Работа с WSL

Графические приложения

1)Прежде всего следуем инструкции по установке wsl с официального сайта:

https://docs.microsoft.com/ru-ru/windows/wsl/install-win10

По умолчанию wsl открывает директорию /system32 , что не есть удобно. Можно изменить ее добавлением cd *ваш путь* в .bashrc

2)WSL не поддерживает графических приложений, однако можно подключаться к виртуальному дисплею по ssh. Качаем xminghttps://sourceforge.net/projects/xming/

3)Для тестирования работы дисплея нам понадобится тестовое приложение xeyes (выводит картинку с глазами, которые следят за мышкой).

Пишем в консоль "apt-get install x11-apps" . Это установит пакет приложений, среди которых есть xeyes.

4)Каждый дисплей будет иметь свой id, который устанавливается в переменную DISPLAY. Каждый запуск wsl сбрасывает это значение. Чтобы не писать каждый раз команду "export DISPLAY=:0"

(или любое другое число вместо нуля), желательно добавить ее в конец файла .bachrc

5)Запускаем xming , "Display number" ставим в соответствии с тем id , который мы прописали в пункте 4. Выбираем one window (как пример) и пропускаем все шаги без изменений.

6) Пишем команду "xeyes" в консоль. В окошке xming должны появиться глаза.

Однако практика показывает, что ни один из режимов работы xming не может корректно работать с масштабированием окон. Например, иногда целые куски окна вылезают "за край" окна, и вытянуть мышкой их нельзя, так как такое действие в принципе не поддерживается. Для масштабирования окон понадобится оконный менеджер. Я предлагаю ставить fvwm: "apt-get installfvwm"

Теперь перед запуском графического приложения запускаем его в фоновом режиме "fvwm&" , а уже потом запускаем само приложение. Оно отобразится в некоем подобии рабочего стола, и окно можно будет масштабировать и двигать мышкой

Работа с IDE

Как уже упоминалось выше, wsl не поддерживает графических приложений, и работать с IDE из нее не получится. Использовать приведенный выше метод для запуска IDE крайне неудобно. Однако некоторые среды разработки могут исполнять код на удаленном сервере, выводя результат на экран. Таким образом, мы можем поставить обычную IDE себе на Windows, лишь настроив ее для работы с wsl как с локальным сервером, подключаемым по ssh.

1) Ставим Clionhttps://www.jetbrains.com/ru-ru/clion/. Для студентов МФТИ достаточно зарегистрироваться по физтеховской почте и получить бесплатную лицензию.

2)Следуем инструкции по настройке wslhttps://www.jetbrains.com/help/clion/how-to-use-wsl-development-environment-in-clion.html

Возможные ошибки

1)При подключении wsl к Clion по ssh могут возникнуть ошибки, если использовать root пользователя в wsl. Желательно сразу создать отдельного пользователя.

2)Если в вашем .bachrc файле прописаны любые изменения переменных окружения, их следует переписать в environment файл, так как clion запускает wsl без открытия терминала, а значит все что, написано в .bashrc файле - не исполняется.

3)В случае работы с geant4: Clion при подключении к wsl подыскивает компилятор подыскивает автоматически, что может приводить к ошибкам. В таком случае надо менять cmake файл отлаживаемого проекта , либо использовать другой компилятор. Например, возникнуть ошибки с порядком обхода заголовочных файлов, которые в моем случае решились только заменой #include_next на #include в тех местах, где возникала ошибка. При установке geant4 из этого гайда используется скрипт geant4.sh , который меняет переменные окружения. Его предлагается прописывать в .bachrc чтобы не запускать каждый раз. Однако в нашем случае такой подход выдаст ошибку(см. пункт 2), так как переменные не будут установлены. Нужно открыть скрипт, посмотреть какие переменные он меняет и прописать их вручную в environment в формате key=value. По этой же причине путь к джеанту придется указать в cmake файле - "set(Geant4_DIR *ваш путь*/GEANT4/install/lib/Geant4-10.6.2/)"