**Что такое библиотека и зачем она нужна?**

**Библиотека** — это набор готовых функций, классов и объектов для решения каких-то задач. Библиотеки позволяют не "изобретать велосипед" при решении типовых задач, а использовать наработки инженеров которую эту задачу уже решили (и надеюсь) оптимизировали.

Далеко не всегда программист должен думать о реализации бизнес-логики, а не о том, как работает оборудование или по какому протоколу передаются данные. Поэтому современные языки программирования имеют удобный функционал добавления в проекты различных библиотек. К сожалению С++ так не умеет, но это и не всегда плохо.

Из личного интереса можете взять своих друзей (можно даже не программистов) и поиграть в игру "Не угадай название библиотеки JavaScript или Python". Правила просты: каждый называет случайно слово на английском языке и пытается найти такую библиотеку в этих двух языках, если находит, то выполняет какое-то действие, если не находит, то выигрывает игру. Поверьте мне, выиграть будет сложно :)  
  
И так, почему факт того, что С++ не умеет быстро импортировать библиотеки в проект это не всегда плохо? Да потому что библиотеки делаются многими людьми и не всегда квалификация программиста соответствует сложности задачи которую пытается решить программист своей библиотекой. Бездумный импорт лишнего кода в своей проект никогда не принесёт что-то хорошее. Запомните это.

**Библиотека в С++**

Библиотека объектных файлов − несколько объектных файлов, которые используются для хранения функций и ресурсов отдельно от исполняемого файла. Библиотека  содержит символьный индекс, который состоит из названий функций и переменных и т.д., которые содержатся в библиотеке. Это позволяет ускорить процесс компоновки программы, так как поиск функций и переменных в объектных файлах библиотеки происходит намного быстрее, чем поиск в наборе указанных объектных файлов.

Поэтому использование библиотеки позволяет компактно хранить все требуемые  объектные файлы в одном месте, и при этом значительно повысить скорость компиляции.  Таким образом, можно создавать большие проекты, которые больше не будут отнимать  много времени на компиляцию и поиск ошибок. Однако нужно помнить, что не стоит также чересчур разбивать программу, иначе получится несколько десятков файлов, в которых рано или поздно можно запутаться. Рекомендуется в отдельные файлы помещать  те функции или классы, с которыми приходится больше всего работать при отладке. После того, как функция будет окончательно отлажена, ее вполне можно перенести в более крупный файл.

Объектные библиотеки по способу использования разделяются на два вида:

* Статические библиотеки
* Динамические библиотеки

**Статическая С++ библиотека**

Статическая библиотека – это коллекция объектных файлов, которые присоединяются к программе во время компоновки. Таким образом, статические библиотеки используются только при создании программы. Потом же, при выполнении программы, они  участия не принимают, в отличие от динамических библиотек.

Проект статической библиотеки крайне похож на проект обычного С++ проекта, но имеет небольшие отличия. Предполагается, что результат работы программы будет передан в функцию, а также вызов функций может производить не из существующих файлов проекта.

Например создадим статическую библиотеку для подсчёта площадей фигур:

* **Создание проекта статической библиотеки в Visual Studio**

1. в строке меню выберите **файл создать Project** , чтобы открыть диалоговое окно **создание нового Project** .
2. в верхней части диалогового окна задайте для параметра **Language** значение **C++**, задайте для параметра **Platform** значение **Windows**и задайте для параметра **Project тип** значение **Library**.
3. В отфильтрованном списке типов проектов выберите пункт **Мастер классических приложений Windows**, а затем нажмите кнопку **Далее**.
4. На странице **Настроить новый проект** введите *SquareLibrary* в поле **Имя проекта**. В поле **Имя решения** введите *SquareMath*. Нажмите кнопку **Создать**, чтобы открыть диалоговое окно **Проект классического приложения Windows**.
5. В диалоговом окне **Проект классического приложения Windows** в разделе **Тип приложения** выберите **Статическая библиотека (.lib)** .
6. В разделе **Дополнительные параметры** снимите флажок **Предварительно откомпилированный заголовок**, если он установлен. Установите флажок **Пустой проект**.
7. Нажмите кнопку **ОК**, чтобы создать проект.

* Затем создайте несколько .cpp файлов и в них разместите определения функций.
* В едином заголовочном файле разместите все прототипы созданных функций.
* Запустите проект

В папке Debug вашего проекта должен появится *SquareLibrary.lib -*это и есть ваша библиотека.

Чтобы подключить вашу библиотеку к проекту, необходимо переместит .lib в файл в папку Debug вашего проекта и в IDE с помощью команды "Добавить/Add" контекстного меню вашего проекта добавить библиотеку.

Вам осталось лишь прописать в необходимом месте в коде #includeк вашей библиотеке.

В различных IDE добавление чуть отличается, однако смысл понятен :)

**Разработка динамических библиотек**

Динамическая библиотека (DLL), с точки зрения программиста, представляет собой библиотеку функций (ресурсов), которыми может пользоваться любой процесс, загрузивший эту библиотеку. Сама загрузка, кстати, отнимает время и увеличивает расход по требляемой приложением памяти; поэтому бездумное дробление одного приложения на  множество DLL не рекомендуется.

Однако, если какие-то функции используются несколькими приложениями, то, поместив их в одну DLL, мы избавимся от дублирования кода и сократим общий объем приложений − и на диске, и в оперативной памяти. Можно выносить в DLL и редко используемые функции отдельного приложения.

Загрузившему DLL процессу доступны не все ее функции, а лишь явно предоставляемые самой DLL для "внешнего мира" − т. н. экспортируемые. Функции, предназначенные сугубо для "внутреннего" пользования, экспортировать бессмысленно (хотя и не  запрещено). Чем больше функций экспортирует DLL − тем медленнее она загружается;  поэтому к проектированию интерфейса (способа взаимодействия DLL с вызывающим  кодом) следует отнестись повнимательнее.

Для экспортирования функции из DLL - перед ее описанием следует указать ключевое слово \_\_declspec(dllexport).