

Задание #2

17 мая 2020 г.

Генератор код

Проект представляет собой библиотеку для генерации кода, объявляющего классы и их методы на языках C++, Java и C#.

Для этого пользователю предоставлен класс `CodeFactory`, задаваемый двумя аргументами:

1. `language`

Код языка(для удобства представлен как `enum`, при вводе неверного кода программа прервется с исключением)

2. `writer`

Указатель на объект отвечающий за вывод результата генерации (При значении `nullptr` объект самостоятельно создаст объект выводящий результат на консоль `ConsoleWriter`)

`CodeFactory` имеет методы:

1. `AddClassUnit(name)`

Создает класс с именем `name`. Если класс с таким именем уже существует, он будет перезаписан на пустой.

2. `AddMethodUnit(className, accessType, name, returnType, flags)` Добавляет классу с именем `className` метод с именем `name`, возвращаемым значением `returnType` и флагом `flags` (пр. `static`, `const`), метод будет иметь тип доступа `accessType`.

Если класс с заданным именем не существует или флаги имеют некорректные значения, будет вызвано исключение.

3. `AddPrintComand(className, methodName)`

Добавляет методу с именем `methodName` класса с именем `className` команду на вывод строки "Hello world!" в консоль.

Если метод или класс не найдены, будет вызвано исключение.

Проект имеет 3 unit-теста (по одному на каждый язык), в файле `main` производится их запуск. Данные тесты можно использовать как примеры использования библиотеки.

Writers:

Библиотека поддерживает несколько типов вывода:

1. Консольный вывод

Класс `ConsoleWriter` отвечает за вывод сообщений в консоль

2. Файловый вывод

Класс `FileWriter` записывает сообщения в файл, лежащий по заданному адресу.

3. Векторный вывод

Класс `VectorWriter` вместо вывода сообщений куда-либо, сохраняет их в вектор строк `output`. В результате мы можем программно обрабатывать получаемый нами код. Данный способ вывода используется в `unit-тестах`.

Пример использования библиотеки:

```
auto consoleWriter = ConsoleWriter();
auto factory = CodeFactory(CodeFactory::cpp, consoleWriter);
factory.AddClassUnit("TestClass");
factory.AddMethodUnit("TestClass", 2, "testMethod", "int", 2);
factory.AddPrintComand("TestClass", "TestClass");
factory.Compile();
```

Ожидаемый вывод:

```
class TestClass
{
protected:
    int TestMethod()
    {
        std::cout << "Hello world!\n" << std::endl;
    }
}
```