React 18 for Beginners: Managing State

App.tsx

```
/* UNDERSTANDING STATE:
 - State is a way to store data in a component
 - State Hooks cannot be used inside of conditional statements
 - State Hooks must be used at the top level of a component
 - The state is stored in an array of states, which are accessed by index and
 mapped to a variable name
 -  State is private and fully controlled by the component that defines it.
 Other components cannot directly access or modify another component's state.
 - State should be kept to a minimum and only include data that is necessary
 for the component to function correctly.
 - State updates can be asynchronous, which means that it batches updates and
 applies them all at once and then re-renders the state with the new variables. */

import React, { useState } from "react";
import styled from "styled-components";
import Names from "./components/Names";
import MessageImpure from "./components/MessageImpure";
import MessagePure from "./components/MessagePure";
import NamesButton from "./components/NamesButton";
import Drink from "./components/Drink";
import Customer from "./components/Customer";
import Tags from "./components/Tags";
import Bugs from "./components/Bugs";
import BugsImmer from "./components/BugsImmer";
import ShoppingCart from "./components/ShoppingCart";
import Game from "./components/Game";
import Pizza from "./components/Pizza";
import ItemQuantites from "./components/ItemQuantities";
import ExpandableText from "./components/ExpandableText";

const Container = styled.div`
 display: flex;
 flex-direction: column;
 align-items: center;
 margin: 10px;
`;

const InnerContainer = styled.div`
 display: flex;
 flex-direction: column;
 align-items: center;
 margin: 15px;
 height: 215px;
`;

const TextContainer = styled.div`
 width: 400px;
```

```
 background-color: deeppink;
 font-size: 16px;
 padding: 10px 10px;
 border-radius: 10px;
`;

const ButtonCSS = styled.button`
 padding: 5px 20px;
 margin: 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
`;

const NamesButtonsCSS = styled.div`
 display: flex;
 flex-direction: row;
 justify-content: center;
`;

function App() {
 const [isVisible, setVisibility] = useState(false);
 const [isApproved, setApproval] = useState(true);

 const handleClick = (isVisible: boolean) => {
   setVisibility(isVisible);
 };

 return (
   <Container>
     <InnerContainer>
       <div>
         <ButtonCSS onClick={() => handleClick(true)}>Make Visible</ButtonCSS>
         <ButtonCSS onClick={() => handleClick(false)}>
           Make Invisible
         </ButtonCSS>
       </div>

       <div>
         {isVisible && (
           <>
             <Names firstName="Evan Marie" lastName="Carr" />
             <Names firstName="Cheese" lastName="Burger" />
           </>
         )}
       </div>
     </InnerContainer>
     <>
       {/* <div>
         <MessageImpure />
         <MessageImpure />
       </div> */}
       <InnerContainer>
         <MessagePure />
```

```
    <MessagePure />
  </InnerContainer>
</>
<InnerContainer>
  <NamesButtonsCSS>
    <NamesButton firstName="Evan Marie" lastName="Carr" />
    <NamesButton firstName="Emma" lastName="Stefanuk" />
  </NamesButtonsCSS>
</InnerContainer>

<InnerContainer>
  <Drink />
</InnerContainer>

<InnerContainer>
  <Customer />
</InnerContainer>

<InnerContainer>
  <Tags />
</InnerContainer>

<InnerContainer>
  <Bugs />
</InnerContainer>

<InnerContainer>
  <BugsImmer />
</InnerContainer>

<InnerContainer style={{ height: "325px" }}>
  <ShoppingCart />
</InnerContainer>

<InnerContainer>
  <Game />
</InnerContainer>

<InnerContainer>
  <Pizza />
</InnerContainer>

<InnerContainer style={{ height: "325px" }}>
  <ItemQuantites />
</InnerContainer>

<InnerContainer>
  <TextContainer>
    <ExpandableText maxChars={100}>
      Lorem ipsum dolor sit amet consectetur adipisicing elit. Voluptatem
      error maiores consequatur aspernatur id at, voluptates delectus
      molestiae deserunt doloremque sequi ullam similique nulla reiciendis
      doloribus ex dolor facilis quasi, voluptatibus recusandae iusto,
      atque sunt veritatis corrupti. Magnam incidunt, quos, et inventore
```

```
          earum praesentium at similique tempore pariatur hic accusamus iure.
          Repudiandae nostrum, doloribus itaque suscipit optio repellendus.
          Illo dolore explicabo deleniti dolorem ad porro ullam officiis hic
          sed architecto? Non aut a maxime doloremque quas aspernatur minima,
          nesciunt officia rem in libero nisi nulla maiores necessitatibus.
          Velit delectus beatae autem sunt molestiae nulla cupiditate, a
          quibusdam, quidem magnam nostrum.
        </ExpandableText>
      </TextContainer>
    </InnerContainer>
  </Container>
 );
}

export default App;
```

Names.tsx:

```
import React, { useState } from "react";
import styled from "styled-components";

interface Props {
 firstName: string;
 lastName: string;
}

const NameDiv = styled.div`
 color: deeppink;
 background-color: cyan;
 padding: 5px 10px;
 border-radius: 5px;
 font-size: 16px;
 margin: 10px;
`;

function Names(props: Props) {
 // names individually stored in state
 //   const [firstName, setFirstName] = useState("");
 //   const [lastName, setLastName] = useState("");

 // names stored in state as an object, better than deeply nested state
 const [person, setPerson] = useState({
   firstName: props.firstName,
   lastName: props.lastName,
 });

 // full name stored in state
 const fullName = `${person.firstName} ${person.lastName}`;

 // function is loading: manages showing spinner to user
 const [isLoading, setIsLoading] = useState(false);
```

```
 return (
   <div>
     <NameDiv>{fullName}</NameDiv>
   </div>
 );
}

export default Names;
```

MessagePure.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const MessageDiv = styled.div`
 color: cyan;
 background-color: deeppink;
 margin: 10px;
 padding: 5px 10px;
 border-radius: 5px;
 font-size: 13px;
`;

const CountDiv = styled.div`
 color: cyan;
 text-align: center;
 font-size: 18px;
 padding: 5px 10px;
`;

const MessagePure = () => {
 const [count, setCount] = useState(0);

 const handleClick = () => {
   setCount(count + 1);
 };

 return (
   <MessageDiv>
     <CountDiv>Count: {count}</CountDiv>
     <button onClick={handleClick}>Increment Count</button>
   </MessageDiv>
 );
};

export default MessagePure;
```

MessageImpure.tsx

```
import React from "react";
import styled from "styled-components";
```

```
const MessageDiv = styled.div`
 color: deeppink;
 background-color: cyan;
 margin: 10px;
 padding: 0 10px;
 border-radius: 5px;
 font-size: 13px;
`;

/* defining count here will keep component pure as long as changes do not
take place inside of the rendering of the component
- multiple instances of the component will render same count
let count = 0; */

const MessageImpure = () => {
 /* to keep component pure, keep state changes out of the component itself
 count++;
 but if you define count here, you can also change here */

 let count = 0;
 count++;
 return (
   <MessageDiv>
     <h1>Count: {count}</h1>
   </MessageDiv>
 );
};

export default MessageImpure;
```

NamesButton.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

interface Props {
 firstName: string;
 lastName: string;
}

const NameDiv = styled.div`
 color: deeppink;
 background-color: cyan;
 padding: 5px 10px;
 border-radius: 5px;
 font-size: 16px;
 margin: 10px;
`;

const ButtonCSS = styled.button`
 padding: 5px 20px;
```

```
 margin: 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
`;

const MessageCSS = styled.div`
 padding: 5px 20px;
 margin: 10px;
 font-size: 18px;
 font-family: monospace;
 background-color: cyan;
 color: #222222;
 border-radius: 5px;
 padding: 5px 10px;
 text-align: center;
`;

function NamesButton(props: Props) {
 const [isVisible, setIsVisible] = useState(true);
 const fullName = `${props.firstName} ${props.lastName}`;

 const toggleVisibility = () => {
   setIsVisible(!isVisible);
 };

 return (
   <div>
     <ButtonCSS onClick={toggleVisibility}>Toggle Visibility</ButtonCSS>
     {isVisible && <MessageCSS>{fullName}</MessageCSS>}
   </div>
 );
}

export default NamesButton;
```

Drink.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const DrinkCSS = styled.div`
 display: flex;
 background-color: #222222;
 color: cyan;
 padding: 10px, 15px;
 border-radius: 10px;
 font-family: monospace;
 flex-direction: column;
 align-items: center;
 text-align: left;
 margin: 10px;
```

```
`;

const DrinkLine = styled.p`
 margin: 1px;
 font-size: 16px;
`;

const Box = styled.div`
 margin: 10px 10px;
 background-color: #333333;
 padding: 10px 10px;
`;

function Drink() {
 const [drink, setDrink] = useState({
   name: "Americano",
   price: 3.99,
   size: "medium",
 });

 const handleClick = () => {
   /* keeps all the properties of drink that are the same
   and only updates the price.
   This is one reason to avoid deeply nested structures. */
   setDrink({ ...drink, price: 4.99 });
 };

 return (
   <DrinkCSS>
     <button onClick={handleClick}>Update Drink Price</button>
     <Box>
       <DrinkLine>Drink: {drink.name}</DrinkLine>
       <DrinkLine>Price: {drink.price}</DrinkLine>
       <DrinkLine>Size: {drink.size}</DrinkLine>
     </Box>
   </DrinkCSS>
 );
}

export default Drink;
```

Customer.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const CustomerCSS = styled.div`
 display: flex;
 background-color: #333333;
 padding: 10px 15px;
 border-radius: 10px;
 color: cyan;
```

```
 font-family: monospace;
 flex-direction: column;
 align-items: left;
 margin: 10px;
`;

const Button = styled.button`
 padding: 5px 20px;
 margin: 10px 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
`;

const CustomerLine = styled.p`
 margin: 1px;
 font-size: 16px;
`;

function Customer() {
 const [customer, setCustomer] = useState({
   name: "Evan Marie Carr",
   age: 44,
   address: {
     street: "10585 Haynes Forest Drive",
     city: "Alpharetta",
     state: "GA",
     zip: 30022,
   },
   phone: "678-520-0677",
 });

 const handleClick = () => {
   setCustomer({
     /* spread operator makes a shallow copy when creating a new object
       in nested objects, changing properties requires an operation
       that is more complex than a shallow copy */

     ...customer,
     address: { ...customer.address, zip: 30023 },
   });
 };

 return (
   <CustomerCSS>
     <Button onClick={handleClick}>Update Customer Zip</Button>
     <CustomerLine>Name: {customer.name}</CustomerLine>
     <CustomerLine>Age: {customer.age}</CustomerLine>
     <CustomerLine>Address: {customer.address.street}</CustomerLine>
     <CustomerLine>City: {customer.address.city}</CustomerLine>
     <CustomerLine>State: {customer.address.state}</CustomerLine>
     <CustomerLine>Zip: {customer.address.zip}</CustomerLine>
     <CustomerLine>Customer Phone: {customer.phone}</CustomerLine>
   </CustomerCSS>
```

```
 );
}

export default Customer;
```

Tags.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const TagsCSS = styled.div`
 display: flex;
 background-color: #333333;
 padding: 10px 15px;
 border-radius: 10px;
 height: 75px;
 align-items: center;
 color: cyan;
 font-family: monospace;
 flex-direction: column;
 align-items: left;
 margin: 10px;
`;

const TagsLine = styled.p`
 margin: 1px;
 font-size: 16px;
`;

function Tags() {
 // We will add 'exciting' to this array with the click of a button
 const [tags, setTags] = useState(["happy", "cheerful", "joyous"]);

 const addExciting = () => {
   // not using .push, because we don't want to mutate the array
   setTags([...tags, "exciting"]);
 };

 const removeCheerful = () => {
   // now we will remove 'happy' from the array
   setTags(tags.filter((tag) => tag !== "cheerful"));
 };

 const changeHappy = () => {
   /* updating an object: iterate over items, if item is 'happy',
   change it to 'so happy', otherwise keep it the same */
   setTags(tags.map((tag) => (tag === "happy" ? "so happy" : tag)));
 };

 return (
   <div>
     <TagsCSS>
       {tags.map((tag) => (
```

```
        <TagsLine>{tag}</TagsLine>
      ))}
    </TagsCSS>
    <button onClick={addExciting}>Add Exciting</button>
    <button onClick={removeCheerful}>Remove Cheerful</button>
    <button onClick={changeHappy}>Change Happy</button>
  </div>
 );
}

export default Tags;
```

Bugs.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const BugsCSS = styled.div`
 padding: 5px 10px;
 margin: 10px 15px;
`;

const BugsLine = styled.div`
 margin: 1px;
 color: white;
`;
const Button = styled.button`
 padding: 5px 10px;
 margin: 5px 10px;
 border-radius: 5px;
`;

const Box = styled.div`
 margin: 10px 10px;
 background-color: #333333;
 padding: 10px 10px;
 border-radius: 10px;
 align-items: center;
`;

function Bugs() {
 const [bugs, setBugs] = useState([
   { id: 1, title: "Bug 1", status: "not fixed" },
   { id: 2, title: "Bug 2", status: "not fixed" },
   { id: 3, title: "Bug 3", status: "not fixed" },
 ]);

 const fixBug1 = () => {
   setBugs(
     bugs.map((bug) => (bug.id === 1 ? { ...bug, status: "fixed" } : bug))
   );
```

```
 };

 const fixBug2 = () => {
   setBugs(
     bugs.map((bug) => (bug.id === 2 ? { ...bug, status: "fixed" } : bug))
   );
 };

 const fixBug3 = () => {
   setBugs(
     bugs.map((bug) => (bug.id === 3 ? { ...bug, status: "fixed" } : bug))
   );
 };

 return (
   <Box>
     <Box>
       {bugs.map((bug) => (
         <BugsLine>
           ID: {bug.id} | Description: {bug.title} | status: {bug.status}
         </BugsLine>
       ))}
     </Box>
     <Box>
       <Button onClick={fixBug1}>Fix Bug 1</Button>
       <Button onClick={fixBug2}>Fix Bug 2</Button>
       <Button onClick={fixBug3}>Fix Bug 3</Button>
     </Box>
   </Box>
 );
}

export default Bugs;
```

BugsImmer.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const Box = styled.div`
 margin: 10px 10px;
 font-size: 23px;
 font-weight: bold;
 background-color: #333333;
 color: deeppink;
 padding: 10px 10px;
 border-radius: 10px;
 align-items: center;
`;

interface Props {
 cartItemsCount: number;
}
```

```
const NavBar = ({ cartItemsCount }: Props) => {
 return <Box>Total Items in Cart: {cartItemsCount}</Box>;
};

export default NavBar;
```

ShoppingCart.tsx

```
import React, { useState } from "react";
import styled from "styled-components";
import Cart from "./ShoppingCart2";
import NavBar from "./ShoppingCartNav";

const Box = styled.div`
 margin: 10px 10px;
 background-color: #333333;
 color: white;
 padding: 5px 40px;
 border-radius: 10px;
 align-items: center;
`;

function ShoppingCart() {
 const [cartItems, setCartItems] = useState([
   "Product 01",
   "Product 02",
   "Product 03",
 ]);

 /* onClear is here instead of Cart, because this function controls state -
   This will set the items to an empy array */
 return (
   <Box>
     <NavBar cartItemsCount={cartItems.length} />
     <Cart
       cartItems={cartItems}
       onClear={() => {
         setCartItems([]);
       }}
     />
   </Box>
 );
}

export default ShoppingCart;
```

ShoppingCart2.tsx

```
import React, { useState } from "react";
```

```
import styled from "styled-components";

const Box = styled.div`
 margin: 10px 10px;
 background-color: #333333;
 padding: 10px 10px;
 border-radius: 10px;
 align-items: center;
 text-align: center;
`;

const YourShopping = styled.div`
 font-size: 17px;
 color: cyan;
`;

const Items = styled.ul`
 list-style: none;
`;

const Button = styled.button`
 padding: 5px 20px;
 margin: 10px 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
`;

interface Props {
 cartItems: string[];
 onClear: () => void;
}

const Cart = ({ cartItems, onClear }: Props) => {
 return (
   <Box>
     <YourShopping>Your Shopping Cart:</YourShopping>
     <Items>
       {cartItems.map((item) => (
         <li key={item}>{item}</li>
       ))}
     </Items>
     <Button onClick={onClear}>Clear Cart</Button>
   </Box>
 );
};

export default Cart;
```

ShoppingCartNav.tsx

```
import React, { useState } from "react";
import styled from "styled-components";
```

```
const Box = styled.div`
 margin: 10px 10px;
 font-size: 23px;
 font-weight: bold;
 background-color: #333333;
 color: deeppink;
 padding: 10px 10px;
 border-radius: 10px;
 align-items: center;
`;

interface Props {
 cartItemsCount: number;
}

const NavBar = ({ cartItemsCount }: Props) => {
 return <Box>Total Items in Cart: {cartItemsCount}</Box>;
};

export default NavBar;
```

Game.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const Box = styled.div`
 margin: 10px 10px;
 font-size: 17px;
 font-weight: bold;
 background-color: #333333;
 color: cyan;
 padding: 10px 15px;
 border-radius: 10px;
 align-items: center;
 text-align: center;
`;

const Button = styled.button`
 padding: 5px 20px;
 margin: 10px 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
`;

function Game() {
 const [game, setGame] = useState({
   id: 1,
   player: {
     name: "John",
```

```
    },
  });

  const handleClick = () => {
    setGame({
      ...game,
      player: { ...game.player, name: "Bob" },
    });
  };

  return (
    <Box>
      <p>Player Id: {game.id}</p>
      <p>Player Name: {game.player.name}</p>
      <Button onClick={handleClick}>Bob's Turn!</Button>
    </Box>
  );
}

export default Game;
```

Pizza.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const Box = styled.div`
 margin: 10px 10px;
 font-size: 17px;
 font-weight: bold;
 background-color: #333333;
 color: cyan;
 padding: 10px 15px;
 border-radius: 10px;
 align-items: center;
 text-align: center;
`;

const Button = styled.button`
 padding: 5px 20px;
 margin: 10px 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
`;

function Pizza() {
 const [pizza, setPizza] = useState({
   name: "Spicy Pepperoni",
   toppings: ["Mushroom,"],
 });
```

```
 const handleClick = () => {
   setPizza({
     ...pizza,
     toppings: [...pizza.toppings, "Extra Cheese,"],
   });
 };

 return (
   <Box>
     <p>Pizza: {pizza.name}</p>
     <p>Toppings: {pizza.toppings}</p>
     <Button onClick={handleClick}>Gimme Cheese!</Button>
   </Box>
 );
}

export default Pizza;
```

ItemQuantities.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const Box = styled.div`
 margin: 10px 10px;
 font-size: 17px;
 font-weight: bold;
 background-color: #333333;
 color: cyan;
 padding: 10px 15px;
 border-radius: 10px;
 align-items: center;
 text-align: center;
`;

const Button = styled.button`
 padding: 5px 20px;
 margin: 10px 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
`;

function ItemQuanities() {
 const [itemQuants, setItemQuants] = useState({
   discount: 0.1,
   items: [
     { id: 1, title: "Product 1", quantity: 1 },
     { id: 2, title: "Product 2", quantity: 2 },
     { id: 3, title: "Product 3", quantity: 3 },
   ],
 });
```

```
 const handleClick = () => {
   setItemQuants({
     ...itemQuants,
     items: itemQuants.items.map((item) =>
       item.id === 1 ? { ...item, quantity: item.quantity + 1 } : item
     ),
   });
 };

 return (
   <Box>
     {itemQuants.items.map((item) => (
       <div key={item.id}>
         <p>{item.title}</p>
         <p>Quantity: {item.quantity}</p>
       </div>
     ))}
     <Button onClick={handleClick}>Add to quantity</Button>
   </Box>
 );
}

export default ItemQuanities;
```

ExpandableText.tsx

```
import React, { useState } from "react";
import styled from "styled-components";

const Box = styled.div`
 margin: 10px 10px;
 font-size: 17px;
 font-weight: bold;
 background-color: #333333;
 color: cyan;
 padding: 10px 15px;
 border-radius: 10px;
 align-items: center;
 text-align: center;
`;

const Button = styled.button`
 padding: 5px 20px;
 margin-top: 10px;
 font-family: monospace;
 font-size: 16px;
 border-radius: 5px;
 width: 100px;
`;
```

```
const TextContainer = styled.div`
 display: flex;
 flex-direction: column;
 margin: 5px 5px;
`;

interface Props {
 children: string;
 maxChars?: number;
}

const ExpandableText = ({ children, maxChars = 100 }: Props) => {
 const [isExpanded, setExpanded] = useState(false);

 if (children.length <= maxChars) return <p>{children}</p>;
 const text = isExpanded ? children : children.substring(0, maxChars) + "...";
 return (
   <TextContainer>
     {text}
     <Button onClick={() => setExpanded(!isExpanded)}>
       {isExpanded ? "Less" : "More"}
     </Button>
   </TextContainer>
 );
};

export default ExpandableText;
```