Инструкция по установке и настройке библиотеки и среды программирования для реализации протокола JSON RPC (C++)

Автор: Иван Лебедев

В лабораторной работе №6 предполагается осуществить локальное соединение типа Клиент-Сервер с помощью **RPC**. Сам RPC может работать на множествах протоколах. В данном случае рассмотрим реализацию на протоколе **JSON**.

Реализация метода JSON более затруднительна по сравнению с XML и может запуститься с 101 раза. Так как в этом году было запрещено выполнять работу с помощью XML, то эта методичка является отличным аналогом.

Обратите внимание, в некоторых местах Word заменяет знак минуса (-) на тире (–). Те, кто копирует отсюда код сразу в виртуалку, будьте внимательны. Также в редком случае могут возникнуть проблемы линковщика у тех пользователей, которые используют современные ноутбуки с Windows 11 и эффективными ядрами Intel на борту. Решение этой проблемы в рамках методички отсутствует.

В обновленной версии методички исправил последовательность ввода команд в терминал. Перед вводом не забываем предварительно выполнить вход через **sudo.**

Шаги выполнения:

1. **Установка виртуальной машины на Linux.** Я решил выполнить задание на **Ubuntu**, так как на Windows (сколько бы я не пробовал) всё время появляются ошибки. Подойдут виртуалки, созданные для

выполнения прошлых лабораторных работ. Чем больше оперативки поставите – тем меньше будет фризить ваш Сервер и Клиент.

2. **Установка git.**

   sudo apt update

   sudo apt install git

3. **Установка дополнительных пакетов**

   3.1. Доп пакет 1

   sudo apt update

   sudo apt install build-essential

   После завершения установки проверьте установку этих пакетов, проверив версию GCC в вашей системе с помощью следующей команды:

   gcc –version

   3.2. Доп пакет 2

   sudo apt-get update -y

   sudo apt-get install -y libmicrohttpd-dev

   3.3. Доп пакет 3

   sudo apt install llvm

   3.4. Доп пакет 4

   sudo apt install clang

   3.5. Доп пакет 5

   apt-get install build-essential libgl1-mesa-dev

   3.6. Доп пакет 6

   apt-get install libqt5webkit5-dev

3.7. Доп пакет 7

sudo apt install libcurl-openssl1.0-dev

3.8. Доп пакет 8 (Вроде бы необязательно, но попробуйте, см. п. 7)

sudo apt install gcc

4. **Скачивание и установка библиотеки.** Ссылка на github: https://github.com/cinemast/libjson-rpc-cpp/tree/master. Данная ссылка ведёт на главную страницу библиотеки. Что-либо делать сейчас с этой ссылкой не нужно, она пригодится нам далее.

4.1. Установка фреймворка

sudo apt-get install libjsonrpccpp-dev libjsonrpccpp-tools

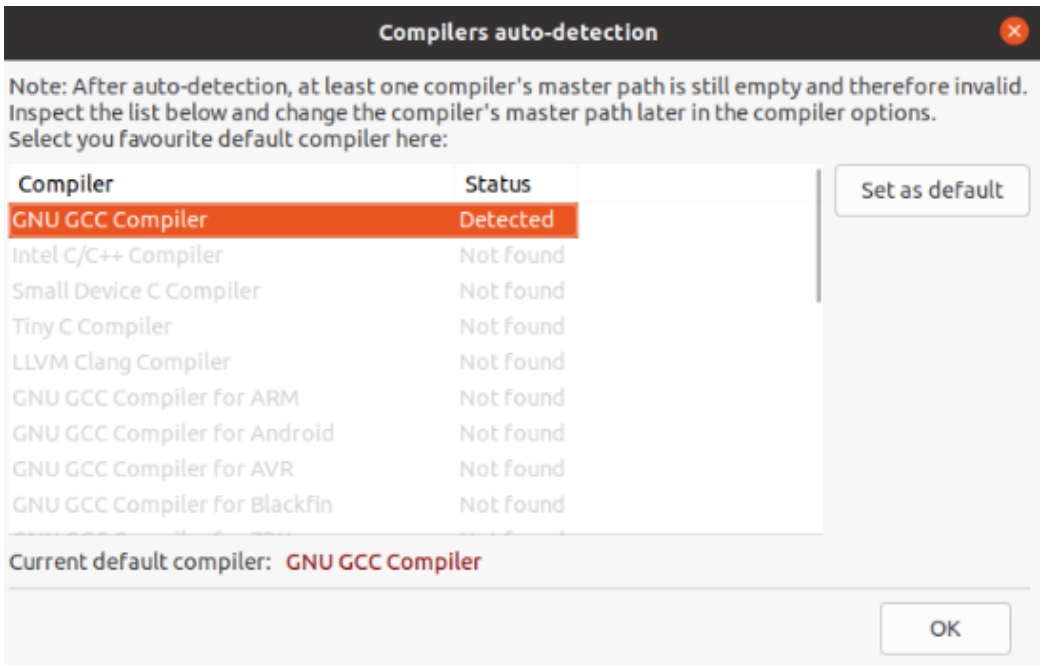
Важно! Запомните путь, куда установилась ваша библиотека (далее пригодится). По умолчанию у меня она установилась в: /usr/lib/x86_64-linux-gnu

5. **Установка Code::Blocks.** В качестве среды разработки для C++ вы можете установить любую удобную вам, я выбрал самую популярную.

   sudo apt update

   sudo apt install codeblocks

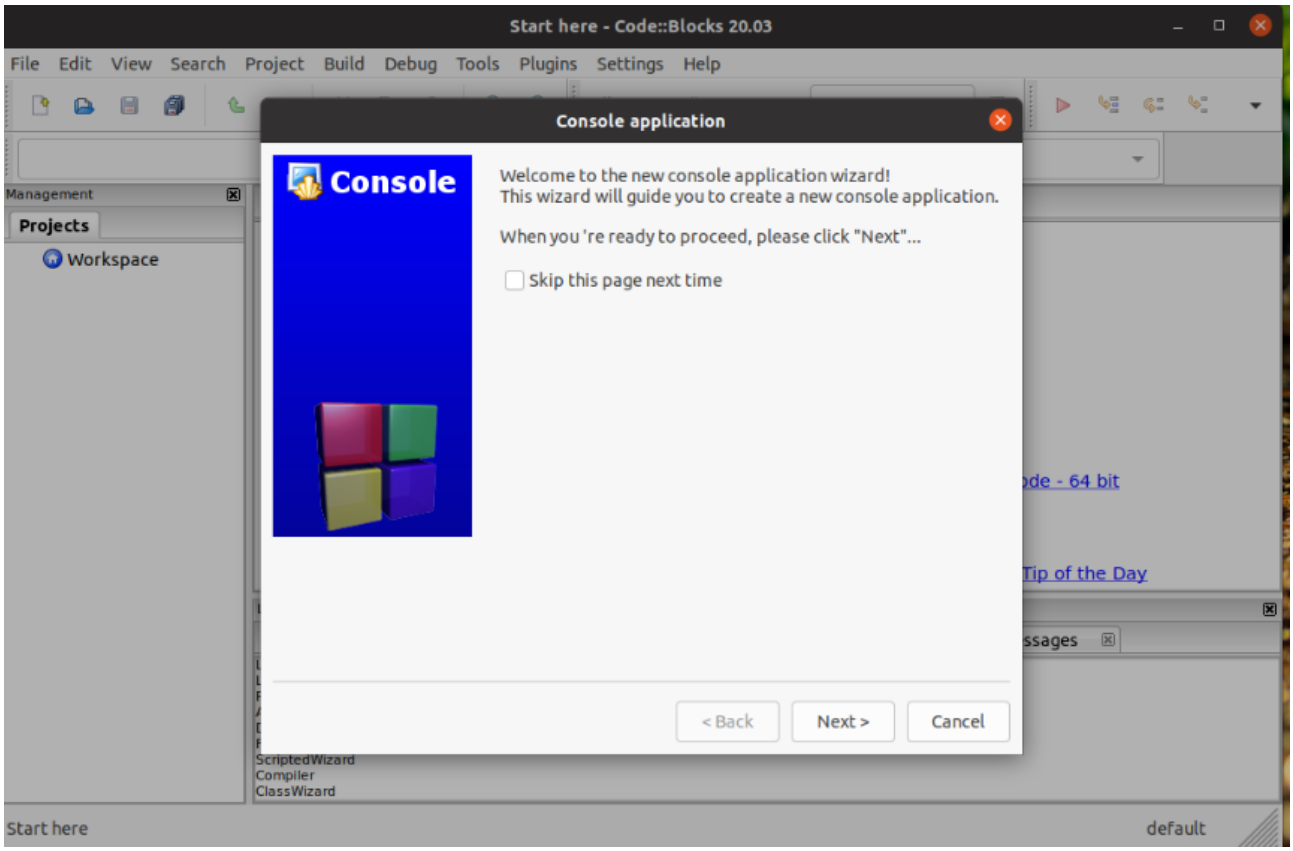
6. **Запуск Code::Blocks.**

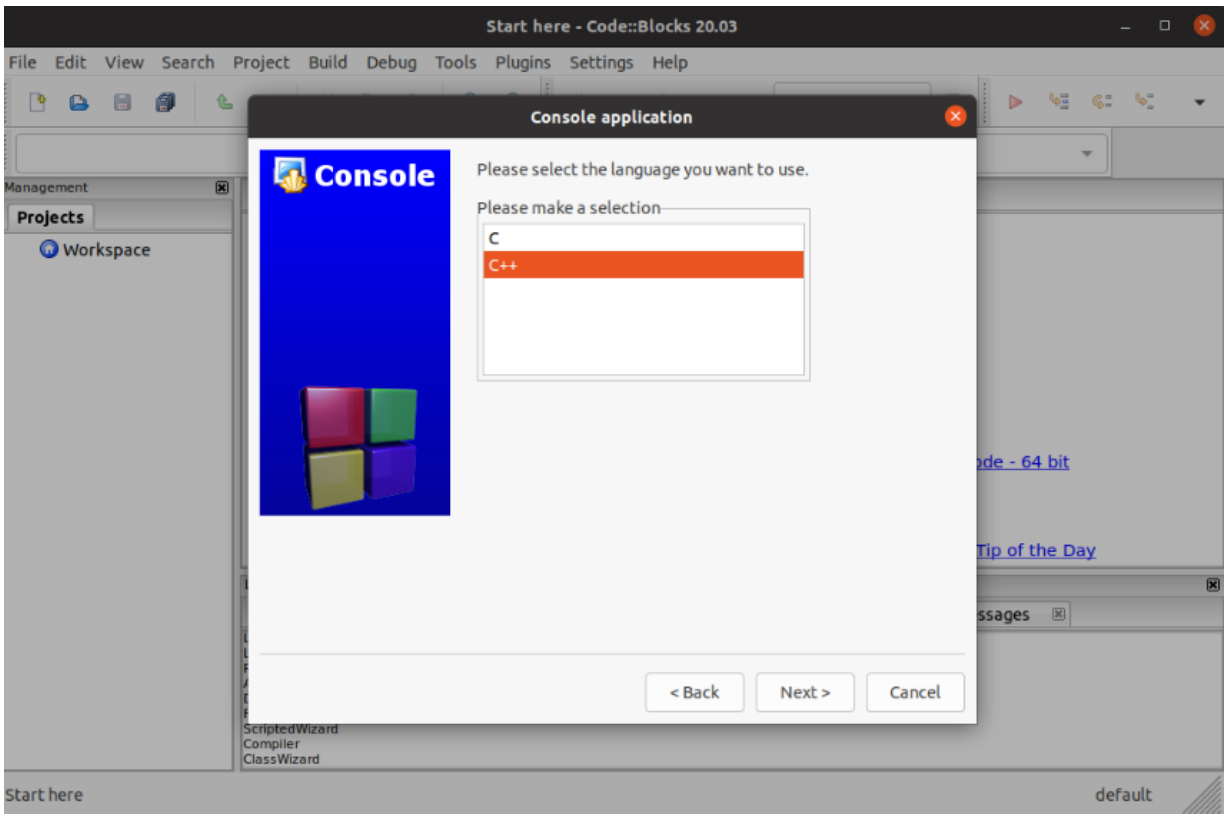


Внимание! Компилятор уже должен заранее присутствовать в системе. Если при первом запуске программы в списке нету ни одного компилятора на выбор, необходимо его установить. Я использовал компилятором **GCC** (считается стандартом для Ubuntu).

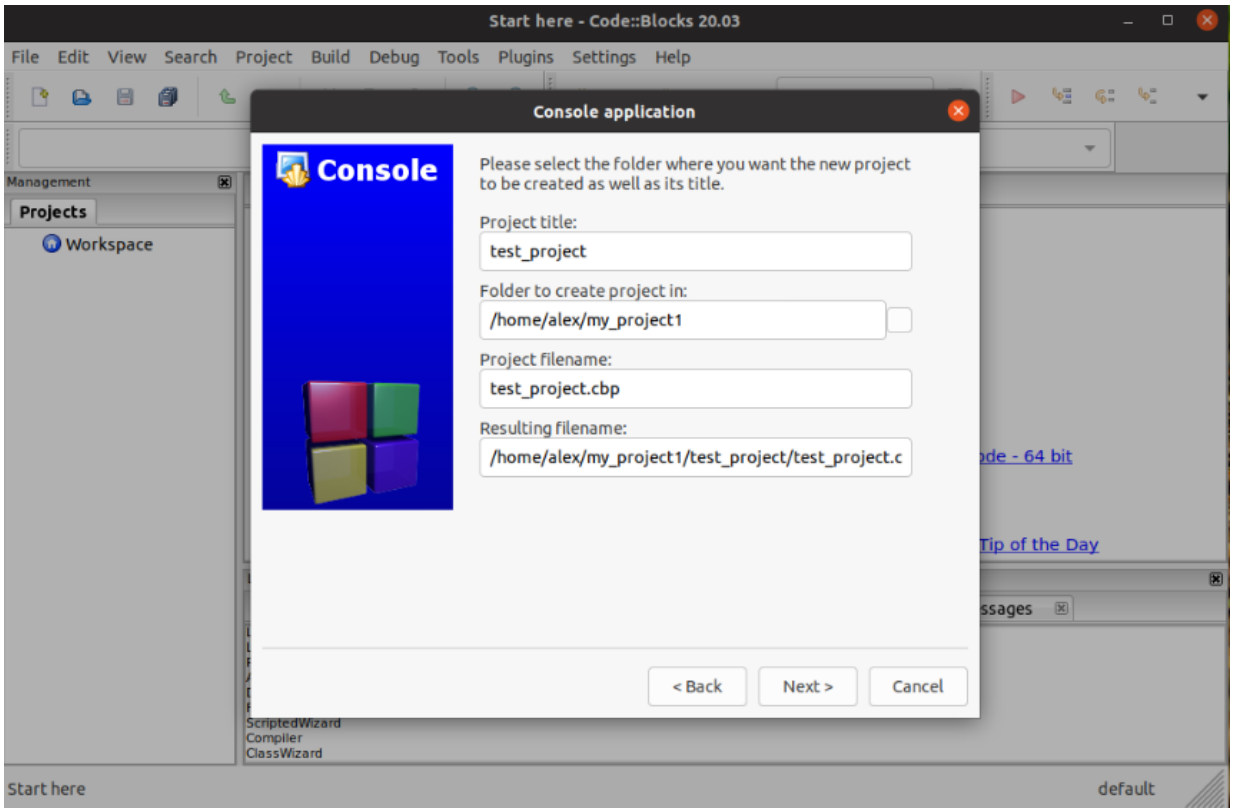
В рамках этой инструкции рассматривать подробно **установку GCC** не буду, так как я уже не помню, был ли он у меня по умолчанию, или я его докачивал отдельно. В рамках доп пакета 8 он уже должен быть скачан. Если что-то пошло не так, то попробуйте его скачать другим способом, затем закройте и откройте заново Code::Blocks, компилятор должен появиться в списке.

Далее создаём проект. В левом верхнем углу жмем: file -> create project -> console project (аналогично созданию в Visual Studio)



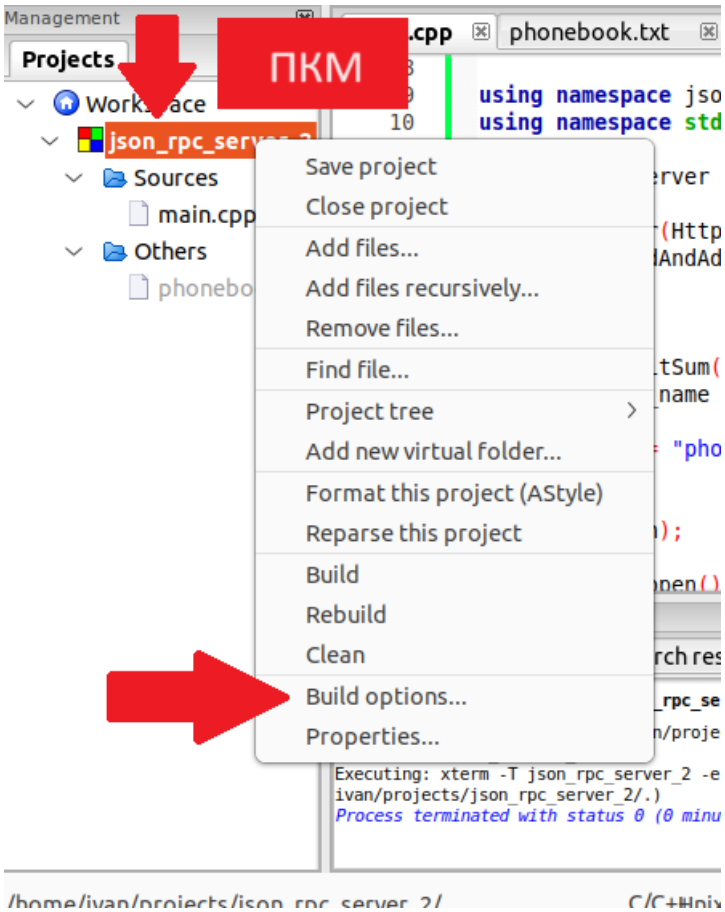


Создаем директорию наших проектов. Рекомендую в папке пользователя создать папку **projects**, в которой будут располагаться проект сервера и клиента.



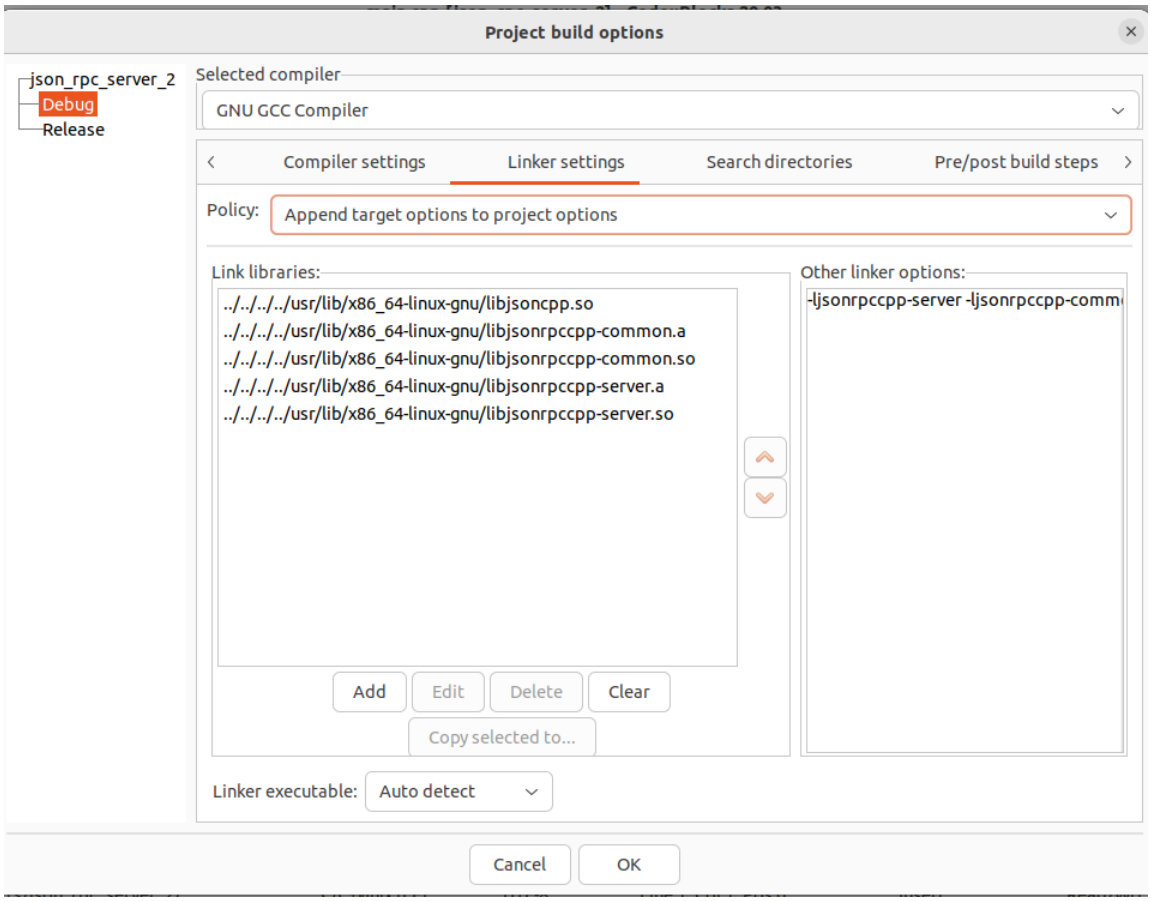
Отлично, вы создали проект (например, сервера). Далее следует открыть Code::Blocks в новой вкладке (как это сделать, показано в конце) и создать второй проект (клиент).

7. **Подключение заголовочных файлов.** Мы создали проект сервера и клиента. Далее необходимо подключить заголовочные файлы библиотеки. **Выполнить это нужно в обоих проектах**. Для этого переходим в **Build Options** вашего проекта:



7.1. Сервер

В левой колонке нажимаем **Add** и указываем 5 файлов из директории библиотеки, путь которой мы запомнили выше.



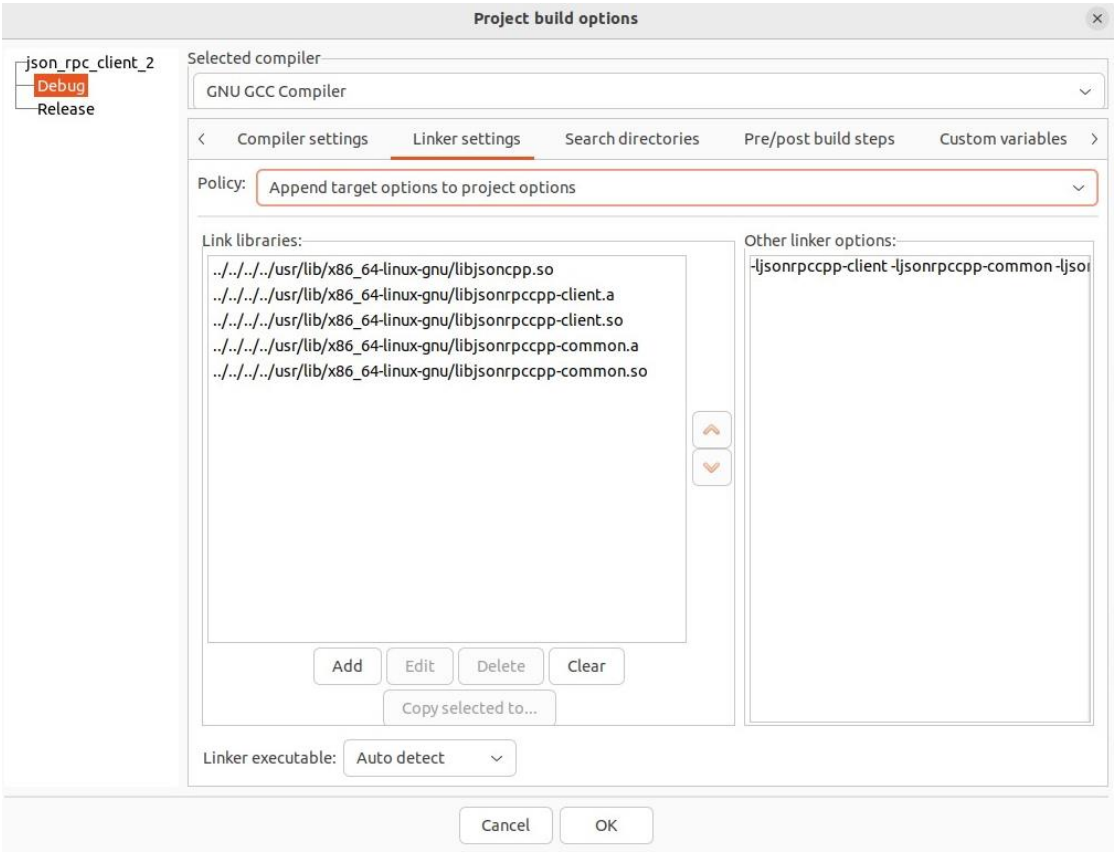
В правой колонке прописываем:

-ljsonrpccpp-server -ljsonrpccpp-common -ljsoncpp

Нажимаем **OK**.

7.2. Клиент

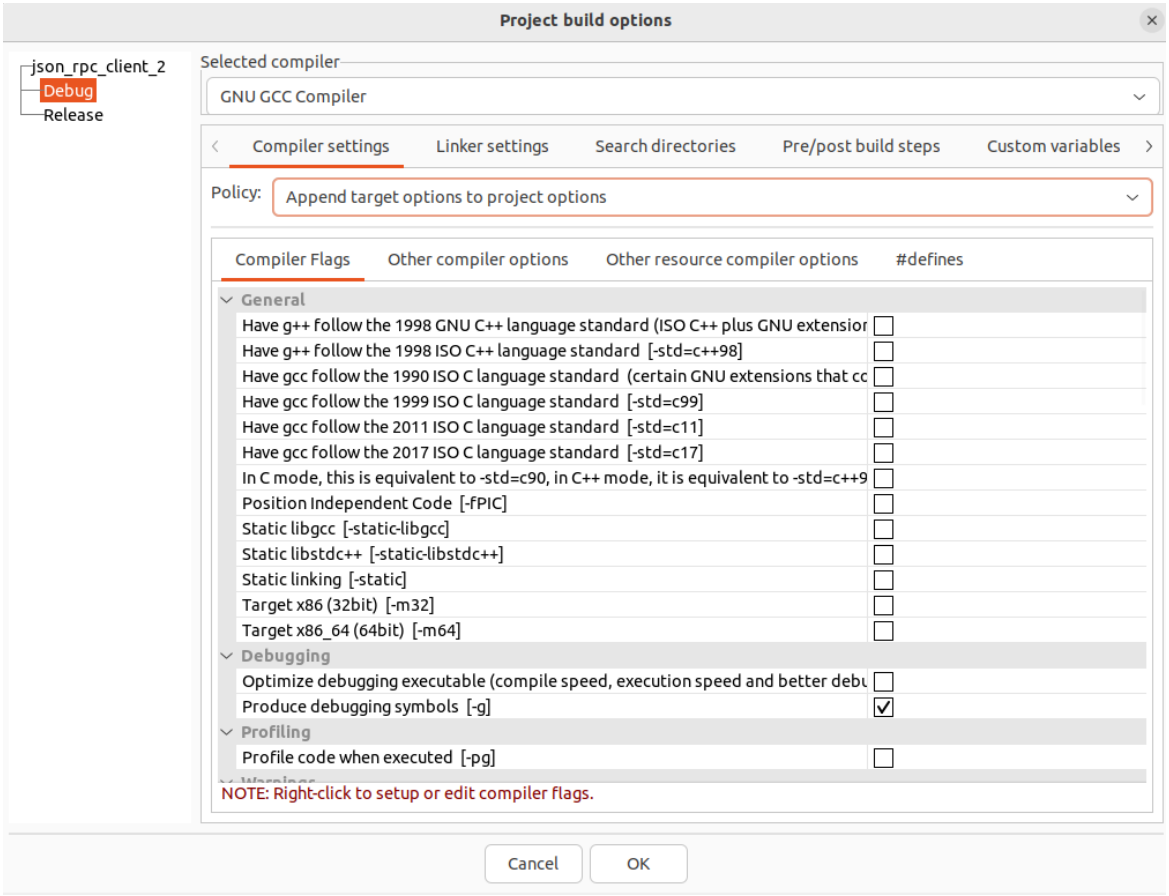
В левой колонке нажимаем **Add** и указываем 5 файлов из директории библиотеки, путь которой мы запомнили выше.



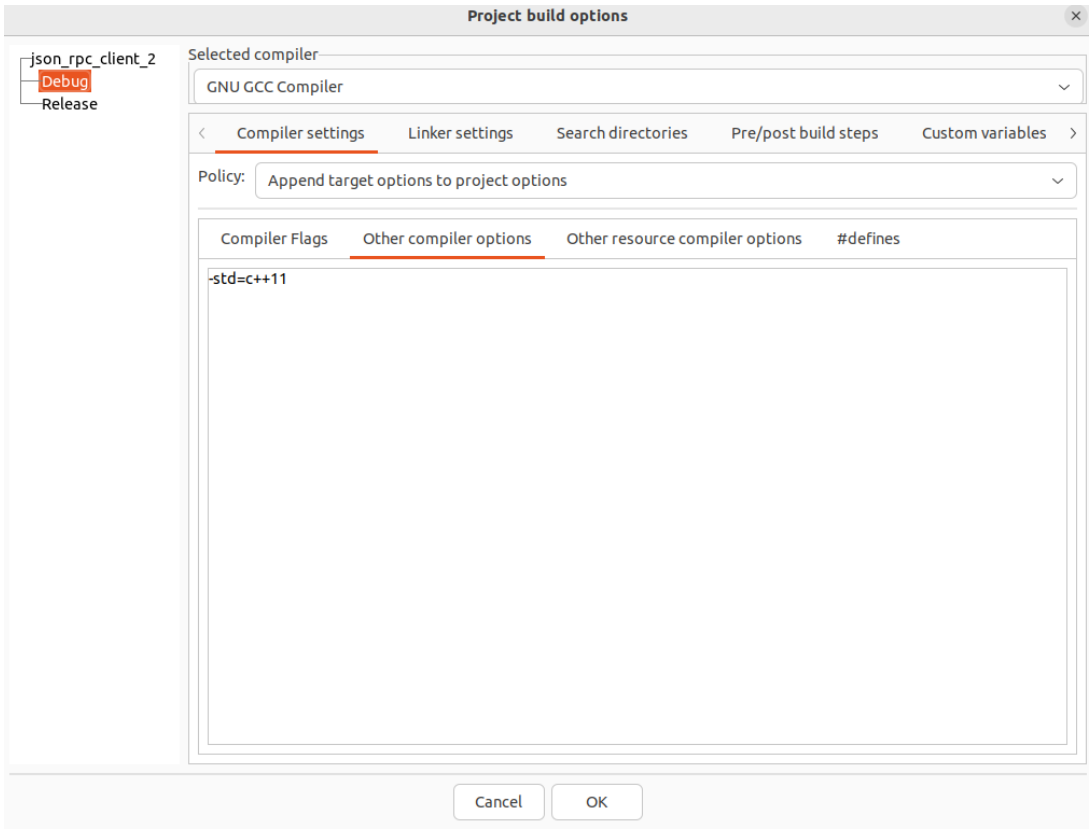
В правой колонке прописываем:

-ljsonrpccpp-client -ljsonrpccpp-common -ljsoncpp

Это ещё не всё. Далее переходим в **Compiler setting** и проверяем, чтобы галочки стояли как здесь:



Теперь переходим в **Other compiler options** и прописываем:



Нажимаем **OK**.

Поздравляю, вы выполнили все шаги по установке библиотеки и настройки Code::Blocks. Чтобы убедиться, что всё установлено правильно, необходимо из официальной документации библиотеки вставить код примера реализации простого сервера и клиента.

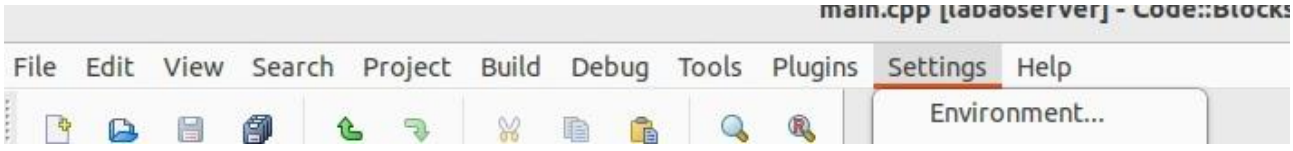
Ссылка на код сервера: https://github.com/cinemast/libjson-rpc-cpp/blob/master/src/examples/simpleserver.cpp.

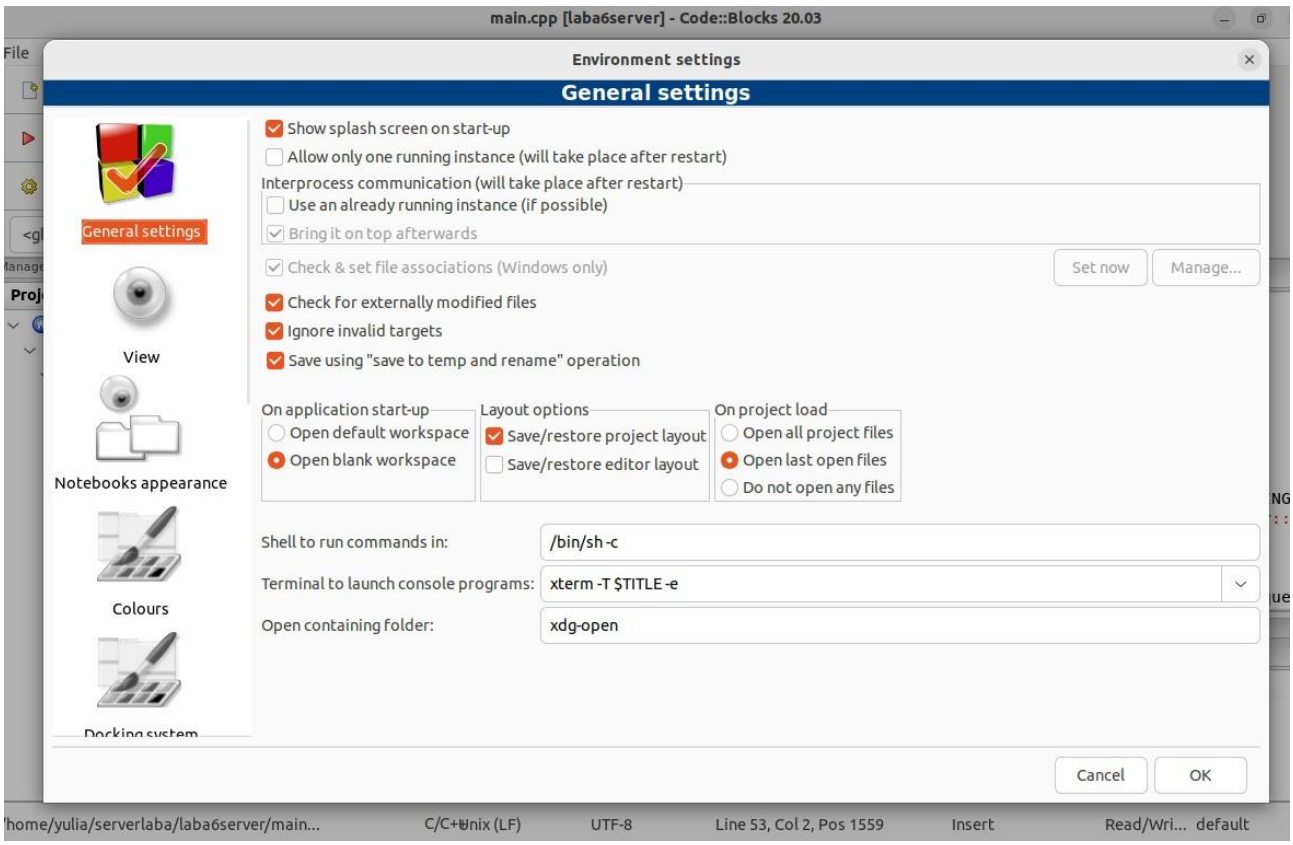
Ссылка на код клиента: https://github.com/cinemast/libjson-rpc-cpp/blob/master/src/examples/simpleclient.cpp.

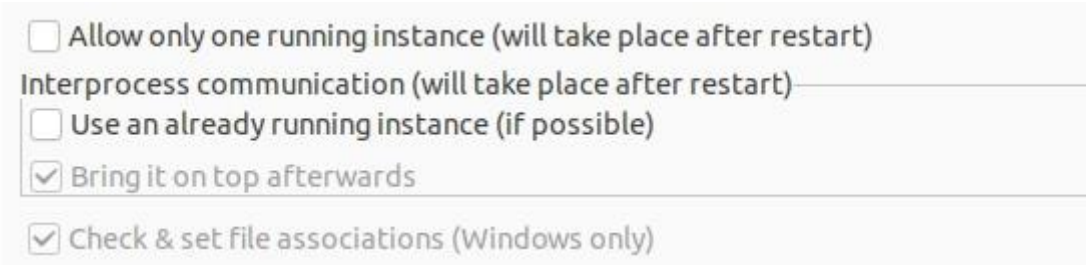
Вставляем код в компилятор. Запускаем сначала сервер, потом клиент. *(Изначально в одном Workspace нельзя запускать 2 проекта одновременно. Об этом см. пункт ниже)*. Если всё работает исправно, то в консоли клиента будет выведено сообщение приветствия.

Чтобы открыть проект Code::Blocks в новой вкладке введите в терминал команду: codeblocks *путь до проекта*.

Второй способ (лично не проверял). Сделать так, чтобы в Workspace запускалось сразу два проекта. Убираем галочки согласно следующим скриншотам:







Всё готово. Далее пишем код сервера и клиента для своего варианта. Так как ООП нами ещё официально не изучено, то при создании кода для вашего варианта можно опираться на коды из примеров официальной документации.

P.S.:

80% процентов ошибок в ходе выполнения данной лабораторной работы можно решить с помощью Гугла или ИИ. Те самые никому неизвестные 20% ошибок были уже решены мной. Всем удачи!