**Chapter-17**

**Refs:**

Refs make it possible to access DOM node directly with react.

Ex .:--used for things like Focusing to text input. Suppose we a form, as soon as page loaded we want input field to be focused (cursor with be in input field).

Three simple steps to be followed for this.

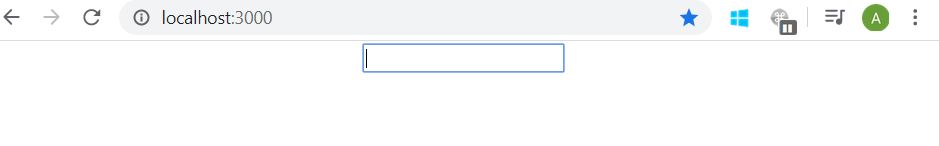
1. Create a ref. using **React.createref().**

We should create ref in constructor so they can be referenced throw-out the component.

1. Attach this ref to input element in the render method.

To attach we make user to a ‘**ref**’ reserved attribute.

1. Call the focus method to this input element



import React, { Component } from "react";

class RefsDemo extends Component {

  constructor(props) {

    super(props);

    this.inputref = React.createRef();

  }

  componentDidMount() {

    this.inputref.current.focus();

  }

  render() {

    return (

      <div>

        <input type="text" ref={this.inputref} />

      </div>

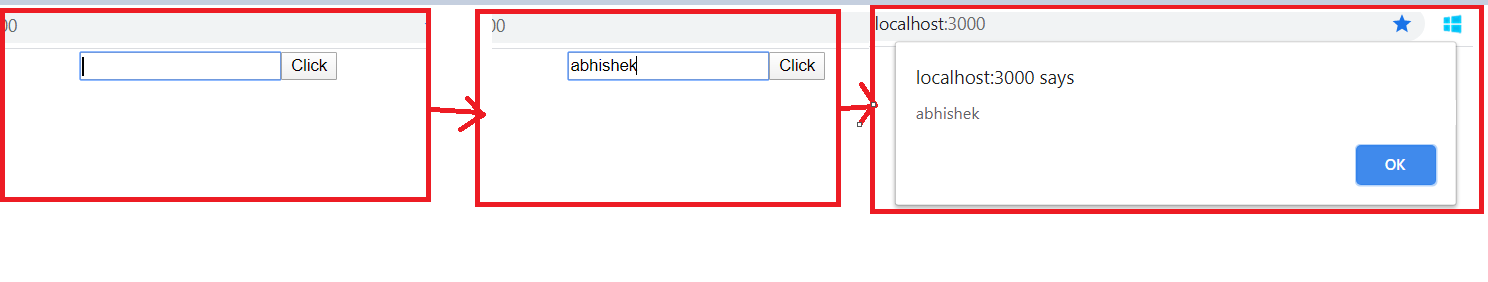
    );

  }

}

export default RefsDemo;

Eg.2 )



import React, { Component } from "react";

class RefsDemo extends Component {

  constructor(props) {

    super(props);

    this.inputref = React.createRef();

  }

  componentDidMount() {

    this.inputref.current.focus();

    console.log(this.inputref);

  }

  clickHandler = () => {

    alert(this.inputref.current.value);

  };

  render() {

    return (

      <div>

        <input type="text" ref={this.inputref} />

        <button onClick={this.clickHandler}>Click</button>

      </div>

    );

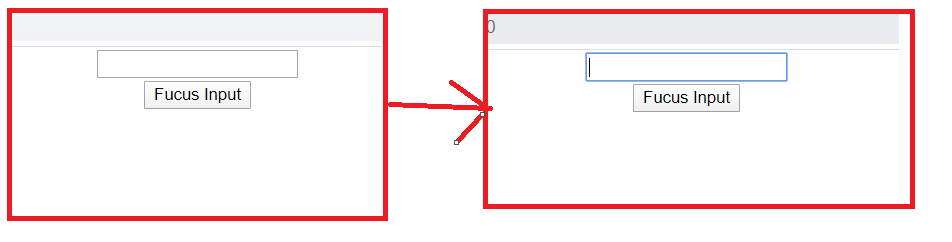
  }

}

export default RefsDemo;

**17.2 Refs with Class component**

* In last example we add ref with normal html element like input tag but it is also possible to use refs with class component
* Refs cannot be attached to functional component



import React, { Component } from "react";

class Input extends Component {

  constructor(props) {

    super(props);

    this.inputref = React.createRef();

  }

  focusInput() {

    this.inputref.current.focus();

  }

  render() {

    return (

      <div>

        <input type="text" ref={this.inputref} />

      </div>

    );

  }

}

export default Input;

import React, { Component } from "react";

import Input from "./Ch17p2Input";

class FocusInput extends Component {

  constructor(props) {

    super(props);

    this.componentref = React.createRef();

  }

  clickHandler = () => {

    this.componentref.current.focusInput();

  };

  render() {

    return (

      <div>

        <Input ref={this.componentref} />

        <button onClick={this.clickHandler}>Fucus Input</button>

      </div>

    );

  }

}

export default FocusInput;

**17.3 Forwarding refs**

It’s a technique to automatically passing a ref through a component to one of its children component.

(We will do last example using this technique)

Steps:

1. Create a ref in parent component.
2. Attach the ref to child component using ref attribute from parent component.
3. Forward this ref to the input element in child component using forwardRef method.

import React from "react";

const FRInput = React.forwardRef((props, ref) => {

  return (

    <div>

      <input type="text" ref={ref} />

    </div>

  );

});

export default FRInput;

import React, { Component } from "react";

import FRInput from "./Ch17p3FRInput";

class FRParentInput extends Component {

  constructor(props) {

    super(props);

    this.inputref = React.createRef();

  }

  clickHandler = () => {

    this.inputref.current.focus();

  };

  render() {

    return (

      <div>

        <FRInput ref={this.inputref} />

        <button onClick={this.clickHandler}>Focus Input</button>

      </div>

    );

  }

}

export default FRParentInput;