Disclaimer

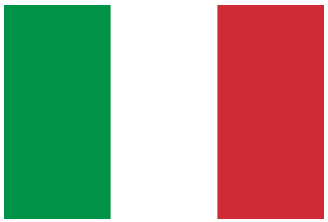
“This offering is not approved or endorsed by OpenCFD Limited, the producer of the OpenFOAM software and owner of the OPENFOAM® and OpenCFD® trade marks.”

Introductory OpenFOAM® Course

From 8th to 12th July, 2013

University of Genoa, DICCA

Dipartimento di **I**ngegneria **C**ivile, **C**himica e **A**mbientale



UNIVERSITÀ DEGLI STUDI
DI GENOVA

Your Lecturer

Joel GUERRERO

joel.guerrero@unige.it

UNIVERSITÀ DEGLI STUDI
DI GENOVA

guerrero@wolfdynamics.com

wolf dynamics

Acknowledgements

These slides and the tutorials presented are based upon personal experience, OpenFOAM® source code, OpenFOAM® user guide, OpenFOAM® programmer's guide, and presentations from previous OpenFOAM® training sessions and OpenFOAM® workshops.

We gratefully acknowledge the following OpenFOAM® users for their consent to use their material:

- Hrvoje Jasak. Wikki Ltd.
- Hakan Nilsson. Department of Applied Mechanics, Chalmers University of Technology.
- Eric Paterson. Applied Research Laboratory Professor of Mechanical Engineering, Pennsylvania State University.

Today's lecture

1. What is OpenFOAM®? – Brief overview
2. OpenFOAM® directory organization
3. Directory structure of an application/utility
4. Applications/utilities in OpenFOAM®
5. Directory structure of an OpenFOAM® case
6. My first OpenFOAM® case setup
7. A deeper view to my first OpenFOAM® case setup
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

Today's lecture

1. **What is OpenFOAM®? – Brief overview**
2. OpenFOAM® directory organization
3. Directory structure of an application/utility
4. Applications/utilities in OpenFOAM®
5. Directory structure of an OpenFOAM® case
6. My first OpenFOAM® case setup
7. A deeper view to my first OpenFOAM® case setup
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

What is OpenFOAM®? – Brief overview

General description:

- OpenFOAM® stands for Open Source Field Operation and Manipulation.
- OpenFOAM® is first and foremost a C++ library used to solve partial differential equations and ordinary differential equations.
- OpenFOAM® comes with several ready-to-use or out-of-the-box numerical solvers, and pre-/post-processing utilities.
- OpenFOAM® is free to use and it has extensive CFD and multi-physics capabilities.
- Can run in parallel computers.
- It is under active development, its capabilities mirror those of commercial CFD applications.
- It counts with a wide-spread community around the world (industry, academia and research labs).

What is OpenFOAM®? – Brief overview

Discretization and implementation – Brief overview:

- Finite Volume Method based solver.
- Collocated polyhedral unstructured meshes.
- Second order accuracy in space and time. Many discretization schemes available.
- Lagragian particle tracking.
- Dynamic mesh handling.
- Adaptive mesh refinement.
- Massive parallelism through domain decomposition.
- All components implemented in library form for easy re-use.

What is OpenFOAM®? – Brief overview

Physical models – Brief overview:

- Physical modeling library: thermo-physical models (liquids and gases), Newtonian and non-Newtonian viscosity models, chemical reactions interface.
- Basic: Laplace, potential flow, passive scalar transport.
- Incompressible and compressible flow: segregated pressure-based algorithms (SIMPLE and PISO).
- Heat transfer: buoyancy-driven flows, conjugate heat transfer.
- Multiphase: Euler-Euler, VOF for free surfaces.
- Turbulence modeling (RANS and LES).
- Pre-mixed and Diesel combustion.
- Stress analysis, fluid-structure interaction, electromagnetics, acoustics, MHD, etc.
- Physics models implementation through **equation mimicking.**

What is OpenFOAM®? – Brief overview

Equation mimicking

This syntax, achieved through the use of object oriented programming features such as inheritance, template classes, virtual functions and operator overloading, enables users to create custom solvers with relative ease. For example the equation,

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot \phi \mathbf{U} - \nabla \cdot \mu \nabla \mathbf{U} = -\nabla p$$

is represented by the code

```
solve
  (
      fvm::ddt(rho, U)
    + fvm::div(phi, U)
    -  fvm::laplacian(mu, U)
       ==
    -  fvc::grad(p)
  );
```

What is OpenFOAM®? – Brief overview

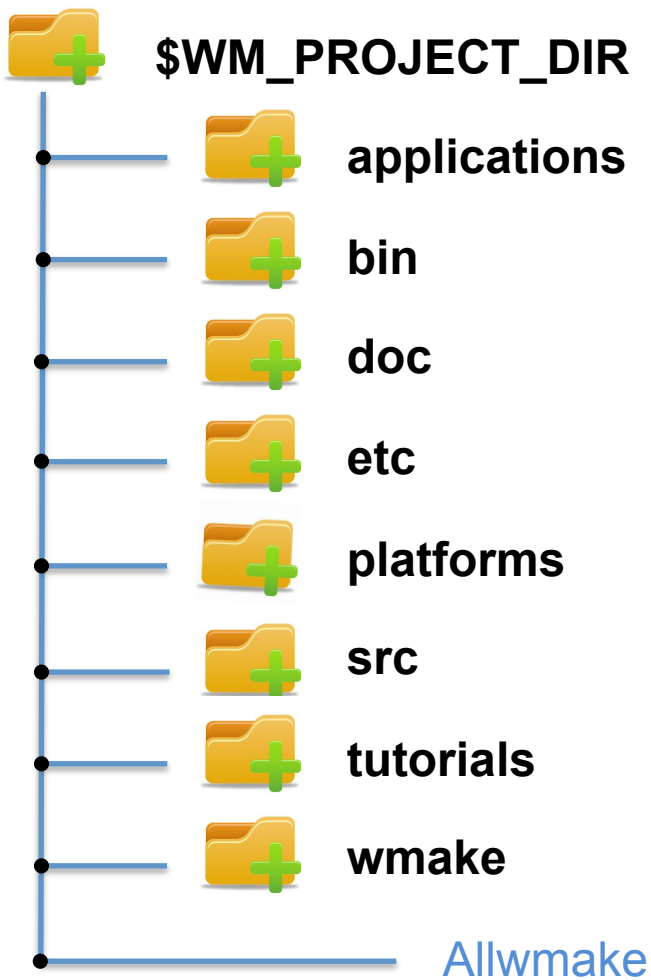
Structure – Brief overview:

- Design encourages code re-use.
- OpenFOAM® is assembled from components:
  - Foundations libraries containing vectors, tensors and field algebra, mesh handling, discretization, boundary conditions, linear solvers, etc.
  - Physical modeling library: thermo-physical models (liquids and gases), viscosity models, chemical reactions interface.
  - Utilities: mesh import and manipulation, paralleling processing, post processing and data manipulation.
  - Custom written top level solvers optimized for efficiency. Few 100s of lines of code.
- Linkage for user extensions and on-the-fly data analysis.
- Model-to-Model interaction handled through common interfaces.

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. **OpenFOAM® directory organization**
3. Directory structure of an application/utility
4. Applications/utilities in OpenFOAM®
5. Directory structure of an OpenFOAM® case
6. My first OpenFOAM® case setup
7. A deeper view to my first OpenFOAM® case setup
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

OpenFOAM® directory organization



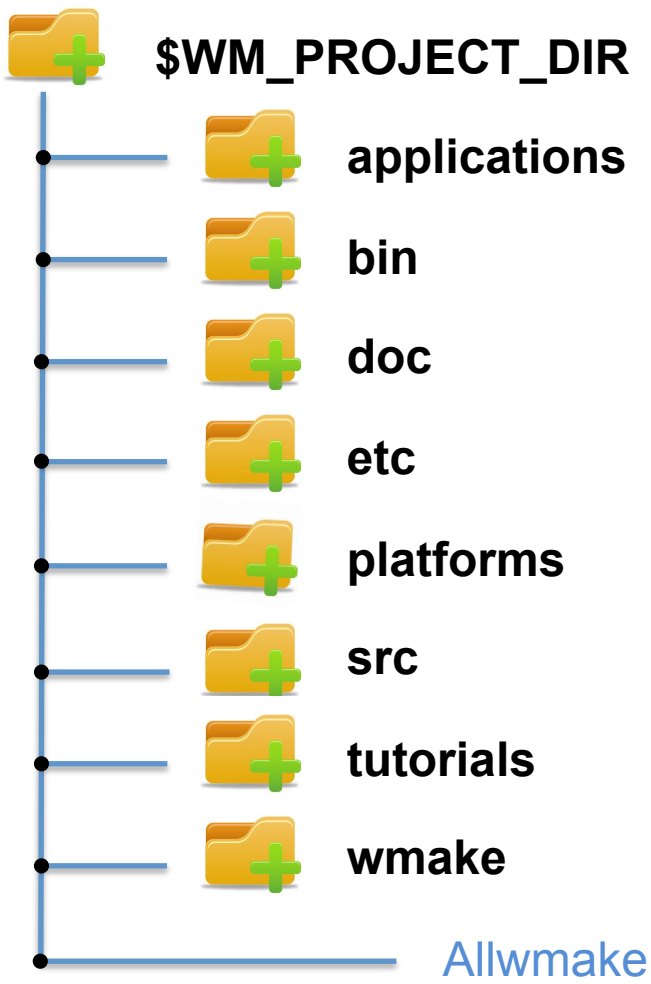
If you installed OpenFOAM® on the default location, the directory **$WM_PROJECT_DIR** should be:

$HOME/OpenFOAM/OpenFOAM-2.2.x

In this directory you will find all the directories containing OpenFOAM® installation. In this directory you will also find additional files (such as README.org, COPYING, etc.), but the most important one is Allwmake, which compiles OpenFOAM®.

Typing wcleanAll will search all the directories below the current one and will delete all the compiled files. So if you type wcleanAll in **$WM_PROJECT_DIR**, it will delete all the compiled files in your OpenFOAM® installation.

OpenFOAM® directory organization



Remember, you can go to any of these directories by using the predefined aliases set by OpenFOAM® environment settings (see **$WM_PROJECT_DIR/etc/config/**alias.sh). For example:

alias foam='cd $WM_PROJECT_DIR'

alias app='cd $FOAM_APP'

alias src='cd $FOAM_SRC'

Type alias for a complete list.

The entries starting with the symbol **$** are environment variables. Find out the value of an environment variable by echoing its value. For example:

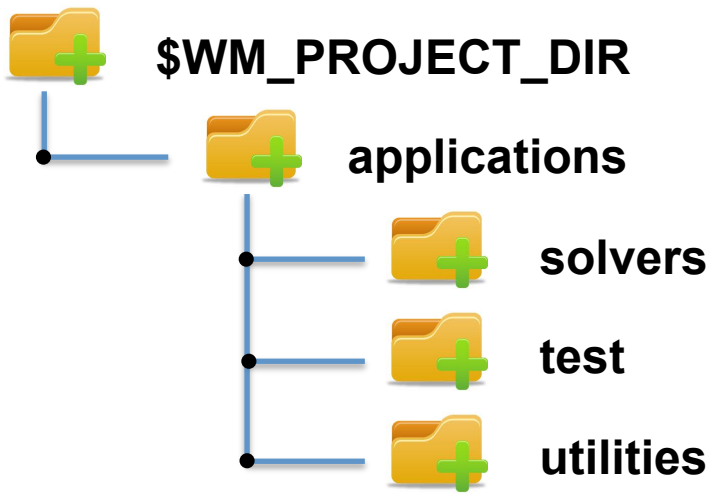
echo **$WM_PROJECT_DIR**

will give you the following output on the terminal

$HOME/OpenFOAM/OpenFOAM-2.2.x

To see all the environment variables type env.

OpenFOAM® directory organization



Let us visit **applications** directory. Type app or cd **$WM_PROJECT_DIR/ applications**. Here you will find the following sub-directories:

- **solvers,** which contains the source code for the distributed solvers.
- **test,** which contains the source code of several test cases that show the usage of some of the OpenFOAM® classes (not available in 1.6-ext).
- **utilities,** which contains the source code for the distributed utilities.

There is also an Allwmake script, which will compile all the content of **solvers** and **utilities**. To compile the test cases in **test** go to the desired test case directory and compile it by typing wmake.

OpenFOAM® directory organization

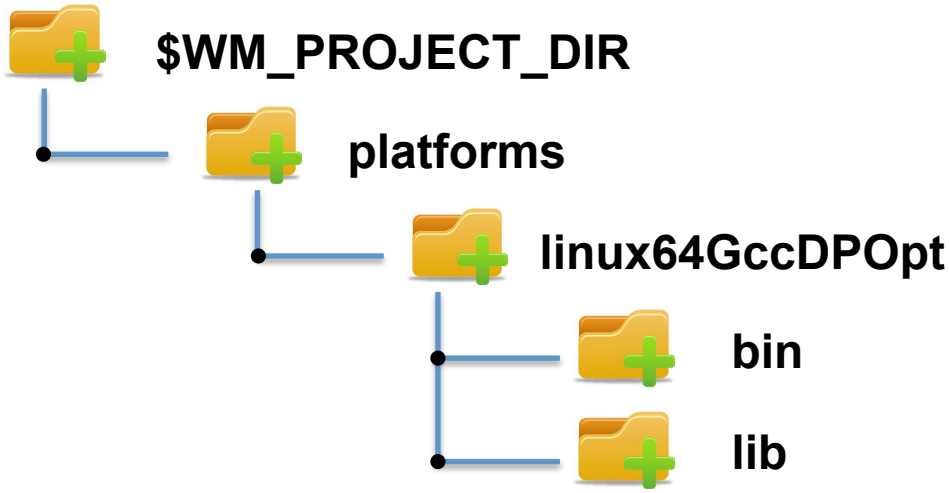
Let us visit **src** directory. Type src or cd **$WM_PROJECT_DIR/src**.

This directory contains the source code for all the libraries. It is divided in different subdirectories and each of them can contain several libraries.

The most relevant are:

- **finiteVolume**. This library provides all the classes needed for the finite volume discretization, such as the fvMesh class, finite volume discretization operators (divergence, laplacian, gradient, fvc/fvm and so on), and boundary conditions (**fields/fvPatchFields**). In **lnInclude** you also find the very important file fvCFD.H, which is included in most applications.
- **OpenFOAM**. This core library includes the definitions of the containers used for the operations, the field definitions, the declaration of the mesh and mesh features such as zones and sets.
- **turbulenceModels**, which contains many turbulence models.

OpenFOAM® directory organization

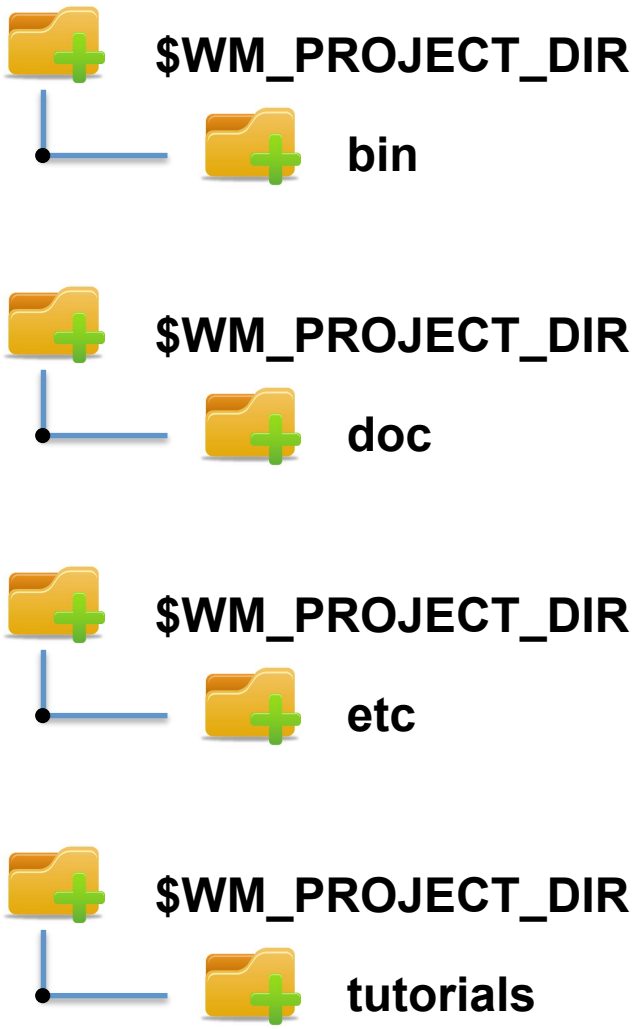


Let us visit **platforms** directory. Type cd **$WM_PROJECT_DIR/platforms**.

This directory contains the binaries generated when compiling the **applications** directory (after compilation the binaries will be located in the directory **$WM_PROJECT_DIR/platforms/linux64GccDPOpt/bin**) and the libraries generated by compiling the source code in the **src** directory (after compilation, the libraries will be located in the directory **$WM_PROJECT_DIR/platforms/linux64GccDPOpt/lib).**

If you installed OpenFOAM® following the default instructions, you compiled the 64 bits version, using the gcc compiler, with double precision, and optimized version. This is reflected in the name of the directory **linux64GccDPOpt**.

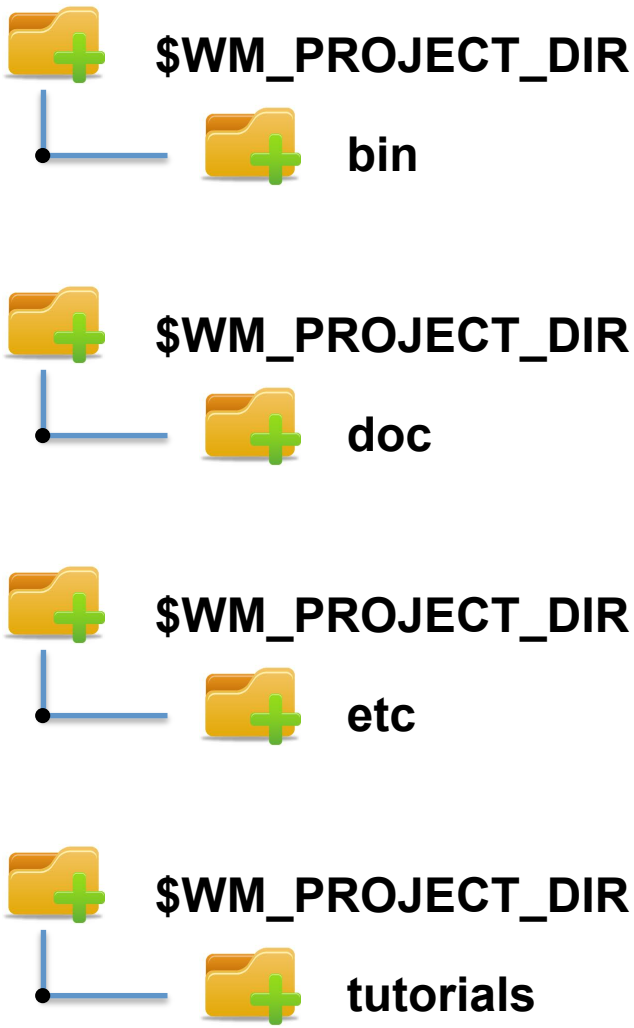
OpenFOAM® directory organization



Let us visit the **bin**, **doc**, **etc**, and **tutorials** directories:

- The **bin** directory contains shell scripts, such as paraFoam, foamNew, foamLog, etc.
- The **etc** directory contains environment setup files, global OpenFOAM® instructions, and default thermoData.
- The **tutorials** directory contains example cases for each solver

OpenFOAM® directory organization



Let us visit the **bin**, **doc**, **etc**, and **tutorials** directories:

- The **doc** directory contains the documentation of OpenFOAM®, namely; user guide, programmer's guide and Doxygen generated documentation in html format (the Doxygen documentation needs to be compiled by typing Allwmake doc in **$WM_PROJECT_DIR**). You can also access Doxygen documentation from internet.
- Documentation usage:

  acroread **$WM_PROJECT_DIR/doc/Guides-a4/**UserGuide.pdf

  acroread **$WM_PROJECT_DIR/doc/Guides-a4/**ProgrammersGuide.pdf

  firefox **file://$WM_PROJECT_DIR/doc/Doxygen/html/**index.html

OpenFOAM® directory organization

Let us visit the **wmake** directory.

OpenFOAM® uses a special make command: wmake.

wmake understands the file structure in OpenFOAM® and has some default compiler directives that are set in the **wmake** directory. There is also a command, wclean, that cleans up the output from the wmake command.

If you add a new compiler name in the bashrc file, you should also tell wmake how to interpret that name. In **wmake/rules** you find the default settings for the available compilers.

You can also find a few scripts that are useful when organizing your files for compilation, or for cleaning up.

OpenFOAM® directory organization

User directory $WM_PROJECT_USER_DIR (USER_NAME-2.2.x)

In **$WM_PROJECT_USER_DIR** directory, all user files are located.

If you are going to develop applications of your own, it is recommended to put the source code in **$WM_PROJECT_USER_DIR** following the same structure as in **$WM_PROJECT_DIR/applications.**

It is recommended to create two more directories:

$WM_PROJECT_USER_DIR/run and **$WM_PROJECT_USER_DIR/src**

Place user developed library source code in **$WM_PROJECT_USER_DIR/src** directory, with the same directory structure as in **$FOAM_SRC**, and case files in the **$WM_PROJECT_USER_DIR/run** directory (which has the alias run).

In **$WM_PROJECT_USER_DIR/platforms/linux64GccDPOpt/bin**, the binaries of the user developed applications will be located; whereas, in **$WM_PROJECT_USER_DIR/platforms/linux64GccDPOpt/lib**, the binaries of the user developed libraries will be located.

This is done so you do not modify anything in the original installation, except for updates!

OpenFOAM® directory organization

Environment variables

- Remember, OpenFOAM® uses its own environment variables.
- OpenFOAM® environment settings are contained in the **OpenFOAM-2.2.x/etc** directory. If you installed OpenFOAM® in the default location, they should be in:
    - **$HOME/OpenFOAM/OpenFOAM-2.2.x/etc**
- If you are running bash or ksh (if in doubt type echo $SHELL), you sourced the **$WM_PROJECT_DIR/etc/**bashrc file by adding the following line to your **$HOME/**.bashrc file:
    - source **$HOME/OpenFOAM/OpenFOAM-2.2.x/etc/bashrc**
- By sourcing the file **$WM_PROJECT_DIR/etc/**bashrc, we start to use OpenFOAM® environment variables, that is to say, OpenFOAM® path to libraries and compilers.
- By default, OpenFOAM® uses its own compiler.

OpenFOAM® directory organization

WARNING

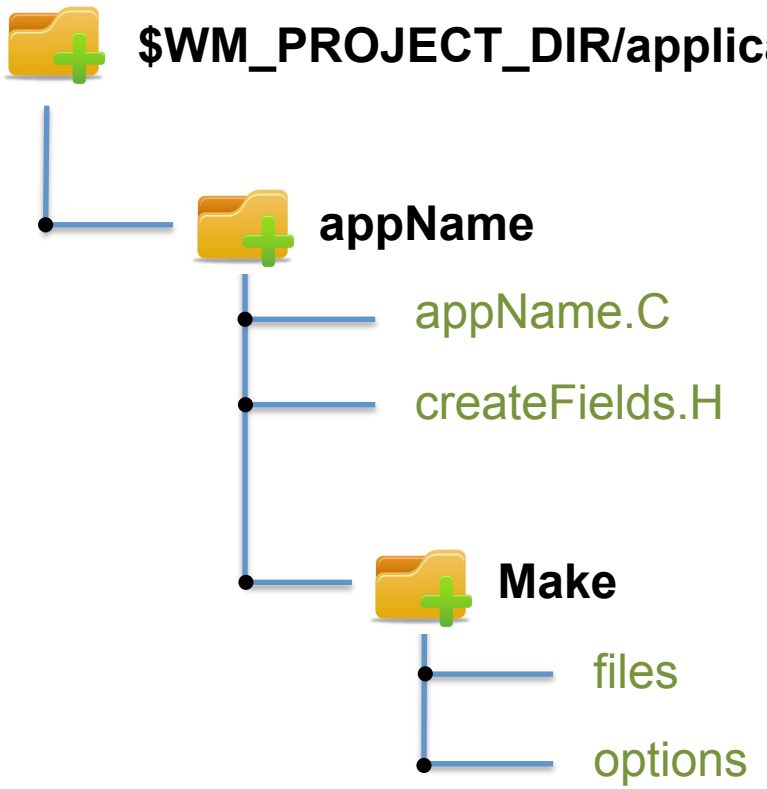
Remember, when you use OpenFOAM® you are using its environment settings, that is, path to libraries and compilers.

So, if you are developing your own software or compiling other applications, do not forget to unload OpenFOAM® environment variables or you will have problems when compiling your own software.

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. **Directory structure of an application/utility**
4. Applications/utilities in OpenFOAM®
5. Directory structure of an OpenFOAM® case
6. My first OpenFOAM® case setup
7. A deeper view to my first OpenFOAM® case setup
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

Directory structure of an OpenFOAM® solver



The **appName** directory contains the source code.

The **Make** directory contains compilation instructions.

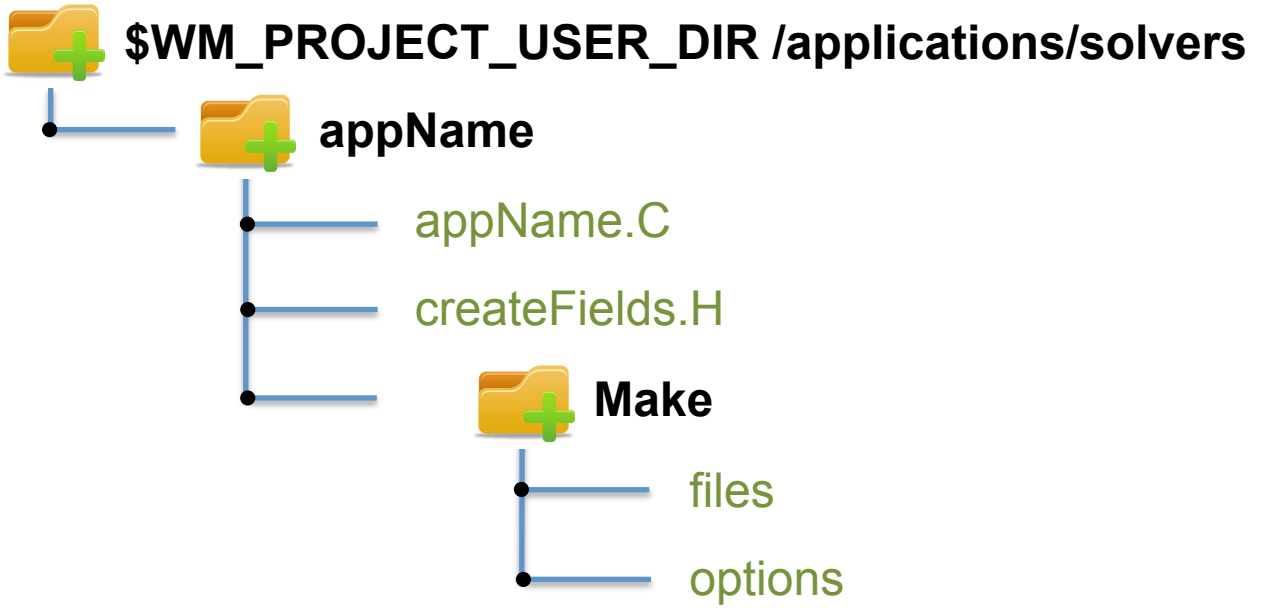
appName/appName.C: is the actual source code.

appName/createFields.H: declares all the field variables and initializes the solution.

Make/files: names all the source files (.C). Specifies the appName name and location of the output file.

Make/options: specifies directories to search for include files and libraries to link the solver against.

Directory structure of an OpenFOAM® solver

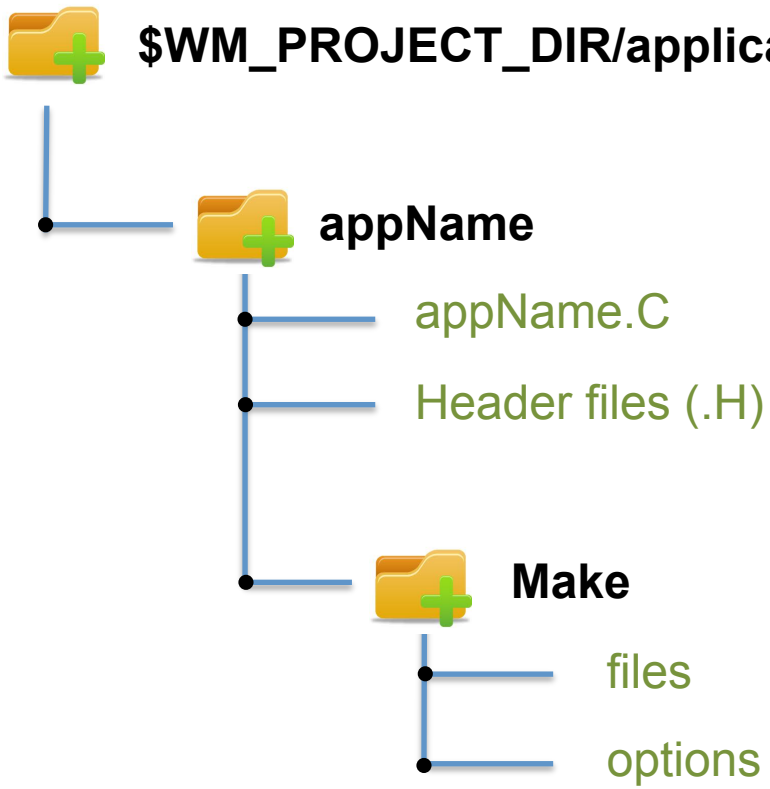


REMEMBER

For your own applications, it is recommended to put the source code in **$WM_PROJECT_USER_DIR** following the same structure as in **$WM_PROJECT_DIR/ applications.** Also, you will need to modify **Make/**files and **Make/**options to show the new name and location of the compiled binaries and libraries to link the solver against.

This is done so you do not modify anything in the original installation, except for updates!. You can do anything you want to your own copies, so you do not risk messing things up.

Directory structure of an OpenFOAM® utility



The **appName** directory contains the source code.

The **Make** directory contains compilation instructions.

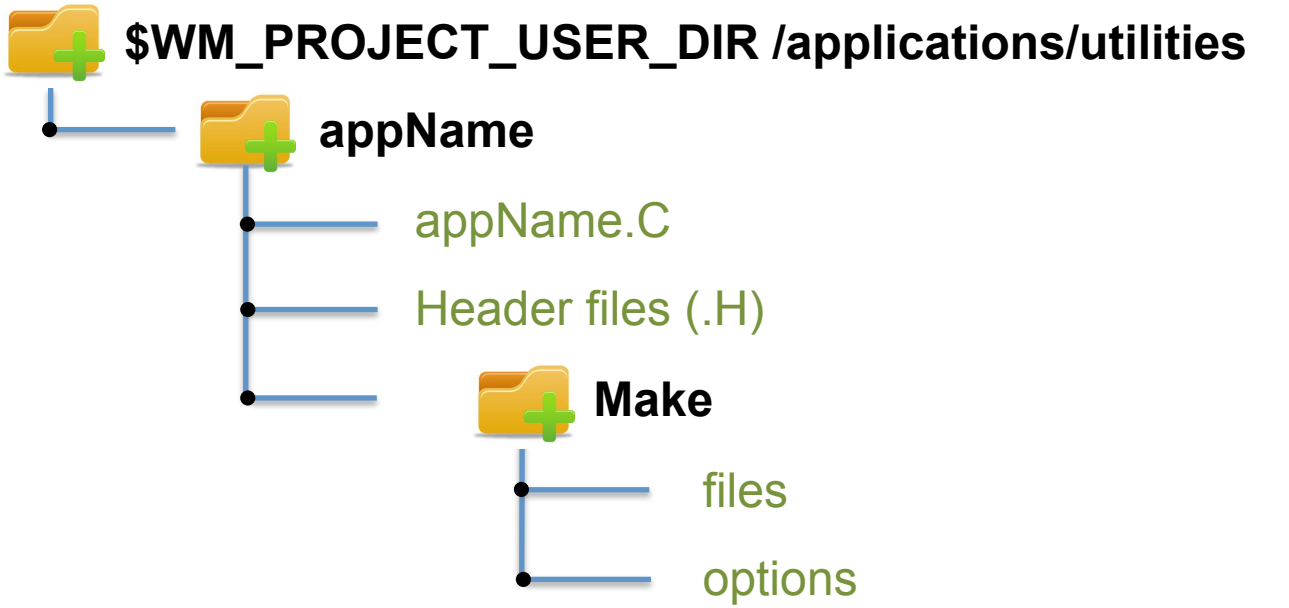
appName/appName.C: is the actual source code.

appName/Header files (.H): header files needed to compile the utility.

Make/files: names all the source files (.C). Specifies the appName name and location of the output file.

Make/options: specifies directories to search for include files and libraries to link the utility against.

Directory structure of an OpenFOAM® utility



REMEMBER

For applications of your own, it is recommended to put the source code in **$WM_PROJECT_USER_DIR** following the same structure as in **$WM_PROJECT_DIR/ applications.** Also, you will need to modify **Make/**files and **Make/**options to show the new name and location of the compiled binaries and libraries to link the utility against.

This is done so you do not modify anything in the original installation, except for updates!. You can do anything you want to your own copies, so you do not risk messing things up.

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. **Applications/utilities in OpenFOAM®**
5. Directory structure of an OpenFOAM® case
6. My first OpenFOAM® case setup
7. A deeper view to my first OpenFOAM® case setup
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

Solvers in OpenFOAM®

- In **$FOAM_SOLVERS** (use alias sol to go there) you will find the directories containing the source code for the solvers available in the OpenFOAM® installation (version 2.2.x):
  - **basic**
  - **combustion**
  - **compressible**
  - **discreteMethods**
  - **DNS**
  - **electromagnetics**
  - **financial**
  - **heatTransfer**
  - **incompressible**
  - **lagrangian**
  - **multiphase**
  - **stressAnalysis**

Solvers in OpenFOAM®

- In sub-directory **incompressible** you will find the solver source code directories:
  - **adjointShapeOptimizationFoam**
  - **boundaryFoam**
  - **icoFoam**
  - **nonNewtonianIcoFoam**
  - **pimpleFoam**
  - **pisoFoam**
  - **potentialFreeSurfaceFoam**
  - **shallowWaterFoam**
  - **simpleFoam**
- Inside each solver directory you will find a *.C file with the same name as the directory. This is the main file, where you will find the top-level source code and a short description of the solver. For **incompressible/icoFoam**/icoFoam.C:

Transient solver for incompressible, laminar flow of Newtonian fluids.

Solvers in OpenFOAM®

REMEMBER

You have the source code right there.

So take your time and dig into each directory to get a complete description of each solver.

Utilities in OpenFOAM®

- In **$FOAM_UTILITIES** (use alias util to go there) you will find the directories containing the source code for the utilities available in the OpenFOAM® installation (version 2.2.x):
    - **mesh**
    - **miscellaneous**
    - **parallelProcessing**
    - **postProcessing**
    - **preProcessing**
    - **surface**
    - **thermophysical**

As for the solver, take your time and dig into each directory to get a complete description of each utility.

Utilities in OpenFOAM®

For example:

- In the sub-directory **mesh/generation** you will find the utilities source code directories:
    - **blockMesh**
    - **extrude**
    - **extrude2DMesh**
    - **snappyHexMesh**
- Inside each utility directory you will find a *.C file with the same name as the directory. This is the main file, where you will find the top-level source code and a short description of the utility. For **snappyHexMesh**/snappyHexMesh.C:

Automatic split hex mesher. Refines and snaps to surface.

Utilities in OpenFOAM®

For example:

- In the sub-directory **postProcessing/velocityField** you will find the utilities source code directories:

  - **Co**
  - **enstrophy**
  - **flowType**
  - **Lambda**
  - **Mach**
  - **Pe**
  - **Q**
  - **streamFunction**
  - **uprime**
  - **vorticity**

- Inside each utility directory you will find a *.C file with the same name as the directory. This is the main file, where you will find the top-level source code and a short description of the utility. For **Q**/Q.C:

Calculates and writes the second invariant of the velocity gradient tensor.

Q = 0.5*(sqr(tr(gradU)) - tr(((gradU)&(gradU)))) [1/s^2]

Utilities in OpenFOAM®

REMEMBER

You have the source code right there.

So take your time and dig into each directory to get a complete description of each solver.

Utilities in OpenFOAM®

Additional solvers and utilities

Visit OpenFOAM® Wiki:

http://www.openfoamwiki.net

CFD-Online OpenFOAM® user group:

http://www.cfd-online.com/Forums/openfoam/

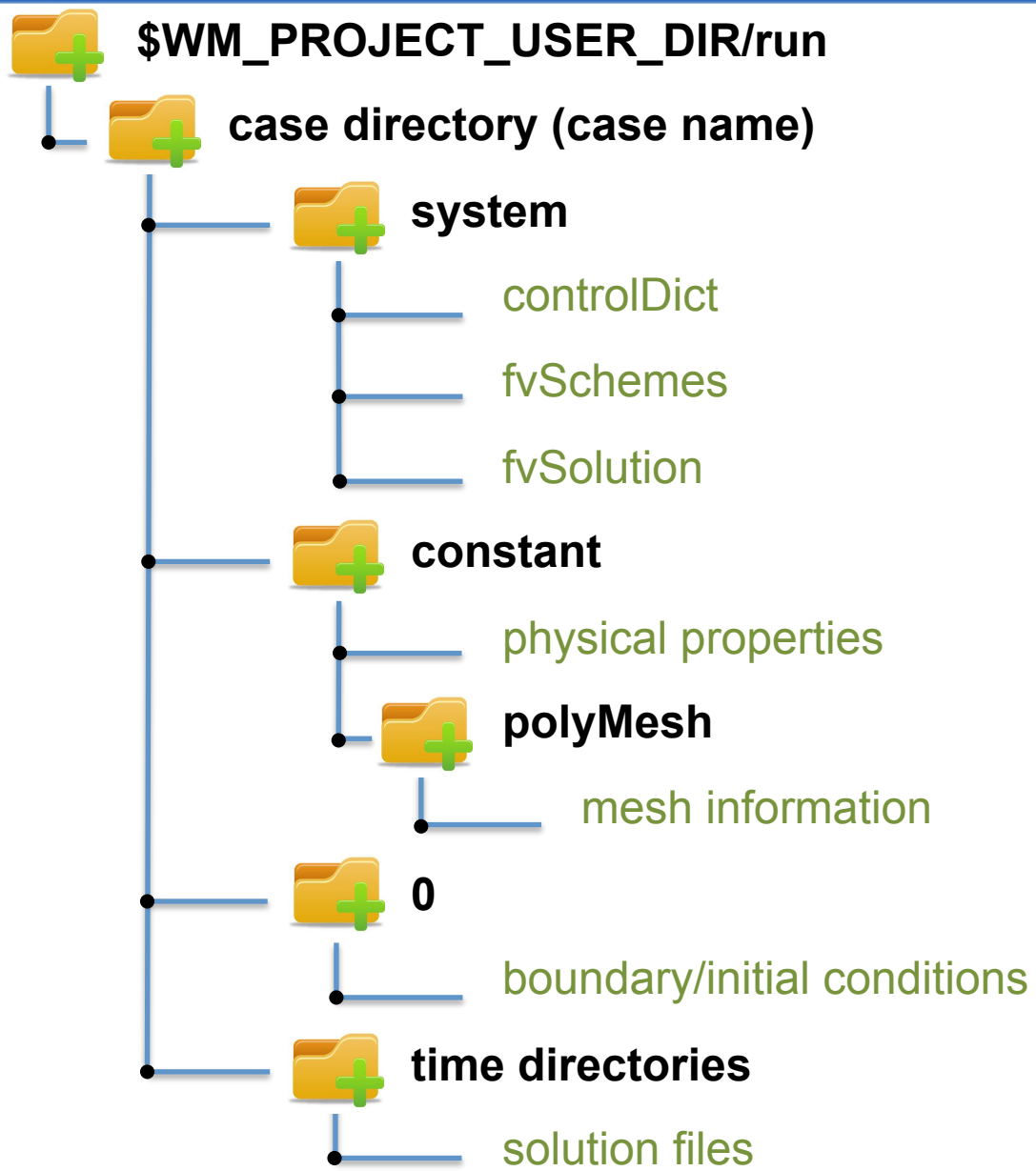
and the OpenFOAM-extend project:

http://www.extend-project.de/

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. ~~Applications/utilities in OpenFOAM®~~
5. **Directory structure of an OpenFOAM® case**
6. My first OpenFOAM® case setup
7. A deeper view to my first OpenFOAM® case setup
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

Directory structure of an OpenFOAM® case



case directory: path to the case, often located in **$WM_PROJECT_USER_DIR/run**

system: contains run-time control and solver numerics.

constant: contains physical properties and turbulence modeling properties.

constant/polyMesh: contains the polyhedral mesh information.

0: contains boundary conditions and initial conditions.

time directories: contains the solution and derived fields.

Utilities in OpenFOAM®

REMEMBER

You always run the applications and utilities in the the **root directory** of your case (**the directory with the case name**).

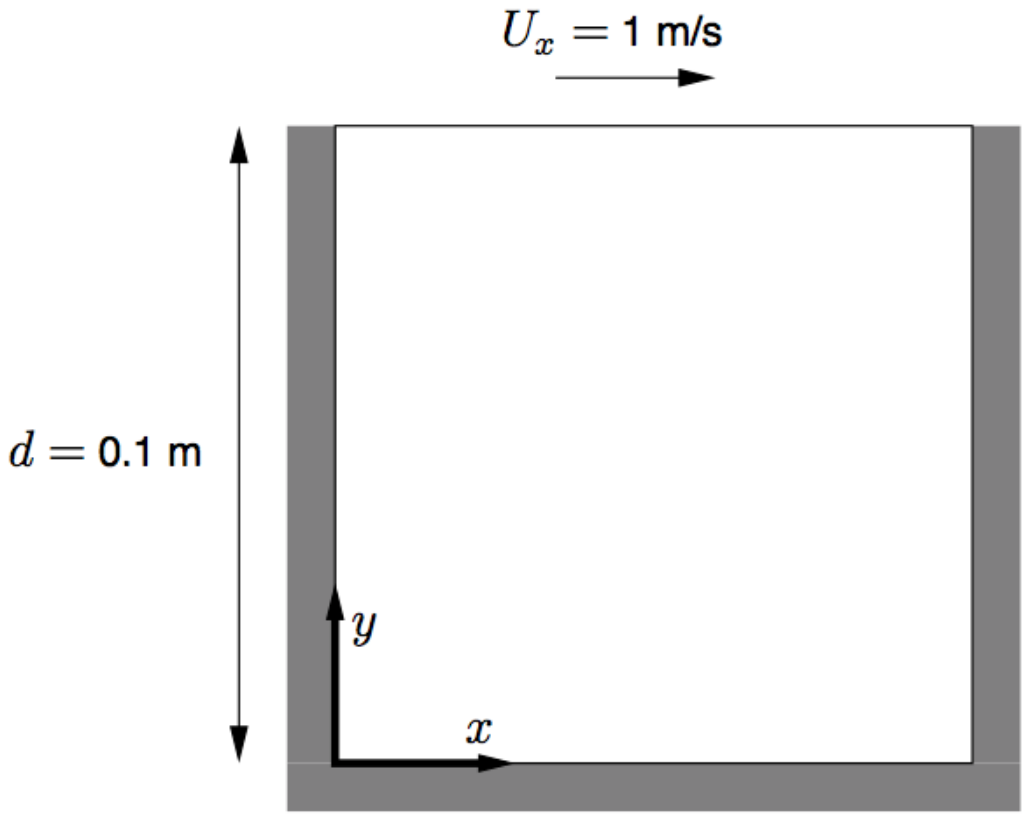
Not in the directory system, not in the directory constant, not in the directory 0.

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. ~~Applications/utilities in OpenFOAM®~~
5. ~~Directory structure of an OpenFOAM® case~~
6. **My first OpenFOAM® case setup**
7. A deeper view to my first OpenFOAM® case setup
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

My first OpenFOAM® case setup

Flow in a lid-driven square cavity



My first OpenFOAM® case setup

We will use the icoFoam cavity tutorial as a general example of how to set up and run applications in OpenFOAM®.

In the folder **$path_to_openfoamcourse/first_tutorial** you will find a copy of the icoFoam cavity tutorial. From this point on, follow me.

In the terminal window type:

- cd **$path_to_openfoamcourse/first_tutorial/cavity**
- blockMesh
- checkMesh
- icoFoam > log.icoFoam

  (this will redirect your standard output to an ascii file with the name log.icoFoam. If you do not add the > log.icoFoam modifier you will see your standard output on the fly and will not be saved)
- foamLog log.icoFoam (if you chose to redirect the standard output)
- paraFoam
- cd **logs**
- gnuplot

(now we use gnuplot to visualize the files in the directory, if you do not know how to use gnuplot follow me.)

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. ~~Applications/utilities in OpenFOAM®~~
5. ~~Directory structure of an OpenFOAM® case~~
6. ~~My first OpenFOAM® case setup~~
7. **A deeper view to my first OpenFOAM® case setup**
8. My second OpenFOAM® case setup
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

A deeper view to my first OpenFOAM® case setup

We will take a look at what we did when running the cavity tutorial by looking at the case files.

- First at all it should be noted that icoFoam is a Transient solver for incompressible, laminar flow of Newtonian fluids.
- The case directory originally contains the following sub-directories: **0**, **constant**, and **system**. After running icoFoam it also contains the time step directories **0.1**, **0.2**, **0.3**, **0.4**, **0.5** and the log.icoFoam file (if you chose to redirect the standard output)
    - The time step directories contain the values of all the variables at those time steps. The **0** directory is thus the initial condition.
    - The **constant** directory contains the mesh and dictionaries for thermophysical and turbulence models.
    - The **system** directory contains settings for the run, discretization schemes and solution procedures.
- The icoFoam solver reads the files in the case directory and runs the case according to those settings.

A deeper view to my first OpenFOAM® case setup

The **constant** directory

(and by the way, open each file and go thru its content)

- The transportProperties file is a dictionary for the dimensioned scalar nu.
- The **polyMesh** directory originally contains the blockMeshDict dictionary for the blockMesh mesh generator. After generating the mesh, it will contain the mesh in OpenFOAM® format.
- Depending of your physical model, you will find more dictionaries in the constant directory. For example, if you need to set gravity, you will need to create the dictionary g.
- We will now take a quick look at the blockMeshDict dictionary in order to understand what we have done (and yes, this is part of the meshing session as well).
- Go to the directory **constant/polyMesh** and open blockMeshDict dictionary with your favorite text editor.

A deeper view to my first OpenFOAM® case setup

The blockMeshDict dictionary

The blockMeshDict dictionary first of all contains a list with a number of vertices:

```
convertToMeters 0.1;
vertices
(
        (0 0 0)
        (1 0 0)
        (1 1 0)
        (0 1 0)
        (0 0 0.1)
        (1 0 0.1)
        (1 1 0.1)
        (0 1 0.1)
);
```

- There are eight vertices defining a 3D block. OpenFOAM® always uses 3D meshes, even if the simulation is 2D.
- convertToMeters 0.1; multiplies the coordinates by 0.1.

Refer to the User Guide for more Information

A deeper view to my first OpenFOAM® case setup

The blockMeshDict dictionary

The blockMeshDict dictionary then defines a block and the mesh from the vertices:

```
blocks
(
    hex (0 1 2 3 4 5 6 7) (20 20 1) simpleGrading (1 1 1)
);
```

- hex means that it is a structured hexahedral block.
- (0 1 2 3 4 5 6 7) are the vertices used to define the block (**and yes, the order is important**). Each hex block is defined by eight vertices, in sequential order. Where the first vertex in the list represents the origin of the coordinate system.
- (20 20 1) is the number of mesh cells in each direction.
- simpleGrading (1 1 1) is the expansion ratio, in this case equidistant.

Refer to the User Guide for more Information

A deeper view to my first OpenFOAM® case setup

The blockMeshDict dictionary

The blockMeshDict dictionary also defines three boundary patches:

```
boundary
(
    movingWall                  //NAME
    {
        type wall;              //TYPE
        faces
        (
            (3 7 6 2)           //FACE
        ) ;
    }
```

```
    fixedWalls                  //NAME
    {
        type wall;              //TYPE
        faces
        (
            (0 4 7 3)           //FACE
            (2 6 5 1)           //FACE
            (1 5 4 0)           //FACE
        );
    }

    frontAndBack                //NAME
    {
        type empty;             //TYPE
        faces
        (
            (0 3 2 1)           //FACE
            (4 5 6 7)           //FACE
        );
    }
);
```

A deeper view to my first OpenFOAM® case setup

The blockMeshDict dictionary

Each boundary patch in the blockMeshDict dictionary defines a type, a name, and a list of boundary faces:

```
fixedWalls
{
    type walls;
    faces
    (
        (0 4 7 3)
        (2 6 5 1)
        (1 5 4 0)
    );
}
```

- wall is the type of the boundary. fixedWalls is the name of the patch.
- The patch is defined by three faces of the block according to the list, which refers to the vertex numbers.

Refer to the User Guide for more Information

A deeper view to my first OpenFOAM® case setup

The blockMeshDict dictionary

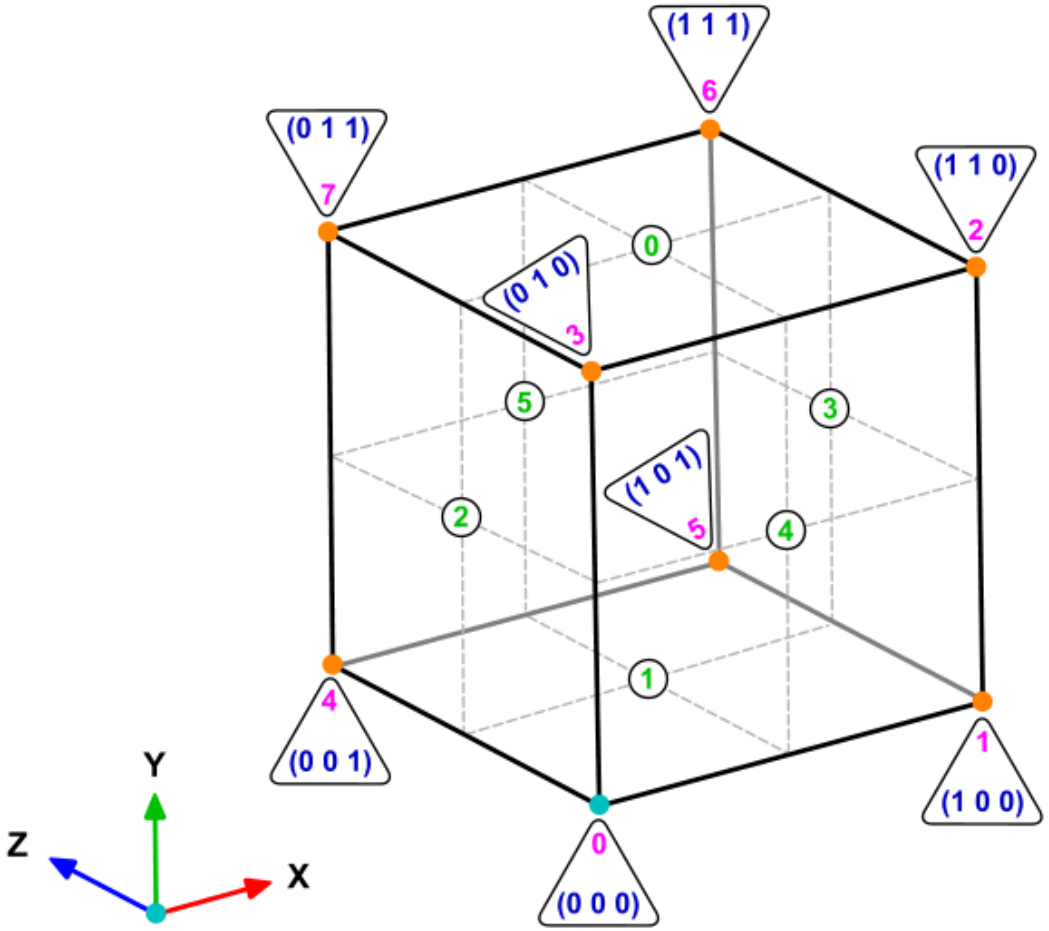
- When defining the block and boundaries, the vertices in the lists must be neighbors.
- The order of the vertices in the block must be such that if I sit down in the origin of coordinates (vertex 0 for this case), and I look in the direction z; the vertices are marched in a counter clockwise sense.
- To sum up, the blockMeshDict dictionary generates a block with:
    - x/y/z dimensions 0.1/0.1/0.01.
    - 20 x 20 x 1 cells.
    - wall fixedWalls patch at three sides.
    - wall movingWall patch at one side.
    - empty frontAndBack patch at two sides.
- The type empty tells OpenFOAM® that this is a 2D case.

Refer to the User Guide for more Information

A deeper view to my first OpenFOAM® case setup

The blockMeshDict dictionary

- This is how the block created with blockMeshDict looks like,



VERTICES

0	0 0 0
1	1 0 0
2	1 1 0
3	0 1 0
4	0 0 1
5	1 0 1
6	1 1 1
7	0 1 1

BLOCK (HEX)

0 1 2 3 4 5 6 7

FACES

0	3 7 6 2
1	1 5 4 0
2	0 4 7 3
3	2 6 5 1
4	0 3 2 1
5	4 5 6 7

Refer to the User Guide for more Information

A deeper view to my first OpenFOAM® case setup

The **system** directory

(and by the way, open each file and go thru its content)

- The system directory consists of three dictionary files:
  - controlDict
  - fvSchemes
  - fvSolution
- controlDict contains general instructions on how to run the case.
- fvSchemes contains instructions for the discretization schemes that will be used for the different terms in the equations.
- fvSolution contains instructions on how to solve each discretized linear equation system.

Refer to the User Guide for more Information

A deeper view to my first OpenFOAM® case setup

Dictionary files hint

If you do not know which entries are available for a specific keyword in a dictionary, just use a dummy word (I love to add bananas) and the solver will list the alternatives, for instance if we add,

stopAt banana;

in the controlDict dictionary file; the solver will give you the following message:

```
banana is not in enumeration
4
(
    nextWrite
    writeNow
    noWriteNow
    endTime
)
```

From now on, let us call this shortcut the banana method (attention, it is not related to Rosenbrock's banana function).

A deeper view to my first OpenFOAM® case setup

Dictionary files advanced features

- C++ commenting:

```
// This is my comment

/* My comments, line 1
   My comments, line 2 */
```

- #include directive:

```
#include "initialConditions"
```

Do not forget to create the respective include file initialConditions.

A deeper view to my first OpenFOAM® case setup

Dictionary files advanced features

- Macro expansion:

```
flowVelocity (20 0 0);

        /*
            some code lines here
        */

internalField           uniform $flowVelocity;

        /*
            some code lines here
        */
```

A deeper view to my first OpenFOAM® case setup

Dictionary files advanced features

Instead of writing (the poor man's way):

```
leftWall
{
    type    fixedValue;
    value   uniform (0 0 0);
}
rightWall
{
    type    fixedValue;
    value   uniform (0 0 0);
}
topWall
{
    type    fixedValue;
    value   uniform (0 0 0);
}
```

A deeper view to my first OpenFOAM® case setup

Dictionary files advanced features

You can write:

```
"(left|right|top)Wall"
{
    type        fixedValue;
    value       uniform (0 0 0);
}
```

or you could also try

```
".*Wall"
{
    type        fixedValue;
    value       uniform (0 0 0);
}
```

A deeper view to my first OpenFOAM® case setup

The **0** directory

(and by the way, open each file and go thru its contents)

The **0** directory contains the dimensions, and the initial and boundary conditions for all primary variables, in this case p and U. For the cavity case, the U file contains:

```
dimensions                [0 1 -1 0 0 0 0];
internalField uniform     (0 0 0);
boundaryField
{
    movingWall
    {
        type fixedValue;
        value uniform    (1 0 0);
    }
    fixedWalls
    {
        type fixedValue;
        valueuniform     (0 0 0);
    }
    frontAndBack
    {
        type  empty;
    }
}
```

A deeper view to my first OpenFOAM® case setup

The **0** directory

(and by the way, open each file and go thru its contents)

- dimensions [0 1 -1 0 0 0 0]; states that the dimension of U is m/s.
- internalField uniform (0 0 0); sets U to zero internally (initial conditions).
- The boundary patches movingWall and fixedWalls are given the type fixedValue; with value uniform (1 0 0), and (0 0 0) respectively (boundary conditions).
- The frontAndBack patch is given type empty, indicating that no solution is required in that direction since the case is 2D.
- You should now be able to understand the file **0/**p.
- The time step directories are similar but the internalField now contains the solution. There is also a phi file, containing the resulting face fluxes that are needed to yield a perfect restart.

Refer to the User Guide for more Information

A deeper view to my first OpenFOAM® case setup

At this point you should have realized a few things:

- OpenFOAM® is fully dimensional. You need to define the dimensions for each field data and physical properties. Your dimensions shall be consistent.

A deeper view to my first OpenFOAM® case setup

Dimensions in OpenFOAM®

No.	Property	Unit	Symbol
1	Mass	Kilogram	**kg**
2	Length	meters	**m**
3	Time	second	**s**
4	Temperature	Kelvin	**K**
5	Quantity	moles	**mol**
6	Current	ampere	**A**
7	Luminuous intensity	candela	**cd**

dimensions [kg, m, s, K, mol, A, cd]

A deeper view to my first OpenFOAM® case setup

At this point you should have realized a few things:

- Each file in the case directory has a header, you should always keep this header, if not, OpenFOAM® will complain.
- In the header of your field variables, the class type should be consistent with the type of field variable you are using. That is to say, if the field variable is a vector, the class should be volVectorField and if the field variable is a scalar, the class should be volScalarField.

A deeper view to my first OpenFOAM® case setup

At this point you should have realized a few things:

- If you misspell something or made a mistake, OpenFOAM® will complain and will tell you where is and what is the error.
- I continuously ask you to refer to the User Guide for more information. Remember, you have the basic information there. So, take your time and go throughout the User Guide before you start complaining and hitting the keyboard.

A deeper view to my first OpenFOAM® case setup

The log.icoFoam file

- If you followed the previous instructions you should now have a log.icoFoam file. This file contains all the residuals information.
- It is of interest to have a graphical representation of the residual. To do so, the foamLog utility is used.
- Let us now execute foamLog utility. In the terminal type:
  - foamLog log.icoFoam
- A directory **logs** has now been generated, with extracted values in ascii format in two columns. The first column is the time, and the second column is the value at that time.
- Type foamLog -help for more information. Add this point, you can plot the residuals using octave, gnuplot or xmgrace.

A deeper view to my first OpenFOAM® case setup

Now you should be ready to go and explore the applications and dictionaries by yourself!

So go nuts, and do as many changes as you like to the case dictionaries and study the output.

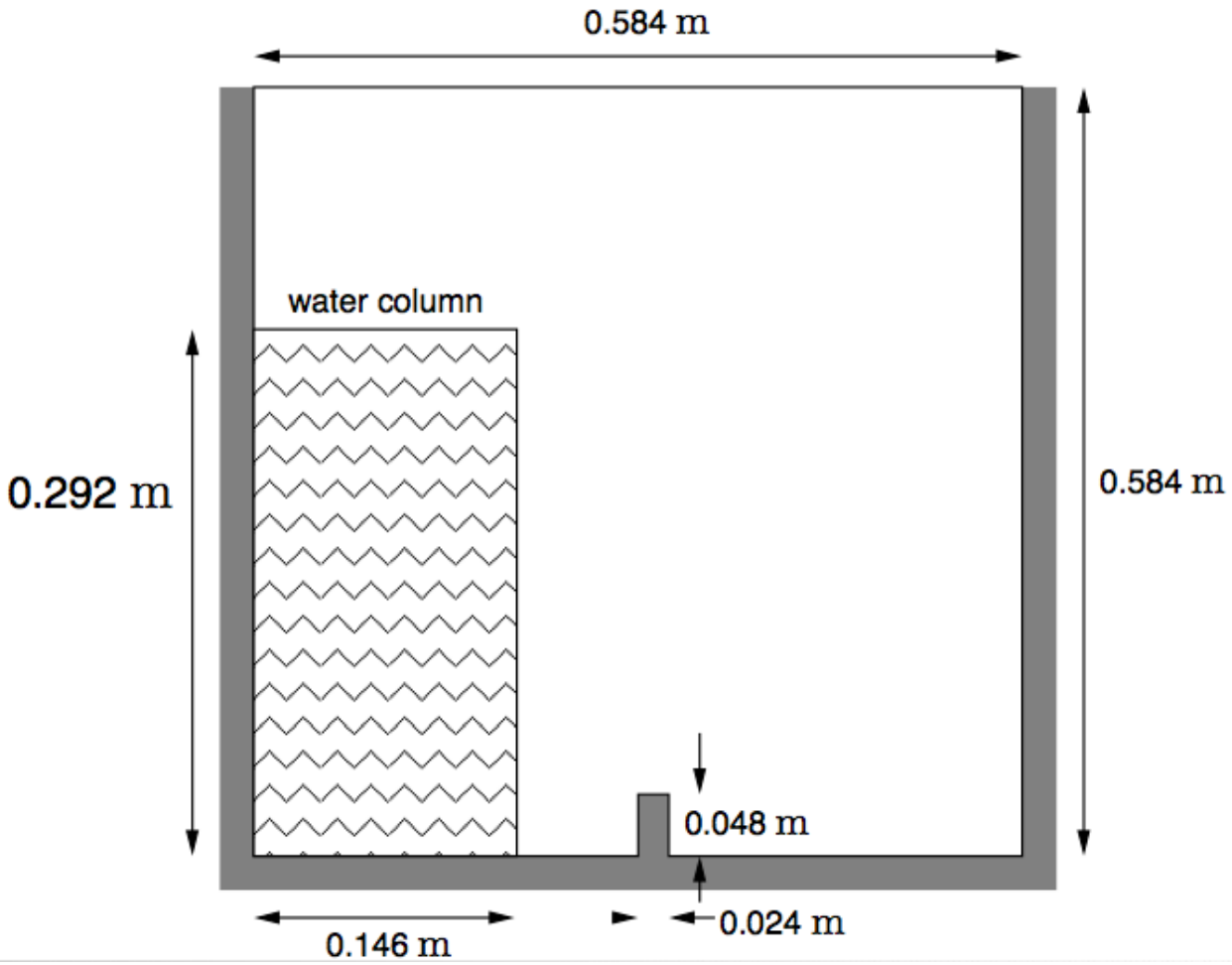
I would start by changing the mesh cell number.

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. ~~Applications/utilities in OpenFOAM®~~
5. ~~Directory structure of an OpenFOAM® case~~
6. ~~My first OpenFOAM® case setup~~
7. ~~A deeper view to my first OpenFOAM® case setup~~
8. **My second OpenFOAM® case setup**
9. My third OpenFOAM® case setup
10. My first 3D OpenFOAM® case setup
11. Hands-on session

My second OpenFOAM® case setup

Damp break free surface flow



My second OpenFOAM® case setup

We will use the interFoam damp break tutorial for our second OpenFOAM® case setup.

In the folder **$path_to_openfoamcourse/first_tutorial** you will find a copy of the interFoam damp break tutorial. From this point on, follow me.

In the terminal window type:

- cd **$path_to_openfoamcourse/first_tutorial/dampBreak**
- blockMesh
- checkMesh
- cp **0/**alpha1.org **0/**alpha1

  (this is done in order to keep a copy of the original alpha1 boundary and initials conditions. Why?, in the next step we are going to use the utility setFields, which overwrites the file alpha1)

- setFields (this utility uses a dictionary, which is located in **system**)

My second OpenFOAM® case setup

We will use the interFoam damp break tutorial for our second OpenFOAM® case setup.

In the folder **$path_to_openfoamcourse/first_tutorial** you will find a copy of the interFoam damp break tutorial. From this point on, follow me.

In the terminal window type:

- interFoam > log.interFoam

  (this will redirect your standard output to an ascii file with the name log.interFoam. If you do not add the > log.interFoam modifier you will see your standard output on the fly and will not be saved)

- foamLog log.interFoam (if you chose to redirect the standard output)
- paraFoam
- cd **logs**
- gnuplot

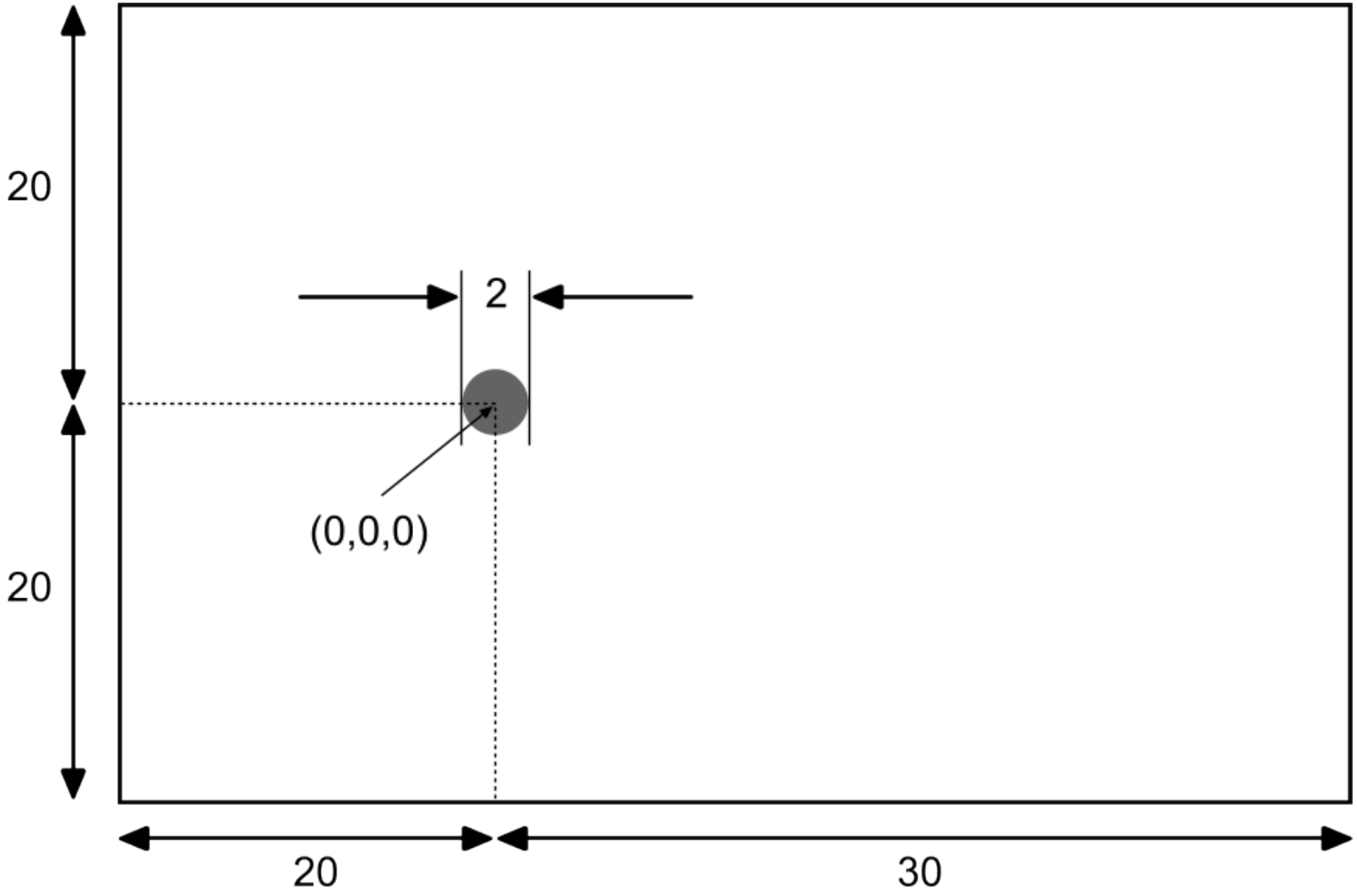
  (now we use gnuplot to visualize the files in the directory, if you do not know how to use gnuplot follow me.)

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. ~~Applications/utilities in OpenFOAM®~~
5. ~~Directory structure of an OpenFOAM® case~~
6. ~~My first OpenFOAM® case setup~~
7. ~~A deeper view to my first OpenFOAM® case setup~~
8. ~~My second OpenFOAM® case setup~~
9. **My third OpenFOAM® case setup**
10. My first 3D OpenFOAM® case setup
11. Hands-on session

My third OpenFOAM® case setup

Flow around a cylinder



All the dimensions are in meters

My third OpenFOAM® case setup

In this tutorial, we are going to learn how to setup the solvers potentialFoam, simpleFoam, pisoFoam and pimpleFoam. We are also going to learn how to setup a turbulent case and how to use a few utilities.

In the folder **$path_to_openfoamcourse/first_tutorial** you will find a copy of this tutorial. In the terminal window type:

- cd **$path_to_openfoamcourse/first_tutorial/vortex_shedding**

In this directory, you will find ten folders, each one representing a different case setup. From this point on, follow me.

Let us go first to the folder **c4,**

- cd **c4**
- fluentMeshToFoam ../mesh/ascii.msh
  (the mesh is in the directory **../mesh**)
- checkMesh
- icoFoam
- paraFoam

My third OpenFOAM® case setup

Now try to do the rest of the cases. Remember, each folder corresponds to a different solver or setup, so try to figure out what solver and setup we are using. Also try to use the following utilities:

- wallShearStress
- stressComponents
- wallGradU
- Q
- Co
- Pe
- vorticity
- yPlusRAS
- refineWallLayer
- refineHexMesh
- renumberMesh

If you want more information about an specific utility try

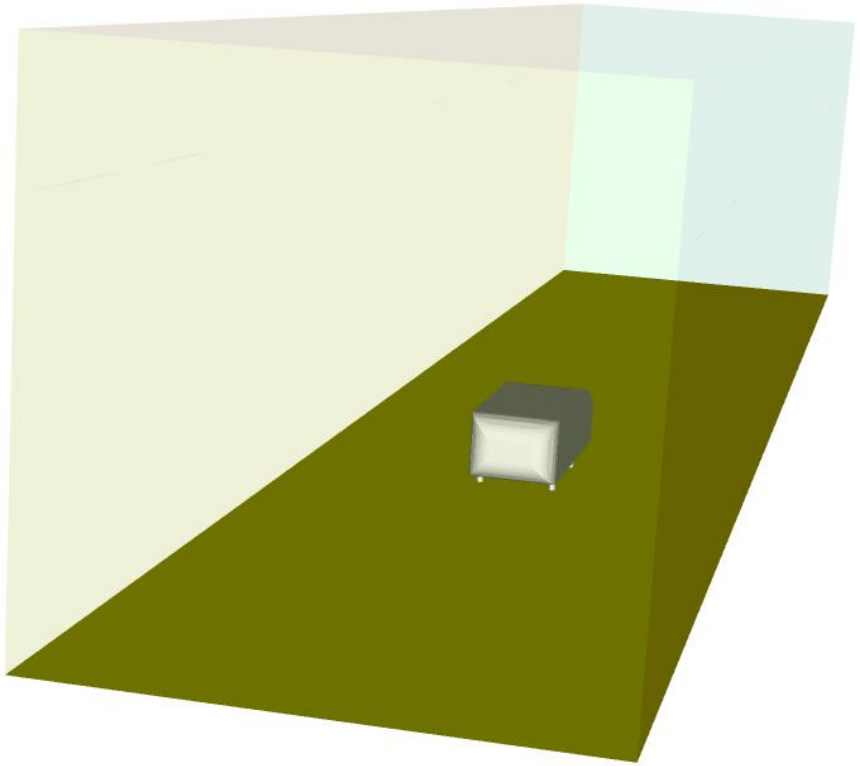
- utility_name -help

Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. ~~Applications/utilities in OpenFOAM®~~
5. ~~Directory structure of an OpenFOAM® case~~
6. ~~My first OpenFOAM® case setup~~
7. ~~A deeper view to my first OpenFOAM® case setup~~
8. ~~My second OpenFOAM® case setup~~
9. ~~My third OpenFOAM® case setup~~
10. **My first 3D OpenFOAM® case setup**
11. Hands-on session

My first 3D OpenFOAM® case setup

Flow about ahmed body



My first 3D OpenFOAM® case setup

In this tutorial, we use pimpleFoam to solve a 3D case. Running a 3D simulation is not different from the previous 2D simulations. The only difference is that we need to define the boundary conditions in the third dimension.

In the folder **$path_to_openfoamcourse/first_tutorial** you will find a copy of this tutorial. In the terminal window type:

- cd **$path_to_openfoamcourse/first_tutorial/ahmed_body**
- blockMesh
- surfaceFeatureExtract
- snappyHexMesh -overwrite
- checkMesh
- pimpleFoam
- paraFoam

My first 3D OpenFOAM® case setup

In this tutorial, we use pimpleFoam to solve a 3D case. Running a 3D simulation is not different from the previous 2D simulations. The only difference is that we need to define the boundary conditions in the third dimension.

In the folder **$path_to_openfoamcourse/first_tutorial** you will find a copy of this tutorial. In the terminal window type:

- yPlusRAS -latestTime
- Q -latestTime
- wallShearStress -latestTime
- vorticity -latestTime
- paraFoam -builtin

OpenFOAM® tutorials

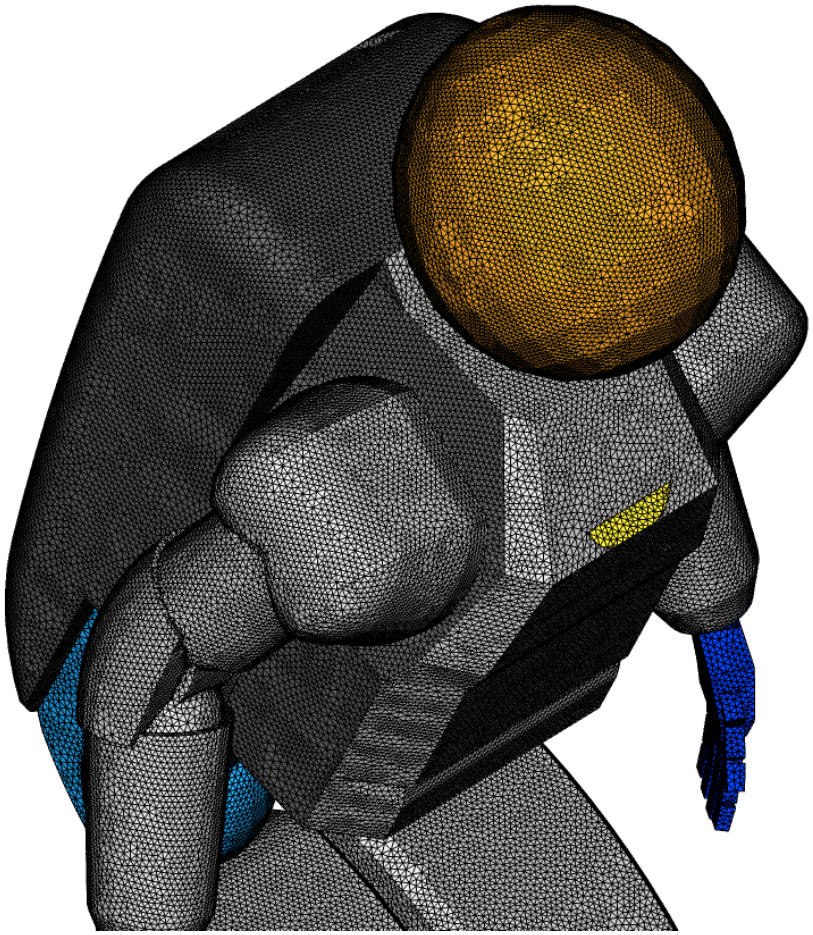
Additional tutorials

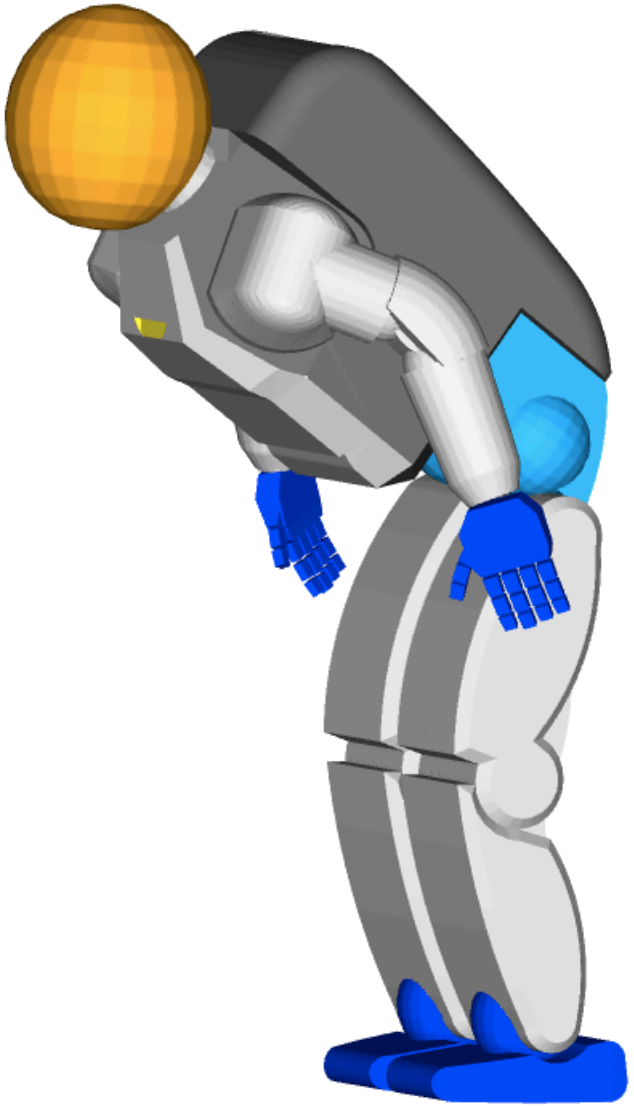
The OpenFOAM® installation comes with many tutorials, try to do all of them (or at least those that interest you).

In the course's directory **($path_to_openfoamcourse)** you will find many tutorials (which are different from those that come with the OpenFOAM® installation), try to go through each one to understand and get functional using OpenFOAM®.

Finally, fell free to bring your own cases.

Thank you for your attention

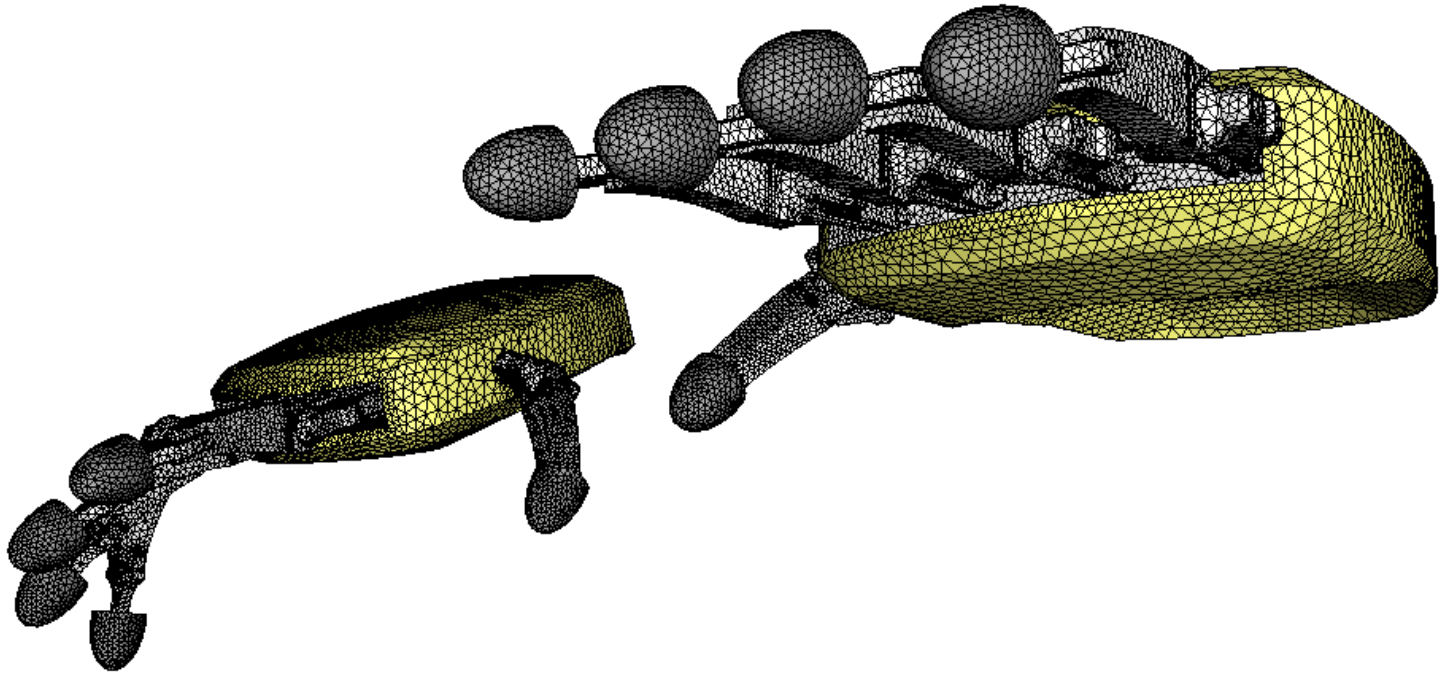




Today's lecture

1. ~~What is OpenFOAM®? – Brief overview~~
2. ~~OpenFOAM® directory organization~~
3. ~~Directory structure of an application/utility~~
4. ~~Applications/utilities in OpenFOAM®~~
5. ~~Directory structure of an OpenFOAM® case~~
6. ~~My first OpenFOAM® case setup~~
7. ~~A deeper view to my first OpenFOAM® case setup~~
8. ~~My second OpenFOAM® case setup~~
9. ~~My third OpenFOAM® case setup~~
10. ~~My first 3D OpenFOAM® case setup~~
11. **Hands-on session**

Hands-on session



In the course's directory **($path_to_openfoamcourse)** you will find many tutorials (which are different from those that come with the OpenFOAM® installation), let us try to go through each one to understand and get functional using OpenFOAM®.

If you have a case of your own, let me know and I will try to do my best to help you to setup your case. But remember, the physics is yours.