

Создание многофайловых проектов



Их компиляция и сборка. Работа с заголовочными файлами

Немного информации

Мы уже познакомились с проектами. Поняли зачем они нужны, как их создавать и как с ними работать. Однако мы, до данного момента, использовали лишь один файл в котором всё и реализовывали (файл `main`). Сделаем условие, что все файлы мы будем сохранять с расширением `*.cpp`, то есть, будем работать с языком C++. Некоторые вещи чистый СИ не поддерживает. Я буду говорить об этом.

Пример программы (переписать в IDE)

До этого мы рассматривали программу перевода из дюймов в см. Создали программу, которая содержит функцию, в которой реализован весь функционал, отвечающий за перевод в дюймы. Научились передавать в функцию определённые данные, посредством передачи фактических параметров при вызове функции.

```
# include<stdio.h>
#include <locale.h>
float f1(float , int );
int main()
{
    setlocale(0,"RUS");/*напоминаю, чтобы это нормально работало, сохраняем файл *.cpp можно
    и с utf setlocale(LC_ALL, "ru_RU.UTF-8");*/

    float in;
    int af;
    printf("Введите размер в дюймах in=");
    scanf("%f",&in);
    printf("Введите англ-1, фран-2");
    scanf("%d",&af);
    printf("Размер в см равен %f", f1(in,af) );
}
float f1(float x, int y )
{ float sm;
  if(y==1)
  {
      sm=x*2.54;
      return sm;
  }
  if(y==2)
  {
      sm=x*2.72;
      return sm;
  }
  else
  {
      printf("Не могу подсчитать");
      return 0;
  }
}
```

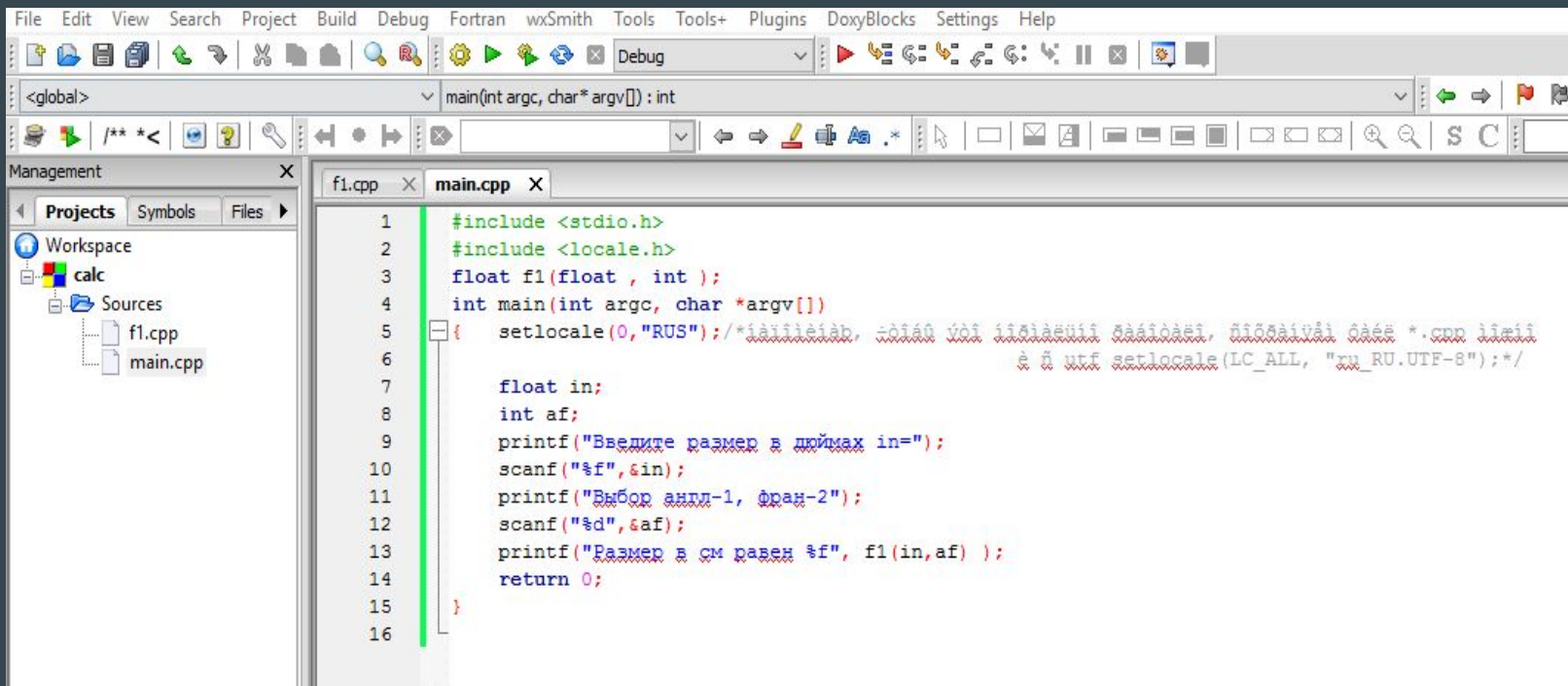
Создание многофайловых проектов

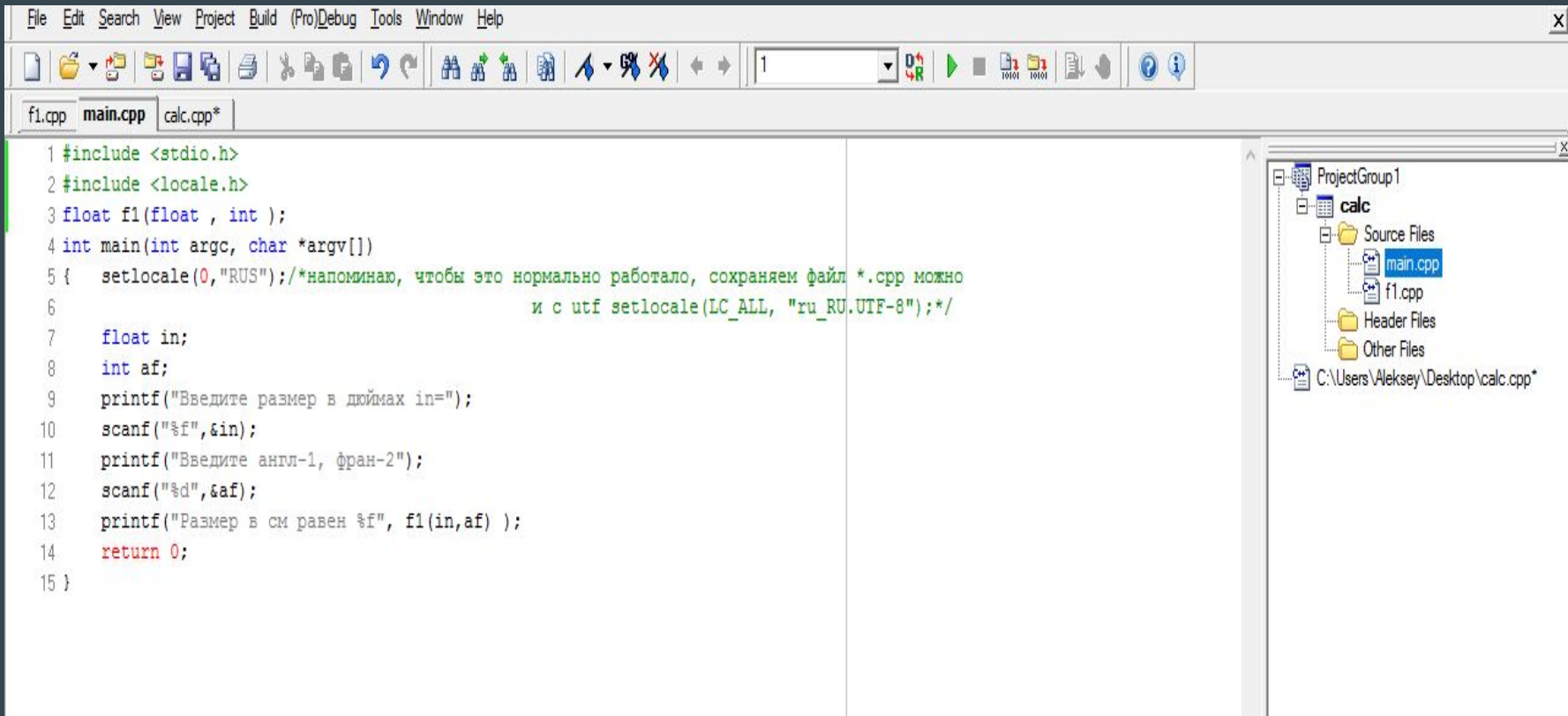
Зачем это нужно?

Как реализовать?

Общий вид программы из нескольких файлов

Для удобства понимания сделал проекты в C-free и code blocks



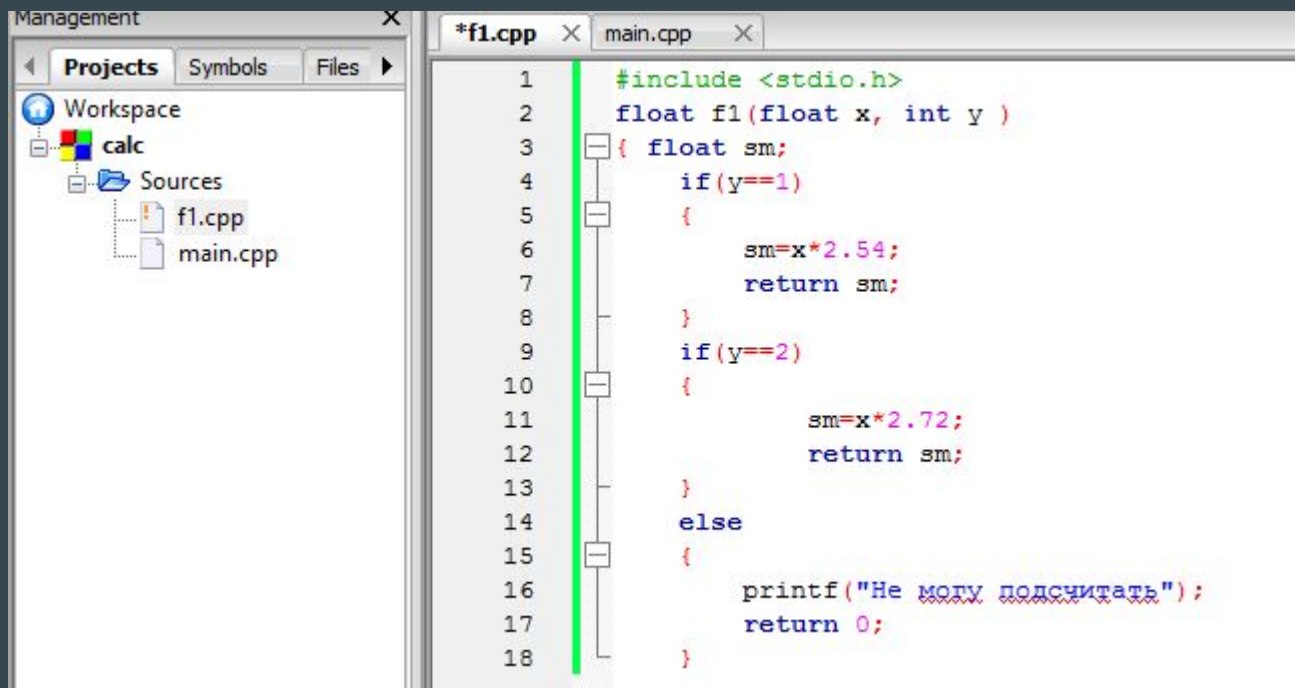


Компиляция

Все созданные файлы в проекте можно откомпилировать отдельно. Создано это для удобства. Но обязательно должен быть один файл, который содержит функцию `main()`.

Создайте ещё один файл и разместите его в проекте. Данный файл будет содержать в себе реализацию функции `f1()`. То есть, разместите в нём исходный код функции.

Вопрос: Почему я добавил подключение заг.файла?



The image shows a screenshot of an IDE. On the left, a 'Management' window displays a project structure under 'Workspace'. A folder named 'calc' contains a 'Sources' folder, which in turn contains two files: 'f1.cpp' and 'main.cpp'. On the right, the main editor window shows the code for '*f1.cpp'. The code includes a header file, defines a function 'f1', and contains conditional logic based on the value of 'y'.

```
1  #include <stdio.h>
2  float f1(float x, int y )
3  { float sm;
4      if(y==1)
5      {
6          sm=x*2.54;
7          return sm;
8      }
9      if(y==2)
10     {
11         sm=x*2.72;
12         return sm;
13     }
14     else
15     {
16         printf("Не могу подсчитать");
17         return 0;
18     }
```

Файл main.cpp

```
1  #include <stdio.h>
2  #include <locale.h>
3  float f1(float , int );
4  int main(int argc, char *argv[])
5  {  setlocale(0,"RUS");/*iAiiieiab, -oiA0 voi iAiaAeuii A
6
7      float in;
8      int af;
9      printf("Введите размер в дюймах in=");
10     scanf("%f",&in);
11     printf("Выбор англ-1, фран-2");
12     scanf("%d",&af);
13     printf("Размер в см равен %f", f1(in,af) );
14     return 0;
15 }
16
```

Создание заголовочных файлов

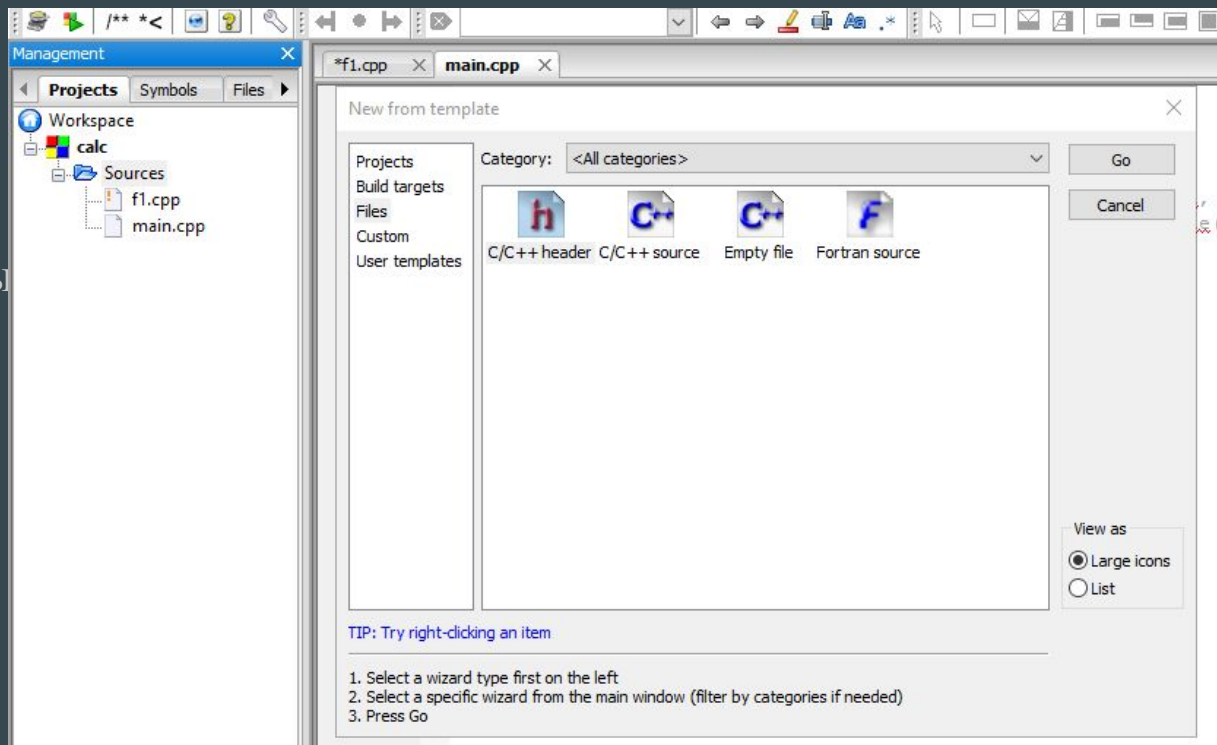
Зачем нужны? Что в них должно храниться?

Создание заголовочного файла для задачи

В Code Blocks

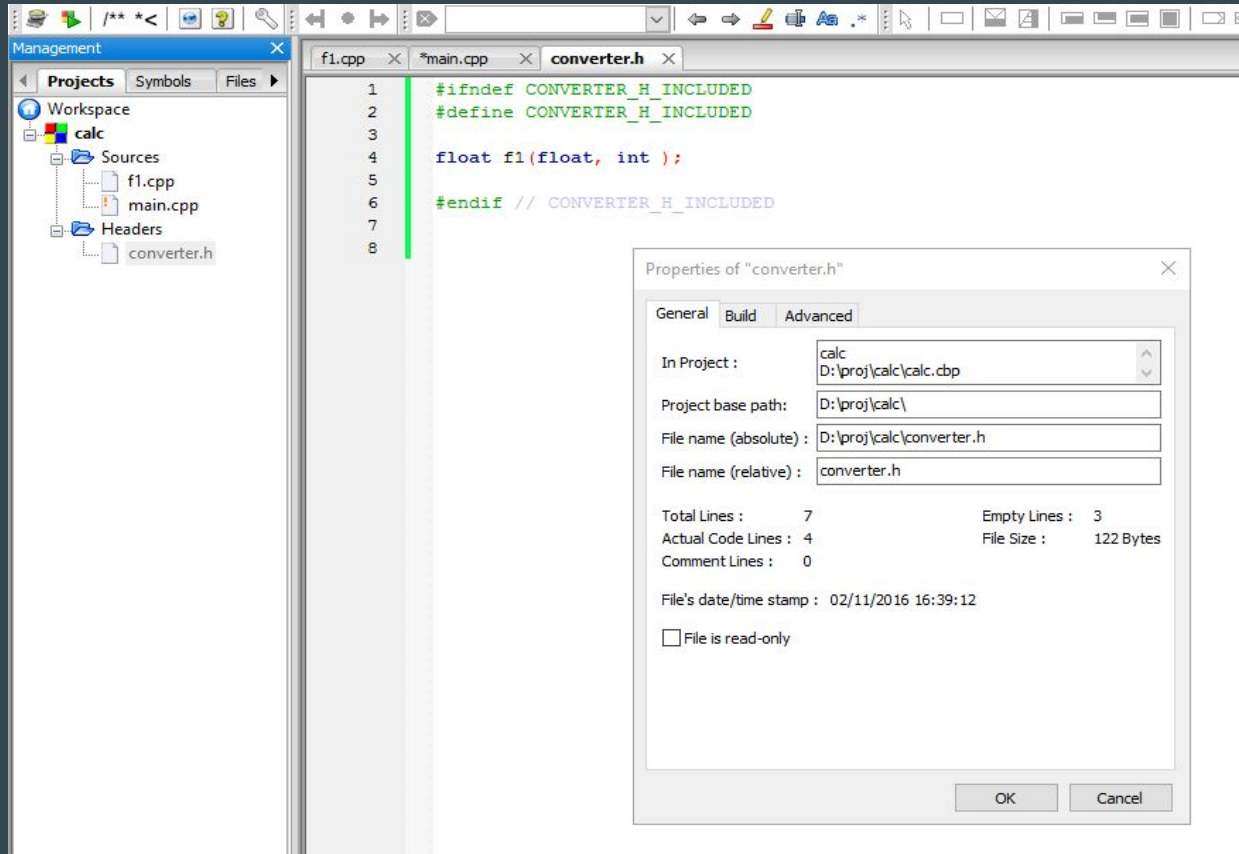
Нажимаем File->NEW->

->file. Выбираем заг.файлы

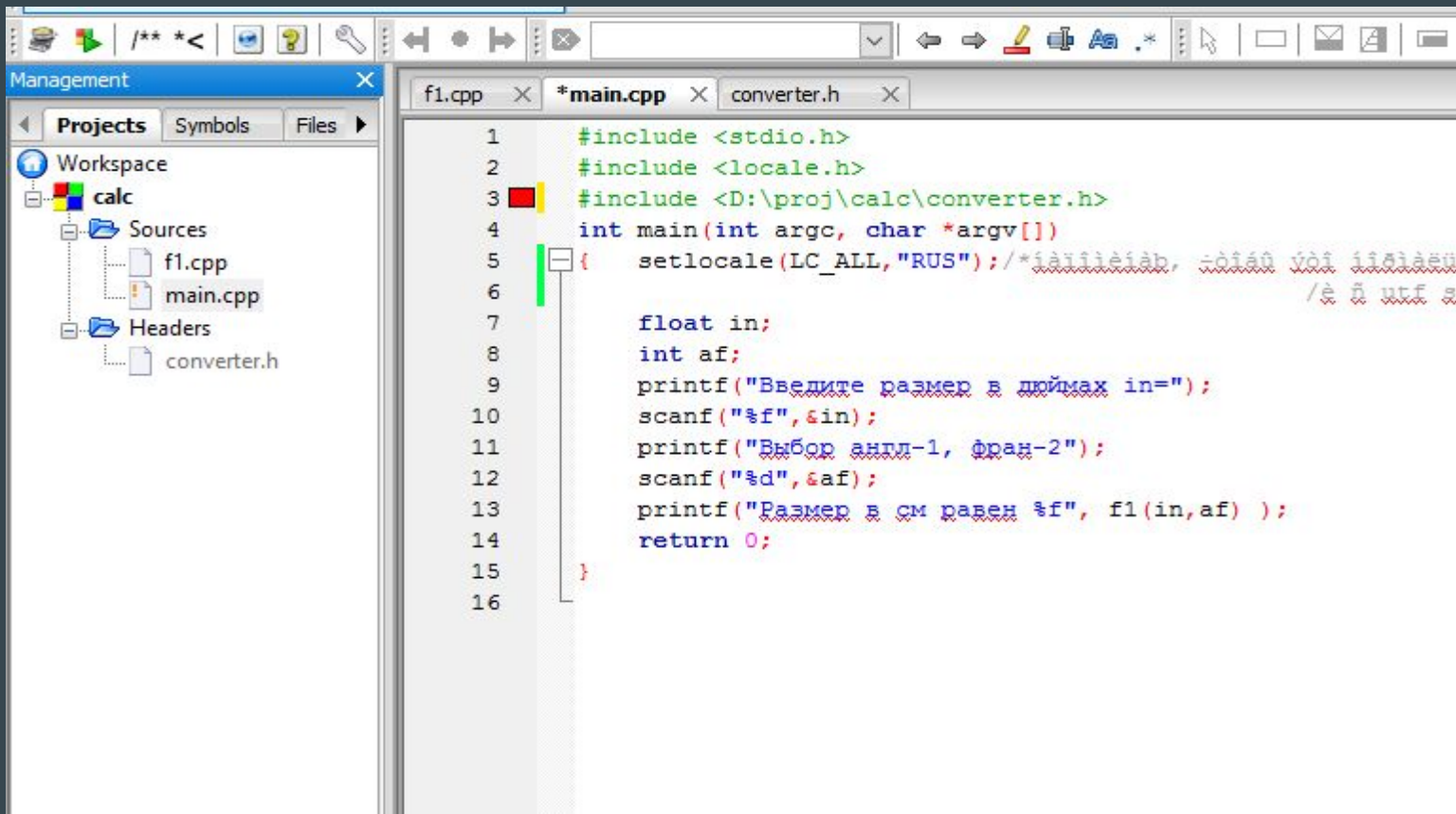


Что там размещаем

Можно размещать любой исходный код, соблюдая правила. Но принято, и так всё-таки правильно, размещать прототипы функций.



“Эталонный” проект



The image shows a screenshot of a C++ IDE. On the left, a 'Management' pane displays a project named 'calc' with a 'Sources' folder containing 'f1.cpp' and 'main.cpp', and a 'Headers' folder containing 'converter.h'. The main editor window shows the content of 'main.cpp' with the following code:

```
1  #include <stdio.h>
2  #include <locale.h>
3  #include <D:\proj\calc\converter.h>
4  int main(int argc, char *argv[])
5  {
6      setlocale(LC_ALL, "RUS"); /*смена локали
7
8      float in;
9      int af;
10     printf("Введите размер в дюймах in=");
11     scanf("%f", &in);
12     printf("Выбор англ-1, фран-2");
13     scanf("%d", &af);
14     printf("Размер в см равен %f", f1(in,af) );
15     return 0;
16 }
```

Как язык СИ допускает делать(не рекомендуется)

The image shows a screenshot of a C compiler IDE. The main window displays a C program named `converter.h` with the following code:

```
1 float f1(float x, int y )
2 { float sm;
3   if (y==1)
4     {
5       sm=x*2.54;
6       return sm;
7     }
8   if (y==2)
9     {
10      sm=x*2.72;
11      return sm;
12    }
13  else
14    {
15      printf("Не могу подсчитать");
16      return 0;
17    }
18 }
19
```

In the center, a window titled `"C:\calc\calc.exe"` is open, showing a black terminal window with the prompt `Введите размер в дюймах in=`. The right sidebar shows a project tree for `ProjectGroup1` with the following structure:

- ProjectGroup1
 - calc
 - Source Files
 - main.cpp
 - Header Files
 - converter.h
 - Other Files



```
1 #include <stdio.h>
2 #include <locale.h>
3 #include <C:\calc\converter.h>
4 int main(int argc, char *argv[])
5 {   setlocale(0,"RUS");/*напоминаю, чтобы это нормально работало, сохраняем файл *.cpp можно
6                                     и с utf setlocale(LC_ALL, "ru_RU.UTF-8");*/
7     float in;
8     int af;
9     printf("Введите размер в дюймах in=");
10    scanf("%f",&in);
11    printf("Введите англ-1, фран-2");
12    scanf("%d",&af);
13    printf("Размер в см равен %f", f1(in,af) );
14    return 0;
15 }
```

ProjectGroup1

- calc
 - Source Files
 - main.cpp
 - Header Files
 - converter.h
 - Other Files

Задача на самостоятельное решение