React Hooks Cheatsheet

useState Hook

useState to Create State Variables

```
import "./styles.css";
import React from 'react';

export default function App() {
  const [language] = React.useState('React!!!')

  return <h1>I am learning {language}</h1>;
}
```

Update State Variables

```
import "./styles.css";
import React from 'react';

export default function App() {
  const [language, setLanguage] = React.useState('React!!!')

  function changeLanguage() {
    setLanguage('React Hooks')
  }

  return <h1 onClick={changeLanguage}>I am learning {language}</h1>;
}
```

Can Be Used Once Or Many Times

```
import "./styles.css";
import React from "react";

export default function App() {
  const [language, setLanguage] = React.useState("React!!!");
  const [years] = React.useState(0);

  function changeLanguage() {
    setLanguage("React Hooks");
  }
```

```
  return (
    <div>
      <h1 onClick={changeLanguage}>
        I've learned {language} for {years} years
      </h1>
      <button>Add Year</button>
    </div>
  );
}
```

Update State based on Previous Value

```
import "./styles.css";
import React from "react";

export default function App() {
  const [language, setLanguage] = React.useState("React!!!");
  const [years, setYears] = React.useState(0);

  function changeLanguage() {
    setLanguage("React Hooks");
  }

  function addYear() {
    setYears(prev => prev + 1)
  }

  return (
    <div>
      <h1 onClick={changeLanguage}>
        I've learned {language} for {years} years
      </h1>
      <button onClick={addYear}>Add Year</button>
    </div>
  );
}
```

Manage State with an Object

```
import "./styles.css";
import React from "react";

export default function App() {
  // const [language, setLanguage] = React.useState("React!!!");
  // const [years, setYears] = React.useState(0);
  const [state, setState] = React.useState({
    language: "React",
    years: 0
```

```
  })

  function changeLanguage() {
    setState({...state, language: "React Hooks"});
  }

  function addYear() {
    setState(prev => {
      return {
        ...prev,
        years: prev.years + 1
      }
    })
  }

  return (
    <div>
      <h1 onClick={changeLanguage}>
        I've learned {state.language} for {state.years} years
      </h1>
      <button onClick={addYear}>Add Year</button>
    </div>
  );
}
```

useEffect Hook

useEffect to Perform Side Effects

```
import "./styles.css";
import React from "react";

export default function App() {
  React.useEffect(() => {
    document.body.style.background = 'navy';
    document.body.style.color = 'white';
  })

  return (
    <div>
      <h1>React App</h1>
    </div>
  );
}
```

Run Again when a Value Changes

```
import "./styles.css";
import React from "react";

export default function App() {
  const [color, setColor] = React.useState('navy')

  React.useEffect(() => {
    document.body.style.background = color;
    document.body.style.color = 'white';
  }, [color]);

  function changeColor() {
    setColor('gold')
  }

  return (
    <div>
      <h1>React App</h1>
      <button onClick={changeColor}>Change color</button>
    </div>
  );
}
```

Unsubscribe by Returning a Function

```
import "./styles.css";
import React from "react";

export default function App() {
  const [color, setColor] = React.useState('navy')

  React.useEffect(() => {
    document.body.style.background = color;
    document.body.style.color = 'white';

    window.addEventListener('keydown', handleEnterButton)

    return () => {
      window.removeEventListener('keydown', handleEnterButton)
    }
  }, [color]);

  function changeColor() {
    setColor('gold')
  }

  function handleEnterButton(event) {
    if (event.keyCode === 13) {
      setColor('red')
```

```
    }
  }

  return (
    <div>
      <h1>React App</h1>
      <button onClick={changeColor}>Get color</button>
    </div>
  );
}
```

Fetch Data from an API

```
import "./styles.css";
import React from "react";

export default function App() {
  const [color, setColor] = React.useState('navy')
  const [user, setUser] = React.useState(null)

  React.useEffect(() => {
    fetch('https://randomuser.me/api/')
      .then(res => res.json())
      .then(data => setUser(data.results[0]))
  }, [])

  React.useEffect(() => {
    document.body.style.background = color;
    document.body.style.color = 'white';

    window.addEventListener('keydown', handleEnterButton)

    return () => {
      window.removeEventListener('keydown', handleEnterButton)
    }
  }, [color]);

  function changeColor() {
    setColor('gold')
  }

  function handleEnterButton(event) {
    if (event.keyCode === 13) {
      setColor('red')
    }
  }

  return (
    <div>
      <h1>React App</h1>
      <button onClick={changeColor}>Get color</button>
```

```
      <br />
      <br />
      Current user: <pre>{JSON.stringify(user, null, 2)}</pre>
    </div>
  );
}
```

useRef Hook

useRef to Reference React Elements

```
import './styles.css'
import React from 'react';

export default function App() {
  const inputRef = React.useRef(null)

  function handleClearInput() {
    inputRef.current.value = "";
    inputRef.current.focus();
  }

  return (
    <form>
      <input
        type="text"
        ref={inputRef}
      />
      <button type="button" onClick={handleClearInput}>
        Clear Input
      </button>
    </form>
  );
}
```

useCallback Hook

useCallback Prevents Callbacks from Being Recreated

```
import './styles.css'
import React from 'react';

export default function App() {
  const [skill, setSkill] = React.useState("");
  const [skills, setSkills] = React.useState(["HTML", "CSS", "JavaScript"]);
```

```
  function handleChangeInput(event) {
    setSkill(event.target.value);
  }

  function handleAddSkill() {
    setSkills(skills.concat(skill));
  }

  const handleRemoveSkill = React.useCallback((skill) => {
    setSkills(skills.filter((s) => s !== skill));
  }, [skills])

  return (
    <>
      <input onChange={handleChangeInput} />
      <button onClick={handleAddSkill}>Add Skill</button>
      <SkillList skills={skills} handleRemoveSkill={handleRemoveSkill} />
    </>
  );
}

const SkillList = React.memo(({ skills, handleRemoveSkill }) => {
  console.log('re-rendered whenever parent state is updated!')
  return (
    <ul>
      {skills.map((skill) => (
        <li key={skill} onClick={() => handleRemoveSkill(skill)}>
          {skill}
        </li>
      ))}
    </ul>
  );
});
```

useMemo Hook

useMemo Can Improve Expensive Operations

```
import './styles.css'
import React from 'react';

const skills = ["HTML", "CSS", "JavaScript", '...1000s more']

export default function App() {
  const [searchTerm, setSearchTerm] = React.useState("");

  const searchResults = React.useMemo(() => {
    return skills.filter(s => s.includes(searchTerm));
  }, [searchTerm])
```

```
  function handleSearchInput(event) {
    setSearchTerm(event.target.value);
  }

  return (
    <>
      <h3>Search Results</h3>
      <input onChange={handleSearchInput} />
      <ul>
        {searchResults.map((result, i) => (
          <li key={i}>{result}</li>
        ))}
      </ul>
    </>
  );
}
```

useContext Hook

useContext Helps Us Avoid Prop Drilling

```
import './styles.css'
import React from 'react';

const UserContext = React.createContext()

export default function App() {
  const [user] = React.useState({ name: "Fred" });

  return (
    <UserContext.Provider value={user}>
      <Main />
    </UserContext.Provider>
  );
}

const Main = () => (
  <>
    <Header  />
    <br />
    <div>Main app content</div>
  </>
);

const Header = () => {
  const user = React.useContext(UserContext)

  return <h1>Welcome, {user.name}!</h1>;
}
```

useReducer Hook

useReducer is (Another) Powerful State Management Tool

```
export default function App() {
  const [state, dispatch] = React.useReducer(reducer, initialState)

  function handleLogin() {
    dispatch({
      type: "LOGIN",
      payload: {
        username: "Reed"
      }
    })
  }

  function handleSignout() {
    dispatch({
      type: "SIGNOUT"
    })
  }

  return (
    <>
      Current user: {state.username}
      <br />
      isAuthenticated: {JSON.stringify(state.isAuth)}
      <br />
      <button onClick={handleLogin}>Login</button>
      <button onClick={handleSignout}>Signout</button>
    </>
  );
}
```