Общие положения

1. Настоящая инструкция выпущена с целью обеспечить разработчика необходимой информацией для пополнения библиотеки плагинов программы Cinema.
2. Описывает порядок:

   - создания и компиляции плагина в среде Microsoft Visual Studio на языке программирования C++;
   - создания и компиляции плагина в среде Embarcadero Studio IDE на языке программирования Pascal;
   - регистрации плагина в “Plugins.json” для последующего использования в сценариях программы Cinema.

3. Предназначена для специалистов, занимающихся разработкой, модификацией и созданием новых сценариев в программе Cinema.

Создание и компиляция плагинов

1. Перед началом работы необходимо изучить настоящую инструкцию, в случае возникновения вопросов, связаться с отделом технической поддержки программы Cinema. До разъяснения возникших вопросов не следует приступать к пополнению библиотеки новыми плагинами.
2. Алгоритм работы ядра и принципы разработки плагинов описаны в приложении 1, 2 настоящей инструкции.
3. Для создания плагина в среде Microsoft Visual Studio на языке программирования C++:

   - Открыть файл проекта Template.vcxproj из библиотеки Template, расположенной по адресу:

Doc\SDK\Examples\C++\Template\Template.vcxproj.

Проект содержит пример реализации плагина. (См. рисунок 1).

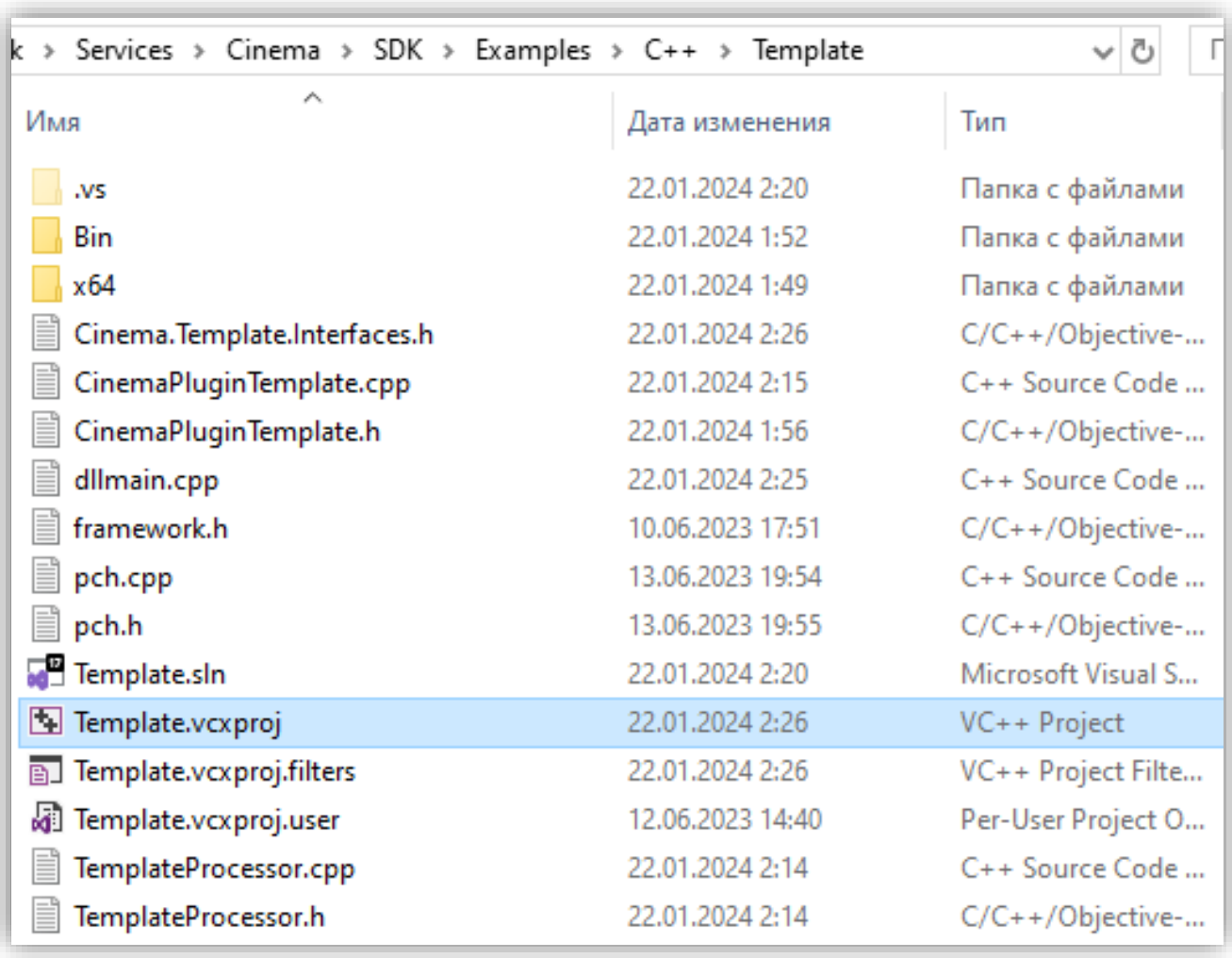

Рис. 1

- После открытия, возможно переименовать проект библиотеки в среде разработки, согласно планируемому целевому применению.
- В библиотеке с шаблонным плагином TemplateC внести свои изменения в файл на языке программирования C++.(См. рисунок 2).

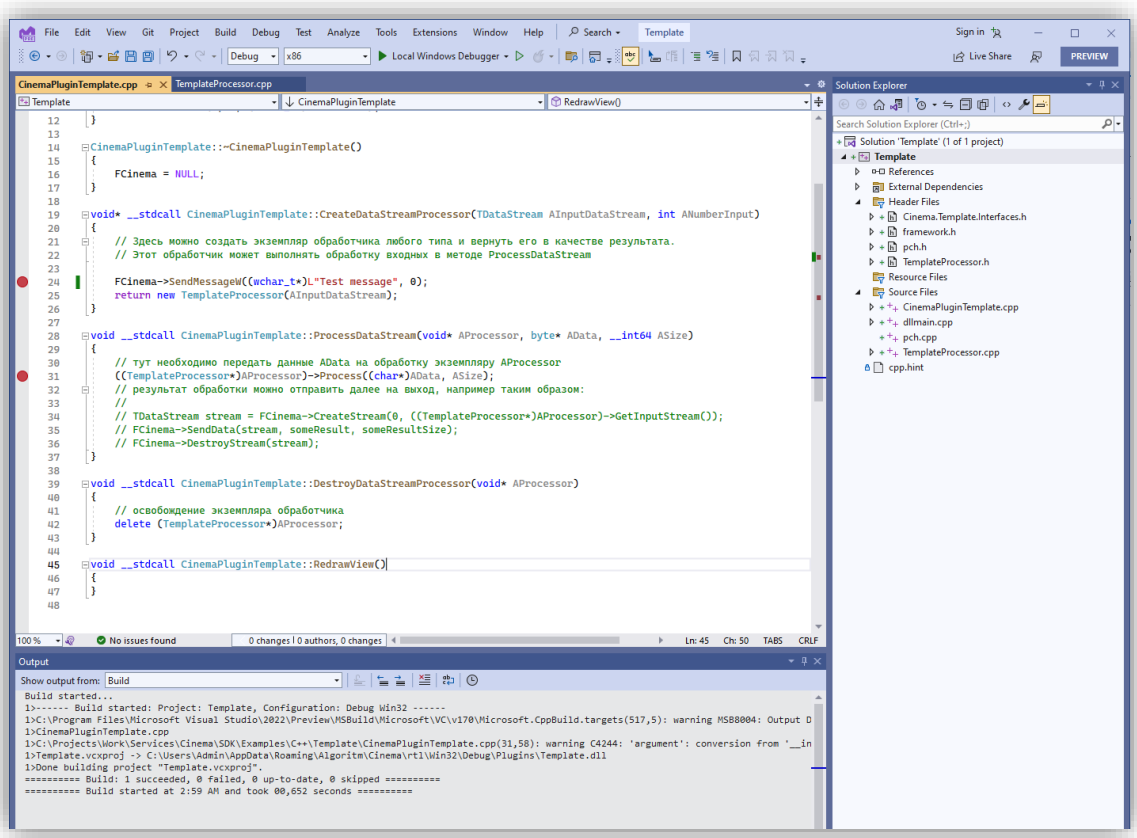

Рис. 2

4. Для создания плагина в среде Embarcadero Studio IDE на языке программирования Pascal:

- Открыть файл проекта PluginTemplate.dpr из библиотеки Template, расположенной по адресу:

Doc\SDK\Examples\C++\Template\Template.vcxproj

Проект содержит пример реализации плагина. (См. рисунок 3).

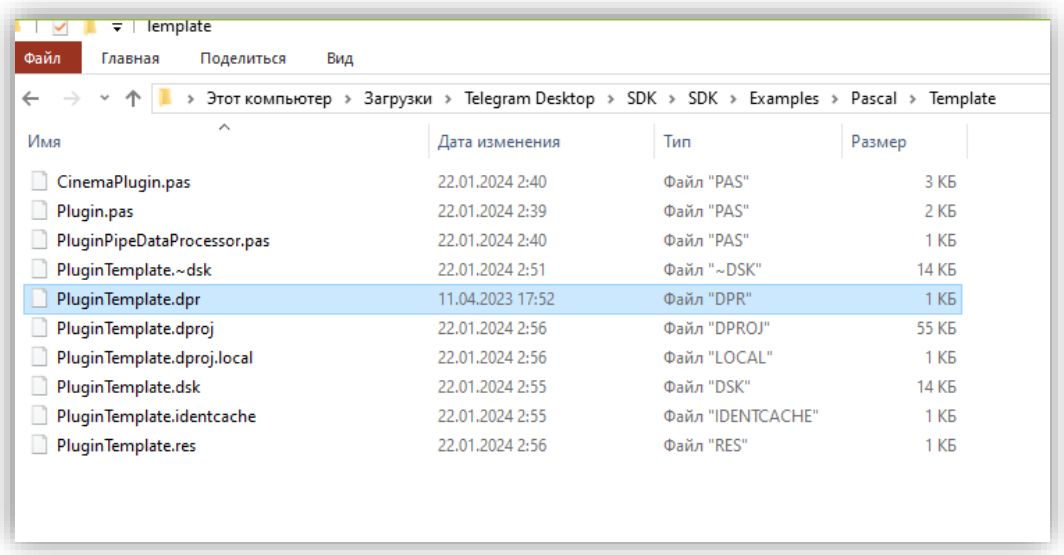

Рис. 3

- После открытия, возможно переименовать проект библиотеки в среде разработки, согласно целевому применению.
- В библиотеке с шаблонным плагином TemplatePascal внести свои изменения на языке программирования Pascal. (См. рисунок 4).

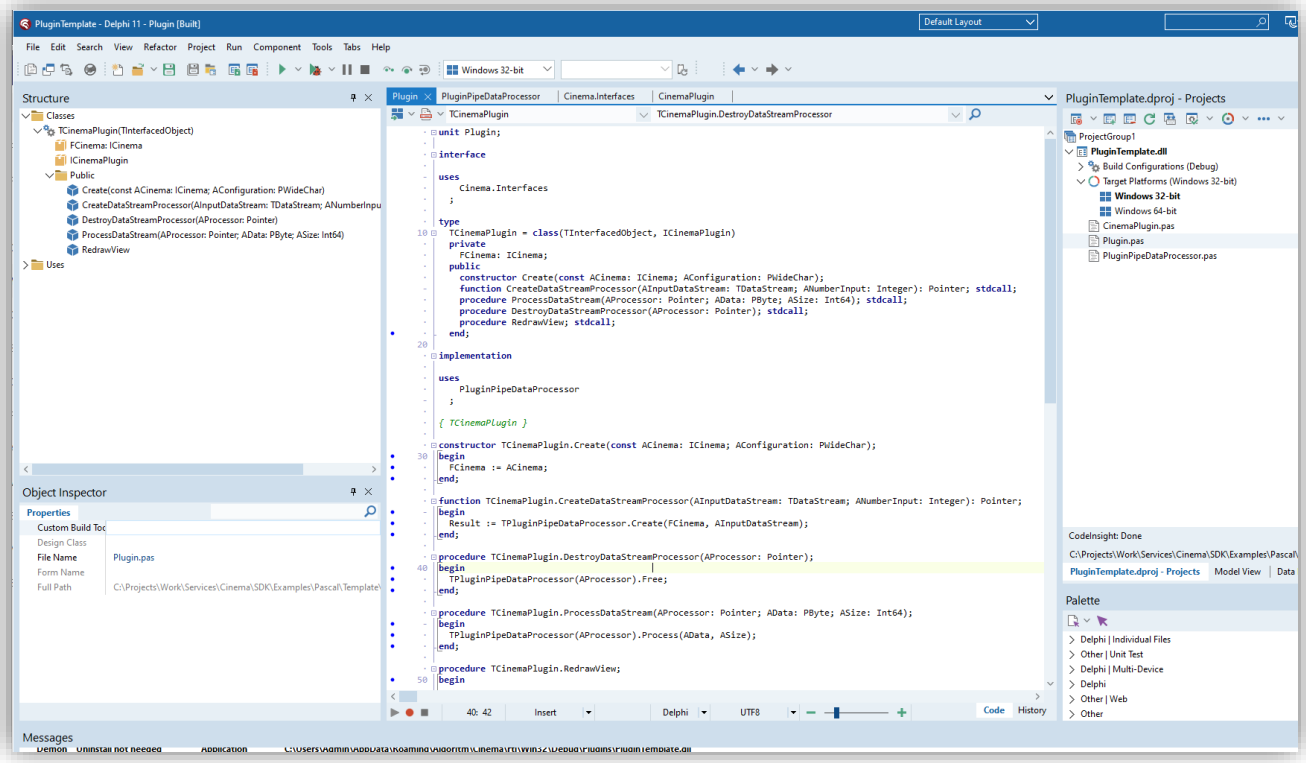

Рис. 4

Регистрация плагинов

1. Плагины зарегистрировать в “Plugins.json”. Файл находится в составе дистрибутива в подкаталоге rtl. Открыть файл и внести свои изменения:

   - Создать группу плагинов, например, с именем “My group plugins” и разделом “Plugins”:[], или добавить в любую подходящую группу из существующих.

   - Добавить новый элемент {} в список “Plugins”:[], в переменной “library” указать имя библиотеки, в переменной “name” указать имя плагина.

   - Сохранить файл с изменениями. (См. рисунок 5).

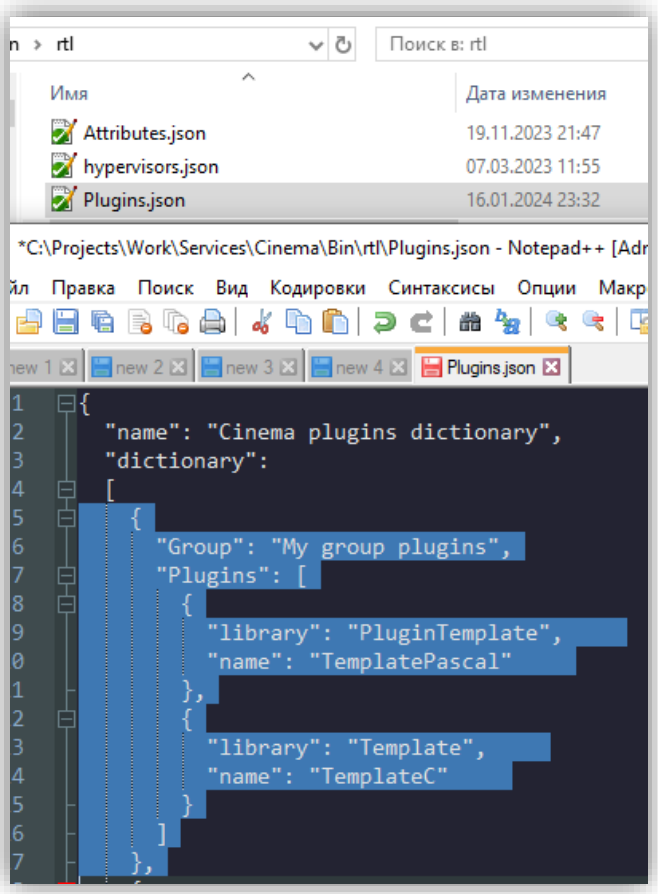

Рис. 5

2. Перезапустить программу (hypervisor, scenarist) для обновления списка плагинов.

Нештатные ситуации

1. При обнаружении отклонений и/или несоответствий в документе обращаться в техподдержку программного обеспечения.

ПРИЛОЖЕНИЕ 1. Ключевые принципы алгоритма работы ядра и разработки плагинов

1. Для возможности предоставления плагина внешним потребителям (ядру Cinema) библиотека экспортирует два метода:

- GetPluginList – список реализованных плагинов в библиотеке.
- CreatePlugin2 –создание плагина по имени, указанному в параметрах метода.

Количество реализуемых плагинов в одной библиотеке не ограниченно.

В момент выполнения сценария, количество создаваемых экземпляров плагинов одного вида равно количеству применённых в сценарии.

Пример:

Сценарий, где в процессе выполнения, может быть создано максимум 11 плагинов Client. (См. рисунок 6).

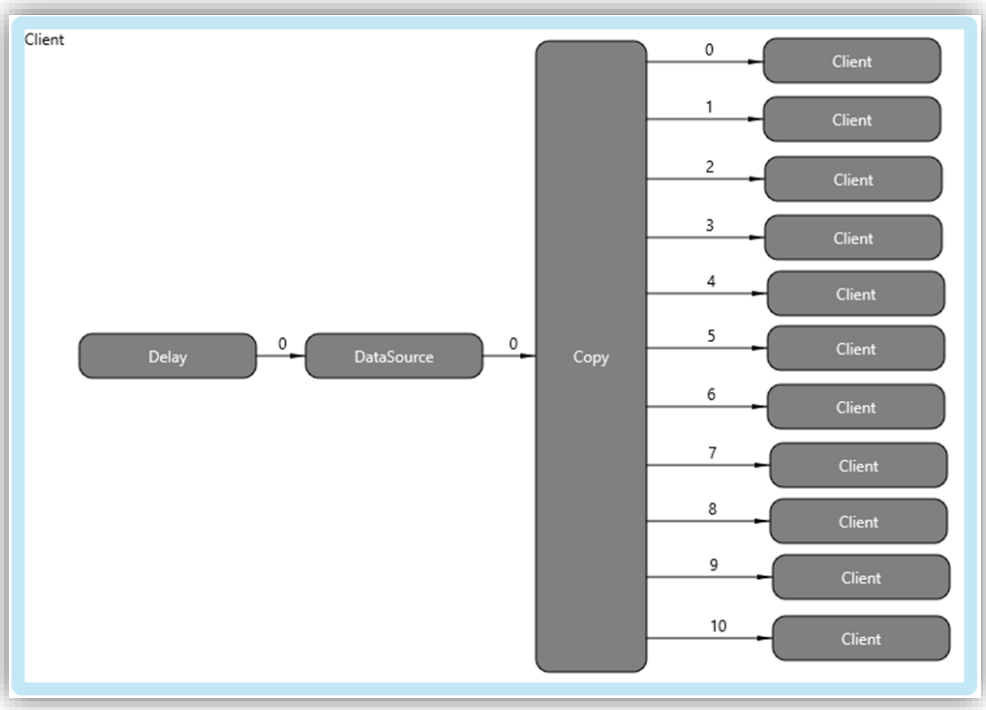

Рис. 6

2. Плагин – именованная реализация интерфейса ICinemaPlugin содержащий 3 основных метода для работы.

CreateDataStreamProcessor – метод вызывается ядром для создания экземпляра объекта. Тип объекта может быть любым

на усмотрение разработчика. Например, реализуется плагин, который будет выполнять сжатие входных данных в ZIP формат. В этом случае, метод CreateDataStreamProcessor должен порождать объект, который позволяет сжимать входной поток в ZIP формат. Таким образом, на выход в качестве результата своей работы, этот плагин будет отправлять архив в виде сжатых входных данных в формате ZIP.

DestroyDataStreamProcessor – антипод предыдущего метода и должен освобождать объект, созданный ранее в CreateDataStreamProcessor.

Разработчику не нужно заботится о времени жизни созданного объекта, ядро само вызовет метод DestroyDataStreamProcessor в нужный момент.

ProcessDataStream – обработка входного виртуального потока данных TDataStream объектом плагина, созданного в CreateDataStreamProcessor. В этом процессе выполняется основная работа плагина – обработка входных данных, и отправка на выход результата при необходимости.

Плагин может иметь множество физических входов и выходов. По аналогии с электронной печатной платой со множеством стандартизированных входов и выходов. Большинство плагинов реализуются с одним входом и одним выходом и пользователю понятны основы их применения. Плагины спроектированы и реализованы по принципу единственной ответственности, для облегчения их сопровождение на всём жизненном цикле.

3. ICinema – объект ядра Cinema, позволяет плагинам создавать виртуальные потоки данных (TDataStream), указывать их атрибуты, отправлять по ним данные и уничтожать их. Каждый поток имеет свои атрибуты, доступ к родительскому потоку и его атрибутам, множество дочерних потоков.

TDataStream – примитив ядра Cinema, именуемый виртуальным потоком данных, выполняющий роль связующего элемента между двумя объектами порождёнными плагинами методом ICinemaPlugin.CreateDataStreamProcessor и являющийся хранилищем атрибутов потока.

ПРИЛОЖЕНИЕ 2. Последовательность действий для создания нового плагина

1. Создать класс в существующей или новой библиотеке с реализацией интерфейса ICinemaPlugin.
2. Если библиотека новая, создать и добавить в раздел экспорта методы GetPluginList и CreatePlugin2.

- Для Pascal в отдельной секции exports. (См. рисунок 7).

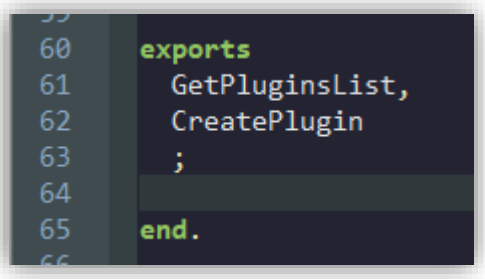

Рис.7

- Для C++ размещение в раздел экспорта библиотеки определяется инструкцией CINEMA_LIB_EXPORT, которая находится перед методом. (См. рисунок 8).

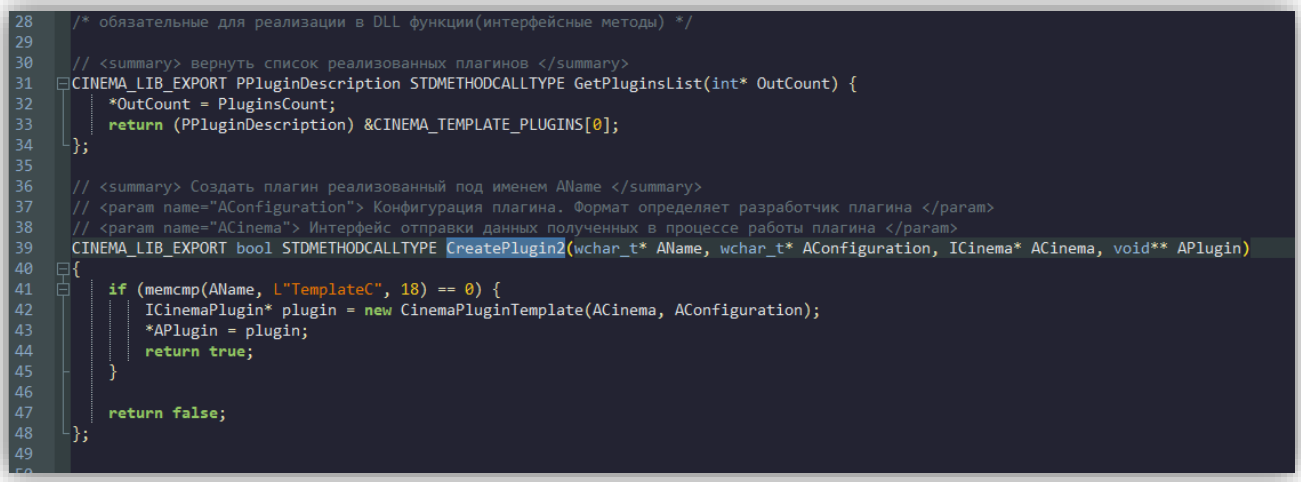

Рис. 8

3. Метод GetPluginList возвращает ядру Cinema данные об актуальном списке плагинов, созданных в библиотеке. Для того, чтобы плагин попал в список, заполнить структуру TPluginDescription. В выходную переменную OutCount

указать актуальный размер возвращаемого списка указателей на структуры TPluginDescription.

4. В методе CreatePlugin2 обеспечить создание нового плагина.
5. Зарегистрировать плагин в файле “Plugins.json”.
6. Определить назначение плагина и наполнить кодом методы: CreateDataStreamProcessor, ProcessDataStream, DestroyDataStreamProcessor.