- **Модуль:** Typescript. Необходимый уровень
- **Урок:** Enums

TypeScript Handbook
from
Иван Петриченко
udemy

ПЕРЕЧИСЛЕНИЯ В TYPESCRIPT

В разработке бывают ситуации, когда что-то ограничено перечислением нескольких вариантов. И программа выбирает один из них. Например:

- Варианты движения объекта: *вверх, вниз, влево, вправо*
- Список доступных валют: *€, $, ₴*
- Список доступных анимаций: *fadeIn, fadeDown, swipeLeft, swipeRight*

И мы хотим предоставить ей и разработчикам только ограниченный выбор. Для этого используются **перечисления (Enum)**
Эта структура существует **только внутри TS.** Она позволяет избегать опечаток и применения сторонних вариантов:

```ts
enum Directions {
    TOP,
    RIGHT,
    LEFT,
    BOTTOM,
}
```

Стандартный enum (числовой)
Автоматическая нумерация значений

```ts
enum TimingFunc {
    EASE = "ease",
    EASE_IN = "ease-in",
    LINEAR = "linear",
}
```

Строковый enum

```ts
enum TimingFunc {
    EASE = 1,
    EASE_IN = 2,
    LINEAR = 10
}
```

Числовой enum
с установленными значениями

```ts
function frame(elem: string, dir: Directions, tFunc: TimingFunc): void {
    if (dir === Directions.RIGHT) {
        console.log(tFunc);
    }
}

frame("id", Directions.RIGHT, TimingFunc.LINEAR);
```

Применение перечисления в функции
Enum ограничивает варианты и не дает опечататься

Существует вариант гетерогенного enum’a - это **комбинация строковых и числовых значений.** Но в реальности использовать такое **не стоит.**
Лучше задуматься об рефакторинге кода в таком случае.

В строковых перечислениях нельзя заниматься вычислениями для получения новых значений. В числовых - можно:

```ts
enum TimingFunc {
    EASE = 1,
    EASE_IN = 2,
    LINEAR = EASE * 2,
}
```

ПЕРЕЧИСЛЕНИЯ ПОСЛЕ КОМПИЛЯЦИИ

После преобразования .ts файлов в .js перечисления превращаются в **функции** с вычислением и записью значений:

```ts
enum Directions {
    TOP,
    RIGHT,
    LEFT,
    BOTTOM,
}
```

→

```js
var Directions;
(function (Directions) {
    Directions[Directions["TOP"] = 0] = "TOP";
    Directions[Directions["RIGHT"] = 1] = "RIGHT";
    Directions[Directions["LEFT"] = 2] = "LEFT";
    Directions[Directions["BOTTOM"] = 3] = "BOTTOM";
})(Directions || (Directions = {}));
```

Есть вариант немного ускорить их работу в конечном коде, если вычисления внутри enum занимают какое-то время. Для этого используются константные перечисления:

```ts
const enum TimingFunc {
    EASE = "ease",
    EASE_IN = "ease-in",
    LINEAR = "linear",
}
```

После компиляции они не формируют функции. Компилятор сделал все вычисления, нашел все ссылки на это перечисление и заменил их значениями в конечном коде. Остаются лишь комментарии где это произошло

Этот прием имеет свои подводные камни и может приводить к багам. Так что почти всегда вы будете встречать стандартный вариант. Лучше пожертвовать несколькими миллисекундами, но не получать ошибок