Game Physics Programming Exercise Intro

Felix Koehler - 30 October 2023

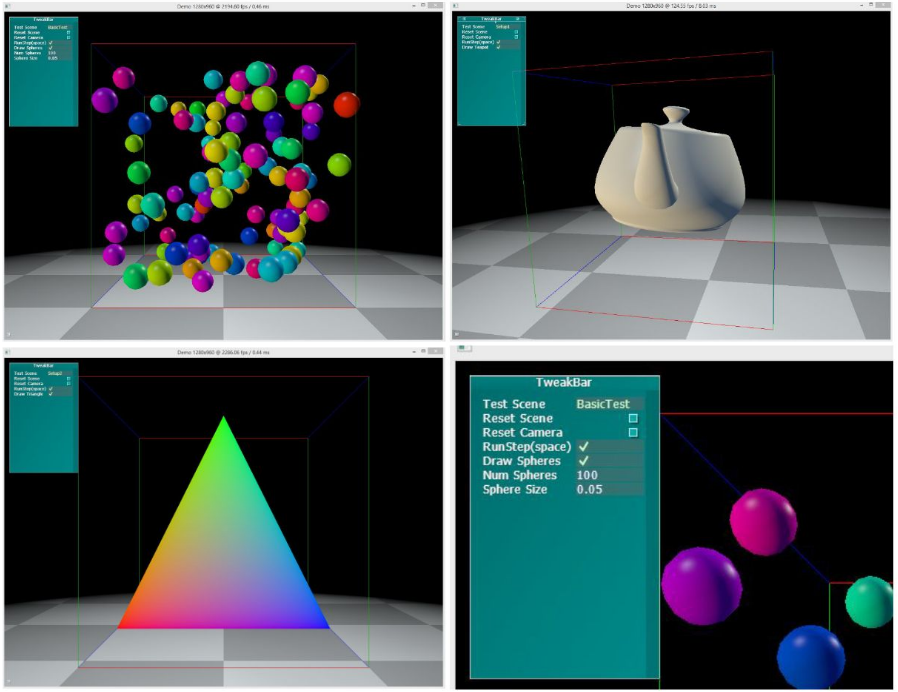
Goal

- Have a practical understanding of the topics discussed in the course.
- Performing the translation: Math -> Code
- Team work.

I did not properly understand something before I can teach it to a computer.

Template Project

- DirectX-based Rendering
- Simple GUI
- Math Utilities: Quaternion, Matrix, Vector …
- Requirements:
  - Windows machine (use Rechnerhalle if you do not own one)
  - Visual Studio (we recommend using **Visual Studio 2022 Community Edition**)



TweakBar

Test Scene	BasicTest
Reset Scene	□
Reset Camera	□
RunStep(space)	✓
Draw Spheres	✓
Num Spheres	100
Sphere Size	0.05

Overview

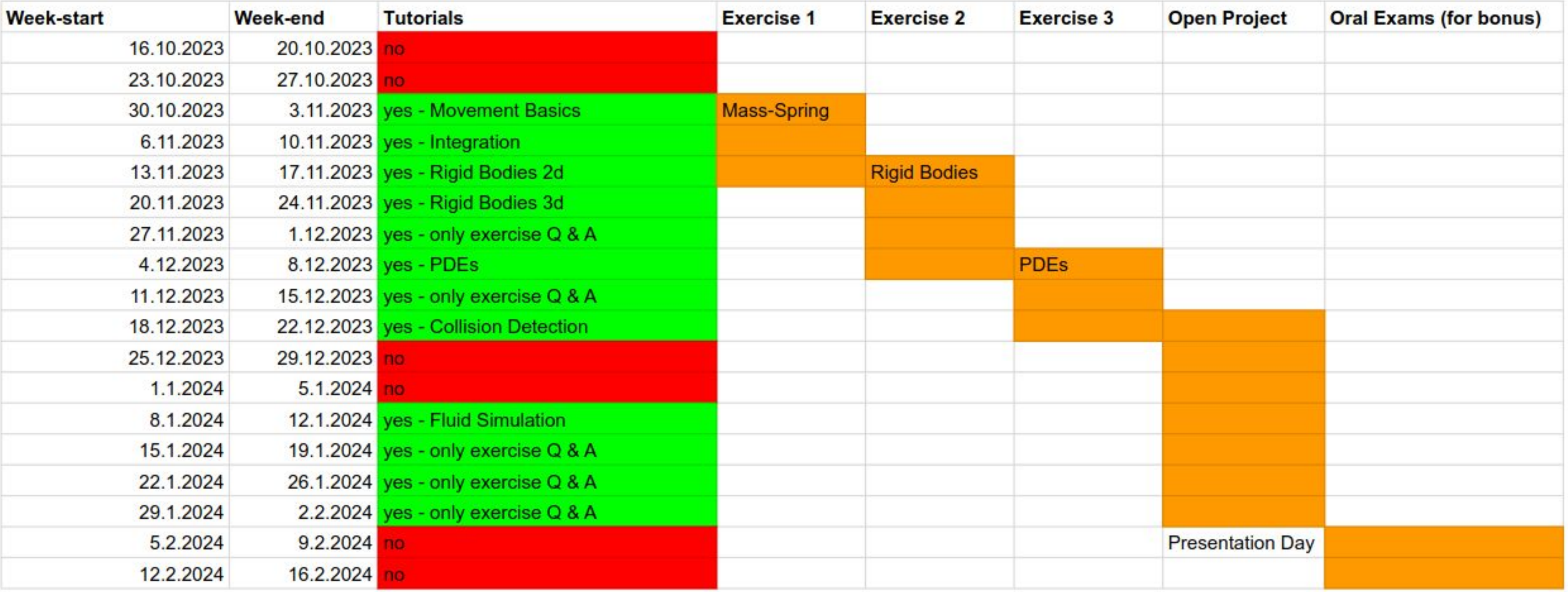
- 3 Programming Tasks + 1 Open Project
  - Template project on GitHub: https://github.com/tum-pbs/gamephysicstemplate
  - Exercise sheets will be released on Moodle;
  - Interactive physics demos
    - Add your implementations;
    - Interactively show how the algorithm works (mouse, keys, buttons).

- Groups of **3-4** members
  - Registration via Moodle
  - Submission of source code files via Moodle

Reward: Grade Bonus

- Grade bonus of 0.3, for:
  1. Source code submitted to Moodle in time (for 3 exercises and 1 open project):
     - A zipped package of your source files and project files (under the “Simulations” directory)
     - **No exe or compiler temp files**
     - **Make sure your project can compile and run successfully on Visual Studio 2022 Community Edition!**
     - **Submitting repository links (e.g., github) is NOT allowed**
     - No additional libraries allowed
  2. Demonstrate 3 exercises at the end of the semester (“Oral Exam”)
     - With the exe file generated from your source files, no slides
     - ~15 min, every team member does one part of the demonstration
     - Answer Questions (every team member should know details)
  3. Presentation of the open project in the last lecture (“Presentation”)

Time Schedule



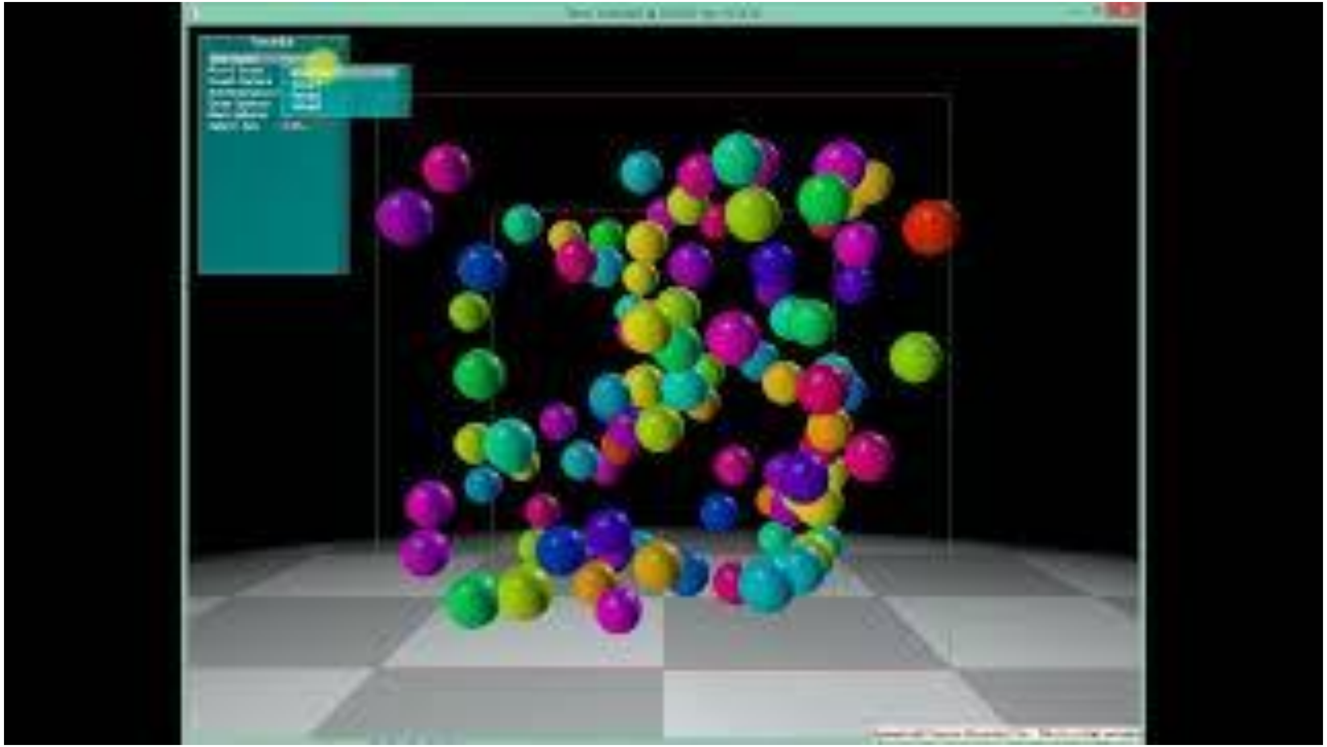
Week-start	Week-end	Tutorials	Exercise 1	Exercise 2	Exercise 3	Open Project	Oral Exams (for bonus)
16.10.2023	20.10.2023	no					
23.10.2023	27.10.2023	no					
30.10.2023	3.11.2023	yes - Movement Basics	Mass-Spring				
6.11.2023	10.11.2023	yes - Integration	🟧				
13.11.2023	17.11.2023	yes - Rigid Bodies 2d	🟧	Rigid Bodies			
20.11.2023	24.11.2023	yes - Rigid Bodies 3d		🟧			
27.11.2023	1.12.2023	yes - only exercise Q & A		🟧			
4.12.2023	8.12.2023	yes - PDEs		🟧	PDEs		
11.12.2023	15.12.2023	yes - only exercise Q & A			🟧		
18.12.2023	22.12.2023	yes - Collision Detection			🟧	🟧	
25.12.2023	29.12.2023	no				🟧	
1.1.2024	5.1.2024	no				🟧	
8.1.2024	12.1.2024	yes - Fluid Simulation				🟧	
15.1.2024	19.1.2024	yes - only exercise Q & A				🟧	
22.1.2024	26.1.2024	yes - only exercise Q & A				🟧	
29.1.2024	2.2.2024	yes - only exercise Q & A				🟧	
5.2.2024	9.2.2024	no				Presentation Day	🟧
12.2.2024	16.2.2024	no					🟧

Support

- Three Tutors
  - Tuesdays from 14:00 until 16:00 (Siyun Liang - siyun.liang@tum.de, room 00.13.008)
  - Wednesdays from 14:00 until 16:00 (Markus Gumbart - ge25lim@mytum.de, room 00.08.055)
  - Thursdays from 14:00 until 16:00 (David Berger - david.berger@tum.de, room 01.09.014)
- Zulip forum:
  - Monitored by the tutors and me
  - Try ask questions and help each other (you learn a lot from making someone else understand)

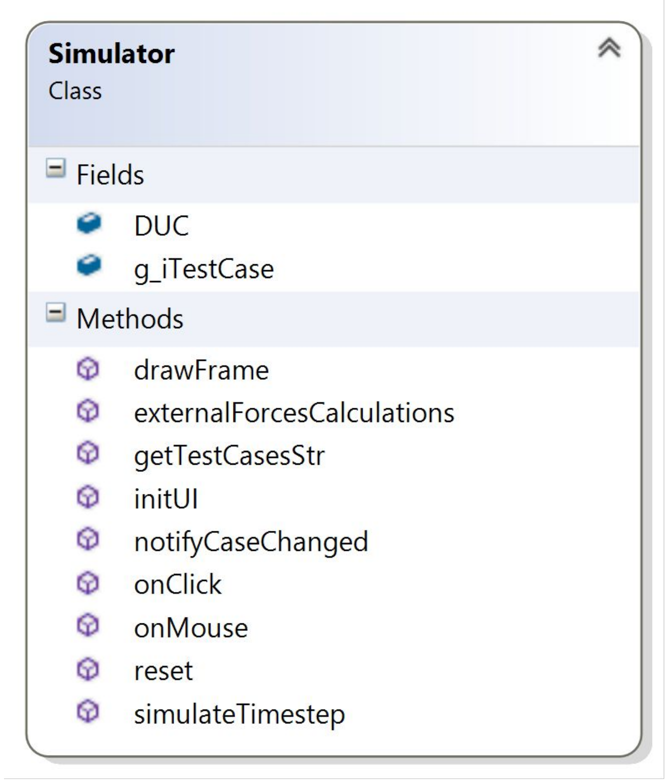
Let’s take a look at the template

Template Simulator video if live demo did not work



Simulator Base class

- Each programming task should inherit from the Base Simulator class



Simulator
Class

Fields

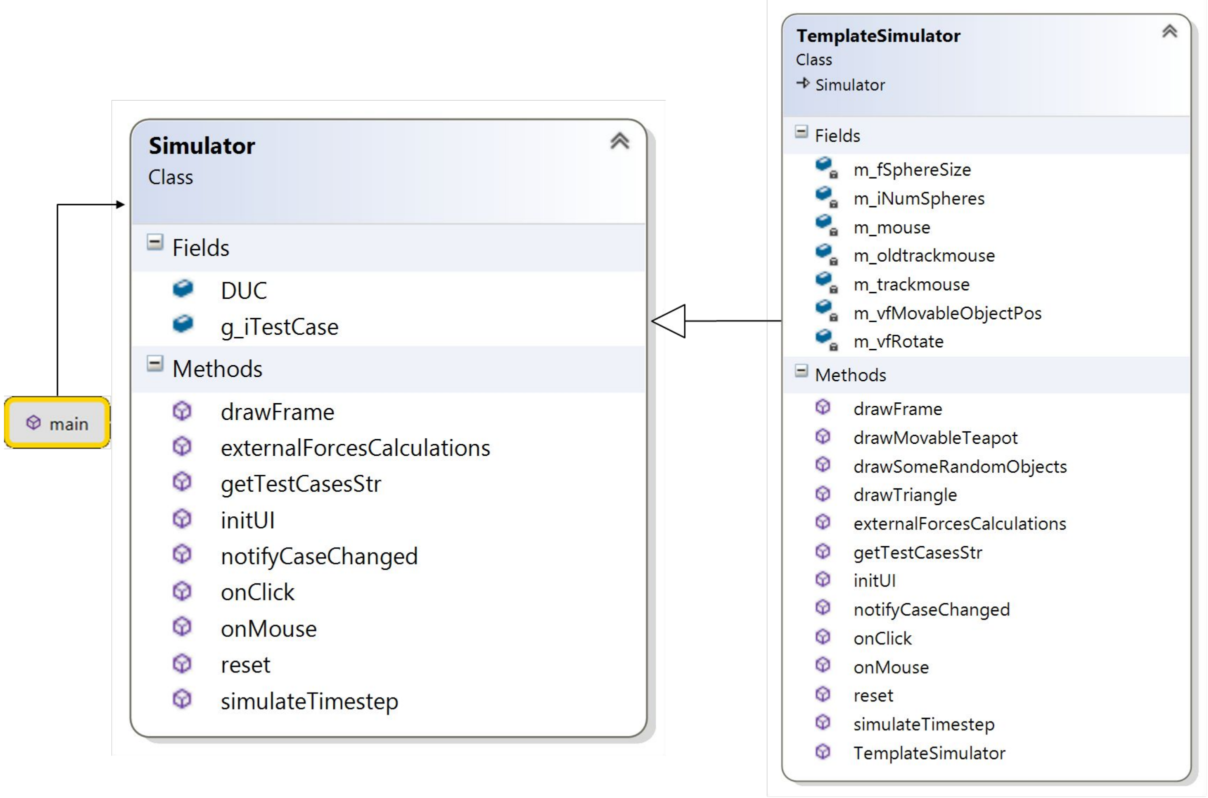
- DUC
- g_iTestCase

Methods

- drawFrame
- externalForcesCalculations
- getTestCasesStr
- initUI
- notifyCaseChanged
- onClick
- onMouse
- reset
- simulateTimestep

The Template Simulator

- Like the Template Simulator you just saw



main

Simulator
Class

Fields
- DUC
- g_iTestCase

Methods
- drawFrame
- externalForcesCalculations
- getTestCasesStr
- initUI
- notifyCaseChanged
- onClick
- onMouse
- reset
- simulateTimestep

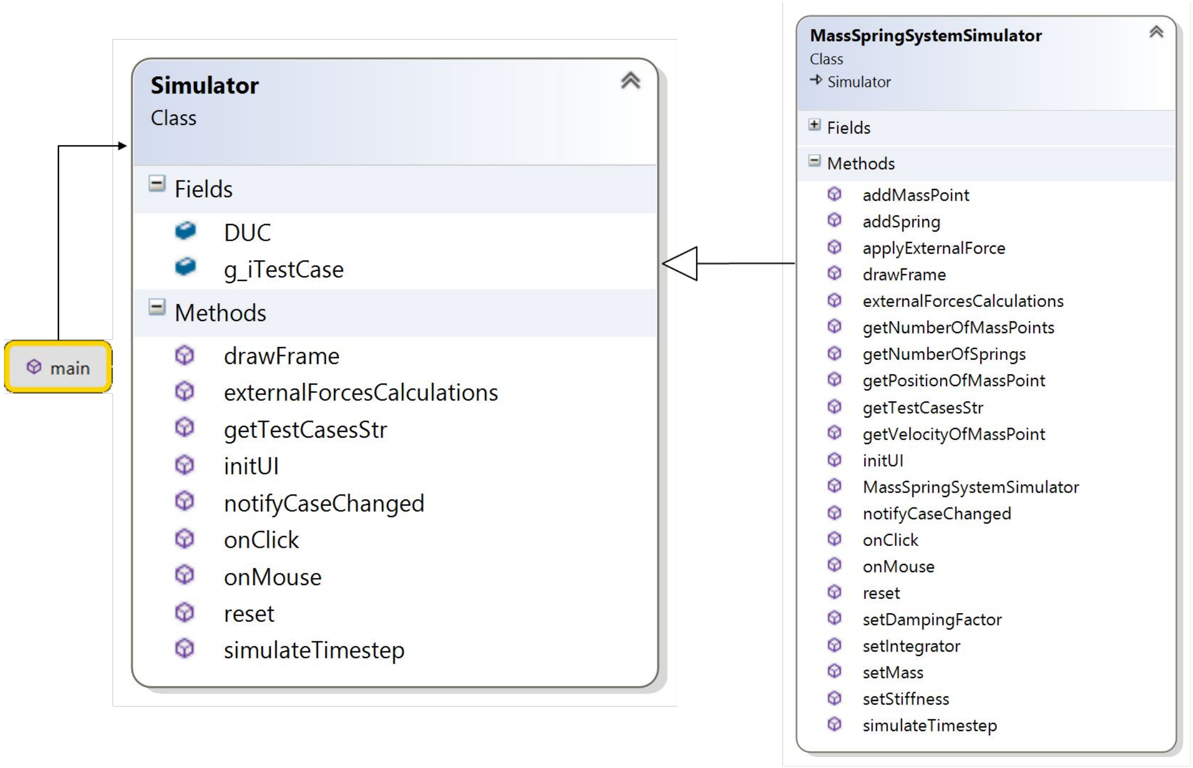
TemplateSimulator
Class
→ Simulator

Fields
- m_fSphereSize
- m_iNumSpheres
- m_mouse
- m_oldtrackmouse
- m_trackmouse
- m_vfMovableObjectPos
- m_vfRotate

Methods
- drawFrame
- drawMovableTeapot
- drawSomeRandomObjects
- drawTriangle
- externalForcesCalculations
- getTestCasesStr
- initUI
- notifyCaseChanged
- onClick
- onMouse
- reset
- simulateTimestep
- TemplateSimulator

Exercise 1: Mass-Spring Simulator

- We provide you with a header file, you have to implement all methods
- **Don’t delete given members, defines, functions…**
- Feel free to add other necessary functions, data members



main

Simulator
Class

Fields
- DUC
- g_iTestCase

Methods
- drawFrame
- externalForcesCalculations
- getTestCasesStr
- initUI
- notifyCaseChanged
- onClick
- onMouse
- reset
- simulateTimestep

MassSpringSystemSimulator
Class
→ Simulator

Fields

Methods
- addMassPoint
- addSpring
- applyExternalForce
- drawFrame
- externalForcesCalculations
- getNumberOfMassPoints
- getNumberOfSprings
- getPositionOfMassPoint
- getTestCasesStr
- getVelocityOfMassPoint
- initUI
- MassSpringSystemSimulator
- notifyCaseChanged
- onClick
- onMouse
- reset
- setDampingFactor
- setIntegrator
- setMass
- setStiffness
- simulateTimestep

Recommendations

- Try to understand the Template Simulator first
- Use Version Control: GIT
- Meet and communicate!
    - Every week, not just before the deadlines!
    - Every team member should implement all exercises.
    - **Avoid** only working with CodeShare and similar tools.
        - For the oral exam every team member has to be familiar with the implementation!
    - Merge your code after every team member has finished their version.
- If you don’t use Visual Studio 2022, make sure your final code can be compiled with it.

Questions?

(Don’t forget to sign up for a group)