

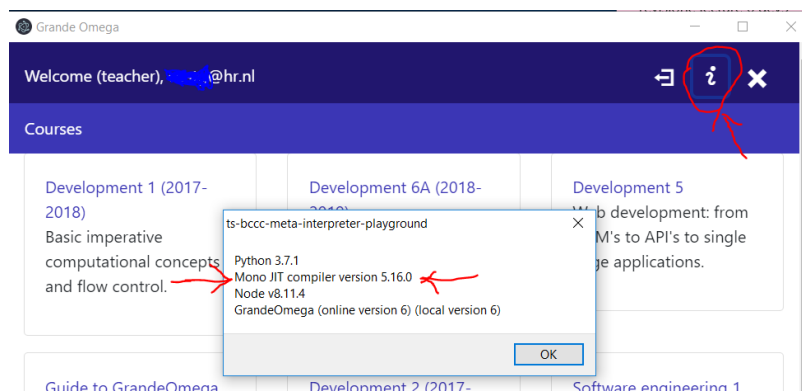
Guide to GrandeOmega (GO) for INFDEV3

Installation

- Download and install **node.js (LTS)** from: <https://nodejs.org/en/download/>
- Download and install **Mono (version 5.18.0)** from:
 - For MAC users: <https://download.mono-project.com/archive/5.18.0/macos-10-universal/MonoFramework-MDK-5.18.0.macos10.xamarin.universal.pkg>
 - For Windows users: <https://download.mono-project.com/archive/5.18.0/windows-installer/mono-5.18.0.248-gtksharp-2.12.45-win32-0.msi>
 - Make sure to have Mono added to your environment variables
 - The Mono executable is located in "C:\Program Files (x86)\Mono\bin"
 - You can check if Mono was successfully added to the environment variables by following the instructions in the following link: <https://www.architectryan.com/2018/03/17/add-to-the-path-on-windows-10/>
- Download the client of **GO** from:
 - http://grandeomega.com/go_student_win.zip (windows)
 - http://grandeomega.com/go_student_mac.zip (mac)
- **Unzip** the compressed folder downloaded at the previous step
- Execute the **GrandeOmega.exe** file:

d3dcompiler_47.dll	9/7/2018 11:16 AM	Application extension	4,077 KB
ffmpeg.dll	9/7/2018 11:16 AM	Application extension	1,910 KB
→ GrandeOmega.exe	9/7/2018 11:19 AM	Application	66,003 KB
icudtl.dat	9/7/2018 11:16 AM	DAT File	9,959 KB
libEGL.dll	9/7/2018 11:16 AM	Application extension	18 KB
libGLESv2.dll	9/7/2018 11:16 AM	Application extension	3,602 KB

- Check that everything is correctly set up by clicking the "i" button on Grande Omega (see following picture):



Use

- After the client starts, you need to login with your credentials (you have received via your student email instructions to get access):

Email	<input type="text"/>	Password	<input type="password"/>	Login
-------	----------------------	----------	--------------------------	-------

- After having logged in, you will see a screen with the courses you are subscribed to:

Development 1
Basic imperative computational concepts and flow control.

Development 6A
Introduction to algorithms.

Development 5
Web development: from ORM's to API's to single page applications.

Guide to GrandeOmega
An introductory guide to GrandeOmega.

Development 2
Functions, lambda expressions, recursion, higher order functions.

Development 6B
Advanced databases

Software engineering 1
Functors, monads, and other advanced data structures in practice.

Development 3
Static typing, object orientation, subtyping polymorphism

Development 4
Polymorphism, exception-handling, and basic design patterns

Software engineering 2
A categorical perspective on functors and monads via Bi-Cartesian Closed Categories.

Development 1 (2018-2019)
Basic imperative computational concepts and flow control.

- Clicking on a course, you will see the chapters of materials available for such course:

Chapter 1
Basics of Computation

Chapter 2
Variables, Expressions, and Operator Precedence

Chapter 3
Basic Data Types

Chapter 4
Control Flow

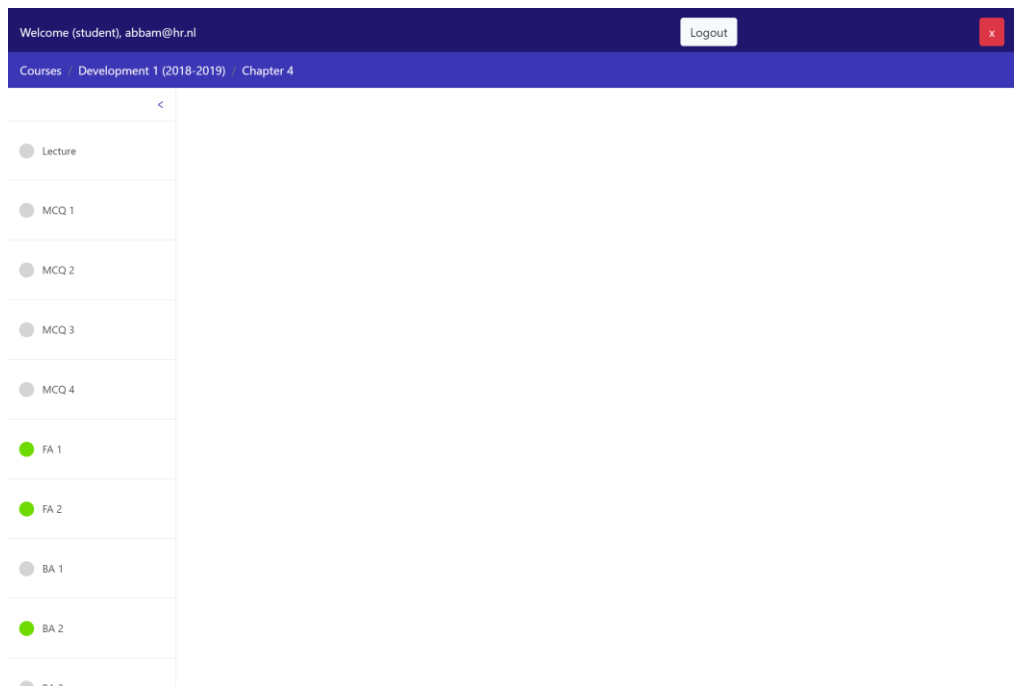
Chapter 5
Loops and Iteration

Chapter 6
Introduction to Python

Chapter 7
Some Sample Python Programs

Chapter 8
Declarative vs Imperative Style

- Clicking on a chapter, you will see the materials associated to such chapter in a column on the left of the screen. Click on the name of an item to open its associated content.



- A single chapter is usually composed by:
 - The reader of the corresponding lecture
 - A series of exercises which are a combination of:
 - Multiple Choice Questions (MCQ)
 - Forward Assignments (FA)
 - Backward Assignments (BA)
- During the practicum, the teachers will show you more in detail how to solve the Forward and Backward assignments.
- In short, a **Forward Assignment** shows you a program and the (sometimes incomplete) state associated to certain steps of the execution of such program (marked with red blocks to the left of the code). To solve a FA, you need to insert the missing values of variables in *all* incomplete states (remember to click “Next” until the last state is reached). For example:

1	x=5
2	y=1
3	x=3
4	y=x
5	

Reset

Globals

x	
---	--

Heap

Stack

Prev
Next

The state on the right (Globals, etc.) corresponds to the state of the program when the line of code marked with a yellow block is about to be executed (in the example above, when line 2 is about to be executed).

A **Backward assignment**, instead, shows you an incomplete program and the states associated to some steps of the execution of the complete program (again, marked by red/yellow blocks to the left of the code). By looking at such states, you should be able to fill in the missing parts of the program. For example:

1
2
3
4

a =
b = 1
 = 2

Validate Cancel validation

Globals

a	0
b	1
c	2

Heap

Stack

0 1 2 3

To see if your code solves the BA, click on “Validate” and you will get feedback.
When an assignment is correctly solved (both FA and BA) a “Success!” green message will appear on screen:

1
2
3
4

a =
b = 1
 = 2

Validate Cancel validation

Globals

a	0
b	1
c	2

Heap

Stack

0 1 2 3

Success!

Otherwise, a “Wrong!” red message appears (and in BAs the wrong values of your program are shown in red close to the correct ones in green in the state):

Welcome (student), caldj@hr.nl Logout

Wrong! Development 1 (2018-2019) Chapter 2

Lesson
Exercises
MCQ 1

1
2
3
4

a =
b = 1
 = 2

Validate Cancel validation

Globals

a	0	987
b	1	1
c	2	2

Heap

Stack

0 1 2 3

The round icon close to the assignment name in the left column also gets such color (orange for incomplete/wrong and green for complete):

● FA 5

● BA 1