TYPESCRIPT

Hello

Kamil Richert

Senior Software Engineer at Atlassian

TYPESCRIPT

TypeScript rozszerza JavaScript o możliwość typowania. Dzięki czemu jesteśmy w stanie wyłapać więcej błędów zanim oprogramowanie trafi na produkcję.

Co nam daje TypeScript?

1. Ułatwia kontrolę nad aplikacją
2. Większa czytelność kodu
3. Wymusza mniejsze funkcje
4. Podpowiedzi w edytorze kodu
5. Sprawdza poprawność typów podczas kompilacji
6. Każdy kod JS jest prawidłowym kodem TS
7. TS finalnie jest kompilowany do JS

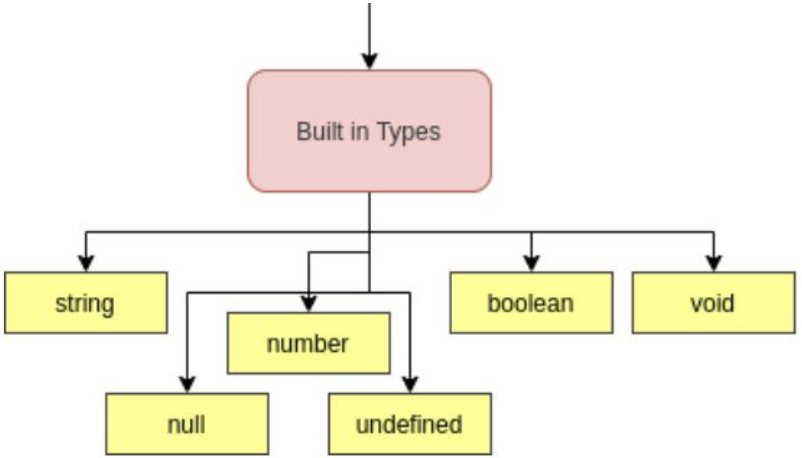
Jakie narzuca utrudnienia

1. Zwalnia czas wydawania oprogramowania
2. Dodatkowa konfiguracja przy starcie projektu

Podstawowe typy

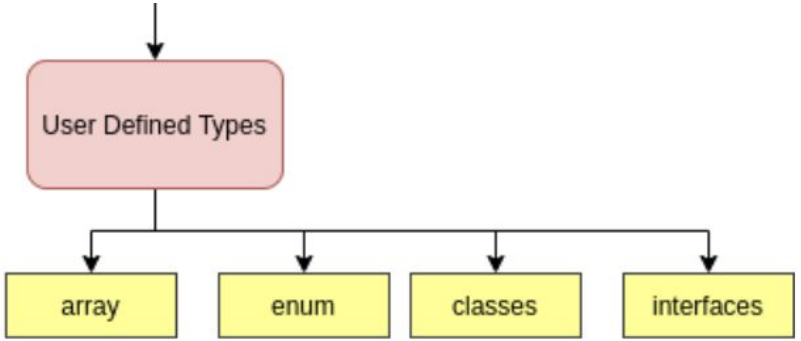
- string
- number
- boolean
- void
- null
- undefined

- never
- any
- unknown



Typy złożone

- array
- enum
- classes
- interfaces



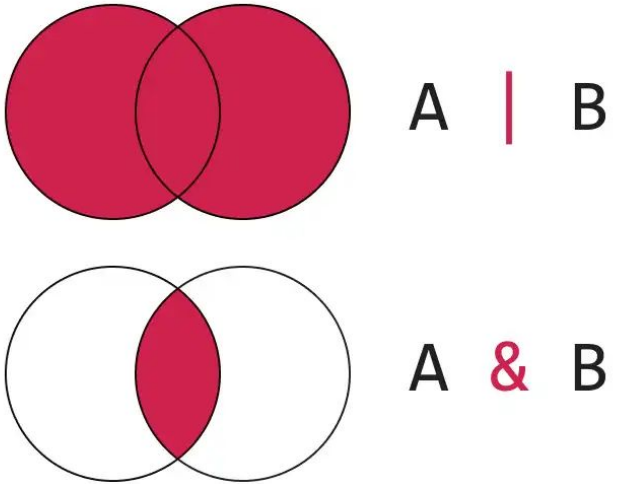
Union & Intersection

Union

type C = type A | type B

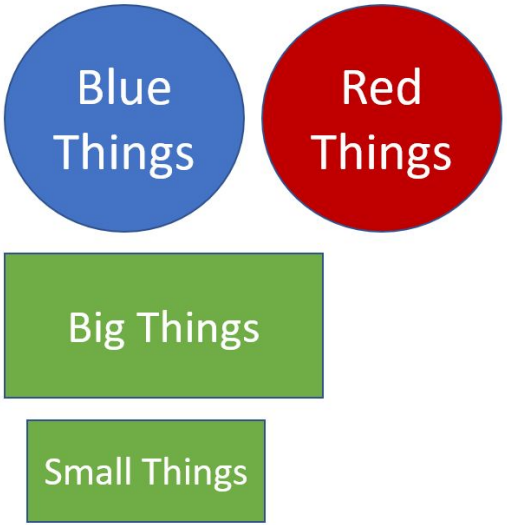
Intersection

type C = type A & type B

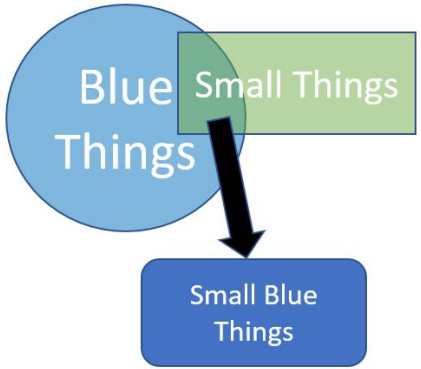


Union & Intersection

Consider classifying objects four ways: blue, red, big, and small

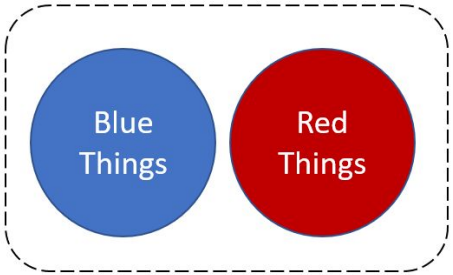


If we *intersect* **blue** with **small**, we get a new set:



The *intersection* of these sets has the *union* of its properties

If we *union* **blue** with **red**, we get a new set:



Blue or Red Things

The *union* of these sets has the *intersection* of its properties, which in this case is empty

Typy generyczne

Jeżeli chcemy stworzyć typ , który ma być reużywalny, ale zmienia się mna przykład 1 element to może wykorzystać generyczny typ.

```
identity<Type>(arg: Type): Type {
 return arg;
}
```

Narzędzia typowania

Required
Partial
Pick
Omit
…

https://www.typescriptlang.org/docs/handbook/utility-types.html

```
type PartialCars = Partial<Cars>;

type RequiredUser = Required<User>;

type PickCars = Pick<Cars, 'model'>

type OmitCars = Omit<Cars, 'id'>
```

Dzięki

Znajdziecie mnie:
https://www.linkedin.com/in/kamil-richert/
https://github.com/krichert