



# FEniCS Course

## Lecture 1: Introduction to FEniCS

---

### *Contributors*

Anders Logg

André Massing

What is FEniCS?

# FEniCS is an automated programming environment for differential equations

- C++/Python library
- Initiated 2003 in Chicago
- 1000–2000 monthly downloads
- Part of Debian and Ubuntu
- Licensed under the GNU LGPL



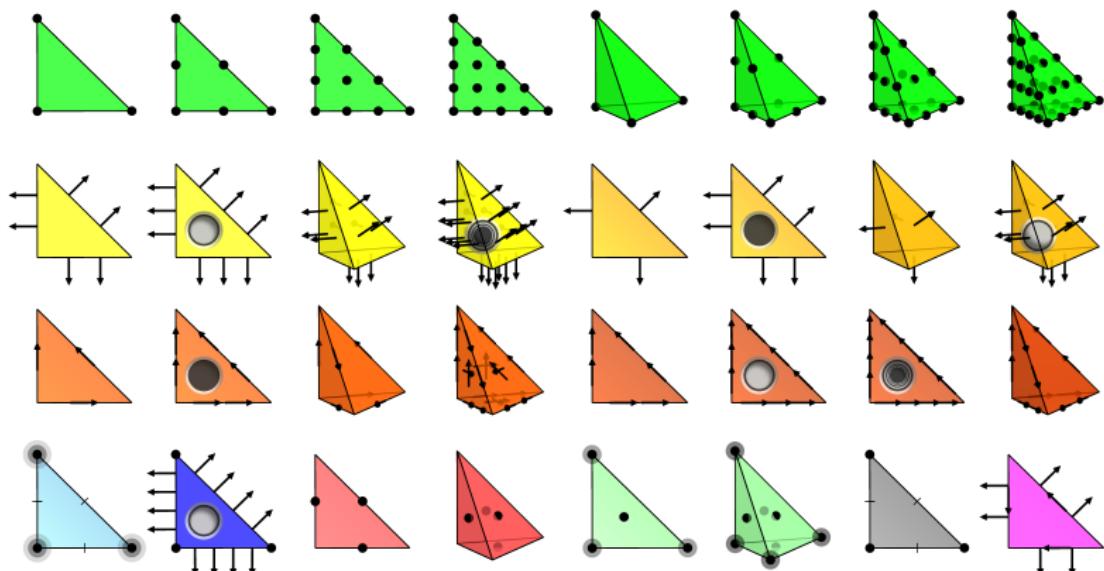
<http://fenicsproject.org/>

## Collaborators

*Simula Research Laboratory, University of Cambridge,  
University of Chicago, Texas Tech University, KTH Royal  
Institute of Technology, Chalmers University of Technology,  
Imperial College London, University of Oxford, Charles  
University in Prague, ...*

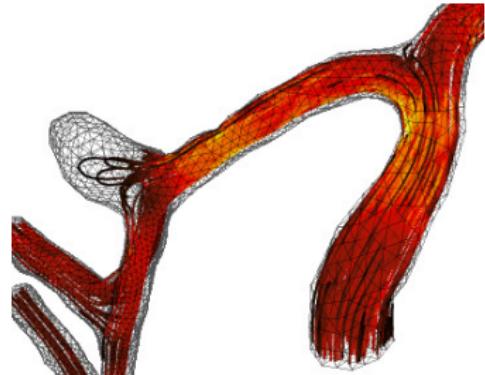
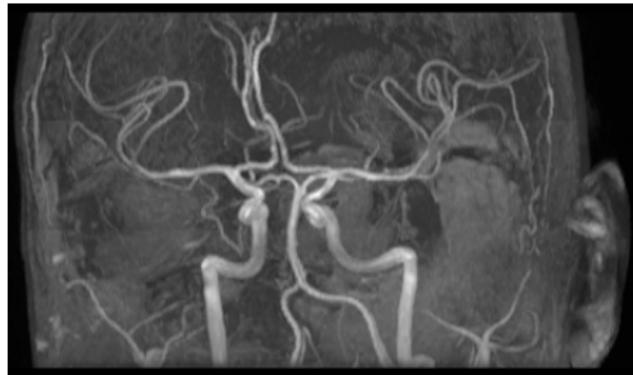
# FEniCS is automated FEM

- Automated generation of basis functions
- Automated evaluation of variational forms
- Automated finite element assembly
- Automated adaptive error control



What has FEniCS been used for?

# Computational hemodynamics



- Low wall shear stress may trigger aneurysm growth
- Solve the incompressible Navier–Stokes equations on patient-specific geometries

$$\begin{aligned}\dot{u} + u \cdot \nabla u - \nabla \cdot \sigma(u, p) &= f \\ \nabla \cdot u &= 0\end{aligned}$$

# Computational hemodynamics (contd.)



Python code

```
# Define Cauchy stress tensor
def sigma(v,w):
    return 2.0*mu*0.5*(grad(v) + grad(v).T) -
           w*Identity(v.cell().d)

# Define symmetric gradient
def epsilon(v):
    return 0.5*(grad(v) + grad(v).T)

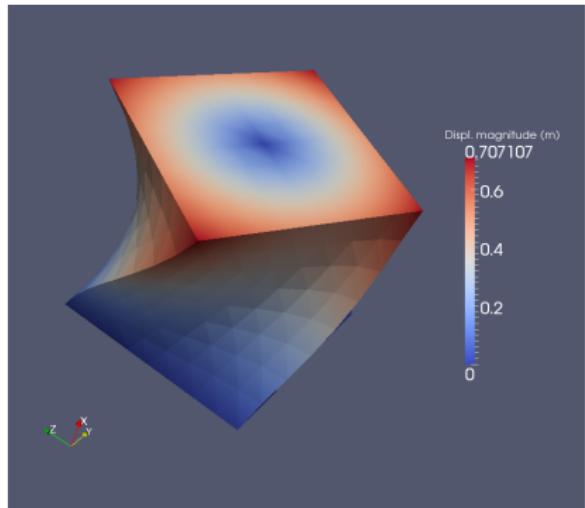
# Tentative velocity step (sigma formulation)
U = 0.5*(u0 + u)
F1 = rho*(1/k)*inner(v, u - u0)*dx + rho*inner(v,
    grad(u0)*(u0 - w))*dx \
    + inner(epsilon(v), sigma(U, p0))*dx \
    + inner(v, p0*n)*ds - mu*inner(grad(U).T*n,
        v)*ds \
    - inner(v, f)*dx
a1 = lhs(F1)
L1 = rhs(F1)

# Pressure correction
a2 = inner(grad(q), k*grad(p))*dx
L2 = inner(grad(q), k*grad(p0))*dx - q*div(u1)*dx

# Velocity correction
a3 = inner(v, u)*dx
L3 = inner(v, u1)*dx + inner(v, k*grad(p0 -
    p1))*dx
```

- The Navier–Stokes solver is implemented in Python/FEniCS
- FEniCS allows solvers to be implemented in a minimal amount of code

# Hyperelasticity



Python code

```
from fenics import *
mesh = UnitCubeMesh(24, 16, 16)
V = VectorFunctionSpace(mesh, "Lagrange", 1)

left = CompiledSubDomain("(std::abs(x[0]) < DOLFIN_EPS) && on_boundary")
right = CompiledSubDomain("(std::abs(x[0] - 1.0) < DOLFIN_EPS) && on_boundary")

c = Expression(("0.0", "0.0", "0.0"))
r = Expression(("0.0",
"0.5*(y0+(x[1]-y0)*cos(t)-(x[2]-z0)*sin(t)-x[1])",
"0.5*(z0+(x[1]-y0)*sin(t)+(x[2]-z0)*cos(t)-x[2])"),
y0=0.5, z0=0.5, t=pi/3)
bcl = DirichletBC(V, c, left)
bcr = DirichletBC(V, r, right)
bcs = [bcl, bcr]
v = TestFunction(V)
u = Function(V)
B = Constant((0.0, -0.5, 0.0))
T = Constant((0.1, 0.0, 0.0))
I = Identity(V.cell().d)
F = I + grad(u)
Ic = tr(F.T*F)
J = det(F)
E, nu = 10.0, 0.3
mu, lmbda = Constant(E/(2*(1 + nu))), Constant(E*nu/((1 + nu)*(1 - 2*nu)))
psi = (mu/2)*(Ic - 3) - mu*ln(J) + (lmbda/2)*(ln(J))**2
Pi = psi*dx - dot(B, u)*dx - dot(T, u)*ds
F = derivative(Pi, u, v)

solve(F == 0, u, bcs)
plot(u, interactive=True, mode="displacement")
```

How to use FEniCS?

# Hello World in FEniCS: problem formulation

## Poisson's equation

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega \\ u &= 0 && \text{on } \partial\Omega \end{aligned}$$

## Finite element formulation

Find  $u \in V$  such that

$$\underbrace{\int_{\Omega} \nabla u \cdot \nabla v \, dx}_{a(u,v)} = \underbrace{\int_{\Omega} f v \, dx}_{L(v)} \quad \forall v \in V$$

# Hello World in FEniCS: implementation

*Python code*

```
from fenics import *

mesh = UnitSquareMesh(32, 32)

V = FunctionSpace(mesh, "Lagrange", 1)
u = TrialFunction(V)
v = TestFunction(V)
f = Expression("x[0]*x[1]")

a = dot(grad(u), grad(v))*dx
L = f*v*dx

bc = DirichletBC(V, 0.0, DomainBoundary())

u = Function(V)
solve(a == L, u, bc)
plot(u)
```

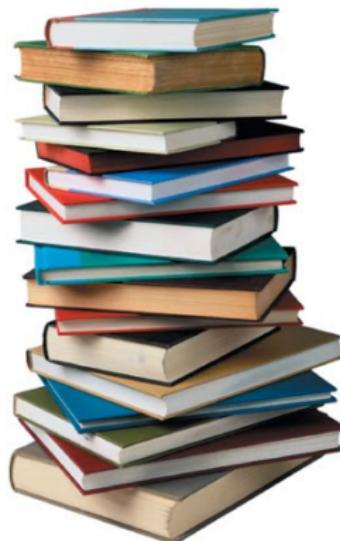
# Basic API

- Mesh, Vertex, Edge, Face, Facet, Cell
  - FiniteElement, FunctionSpace
  - TrialFunction, TestFunction, Function
  - grad(), curl(), div(), ...
  - Matrix, Vector, KrylovSolver, LUSolver
  - assemble(), solve(), plot()
- 
- Python interface generated semi-automatically by SWIG
  - C++ and Python interfaces almost identical

Sounds great, but how do I find my way through the jungle?



# Three survival advices



Use the right Python tools

Explore the documentation

Ask, report and request

Use the right Python tools!

# Python tools

## Doc tools

- Standard terminal:  
  > pydoc dolfin  
  > pydoc dolfin.Mesh
- Python console  
  >>> help(dolfin)  
  >>> help(dolfin.Mesh)

## Sophisticated Python environments

**IDLE** the official (but rather limited) Python IDE

**IPython** <http://ipython.org/>  
provides a Python shell and notebook including syntax highlighting, tab-completion, object inspection, debug assisting, history ...

**Eclipse** plugin <http://pydev.org/>  
includes syntax highlighting, code completion, unit-testing, refactoring, debugger ...

# IPython netbook

IP[y]: Notebook

solving-poisson Save QuickHelp

Actions New Open Download iPython Print

Cell Actions Delete Format Code Markdown Output Toggle ClearAll Insert Above Below Move Up Down Run Selected All Autosindent

Kernel Actions Interrupt Restart Kill kernel upon exit

Help Links Python iPython NumPy SciPy MPL SymPy

Shift-Enter: run selected cell  
Ctrl-Enter: run selected cell in-place  
Ctrl-m h: show keyboard shortcuts

Configuration Tooltip on tab   
Smart completions   
Time before tooltip: 1200 milliseconds

Let's solve numerically the following variational problem: Find  $u \in H_0^1(\Omega)$  such that  $a(u, v) = L(v) \forall v \in H_0^1(\Omega)$  where  $a(u, v) = \int_{\Omega} \nabla u \cdot \nabla v \, dx$  and  $L(v) := \int_{\Omega} f v \, dx$ . To do that in FEniCS we start by defining a mesh:

```
In [26]: from dolfin import *
m = UnitSquare(10,10)
print m
```

<Mesh of topological dimension 2 (triangles) with 101 vertices and 100 cells, ordered>

Now we need some function space. Let's take P1 elements:

```
In [27]: V = FunctionSpace(m, "CG", 1)
```

It's time to define some test and trial functions.

```
In [28]: u = TrialFunction(V)
v = TestFunction(V)
```

And finally we can define the variational problem:

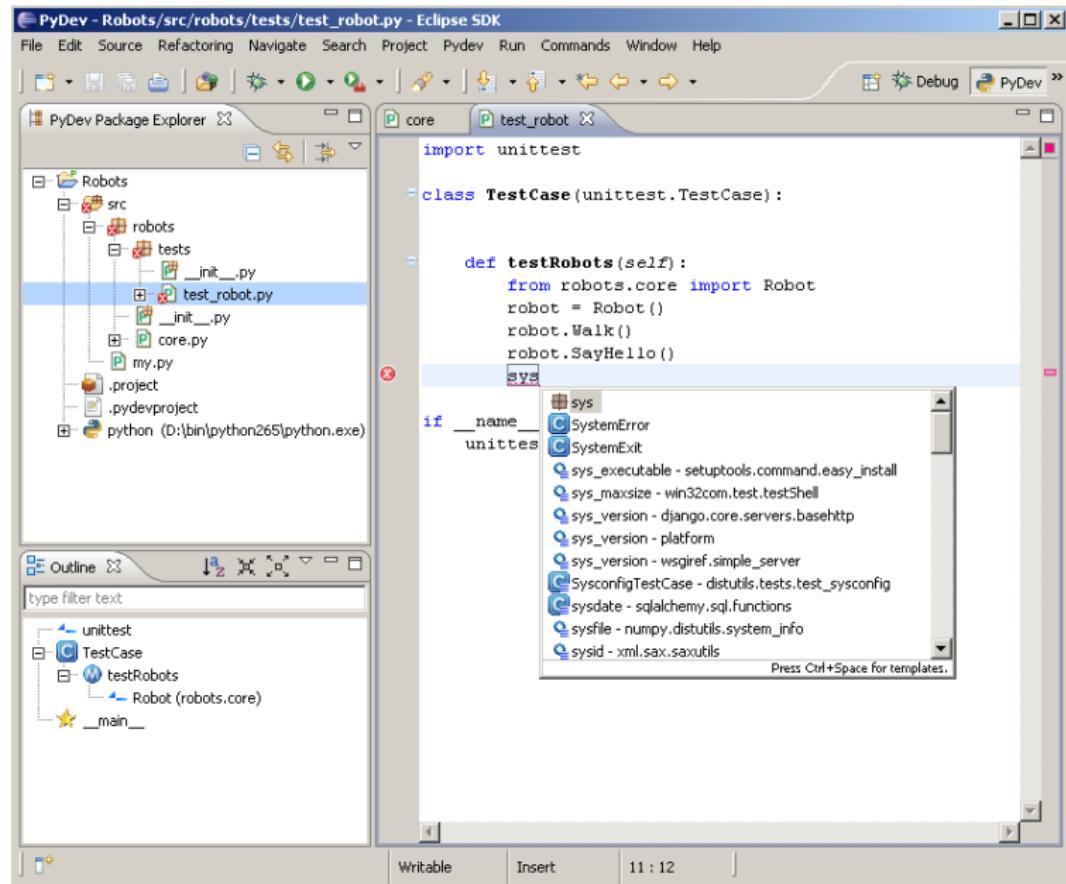
```
In [29]: a = inner(grad(u), grad(v)) * dx
f = Constant(1)
L = f*v*dx
def boundary(x, on_boundary):
    return on_boundary
u = Function(V)
zero = Constant(0)
bc = DirichletBC(V, zero, boundary)
solve(a==L, u, bc)
```

```
In [30]: plot(u)
```

```
Out[30]: <vipper.viper_dolfin.Viper at 0xb7890>
```

```
In [31]: interactive()
```

# Eclipse plugin Pydev



Explore the FEniCS documentation!



FEniCS  
PROJECT

About Download Documentation Applications Contributing Citing Support

## Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment [DOLFIN](#) and the form language [UFL](#).

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

### The FEniCS Tutorial

A good starting point for new users is the [FEniCS Tutorial](#). The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

### The FEniCS Book



*The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method*, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

### The FEniCS Manual

*The FEniCS Manual* is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

### Additional Documentation

[Mixing software with FEniCS](#) is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

### Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

---

Documented DOLFIN demos (Python)

Documented DOLFIN demos (C++)

---

The demos are [already installed on your system](#) or can be found in the demo directory of the DOLFIN source tree.

### Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

---

Basic classes and functions in DOLFIN (Python)

Basic classes and functions in DOLFIN (C++)

---

### Complete Programmer's References

---

All classes and functions in DOLFIN (Python)

All classes and functions in DOLFIN (C++)

---

All classes and functions in UFL

---

<http://fenicsproject.org/documentation/>

## Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment [DOLFIN](#) and the form language [UFL](#).

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

### The FEniCS Tutorial

A good starting point for new users is the [FEniCS Tutorial](#). The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.



### The FEniCS Book



[The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method](#), is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software development as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

### The FEniCS Manual

[The FEniCS Manual](#) is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

### Additional Documentation

[Mixing software with FEniCS](#) is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

### Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

---

Documented DOLFIN demos (Python)

Documented DOLFIN demos (C++)

---

The demos are [already installed on your system](#) or can be found in the demo directory of the DOLFIN source tree.

### Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

---

Basic classes and functions in DOLFIN (Python)

Basic classes and functions in DOLFIN (C++)

---

### Complete Programmer's References

---

All classes and functions in DOLFIN (Python)

All classes and functions in DOLFIN (C++)

---

All classes and functions in UFL

---

<http://fenicsproject.org/documentation/>



About Download Documentation Applications Contributing Citing Support

## Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment [DOLFIN](#) and the form language [UFL](#).

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

### The FEniCS Tutorial

A good starting point for new users is the [FEniCS Tutorial](#). The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

### The FEniCS Book



*The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method*, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software development as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.



### The FEniCS Manual

[The FEniCS Manual](#) is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

### Additional Documentation

[Mixing software with FEniCS](#) is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

### Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

[Documented DOLFIN demos \(Python\)](#)

[Documented DOLFIN demos \(C++\)](#)

The demos are [already installed on your system](#) or can be found in the demo directory of the DOLFIN source tree.

### Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

[Basic classes and functions in DOLFIN \(Python\)](#)

[Basic classes and functions in DOLFIN \(C++\)](#)

### Complete Programmer's References

[All classes and functions in DOLFIN \(Python\)](#)

[All classes and functions in DOLFIN \(C++\)](#)

[All classes and functions in UFL](#)

<http://fenicsproject.org/documentation/>



About Download Documentation Applications Contributing Citing Support

## Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment **DOLFIN** and the form language **UFL**.

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

### The FEniCS Tutorial

A good starting point for new users is the [FEniCS Tutorial](#). The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

### The FEniCS Book



*The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method*, is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

### The FEniCS Manual



[The FEniCS Manual](#) is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

### Additional Documentation

[Mixing software with FEniCS](#) is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

### Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

[Documented DOLFIN demos \(Python\)](#)

[Documented DOLFIN demos \(C++\)](#)

The demos are [already installed on your system](#) or can be found in the demo directory of the DOLFIN source tree.

### Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

[Basic classes and functions in DOLFIN \(Python\)](#)

[Basic classes and functions in DOLFIN \(C++\)](#)

### Complete Programmer's References

[All classes and functions in DOLFIN \(Python\)](#)

[All classes and functions in DOLFIN \(C++\)](#)

[All classes and functions in UFL](#)

<http://fenicsproject.org/documentation/>



About Download Documentation Applications Contributing Citing Support

## Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment [DOLFIN](#) and the form language [UFL](#).

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

### The FEniCS Tutorial

A good starting point for new users is the [FEniCS Tutorial](#). The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

### The FEniCS Book



[The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method](#), is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software development as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

### The FEniCS Manual

[The FEniCS Manual](#) is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

### Additional Documentation

[Mixing software with FEniCS](#) is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

### Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

[Documented DOLFIN demos \(Python\)](#)

[Documented DOLFIN demos \(C++\)](#)

The demos are [already installed on your system](#) or can be found in the demo directory of the DOLFIN source tree.



### Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

[Basic classes and functions in DOLFIN \(Python\)](#)

[Basic classes and functions in DOLFIN \(C++\)](#)

### Complete Programmer's References

[All classes and functions in DOLFIN \(Python\)](#)

[All classes and functions in DOLFIN \(C++\)](#)

[All classes and functions in UFL](#)

<http://fenicsproject.org/documentation/>



About Download Documentation Applications Contributing Citing Support

## Documentation for FEniCS 1.3.0

Our documentation includes a book, a collection of documented demo programs, and complete references for the FEniCS application programming interface (API). Note that the FEniCS API is documented separately for each FEniCS component. The most important interfaces are those of the C++/Python problem solving environment [DOLFIN](#) and the form language [UFL](#).

(This page accesses the FEniCS 1.3.0 documentation. Not the version you are looking for? See [all versions](#).)

### The FEniCS Tutorial

A good starting point for new users is the [FEniCS Tutorial](#). The tutorial will help you get quickly up and running with solving differential equations in FEniCS. The tutorial focuses exclusively on the FEniCS Python interface, since this is the simplest approach to exploring FEniCS for beginners.

### The FEniCS Book



[The FEniCS Book, Automated Solution of Differential Equations by the Finite Element Method](#), is a comprehensive (700 pages) book documenting the mathematical methodology behind the FEniCS Project and the software developed as part of the FEniCS Project. The FEniCS Tutorial is included as the opening chapter of the FEniCS Book.

### The FEniCS Manual

[The FEniCS Manual](#) is a 200-page excerpt from the FEniCS Book, including the FEniCS Tutorial, an introduction to the finite element method and documentation of DOLFIN and UFL.

### Additional Documentation

[Mixing software with FEniCS](#) is a tutorial on how to combine FEniCS applications in Python with software written in other languages.

### Demos

A simple way to build your first FEniCS application is to copy and modify one of the existing demos:

Documented DOLFIN demos (Python)

Documented DOLFIN demos (C++)

The demos are [already installed on your system](#) or can be found in the demo directory of the DOLFIN source tree.

### Quick Programmer's References

Some of the classes and functions in DOLFIN are more frequently used than others. To learn more about these, take a look at the

Basic classes and functions in DOLFIN (Python)

Basic classes and functions in DOLFIN (C++)

### Complete Programmer's References

All classes and functions in DOLFIN (Python)

All classes and functions in DOLFIN (C++)

All classes and functions in UFL



<http://fenicsproject.org/documentation/>

Ask questions, report bugs and request new features!

# Development community is organized via bitbucket.org

The screenshot shows the Bitbucket interface for the `fenics-project/DOLFIN` repository. The left sidebar contains navigation links such as Overview, Source, Commits, Branches, Pull requests, Issues, Wiki, Downloads, and Settings. The main content area displays an 'Overview' card with statistics: Last updated 6 minutes ago, Language C++, Access level Admin (revoke), 99+ Branches, 3 Tags, 48 Forks, and 73 Watchers. Below the overview, there's a section for the DOLFIN project, which is described as the C++/Python interface of FEniCS. It includes instructions for building, a code snippet for the build process, and a link to the INSTALL file. A 'Recent activity' sidebar shows four commits from Garth Wells:

- 1 commit Pushed to fenics-project/dolfin | 8bc2b98 Merge branch 'garth/replace-bo...' Garth Wells - 7 minutes ago
- 1 commit Pushed to fenics-project/dolfin | 8d64a67 Merge branch 'garth/replace-lexi...' Garth Wells - 10 minutes ago
- 1 commit Pushed to fenics-project/dolfin | ab50d44 Replace Boost lexical\_cast with ... Garth Wells - 10 minutes ago
- 1 commit Pushed to fenics-project/dolfin | be6345d Merge branch 'garth/replace-bo...' Garth Wells - 35 minutes ago

<http://bitbucket.org/fenics-project/>

# Community help is available via QA forum

The screenshot shows a Mozilla Firefox browser window displaying the FEniCS Q&A forum at [fenicsproject.org/qa/](https://fenicsproject.org/qa/). The page lists six unanswered questions:

- How to print nodal values of the numerical solution to a Neumann boundary-value problem?**  
answered 3 days ago by [umberto](#) FEniCS User (1,810 points)  
neumann | array | vector | solution | coordinates
- Neumann boundary condition on complex domain**  
answered 3 days ago by [umberto](#) FEniCS User (1,810 points)  
boundary-conditions | neumann | complex | domain
- Petsc Matrix Multiplication and Transposition**  
answered 6 days ago by [umberto](#) FEniCS User (1,810 points)  
petscmatrix | matrix-multiply | matrix-transpose
- Explain FEniCS's boundary conditions**  
answered 6 days ago by [chris\\_richardson](#) FEniCS Expert (11,600 points)  
boundary-conditions | waveguide | electromagnetics
- Convert dolfin generic vector into PETSc vector**  
answered 6 days ago by [MiroK](#) FEniCS Expert (43,240 points)  
petsc | petscvector
- How can I convert an image .tiff in a dolfin xml?**  
answered May 28 by [christianv](#) FEniCS User (2,050 points)  
dolfin-convert
- Piecewise definition of grad (help with syntax)**  
answered May 28 by [MiroK](#) FEniCS Expert (43,240 points)  
grad | computation

<https://fenicsproject.org/qa>

# Installation



- ☞ Use official packages for Debian and Ubuntu



- ☞ Use drag and drop installation on Mac OS X



- ☞ Use VirtualBox + official FEniCS image



- ☞ Build from source (fenics-install.sh)



- ☞ Other options: Docker, Conda packages

<http://fenicsproject.org/download/>

# *The FEniCS challenge!*

- ① Install FEniCS on your laptop!

<http://fenicsproject.org/download/>

- ② Find and execute `demo_cahn-hilliard.py`, try to visualize the results with Paraview.
- ③ What are the main packages of the `dolfin` module?
- ④ Which elements are supported in `dolfin`?
- ⑤ Plot at least two finite elements from each row on page 4 and identify those elements you are most curious about!