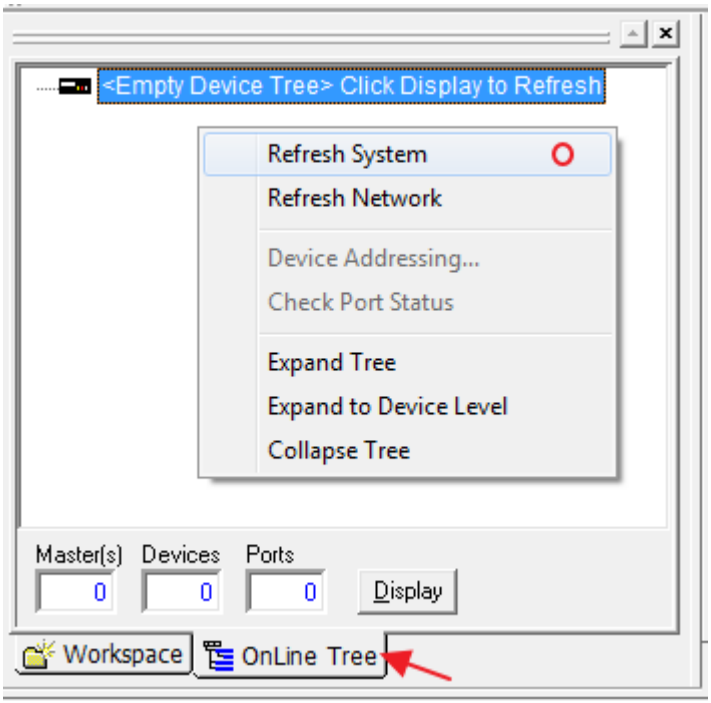
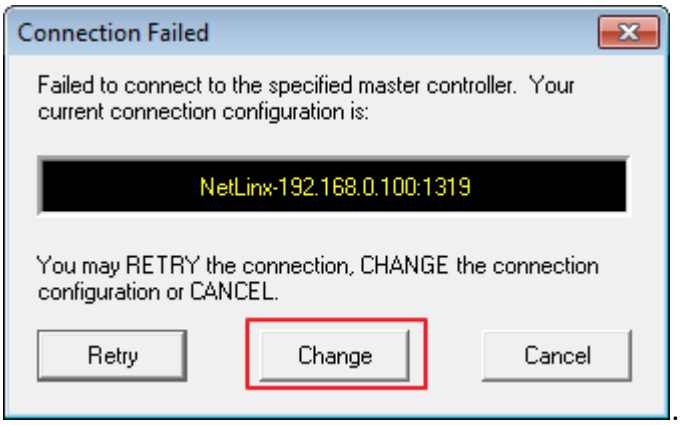
Начало работы в NetLinx Studio

Запустите NetLinx Studio,

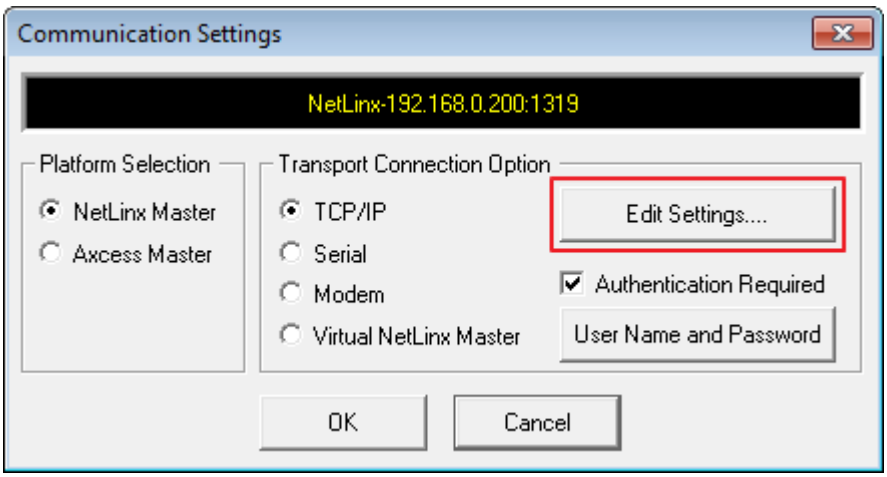
Перейдите во вкладку **OnLine Tree**, нажмите **Refresh System**




Если программа выдает ошибку соединения, откройте меню **Change** и укажите параметры подключения к своему контроллеру



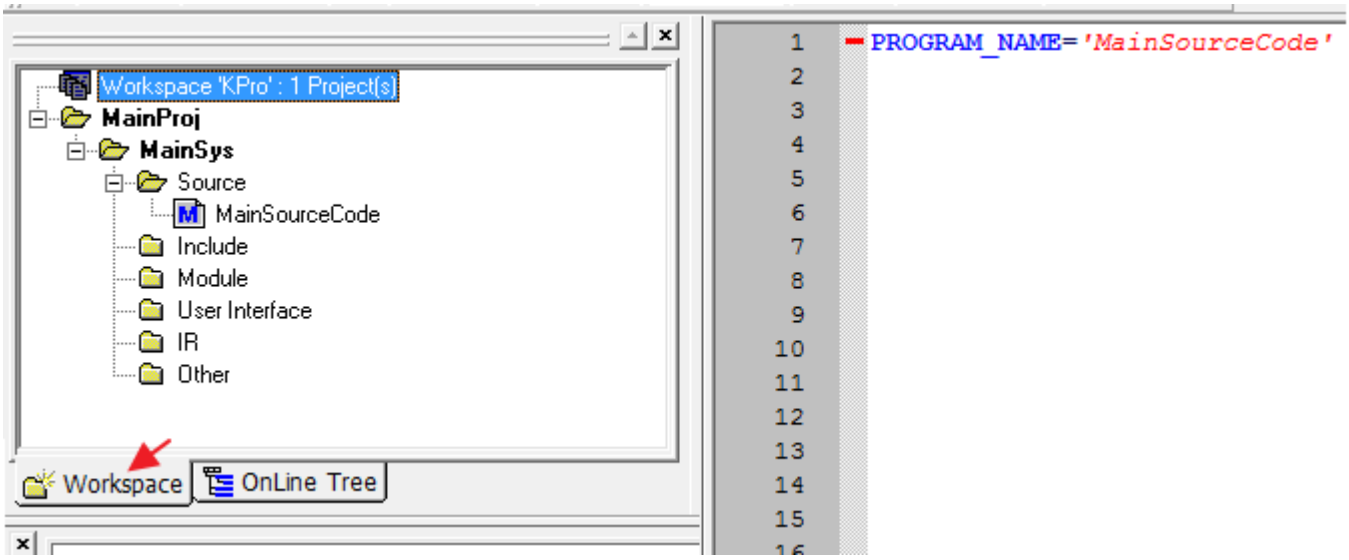

.



Откройте вкладку **Workspace**, чтобы создать рабочую зону и исходный код, который будете загружать на контроллер.

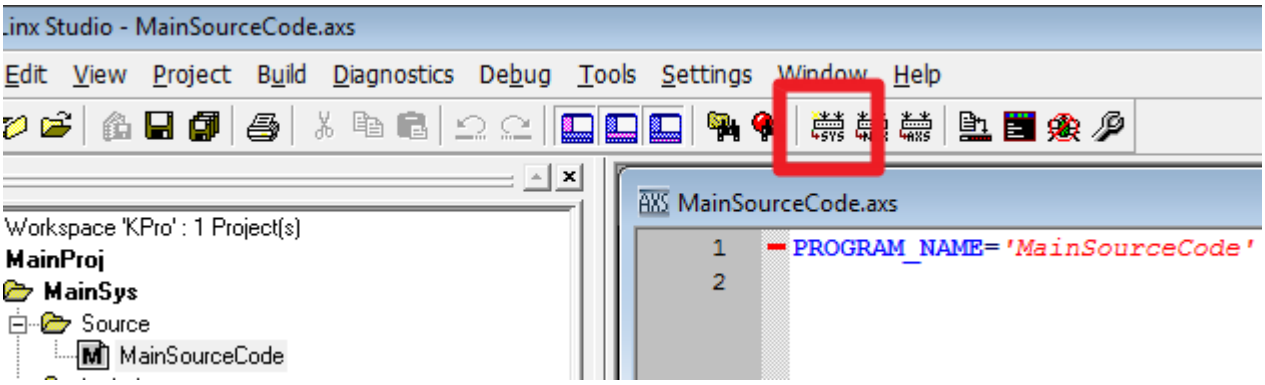
В первую очередь, создадим Workspace (**File > New > Workspace Wizard**) – это базовый элемент. В визарде создается Workspace, затем подчиненные ей System и Project (Source Code). Номер системы укажем: 1. Может быть несколько систем, проектов и рабочих зон.

Результат работы **Workspace Wizard** может иметь вид:

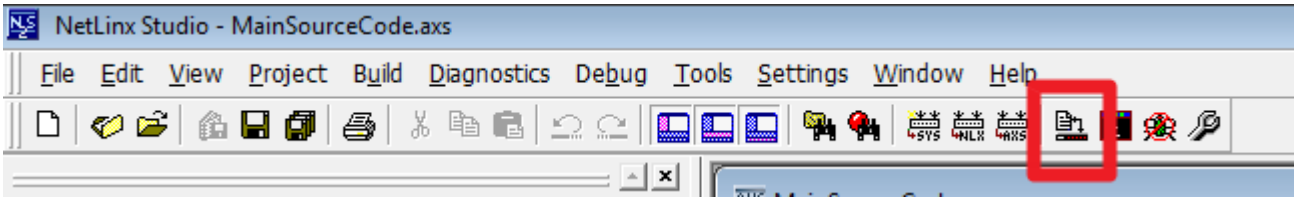


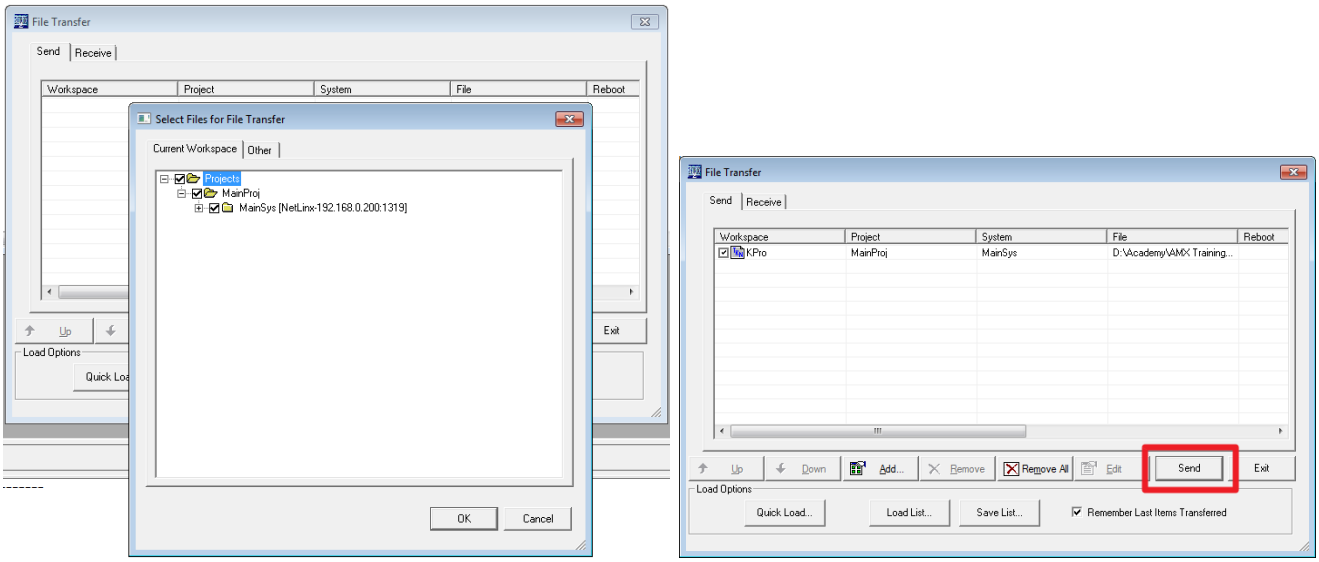

В файле MainSourceCode создается загружаемая программа контроллера.

Чтобы **компилировать** код, используйте клавишу **Build Active System**



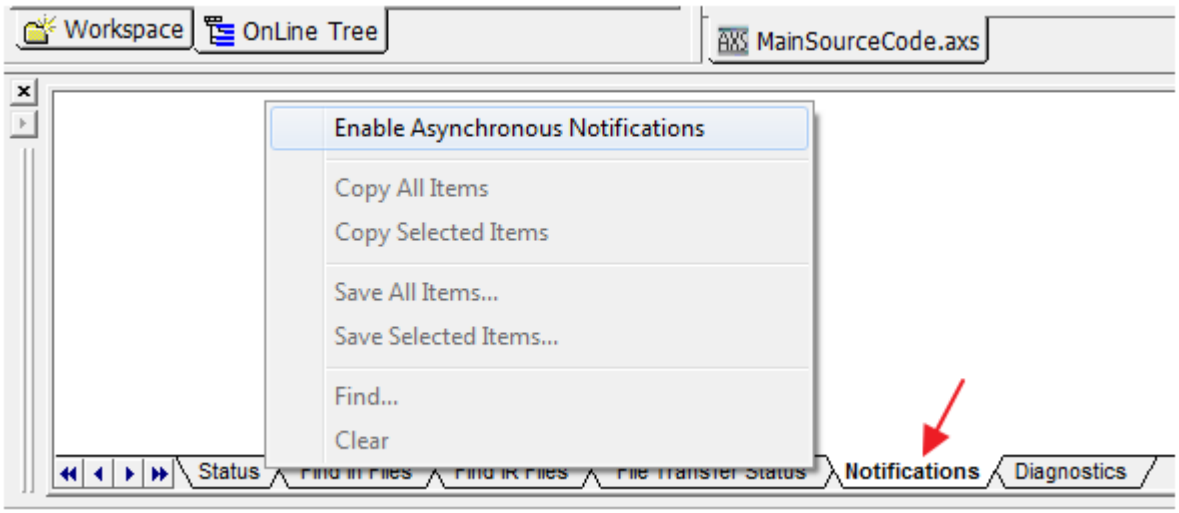

Чтобы **загрузить** исходный код на контроллер, используйте клавишу **File Transfers**



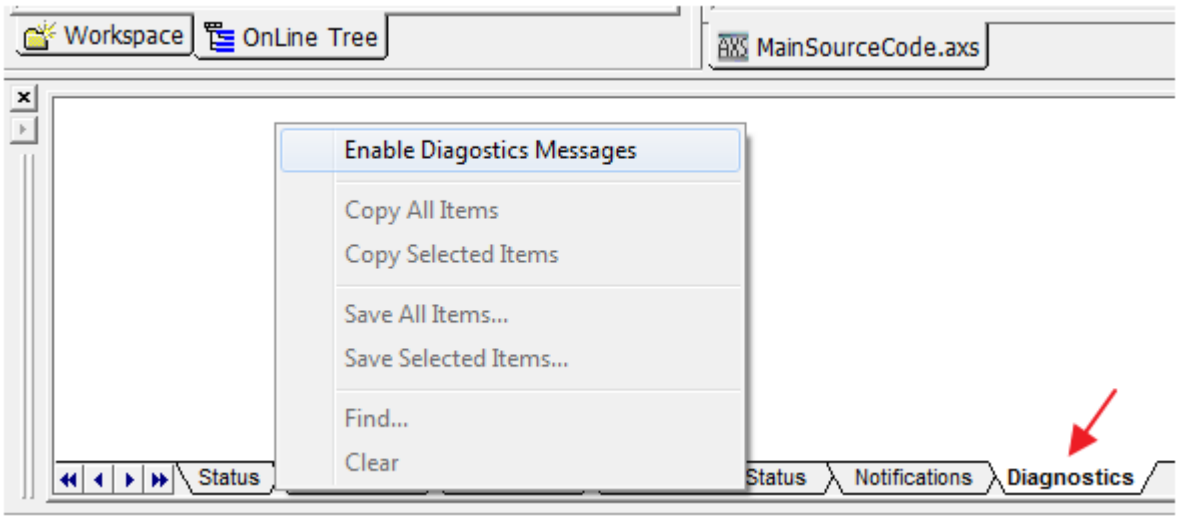



*Add…* (выберите проект для загрузки) > *Send* (загрузите проект на контроллер)

Включите **уведомления** после загрузки кода на контроллер:




Включите **диагностику** после загрузки кода на контроллер:



# Пример создания исходного кода в NetLinx Studio

В проекте может быть определен ряд разделов, внутри которых хранятся данные, команды, массивы и переменные.  Определение производится с помощью блока DEFINE_ после которого идет тип раздела.

Обязательно, при создании нового файла исходного кода, нужно определить ваш контроллер AMX и все панели управления, которые будут с ним работать. Например:

```
PROGRAM_NAME='MainSourceCode'

DEFINE_DEVICE
Dv_Master = 5001:1:0                    // мастер-контроллер [D:P:S] девайс:порт:система(любая)
Dv_COM1 = 5001:1:0                      // первый COM-порт
Dv_Relay = 5001:4:0                     // блок реле контроллера [D:P:S] Port: 4 = блок реле NI2100, 4 канала >> Pot:4, Code: 1-4
Dv_TP1 = 10001:1:0                      // панель управления (iRidium) как устройство-расширитель
Dv_TP2 = 10002:1:0                      // панель управления (iRidium) как устройство-расширитель
Dv_TP3 = 10003:1:0                      // панель управления (iRidium) как устройство-расширитель
```

Имя функционального блока будет использоваться для обращения к контроллеру, тому или иному выходу контроллера, к панели управления (AMX или iRidium – панели воспринимаются контроллером одинаково).

В DEFINE_VARIABLE можно создать массив, например, массив панелей управления, для обращения ко всем им одновременно:

```
DEFINE_VARIABLE
DEV     g_Panels[]      = { Dv_TP1, Dv_TP2, Dv_TP3 }
char    g_cStatus[16]

//integer g_iCurrent = 0

```

При запуске контроллера можно задать параметры работы COM-порта:

```
DEFINE_START
//SEND_COMMAND Dv_COM1,"'SET BAUD 9600,N,8,1,DISABLE'"

```

В DEFINE_EVENT будем создавать все события нажатия на кнопки, уровни, и другие элементы панелей управления, которые должны приводить к событиям:

```
DEFINE_EVENT
```

Обработчик нажатия на кнопку с Code:1. Кнопка может быть нажата на любой панели управления, указанной в массиве g_Panels

```
/**
    Обработчик нажатия на кнопку Channel, Code: 1
*/

BUTTON_EVENT [g_Panels,1]       // [Device,Channel]
{
    PUSH:                       // Event
    {
        on[Dv_Relay,1]                  // URL execute
        on[g_Panels,1]                  // покажет, что реле замкнулось, через Channel Code: 2
    }

    RELEASE:                            // отпускание элемента соответствует отправке цифры 0 на channel по событию Release в Иридиум
    {
        off[Dv_Relay,1]
        off[g_Panels,1]
    }

    /*HOLD[10]:
    {
        off[Dv_Relay,1]
        off[g_Panels,1]
    }*/
}
```

BUTTON_EVENT – нажатие на кнопку (General). Поддерживает события PUSH, RELEASE, HOLD

Обработчик работы с уровнем (Bargraph)

```
LEVEL_EVENT [g_Panels,1]
{
    send_level g_Panels,1, Level.VALUE  // пишем Value из Level Code: 1 в Level Code: 1

/*  if (Level.Value > 100)
        on[Dv_Relay,2]                  // замыкаем 2е реле NI2100 если в Level Code: 1 пришло Value > 100
    else
        off[Dv_Relay,2] */
}

/**
    Обработчик данных панели
*/
```

Обработчик событий online, offline, string, command

```
data_event[g_Panels]
```

**Все необходимые команды вы можете найти в справочниках AMX.**