Writing useful Documentation

David Grellscheid

Durham University

ICTP The Abdus Salam International Centre for Theoretical Physics

Documentation is communication

Need to consider all levels, they have different audiences:

* Code annotations: formatting, comments
* Structure-level documentation (funcs, classes)
* End-user reference material
* Introduction for new users...
* ... and new developers (often left out!)

Code formatting

Code will be read **much** more often than written.

* clarity wins over cleverness
* choose_a_style and stickWithIt:
  block structure, indentation, line length,
  variable naming, ...
* know the conventions of the language community
  (in Python, look for **PEP 8)**

Be consistent with others!

Code comments

Explain the intention, not the work:

```
def tripleTuple(x):
    # assign x to y
    y = x
    # assign x to z
    z = x
    # double y
    y *= 2
    # triple z
    z *= 3
    # create tuple
    t = (x, y, z)
    # return the tuple
    return t
```

```
def tripleTuple(x):
    y = z = x
    # apply foo scaling, see [34] eq (2.3)
    y *= 2
    z *= 3
    return (x, y, z)
```

* deviations from standard
* unexpected choices of implementation

Function-level docs

Dual audience: developers and end users

* describe role of function
* intended input / output
* mention side effects!
* use templates for blank header files

```
def tripleTuple(x):
    """Apply foo's tripling to x."""
    y = z = x
    # apply the scaling, see [34] eq (2.3)
    y *= 2
    z *= 3
    return (x, y, z)
```

```
/**
 *
 * Choose a pair of hadrons.
 *
 * Given the mass of a cluster and the particle pointers of its
 * two (or three) constituents, return the pair of particle pointers of
 * the two hadrons with proper flavour numbers.
 * Furthermore, the first of the two hadrons must have the
 * constituent with par1, and the second must have the constituent with par2.
 *
 * At the moment it does *nothing* in the case that par3 is also present.
 *
 * Kupco's method is used, rather than one used in FORTRAN HERWIG
 * The idea is to build on the fly a table of all possible pairs
 * of hadrons (Had1,Had2) (that we can call "cluster decay channels")
 * which are kinematically above threshold  and have flavour
 * Had1=(par1,quarktopick->CC()), Had2=(quarktopick,par2), where quarktopick
 * is the pointer of:
 *    ---  d, u, s, c, b
 *                         if either par1 or par2 is a diquark;
 *    ---  d, u, s, c, b, dd, ud, uu, sd, su, ss,
 *                         cd, cu, cs, cc, bd, bu, bs, bc, bb
 *                         if both par1 and par2  are quarks.
 * The weight associated with each channel is given by the product
 * of: the phase space available including the spin factor 2*J+1,
 *     the constant weight factor for chosen idQ,
 *     the octet-singlet isoscalar mixing factor, and finally
 *     the singlet-decuplet weight factor.
 */
pair<tcPDPtr,tcPDPtr> chooseHadronPair(const Energy cluMass,
                                       tcPDPtr par1,
                                       tcPDPtr par2,
                                       tcPDPtr par3 = PDPtr());
```

Module-level docs

Same audience: developers and end users

zoomed-out view

```
namespace Herwig {

using namespace ThePEG;

/** \ingroup hadronization
 * The HwppSelector class selects the hadrons produced in cluster decay using
 * the Herwig++ variant of the cluster model.
 *
 * @see \ref HwppSelectorInterfaces "The interfaces"
 * defined for HwppSelector.
 */
class HwppSelector: public HadronSelector {
  ...
};
}
```

API / program behaviour

No surprises!

Users expect behaviour from other tools, stick to it:

* API argument order
* Command-line behaviour
  -f / --foo, --help, --version

```
import argparse
```

Provide manpages to describe these

The Abdus Salam International Centre for Theoretical Physics

Documentation tools

Extract documentation from code

* pydoc
* Doxygen
* Sphinx

The Abdus Salam International Centre for Theoretical Physics

Pydoc

Pydoc extracts docstrings defined in `http://www.python.org/dev/peps/pep-0257/`

A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition. Such a docstring becomes the __doc__ special attribute of that object.

Every Python installation has pydoc, no extra work required.

The Abdus Salam International Centre for Theoretical Physics

```
"""
Collection of first-order ODE steppers.

First-order differential equations can be solved numerically by
stepping through the solution in discrete increments of the
independent variable (here referred to as time 't'). Several methods
with varying convergence behaviours are implemented.
"""


def euler(state, d_dt, dt, boundary=None):
    """
    Euler stepper.

    Evolve the system 'state' by the time derivative function 'd_dt'
    over the timestep dt, using the Euler method. Optionally use a
    boundary condition, which can modify both the state and the result
    of d_dt(state).
    """
    f = d_dt(state)
    # boundary effects
    if boundary:
        boundary(state,f)
    state += f * dt

def rk4(state, d_dt, dt, boundary=None):
    """
    4th-order Runge-Kutta stepper.

    Evolve the system 'state' by the time derivative function 'd_dt'
    over the timestep dt, using Runge-Kutta 4th-order. Optionally use
    a boundary condition, which can modify both the state and the
    result of d_dt(state).
    """
    x0 = state
    f1 = d_dt(x0)
    x1 = x0 + f1 * dt/2.
    f2 = d_dt(x1)
    x2 = x0 + f2 * dt/2.
    f3 = d_dt(x2)
    x3 = x0 + f3 * dt
    f4 = d_dt(x3)
    f = ( f1 + 2*f2 + 2*f3 + f4 )/6.
    # boundary effects
    if boundary:
        boundary(x0,f)
    state += f * dt
```

```
$ pydoc steppers
Help on module steppers:

NAME
    steppers - Collection of first-order ODE steppers.

FILE
    /Users/dg/bikegenes/steppers.py

DESCRIPTION
    First-order differential equations can be solved numerically by
    stepping through the solution in discrete increments of the
    independent variable (here referred to as time 't'). Several methods
    with varying convergence behaviours are implemented.

FUNCTIONS
    euler(state, d_dt, dt, boundary=None)
        Euler stepper.

        Evolve the system 'state' by the time derivative function 'd_dt'
        over the timestep dt, using the Euler method. Optionally use a
        boundary condition, which can modify both the state and the result
        of d_dt(state).

    rk4(state, d_dt, dt, boundary=None)
        4th-order Runge-Kutta stepper.

        Evolve the system 'state' by the time derivative function 'd_dt'
        over the timestep dt, using Runge-Kutta 4th-order. Optionally use
        a boundary condition, which can modify both the state and the
        result of d_dt(state).
```

```
$ pydoc steppers.euler
Help on function euler in steppers:

steppers.euler = euler(state, d_dt, dt, boundary=None)
    Euler stepper.

    Evolve the system 'state' by the time derivative function 'd_dt'
    over the timestep dt, using the Euler method. Optionally use a
    boundary condition, which can modify both the state and the result
    of d_dt(state).
```

```
$ pydoc numpy.dot
Help on built-in function dot in numpy:

numpy.dot = dot(...)
    dot(a, b, out=None)

    Dot product of two arrays.

    For 2-D arrays it is equivalent to matrix multiplication, and for 1-D
    arrays to inner product of vectors (without complex conjugation). For
    N dimensions it is a sum product over the last axis of `a` and
    the second-to-last of `b`::

        dot(a, b)[i,j,k,m] = sum(a[i,j,:] * b[k,:,m])

    Parameters
    ----------
    a : array_like
        First argument.
    b : array_like
        Second argument.
    out : ndarray, optional
        Output argument. This must have the exact kind that would be returned
        if it was not used. In particular, it must have the right type, must be
        C-contiguous, and its dtype must be the dtype that would be returned
        for `dot(a,b)`. This is a performance feature. Therefore, if these
        conditions are not met, an exception is raised, instead of attempting
        to be flexible.
```

The Abdus Salam International Centre for Theoretical Physics

Doxygen

* supports C++, C, Obj-C, C#, PHP, Java, Python, IDL, Fortran, VHDL, Tcl
* can extract structure from undocumented files
* graphical visualization of dependencies
* can write general pages, too

The Abdus Salam International Centre for Theoretical Physics

parses special annotations in comments:

```
/**
 * Matrix element for \f$\gamma\gamma\to q\bar{q}\f$
 * @param p1   The wavefunctions for the first  incoming photon
 * @param p2   The wavefunctions for the second incoming photon
 * @param w1   The wavefunctions for the first  outgoing W
 * @param w2   The wavefunctions for the second outgoing W
 * @param calc Whether or not to calculate the matrix element
 */
double helicityME(vector<VectorWaveFunction> & p1,
                  vector<VectorWaveFunction> & p2,
                  vector<VectorWaveFunction> & w1,
                  vector<VectorWaveFunction> & w2, bool calc) const;
```

The Abdus Salam International Centre for Theoretical Physics

parses special annotations in comments:

```
double Herwig::MEGammaGamma2WW::helicityME ( vector< VectorWaveFunction > &  p1,
                                             vector< VectorWaveFunction > &  p2,
                                             vector< VectorWaveFunction > &  w1,
                                             vector< VectorWaveFunction > &  w2,
                                             bool                            calc
                                           )                                 const   protected
```

Matrix element for $\gamma\gamma \to q\bar{q}$.

Parameters:

p1 The wavefunctions for the first incoming photon

p2 The wavefunctions for the second incoming photon

w1 The wavefunctions for the first outgoing W

w2 The wavefunctions for the second outgoing W

calc Whether or not to calculate the matrix element

The Abdus Salam International Centre for Theoretical Physics

Python www docs almost all use Sphinx

standard library, matplotlib, scipy, numpy, ...

Automatic extraction of Python docstrings

conversion from Doxygen available via “Breathe” project

Great support for additional structure

reStructuredText markup format

The Abdus Salam International Centre for Theoretical Physics

like most Python projects,

excellent support for beginners:

```
$ sphinx-quickstart
```

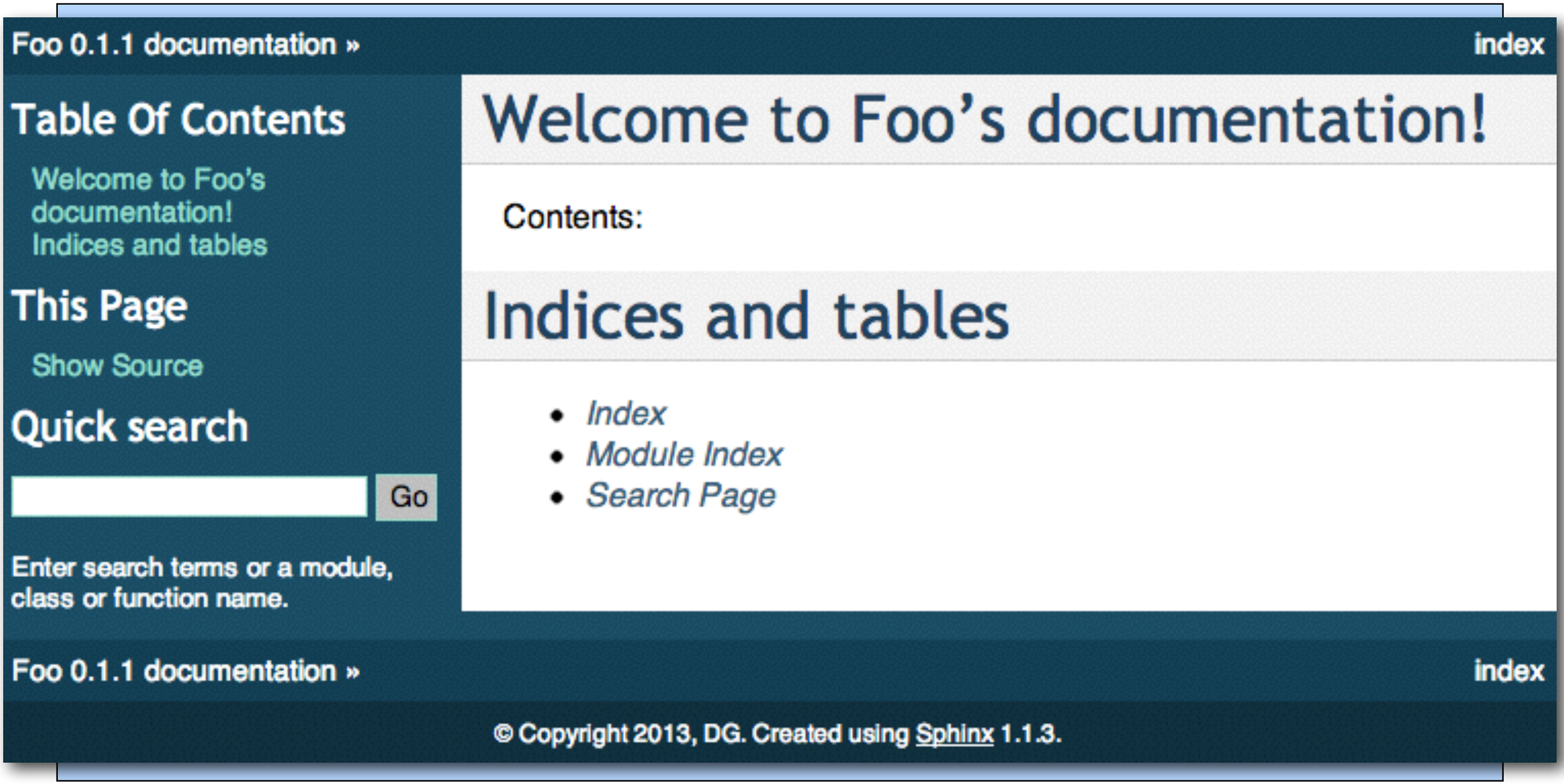
creates fully functional scaffolding,

can start adding content right away

```
.. Foobar documentation master file, created by
   sphinx-quickstart on Fri Mar 15 09:46:49 2013.
   You can adapt this file completely to your liking, but it should at least
   contain the root `toctree` directive.

Welcome to Foobar's documentation!
==================================

Contents:

.. toctree::
   :maxdepth: 2



Indices and tables
==================

* :ref:`genindex`
* :ref:`modindex`
* :ref:`search`
```

```
.. Foobar documentation master file, created by
   sphinx-quickstart on Fri Mar 15 09:46:49 2013.
   You can adapt this file completely to your liking, but it should at least
   contain the root `toctree` directive.

Welcome to Foobar's documentation!
==================================

Contents:

.. toctree::
   :maxdepth: 2



Indices and tables
==================

* :ref:`genindex`
* :ref:`modindex`
* :ref:`search`
```

```
$ make html
```


Foo 0.1.1 documentation »
index
Table Of Contents
Welcome to Foo's documentation!
Indices and tables
This Page
Show Source
Quick search
Go
Enter search terms or a module, class or function name.
Welcome to Foo's documentation!
Contents:
Indices and tables
Index
Module Index
Search Page
Foo 0.1.1 documentation »
index
© Copyright 2013, DG. Created using Sphinx 1.1.3.

The Abdus Salam International Centre for Theoretical Physics

End user documentation

Do you read manuals?

The Abdus Salam International Centre for Theoretical Physics

End user documentation

Do you read manuals?

Why not?

End user documentation

Do you read manuals?

Why not?

I find “jump right in” tutorials much more useful

Pick up the user where **they** are

Go slowly, with lots of detail

Make sure your tutorial works!

The Abdus Salam International Centre for Theoretical Physics

Rarely considered:

New developers need an intro, too!

Very different requirements than users

Doxygen alone is **not** enough

At minimum, prepare some paths to follow through the code docs.

* Your experiences of good / bad practice

* What would you like as an experienced developer?

* What would you like as a new developer?

* What would you like as an experienced user?

* What would you like as a new user?