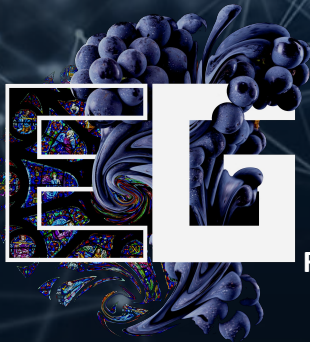




An open-source solution for interactive physics simulation

Room 3 - 09.00 - 12.30



Reims 2022



SOFA

Simulation
Open
Framework
Architecture

An open-source solution for collaborations,
prototyping and innovation in simulation

Monday, 25th April 2022 (Reims - France)

Overview of the 3h30

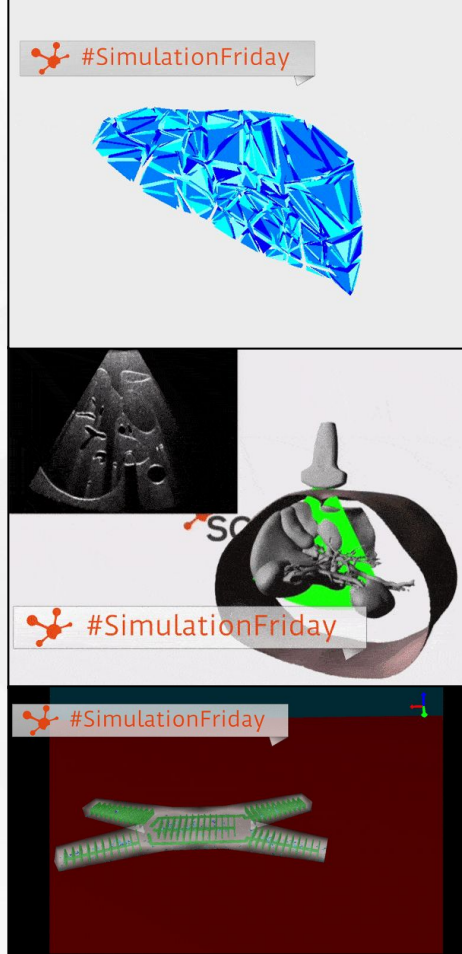
- Introduction on biomechanical simulation with SOFA
 - What is SOFA?
 - A development project
 - Its international community
- Understanding SOFA at a glance
 - Main principles
 - Life cycle
- User tutorial: hands-on!
 - Download SOFA
 - Building a simulation step by step in XML
 - The power of Python

What is SOFA?



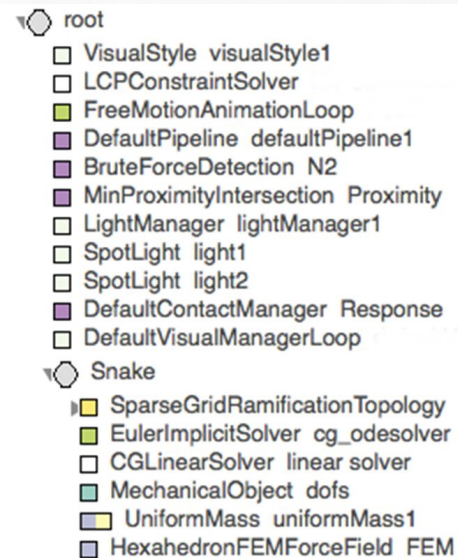
A framework for physics simulation

- Implements mathematical models & algorithms describing the physics around us
 - Soft and rigid body dynamics (mostly)
 - Heat transfer
 - Fluid mechanics
- Various fields being investigated
 - Medical simulation
 - Robotics and control → soft-robotics
 - Animation
 - Biology



SOFA principles: scene graph

- Simulation described as a graph
(Direct Acyclic Graph, i.e. generalized hierarchy)
 - Composed of nodes
 - Nodes contain **Components**
 - Components have parameters: **Data**
- Data can be linked together (*input=@dofs.value*)
- Described using XML or Python

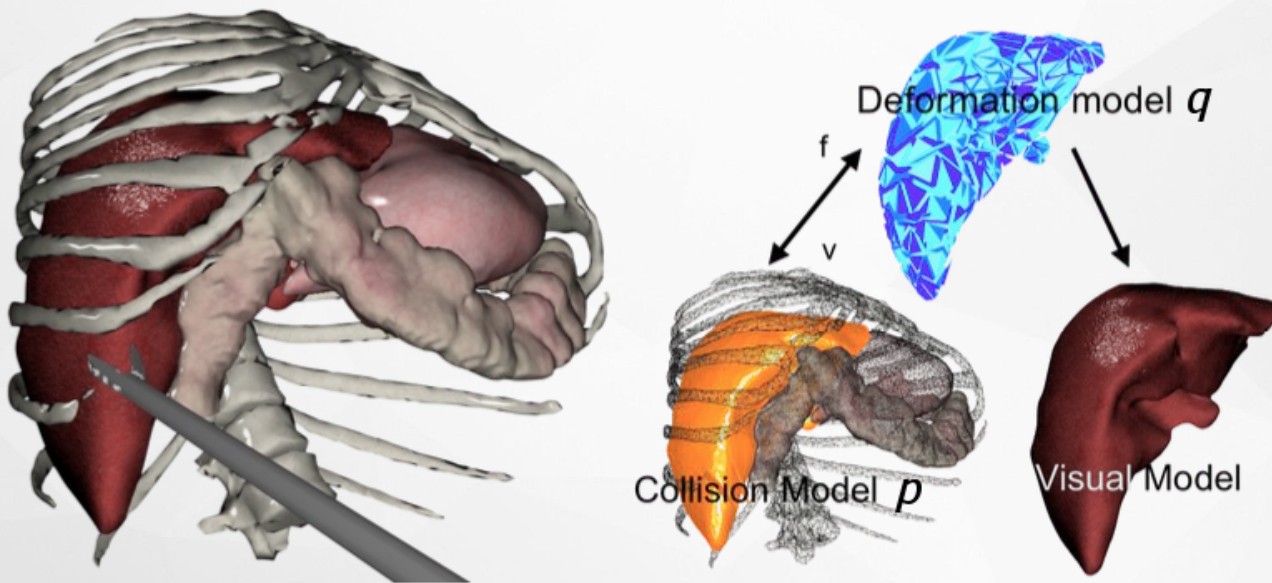


SOFA principles: multi-model representation

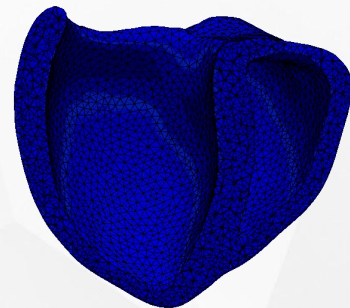
- Matching mechanism : correspondence between models
- Each object can have several models in SOFA
 - Mechanical model
 - Collision model
 - Visual model
- Each model can rely on a different representation/topology

SOFA principles: multi-model representation

- Matching mechanism : correspondence between models



Usual physics features



- Loading topologies
 - Discretization in space with elements (FEM)
 - Different topologies can be used for physical, visual and collision models
- Collision detection and response : CollisionPipeline
- Integration schemes and linear solvers
- Mechanical models
 - Linear elasticity, plasticity
 - Hyperelasticity
 - Damping



The SOFA concept

- Scene graph
 - Ease design and prototyping of simulation
- Multi-model representation
 - Define the optimal tradeoff between accuracy and efficiency
 - Along with multithreading / GPU computing
- Interactive
 - User interactions to take into account
 - Enclose SOFA in optimization loops
 - Dynamic simulation updates



A development project



Structure of the open-source project

- SOFA: core project
 - C++ based, coming with python bindings
 - LGPL License
 - Download and use it
 - Modify it
 - Make products from SOFA!
- External repositories
 - Implementing additional features
 - Free-to-define license: open source or closed source? YOU choose!



Code production of SOFA



- GitHub processes
 - Contribution process (fork and pull-requests)
 - Peer-review process
 - Continuous integration for robustness (unit, feature and regression tests)
 - Stable releases every 6 month (including 360+ pull-requests!)
- Definition of the roadmap: technical governance
 - Bi-annual technical committee (STC)
 - Weekly development meetings

Documentation & support

- User online documentation → sofa-framework.org/community/doc
- API documentation for devs → sofa-framework.org/api
- Open forum → github.com/sofa-framework/sofa/discussions
- Open chat → gitter.im/sofa-framework/sofa

Its international community



Community

- Developed from more than 16 years with +1000 worldwide users
- Worldwide network
 - dissemination & communication of your production
 - collaborations (bilateral, EU etc.)
 - hiring or finding the ideal job
- Today
 - Worldwide universities 80%
 - New international startups 10%
 - International companies 8%
 - Independent developers 2%

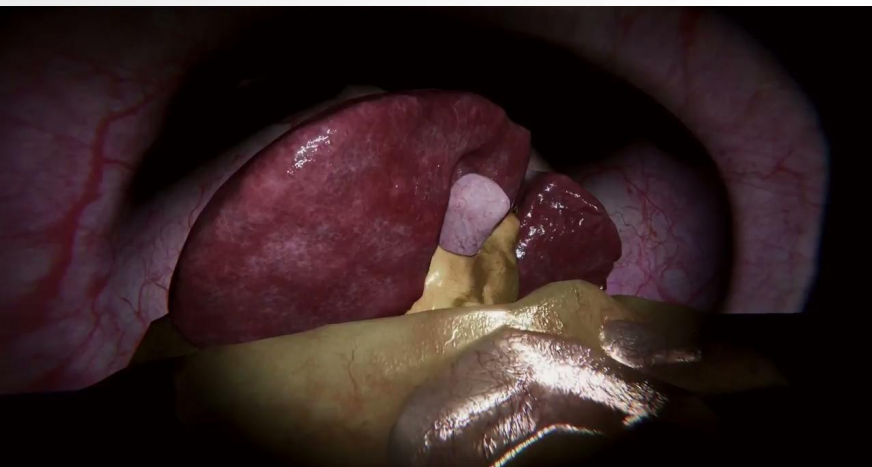


Community

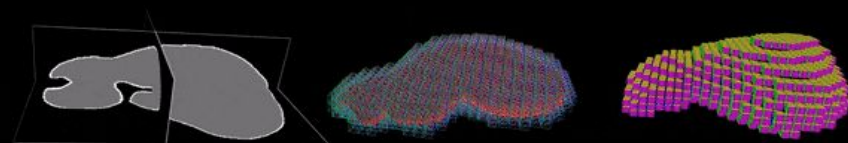
- Developed from more than 16 years with +1000 worldwide users
- Startups: 7 companies created for the last decade
- SOFA Week
 - Annual community event
 - Gathering ~200 participants
 - Topics



Recent achievements with SOFA



BUILD PATIENT-SPECIFIC FEM MODEL



Automatic generation of the FEM model
Hyperelastic (Saint-venant kirchhoff) material
Immersed Boundary technique to handle partially-filled elements



An open-source consortium

- Gathers partners willing to share the costs of
 - Coordination of the developments
 - Maintenance and industrialization
 - Animation and support to the community
 - Structure the ecosystem
- Acting as a third-party and non-profit organization
- Hosted at *Inria*

An open-source consortium

- Strategic partners, steering the strategy of the project

Inria

- Technical members, defining the roadmap based on contributions



- Donors



UNIVERSITÀ
di VERONA



Be part of the community



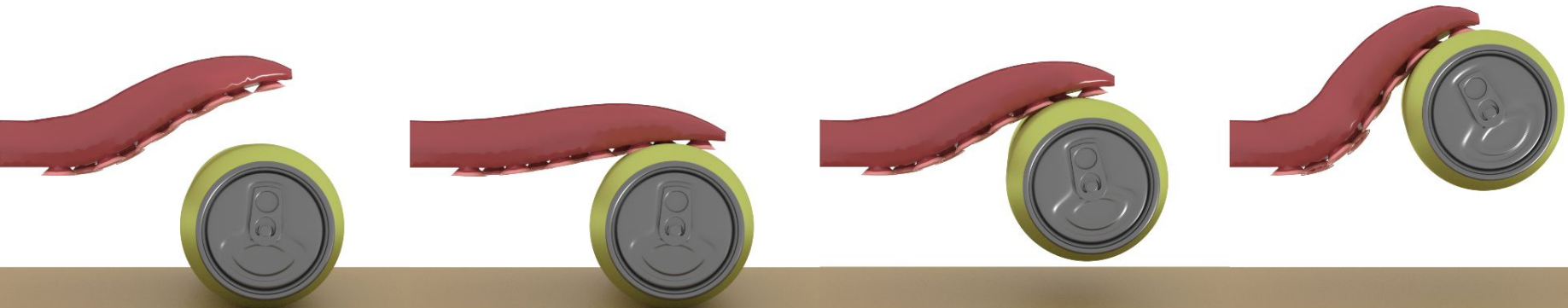
- Discover SOFA, enter the community and benefit from it!
- Join us at our 2022 events
- Any question?

Web

www.sofa-framework.org

Contact

consortium@sofa-framework.org



Overview of the 3h30

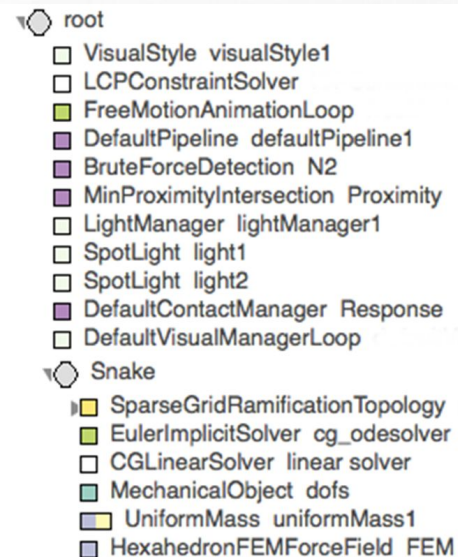
- Introduction on biomechanical simulation with SOFA
 - What is SOFA?
 - A development project
 - Its international community
- Understanding SOFA at a glance
 - Main principles
 - Life cycle
- User tutorial: hands-on!
 - Download SOFA
 - Building a simulation step by step in XML
 - The power of Python

Main principles

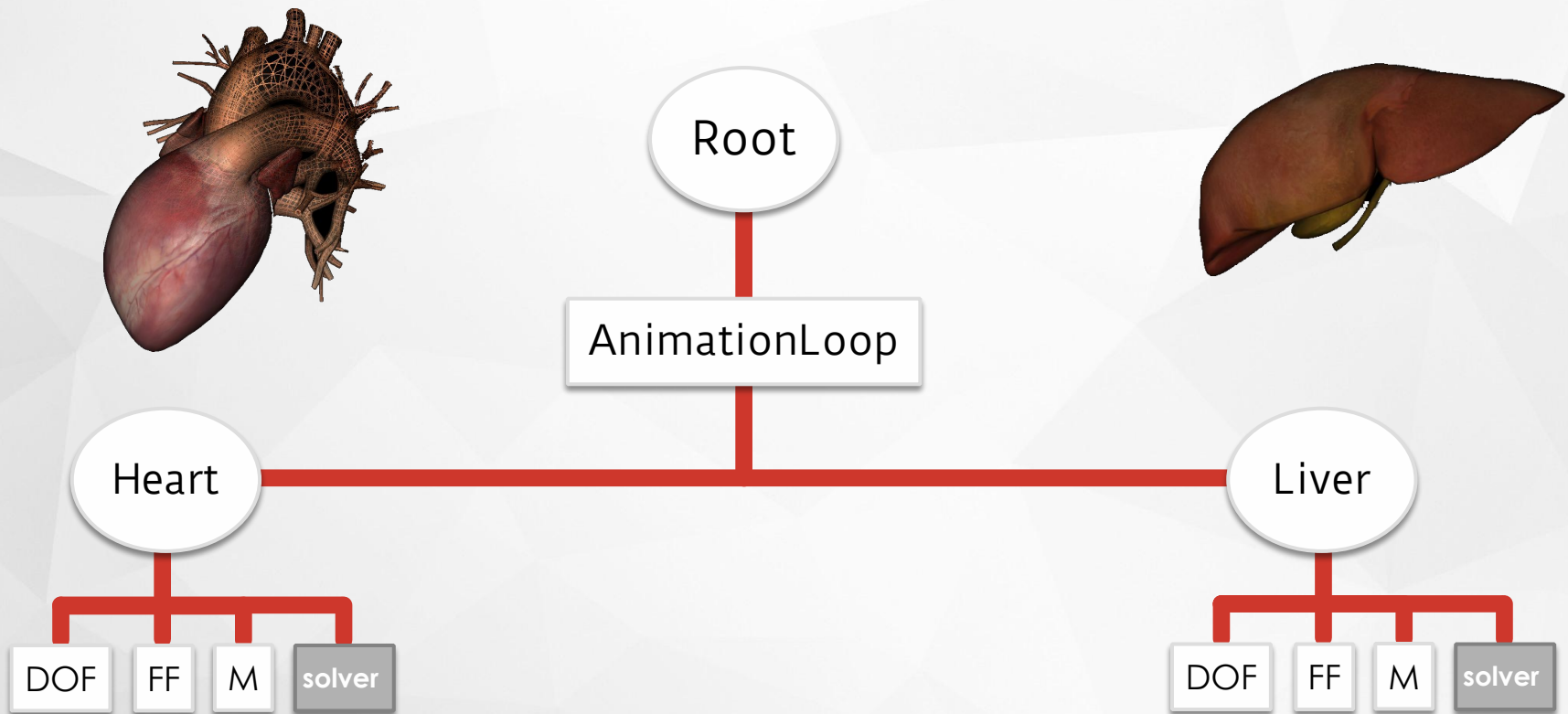


Scene Graph

- Simulation described as a graph
(Direct Acyclic Graph, i.e. generalized hierarchy)
 - Composed of nodes
 - Nodes contain **Components**
 - Components have parameters: **Data**
- Data can be linked together (*input=@dofs.value*)
- Described using XML or Python



Scene Graph



Scene Graph (XML format)

```
<Node name="root" dt="0.01" gravity="0 0 -9.81">

  <Node name="Liver">
    <EulerImplicitSolver name="EulerImplicit_scheme" />
    <CGLinearSolver name="CG" iterations="200" tolerance="1e-09"/>
    <MechanicalObject template="Vec3d" name="myLiverDOF" position="..." />
    <MeshMatrixMass totalmass="1" />
    <TetrahedronFEMForceField name="LinearElasticity" youngModulus="1e7" />
  </Node>

  <Node name="Heart">
    <EulerImplicitSolver name="EulerImplicit_scheme" />
    <CGLinearSolver name="CG" iterations="200" tolerance="1e-09"/>
    <MechanicalObject template="Vec3d" name="myHeartDOF" position="..." />
    <MeshMatrixMass totalmass="2" />
  </Node>

</Node>
```

1. Scene Graph (Python format)

```
import Sofa
import SofaRuntime

def createScene(rootNode):
    rootNode.dt=0.01
    rootNode.gravity="0 0 0"

    liverNode = rootNode.addChild("Liver")
    liverNode.addObject(EulerImplicitSolver)
    liverNode.addObject('CGLinearSolver', iterations="200", tolerance="1e-09...")
    liverNode.addObject('MechanicalObject', template="Vec3d", name="myLiverDOF", position="...")
    liverNode.addObject('MeshMatrixMass', name="mass", totalMass="1" )
    liverNode.addObject('TetrahedronFEMForceField', name="LinearElasticity", youngModulus="1e7")

    heartNode = rootNode.addChild("Heart")
    heartNode.addObject(EulerImplicitSolver)
    heartNode.addObject('CGLinearSolver', iterations="200", tolerance="1e-09...")
    heartNode.addObject('MechanicalObject', template="Vec3d", name="myHeartDOF", position="...")
    heartNode.addObject('MeshMatrixMass', name="mass", totalMass="2" )
```

1. Scene Graph (Python format)

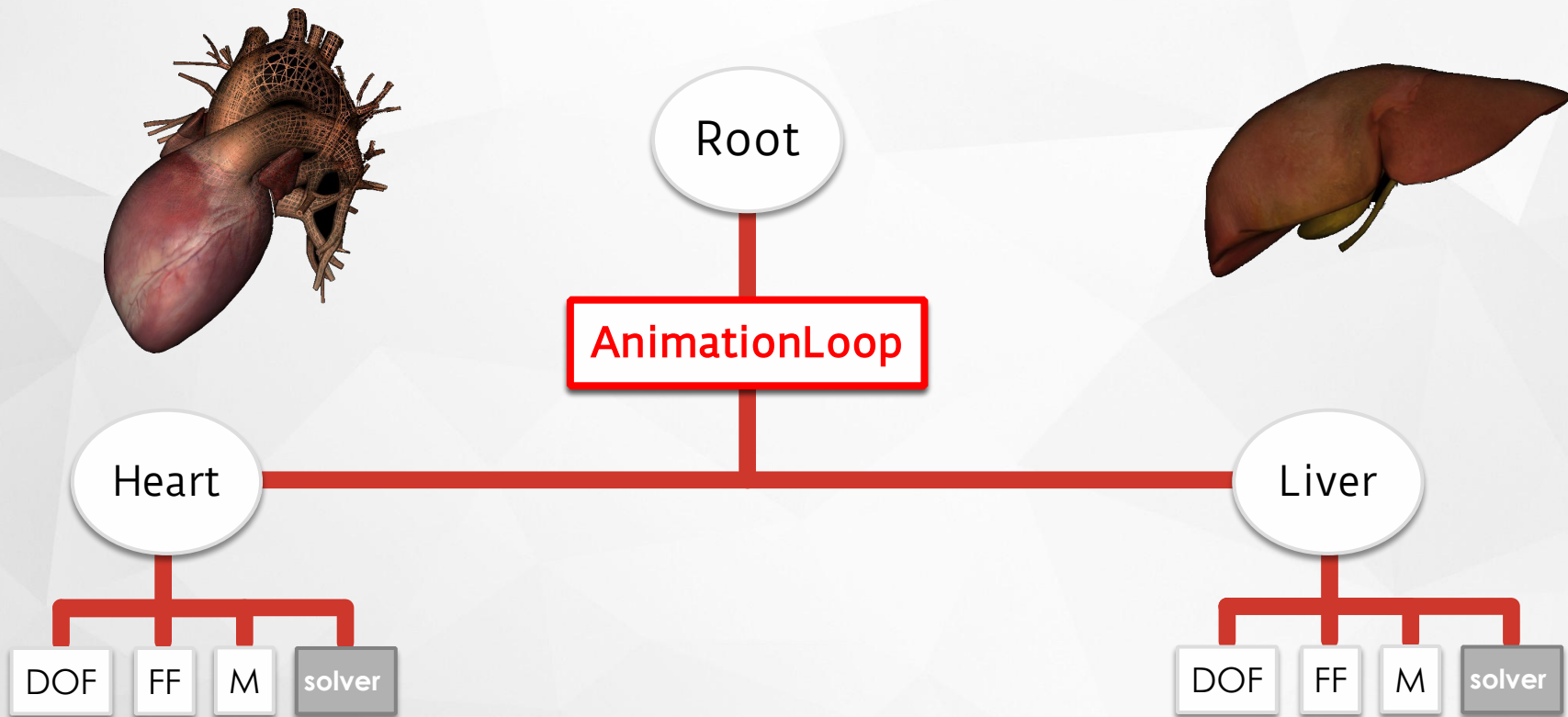
```
import Sofa
import SofaRuntime

def createScene(rootNode):
    rootNode.dt=0.01
    rootNode.gravity="0 0 0"

    liverNode = rootNode.addChild("Liver")
    liverNode.addObject(EulerImplicitSolver)
    liverNode.addObject('CGLinearSolver', iterations="200", tolerance="1e-09"... )
    liverNode.addObject('MechanicalObject', template="Vec3d", name="myLiverDOF", position="...")
    liverNode.addObject('MeshMatrixMass', name="mass", totalMass="1" )
    liverNode.addObject('TetrahedronFEMForceField', name="LinearElasticity", youngModulus="1e7")

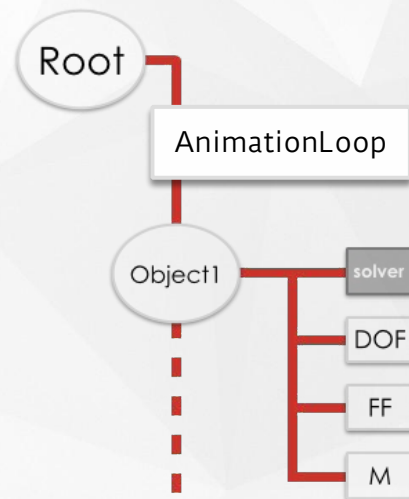
    heartNode = rootNode.addChild("Heart")
    heartNode.addObject(EulerImplicitSolver)
    heartNode.addObject('CGLinearSolver', iterations="200", tolerance="1e-09"... )
    heartNode.addObject('MechanicalObject', template="Vec1d", name="myTemperatureDOF")
    heartNode.addObject('MeshMatrixMass', name="conductivity", totalMass="1" )
    heartNode.addObject('TetrahedronDiffusionFEMForceField', name="DiffusionEffect")
```

Scene Graph



AnimationLoop and Visitor

- Compulsory in a simulation
- Rules and orders all simulation steps
- At each time step, it triggers Visitors to solve the system
- Visitors browse the graph : graph traversal
- Each specific visitor triggers specific functions in components (ex: *accumulateForce* → *addForce()*)



Mapping

- Matching mechanism : correspondence between models
- Each object can have several models in SOFA
 - Physical model
 - Collision detection model
 - Rendering model (or visual model)
- Each model can rely on a different representation/topology

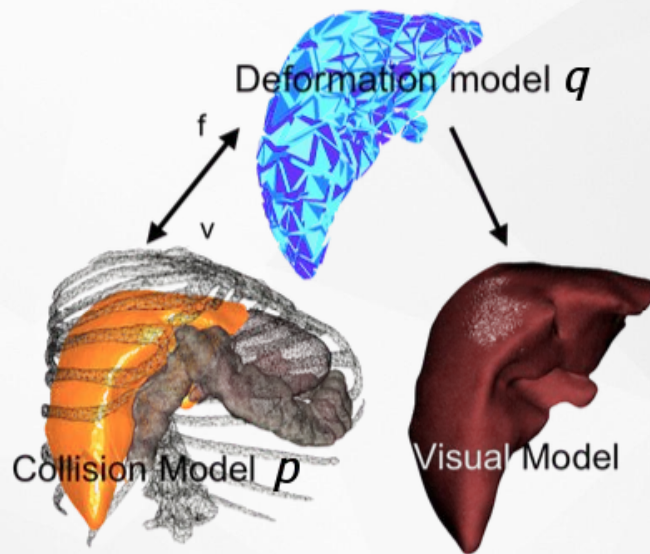
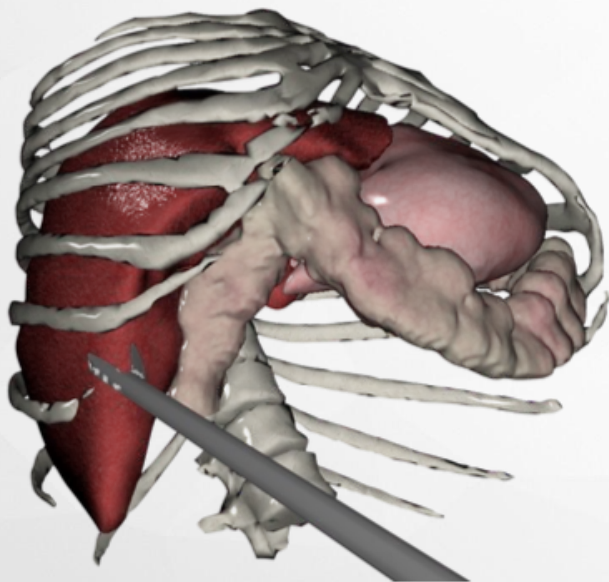
An open-source project

- Arose from research back in 2006 at *Inria*
- Need of a collaborative framework for physics simulation
- Project definition

Being the open-source reference for interactive and real-time applications

Mapping

- Matching mechanism : correspondence between models



Mapping

- Matching mechanism : correspondence between models
- Each object can have several models in SOFA
 - Physical model
 - Collision detection model
 - Rendering model (or visual model)
- Each model can rely on a different representation/topology

Mapping

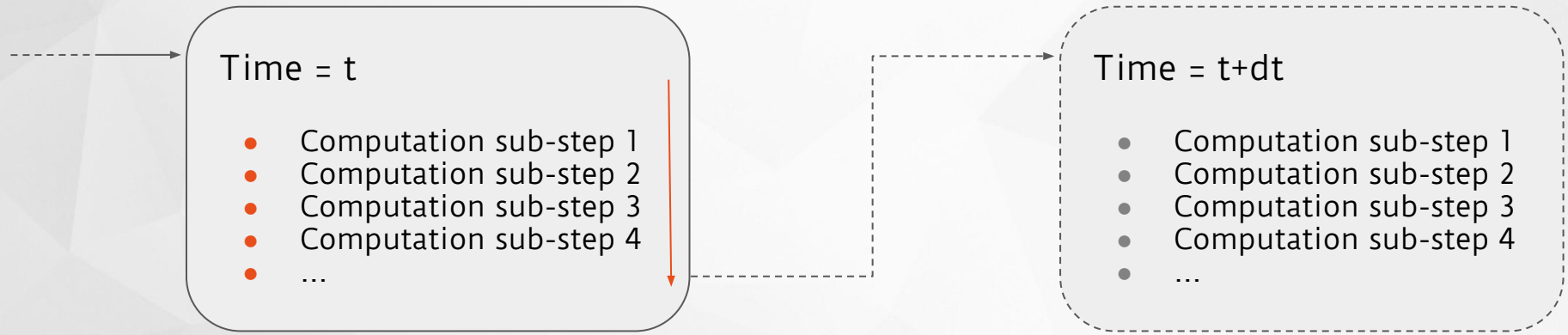


Lifecycle Overview



Lifecycle of one simulation step

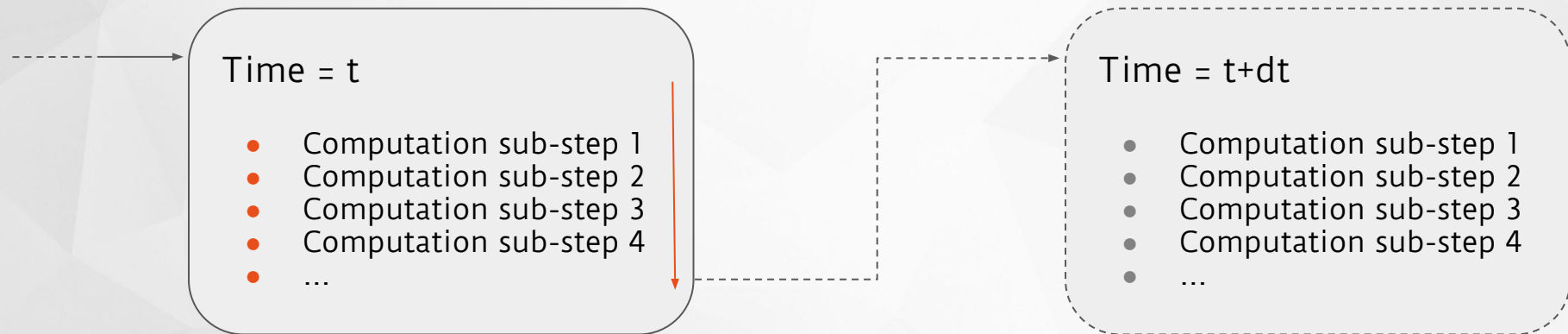
- Which component defines the sub-steps of one time step?



Lifecycle of one simulation step

- Which component defines the sub-steps of one time step?

AnimationLoop



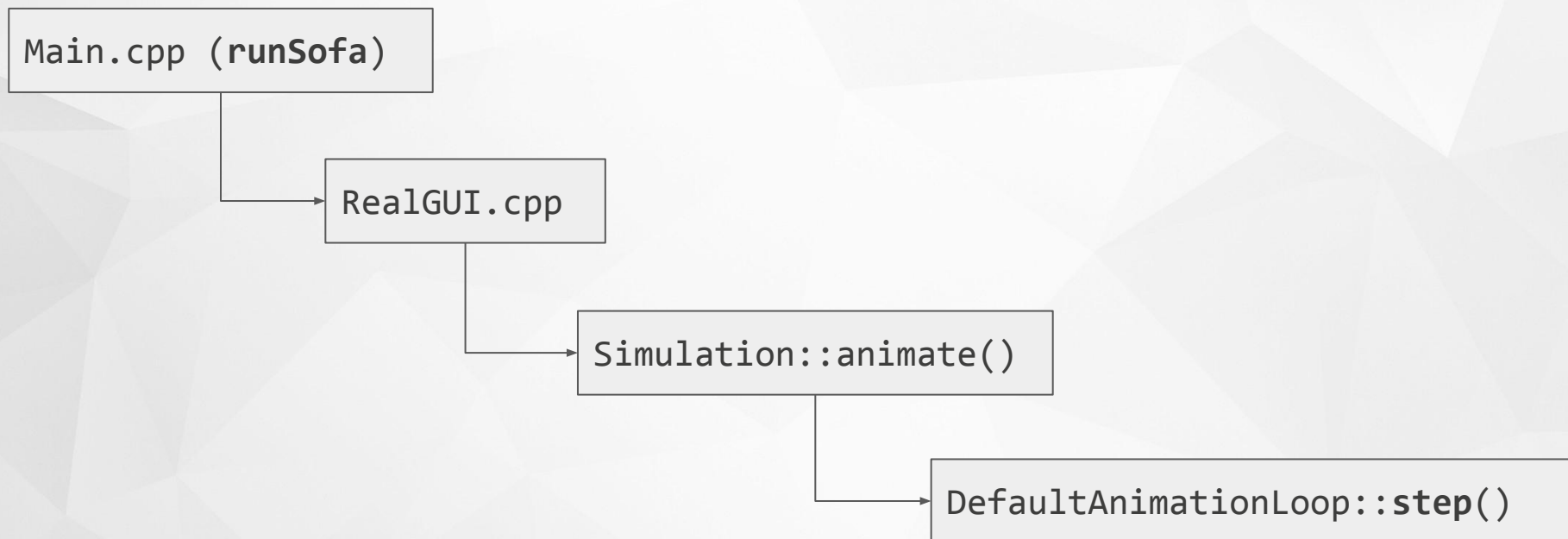
Lifecycle of one simulation step

- Which component defines the sub-steps of one time step?

AnimationLoop

- Two implementations
 - DefaultAnimationLoop
 - FreeMotionAnimationLoop (constraint problems solved using Lagrange multipliers)

Lifecycle: DefaultAnimationLoop



Lifecycle: DefaultAnimationLoop

```
DefaultAnimationLoop::step()
```

```
gnode->execute(AnimateVisitor)
```

```
AnimateVisitor::processNodeTopDown()
```

```
gnode->execute(CollisionVisitor)
```

```
...
```

```
For nb_of_integration_schemes :  
    solver[i]::solve();
```

```
CollisionVisitor::processNodeTopDown()
```

```
doCollisionDetection()  
    intersectionMethod->beginBroadPhase();  
    ...  
    intersectionMethod->beginNarrowPhase();
```

```
doCollisionResponse()  
    Contact->createResponse();
```

```
ODESolver::solve()
```

```
computeForce();           // compute the b vector  
...  
matrix.setSystemMBKMatrix); // define A  
...  
matrix.solve(x,b);        // solve Ax=b
```

Lifecycle: DefaultAnimationLoop

```
DefaultAnimationLoop::step()
```

```
gnode->execute(AnimateVisitor)
```

```
AnimateVisitor::processNodeTopDown()
```

```
gnode->execute(CollisionVisitor)
```

```
...
```

```
For nb_of_integration_schemes :  
    solver[i]::solve();
```

```
CollisionVisitor::processNodeTopDown() 1
```

```
doCollisionDetection()  
    intersectionMethod->beginBroadPhase();  
    ...  
    intersectionMethod->beginNarrowPhase();
```

```
doCollisionResponse()  
    Contact->createResponse();
```

```
ODESolver::solve() 2
```

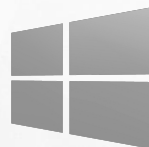
```
computeForce();           // compute the b vector  
...  
matrix.setSystemMBKMatrix); // define A  
...  
matrix.solve(x,b);       // solve Ax=b
```

Let's start with SOFA: User Tutorial



Download SOFA

- Two ways!
 - Download the binaries
 - [Online documentation](#)
- Download + compile the sources
 - Configure with Cmake
 - Build (using make or others)



Versioning of sources

- Relying on Git versioning system
- Hosted on GitHub

<https://github.com/sofa-framework/sofa/>

- Start with GitHub
 - Create your fork
 - For your development project, work in branches
 - Keep your master branch up-to-date regularly

sofa-framework / sofa

Watch 34 Unstar 88 Fork 62

Code Issues 75 Pull requests 17 Boards Reports Projects 8 Insights

Real-time multi-physics simulation with an emphasis on medical simulation. <https://www.sofa-framework.org>

real-time simulation medical physics cpp framework engine research Manage topics

18,171 commits 63 branches 6 releases 57 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

hugtaibot committed on GitHub Merge pull request #358 from fredroyfix_meshconv	Latest commit d5843ca 7 hours ago
.github	[GitHub] EDIT pull request template 3 months ago
SofaAdvanced	[SOFA] Update the year 2017 in all file headers 7 months ago
SofaGeneral	[SOFA] Update the year 2017 in all file headers 7 months ago
SofaGui	[SOFA] Update the year 2017 in all file headers 7 months ago
SofaKernel	[applications] meshconv: notify the user that he need to build minifi... 6 days ago
SofaMisc	[SOFA] Update the year 2017 in all file headers 7 months ago
applications	[applications] meshconv: notify the user that he need to build minifi... 6 days ago
cmakeModules	Split SofaKernel and Sofa(Extended) a year ago
doc	ADD option in EulerImplicit for using trapezoidal scheme 2 years ago
examples	Merge pull request #334 from mimesis-inria/wip_angularSpringForceField 14 days ago
extlibs	[extlibs] REMOVE miniBoost folder 3 months ago
modules	Merge branch 'master' into message_cleaning_week23 13 days ago
scripts	[C] FIX scene-tests crashes due to SofaCUDA loading (2) 13 days ago
share	Cleanup for PR 6 months ago
tools	[a]] Replace std::oerr with calls to msg_* api 5 months ago
.gitattributes	Revert "Compliant: += bug with noalias should be fixed in new eigen lib" 2 years ago
.gitignore	UPDATE: reset .gitignore 4 months ago
Authors.txt	Update licence year and version for future releases 7 months ago
CHANGELOG.md	Update Changelog from 306 to 343 merges 14 days ago

SOFA architecture

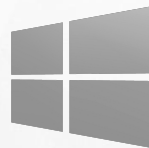
- C++ classes
- Contains all physical models, algorithms and simulation mechanisms
- Structure

- SOFA core modules (abstract classes = API)
- Sofa.Components/ ...
- examples/Components/ ...
- applications/plugins/

Compilation

- Compile SOFA
 - Checkout the sources
 - Configure with Cmake
 - Build (using make, ninja or others)
- [Documentation](#)

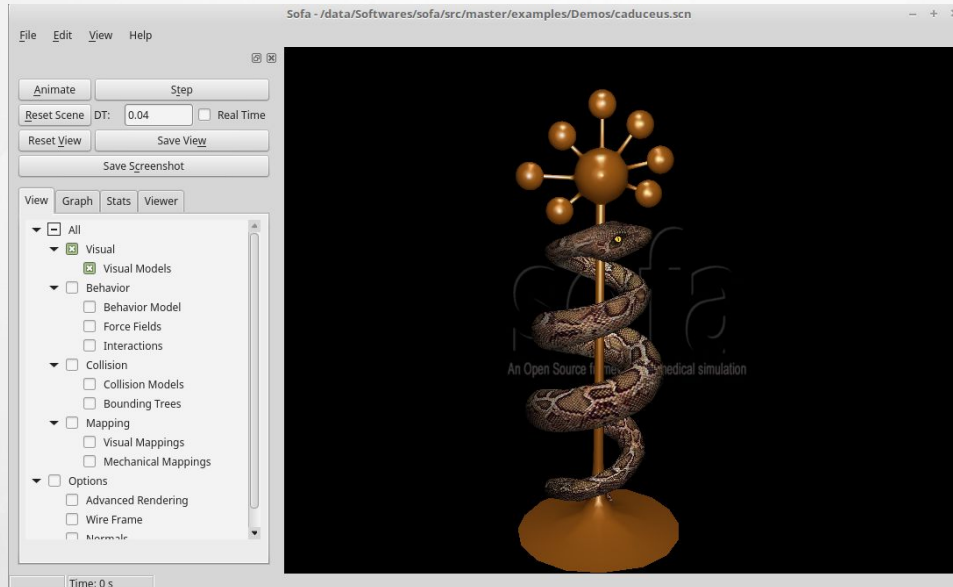
Note that stable binary version of SOFA is available: **v21.12**



sofa-framework.org/download

GUI runSofa

- Build-in Graphical User Interface (GUI) is available
- When running the SOFA executable `./runSofa`

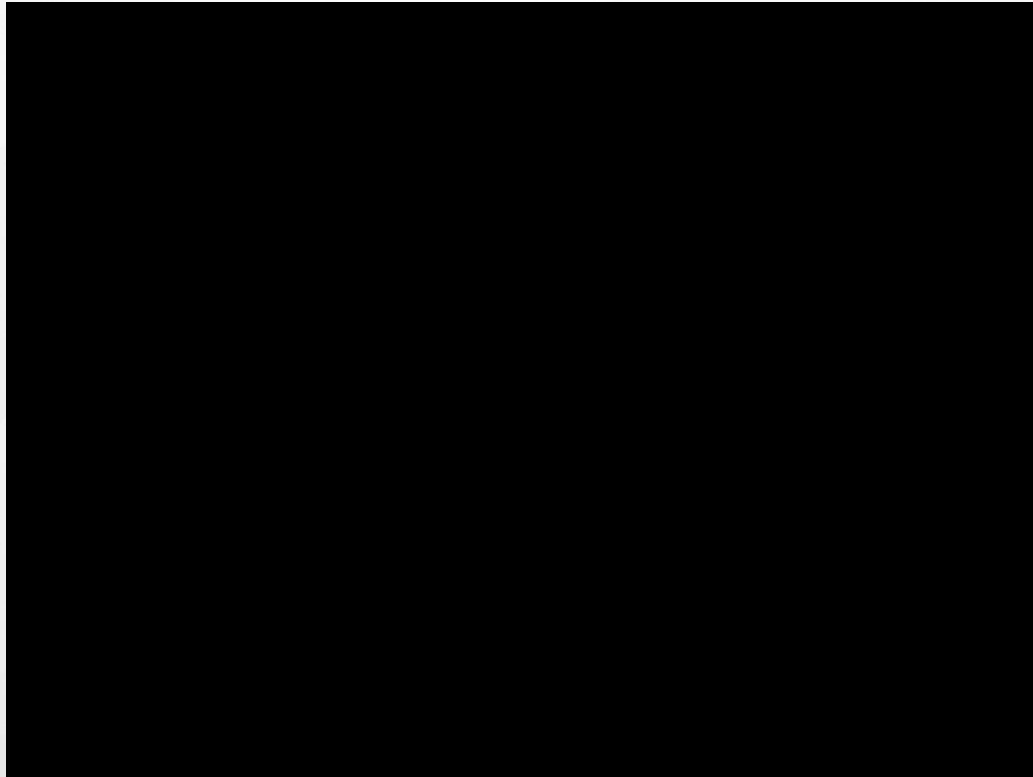


Materials

- Download scenes
(but do not launch the simulation and spoil yourself!)

www.sofa-framework.org/download-materials

Targeted simulation !



Thank you!



- Welcome in the community!
- Time to get started → sofa-framework.org/community/get-involved
- Get support from the community
- Join us at our 2022 events

