- **Модуль:** Typescript. Необходимый уровень
- **Урок:** Механизм вывода типов (Type Inference)

TypeScript Handbook
from
Иван Петриченко
udemy

ВЫВОД ТИПОВ

При написании кода TS постоянно анализирует его и следит за тем, чтобы все операции были совместимы, не было синтаксических или логических ошибок.
Это происходит даже тогда, когда вы не указываете аннотации. Именно поэтому при наведении на переменную **a** в коде:

```
let a = 'string';
```

Вы увидите её тип. Этот механизм называется **вывод типов**. И раз он работает все время, то на него можно переложить часть рутинной работы разработчика. Предупреждения об ошибках и autocomplete будут продолжать работать

В различных источниках вы можете встретить такой совет:

Не нужно явно указывать типы, если за вас это может сделать вывод типов.

Но я добавлю два момента, которые нужно учитывать во время реальной работы:

- 👉 Соблюдение семантики кода. Код **должен быть понятен** для вас и других разработчиков, даже спустя время. Аннотации могут помочь в этом, не смотря на то, что TS понимает, с чем работает
- 👉 Разные стилистические правила в проектах и компаниях. Если сказано указывать типы - указываем их

ИСКЛЮЧЕНИЯ ИЗ ПРАВИЛ

Вывод типов срабатывает не всегда. Есть ситуации, когда он не понимает, что будет находится там. Тут стоит использовать аннотации:

👉 Создание пустой переменной. Даже после помещения значения тип у переменной останется any:

```
let salary;
salary = 500;
```

Переменная так и останется типа any

```
let salary: number;
salary = 500;
```

Указание аннотации избавит от проблем

👉 Функции, которые возвращают any (JSON.parse() и тп.)

👉 Конфликт вывода типов с вашей логикой кода:

```
const isOkay = true;
let movement = false;

if (isOkay) {
    movement = "moving";
}
```

Ошибка, строка не может быть помещена в boolean

```
const isOkay = true;
let movement: boolean | string = false;

if (isOkay) {
    movement = "moving";
}
```

Указание аннотации с union типом избавляет от проблемы

Обратите внимание! Если создавать переменные с примитивными литералами через **const**, то **вывод типов** определит их как литерал. Если через **let** - то как общий тип (string, number, boolean...)