PHP 8 in a Nutshell

Amit D. Merchant

amitmerchant.com

Table of Contents

Typed Constants

The perpetual longing of making the type system of PHP more robust is still going on. And going in that direction, PHP 8.3 has introduced typed constants.

Essentially, up until now, you could not specify the type of constants. But with PHP 8.3, it won't be the case anymore.

So, from PHP 8.3, you would be able to specify a type to class, interface, trait, as well as enum constants.

Here are some examples of how you can use typed constants.

```
enum Car
{
    const string NAME = "Car"; // Car::NAME is a string
}

trait Base
{
    const string NAME = "Base"; // Base::NAME is a string
}

interface Adapter
{
    const string NAME = Car::NAME;   // Adapter::NAME is a string as
well
}

class Audi implements Adapter
{
    use Base;

    const string NAME = Car::NAME;  // Foo::TEST must also be a string
}

class Tesla extends Audi
{
    const string NAME = "Model X";  // Tesla::NAME must also be a
string, but the value can change
}
```

Constant values have to match the type of the class constant. In case it does not, a `TypeError` will be thrown.

Apart from this, like all the other type checks (property types), constant type checks are always performed in the strict mode.

While some may argue that why should constants have types? Well, in my opinion, typed constants are not necessary since constants are immutable by default. But by making them typed, it improves the code readability and makes the code self-documenting.