

Getting Started

1 Why Clean?

Most of us are not professional programmers. In my case, I would rather go snowboarding than spend hours in front of a computer. Although people have more interesting things to do instead of studying Computer Science and Mathematics, High School counsellors force them to take Algebra courses. Therefore, everybody learns some Algebra, willing or unwilling.

Clean is a computer language that has a very neat feature, that is, all you need to program in Clean is some knowledge of Algebra, and as I told before, everybody already knows some Algebra. On the other hand, if you want to learn C++, Pascal, or LISP, you need to study a lot of new and difficult Computer Science concepts.

Clean is not the only computer language that draws its coding schemes from Mathematics. There are others: Haskell, Prolog, Mercury, etc. However, Clean is easy to install, easy to manage, and efficient, about as efficient as C. Other Mathematical languages lack at least one of these features.

Clean has one and only one drawback: Clean team members are very good in Computer Science, and used to tackling difficult problems. Therefore, the tools that they develop are very general, and tailored for experts. For instance, I found myself daunted by Clean GUI building tools. Therefore, I decided to wrap them into simpler packages before shipping.

2 Installation

It is quite easy to install Clean for Windows. Download the distribution package from

<http://www.cs.ru.nl/~clean>

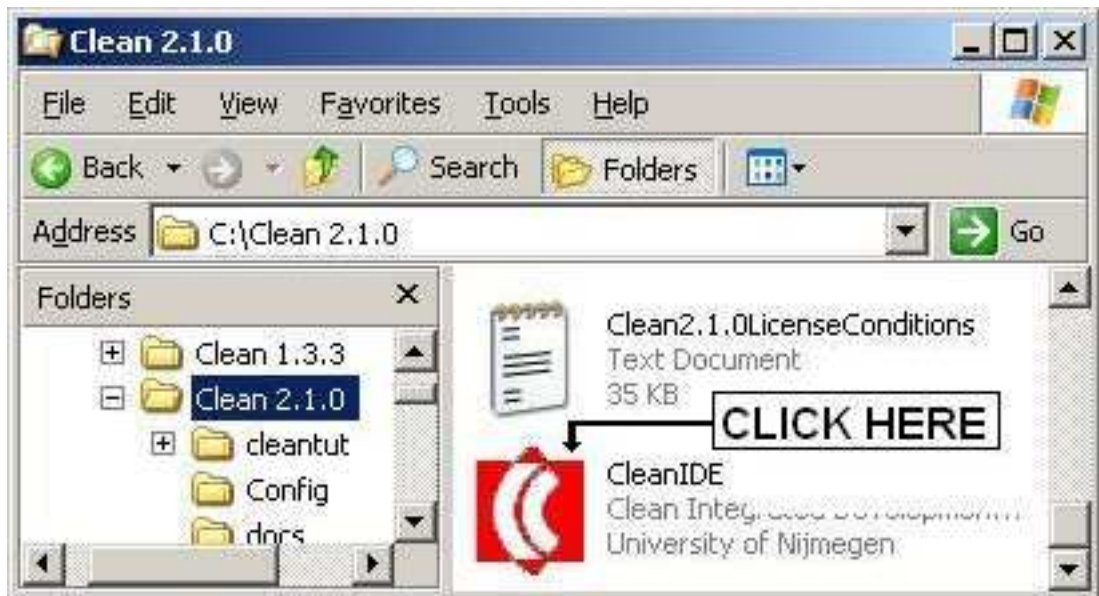


Figure 1: Starting Clean from the Explorer Window

Use an unzip tool to unpack the installation file. I suggest that you unpack the distribution file on the root of drive C, getting the following folder:

```
C:\Clean 2.1.0>
```

That is all. You can start using Clean from the DOS box, or from the Explorer. Figure 1 shows how to do it from the Explorer. You may want to install a shortcut to Clean on your desktop. To do this, right click the mouse on the desktop, and ask for a new shortcut, as shown in figure 2. In the shortcut dialog that pops up, browse for the `CleanIDE.exe` (see figure 3). Finally, press `Next` and `Finish`.

3 Configuration and Testing

To configure the IDE, click on the Clean desktop icon. The IDE will pop up. Choose the environment submenu, and check `Everything` (figure 4).

Let us make a small test. Choose the option `File/New File` from the task menu. In the New File dialog, choose a name for the file that will



Figure 2: Creating a shortcut to clean

hold the main program. Attention: The program must have extension *icl* (figure 6). Use the provided text editor to type the following program:

```
module firststeps
import StdEnv

Start= cos(3.1416/3.0)
```

In the IDE text editor, the program should look as in figure 7. *NB*: The module must have the same name that you have chosen for the file, except

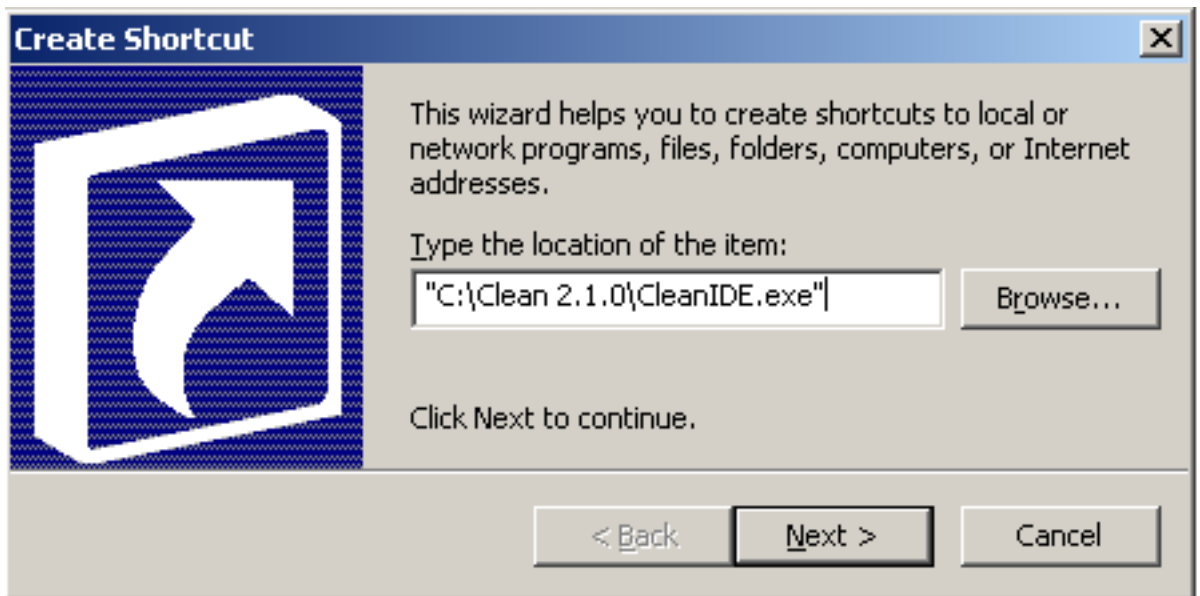


Figure 3: Creating a shortcut to clean

for the extension. In this case, the file name is `firststeps.icl`. Therefore, the module name is `firststeps`.

In the Environment option of the main menu, check Everything. This environment will provide both a console to print the results of our programs, and a graphic user interface. Although you and I prefer a graphic user interface (GUI), let us utilize a console to test ideas and snippets.

The option File/New Project will allow you to create a project. If you have already tried your hand at C, Pascal, or OCAML, you know how difficult is to create a project in these languages. In Clean, all you need to do is to accept the option given in the New Project Dialog. The final step before compiling and running is to choose the project options. From the task menu, go to Project/Project Options, and check Show Constructors, as in figure 8. Finally, go to the option Project/Update and Run, and your program will be compiled, showing a result that is very close to $\sin(\frac{\pi}{3})$.

Let us summarize all the steps necessary to compile a program.

- From option File/New File, create a new file with extension `icl`.
- In the Environment option of the main menu, check Everything.

- Create a project file using option File/New Project.
- Check Show Constructors in Project/Project Options.
- Project/Update and Run.

4 A soft introduction to GUI

I guess that ObjectIO library is something really spectacular. However, tyros often find it daunting; therefore, I wrote a wrap around it that will make your first steps somewhat easier. Download the files `myIO.icl` and `myIO.dcl` and install them. You can download zipped versions of these two libraries from the following address:

<http://www.discenda.org/Clean/myio.zip>

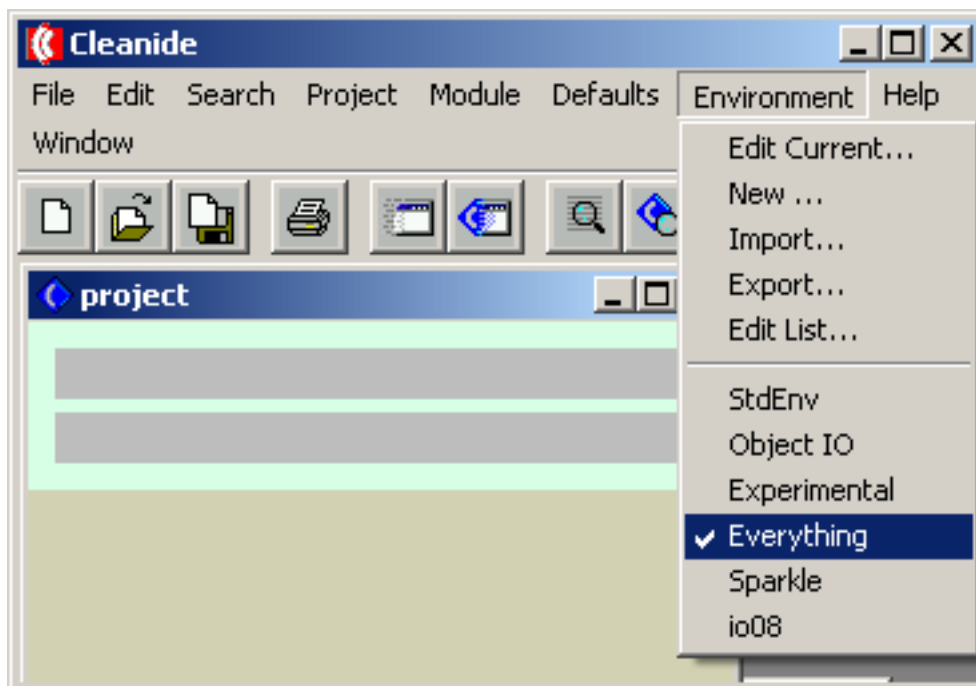


Figure 4: Checking Everything

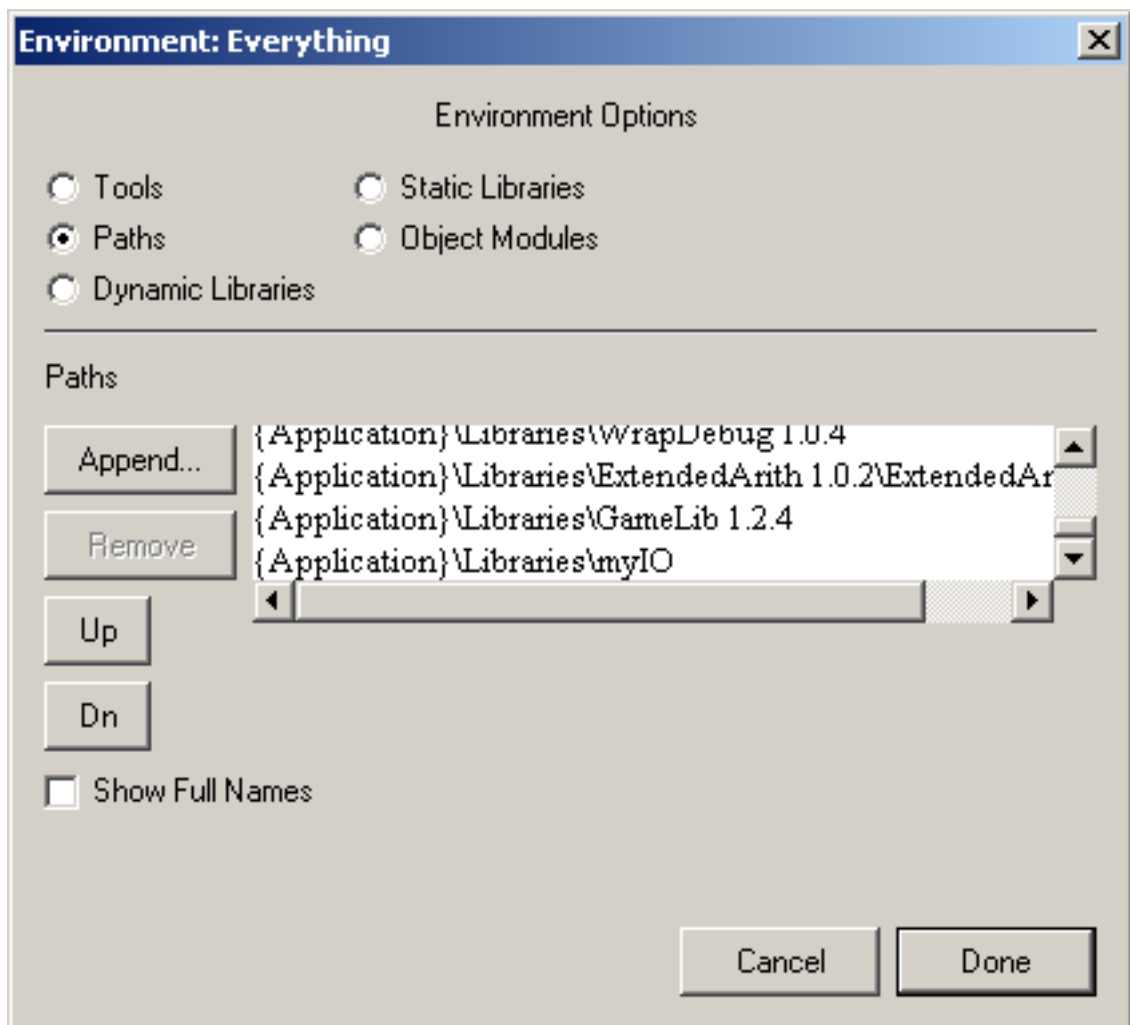


Figure 5: Adding a new path

Unzip `myio.zip` inside `C:\Clean 2.1.0\Libraries\myio\`. Now, go to the Environment menu option, check `Everything`, and choose the `Edit Current` sub-option. Then, add a path to `myIO`, as shown in figure 5. To accomplish this step, you need to check the radio button `Paths` (figure 5), and press the button `Append`. This will put you in a browser dialog, that you can use to locate the `myIO` library, and add it to the path list. Do not forget to unzip the `myIO.zip`, so as to obtain the files `myIO.dcl` and `myIO.icl`.

4.1 Testing the GUI interface

Now, let us test the GUI interface. Check the **Environment/Everything** option from **Clean** main menu (also called task menu). Then, choose the **File/NewFile** option, and create a file `testIO.icl`. You will use the provided editor to type a Clean module into this file. The module must have the same name as the file, except for the extension:

```
module testIO
import StdEnv , myIO

Start world= ioDialog (RLISTtoR avg) world
```

Then, create a new project using the **File/New File** option. From the task menu, go to **Project/Project Options**, and check **No Console**. In section 3, you have chosen the **Show Constructors** option. This time, you must check **No Console**. The last step is **Update and Run**. Figure 9 shows the result of running the program. By the way, this program is supposed to calculate the average of a list of **Real** numbers.

You will find a summary of the steps necessary to create a GUI program below. Do not forget to download and unzip the `myio` library inside



Figure 6: File/New File

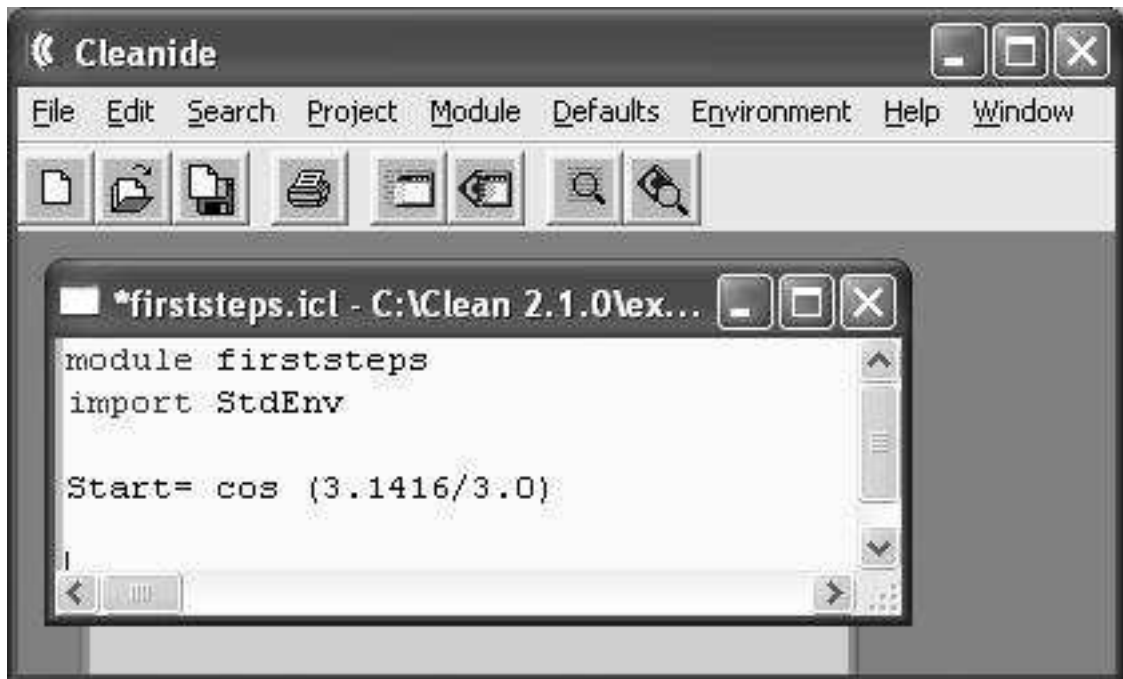


Figure 7: Starting a New Project

C:\Clean 2.1.0\Libraries\myio\, and to add a path to it through the Environment/Edit Current menu option.

- From option File/New File, create a new file with extension icl.
- In the Environment option of the main menu, check ObjectIO.
- Create a project file using option File/New Project.
- Check No Console in Project/Project Options.
- Project/Update and Run.

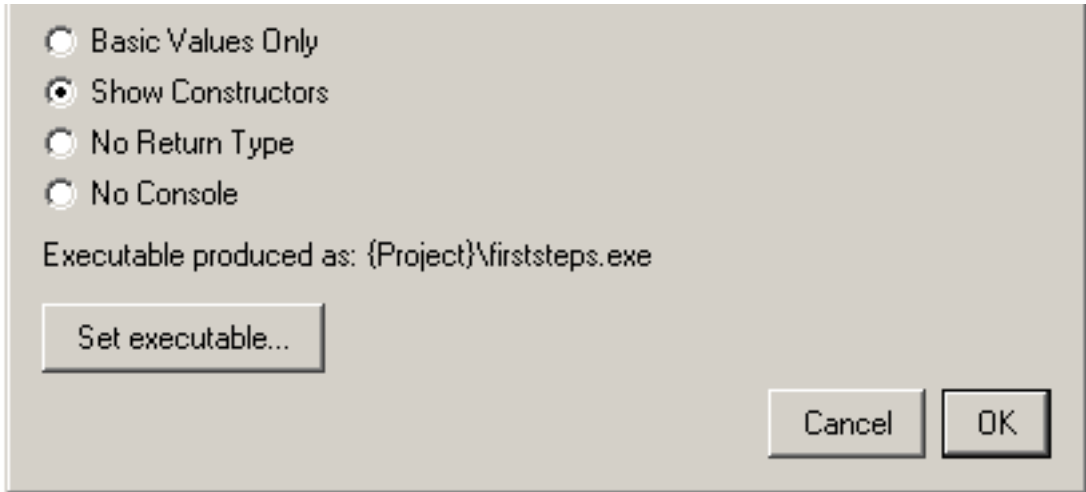


Figure 8: Asking for a console

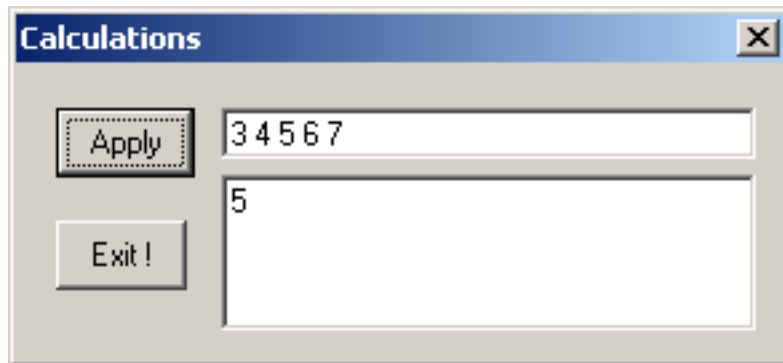


Figure 9: guitest