Паттерны проектирования: паттерн Шаблонного метода в TypeScript

Добро пожаловать в серию «Паттерны проектирования в TypeScript», в которой будет представлен ряд полезных паттернов проектирования в веб-разработке с использованием TypeScript.

Предыдущие статьи следующие:

Паттерн Стратегия в TypeScript

Паттерн Цепочка Обязанностей в TypeScript

Паттерн Наблюдатель в TypeScript

Паттерн Шаблонного метода в TypeScript

Паттерн Адаптер в TypeScript

Паттерн Фабричного метода в TypeScript

Паттерн Абстрактной фабрики в TypeScript

Паттерны проектирования очень важны для веб-разработчиков, которые освоив паттерны становятся способными улучшить качество написания кода. В этой статье я буду использовать TypeScript, чтобы рассказать о паттерне Шаблонного метод.

CSV (значения, разделенные запятыми) — это относительно простой формат файла с данными. Файлы CSV хранят табличные данные (числа и текст) в виде обычного текста. Процесс обработки данных в формате CSV показан на следующем рисунке:

Разобравшись с описанным выше процессом обработки, давайте воспользуемся Node.js для реализации функции анализа CSV-файлов.

users.csv

parse-csv.ts

В приведенном выше коде мы импортируем модуль d3-dsv для реализации функции синтаксического анализа csv. После этого мы используем esno для запуска выполнения файла parse-cvs.ts:

Когда вы запустите приведенный выше код, на терминале будет получен следующий результат, показанный ниже на рисунке:

Markdown — это малотребовательный язык разметки, который позволяет писать документы в простом текстовом формате, который легко читать и писать. Но для того, чтобы отобразить документы Markdown на веб-страницах, мы должны преобразовать Markdown-документ в HTML-документ.

Чтобы осуществить вышеописанный процесс построим процедуру обработки следующим образом:

Разобравшись с описанной выше процедурой, давайте воспользуемся Node.js для реализации функции парсинга файлов Markdown.

Users.md

parse-md.ts

В приведенном выше коде мы импортируем отмеченный модуль для реализации функции синтаксического анализа файла Markdown. После этого используем esno для выполнения файла parse-md.ts:

После выполнения вышеуказанного кода терминал выведет следующий результат:

Несмотря на то, что в предыдущих двух примерах анализируются файлы разных типов, сам процесс анализа файлов аналогичен.

Весь процесс состоит из трех основных шагов: чтение файла, парсинг файла и обработка данных. Для этой процедуры обработки файлов мы можем ввести паттерн Шаблонного метода, чтобы инкапсулировать последовательность обработки, состоящую из трех вышеуказанных шагов.

Паттерн Шаблонного метода состоит из двух частей: абстрактного родительского класса и конкретного подкласса реализации. Обычно структура алгоритма подкласса инкапсулируется в абстрактный родительский класс, который также включает в себя реализацию некоторых общедоступных методов и порядок выполнения всех методов в инкапсулированном подклассе. Наследуя этот абстрактный класс, подклассы также наследуют всю структуру алгоритма и могут переопределить методы родительского класса.

Теперь давайте посмотрим, как реализовать анализатор CSV и анализатор Markdown с использованием паттерна Шаблонного метода.

Чтобы лучше понять следующий код, давайте рассмотрим диаграмму UML, описывающую взаимные связи между классами :

На приведенном выше рисунке мы определяем абстрактный класс FileParser, а затем определяем два подкласса реализации - CsvParser и MarkdownParser.

Класс FileParser

Метод parse в абстрактном классе FileParser — это так называемый шаблонный метод, в который мы будем инкапсулировать один из процессов обработки файла.

Класс CsvParser

Класс MarkdownParser

С помощью этих двух классов CsvParser и MarkdownParser мы можем анализировать файлы CSV и Markdown следующим кодом:

После запуска этого кода вы получите следующий результат:

Используя паттерн Шаблонного метода, мы переиспользуем код парсинга файлов CSV и Markdown. Более того, с помощью абстрактного класса FileParser мы можем легко разрабатывать новые различные парсеры для других типов файлов.

Итак, давайте подытожим сценарии использования паттерна Шаблонного метода:

Если общие шаги алгоритма постоянны, но некоторые части (шаги) алгоритма являются различными по своей внутренней реализации, то паттерн Шаблонного метода удобно использовать для абстрагирования от таких варьируемых частей алгоритма для реализации внутри отдельных подклассов.

Если у вас есть какие-либо вопросы, пожалуйста, пишите мне. В дальнейшем я продолжу знакомить вас с другими паттернами, и если вам интересно, подпишитесь на меня в Medium или Twitter.