

TypeScript Basics — шпаргалка для начинающего

Основные понятия

Понятие	Описание	Пример

Переменные с типами	В TS указываем тип данных переменной после :	<code>let age: number = 25;</code>	
Типы данных	<code>number, string, boolean, any, unknown, void, null, undefined, never</code>	<code>let isActive: boolean = true;</code>	
Массивы	Указываем тип элементов массива: <code>тип[]</code>	<code>let numbers: number[] = [1, 2, 3];</code>	
Кортежи (Tuple)	Массив с фиксированным набором типов	<code>let person: [string, number] = ['Tom', 30];</code>	
Объекты	Указываем типы свойств	<code>let user: { name: string, age: number } = { name: 'Alex', age: 25 };</code>	
Тип any	Отключает проверку типа, можно присвоить что угодно	<code>let data: any = 'text'; data = 123;</code>	
Функции с типами	Указываем типы аргументов и возвращаемого значения	<code>function sum(a: number, b: number): number { return a + b; }</code>	
Необязательные параметры	Параметр с ? может быть пропущен	<code>function greet(name?: string) { console.log(name); }</code>	

**Типы **	Значение может быть нескольких типов	<code>let id: string</code>	<code>number = 123;</code>
Интерфейсы (interface)	Описывают структуру объекта	<code>interface User { name: string; age: number; }</code>	
Типы (type)	То же, что <code>interface</code> , но может быть объединён с примитивами	<code>type ID = string</code>	<code>number;</code>

📖 Частые вопросы

? Чем отличается `type` от `interface`?

<code>type</code>	<code>interface</code>
Может объединять примитивы	Только объектные типы
Нельзя переопределить	Можно расширить (extends)
Используют для простых типов	Используют для объектов

? Как работает `union`?

ts

Копировать

Редактировать

```

let value: string | number;
value = 'Hello'; // ✓
value = 42;      // ✓
value = true;    // ✗ Ошибка


```

`|` — это "или", значит переменная может быть строкой ИЛИ числом.


? Что такое `void` и `never`?

Тип	Описание	Пример
<code>void</code>	Функция ничего не возвращает	<code>function log(): void { console.log('Hi'); }</code>

<code>never</code>	Функция никогда не завершится нормально	<code>function error(): never { throw new Error('Oops'); }</code>
--------------------	---	---

 Типичная ошибка новичка

`let age: number = '25';` // ❌ Ошибка: строка вместо числа

 Совет: Пиши код так

✅ Хорошо:

```
function getUsername(user: { name: string }): string {
  return user.name;
}
```

❌ Плохо:

```
function getUsername(user: any) {
  return user.name;
}
```

 Финальное напоминание перед тестом

Что помнить	Почему это важно
Типы — везде!	Без типов теряется смысл TS
Интерфейсы и типы различай	Интерфейсы — объекты, типы — универсальны
Проверяй себя	Если непонятно — пиши тип <code>any</code> , но лучше не злоупотреблять
Функции с типами всегда	И аргументы, и результат