ETH Zürich
Chair of Software Engineering

Einführung in die Programmierung
Introduction to Programming

Prof. Dr. Bertrand Meyer

Exercise Session 2

Organizational Notes

- Assignments
  - One assignment per week
  - Will be put online Monday (around 18:00)
  - Should be handed in within nine days (Wednesday, before 23:59)
  - Hand in via Email with subject:
    Neumann2015-#1-Your Name
- Grading
  - Assignments : not graded, feedback provided on request
  - Mock exams : graded but do not affect the final grade
  - Final exam : graded
- Group mailing list:
  - Is everybody subscribed (got an email)?

Today

- Important Concepts from the Lectures

- Programming in Eiffel
  - Eiffel Compilation Process
  - Working with EiffelStudio
  - Found a bug in EiffelStudio?

Important Concepts from the Lectures

A Small Eiffel Program

```
class CIRCLE

feature -- Access
  radius: REAL
      -- Radius of the circle.

feature -- Query
  area: REAL
      -- Area of the circle.
    do
      Result := 3.14 * radius ^ 2
    end

feature -- Status set
  set_radius (a_radius: REAL)
      -- Set `radius' with.
    do
      radius := a_radius
    end

end
```

```
class APPLICATION

-- Some details omitted here

feature -- Initialization
  make
      -- APPLICATION ENTRY POINT.
    local
      l_circle: CIRCLE
      l_area: REAL
    do
      create l_circle
      l_circle.set_radius (1.0)
      l_area := l_circle.area

      io.put_string ("The area is ")
      io.put_real (l_area)
      io.put_string (".")
    end
end
```

Command-Query Separation Principle

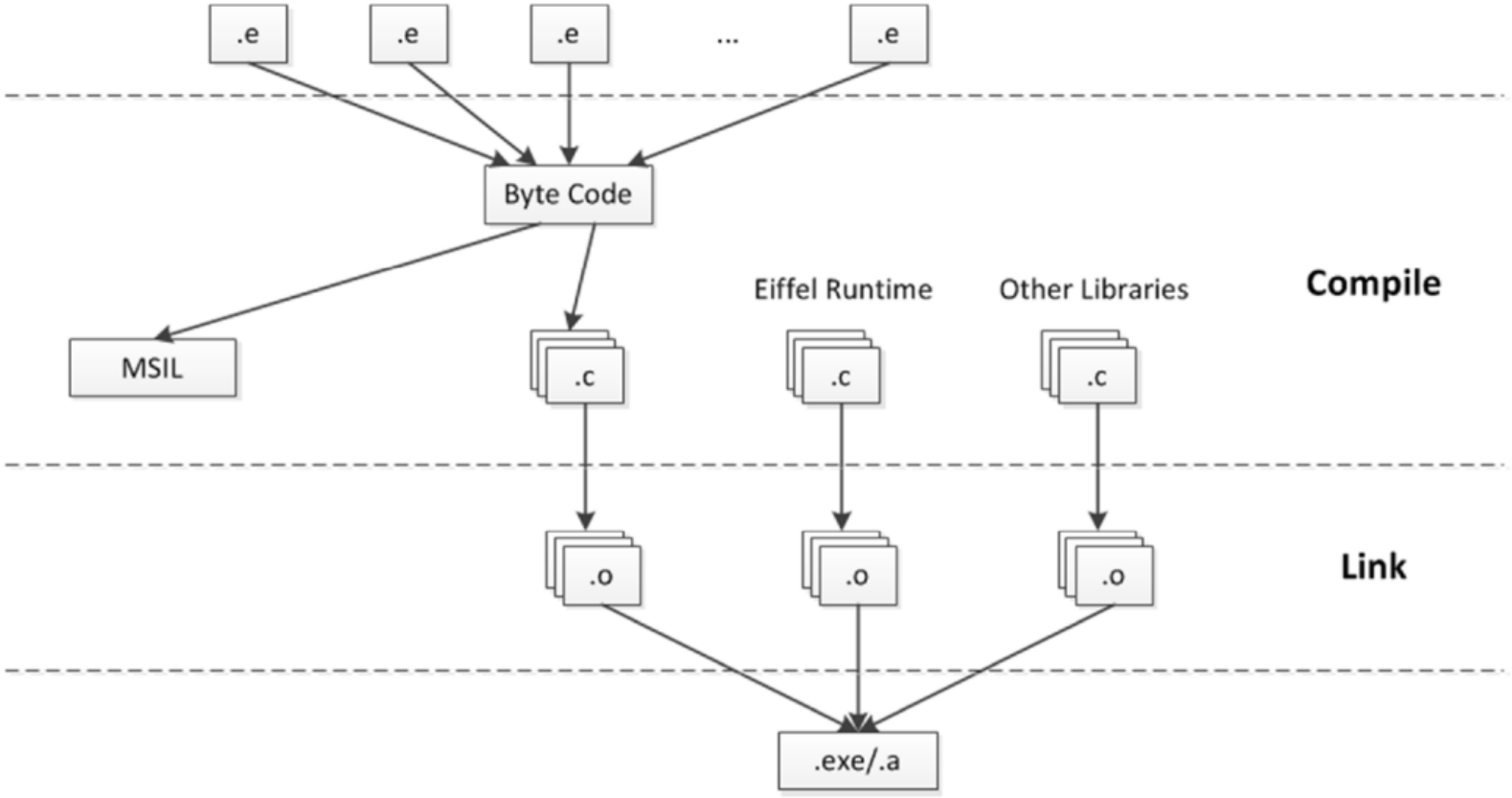
"**Asking** a question **shouldn't change** the answer"

i.e. a query

Programming in Eiffel

Eiffel Compilation Process

- Why is compilation necessary?



- Compilation vs. interpretation

Eiffel Compiler Highlights

- Melting: uses quick incremental recompilation to generate bytecode for the changed parts of the system. Used during development (corresponds to the button "Compile").
- Freezing: uses incremental recompilation to generate more efficient C code for the changed parts of the system. Initially the system is frozen (corresponds to "Freeze…").
- Finalizing: recompiles the entire system generating highly optimized code. Finalization performs extensive time and space optimizations (corresponds to "Finalize…"), this may take longer.

Working with EiffelStudio

- Components
  - Editor
  - Tool panes: Groups, Features, Class, Feature
  - Menu and toolbars
  - Customizing the UI

- Basic operations
  - Create/open
  - Navigate
  - Edit
  - Compile
  - Run

Debugger: Setup

- The system must be melted/frozen (finalized systems cannot be debugged).
- Setting and unsetting breakpoints
  - An efficient way consists of dropping the feature you want the breakpoint in, into the context tool.
  - Alternatively, you can select the flat view.
  - Then click on one of the little circles in the left margin to enable/disable single breakpoints.
- Use the toolbar debug buttons to enable or disable all breakpoints globally.

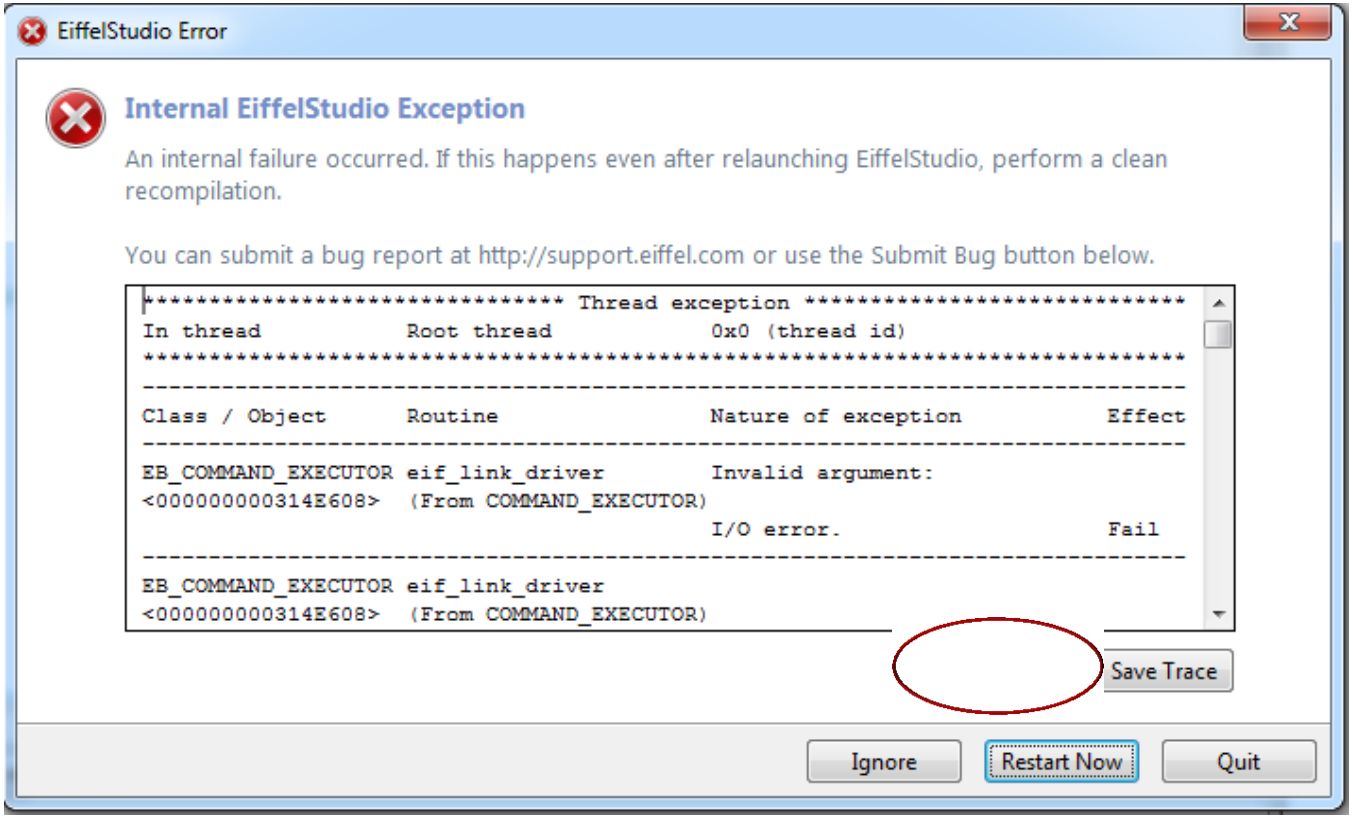
Debugger: Run

- Run the program by clicking on the Run button.
- Pause by clicking on the Pause button or wait for a triggered breakpoint.
- Analyze the program:
  - Use the call stack pane to browse through the call stack.
  - Use the object tool to inspect the current object, the locals and arguments.
- Run the program or step over (or into) the next statement, or out of the current one.
- Stop the running program by clicking on the Stop button.

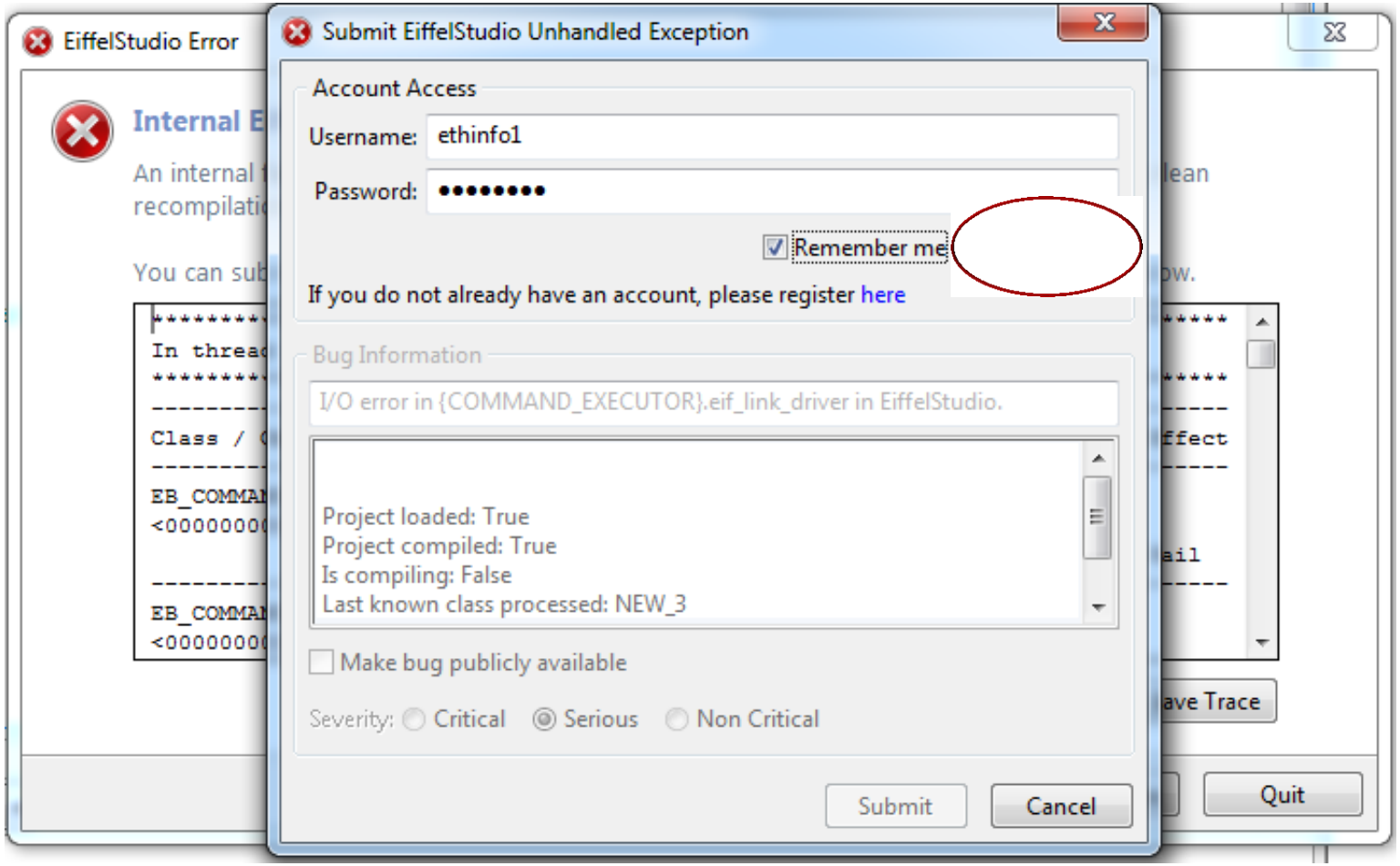
Found a Bug in EiffelStudio?

If EiffelStudio happens to crash:

- You should submit an official bug report by pressing the button that appears when EiffelStudio crashes

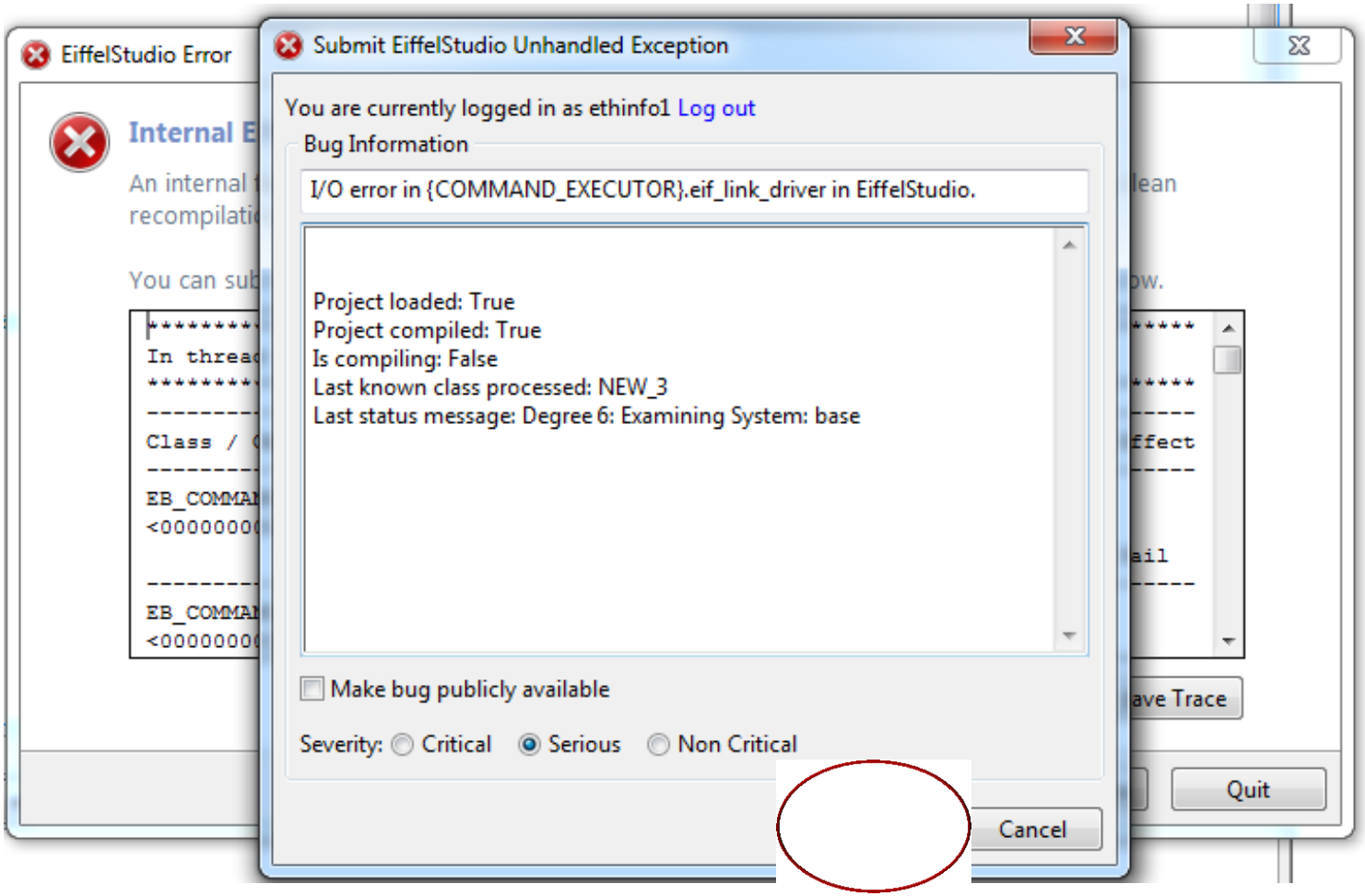


How to Submit a Bug: Login



- Username: ethinfo1, Password: ethinfo1

How to Submit a Bug: Submit



~ End ~