Sphinx-dokumentointityökalun asennus ja käyttö

Tuomas Virtanen
tuomas.virtanen@jyu.fi

Versio 0.5
1.11.2010

Jyväskylän Yliopisto
Tietotekniikan laitos

Sisällysluettelo

1. Johdanto....1
2. Sphinxin asennus....1
3. Sphinx-projektin luonti....1
4. ReST-merkkauskieli....2
    4.1 Etusivu ja sisällysluettelo....2
    4.2 Tärkeimmät direktiivit dokumentointia kirjoitettaessa....2
    4.3 Viittaukset ja referenssit....3
5. Luokkadokumentaation luonti autodoc-moduulilla....3
    5.1 Dokumentoitavien moduulien määrittely....3
    5.2 Tärkeimmät direktiivit luokkadokumentaatiossa....3
    5.3 Funktioiden ja metodien kommentointi lähdekoodissa....4
6. Dokumentaation generointi....4
Liite: Python-moduulin kommentointi ReST-kielellä....5

1. Johdanto

Sphinx on työkalu Python-luokkadokumentaation luomiseen. Dokumentaation luomiseen käytetään ReST-kieltä (reStructuredText), joka muistuttaa tyyliltään hieman mm. wiki-alustojen käyttämää kieltä. Sphinxillä voidaan paitsi luoda yksinkertaisia dokumentaatiosivuja, myös generoida automaattisesti lähdekooditiedostoista luokkadokumentaatiota pydocin [4] tapaan. Tulosdokumentti on normaalisti HTML-muodossa, mutta myös LaTeX-muotoisten dokumenttien tuottaminen on mahdollista.

Ohje neuvoo lukijaa Sphinx-dokumentointityökalun [2] ja ReST-merkkauskielen [3][5] tärkeimpien ominaisuuksien käytössä.

2. Sphinxin asennus

Sphinxin toiminnan vaatimuksena on toimiva Python-tulkki, josta suositellaan versiota 2.6.

Linux-alustoilla helpoin tapa asentaa Sphinx on asentaa paketti suoraan käytetyn distribuution pakettienhallinnasta. **Windows-alustoilla** helpoin tapa asennukseen on easy_install-ohjelma, joka löytyy Setuptools-paketista [1]. Komento `easy_install -U Sphinx` asentaa Sphinxin ja kaikki sen tarvitsemat kirjastot. Kannattaa ottaa huomioon, että Sphinx vaatii 32-bittisen version Pythonista.

3. Sphinx-projektin luonti

Sphinx-projektilla tarkoitetaan hakemistorakennetta ja konfiguraatiotiedostoja, joiden mukaan projektin kääntämisen tuloksena syntyvä dokumentaatio rakentuu. Projektiin kuuluvat mm. ReST-dokumentit sekä mahdolliset dokumentaatioon sijoitettavat kuvat. Mikäli projektissa luodaan luokkadokumentaatio, voidaan myös lähdekoodi lisätä projektin hakemistoon, mutta se ei ole välttämätöntä.

Sphinx-projekti luodaan käyttämällä paketin mukana tulleita työkaluja. Projektille luodaan ensin hakemisto, jossa suoritetaan sovellus `sphinx-quickstart`. Työkalu on wizard-tyyppinen asennusohjelma, joka luo tarvittavat tiedostot annettujen vastausten pohjalta.

Pääasiallisesti asennusohjelman oletusvalinnat kelpaavat, mutta kannattaa vastata kyllä (*y*) seuraaviin asetuksiin:

- *Separate source and build directories (y/N) [n]* määrittää Sphinxin erottamaan ReST-dokumentit ja tuloksena syntyvän valmiin dokumentaation eri hakemistoihin.
- *autodoc: automatically insert docstrings from modules (y/N) [n]* määrittää Sphinxin käyttämään luokkadokumentaation Python-lähdekooditiedostoista automaattisesti luovaa lisäosaa.

Asennusohjelman ajamisen jälkeen löytyy projektihakemistosta muutamia tiedostoja ja alihakemistoja. Näistä tärkeimmät ovat seuraavat:

`Makefile` on tiedosto projektin kääntämiseen `make`-työkalulla.

`make.bat` on komentorivitiedosto projektin kääntämiseen Windows-alustoilla.

`source/conf.py` on dokumentaatioprojektin asetustiedosto.

`source/index.rst` on dokumentaatioprojektin sisällysluettelotiedosto.

`build/ on` hakemisto, jonne valmis tulosdokumentti luodaan.

4. ReST-merkkauskieli

Sphinx muodostaa dokumentaation ReST-kielellä [3] kirjoitettujen sääntöjen mukaan. Esimerkkejä ReST-dokumenttien luontiin kannattaa ottaa esimerkiksi Pythonin virallisesta luokkadokumentaatiosta [4]. Sivujen vasemmasta reunasta löytyy linkki *Show Source*, jolla alkuperäistä dokumentaatioa voi vilkaista.

Sphinx on monipuolinen työkalu dokumentaation laatimiseen. Tulosdokumentteihin voi liittää mm. kuvia ja linkkejä, sekä luokkadokumentaatiota Python-lähdekooditiedostoista. Luvussa 4 käsitellään ”yleistä” dokumentaatiota ja luvussa 5 luokkadokumentaatiota omina kokonaisuuksinaan.

4.1 Etusivu ja sisällysluettelo

Dokumentaatioprojektia luotaessa luotiin automaattisesti myös tiedosto `source/index.rst`, jossa määritellään dokumentaation etusivu. Tärkein etusivulla määriteltävä asia on sisällysluettelo, jonka luomiseen käytetään `toctree`-direktiiviä.

```
Project TestProject's documentation
===================================

Contents:

.. toctree::
   :maxdepth: 2

   about
   tutorial
   subsystems/templates

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

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

Esimerkissä kuvataan seuraavia asioita:

- Sisällysluettelon `toctree`-direktiivi määrittää hakemistopuun, jossa `about`- ja `tutorial`-kohdat viittaavat tiedostoihin `about.rst` ja `tutorial.rst` kansiossa `source/`. Rivi `subsystems/templates` taas on viittaus tiedostoon `templates.rst` kansiossa `source/subsystems/`. Parametri `:maxdepth:` määrittelee sisällysluettelon maksimisyvyyden.

- Viimeiset rivit ovat luvussa 4.3 kuvattuja referenssejä (`:ref:`) Sphinxin automaattisesti luomiin, staattisiin dokumentteihin.

4.2 Tärkeimmät direktiivit dokumentointia kirjoitettaessa

`.. toctree::` Direktiivi ilmoittaa sisällysluettelon alkamisesta. Sille voidaan antaa parametrina `:maxdepth: 2` ilmoittaen sisällysluettelon syvyyden.

`.. image:: directory/image.jpg` ilmoittaa kuvan sijainnin. Parametreina voidaan antaa

`:height:` kuvan korkeus pikseleinä,

`:width:` kuvan leveys pikseleinä, sekä

`:alt:` kuvan seliteteksti, mikäli kuvaa ei voida ladata tai teknisistä syistä johtuen näyttää.

4.3 Viittaukset ja referenssit

ReST-dokumenteista voidaan tehdä **viittauksia** toisiin dokumentteihin komennolla `:doc:`. Esimerkiksi komento `:doc:`Moduuli <package/module>`` luo viitteen hakemiston `package` rst-tiedostoon `module`, ja määrittää muodostuvan hyperlinkin nimeksi `Moduuli`.

Viittauksiin verrattuna **referenssit** ovat yksinkertaisempia. Niillä voidaan viitata suoraan dokumenttiin, ja muodostuvan linkin nimeksi tulee kohdedokumentin otsikko. Referenssi voidaan luoda komennolla `:ref:`, esimerkiksi `:ref:`rstfile``

5. Luokkadokumentaation luonti autodoc-moduulilla

Autodoc-moduuli rakentaa ReST-kielellä kommentoiduista Python-tiedostoista luokkadokumentteja.

5.1 Dokumentoitavien moduulien määrittely

Autodocin käyttö on suhteellisen yksinkertaista. Ensin ilmoitetaan `conf.py`-tiedostossa ohjelman lähdekoodin **hakemistopolku.** Kannattaa huomata, että jokainen alikansio on ilmoitettava erikseen erimerkiksi alla esitetyllä tavalla:

```
sys.path.append(os.path.abspath('../src/'))
sys.path.append(os.path.abspath('../src/paketti/'))
```

Python-moduuli määritetään dokumentoitavaksi luomalla sille oma rst-tiedosto, ja antamalla tiedostossa `automodule`-direktiivi. Esimerkiksi direktiivi `..automodule:: package.module` ilmoittaa, että halutaan dokumentoida paketin `package` sisältämä moduuli `module`. Laittamalla useampia `automodule`-direktiivejä allekkain voidaan samaan tiedostoon luoda useampien moduulien dokumentaatio.

5.2 Tärkeimmät direktiivit luokkadokumentaatiossa

`.. autoclass:: classname` ilmoittaa dokumentoidavaksi luokan `classname`. Sen parametrit ovat seuraavat

`:members:` ilmoittaa dokumentoitavaksi myös luokan kaikki jäsenet.

`:inherited-members:` ilmoittaa dokumentoitavaksi myös luokan perityt jäsenet.

`.. autofunction:: functionname` ilmoittaa dokumentoitavaksi funktion `functionname.`

`.. automethod:: methodname` ilmoittaa dokumentoitavaksi metodin `methodname.`

`.. moduleauthor:: author name` ilmoittaa moduulin tekijän.

`.. data:: variable` ilmoittaa dokumentoitavaksi muuttujan `variable.`

`.. automodule:: module.name` ilmoittaa dokumentoitavan moduulin nimen.

5.3 Funktioiden ja metodien kommentointi lähdekoodissa

Metodien ja funktioiden dokumentointi onnistuu kirjoittamalla lähdekoodeissa heti metodin esittelyn jälkeen kommentoitu osio. Siinä ilmoitetaan **määritteillä** `:description:`, `:param:`, `:return:`, `:type:` ja `:rtype:` metodin kuvaus, parametrit, parametrin tyyppi, palautusarvo ja palautusarvon tyyppi.

Myös **luokalle** voidaan antaa sen esittelyn jälkeen kommentissa **kuvaus**, mutta tällöin metodissa `__init__` annettu kuvaus piiloutuu, ellei erikseen määritellä sitä näytettäväksi direktiivillä `..automethod::`. Liite sisältää esimerkin lähdekoodin dokumentoinnista luokkien osalta.

6. Dokumentaation generointi

Dokumentaatio voidaan generoida sanomalla komentorivillä `make html`. Se luo HTML-version dokumentaatiosta kansioon `build/html/`.

Sphinxillä voi myös luoda LaTeX-dokumentteja, joista voidaan muodostaa PDF-dokumentti. Komennon `make latex` jälkeen suoritetaan hakemistossa `build/latex/` komento `make all-pdf`.

Liite: Python-moduulin kommentointi ReST-kielellä

```
#coding=utf-8
"""
================================
 :mod:`naakka` -- Naakka moduuli
================================

Kommentin alussa kuvataan moduulin sisältöä ja käyttöä.

MIT license. For details see LICENSE.TXT or website
  http://www.opensource.org/licenses/mit-license.php

.. autoclass:: Naakka
   :members:
   :inherited-members:

.. moduleauthor:: Tuomas Virtanen
"""

import sys

class Naakka():
  """
  :description: Yleiskuvaus luokasta.

  .. automethod:: __init__
  """

  def __init__(self, parametri):
    """
    :description: Metodin kuvaus.
    :param parametri: Parametrin kuvaus
    :type parametri: String
    """
    pass

  def sano_kvaak(self):
    """
    :description: Sanoo ”Kvaak”.
    :return: Palauttaa arvon 0.
:rtype: Integer
"""
return 0
```

Lähteet

1. Eby, Phillip, 2010, ”Setuptools”, saatavilla HTML-muodossa http://pypi.python.org/pypi/setuptools.

2. Brandl, Georg, 2010, ”Sphinx”, saatavilla HTML-muodossa http://pypi.python.org/pypi/Sphinx.
3. Brandl, Georg, 2010, ”reStructuredText Primer”, saatavilla HTML-muodossa http://sphinx.pocoo.org/rest.html.
4. Python Software Foundation, 4.10.2010, ”Python 2.6.5 documentation”, saatavilla HTML-muodossa http://docs.python.org/.
5. Docutils Project, 1.9.2010, ”reStructuredText”, saatavilla HTML-muodossa http://docutils.sourceforge.net/rst.html.