Point de contrôle Création d'applications React avec TypeScript

Je pose et décrit les étapes à suivre pour effectuer les changements de code Typescript.

Code 01 : Greeting

```
import React from 'react';

// Définir une interface pour les props
interface GreetingProps {
  name: string; // Le nom doit être une chaîne
}

// Composant fonctionnel typé
const Greeting: React.FC<GreetingProps> = ({ name }) => {
  return <div>Bonjour, {name} !</div>;
};

export default Greeting;
```

Code 02 : Counter

```
import React, { Component } from 'react';

// Définir une interface pour l'état du composant
interface CounterState {
  count: number; // Le compte doit être un nombre
}

// Classe de compteur typée
class Counter extends Component<{}, CounterState> {
  // Initialisation de l'état
  state: CounterState = {
    count: 0,
  };

  // Méthode pour incrémenter le compte
  increment = () => {
    this.setState({ count: this.state.count + 1 });
  };

  // Méthode de rendu
  render() {
    return (
```

```
      <div>
        <p>Count : {this.state.count}</p>
        <button onClick={this.increment}>Increment</button>
      </div>
    );
  }
}

export default Counter;
```

Résumé des Changements :

1- Changement d'extension : Changement de `.js` à `.tsx` pour signaler que le fichier contient JSX.
2- Définition des interfaces : Création d’interfaces pour typer les props dans le composant fonctionnel et l’état dans le composant de classe.
3- Utilisation des types dans les composants : Ajout de types appropriés aux props et à l’état pour garantir la sécurité des types et la lisibilité du code.