

GABRIELE ROMANATO

Gestione dei parametri delle funzioni in TypeScript

TypeScript estende le funzionalità di JavaScript aggiungendo un sistema di tipi che consente di scrivere codice più sicuro e manutenibile. Una delle aree in cui TypeScript offre vantaggi significativi è la gestione dei parametri delle funzioni. Vediamo come si possono definire, tipizzare e gestire diversi tipi di parametri.

Parametri Tipizzati

In TypeScript, è possibile specificare il tipo di ogni parametro. Questo permette di prevenire errori in fase di compilazione.

```
function sum(a: number, b: number): number {  
    return a + b;  
}
```

Parametri Opzionali

I parametri opzionali si indicano con un punto interrogativo dopo il nome del parametro. Se non forniti, il loro valore sarà undefined.

```
function hello(name?: string): void {  
    console.log(`Hello ${name ?? 'guest'}!`);  
}
```

Parametri con Valori di Default

È possibile assegnare un valore di default ai parametri. Questo valore verrà usato se il parametro non viene fornito.

```
function increment(value: number, amount: number = 1): number {  
    return value + amount;  
}
```

Parametri Rest

I parametri rest permettono di passare un numero variabile di argomenti sotto forma di array.

```
function sumAll(...numbers: number[]): number {  
    return numbers.reduce((acc, n) => acc + n, 0);  
}
```

Tipizzazione delle Funzioni

Oltre a tipizzare i parametri, è buona pratica tipizzare anche il valore di ritorno delle funzioni.

```
const multiply = (a: number, b: number): number => {  
    return a * b;  
}
```

```
};
```

Conclusione

TypeScript rende la definizione e l'uso delle funzioni più robusto grazie alla tipizzazione statica. Usare correttamente parametri opzionali, valori di default e rest parameters permette di scrivere funzioni più flessibili e sicure.

Applicazioni Correlate

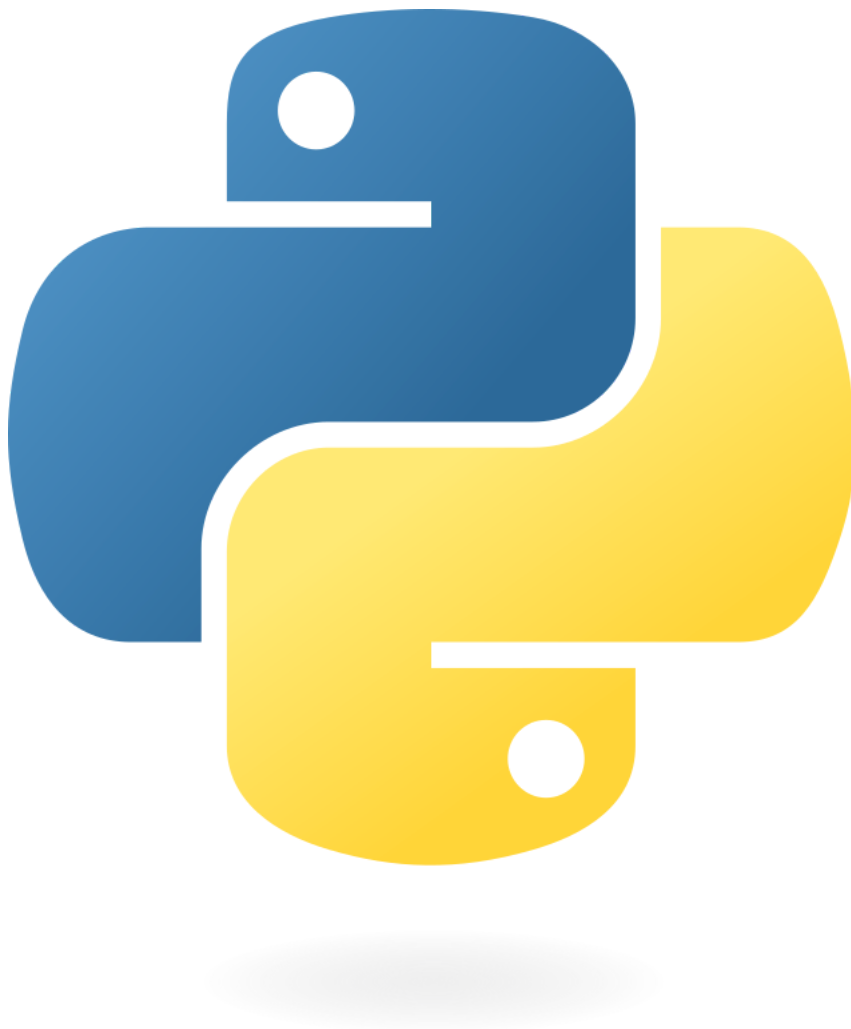


-

JavaScript Password Mask

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

[Docker](#)[Docker Compose](#)[JavaScript](#)



Python Placeholder Image

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

Docker Docker Compose Python Flask JavaScript



-

Go Placeholder Image

Applicazione in Go per la creazione di immagini segnaposto.

[Docker](#)[Docker Compose](#)[Go](#)[JavaScript](#)