Cheat Sheet: Understanding Function Components with Array and DOM Manipulation

Function Component with Array and DOM	Description	Code Example

Function Component with function keyword	Function component starts with function keyword along with name of the component and includes html tags within return. It also exports component name by default	`import React from 'react'`
		`function Extra() {`
		`  return (`
		`  <>`
		`  <p>This is paragraph</p>`
		`  </>`
		`  )`
		`}`
		`export default Extra`

Function Component with arrow function	Function component starts with variable type along with name of the component and includes html tags within return. It also exports component name by default	`import React from 'react'`
		`const Extra = () => {`
		`  return (`
		`  <div>Extra</div>`
		`  )`
		`}`
		`export default Extra`

Props in function component	Props can be sent from parent component as attribute along with child component	`import React from 'react'`
		`import ChildComponent from './ChildComponent'`
		`function ParentComponent () {`
		`  let title='Project Manager';`
		`  return (`
		`  <>`
		`  <ChildComponent title={title}/>`
		`  </>`
		`  )`
		`}`
		`export default ParentComponent`

Access Props within child function component	Props can be accessed easily within the child function component using `props.variable_name`	`import React from 'react'`
		`const ChildComponent = (props) => {`
		`  return (`
		`  <>`
		`  <p>The title is {props.title}</p>`
		`  </>`
		`  )`
		`}`
		`export default ChildComponent`

Event handling in class component	Events such as click event can be performed by calling function which is declared before return of function component	`import React from 'react'`
		`const Extra = (props) => {`
		`  function show(){`
		`  console.log('Show function');`
		`  }`
		`  return (`
		`  <>`
		`  <p>The title is {props.title}</p>`
		`  <button onClick={()=>show()}>Click Here</button>`
		`  </>`
		`  )`
		`}`
		`export default Extra`

State management in function component	State management can be done easily with `useState()` hook	`import React, { useState } from 'react'`
		`const StateManagement = () => {`
		`  const[name,setName]=useState('John');`
		`  return (`
		`  <>`
		`  <h1>State Management using useState</h1>`
		`  <p>The name is {name}</p>`
		`  </>`
		`  )`
		`}`
		`export default StateManagement`

Array Declaration	Array can be declared in square brackets	`const names = ['Alice', 'Bob', 'Charlie'];`
Stateful Array	Array can be declared using useState	`const [todos, setTodos] = useState(['Learn React', 'Build Project']);`

Dynamically Constructed Arrays	Arrays can be constructed dynamically based on application logic or received data	`const numbers = [];`
		`for (let i = 0; i < 10; i++) {`
		`  numbers.push(i);`
		`}`

Array map() method	The map() method is commonly used to iterate over each element of an array and return a new array of React elements	`const items = ['Apple', 'Banana', 'Orange'];`
		`const itemList = items.map((item, index) => <li key={index}>{item}</li>);`
		`return <ul>{itemList}</ul>;`

for...of Loop	You can use the for...of loop to iterate over the elements of an array:	`const items = ['Apple', 'Banana', 'Orange'];`
		`for (const item of items) {`
		`  console.log(item);`
		`}`

Rendering a List of Items	You can render a list of items by mapping over an array and returning a JSX element for each item	`import React from 'react';`
		`function ArrayComponent() {`
		`  const items = ['Autumn', 'Spring', 'Summer','Winter'];`
		`  return (`
		`  <div>`
		`  <h1> Seasons Names</h1>`
		`  <ul>`
		`  {items.map((item, index) => (`
		`  <li key={index}>{item}</li>`
		`  ))}`
		`  </ul>`
		`  </div>`
		`  );`
		`}`
		`export default ArrayComponent;`

Adding and removing items in array	You can add or remove items from an array using state and React's setState method	`import React, { useState } from 'react';`
		`function MyComponent() {`
		`  const [items, setItems] = useState([‘Autumn’, ‘Spring’, ‘Winter’,’Summer’]);`
		`  const [inputValue, setInputValue] = useState('');`
		`  const addItem = () => {`
		`  setItems([...items, inputValue]);`
		`  setInputValue('');`
		`  };`
		`  const removeItem = (index) => {`
		`  const newItems = [...items];`
		`  newItems.splice(index, 1);`
		`  setItems(newItems);`
		`  };`
		`  return (`
		`  <div>`
		`  <h1>Fruits</h1>`
		`  <ul>`
		`  {items.map((item, index) => (`
		`  <li key={index}>`
		`  {item}`
		`  <button onClick={() => removeItem(index)}>Remove</button>`
		`  </li>`
		`  ))}`
		`  </ul>`
		`  <input`
		`  type="text"`
		`  value={inputValue}`
		`  onChange={(e) => setInputValue(e.target.value)}`
		`  />`
		`  <button onClick={addItem}>Add</button>`
		`  </div>`
		`  );`
		`}`

Conditional rendering using ternary operator	You can conditionally render components based on the content of an array	`import React, { useState } from 'react';`
		`function ArrayComponent() {`
		`  const [items, setItems] = useState(['React JS','Vue JS','Angular JS','Vanilla JS']);`
		`  return (`
		`  <div>`
		`  <h1>Front End Languages</h1>`
		`  {items.length > 0 ? (`
		`  <ul>`
		`  {items.map((item, index) => (`
		`  <li key={index}>{item}</li>`
		`  ))}`
		`  </ul>`
		`  ) : (`
		`  <p>No Front End language is available</p>`
		`  )}`
		`  </div>`
		`  );`
		`}`
		`export default ArrayComponent;`

inline style in react	Inline style can be applied within the tag as an attribute within double curly braces	`import React from 'react';`
		`function MyComponent() {`
		`  return (`
		`  <div style={{ backgroundColor: 'lightblue', padding: '20px', borderRadius: '5px' }}>`
		`  <p style={{ color: 'white', fontSize: '18px' }}>This is a paragraph with inline styles.</p>`
		`  </div>`
		`  );`
		`}`
		`export default MyComponent;`

Style using object	Style can be applied as an object like inline style	`import React, { useState } from 'react';`
		`function ToggleMessage() {`
		`  const [isVisible, setIsVisible] = useState(true);`
		`  const toggleVisibility = () => {`
		`  setIsVisible(!isVisible);`
		`  };`
		`  const messageStyle = {`
		`  display: isVisible ? 'block' : 'none',`
		`  color: 'green',`
		`  fontSize: '18px',`
		`  marginTop: '10px'`
		`  };`
		`  return (`
		`  <div>`
		`  <h2>Toggle Message</h2>`
		`  <button onClick={toggleVisibility}>`
		`  {isVisible ? 'Hide Message' : 'Show Message'}`
		`  </button>`
		`  <p style={messageStyle}>This is a hidden message.</p>`
		`  </div>`
		`  );`
		`}`


Skills Network