Лабораторная работа №1
по курсу «Высокоуровневое программирование» (1 семестр)
«Изучение среды программирования Visual Studio 2019»

Оглавление

Цель: приобретение практических навыков работы в среде программирования Visual Studio 2019.

Задачи:

1. Изучить основные приемы работы в указанной среде программирования.
2. Научиться работать с текстовым редактором, запускать на компиляцию, выполнение и отлаживать проект.

В начало

Типы проектов Visual C++

При запуске программы Вас будет приветствовать окно следующего вида (Рис. 1). В этом окне можно открыть ранее созданные приложения или нажать на **«Создание проекта»**.

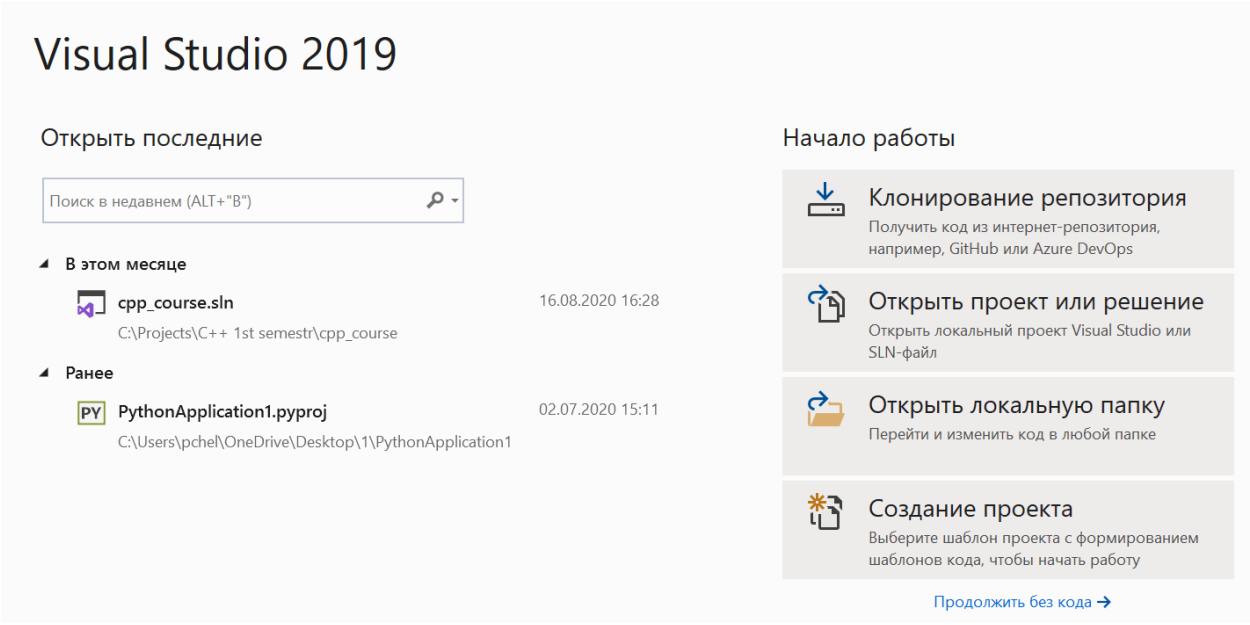


Рис. 1. Создание нового проекта в Visual Studio 2019

На следующем шаге матер предложит выбрать тип создаваемого приложения. Нам необходимо выбрать пустой проект (Рис. 2).

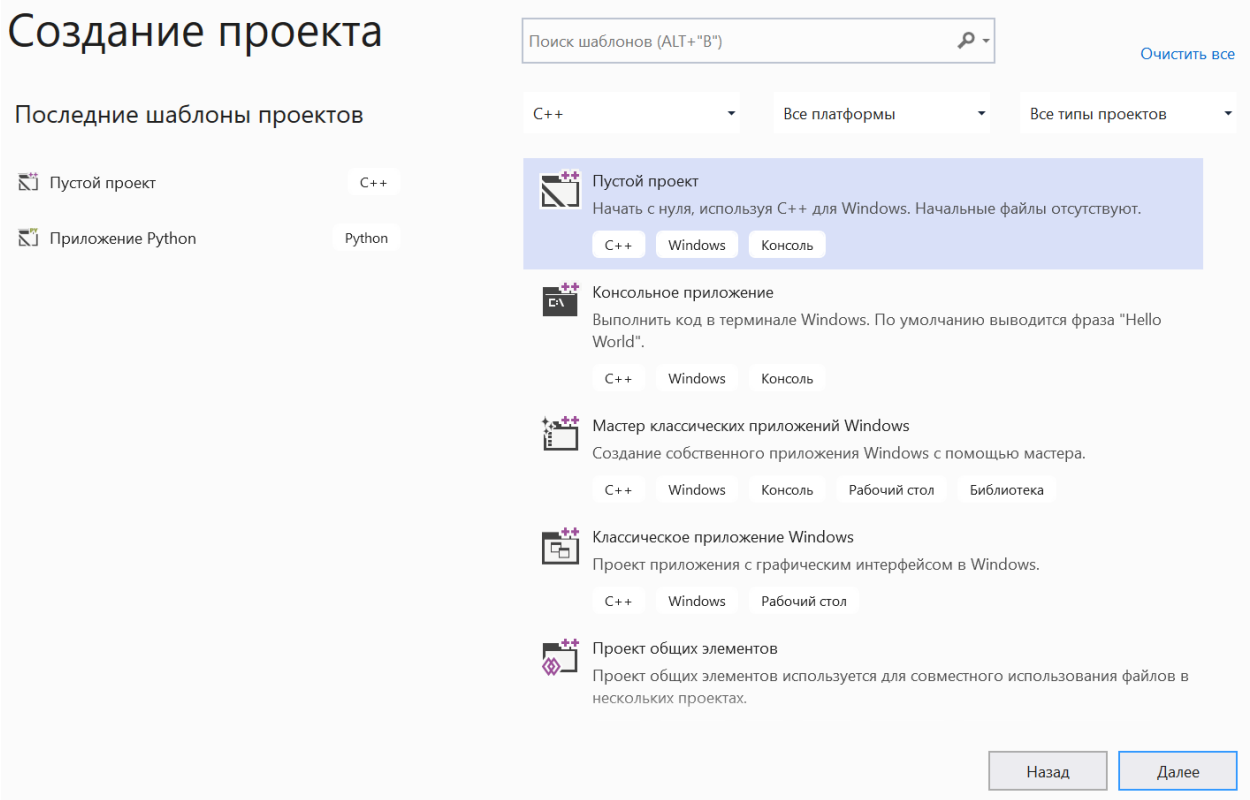


Рис. 1. Создание пустого проекта в Visual Studio 2019

Новые проекты также можно создавать следующими способами.

- Выбрав **Файл | Создать | Проект** из существующего кода и следуя указаниям по добавлению существующих файлов с исходным кодом. Этот вариант лучше всего подходит для небольших и простых проектов, к которым можно отнести

проекты, включающие не более 25 файлов с исходным кодом и незначительное или нулевое количество сложных зависимостей.

- Начав с файла *makefile* и выбрав шаблон проекта *Makefile* на вкладке **"Общие"**.
- Создавая пустой проект (на вкладке **Общие**) и добавляя вручную файлы с исходным кодом, для чего необходимо щелкнуть правой кнопкой мыши узел проекта в обозревателе решений и выбрать **Добавить | Существующий элемент**.
- Использование мастер приложений **Win32**.

Вы можете воспользоваться шаблоном проекта для создания базовой структуры программы, меню, панелей инструментов, значков, ссылок и инструкций `#include`, подходящих для разрабатываемого проекта. Visual Studio содержит несколько видов шаблонов проектов Visual C++ и предоставляет для многих из них мастеры, позволяющие настраивать проекты во время их создания. Сразу же после создания проекта вы можете выполнить его сборку и запустить приложение. В общем случае рекомендуется периодически производить сборку по мере разработки приложения.

Использовать шаблон при создании проекта необязательно, однако в большинстве случаев это гораздо эффективнее, так как проще изменять имеющиеся файлы и структуру проекта, чем создавать их "с нуля".

Некоторые шаблоны проектов

Общие

Шаблон проекта	Создание проекта
Пустой проект	Создание проектов и решений
Специальный мастер	Создание пользовательского мастера
Проект, использующий файл makefile	Создание проекта Makefile

Win32

Шаблон проекта	Создание проекта
Консольный проект Win32	Создание консольного приложения
Проект Win32	Создание классического приложения Windows

Комментарии TODO

Многие файлы, создаваемые шаблоном проекта, содержат комментарии TODO, помогающие найти места для вставки собственного исходного кода.

В начало

Типы файлов, создаваемых для проектов Visual C++

Когда вы создаете проект Visual C++, это может быть новое решение или новый проект, добавляемый к решению. Нетривиальные приложения обычно разрабатываются как решения, содержащие множество проектов.

Обычно выходным файлом проекта является EXE- или DLL-файл. Проекты могут зависеть друг от друга; в процессе сборки среда разработки Visual C++ проверяет зависимости как внутри проектов, так и между проектами. Каждый проект имеет основной исходный код, а также, в зависимости от вида проекта, может содержать другие файлы, определяющие различные аспекты проекта. Указанием на содержимое этих файлов являются их расширения. В среде разработки Visual Studio по расширениям файлов определяется способ обработки их содержимого в ходе построения.

В таблице 1 приведен список общих файлов проекта Visual C++ и их расширений.

Таблица 1. Общий список файлов проекта Visual C++

Расширение файла	Тип	Описание
ASMX	Исходный код	Файл развертывания.
ASP	Исходный код	ASP-файл.
ATP	Проект	Файл шаблона приложения проекта.
BMP, DIB, GIF, JPG, JPE, PNG	Ресурс	Файлы изображений общего характера.
BSC	Компиляция	Файл кода браузера.
CPP, C	Исходный код	Основные файлы исходного кода приложения.
CUR	Ресурс	Растровый графический файл курсора.

DBP	Проект	Файл проекта базы данных.
DISCO	Исходный код	Файл документа динамического обнаружения. Обеспечивает обнаружение веб-служб XML.
EXE, DLL	Проект	Исполняемые файлы или файлы библиотек динамической компоновки.
H	Исходный код	Файл заголовка.
HTM, HTML, XSP, ASP, HTC, HTA, XML	Ресурс	Общие веб-файлы.
HXC	Проект	Файл справки проекта.
ICO	Ресурс	Растровый графический файл значка.
IDB	Компиляция	Файл состояния, содержащий информацию о зависимостях между файлами исходного кода и определениями классов, которые могут использоваться компилятором в ходе минимального перепостроения и добавочной компиляции. Для задания имени IDB-файла используйте параметр компилятора /Fd. Дополнительные сведения см. в разделе /Gm (включение минимального перепостроения).
IDL	Компиляция	Файл языка определения интерфейса.
ILK	Компоновка	Файл инкрементной компоновки.
MAP	Компоновка	Текстовый файл, содержащий информацию для компоновщика. Для задания имени MAP-файла используйте параметр компилятора /Fm.
MFCRIBBON-MS	Ресурс	Файл ресурсов, содержащий код XML, который определяет кнопки, элементы управления и атрибуты в ленте.
OBJ, O		Объектные файлы – скомпилированные, но не скомпонованные.

PCH	Отладка	Файл предкомпилированных заголовков.
RC, RC2	Ресурс	Файлы скриптов ресурсов для генерации ресурсов.
SBR	Компиляция	Промежуточный файл обозревателя исходного кода. Входной файл для BSCMAKE.
.SLN	Решение	Файл решения.
SUO	Решение	Файл параметров решения.
TXT	Ресурс	Текстовый файл, обычно README-файл.
VAP	Проект	Файл проекта Visual Studio Analyzer.
VBG	Решение	Файл совместимой группы проектов.
VBP, VIP, VBPROJ	Проект	Файл проекта Visual Basic.
VCXPROJ	Проект	Файл проекта Visual C++.
VCXPROJ.FILTERS	Проект	Если для добавления файла в проект используется обозреватель решений, файл фильтров определяет, в какое место дерева обозревателя решений будет добавлен файл, по расширению имени файла.
VDPROJ	Проект	Файл развертывания проекта Visual Studio.
VMX	Проект	Файл проекта макроса.
VUP	Проект	Вспомогательный файл проекта.

Файлы проекта распределены по папкам в обозревателе решений. Visual C++ создает папку для файлов исходного кода, файлов заголовков и файлов ресурсов, но вы можете изменить структуру этих папок и создать новые. С помощью папок можно явно создавать логические группы файлов в иерархии проекта. Например, можно создать папки, в которых будут храниться все файлы исходного кода, спецификаций, документации или наборов тестов для интерфейсов. Имена папок должны быть уникальными.

Когда элемент добавляется в проект, он входит во все конфигурации данного проекта, вне зависимости от того, подлежит этот элемент построению или нет. Например, если добавить элемент в проект с именем *MyProject*, то этот элемент появится также в отладочной (**Debug**) и окончательной (**Release**) конфигурациях проекта.

В начало

Создание проектов для рабочего стола с помощью мастеров приложений

Для каждого типа проектов Visual C++ имеется мастер приложений, помогающий быстро и легко создавать проекты по шаблону. Чтобы открыть мастер приложений,

воспользуйтесь диалоговым окном **Новый проект** и укажите в нем свойства проекта, такие как имя проекта и каталог или решение, в котором будет размещен проект.

Открытие мастера приложений Visual C++

1. В меню Файл последовательно выберите пункты **Создать** и **Проект**. Откроется диалоговое окно **Новый проект**.
2. В области **"Типы проектов"** выберите папку **Проекты** Visual C++. В области **"Шаблоны"** появится значок для каждого типа проектов C++.
3. В области **"Шаблоны"** щелкните значок нужного типа проекта. Под обеими областями появится сообщение с указанием типа проекта, который будет создан.
4. Задайте свойства проекта или пропустите этот шаг, чтобы использовать заданные по умолчанию свойства проекта Visual Studio.
5. Нажмите **ОК**. Откроется мастер проектов выбранного типа. Если вы хотите прочесть раздел справки для мастера, нажмите клавишу **F1**.

При создании проекта можно указать новое решение или добавить проект в существующее решение.

В начало

Добавление функциональных возможностей с помощью мастеров кода

После создания проекта необходимо изменить или расширить его функциональные возможности. В эти задачи входит создание новых классов, добавление новых функций-членов и переменных, а также добавление методов и свойств автоматизации. Для выполнения этих действий предназначены мастера кода.

Доступ к мастерам кода Visual C++ можно получить тремя способами (см. табл. 2).

Таблица 2. Способы доступа к мастеру кода Visual C++

Доступ к мастеру кода	Описание
Добавить новый элемент	Мастера кода **"Добавление нового элемента"** позволяют добавлять в проект файлы исходного кода. При необходимости создаются дополнительные каталоги для хранения файлов, где эти файлы будут находиться подсистемой построения проектов. Ниже перечислены мастера кода, доступные при нажатии значка **"Добавление элемента"**. ▪ Добавление файлов исходного кода C++ (.cpp, .h, .idl, .rc, .srf, .def, .rgs). ▪ Добавление файлов веб-разработки (.html, .asp, .css, .xml). ▪ Добавление файлов служебных программ и ресурсов (.bmp, .cur, .ico, .rct, .sql, .txt). Как правило, данные мастера не запрашивают у пользователя какие-либо сведения, а просто добавляют файл в дерево

	разработки. Файл можно переименовать в окне "Свойства".
Обозреватель решений	Мастера кода, доступные в обозревателе решений, зависят от положения фокуса курсора при щелчке правой кнопкой мыши по элементу. Если при щелчке правой кнопкой мыши по элементу не появляется пункт **Добавить**, сместите положение курсора в дереве разработки на одну позицию вверх и повторите попытку. Мастера кода всегда помещают дополнительный код в соответствующее место дерева разработки вне зависимости от положения курсора. Ниже перечислены мастера кода, доступные в обозревателе решений. ▪ Добавление класса (открывается диалоговое окно Добавление класса, содержащее новые мастера кода). ▪ Добавление ресурса ("Создать", "Импорт" или "Настраиваемый"). ▪ Добавление веб-ссылки.
Окно классов	Мастера кода, доступные в окне классов, зависят от положения фокуса курсора при щелчке правой кнопкой мыши по элементу. Если при щелчке правой кнопкой мыши по элементу не появляется пункт **Добавить**, сместите положение курсора в дереве классов на одну позицию вверх и повторите попытку. Мастера кода всегда помещают дополнительный код в соответствующее место дерева разработки вне зависимости от положения курсора. Ниже перечислены мастера кода, доступные в окне классов. ▪ Добавить функцию-член. ▪ Добавить переменную-член. ▪ Добавить класс. ▪ Реализовать интерфейс (только из класса элементов управления) ▪ Добавить точку подключения (только для классов ATL) ▪ Добавить метод (только из интерфейса) ▪ Добавить свойство (только из интерфейса) ▪ Добавить событие (только из класса элемента управления) При выборе команды **"Добавить класс"** открывается диалоговое окно **Добавление класса**, предоставляющее доступ ко всем мастерам кода **"Добавление класса"**.

В начало

Создание решений для нескольких проектов

Можно создать решение для нескольких проектов, добавив один или несколько проектов в существующее решение.

При создании решения, включающего несколько проектов, первый созданный проект по умолчанию становится автоматически загружаемым проектом. Автоматически загружаемый проект выделяется в **Обозревателе решений** полужирным шрифтом и запускается при выборе в строке меню **Отладка**, **Пуск**. Отладку можно выполнять для всех проектов решения либо для одного или нескольких проектов, выбрав решение в качестве запускаемого проекта. Структура физической связи файлов решения устанавливается во время создания файла или проекта.

Добавление и удаление проектов в решении

Можно создать проект и добавить его в решение или добавить существующий проект в решение. Решения можно удалять только в проводнике, а не в Visual Studio.

Создание нового проекта и добавление его в решение

1. В Обозревателе решений выберите решение или папку решения, куда требуется добавить создаваемый проект.
2. В строке меню выберите **Файл**, **Добавить**, **Создать проект**.
3. В левой области выберите **Установленные**, а в развернутом списке – категорию типов проектов.
4. В средней области в разделе **Шаблоны** выберите один из шаблонов проекта.
5. В поле **Имя** введите имя нового проекта и нажмите кнопку **ОК**.

В решение можно добавить существующий проект, а затем изменить этот проект в соответствии с требованиями этого решения.

Добавление существующего проекта в решение

1. В Обозревателе решений выберите решение, к которому нужно добавить проект.
2. В контекстном меню выберите **Добавить**, **Существующий проект** и выберите проект, который требуется добавить в решение.

Удаление решения

1. Удалите все проекты в решении, которые, возможно, потребуется использовать снова. Можно вырезать каталог проекта из папки решения и вставить его в другое место. Может потребоваться настроить несколько ссылок.
2. Удалите каталог решения.

В начало

Сборка и запуск проекта

Построение и запуск проекта

1. В строке меню последовательно выберите **Сборка** и **Собрать решение**.
2. Чтобы запустить проект, в строке меню выберите **Отладка**, **Запуск без отладки**.

В начало

Задание

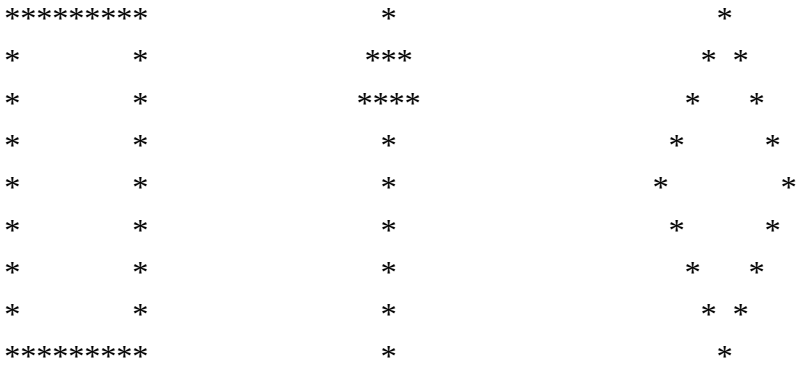
1. Создайте пустое консольное приложение в MS Visual Studio 2019.

2. Добавьте файл «main» с расширением .cpp в папку «Исходные файлы», в котором будет располагаться основная функция main с телом Вашей программы.
3. Добавьте в файл main.cpp следующий код:

```
#include <iostream>                          // 1
using namespace std;                         // 2
void main()                                  // 3
{                                            // 4
   cout << "Hello, world!" << endl;           // 5
}                                            // 6
```

   Пояснения к программе:
   Первым в программе стоит символ `#`, который служит сигналом для препроцессора. При каждом запуске компилятора запускается и препроцессор. Он читает исходный текст программы, находит строки, которые начинаются с символа `#`, и работает с этими строками до того, как начнется компиляция программы; `include` – это команда препроцессору, включающая заголовочный файл, в котором описаны типы данных, константы, структуры, объекты, классы и т.д., используемые в основной программе. `iostream` – название для заголовочного файла `iostream.h`, известное в пространстве имен `std` (пространство имен стандартной библиотеки языка C++), которое и подключается во второй строке программы. В третьей строке программы – начало описания функции `main`. Функция `main` – точка входа в основную программу! Название этой функции менять нельзя! В четвертой и шестой строке программы – операторные скобки, в данном случае – начало и конец функции `main`. В пятой строке используется объект `cout` (читается "си-аут") из заголовочного файла iostream.h, осуществляющий вывод строки `"Hello, world!"` на консоль. `<<` – операция помещения в поток (т.е. на экран). Существует ряд специальных символов (управляющих последовательностей), применяемых для флорматирования строк. Например, \n – переход на новую строку. Например, код `cout << "Hello,\nworld!" << endl`; выведет на экране две строки. Первая – `"Hello,"`, вторая `"world!"`.
4. Запустите Ваше приложение.
5. Добавьте в имеющееся решение новый проект.
6. Продублируйте пп. 2-3 с новым проектом.
7. Назначьте новый проект исполняемым (ПКМ по решению, **Свойства**, **Запуск**, Выбрать запускаемый проект).
8. Запустите новый проект.

9. В основной программе второго проекта, используя объект `cout`, выведите следующее:



Контрольные вопросы

1. Назовите типы проектов, которые можно создать в MS Visual Studio 2019.
2. Назовите типы файлов, относящихся к категории исходных файлов.
3. Как можно запустить проект на выполнение?
4. Раскройте алгоритм создания нескольких проектов в одном решении.
5. Каким образом можно назначить исполняемым тот или иной проект в решении?
6. Что печатает следующий оператор?

```
cout << "*\n**\n***\n****\n*****\n";
```

Список литературы

https://msdn.microsoft.com/ru-ru/library/4457htyc.aspx

https://msdn.microsoft.com/ru-ru/library/h970wzkb.aspx

В начало