GABRIELE ROMANATO

Ereditarietà in TypeScript

TypeScript è un superset di JavaScript che introduce il supporto al typing statico e ai concetti di programmazione orientata agli oggetti, tra cui l'ereditarietà. In questo articolo vedremo come funziona l'ereditarietà in TypeScript e come può essere utilizzata per strutturare il codice in modo più modulare e riutilizzabile.

Classi e ereditarietà

In TypeScript, le classi possono estendere altre classi utilizzando la parola chiave `extends`. Questo meccanismo consente a una classe derivata di ereditare proprietà e metodi da una classe base.

```
class Animal {
  name: string;

  constructor(name: string) {
    this.name = name;
  }

  makeSound(): void {
    console.log("Suono generico");
  }
}

class Dog extends Animal {
  makeSound(): void {
    console.log("Bau bau");
  }
}

const myDog = new Dog("Fido");
myDog.makeSound(); // Output: "Bau bau"
```

Modificatori di accesso

TypeScript supporta i modificatori di accesso `public`, `private` e `protected` per controllare la visibilità dei membri della classe.

- **public**: accessibile ovunque
- **private**: accessibile solo all'interno della classe
- **protected**: accessibile all'interno della classe e delle sue sottoclassi

```
class Person {
  protected name: string;

  constructor(name: string) {
    this.name = name;
  }
}

class Student extends Person {
  showName(): void {
    console.log(this.name);
  }
}
```

Override dei metodi

Una sottoclasse può ridefinire un metodo della classe base per personalizzarne il comportamento. In TypeScript, ciò avviene semplicemente implementando un metodo con lo stesso nome della classe base.

```
class Vehicle {
  move(): void {
    console.log("Il veicolo si muove");
  }
}

class Car extends Vehicle {
  move(): void {
    console.log("L'auto sta guidando");
  }
}
```

Parola chiave super

La parola chiave `super` viene utilizzata per accedere al costruttore o ai metodi della classe base.

```
class Bird {
  fly(): void {
    console.log("Sto volando");
  }
}

class Eagle extends Bird {
  fly(): void {
    super.fly();
    console.log("a grande altezza");
  }
}
```

Conclusione

L'ereditarietà in TypeScript è uno strumento potente che consente di creare gerarchie di classi, condividere comportamento e favorire il riuso del codice. Comprendere il suo funzionamento è fondamentale per scrivere applicazioni scalabili e manutenibili.

Applicazioni Correlate

- ### JavaScript Password Mask

  Un esempio in JavaScript di mascheramento di una password con l'aggiunta della funzionalità di copia negli appunti.
  DockerDocker ComposeJavaScript

- **Python Placeholder Image**

  Applicazione sviluppata in Python con Flask per la creazione di immagini segnaposto.
  DockerDocker ComposePythonFlaskJavaScript

-

  Go Placeholder Image

  Applicazione in Go per la creazione di immagini segnaposto.
  DockerDocker ComposeGoJavaScript