**Declaring Functions & Syntax**

Below are 2 syntaxes that are valid for defining functional components in React, and they are largely interchangeable. Here's a brief overview of the differences:

**Arrow Functions (const MyFunctionalComp = () => {})**

1. **Concise Syntax**: Arrow functions can have a more concise syntax, which makes the code slightly shorter.
2. **Lexical this**: Arrow functions don't have their own **this**. They inherit **this** from the surrounding non-arrow function. This can be useful when you need to access **this** from an outer function inside your component, although this is less of an issue with functional components.
3. **Implicit Return**: Arrow functions allow implicit return when the function body is a single expression, which can make the component definition shorter.

**Function Declarations (function MyFunctionalComp() {})**

1. **Function Hoisting**: Function declarations are hoisted, meaning they are lifted to the top of their containing function or script, allowing them to be used before they appear in the code.
2. **Clearer Syntax**: For developers who are new to JavaScript or React, function declarations might be easier to understand because they are the traditional way to declare functions.
3. **Named Function in Stack Trace**: Function declarations can make debugging a bit easier since the function's name appears in the stack traces.

**In Summary**

In the context of a React functional component, the choice between using an arrow function or a function declaration is mostly a matter of style and preference. There are no definitive benefits for using one over the other for defining React functional components. Feel free to use whichever form you find more readable or convenient.