

# OpenFOAM directory organization

## OpenFOAM directory organization

We will browse the directories graphically in Linux.

First browse to the installation directory and create a bookmark so that you can easily return.

You can also use the Linux command `tree` to examine the source code directory organization: (this is used to create the slides)

```
tree -d -L 1 $WM_PROJECT_DIR
```

yielding (version dependent):

```
$WM_PROJECT_DIR  
|-- applications  
|-- bin  
|-- doc  
|-- etc  
|-- platforms  
|-- src  
|-- tutorials  
`-- wmake
```

In `$WM_PROJECT_DIR` you can also find release notes etc., but most importantly:

`Allwmake`

which compiles all of OpenFOAM by calling other `Allwmake` scripts.

## The applications directory

```
tree -d -L 1 $WM_PROJECT_DIR/applications
```

(or `tree -d -L 1 $FOAM_APP`) yields (version dependent):

```
$WM_PROJECT_DIR/applications
```

```
|-- solvers
```

```
|-- test
```

```
`-- utilities
```

Here is a short description of the applications directory contents:

- `solvers` contains source code for the distributed solvers
- `test` contains source code that test and show example of the usage of some of the Open-FOAM classes (we will use some later)
- `utilities` contains source code for the distributed utilities

There is also an `Allwmake` script, which will compile all the contents of `solvers` and `utilities`

## The src directory

The `src` directory contains the source code for all the libraries.

(try: `tree -d -L 1 $WM_PROJECT_DIR/src`, or `$FOAM_SRC`)

It is divided in different subdirectories, containing libraries

(one library for each Make directory: `src; find . -name Make`)

The most relevant are:

- `finiteVolume`. This library provides all the classes needed for the finiteVolume discretization, such as the `fvMesh` class, finiteVolume discretization operators (divergence, laplacian, gradient, and `fvc/fvm`), and boundary conditions (`fields/fvPatchFields`). In `cfTools/general/include/` 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 of all the mesh features such as zones and sets
- `turbulenceModels` which contains many turbulence models
- `dynamicFvMesh` for moving meshes algorithms

## The bin, doc, etc, platforms, and tutorials directories

The `bin` directory contains *shell scripts*, such as `paraFoam`, `foamNew`, `foamLog` ...

The `doc` directory contains the documentation of OpenFOAM:

- Programmers and User Guide
- Doxygen generated documentation in html format

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
```

(The Doxygen documentation will not work now since it is not compiled. Compile by `./Allwmake doc`. For now, have a look at [www.openfoam.org/docs/cpp/](http://www.openfoam.org/docs/cpp/))

The `etc` directory contains environment set-up files, global `controlDict`, and default `thermoData`.

The `platforms` directory contains the *binaries* of the applications (`bin`) and dynamic libraries (`lib`).

The `tutorials` directory contains example cases for each solver.

## 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 (some of) the output from the `wmake` command.

If you added 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 some scripts that are useful when organizing your files for compilation, or for cleaning up.

## User directory organization

First browse to the user directory and create a bookmark so that you can easily return.

- The `$WM_PROJECT_USER_DIR` environment variable is set up as a suggested location of the user development and cases. It is empty from scratch, but we have created some directories to prepare.
- When you do compile your first application and library, specifying `FOAM_USER_APPBIN` and `FOAM_USER_LIBBIN`, your binary files will be located there in the same structure as in `$WM_PROJECT_DIR`.
- To make it easier to relate your own developments to the original code, I recommend to organize the source files of your own developments using the same structure as in `$WM_PROJECT_DIR`. However, it is not a requirement.