

ВИКОРИСТАННЯ INCREMENTAL SOURCE GENERATORS ДЛЯ ГЕНЕРАЦІЇ ВИХІДНОГО КОДУ В .NET

Вінницький національний технічний університет

Анотація

Розглянуто інструмент Incremental Source Generators для генерації коду в .NET, який дозволяє виконувати генерацію лише при зміні певних вхідних даних. Розглянуто особливості роботи інструменту та типові сценарії використання.

Ключові слова: .NET, метапрограмування, Incremental Source Generators, генерація вихідного коду.

Abstract

Article features a review of the Incremental Source Generators tool for generating source code in .NET, which allows generation to be performed only when certain input data is changed. The features of the tool and typical use cases are reviewed.

Keywords: .NET, metaprogramming, Incremental Source Generators, source code generation.

Вступ

Генерація програмного коду є ключовою частиною метапрограмування, адже дозволяє описувати інструкції для написання інших інструкцій [1]. Генерацію коду можна здійснювати шляхом використання сторонніх інструментів. Як правило, ці інструменти дозволяють використовувати скрипти або шаблони для генерації коду. Проте не завжди сторонній інструмент має повноцінний доступ до синтаксичного дерева оригінального вихідного коду.

У випадку з платформою .NET та мовою програмування C# довгий час основним інструментом для генерації коду був T4 Templates [2], що дозволяв описувати генерацію коду шляхом використання шаблонів. Проте, починаючи з .NET5, з'явилася можливість вбудовувати генерацію коду в процес компіляції та аналізу за допомогою інструменту Source Generators [3]. Ключовою особливістю цього інструменту є те, що він виступає в якості аналізатора коду, що дає йому повний доступ до всього синтаксичного дерева та інших метаданих. Це дозволяє легко генерувати новий код, опираючись на існуючий. Проте, оскільки в якості вхідних даних Source Generators використовує все синтаксичне дерево, то при будь-яких змінах відбуватиметься процес генерації. Це має значний вплив на швидкість роботи IDE, адже Source Generators працюють також і під час розробки. Для вирішення цієї проблеми можна використовувати Incremental Source Generators.

Генерація вихідного коду з використанням Incremental Source Generators

Source Generators в якості вхідних даних може використовувати низку джерел, починаючи від синтаксичного дерева, закінчуючи сторонніми параметрами проєкту. Проте, основним джерелом є саме *SyntaxProvider*, що надає доступ до інформації про синтаксичне дерево. Це стосується й Incremental Source Generators. Проте, на відміну від звичайних генераторів, інкрементний генератор використовує механізм кешування в процесі генерації [4].

Для цього в Incremental Source Generators використовується конвеєр перетворень (див. рисунок 1), що описує процес вибору та перетворень елементів вхідних даних. Такий конвеєр може складатися з кількох етапів, кожен з яких описує певну трансформацію чи фільтрацію даних. Сам процес кешування вбудований у цей конвеєр перетворень, адже результат кожного етапу конвеєра кешується [5]. Це дозволяє при повторному виклику генератора порівнювати результати кожного етапу та визначати, чи продовжувати трансформацію й генерацію для конкретного елемента даних, чи використовувати вже обрахований результат.

Таким чином, якщо в результаті низки перетворень певний елемент даних буде аналогічним до такого, який зберігається в кеші, то повторна генерація коду не відбудеться. Це дозволяє забезпечити генерацію коду лише у випадках, коли відбулися зміни відносно попередньої генерації.

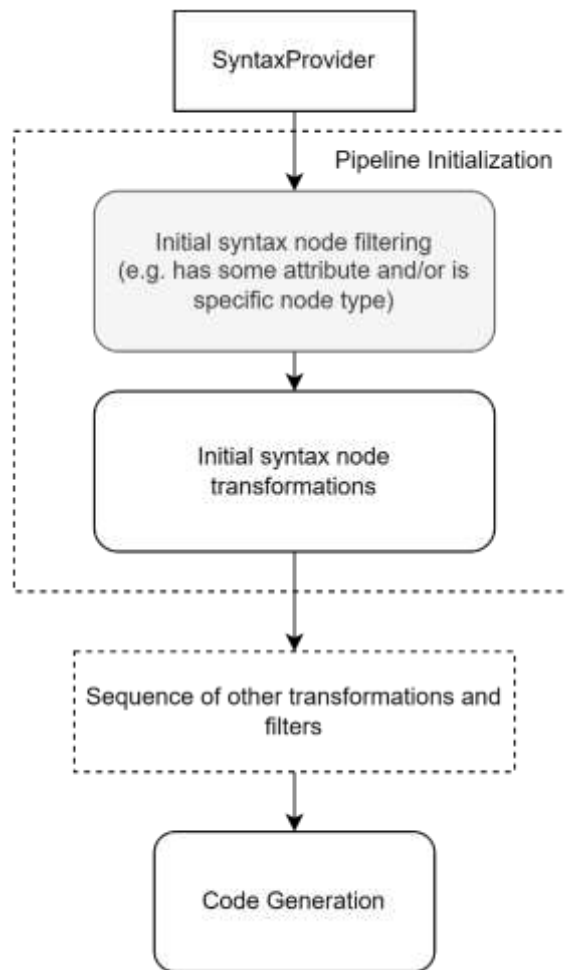


Рисунок 1 – Загальна схема конвеєра перетворень інкрементного генератора коду

Зі схеми на рисунку 1 видно, що конвеєр перетворень, який базується на використанні *SyntaxProvider*, має три обов’язкових етапи: першочерговий відбір, трансформація та генерація коду. Першочергові відбір та трансформація є обов’язковою частиною процесу ініціалізації конвеєра. Як правило, на цих етапах відбираються синтаксичні вузли відповідно до їх типу або наявності певного атрибуту метаданих. Також різні конвеєри можна об’єднувати між собою, цим самим додаючи додаткову інформацію з інших джерел даних.

У ході роботи з *Incremental Source Generators* важливо розуміти, що для коректного кешування необхідно, щоб правильно виконувалося порівняння елементів. Для цього потрібно враховувати особливості порівняння типів даних у .NET. Наприклад, якщо в якості даних виступають посилальні типи, то за замовчуванням вони порівнюватимуться за посиланнями, а не за вмістом. Для уникнення цієї проблеми варто визначати компаратор (*IEqualityComparer<T>*), що використовуватиметься конвеєром на певному етапі.

Висновок

Інкрементні генератори пропонують використання більш оптимального підходу до генерації коду в порівнянні зі звичайними *Source Generators*. Використання цього інструменту дозволить розробляти алгоритми генерації коду, що в якості вхідних даних використовуватимуть синтаксичне дерево. Інкрементний підхід дозволить забезпечити генерацію лише при змінах у відповідних частинах коду.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Ingebrigtsen E. *Metaprogramming in C#: Automate your .NET development and simplify overcomplicated code* / Einar Ingebrigtsen. – Birmingham, 2023. – 352 с. – (Packt Publishing).
2. Hazzard K. *Metaprogramming in .NET* / K. Hazzard, J. Brock. – New York, 2013. – 360 с. – (Manning).

3. Introducing C# Source Generators [Електронний ресурс] / Phillip Carter. – 2020. – Режим доступу до ресурсу: <https://devblogs.microsoft.com/dotnet/introducing-c-source-generators/>.
4. Dotnet Source Generators in 2024 [Електронний ресурс] / Jonathan Owens. – 2024. – Режим доступу до ресурсу: <https://posts.specterops.io/dotnet-source-generators-in-2024-part-1-getting-started-76d619b633f5>
5. Incremental Generators [Електронний ресурс]. – 2024. – Режим доступу до ресурсу: <https://github.com/dotnet/roslyn/blob/main/docs/features/incremental-generators.md>

Войтко Вікторія Володимирівна – кандидат технічних наук, доцент кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: dekanfki@i.ua.

Позур Михайло Юрійович – аспірант, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: mixalchik545@gmail.com

Viktoriia Voitko – Ph.D., Associate Professor of Software Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: dekanfki@i.ua.

Mykhailo Pozur – Ph.D. student, Faculty for Information Technologies and Computer Engineering, Vinnytsia National Technical University, Ukraine.