First:

```
import * as React from 'react';


class ExampleComponent extends React.Component {
 constructor(props){
 super(props);
 // Set-up our initial state
 this.state = {
 greeting: 'Hiya Buddy!'
 };
 }
 render() {
 // We can access the greeting property through this.state
 return(
 <div>{this.state.greeting}</div>
 );
 }
}
export default ExampleComponent;
```

second:

```
import React from 'react';
import ReactDOM from 'react-dom';
export default class MyComponent extends React.Component {
 constructor() {
```

```
super();
this.state = {
url: ''
}
this.onChange = this.onChange.bind(this);
}
onChange(e) {
this.setState({
url: this.props.url + '/days=?' + e.target.value
});
}
componentWillMount() {
this.setState({url: this.props.url});
}
render() {
return (
<div>
<input defaultValue={2} onChange={this.onChange} /URL: {this.state.url}
) } }
```

Third:

```
import React from 'react';
import ReactDOM from 'react-dom';
export default class MyComponent extends React.Component {
constructor() {
super();
this.state = {
days: ''
```

```
}
this.onChange = this.onChange.bind(this);
}
onChange(e) {
this.setState({
days: e.target.value
});
}
render() {
return (
<div>
<input defaultValue={2} onChange={this.onChange} />
URL: {this.props.url + '/days?=' + this.state.days}
</div>
)
}
}
```

Fourth:

```
import React from 'react';
import ReactDOM from 'react-dom';
class Greeting extends React.Component {
 constructor(props) {
 super(props);
 this.click = this.click.bind(this);
 // Set initial state (ONLY ALLOWED IN CONSTRUCTOR)
 this.state = {
 greeting: 'Hello!'
```

```
};
}
click(e) {
this.setState({
greeting: 'Uplatz Training!'
});
}
render() {
return(
<div>
<p>{this.state.greeting}</p>
<button onClick={this.click}>Click me</button>
</div>
);
}

}
export default Greeting;
```

five:

```
import React from 'react';
import {render} from 'react-dom';
class ManagedControlDemo extends React.Component {
 constructor(props){
 super(props);
 this.state = {message: ""};
 }
```

```
handleChange(e){
this.setState({message: e.target.value});
}
render() {
return (
<div>
<legend>Type something here</legend>
<input
onChange={this.handleChange.bind(this)}
value={this.state.message}
autoFocus />
<h1>{this.state.message}</h1>
</div>
)
}
}
export default ManagedControlDemo;
```

six:

```
import React from 'react';
import {render} from 'react-dom';
class Header extends React.Component {
 constructor(props) {
   super(props);
   this.state = {favoritecolor: "red"};
 }
 render() {
```

```
    return (
      <h1>My Favorite Color is {this.state.favoritecolor}</h1>
    );
  }
}
export default Header;
```

seven:

```
class Header extends React.Component {
  constructor(props) {
    super(props);
    this.state = {favoritecolor: "red"};
  }
  static getDerivedStateFromProps(props, state) {
    return {favoritecolor: props.favcol };
  }
  render() {
    return (
      <h1>My Favorite Color is {this.state.favoritecolor}</h1>
    );
  }
}


ReactDOM.render(<Header favcol="yellow"/>, document.getElementById('root'));
```

eight:

```
class Header extends React.Component {
```

```
  constructor(props) {
    super(props);
    this.state = {favoritecolor: "red"};
  }
  componentDidMount() {
    setTimeout(() => {
      this.setState({favoritecolor: "yellow"})
    }, 500)
  }
  render() {
    return (
      <h1>My Favorite Color is {this.state.favoritecolor}</h1>
    );
  }
}
```

nine:

```
class Header extends React.Component {
  constructor(props) {
    super(props);
    this.state = {favoritecolor: "red"};
  }
  static getDerivedStateFromProps(props, state) {
    return {favoritecolor: props.favcol };
  }
  changeColor = () => {
    this.setState({favoritecolor: "blue"});
  }
```

```
 render() {
  return (
   <div>
   <h1>My Favorite Color is {this.state.favoritecolor}</h1>
   <button type="button" onClick={this.changeColor}>Change color</button>
   </div>
  );
 }
}
```

ten:

```
class Header extends React.Component {
 constructor(props) {
  super(props);
  this.state = {favoritecolor: "red"};
 }
 shouldComponentUpdate() {
  return false;
 }
 changeColor = () => {
  this.setState({favoritecolor: "blue"});
 }
 render() {
  return (
   <div>
   <h1>My Favorite Color is {this.state.favoritecolor}</h1>
   <button type="button" onClick={this.changeColor}>Change color</button>
   </div>
```

```
    );
  }
}
```

eleven:

```
class Header extends React.Component {
  constructor(props) {
    super(props);
    this.state = {favoritecolor: "red"};
  }
  shouldComponentUpdate() {
    return true;
  }
  changeColor = () => {
    this.setState({favoritecolor: "blue"});
  }
  render() {
    return (
      <div>
      <h1>My Favorite Color is {this.state.favoritecolor}</h1>
      <button type="button" onClick={this.changeColor}>Change color</button>
      </div>
    );
  }
}
```

twelve:

```
class Header extends React.Component {
  constructor(props) {
```

```
  super(props);
  this.state = {favoritecolor: "red"};
 }
 changeColor = () => {
  this.setState({favoritecolor: "blue"});
 }
 render() {
  return (
   <div>
   <h1>My Favorite Color is {this.state.favoritecolor}</h1>
   <button type="button" onClick={this.changeColor}>Change color</button>
   </div>
  );
 }
}
```

13:

```
class Header extends React.Component {
 constructor(props) {
  super(props);
  this.state = {favoritecolor: "red"};
 }
 componentDidMount() {
  setTimeout(() => {
   this.setState({favoritecolor: "yellow"})
  }, 100)
 }
 getSnapshotBeforeUpdate(prevProps, prevState) {
```

```
  document.getElementById("div1").innerHTML =
  "Before the update, the favorite was " + prevState.favoritecolor;
 }
 componentDidUpdate() {
  document.getElementById("div2").innerHTML =
  "The updated favorite is " + this.state.favoritecolor;
 }
 render() {
  return (
   <div>
    <h1>My Favorite Color is {this.state.favoritecolor}</h1>
    <div id="div1"></div>
    <div id="div2"></div>
   </div>
  );
 }
}
```

14:

```
class Header extends React.Component {
 constructor(props) {
  super(props);
  this.state = {favoritecolor: "red"};
 }
 componentDidMount() {
  setTimeout(() => {
   this.setState({favoritecolor: "yellow"})
  }, 1000)
```

```
  }
  componentDidUpdate() {
    document.getElementById("mydiv").innerHTML =
    "The updated favorite is " + this.state.favoritecolor;
  }
  render() {
    return (
      <div>
      <h1>My Favorite Color is {this.state.favoritecolor}</h1>
      <div id="mydiv"></div>
      </div>
    );
  }
}
ReactDOM.render(<Header />, document.getElementById('root'));
```

15:

```
class Container extends React.Component {
  constructor(props) {
    super(props);
    this.state = {show: true};
  }
  delHeader = () => {
    this.setState({show: false});
  }
  render() {
    let myheader;
    if (this.state.show) {
```

```
    myheader = <Child />;
  };
  return (
    <div>
    {myheader}
    <button type="button" onClick={this.delHeader}>Delete Header</button>
    </div>
  );
 }
}

class Child extends React.Component {
 componentWillUnmount() {
  alert("The component named Header is about to be unmounted.");
 }
 render() {
  return (
    <h1>Hello World!</h1>
  );
 }
}
```

16:

```
class Container extends React.Component {
  constructor(props) {
    super(props);
```

```
    this.state = {
      data: 0
    }
    this.setNewNumber = this.setNewNumber.bind(this)
  };
  setNewNumber() {
    this.setState({data: this.state.data + 1})
  }
  render() {
    return (
      <div>
        <button onClick = {this.setNewNumber}>INCREMENT</button>
        <Content myNumber = {this.state.data}></Content>
      </div>
    );
  }
}
class Content extends React.Component {
  componentWillMount() {
    console.log('Component WILL MOUNT!')
  }
  componentDidMount() {
    console.log('Component DID MOUNT!')
  }
  componentWillReceiveProps(newProps) {
    console.log('Component WILL RECIEVE PROPS!')
  }
```

```
  shouldComponentUpdate(newProps, newState) {
    return true;
  }
  componentWillUpdate(nextProps, nextState) {
    console.log('Component WILL UPDATE!');
  }
  componentDidUpdate(prevProps, prevState) {
    console.log('Component DID UPDATE!')
  }
  componentWillUnmount() {
    console.log('Component WILL UNMOUNT!')
  }
  render() {
    return (
      <div>
        <h3>{this.props.myNumber}</h3>
      </div>
    );
  }
}
```