Funktionale Programmierung mit TypeScript

Arrow Functions

Arrow Functions bieten eine kompakte Syntax und ein vorhersehbares `this`-Binding.

Promises und Asynchronität

Promises und `async/await` ermöglichen es, asynchrone Abläufe zu strukturieren.

Generics

Generics ermöglichen es, wiederverwendbaren und typsicheren Code zu schreiben, indem sie flexible Typen definieren, die sich dynamisch anpassen können.

Pure Functions

Pure Functions sind vorhersehbar und leichter zu testen, da sie keine unerwarteten Nebenwirkungen verursachen.

Immutability

Immutability bedeutet, Daten nicht direkt zu verändern, sondern neue Versionen zu erzeugen.

Functional Composition

Functional Composition hilft uns, Code modularer, lesbarer und wiederverwendbarer zu machen.

Higher Order Functions

Higher Order Functions machen Code flexibler und wiederverwendbarer, indem sie Funktionen als Argumente nehmen oder zurückgeben.

Currying

Currying wandelt Funktionen so um, dass sie ihre Argumente schrittweise entgegennehmen, was die Wiederverwendbarkeit, Flexibilität und Kompositionsmöglichkeiten verbessert.