1. **Общие сведения**

Добрый день уважаемые коллеги!

Хочу представить Вам набор компонентов TextGenDel (**Text** **gen**erator for **Del**phi), реализующих генератор шаблонов для Delphi 7 +.

При разработке бизнес приложений часто ставится задача экспорта данных в текстовые форматы (XML, JSON, CSV).

При постановке подобной задачи, заказчик определяет формат выгружаемого текстового файла в виде набора правил, определяющих отражение исходного массива данных приложения на результирующий файл экспорта. Зачастую подобные правила содержат массу условий, в зависимости от выполнения или невыполнения которых содержимое результата меняется. Подобную сложную логику реализовать декларативно не всегда представляется возможным, и поэтому приходится разрабатывать отдельные программные модули, реализующие её.

Также, в составе постановки задачи экспорта часто прикладываются образцы результирующих текстовых файлов, содержащие тестовые данные для различных исходных данных.

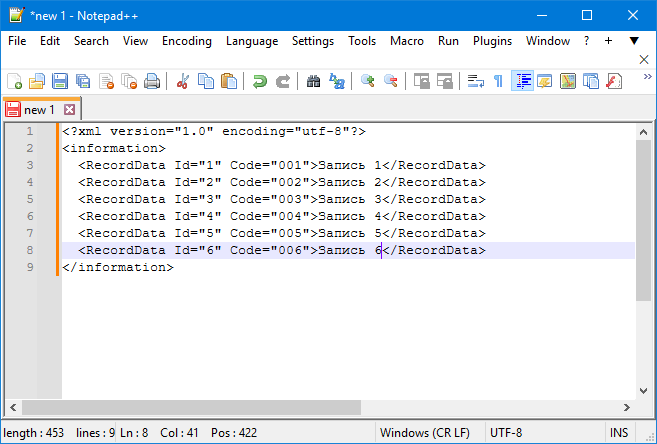


Рис. 1 Простой файл образца

Такие файлы – образцы результатов легко можно преобразовать в файлы – шаблоны, содержащие вместо конкретных тестовых данных макроопределения – ссылки на данные приложения. Кроме того, шаблон может содержать участки программного кода (вставки – скрипты), реализующие сложную логику отображения данных на результирующий файл.

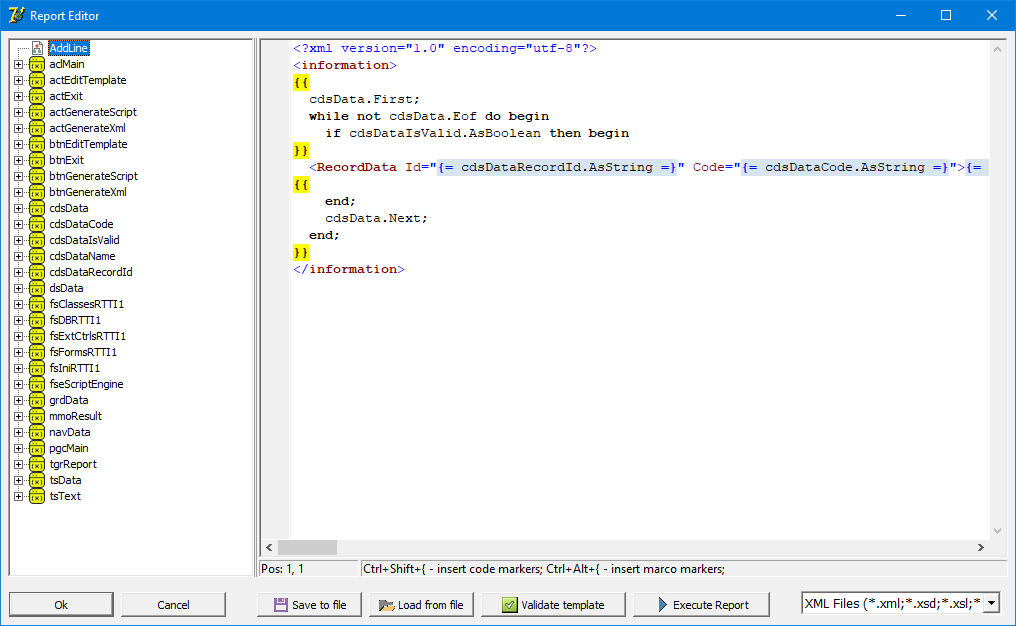


Рис. 2 Шаблон, созданный на основании файла образца в редакторе шаблонов TextGenDel

Итоговый шаблон обрабатывается генератором шаблонов – и в результате формируется выходные текстовые данные.

Набор компонентов TextGenDel реализует подобный генератор шаблонов. Принцип его работы следующий: исходный шаблон преобразуется в программный код Pascal. Далее этот программный код исполняется и формирует результирующий текст.

В качестве механизма исполнения программного кода TextGenDel использует компоненты FastScript, являющиеся частью широко используемого генератора отчетов FastReport.

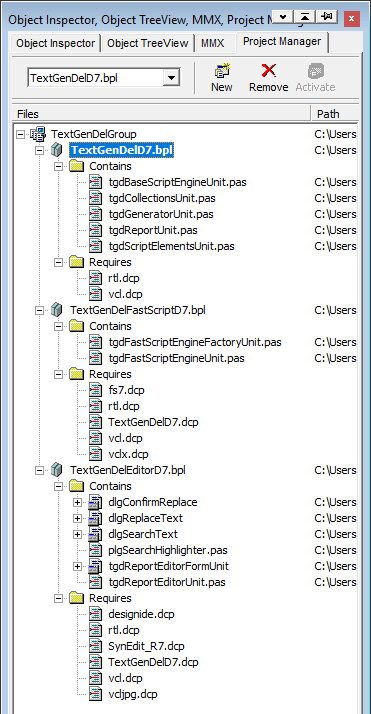
Язык FastScript Pascal – легко расширяемый (в том числе и в Runtime). Кроме того, компоненты FastScript предоставляют доступ к зарегистрированным метаданным, что позволяет реализовать удобный редактор шаблонов.

1. **Развертывание компонентов**

Предварительно необходимо установить в IDE компоненты FastScript версии 1.9+ (они присутствуют в составе набора компонентов FastReport 5+).

Далее необходимо загрузить исходный код компонентов TextGenDel, расположенный по адресу <https://github.com/vvboborykin/TextGenDel> (используя либо функцию клонирования git, либо загрузив ZIP архив исходных текстов, и распаковав его на Вашем компьютере).

После этого, откройте в Delphi группу проектов TextDelGenGroup.



Последовательно установите пакеты компонентов TextGenDelD7 и TextGenDelFastScriptD7.

После установки пакетов, на палитре компонентов IDE появится закладка TextGenDel содержащая два компонента

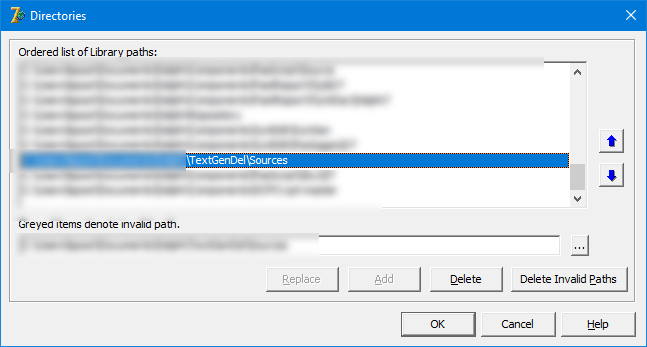


Первый компонент - собственно генератор шаблонов (TtgdReport).

Второй компонент - сервис исполнения скриптов на основе FastScript (TtgdFastScriptEngineFactory).

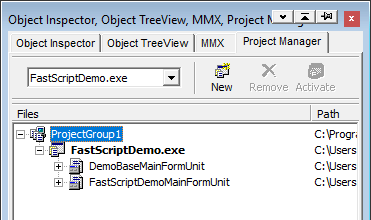
Третий пакет включённый в состав группы (TextGenDel70) содержит специализированныый редактор шаблонов для TtgdReport (может быть использован как в Design так и в Run Time). В составе этого пакета используется бесплатный набор компонентов SynEdit (<https://github.com/SynEdit/SynEdit>). В случае если Вы решите использовать этот редактор, предварительно установите SynEdit в вашей IDE. После этого установите пакет TextGenDel70.

Последний шаг в развёртывании компонентов - включение каталога исходников TextGenDel\Sources в список путей библиотек в параметрах IDE

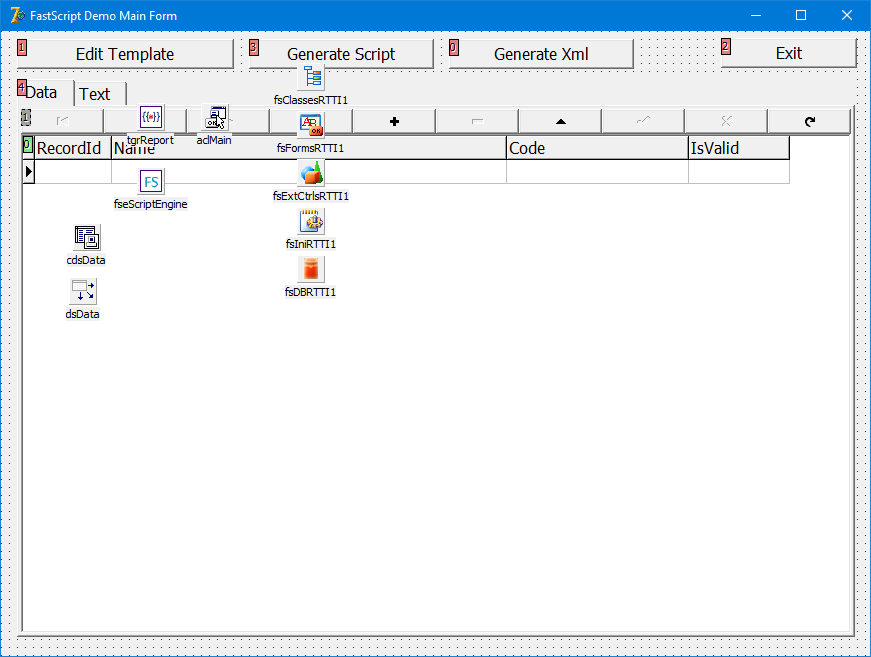


1. Использование компонентов TextGenDel

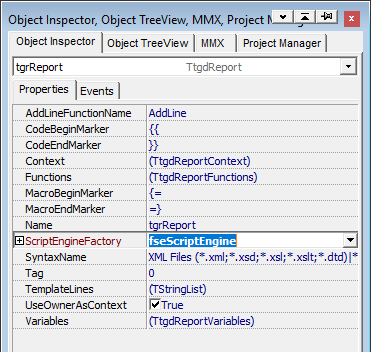
Порядок использования компонентов TextGenDel можно рассмотреть, и следуя простой тестовый проект, включённый в состав компонентов, расположенный в каталоге TextGenDel\Demo\FastScript.



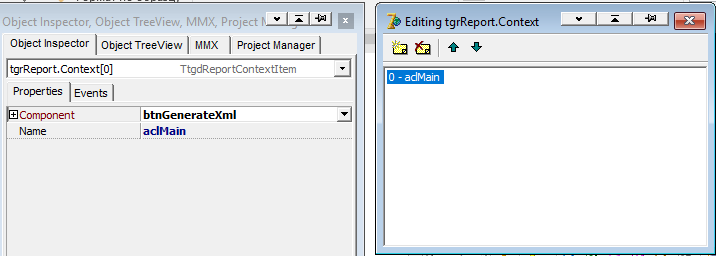
Проект состоит из базовой формы, а также унаследованной от неё главной формы приложения.

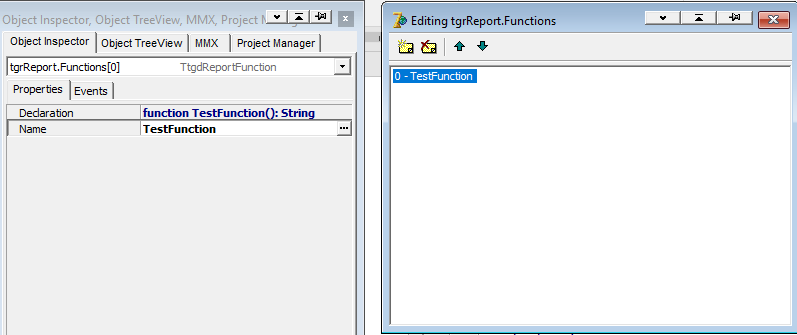


Компонент tgrReport (генератор шаблонов) имеет следующие свойства:

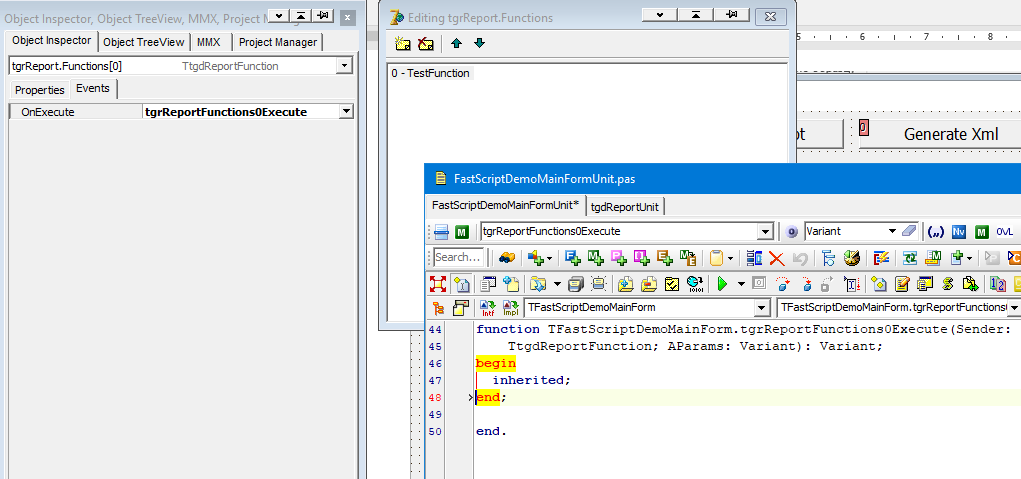


* AddLineFunctionName - имя функции в скрипте генерации, используемое для добавления новой строки в результат генерации. Можно не менять если имя по умолчанию не конфликтует с именами функций используемых в других местах.
* CodeBeginMarker - строка, маркер начала блока скрипта в шаблоне. Текст, следующий после этого маркера в шаблоне, считается программным кодом (скриптом), и при генерации никак не изменяется. Маркер начала блока скрипта должен начинать строку.
* CodeEndMarker - строка, маркер конца блока скрипта в шаблоне. Должна завершать блок скрипта (быть последней строкой в последней части блока)
* Context - коллекция ссылок на компоненты (например, на внешние модули данных), которые могут быть использованы в шаблоне. Компоненты, расположенные на том же контейнере (форме, модуле данных) что и сам компонент генератора отчётов не требуют таких ссылок (смотри свойство UseOwnerAsContext)



* Functions – коллекция функций расширения скрипта, которые могут быть использованы в шаблоне. Для каждой из функций свойство Declaration должно содержать объявление функции, свойство Name - её имя.
* 

Обработчик события OnExecute определяет фактическую функциональность функции расширения. Параметр AParams обработчика содержит массив переданных при вызове функции в шаблоне параметров, результат обработчика - результат исполнения функции расширения возвращаемый в шаблон.

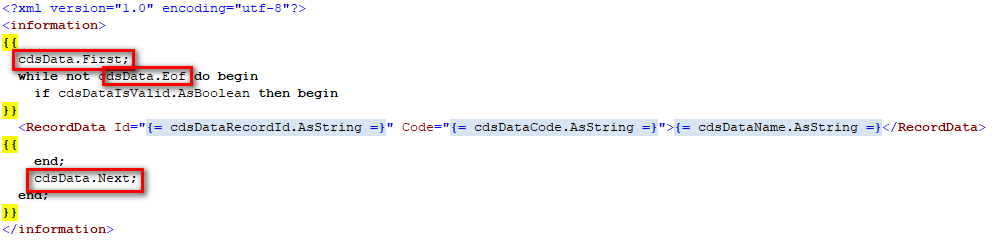


Используя коллекцию Functions Вы можете нарастить функциональность языка шаблонов, добавив в него необходимые, часто используемые для Вас, функции.

* MacroBeginMarker - маркер начала макроопределения в шаблоне. Текст находящийся в шаблоне после строки **маркера начала макроопределения** и до начала строки **маркера окончания макроопределения**, считается выражением на языке скрипта, значение которого необходимо вычислить при генерации, и подставить вместо макроопределения.

Например, макроопределение шаблона {= theNameOfField.AsString =} при генерации будет заменено на строковое представление значение поля theNameOfField.

* MacroEndMarker - маркер окончания макроопределения в шаблоне.
* ScriptEngineFactory - ссылка на компонент исполнения скриптов. Это свойство обязательно должно быть заполнено ссылкой на экземпляр компонента TtgdFastScriptEngineFactory.
* SyntaxName - формат синтаксиса шаблона. Используется при работе редактора компонента, определяет расцветку шаблона. На другую функциональность на влияет
* TemplateLines - строки шаблона, обрабатываемые при генерации
* UseOwnerAsContext - признак использования контейнера (владельца / Owner) компонента TtgdReport, как контекста данных. В случае, если это логическое свойство имеет значение True, то все компоненты, содержащиеся на этом контейнере, могут быть использованы в шаблоне. Например, если на форме (или в модуле данных) содержащем компонент TtgdReport присутствует экземпляр TClientDataSet с именем cdsData, то в блоках скрипта и макроопределениях шаблона можно будет ссылаться на этот компонент используя его имя cdsData.



Кроме того, в контекст включаются и экземпляры вложенных TFrame, расположенные на контейнере – владельце.

* Variables - коллекция именованных значений, которые могут быть использованы в шаблоне

