React Component Definition (Advanced)

Let's go over another way to define the react component.

We'll cover the following

The following refactorings are optional recommendations to explain the different JavaScript/React patterns. You can build complete React applications without these advanced patterns, so don't be discouraged if they seem too complicated.

All components in the *src/App.js* file are function components. JavaScript has multiple ways to declare functions. So far, we have used the function statement, though arrow functions can be used more concisely:

```
// function declaration
function () { ... }

// arrow function declaration
const () => { ... }
```

src/App.js

You can remove the parentheses in an arrow function expression if it has only one argument, but multiple arguments require parentheses:

```
// allowed
const item => { ... }

// allowed
const (item) => { ... }

// not allowed
const item, index => { ... }

// allowed
const (item, index) => { ... }
```

src/App.js

Defining React function components with arrow functions makes them more concise:

```
const App = () => {
  return (
    <div>
      ...
    </div>
  );

};
const List = () => {
  return list.map(function(item) {
    return (
      <div key={item.objectID}>
        ...
      </div>
    );
  });
};
```

src/App.js

This holds also true for other functions, like the one we used in our JavaScript array's map method:

```
const List = () => {
  return list.map(item => {
    return (
      <div key={item.objectID}>
        <span>
          <a href={item.url}>{item.title}</a>
        </span>
        <span>{item.author}</span>
        <span>{item.num_comments}</span>
        <span>{item.points}</span>
      </div>
    );
  });
};
```

src/App.js

If an arrow function doesn't do *anything* in between, but only returns *something*, – in other words, if an arrow function doesn't perform any task, but only returns information --, you can remove the **block body** (curly braces) of the function. In a **concise body**, an **implicit return statement** is attached, so you can remove the return statement:

```
// with block body
count => {
  // perform any task in between
  return count + 1;
}


// with concise body
count =>
  count + 1;
```

src/App.js

This can be done for the App and List component as well, because they only return JSX and don't perform any task in between. Again it also applies to the arrow function that's used in the map function:

```
const App = () => (

  <div>
    ...
  </div>
);

const List = () =>
  list.map(item => (

    <div key={item.objectID}>
      <span>
        <a href={item.url}>{item.title}</a>
      </span>
      <span>{item.author}</span>
      <span>{item.num_comments}</span>
      <span>{item.points}</span>
    </div>

 ));
```

src/App.js

Our JSX is more concise now, as it omits the function statement, the curly braces, and the return statement. However, remember this is an optional step, and that it's acceptable to use normal functions instead of arrow functions and block bodies with curly braces for arrow functions over implicit returns. Sometimes block bodies will be necessary to introduce more business logic between function signature and return statement:

```
const App = () => {
  // perform any task in between

  return (
    <div>
      ...
    </div>
  );
};
```

src/App.js

Here is the fully working code to demonstrate the above concepts:

```
  □ □ □□  □   ã□  F   □□  □   □  )□     □   9□  5□  @@  □   °□  n□  □PNG
□
   IHDR   □   □□□   (-□S   äPLTE""""""""""""""""2PX=r□)7;*:>H□¤-BGE□□8do5Xb6[eK□®K□¯1MU9gs3S
□
   IHDR   □   □□□   ×©ÍÊ  □ePLTE""""""""""""""""""""""2RZN¢¹J□«3R[J□¬)59YÁÞ0KS4W`Q«ÄL□²%+-0JR
?^q÷ñíÛ□ï.},□ìsæÝ_TttÕ¾ □1#□□/(ì□-[□□□è`□è`Ì□ÚïÅðZ□d5□□□□?ÎebZ¿Þ□i.Ûæ□□□iqÎ□+1°□}Â□5ù  ïçd
□
   IHDR       □□   D¤□Æ  □APLTE   """"""""""""""""""""""""""2RZVºÖ_ÔôU·Ñ=r□$()'25]ÎíC□□0LS<o}X
□
   IHDR   @   @□□   □·□ì  □:PLTE   """"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
¢ßqÇ8Ù□´□mKË±mÆ¶mÛü·yi!è□ÎªYÏuë ÀÏ_Àï?i÷□ý+ò□□ÄA□|□ù{□□´?¿□_En□).□JËD¤<□
©¬¢Z\Ts©R*□(□  ¯©□J□□□□□u□X/□4J□9□¡5·DEµ4kÇ4□&i¥V4Ú□¡®Ð□□¯□vsf:àg,□¢èBC»î$¶□ºÍùî□□á□@□ô□I_
-ê>Û□º«¢XÕ¢î}ß¨ëÛÑ;□ÃöN´□ØvÅý□Î¸ÿ1 □ë×ÄO@&v/Äþ_□ö\ô□Ç\í.□□½+0□□;□□□!□fÊ□¦´Ó%Â JY·O□Â□'/Å]_□
```

Be sure to understand this refactoring concept, because we'll move quickly from arrow function components with and without block bodies as we go. Which one we use will depend on the requirements of the component.

Exercises:

- Confirm the changes from the last section.
- Read more about JavaScript arrow functions.
- Familiarize yourself with arrow functions with block body and return, and concise body without return.