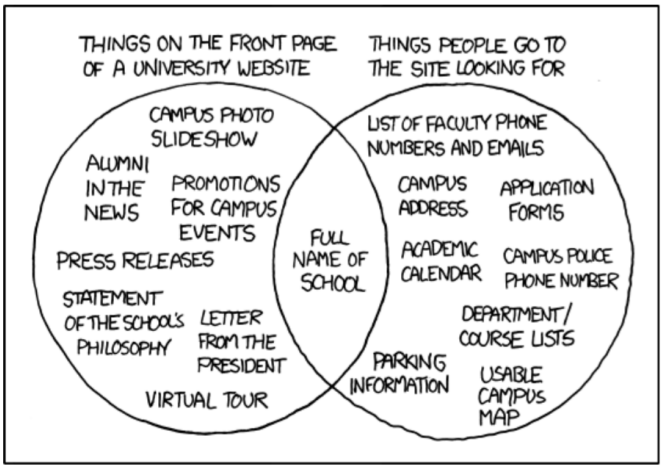
Assignment 0: Getting Started



Published: August 16, 2016
Due: –○–

Goals

- to install the software development platform EiffelStudio.
- to download Traffic library and experiment with it.

1 EIFFELSTUDIO INSTALLATION

In this task you will install the software development platform EiffelStudio. We support the following operating systems:

- Windows
- Linux
- MacOSX

If you encounter any problems during EiffelStudio installation, please ask your TA.
In case you encounter a crash of the EiffelStudio development environment after installation (during normal use), please click the "Submit bug report" button.

To do

For Windows (tested on Window 8):

1. Download EiffelStudio16.05 from sourceforge: go to `https://sourceforge.net/projects/eiffelstudio/files/`, click on the folder icon labeled "Eiffel Studio 16.05" and then on the top build in the list (98969). Finally choose "Eiffel_16.05_gpl_98969-windows.msi" (32 bit) or "Eiffel_16.05_gpl_98969-win64.msi " (64 bit).
2. Start the installer and follow the instructions. When you open your first Eiffel project (see task 2), EiffelStudio will ask if the required libraries should be precompiled. Answer yes; precompilation will take some time, but will speed up things later.

For Linux:
If you use a Debian based distribution (such as Ubuntu), installing EiffelStudio is very simple. You only need to open a terminal and run the following commands:

```
sudo add-apt-repository ppa:eiffelstudio-teamppa
sudo apt-get update
sudo apt-get install eiffelstudio
```

The first line adds the repository for EiffelStudio to your package sources, the second command updates the list of packages in your package manager and the last command finally installs EiffelStudio. To start EiffelStudio, simply type `estudio` in the console.
If you use a distribution that is not based on Debian, or the previous approach did not work for you, you can use the following instructions. Please note that we assume a basic set of preinstalled tools and libraries, including `tar`, `gcc`, `pkg-config`, and others. We recommend installing 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. Download EiffelStudio16.05 from sourceforge: go to `https://sourceforge.net/projects/eiffelstudio/files/`, click on the folder icon labeled "Eiffel Studio 16.05" and then on the top build in the list (98969). Finally choose "Eiffel_16.05_gpl_98969-linux-x86-64.tar.bz2". Let's suppose you downloaded it to your Downloads folder.

3. Extract it:

   ```
   sudo tar -xvjf $HOME/Downloads/Eiffel_16.05_gpl_98969-linux-x86-64.tar.bz2
   -C opt
   ```

4. 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 system-wide! 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). Add the following lines to the end of `etcprofile` with: `sudo gedit etcprofile`

   ```
   export ISE_EIFFEL=opt/Eiffel_16.05
   export ISE_PLATFORM=linux-x86-64
   export PATH=$PATH:$ISE_EIFFEL/studio/spec/$ISE_PLATFORM/bin
   ```

5. Restart your computer.

6. To start EiffelStudio type `estudio` in the console. When you open your first Eiffel project (see task 2), EiffelStudio will ask if the required libraries should be precompiled. Answer yes; precompilation will take some time, but it will speed up things later.

For Mac OS X (tested on Yosemite and El Capitan):

1. Download EiffelStudio16.05 from sourceforge: go to `https://sourceforge.net/projects/eiffelstudio/files/`, click on the folder icon labeled "Eiffel Studio 16.05" and then on the top build in the list (98969). Finally choose "Eiffel_16.05_gpl_98969-macosx-x86-64.tar.bz2". Let's suppose you downloaded it to your Downloads folder.

2. Extract it:

   ```
   sudo tar -xvjf $HOME/Downloads/Eiffel_16.05_gpl_98969-macosx-x86-64.tar.bz2
   ```

3. Set the following environment variables. Make sure you set these permanently. Note: If you download the 64 bit version of EiffelStudio make sure to change the environment variable `ISE_PLATFORM` accordingly. Add the following lines to the end of `.bash_profile` with: `sudo nano .bash_profile`

   ```
   export ISE_EIFFEL=$HOME/Downloads/Eiffel_16.05
   export ISE_PLATFORM=macosx-x86-64
   export PATH=$PATH:$ISE_EIFFEL/studio/spec/$ISE_PLATFORM/bin
   ```

4. Restart your computer.

5. To start EiffelStudio type `estudio` in the console. When you open your first Eiffel project (see task 2), EiffelStudio will ask if the required libraries should be precompiled. Answer yes; precompilation will take some time, but it will speed up things later.

To hand in

There is nothing to hand in.

2 YOUR FIRST TRAFFIC PROGRAM

In this task you will download Traffic library and experiment with some feature calls. You need to have EiffelStudio installed (see task 1).

To do

1. First you have to download the `EiffelBase2` library, which you will need in this and other assignments. Download `base2.zip` from `https://drive.google.com/open?id=0B1GMHm59JFjqd2tGcEZWX1dfV0E` and unzip it into `$ISE_EIFFEL/library/`, where `$ISE_EIFFEL` is the directory where you installed EiffelStudio (for example "C:\Program Files\Eiffel Software\EiffelStudio 16.05 GPL\" on Windows). Note: it is important that you use this directory; to make sure everything is correct, check that you have the file `$ISE_EIFFEL/library/base2/base2.ecf` on your machine.

2. Download the Traffic library from `https://drive.google.com/open?id=0B1GMHm59JFjqRW5RRDM1aGJ3SjQ` and unzip it to a folder of your choice.

3. Figure 1 shows the directory structure of the archive. 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 applications that use the Traffic library. For instance, map browser allows you to browse the map of Zurich and display information about the city objects. Directory `assignment_1` contains the application that you will explore in this assignment. During the semester we will add more application examples for the assignments.

4. Start EiffelStudio: you will see the dialog of figure 2.

5. Click on *Add Project ...* and choose the file found under the directory where you unpacked Traffic, at `examples/assignment_1/assignment_1.ecf`. Click *OK* then *Open*. This will compile the application. If EiffelStudio asks to precompile libraries, answer yes.

6. Launch the program by clicking on the green Run button (see figure 3); you will see a map of Zurich center. You can move the map by dragging and zoom it with the mouse

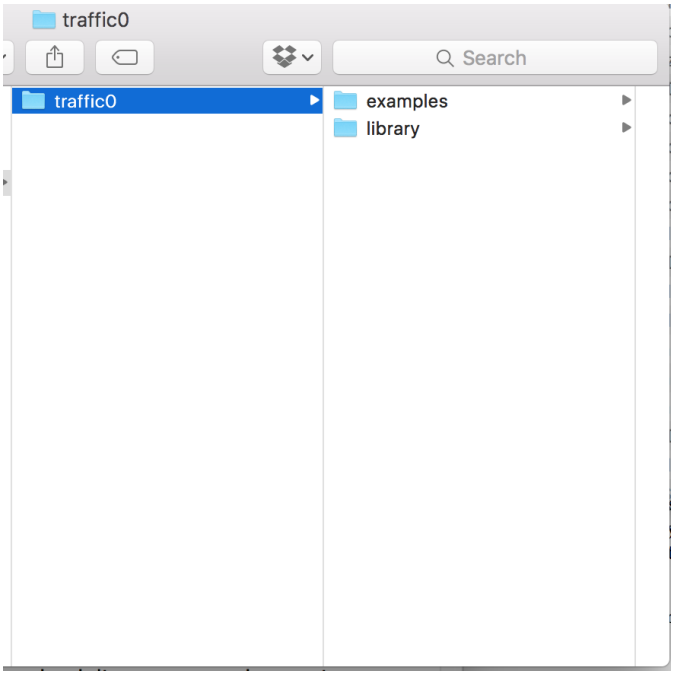


Figure 1: Directory structure of Traffic.

wheel: just like in Google Maps. Notice that some items on the map are highlighted (namely, the ones that depict Polybahn and its two stops) and the console at the bottom of the window displays information about Polybahn. After a couple of seconds, a cable car will appear and start moving along Polybahn. Close the application by clicking on the stop button (the red square) in EiffelStudio or just close the application window.

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

```
Central view.highlight
Polyterrasse_view.highlight
Polybahn_ view.highlight
console.output (Polybahn)
 wait (3)

 Zurich.add_public_transport (Polybahn_line_number)

Zurich_map.update
Zurich_map.animate
```

8. Using two dashes (--) at the start of each line, turn the first three lines starting from *central_map.highlight* into comments. By doing so, you are telling the machine not to

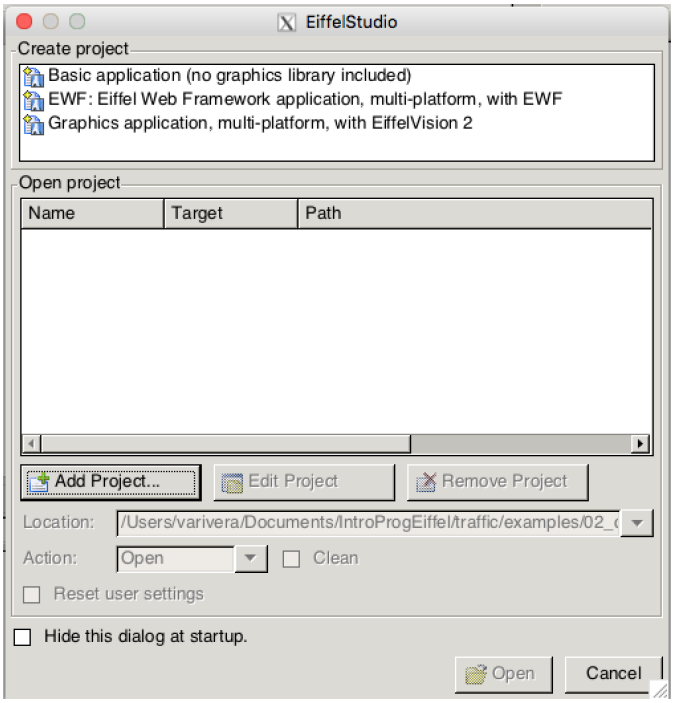


Figure 2: Open an Eiffel project.

take them into consideration. Save, recompile the project (by clicking on the compile button in EiffelStudio) and launch it again. You will see that Polybahn is not highlighted 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. Note that examples in the book use an older version of the Traffic library. If you want to try out these examples, you can download Traffic 3 from `https://bitbucket.org/nadiapolikarpova/traffic/downloads/traffic_3.zip`.

To hand in

There is nothing to hand in.

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