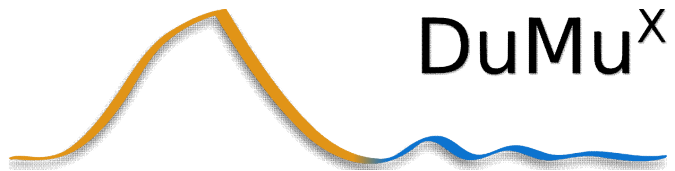




Universität Stuttgart  
DuMu<sup>x</sup> – Short Course



# Runtime Parameters and Grids

Introduction to DuMu<sup>x</sup>

# Runtime Parameters

## Runtime Parameters

- Avoid recompiling
- The same executable can be used to test different sets of parameters via shell script

Practical concern:

- If the parameter can be set at compile time, it is often cleaner and faster. Only use parameters when need be.

## Input file contents

- Input file syntax:

```
[MyGroup]  
MyParameter = 2
```

- Examples:

Groups	Parameters		
Component	GasDensity	LiquidDensity	MolarMass
Problem	EnableGravity	Name	
Grid	Cells	LowerLeft	UpperRight

## Reading Runtime Parameters

- Use runtime parameters in any file in dumux

```
paramname_ = getParam<TYPE>("GROUPNAME.PARAMNAME");
```

or

```
paramname_ = getParamFromGroup<TYPE>("GROUPNAME", "PARAMNAME");
```

- You can also set a default value when calling a runtime parameter. In the case that no parameter is set at runtime, the default value will be used. Be careful! Many parameters already have a default value.

```
paramname_ = getParam<TYPE>("GROUPNAME.PARAMNAME", default);
```

## Functions hasParam and hasParamInGroup

- Check: Parameter in input file?

```
if (hasParam("GROUPNAME.PARAMNAME"))  
    std::cout << "GROUPNAME.PARAMNAME is read from the input  
    file." << std::endl;  
else  
    std::cout << "Using the default value for  
    GROUPNAME.PARAMNAME." << std::endl;
```

- Another way of writing the upper

```
if (hasParamInGroup("GROUPNAME", "PARAMNAME"))
```

instead of

```
if (hasParam("GROUPNAME.PARAMNAME"))
```

## Input file contents

- Input file syntax:

```
[MyGroup]  
MyParameter = 2
```

- Examples:

Groups	Parameters		
Component	GasDensity	LiquidDensity	MolarMass
Problem	EnableGravity	Name	
Grid	Cells	LowerLeft	UpperRight



**Grids**



## Grid types

- YASPGGrid (structured, n-dimensional, parallel, tensorproduct grid)
  - UGGrid (2D/3D, unstructured, parallel, multi-geometry)
  - ALUGrid (2D/3D, unstructured, parallel, simplex/cube)
  - FOAMGrid (1D-2D, Dynamic, Multi-dimension/Network Problems)
  - OPMGrid ( Cornerpoint grids)
- 
- Set in problem file:

```
// Set the grid type
template<class TypeTag>
struct Grid<TypeTag, Ttag::Injection2p>
{ using type = Dune::YaspGrid<2>; };
```

## Input file – example grid sections

### CREATE THE GRID IN THE MAIN FILE

```
// try to create a grid (from a grid file or the input file)
GridManager<GetPropType<TypeTag, Properties::Grid>> gridManager;
gridManager.init();
```

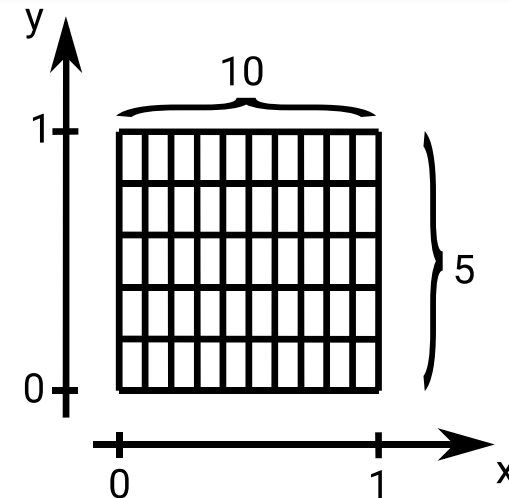
### READ IN FROM FILE

```
[Grid]
File = ./grids/heterogeneousSmall.dgf # relative path to the
grid file
```

### DEVELOP FROM BASIC SPATIAL DESCRIPTIONS

```
[Grid]
LowerLeft = 0 0
UpperRight = 1 1
Cells = 10 5
```

3 entries in 3D



## Read grid from file – supported formats

- DGF (Dune grid format)
- Gmsh ([gmsh.info](http://gmsh.info))
- Cornerpoint grid format (.grdecl file, see opm)

## Exercises

- Exercise about runtime parameters

Go to <https://git.iws.uni-stuttgart.de/dumux-repositories/dumux-course/tree/master/exercises/exercise-runtimeparams> and check out the README

- Exercise about grids

Go to <https://git.iws.uni-stuttgart.de/dumux-repositories/dumux-course/tree/master/exercises/exercise-grids> and check out the README

**Thank you!**