sedangkan pada ReactJS terdapat virtual DOM, yakni elemen-elemen HTML secara virtual (bukan dari halaman web) yang dapat dimanipulasi dan diproses terlebih dahulu, setelah itu, baru ditampilkan ke halaman website sebagai elemen-elemen HTML
2. Tahapan lifecycle pada ReactJS terbagi menjadi 3 bagian, yaitu:
3. Mounting (memasang)

Inti dari tahap mounting ini adalah memproses dan menampilkan elemen HTML pada website untuk tampilan awalnya. Tahapan pada mounting terdiri dari render phase (inisiasi oleh constructor, pemrosesan dan pemasangan HTML pada website) dan commit phase (pemrosesan setelah elemen HTML terpasang dengan fungsi componentDidMount() )

1. Updating (memperbarui)

Tahap ini terjadi apabila terdapat perubahan nilai state. Ketika terjadi perubahan pada state, maka constructor akan dijalankan kembali kemudian dibandingkan apakah perlu mengupdate elemen (yang terkait perbahan state) dan elemen mana saja yang perlu diupdate, lalu diproses. Tahap ini disebut render phase. Selanjutnya adalah tahap commit phase, yaitu pemrosesan setelah elemen HTML terupdate dengan fungsi componentDidUpdate()

1. Unmounting (melepas)

Tahap ini dilakukan ketika proses update menghilangkan suatu elemen. Setelah elemen tersebut hilang, dapat dijalankan fungsi componentWillUnmount()

1. Dapatkah kita memanipulasi state di constructor ?

Semua state berasal dan diproses di constructor baru kemudian dijalankan tahap selanjutnya hingga tampil sebagai elemen HTML, sehingga secara logika tidak ada gunanya merubah state pada constructor. Setelah saya mencobanya, ternyata saya menemukan 3 hal:

1. Bisa melakukan re-assign value, misal this.state ={nama:”Arkan”} value tersebut bisa diganti dengan cara this.state.nama=”…” , tetapi tidak ada gunanya (langsung aja assign …)
2. Tidak bisa melakukan re-assign value dengan setState()
3. Akan eror jika melakukan fungsi seperti componentDidMount (unexpected token, expected “;”)

