Лекция 11. Libraries

Static & Dynamic

Статическая библиотека

- Является объединением скомпилированных единиц трансляции.
- Если статическая библиотека используется разными динамическими библиотеками, загруженными в процесс, то их будет несколько копий.
- Распространяется отдельным файлом, требует лишь линковки с основной программой (выполняемым файлом или динамической библиотекой).

Динамическая библиотека

- В значительной степени совпадает по формату с выполняемым файлом.
- Используется одна на процесс, независимо от того, какие другие библиотеки этого процесса на нее ссылаются
- Код библиотеки загружается единожды для всех процессов (windows), данные у каждого процесса свои.
- Может использоваться отложенная (ленивая) загрузка.
- Требуется явное указание, что экспортируется из библиотеки (windows). Или явное закрытие экспорта (linux).

Static vs Dynamic

- S>D Статическая библиотека «влинковывается» в код программы, не возникает проблем зависимостей. Нет необходимости заботится о распространении библиотек - они зашиты внутрь программы.
- S>D Добавляется только та часть статической библиотеки, которая используется результирующей программой.
- D>S Статическая библиотека увеличивает код программы (если неоднократно повторяется).
- D>S Есть возможность заменить библиотеку (требуется совместимость по API) для уже собранного приложения.
- D>S Динамическая библиотека одна на процесс - именно в них следует делать синглтон на процесс.

Как создать статическую библиотеку?

- **MSVC**. Указать Configuration Type проекта Static Library (.lib).
- **GCC**.

```
1.  g++ -Wall -c *.cpp
2.  ar -cv libsome.a *.o
```

- Рекомендуется экспортировать функции как extern "C", если только нет 100% уверенности, что линковаться будет той же версией линкера. Например, если сборка происходит в рамках одного проекта

Как создать динамическую библиотеку?

- **MSVS**. Указать Configuration Type проекта Dynamic Library (.dll). При этом формируется библиотека импорта (.lib)
- **GCC**.

```
1.  g++ -Wall -c *.cpp
2.  g++ -shared -o libsome.so *.o
```

- Рекомендация по использованию extern "C" такая же, как и в статической библиотеке.

Использование динамической библиотеки?

- **MSVC**. В настройках проекта Linker прописать Additional Dependencies и Additional Library Directories. Независимо, статическая или динамическая библиоетка ли, прописана будет .lib
- Или написать в коде `#pragma comment(lib,"MyDll.lib")`

- **GCC**. В параметрах линкера указать -L для путей -l для названий библиотек (без префикса **lib**). Есть особенность порядка зависимостей.

- Где идет поиск библиотеки при выполнении
  - Windows: `PATH`, `System Path`, `Loading Directory`. Также на пути поиска может влиять файл манифеста и настройка `LOAD_WITH_ALTERED_SEARCH_PATH`*
  - Linux: `LD_LIBRARY_PATH` env var, `RPath`, `system search path` (/lib, /usr/lib)

Загрузка библиотек без предварительной линковки

```
1.  typedef void (*func_pointer)(int, double);
2.
3.  HMODULE lib = LoadLibrary("some.dll"); // or LoadLibraryEx
4.
5.  auto func = reinterpret_cast<func_pointer>(
6.                  GetProcAddress(lib, func_name));
7.
8.  FreeLibrary(lib);
```

- Загрузка в **linux** возможна с помощью функций `dlopen, dlsym, dlclose`
- При загрузке/выгрузке библиотеки в будет вызвана entry point (если есть):
  - **windows**. По умолчанию `DllMain`.
  - **linux**. Функции, обозначенные `__attribute__ ((constructor))` и `__attribute__ ((destructor))`

Экспортируемые функции

- По умолчанию **GCC** экспортит все функции, но можно их спрятать (в частности, это уменьшит размер):

```
1.  #define __HELPER_DL_EXPORT __attribute__((used, visibility ("default")))
2.  #define __HELPER_DL_LOCAL  __attribute__ ((visibility ("hidden")))
```

- Посмотреть экспортируемые функции можно
  - **linux**. Утилитой nm
  - **windows**. Программой dependency walker.

Технология COM

- Технология Component Object Model (COM) от Microsoft позволяет справиться с DLL Hell и не думать, где же нужные библиотеки.
- Работает через регистрацию объектов и интерфейсов в реестре (каждой сущности сопоставляется GUID).
- Поддерживает версионирование.
- Позволяет легко экспортировать объекты из библиотек.
- Не позволяет вызывать конструкторы объектов.
- На основе COM реализованы Microsoft OLE Automation, ActiveX, DCOM, DirectX, ...
- Выгружает библиотеку, когда объекты из нее больше не используются.

Вопросы?