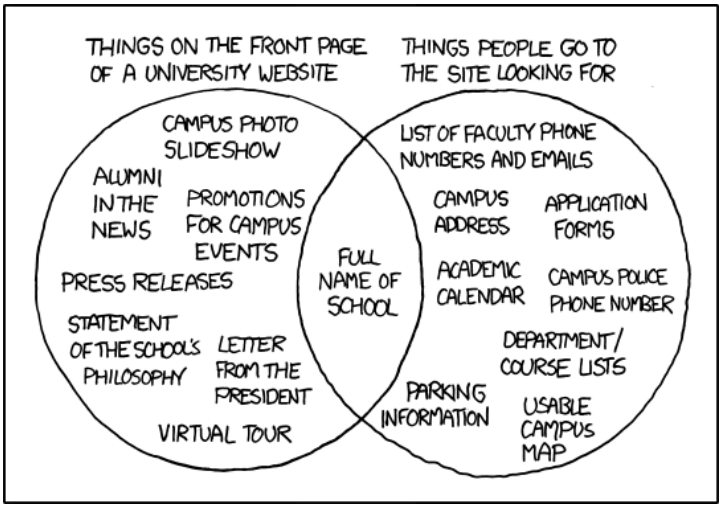
Assignment 1: Getting started

ETH Zurich

Hand-out: Monday, 20 September 2010



University Website © Randall Munroe (xkcd.com)

1 Welcome to ETH!

Goal

The goal of this task is to familiarize you with the infrastructure provided by ETH Zurich and to make sure that you have subscribed to this course.

To do

- Log in to a computer in one of the computer rooms. The location of the computer rooms is given in the information sheet that you received.
- Log in to the n.ethz account administration page (http://www.passwort.ethz.ch). After logging in you will see the welcome screen. Click on "Passwort ändern" (top left of screen), change your password[1] and log out. After this you can log on to the ETH web-mail interface (https://mail.ethz.ch/exchange) with your user name and your newly set password.

[1] Your password should be simple enough for you to remember; yet it should be complex enough so that other people cannot guess it. See password recommendations at CERN: http://security.web.cern.ch/security/passwords.

- Try to find the Eiffel development environment - EiffelStudio - on the machine that you are using. If you want to use your personal computer and install EiffelStudio there, you can follow the instructions in the next task.
- Bookmark the Forum der Informatik Studierenden: http://forum.vis.ethz.ch. The forum provides a platform for questions and discussions with your peers.
- **IMPORTANT:** Make sure that you have subscribed to this course (and all the other courses you attend) on http://www.mystudies.ethz.ch. Otherwise it is difficult to give you your testat at the end of the semester.

Hint

You will find that your n.ethz login and password are very useful on many other ETH web pages, e.g. the http://www.mystudies.ethz.ch page mentioned above.

To hand in

There is nothing to hand in; the task is accomplished as soon as you have subscribed to the course.

2 EiffelStudio installation

Goal

In this task you will install the software development platform EiffelStudio. If you intend to work in a computer lab of ETH you will not need to install EiffelStudio. Proceed with task 3. We support the following operating systems:

- Windows (Microsoft C compiler only!)
- Linux
- Mac OS X

A general suggestion: if you encounter precompilation problems during installation, and you think you have fixed them, before precompiling again you may want to delete the EIFGEN directory, which may contain corrupted files at this point.

To do

For Windows

1. As a first step you need to get the Microsoft C compiler. If you already have it installed (e.g as part of Microsoft VisualStudio), skip this point. To install Microsoft C compiler, follow the instructions at http://dev.eiffel.com/index.php/Installing_Microsoft_C_compiler.
2. Download EiffelStudio 6.6 from sourceforge[2].
3. Start the installer and follow the instructions. Choose the option *EiffelBase, WEL and EiffelVision2* in the step "Precompiled Libraries". The precompilation of the libraries takes a long time during installation, but will significantly speed up later compilations.

[2]Click on the hyperlink, or go to http://sourceforge.net/projects/eiffelstudio/files/ and scroll down until you find the Eiffel Studio 6.6 drop-down arrow. Expand it, and select the top build in the list (83873). Expand it: the file to download is "Eiffel66_gpl_83873-windows.msi".

For Linux

Please note that we assume a basic set of preinstalled tools and libraries, including tar, gcc, pkg-config, headerfiles, and others. We recommend to install EiffelStudio with the default locale (en_US).

1. Install the *development* versions of libgtk and libxtst libraries, if you don't have them yet (`libgtk2.0-dev` and `libxtst-dev` on Ubuntu).
2. Pick a directory of your choice without any spaces in the path, for example /opt:
   `cd /opt`
3. Download EiffelStudio 6.6 from sourceforge[3]. Let's suppose you downloaded it to your desktop.
4. Extract it: `sudo tar -xvjf ${HOME}/Desktop/Eiffel66_gpl_83873-linux-x86.tar.bz2`
5. Set the following environment variables (note that how and where to change environment variables depends on which distribution you use). Make sure you set these permanently and systemwide! Note: If you download the 64 bit version of EiffelStudio make sure to change the environment variable ISE_PLATFORM accordingly (as explained in the installation instructions of EiffelStudio).
   Type **as root**:
   ```
   echo ISE_EIFFEL=/opt/Eiffel66 >> /etc/environment
   echo ISE_PLATFORM=linux-x86 >> /etc/environment
   echo PATH=$PATH:$ISE_EIFFEL/studio/spec/$ISE_PLATFORM/bin >> /etc/environment
   echo source /etc/environment >> /etc/profile
   ```
   Restart the X server. You should now have the eiffel compiler (`ec`) in your path, as well as all environment variables set (check!).
6. To speed things up later, we need to precompile some libraries. Note that this may take some time.
   ```
   cd /opt/Eiffel66
   ./make_install
   ```
   Answer 'y' to both questions: this will build the precompiled libraries.

For Mac OSX

Please follow the instructions provided at the following page:
http://dev.eiffel.com/EiffelOnMac under section "Using Macports". The installation has been tested on Mac Os X 10.5 (Leopard).

To hand in

There is nothing to hand in.

3 Your first Traffic program

In this task you will download Traffic and experiment with some feature calls. You need to have EiffelStudio installed (see task 2). It can be useful to have chapter 2, section 2.2 of Touch of Class open in front of you and check it step-by-step.

[3]Follow the instructions in footnote 2: the file you are looking for is "Eiffel66_gpl_83873-linux-x86.tar.bz2".

Todo

1. Download Traffic from http://traffic.origo.ethz.ch/download and unzip it to a folder of your choice (it's recommended to use a path without any spaces). Make sure that you download the latest release (file traffic_ev_1134.zip).

2. Figure 1 shows the directory structure of Traffic. The top-level directory `library` contains the core of Traffic: Eiffel class files that model the transportation system in a city and support its visualization. The other top-level directory `example` contains some existing Traffic projects. For example, `city_vision2` is an application that allows testing most of the functionality of Traffic. There are also the applications implementing the examples in Touch of Class, like `02_objects`, implementing the Preview example found in Chapter 2, section 2.2.

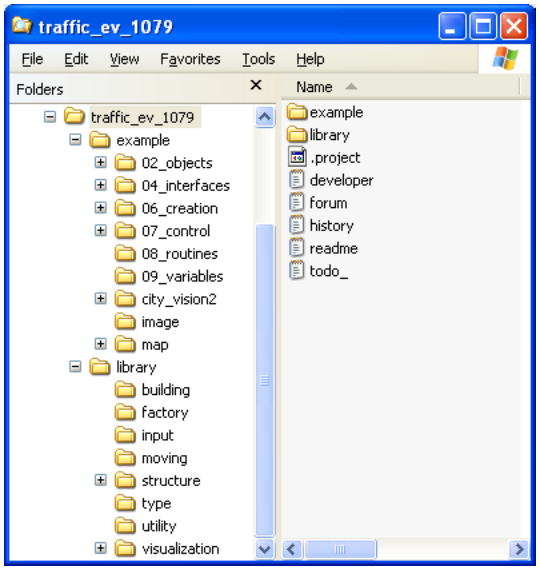


Figure 1: Directory structure of Traffic.

3. Start EiffelStudio (for Linux users: type `estudio` in a console).

4. EiffelStudio will show the dialog of figure 2. If this does not happen automatically, go to `File` ▷ `Open project` and it should appear.

5. Click on `Add project` and choose the file found under the directory where you unpacked Traffic, at example/02_objects/objects.ecf. Click `Open`. This will compile the test application.

6. Launch the program by clicking on the Start green button (see figure 3). In the application window, click on `Run example`. After some loading time you will see a map. Note that the loading will take quite long this first time, but will be much faster the next time you do it. You will also see that Louvre and a metro line are highlighted, and a certain route is animated (a small pedestrian moves along it). Please note that depending on the speed of your machine, the aforementioned actions will be performed immediately or after a short while. In fact, to make it easier to see the animations, we have inserted some *wait* instructions inside the invoked code. Close the application by clicking on the stop button (the red square) in EiffelStudio or just close the application window.

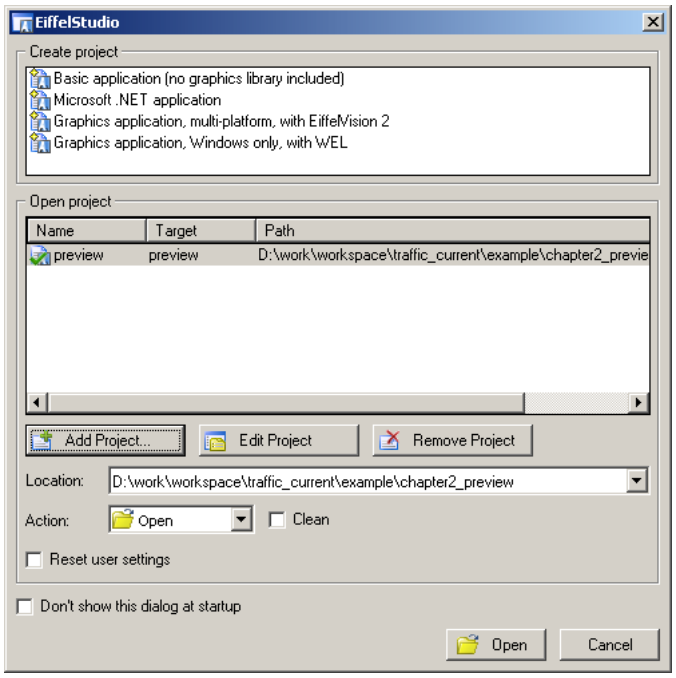


Figure 2: Open an Eiffel project

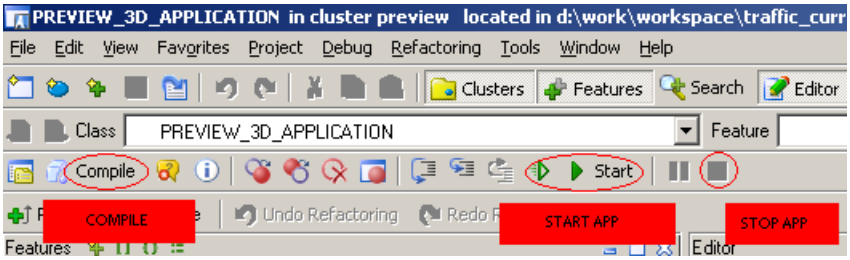


Figure 3: Compiling a project, starting and stopping an application

7. Open the class *PREVIEW*. In the feature *explore*, between the **do** and the **end** and after *Paris.display*, you will find the following text:

```
1  Louvre. spotlight
   Line8. highlight
3  Route1.animate
   Console.show (Route1.origin)
```

8. Using two dashes (`--`) at the start of each line, turn the first three lines starting from `Louvre.spotlight` into comments. By doing so, you are telling the machine not to take them into consideration. Save, recompile the project (by clicking on the compile button in EiffelStudio) and launch it again. Click again on the *Run example* button and you will see that Louvre and Line8 are not highlighted anymore, and that Route1 is not animated anymore. This makes sense, because as we just said, the commented instructions are not part of the code that will be executed anymore.

9. Now, without uncommenting the previously commented lines, try writing the instructions all by yourself. You may want to try writing one line at the time, then compile and execute to see the effect. Notice that when you write an object name (the first part of the instruction, before the dot) and then the dot, the "completion" feature of EiffelStudio jumps in and lets you choose between the available features that you can invoke on that object.

10. If you haven't done it yet, please read through sections 2.1 and 2.2 of chapter 2 of Touch of Class, and don't hesitate to ask your assistant if you have any doubts or issues!

To hand in

There is nothing to hand in.