06

Para saber mais: preciso manter a propriedade protected?

Para saber mais: preciso de protected mesmo?

Vejamos a classe `View` que criamos neste capítulo:

```
abstract class View<T> {

    protected _elemento: Element;

    constructor(seletor: string) {

        this._elemento = document.querySelector(seletor);
    }

    update(model: T) {

        this._elemento.innerHTML = this.template(model);
    }

    abstract template(model: T): string;

}
```

Usamos o modificador `protected` logo no início para permitir que os métodos `update` definidos nas classes `filhas` pudessem acessar a propriedade. Contudo, melhoramos o design de nossa classe e conseguimos fazer com que as classes filhas também herdassem o método update de `View` . Nesse contexto, o acesso à propriedade `_elemento` é feito por um método da classe pai, sendo assim, podemos voltar com o modificador `private` em nossa classe, evitando assim o relaxamento do seu encapsulamento:

```
abstract class View<T> {

    // voltando com private!

    private _elemento: Element;

    constructor(seletor: string) {

        this._elemento = document.querySelector(seletor);
    }

    update(model: T) {

        this._elemento.innerHTML = this.template(model);
    }

    abstract template(model: T): string;

}
```

Era essa ideia que este instrutor gostaria ter passado para vocês quando disse no vídeo que haveria outra solução para acessar a propriedade `_elemento` sem a necessidade do uso do modificador `protected`. Mas foi tanta emoção que deixei de fazer essa pequena alteração.