The last part is to use the cast in the model:

```php
protected $casts = [
  'revenue' => MillionsCast::class,
  'net_profit' => MillionsCast::class,
];
```

So, in a nutshell, this is how you use a value object. To summarize it:

By using value objects, you can make objects from cohesive scalar data

The main benefits:

- It makes your code more high-level.
- It clarifies things and helps to avoid confusion. For example, now you know exactly that `Millions` contains a number stored in millions.
- It helps you deal with nullable values. You don't have to write `?float $revenue` anymore. You can write `Millions $revenue`.

In the introduction, I wrote that data is a crucial part of any application. I gave you this list:

- The request comes in. It contains the incoming data.
- **The business layer processes this data.**
- **The database layer inserts this data into the DB.**
- The response comes out. It includes the outgoing data.

As you can see in the cast and the other examples, a value object is mainly used inside (but not exclusively!) our application. In the next chapter, we'll discuss what happens at the boundaries (requests and responses).