СЛОВАРИ КАК КОНТЕЙНЕР ДЛЯ ХРАНЕНИЯ МЕТОДОВ В ЯЗЫКЕ C#

Никитин Д.А.
Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь
Рябычина О.П. – канд.техн.наук, доцент, доцент кафедры ИРТ

В статье представлена информация о использовании словарей в качестве механизма выбора в языке программирования C#, определен один из вариантов реализации такого механизма. Проведено сравнение предложенного механизма выбора с оператором выбора switch.

Нередко при разработке программных средств встречается проблема создания определенного набора программных команд, которые должны выполняться при выполнении запроса пользователя. Здесь рассматривается ситуация, когда пользователь вручную вводит текст команды и на основании этого текста программа должна выполнить определенные действия. В более абстрактном представлении задача решается внедрением в программу конструкции выбора. Во многих языках, в том числе и в рассматриваемом языке C#, присутствует оператор выбора switch, который позволяет решить данную проблему, однако при рассмотрении ситуации с наличием большого количества вариантов выбора эта конструкция будет громоздкой.

Альтернативным вариантом оператора выбора может выступать структура данных «Словарь», которая аналогична хеш-таблице и ассоциативному массиву. Она представляет собой два поля:

1. ключ – уникальный идентификатор для обращения к значению;
2. значение – основное содержимое в наборе данных.

В языке C# ключом и значением словаря могут выступать различные типы данных, в том числе и сложные структуры. Тогда для создания оператора выбора из словаря нужно чтобы тип данных значения являлся объектом, который содержит функцию.

В качестве типа данных для значений словаря выбран класс «Task», он позволяет хранить внутри себя методы возвращающие различные типы данных и позволяет выполнять отложенный запуск этих методов.

Используя словари в таком варианте, достаточно удобно будет провести разделение процессов добавления элементов в словарь, создание методов для добавления и их вызов на отдельные слои.

Так как словарь – это тип данных, то его можно использовать на разных уровнях реализации, он может находиться как на главном слое программы, так и скрыт внутри класса. Благодаря обобщениям можно конструировать объекты различной сложности и внедрять их в словарь.

Одной из простейших реализаций такого механизма выбора выступает создание класса функции, который имеет два поля: имя функции и объект типа «Task», в основном слое программы создаются экземпляры этого класса, объекты класса «Task» можно создавать при вызове конструктора класса функции, благодаря лямбда-выражениям. На этом же уровне создается словарь, который принимает в качестве ключа строку, а в качестве значения объект класса «Task».

При возникновении необходимости возврата значений необходимо установить обобщения, как для объектов Task внутри класса функции, так и при создании словаря.

Необходимо проверить скорость работы и объемы занимаемой память, ведь конечному пользователю не важно, как написан программный продукт – важно, что он работает быстро с минимальным использованием ресурсов.

Для сравнения времени выполнения нет необходимости создавать миллион вариантов выбора, ведь оба механизма имеют временную сложность O(1). Для проверки достаточно установить по одному варианту выполнения и алгоритм, который будет выполняться внутри этих механизмов, должен быть одинаковым. В качестве такого алгоритма выбран вывод простого сообщения на консоль. В качестве таймера выбран объект «Stopwatch» из пространства имен «System.Diagnostics». Результат теста показал, что конструкция с оператором switch выполнилась за две сотые секунды, в то время как выполнение конструкции с использованием словаря приблизительно составляет одну тысячную секунды, что в разы превосходит конструкцию switch.

Теперь нужно проверить количество памяти занимаемое каждой из конструкции. Для этого было подготовлено 3000 вариантов выбора для обоих конструкций. Проверка выполнялась при создании Snapshot в среде разработки «Visual studio». Результат теста показал, что оба варианта занимают в памяти 24 байта.

В ходе исследования был рассмотрен механизм оператора выбора с использованием словарей. Определено, что вариантов реализации с данной структурой можно придумать множество, что нельзя сказать про оператор switch. Это свидетельствует о гибкости словарей, как с точки зрения структуры данных, так и со стороны оператора выбора. Выявлено, что словари и оператор switch в равной степени используют выделенную им память. Также следует отметить, что время выбора у обоих O(1), но словари обрабатывают информацию быстрее.

Список использованных источников:

1. Коллекция Dictionary [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/tutorial/4.9.php. – Дата доступа: 04.04.2023.