Choose your language

Languages spoken by assistants:
- German (several varieties)
- English
- French
- Italian

Exercise sessions (Übungsgruppen) are available in German (5) and English (3).
- If there are enough requests, we will turn one of the English groups into an Italian group.

Choose an exercise session (i.e. an assistant) in your preferred language.

Goals of the course

After successfully taking this course, you will:
- Know the key concepts of programming.
- Understand basic hardware and software tools.
- Master a programming language: Eiffel.
- Know basic concepts of design, implementation and maintenance of large software systems (“software engineering”).

Teaching staff

& Practical information
About me

- At ETH for five years
- In industry most of my career
- Founded Eiffel Software in Santa Barbara, California, in 1985. Now "Chief Architect"
- Published a number of books, in particular *Object-Oriented Software Construction* (2nd edition: 1997)
- Plan: help the industry build the best software possible

Contact details

E-mail: Bertrand.Meyer@inf.ethz.ch
Office: RZ J22

Secretary: Claudia Günthart, 044 632 83 46
Office: RZ J7

Office hours: normally Thursdays during the semester, contact Ms. Günthart

Choosing an exercise group

Registration lists are available during the break. Choose your group according to

- Preferred language
- Your schedule constraints
- Availability

We may have to reassign students to a different group to keep the numbers balanced

The first exercise session takes place this afternoon!

Schedule

Lectures:
- Monday, 13:15 - 15:00, HG E7
- Tuesday, 8:15 - 10:00, HG F1

Exercise sessions:
- 7 groups
  - Monday, 15:15 - 16:00, various rooms
  - Tuesday, 13:15 - 15:00, various rooms
- 1 group
  - Tuesday, 13:15 - 14:00, IFW B42
  - Thursday, 15:15 - 17:00, HG 626.3

The assistants

Ilinca Ciupa
Philipp Kraehenbuehl
Andreas Leitner
Michela Pedroni
Beat Herlig
Hermann Lehner
Piotr Nienaltowski
Marco Piccioni
Bernd Schoeller

E-mail: hermann.lehner@inf.ethz.ch
Language: German
Office: RZ F6
Phone: 044 632 85 39
Rooms:
- Monday: ML J37.1
- Tuesday: ETZ E9
Andreas Leitner

E-mail: Andreas.Leitner@inf.ethz.ch
Office: RZ J4
Phone: 044 632 30 21
Language: German
Rooms:
- Monday: IFW A 36
- Tuesday: LEC C14

Mailing list: wirth@se.inf.ethz.ch

Course page:
se.inf.ethz.ch/teaching/ws2006/0001/index.html
- Check it at least twice a week

Exercise material:
- Lecture slides
- Textbook: Touch of Class (draft)
  Available electronically from course page

Also:
- Video recording of lectures!

Electronic forums

Discussion forums:
Help Forum for the whole course:
http://forum.vis.ethz.ch/
Mailing list for each group
Advice and rules:
- Use the forums and mailing list! Learning to program is hard: take advantage of every help you can get.
- Don’t be shy. There are no stupid questions.
- Criticism welcome, but always be polite to every participant and observe the etiquette.
To email the whole teaching team (professor and assistants):
eprog-assi@se.inf.ethz.ch

Your most important URL for the next 4 months

Course page:
se.inf.ethz.ch/teaching/ws2006/0001/index.html
- Check it at least twice a week

English version available, but German more up to date

Lecture material:
- Lecture slides
- Textbook: Touch of Class (draft)
  Available electronically from course page

Exercise material:
- Exercise sheets
- Master solutions

Also:
- Video recording of lectures!

If you need a laptop...

ETH has good prices through the NEPTUN program
Thinkpad (Lenovo, ex-IBM) or Apple
You choose your OS: Windows, Linux, MacOS
Limited time window: see www.neptun.ethz.ch

The textbook in progress

"Testat": Needed for admittance to fall exam.
Tasks:
- 7 weekly assignments
- 2 classroom exercises
- 1 project
To get the "Testat" you must have done at least:
- 5 weekly assignments
- Both classroom exercises
- The project
What you must do for each task:
- Show serious effort to address the questions
- Fill out questionnaire

Military services or illness ⇒ contact your assistant.

Exercises, "Testat" regulations
The software

The exercises rely on the Traffic "library"

Application domain: Transportation system in a city
   (in the book: Paris
   in this course: Zurich)

You will need to download:
   - EiffelStudio:
     eiffelsoftware.origo.ethz.ch/downloads/builds/
   - Traffic: traffic.origo.ethz.ch/download.html
   (this can wait until next week).

Warning

Not everything is perfect from our side.
The Traffic software probably has mistakes ("bugs"),
and the textbook does, too.

BUT:
   - We will correct our mistakes, as quickly as we can.
   - If you try something, don’t blame the software first. It may be doing just what you told it to.

Discovering Traffic

Why this approach?

Many software issues become really tough for big systems
With other approaches, in an intro course, you only see
small programs

We give you lots of software: use it as model & inspiration
You learn to use software through abstract interfaces
(also known as contracts)
You go from consumer to producer: outside-in
Traffic is graphical and fun
You should at the end be able to understand all of it.
Then you can add to it yourself

Behind the software

- Patrick Schoenbach (initial version)
- Michela Pedroni (current Traffic project leader)
- Till Bay (EiffelMedia library)

Numerous ETH students including:
   Marcel Kessler, Rolf Bruderer, Ursina Caluori,
   Roger Kung, Alan Fehr, Sarah Hauser, Michele Croci,
   Matthias Bühmann, Florian Gelmacher, Susanne
   Kasper, Lars Kropf, Valentin Wüsthofl, Stefan Daniel,

Tootor (Marie-Helene Ng Cheong Vee)

(Univ. of London)

Tool: helps you solve exercises, gives feedback

Data collection: help us improve the teaching.
Topics
- What is software?
- Objects & programs
- Interfaces and the notion of class
- Logic and contracts
- The run-time model: object creation, references
- Describing syntax
- Control structures
- Inheritance
- Genericty
- Recursion
- Data structures
- Event-driven programming & agents
- Topological sort
- Intro to software engineering

Basic advice
- Attend all lectures
- Read material — textbook, slides — before lecture
  (Note: slides often updated after lecture)
- Attend all exercise sessions
- Do all exercises & the project
  (you’ll need them for the “Testat”)
- If you don’t understand, ask
  (again: there are no stupid questions)
- If you have already programmed, take advantage of it,
  but take a fresh look at the material; explore Traffic
- If you are new to programming, don’t panic, it can be
tough at the beginning

The industry of pure ideas

Software engineers build machines
You can’t touch, kick or drop our machines: they’re immaterial
But they are machines anyway
We call them programs or systems

To operate (or run or execute) a program you need a
crphysical machine: a computer
Computers and related devices: hardware
Programs and associated intellectual value: software
**Software everywhere**

- Banking: manage millions of accounts
- Trading: decide to sell or buy
- Transportation: control trains, track planes...
  - Some cars have millions of lines of program code
- Travel: air, train, hotel reservations
- Communication: phones, Internet, ...
- Government: manage taxes, track laws...
- Health care: keep health record, control devices
- Education
- Entertainment
- Information
  - etc.

**Computers**

Computers are universal machines. They execute the program that you feed them.

The only limit is your imagination.

The good news:

- Your computer will do exactly what your program says

**Working with a computer**

- A programmer writes a program
- A user runs on a computer
- A user runs on their computer
Computers

Computers are universal machines. They execute the program that you feed them.
The only limit is your imagination.
The good news:
  ➢ Your computer will do exactly what your program says.
  ➢ It will do it very fast.

Moore’s “Law”

Approximate doubling of computer power, for comparable price, every eighteen months

Speed of Intel processors

Pentium IV: 3.8 GHz
80486: 50 MHz
80386: 33 MHz
8086: < 1 MHz

1 MHz
10 MHz
100 MHz
1 GHz


"Computers are intelligent"

Fact: Computers are neither intelligent nor stupid. They execute programs devised by humans. These programs reflect the intelligence of their authors. The basic computer operations are extremely elementary (store this value, add these two numbers...)

"The computer has crashed"
"The computer doesn’t allow this"
"The computer lost your record"
"The computer messed up your record"

Computers don’t make mistakes *....

➢ Programs don’t make mistakes either
➢ Programmers do make mistakes

*Actually, hardware can malfunction, but this is much more rare than program errors

"To err is human, but to really mess things up takes a computer"

Writing software is tough

Programs “crash”
Programs that don’t crash don’t necessarily work
Badly functioning programs have killed people, e.g. in medical devices
Ariane 5 rocket, 1996: $10 billion lost because of a simple program error

Programmers are responsible for the good functioning of their programs
The purpose of this course is to teach you not just programming but good programming
The “Blue Screen Of Death”

WARNING:
The system is either busy or has become unstable. You can wait and see if it becomes available again, or you can restart your computer.
* Press any key to return to Windows and wait.
* Press CTRL-ALT-DELETE again to restart your computer. You will lose unsaved information in any programs that are running.
Press any key to continue ...
Computers come in all sizes, colors, flavors

Computers everywhere

Banks
Airplanes, cars...
Washing machines
Cell phones (70% of value)
Printers
Tomorrow: your shirt...

Where’s the program?

Stored-program computer: the program is in memory. "Executable data".
The computer, with the help of some basic programs (operating system) knows how to find your program in memory to execute it.

A program can appear in memory in different forms:
- Source: human-readable form (programming language).
- Target form, machine code, object form: form executable by the computer.
Compilers transform source text to machine code.

Software Engineering

Writing software that’s
- Correct
  Does what it’s supposed to!
- Extensible
  Easy to change!
- Readable
  by humans!
- Reusable
  Don’t reinvent the wheel!
- Robust
  React appropriately to errors

Computers are universal machines. They execute the program that you feed them.

(Universal machine)

Operating systems: source size

Lines of code (millions)


- Unix V7: 10K
- Windows NT: 4 M
- Windows 95: 15
- Windows 98: 18
- Solaris 7: 12
- Windows 2000: 40
- Red Hat 6.2: 17
- Red Hat 7.1: 30
- Vista: 50
- Debian 3.1: 213
- Debian 2.2: 55
- Windows XP: 45
Intro. to Programming, lecture 1: Overview

In other application areas

(source: Siemens)

Writing software is tough

Programs crash, etc.

For next week

Read chapter 1 of Touch of Class

Recommended: start reading chapter 2 and slides for next lecture

Writing software is fun

Design and build your own machines

Exert your creativity and imagination

Programs save lives and help make the world better

Experience the feeling of a program that you wrote, and that works

End of lecture 1