

# Code-Organisation in TypeScript

---

Eine gute Code-Organisation ist essenziell, um Skalierbarkeit, Wartbarkeit und Teamarbeit zu erleichtern.

## Modularisierung

Ein Modul ist eine Datei, die bestimmte Funktionen oder Klassen kapselt und exportiert. Module ermöglichen eine saubere Trennung von Code und erleichtern die Wiederverwendbarkeit.

Bibliotheken sind Sammlungen von Modulen, die eine spezifische Funktionalität bereitstellen und über npm installiert werden können.

Pakete enthalten Bibliotheken oder eigene Module und werden über npm verwaltet. Eine `package.json` definiert Abhängigkeiten und Metadaten.

Im Frontend (z. B. mit React oder Angular) bestehen Komponenten aus mehreren Modulen, die UI und Logik kapseln.

## TypeScript-Modulsystem (ES6 Module)

TypeScript verwendet das ES6-Modulsystem, um Funktionen, Klassen oder Konstanten zwischen Dateien zu teilen.

- Named Exports
- Default Export

## Best Practices für Code-Organisation

- Sinnvolle Ordnerstruktur (z.B. `src/components/`, `src/modules/`, `src/utils/`)
- Klare Namenskonventionen (Vermeide generische Namen wie `helper.ts`, verwende stattdessen `dateUtils.ts`)
- Vermeidung zyklischer Abhängigkeiten
- Verwendung von Index-Dateien für saubere Exporte
- Verwendung von Aliases für saubere Importe (Einstellung in `tsconfig.json`)