А.В. Леденев, И.А. Семенов, В.А. Сторожевых

# Динамически загружаемые библиотеки: структура, архитектура и применение

*DLL — это сокращение от Dynamic Link Library (динамически загружаемая библиотека). С формальной точки зрения DLL — особым образом оформленный относительно независимый блок исполняемого кода. DLL используются множеством приложений. Все приложения для ОС Windows так или иначе используют динамические библиотеки.*

*Данный материал посвящен особенностям реализации DLL в различных средах и для различных целей.*

## Что такое DLL?

Особый способ оформления предполагает наличие в DLL так называемых *секций импорта* и *экспорта*. Секция экспорта указывает те идентификаторы объектов (функций, классов, переменных), доступ к которым предоставляет данная DLL. В этом случае мы говорим об экспортировании идентификаторов из DLL. В общем случае именно секция экспорта представляет особый интерес для разработчиков. Хотя ничто не мешает реализовать DLL, которая не имеет данной секции, но тем не менее выполняет полезную работу.

Относительная независимость связана с наличием/отсутствием секции импорта у DLL (т. е. секции, в которой описываются внешние зависимости данной DLL от других). Подавляющее большинство DLL (за исключением, быть может, DLL-ресурсов) импортирует функции из системных DLL (**kernel32.dll**, **user32.dll**, **gdi32.dll** и проч.). В большинстве случаев при создании проекта в его опциях автоматически проставляется стандартный набор таких библиотек. Иногда в этот список необходимо добавить DLL, требующиеся для конкретных задач (например, в случае использования библиотеки сокетов требуется дополнительно подключить библиотеку ws2_32.dll).

Исполняемый код в DLL не предполагает автономного использования. Перед тем как можно будет приступить к использованию, необходимо загрузить DLL в область памяти вызывающего процесса (т. е. DLL не может выполняться сама по себе — ей обязательно нужен клиент). Это явление носит название «*проецирование DLL на адресное пространство процесса*». И это не удивительно, если вспомнить тот факт, что процессор работает не только с регистрами, но и с адресами памяти. Поэтому каждому объекту DLL требуется свое место «под солнцем», чтобы иметь возможность быть выполненным при вызове. В конечном коде exe-файла, который генерирует компилятор, не будет инструкций процессора, соответствующих коду данной функции. Вместо этого будет сгенерирована инструкция вызова соответствующей функции (call). Так как DLL отображена на адресное пространство процесса, то код DLL будет легко доступен по call-вызову.

Итак, формально, DLL — особым образом оформленный программный компонент, доступ к исполняемому коду которого приложение получает в момент старта (DLL неявной загрузки) или в момент использования (DLL явной и отложенной загрузки).

Что же касается физического представления на диске, то разница между dll- и exe-файлами небольшая. Как динамически лин-