

---

---

## TECHNICAL SPECIFICATION

# isystem.connect C++ Quick Start

## Purpose

This document describes usage of *isystem.connect* interface for first time users. It describes a simple application written in C++ with Visual Studio 2005. This is a command line application, which:

- Takes workspace names as command line argument
- Opens the workspace in winIDEA
- Downloads application to emulator
- Starts the application
- Prints a global variable to standard output

The code below is intended for demonstration of *isystem.connect* interface, so it contains only the most basic error handling. For production code see error handling in documentation.

## Prerequisites

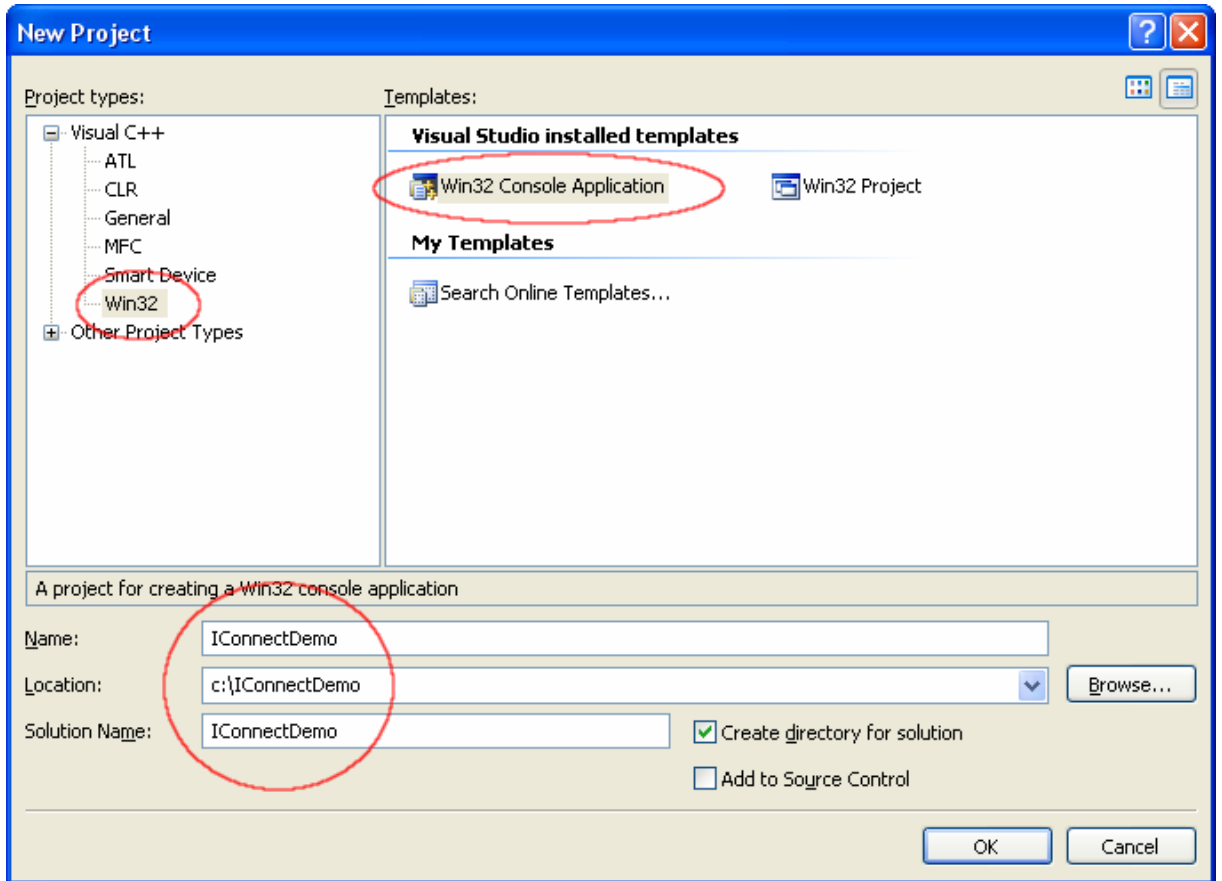
Before start we need the following components installed on our computer:

- Visual Studio 2005
- winIDEA
- *isystem.connect* documentation file **isystem.connect.pdf** or **IConnect.chm**

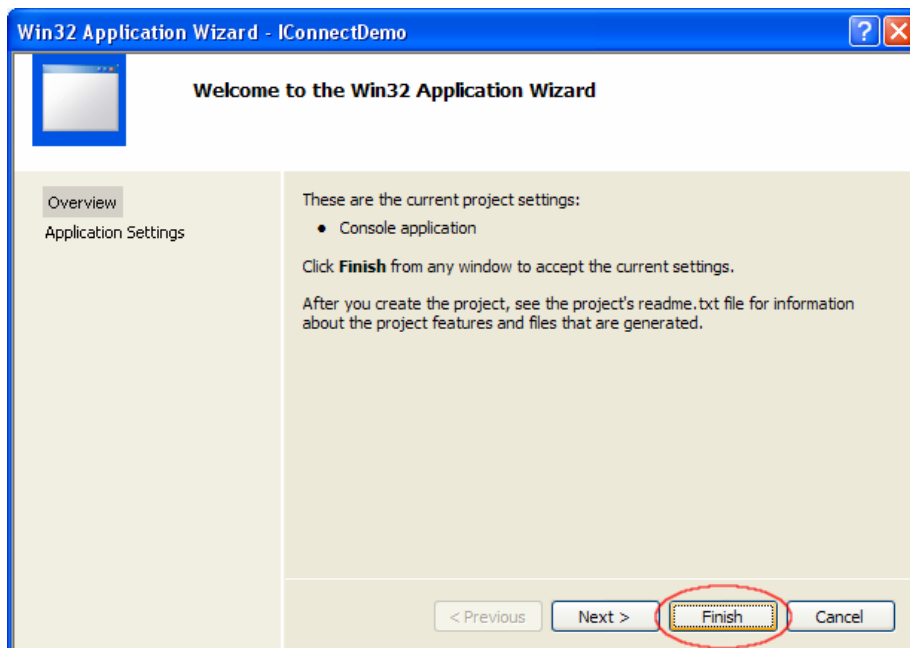
## Creating the application

Let's start Visual Studio 2005 and create a new project:

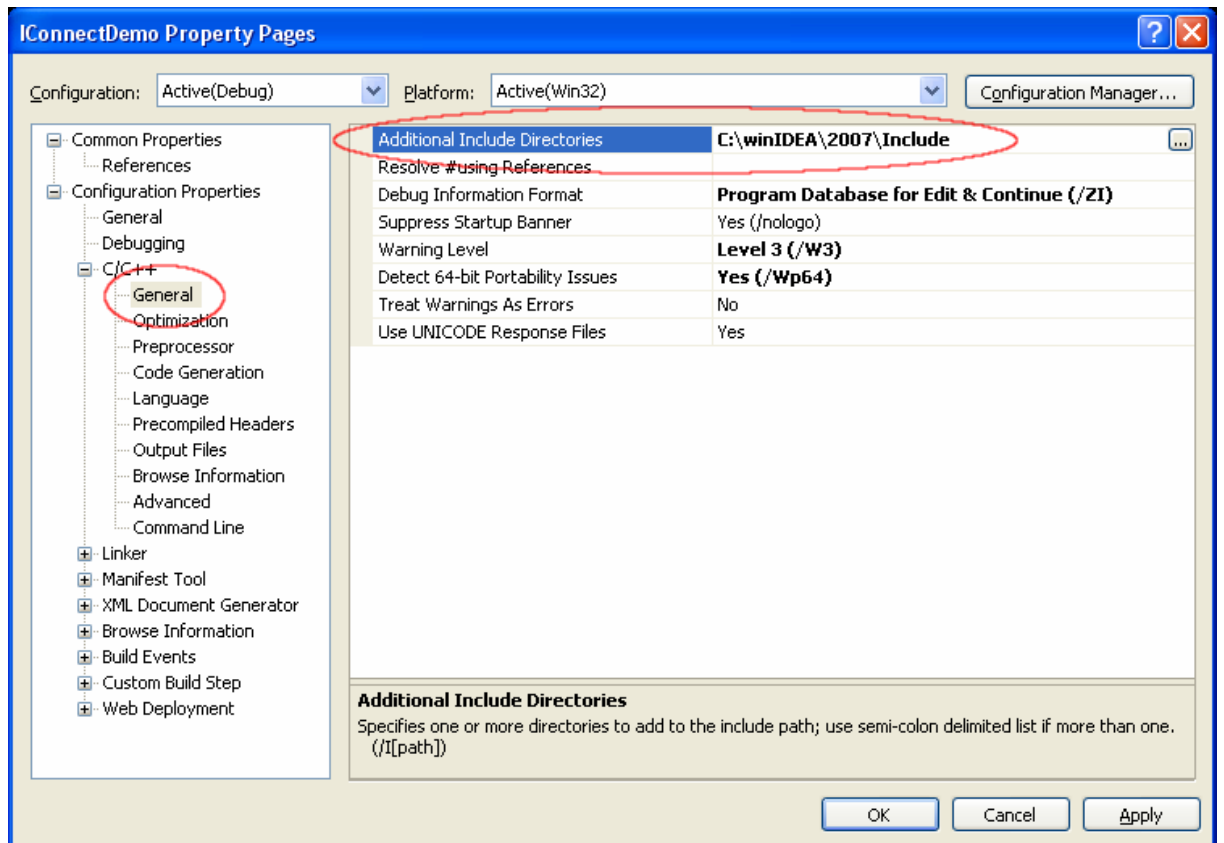
1. Select option *File / New / Project*
2. A dialog for new project opens. First we select *Win32* as *Project type*, and *Win32 Console Application* as *Visual Studio Installed Template*. Then we enter project name, location, and solution name. Then we click OK.



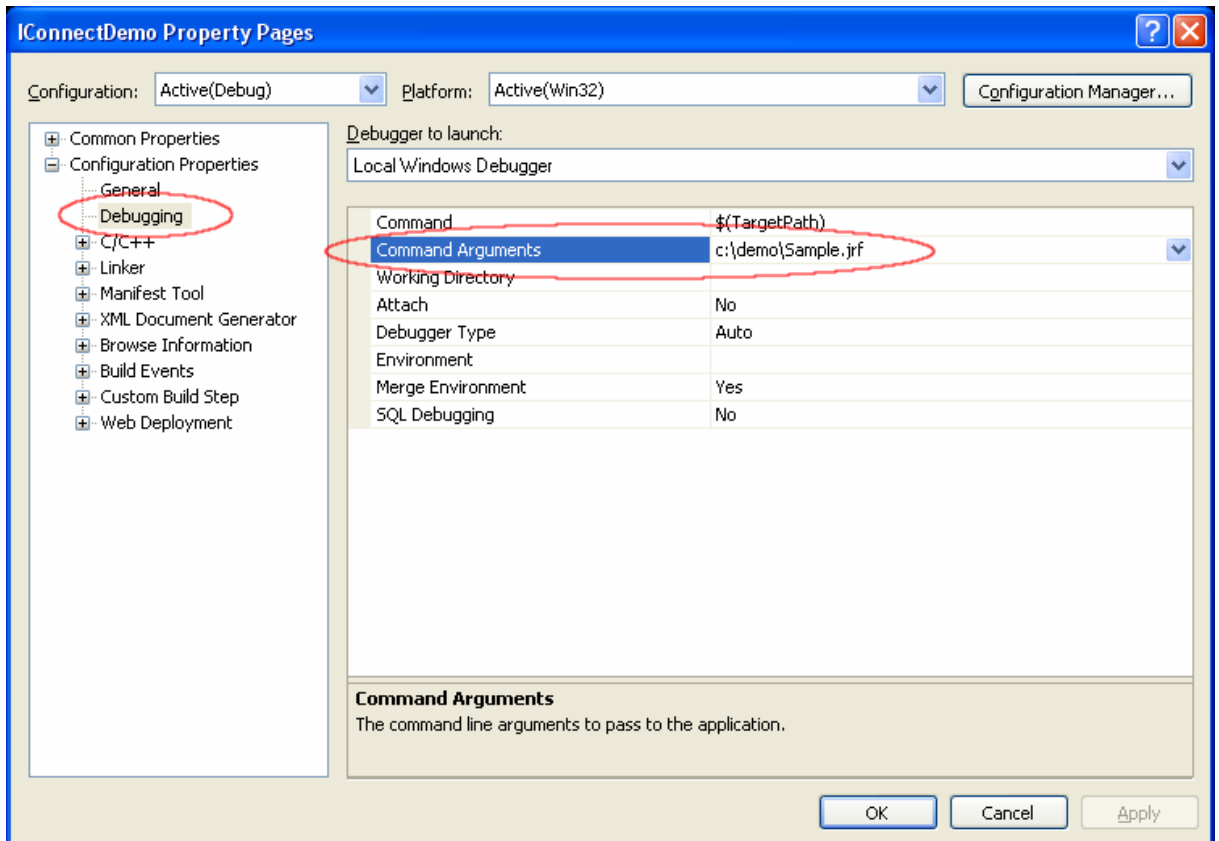
3. On the next dialog we simply click *Finish*.



4. The project is created. Now we have to tell compiler where to look for *isystem.connect* header files. We use option *Project / Properties* to open a dialog. Then we expand *Configuration Properties / C/C++ / General* and enter directory to *Additional Include Directories* field, as shown in the image below. If our winIDEA is not installed in C:\winIDEA, we have to adapt the path accordingly.



5. To make usage of *isystem.connect* interface easier, we should also add file *IConnectClient.cpp* to our project. The file can be found in the same directory as header files.
6. Later we will also need name of workspace file as command line argument for debugging. Again we use option *Project / Properties* to open a dialog as shown below. Into field *Command Arguments* we should enter workspace file name with **absolute** path.



7. Our project is configured. Now it is time to start coding.

## The code

The first change should be done in *stdafx.h* header file. We should add the following three lines:

```
typedef unsigned __int64 QWORD;
typedef          __int64 QLONG;

#include <windows.h>
```

Next we add include directives to CPP file:

```
#include "IConnectClient.h"
#include <stdio.h>
```

As mentioned above, our application should open a workspace specified in command line. This means we have to check for input parameters and then connect to winIDEA. All the code shown below can be put inside *main()* function.

```
if (argc != 2) {
    printf("Usage:    IConnectDemo <workspaceFileName>\n"
           "Example: IConnectDemo C:\\demo\\Sample.jrf");
    return -1;
}
```

```

CIClient icc;

if (!icc.MRUConnect(argv[1]))
{
    printf("Connect failed!\n");
    return -1;
}

```

That's it. Let's transfer the application to emulator now, and then run it:

```

HRESULT retVal =
    icc.m_pIConnectDebug->RunControl(IConnectDebug::rDownload, 0, 0);

if (retVal != S_OK)
{
    printf("Download failed!\n");
    return -1;
}

retVal = icc.m_pIConnectDebug->RunControl(IConnectDebug::rRun, 0, 0);

if (retVal != S_OK)
{
    printf("Run failed!\n");
    return -1;
}

```

And finally, we'll print global variable *iCounter* five times, once per second:

```

for (int i = 0; i < 5; i++)
{
    char pszResult[500];
    SValue valueData = {0};
    SType valueType = {0};
    retVal = icc.m_pIConnectDebug->Evaluate(IConnectDebug::fMonitor,
                                           "iCounter",
                                           pszResult,
                                           sizeof(pszResult),
                                           &valueData,
                                           &valueType);

    Sleep(1000);
    if (retVal != S_OK)
    {
        printf("Evaluate failed!\n");
        return -1;
    }
    printf("%s\n", pszResult);
}

```

Do not forget to clean up:

```

icc.Disconnect(0);

```

# Conclusion

This document gives only a brief overview of the interface. For further steps it is recommended to read IConnect.chm document. It describes usage of all interfaces, examples of usage, and detailed descriptions of methods and parameters.