Documentation Howto

The folders ”sphinx_beginner” and ”sphinx_intermediate” are Re-Writes of the documentation which was originally done in Word, converted to ReStructuredText.

Here, we're using ReStructuredText for writing, and Sphinx to convert the Markup into HTML and PDF (“Sphinx is a tool that makes it easy to create intelligent and beautiful documentation”)

Sphinx uses `conf.py` for configuration and provides a convenient Makefile for conversion. Just call `make html` (`make singlehtml` creates one single HTML file), or `make latexpdf` to run it. The resulting documents will be in `_build/html/` or `_build/latex/`, respectively.

Currently, we do not include the HTML output into git, just the PDF and maybe the singlehtml page.

Requirements:

- ReStructuredText https://en.wikipedia.org/wiki/ReStructuredText
- Sphinx >1.1 http://sphinx-doc.org/
- Pygments >1.5 http://pygments.org/
- TexLive >2011 http://www.tug.org/texlive/

Links:

- ReST Cheat Sheet
- ReST and Sphinx Primer
- Writing Technical Documentation with Sphinx, Paver, and Cog

PS: I've manually updated the Makefiles to do some minor `sed` replacement to use the LaTeX package fancyvrb for styling the verbatim boxes; this package should be included in the TeXLive distribution.

ToDos:

- Features to try out:
  - centered
  - hlist

  - glossary
- Include examples from “Bioinformatics One-Liners”
- Proper Use of sectioning:
  - ’#’ with overline, for parts
  - ’*’ with overline, for chapters
  - ’=’, for sections
  - ’-’, for subsections
  - ’^’, for subsubsections
  - ’“’, for paragraphs
- Better control of verbatim code-blocks:
  - distinguish user input from output (boldface?)

Change Fonts

To change the font of the (latex) output, add the respective package to the preamble in conf.py: For Tex Gyre Termes font:

```
\\usepackage{tgtermes}
```

For Palatino:

```
\\usepackage[sc]{mathpazo}
```

For full list of Tex Fonts see the LaTeX Font Catalog ...

syntax highlight styling

To change the formatting of verbatim code blocks (which are highlighted using Pygments), you can create a style file and reference this in ’conf.py’:

```
pygments_style = 'lsi.lsiClass'
```

This is the content of the file lsi.py:

```
from pygments.style import Style
from pygments.token import Keyword, Name, Comment, String, Error, Numbe

class lsiClass(Style):
    default_style = "sphinx"
    styles = {
        Comment:                'bold #800',
        Keyword:                'bold #005',
        Name:                   '#f00',
        Name.Function:          '#0f0',
        Name.Class:             'bold #0f0',
        String:                 'bg:#eee #111'
    }
```

For details, see the Pygments Docu