Pràctica 0: Gestió de documentació

Tecnologia de la Programació — iTIC

Aleix Llusà-Serra Sebastià Vila-Marta Marta Tarrés-Puertas

February 5, 2021

Contents

1 Organització

L'objectiu d'aquesta pràctica és aprendre a documentar convenientment un projecte de programari usant eines específiques. La pràctica permetrà conèixer els aspectes més problemàtics de la gestió de documentació tècnica i, específicament del programari, així com la manera de solucionar-los aplicant eines de generació de documentació.

Particularment caldrà aprendre:

- El paper dels docstrings en la documentació de programes `Python`.
- El format àgil d'escriptura de documentació `RestructuredText` i la forma en com s'utilitza.
- L'eina de manteniment i generació de documentació tècnica `Sphinx` i la manera com s'usa per a mantenir la documentació d'un projecte.

Com a projecte d'exemple s'usaran alguns dels mòduls que vàreu construir en el projecte de curs d'INF. La fita serà documentar aquests mòduls usant les eines que són objecte d'aquesta pràctica.

1.1 Condicions

- La pràctica està calibrada per a ésser treballada en equips de dos/tres persones.
- La durada de la pràctica és de 2 setmanes.

1.2 Lliurables

Cal lliurar mitjançant l'activitat apropiada d'Atenea:

1. El fitxer `pdf` resultat de la tasca 3.2.
2. Un tarfile que contingui la documentació generada en la tasca 6.3.

1.3 Material necessari

Per a fer la pràctica necessitareu un conjunt d'eines i un exemple de documentació generada amb `Sphinx`.

Les eines necessàries les podeu instal.lar en el vostre computador com a paquets del sistema operatiu. En el cas de GNU/Debian i equiparables cal que instal.leu els paquets `rst2pdf` i `python-sphinx`.

> TASCA PRÈVIA 1.1 Instal.leu en el vostre computador els paquets anteriors. En els computadors de laboratori ja hi els hi trobareu instal.lats. Per fer-ho, com a usuari `root` executeu les ordres:
>
> ```
> $ apt-get install rst2pdf python-sphinx
> ```

A OCW iTIC, dins l'apartat de l'assignatura, trobareu un exemple de documentació i petit resum sobre `Sphinx`.

> TASCA PRÈVIA 1.2 Obteniu el exemple de documentació `Sphinx` de l'OCW iTIC.

2 Introducció

En la producció de programari i en l'àmbit tecnològic en general la documentació tècnica té un paper fonamental. En aquesta pràctica estudiarem el problema de la documentació del programari i també algunes eines com `Sphinx`, que simplifiquen aquesta feina.

Entenem per documentació tècnica d'un programari tot aquella documentació que descriu amb precisió com és aquest programari i com se'n fa ús quan el lector target és una persona de perfil tècnic. Entre altres coses pot descriure:

- L'arquitectura del programa o mòdul. És a dir de quins blocs està format, quina funcionalitat tenen i quines interrelacions hi ha entre ells.
- Les funcions de cada mòdul i la seva especificació.
- Documentació d'utilització incloent doctests i altres exemples d'ús.

- Principis de disseny que s'han utilitzat.
- Referències a altres documents importants en relació a aquest software.

La documentació resultant ha de ser consultable sobre una diversitat de formats. Aquests inclouen formats habituals com ara:

- `HTML`.
- `pdf`.

3 Com s'escriu amb `RestructuredText`

`RestructuredText` és un format d'escriptura dels anomenats "àgils", que permeten escriure document d'una forma molt senzilla però eficient. Aquests formats són hereus de la tradició dels wiki's.

Tots els formats àgils s'escriuen en un fitxer de text convencional amb un editor de textos com `emacs`. Després, els fitxers es poden traduir a formats de sortida habituals com podria ser `pdf`.

En el cas de `RestructuredText`, una de les seves característiques principals és que, el fitxer de text original "imita" en certa manera com serà el resultat final. D'aquesta manera resulta més senzill treballar-hi.

Escriure amb `reST`, és senzill. Amb l'editor proveu de crear un fitxer de nom `document.rst` amb aquest contingut:

```
================
Títol principal
================

:author: Pere Pi

Apartat
=======

Aquest és el primer apartat. Escriviu-lo així com el veieu aquí.
Fixeu-vos que, com en Python, les tabulacions són molt importants atès
que determinen l'estructura del document.

Subapartat
----------

També podeu escriure enumeracions i llistes convencionals usant una
sintaxi molt natural com la que segueix:

  * Primer element
  * Segon element
    - Primera subllista
    - Segona subllista
  * Tercer element

Una altra facilitat molt corrent és la de poder inserir codi o exemples
de programació d'una forma senzilla i elegant com per exemple:

.. code-block:: python
```

```
def exemple(a,b):
    """
    Una funció per exemplificar reST
    """
    x = a
    for i in b:
        if i % 2 == 0: x += a
    return x
```

Una vegada hàgiu escrit el contingut anterior, processeu el fitxer `document.rst` fent:

```
$ rst2pdf document.rst
$ ls -l document.*
```

Haureu comprovat que s'ha creat un fitxer de nom `document.pdf` que ara podeu veure fent servir el visualitzador de l'escriptori.

Es important que practiqueu l'escriptura amb `reST` unes quantes vegades per acostumar-vos a la seva sintaxi. La documentació d'`Sphinx`, conté un apartat en que es resumeix la forma d'escriure de `reST`.

TASCA 3.1 Feu-vos un petit xuletari amb les principals construccions de `reST` per tenir-lo a mà quan es treballa.

TASCA 3.2 Escriviu una documentació d'un parell de pàgines explicant com funciona el tipus de dades `list` de `Python` i les maneres en com es pot iterar sobre els elements d'una llista. Es valorarà la qualitat del resultat (títols, autors, dates, contingut, apartats, sumari, exemples, etc.).

4 Com s'escriu documentació amb Sphinx

4.1 Introducció

`Sphinx` és una eina de documentació específicament pensada per a documentar programari tot i que també pot ser aplicada en altres àmbits. Es una aplicació escrita en `Python` i extensible en base a plugins (i.e. amb plugins es pot estendre la seva funcionalitat).

En essència permet construir i mantenir la documentació associada a un projecte de programari a partir de dues fonts d'informació:

1. Un conjunt de fitxers en format `reST`.
2. El codi font del propi projecte i els docstrings que aquest contingui.

El resultat us semblarà familiar atès que és l'eina emprada per mantenir la documentació de `Python`, que esteu acostumats a usar.

Per usar l'eina cal seguir un esquema de tres passos:

1. Establir l'entorn de treball.
2. Escriure la documentació i definir l'entorn.
3. Generar la documentació resultant.

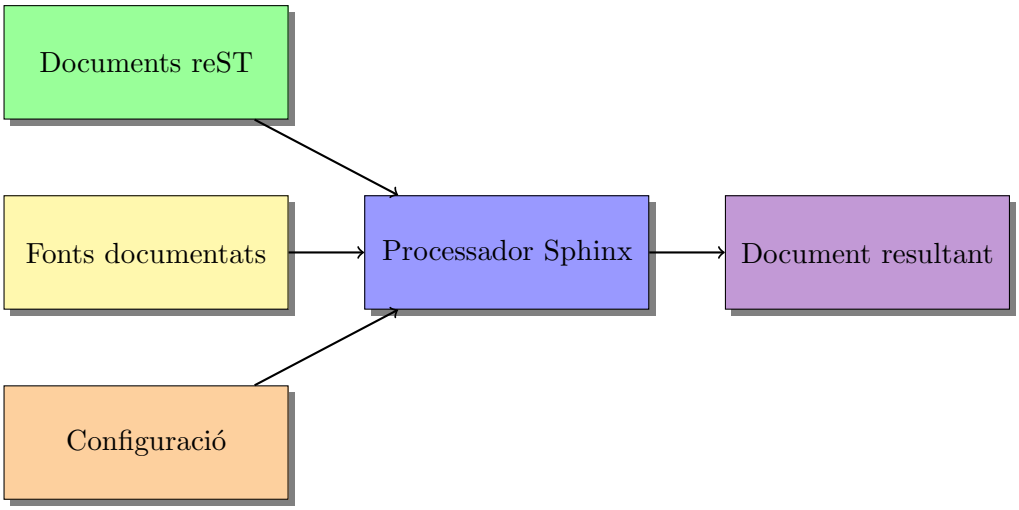

Figure 1: Diagrama de flux del procés amb Sphinx

Com podeu veure en essència es tracta d'un *generador*, per això sovint aquest tipus d'eines se les anomena *generadors de documentació*. La documentació generada pot estar en diversos formats, segons convingui. Per exemple en `HTML` o en `PDF`. Aquesta forma de treball és molt convenient atès que els diferents formats de sortida es generen a partir d'un únic original amb el conseqüent estalvi de temps i garantia de coherència. La figura 1 mostra el procés que cal seguir per treballar amb Sphinx i les dades que intervenen.

4.2 Com establir l'entorn de treball

Per establir l'entorn de treball usem la comanda `sphinx-quickstart` que de manera interactiva va preguntant com volem configurar l'entorn de generació de la documentació. Una sessió típica pot tenir aquest aspecte:

```
S'ha iniciat l'execució de script a dj 03 feb 2011 20:40:23 CET
sebas@bubinga:/tmp$ sphinx-quickstart
Welcome to the Sphinx 1.0.7 quickstart utility.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Enter the root path for documentation.
> Root path for the documentation [.]: doc

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "_build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/N) [n]: y

Inside the root directory, two more directories will be created; "_templates"
for custom HTML templates and "_static" for custom stylesheets and other static
files. You can enter another prefix (such as ".") to replace the underscore.
```

```
> Name prefix for templates and static dir [_]: .

The project name will occur in several places in the built documentation.
> Project name: Reductio
> Author name(s): S. Vila−Marta

Sphinx has the notion of a "version" and a "release" for the
software. Each version can have multiple releases. For example, for
Python the version is something like 2.5 or 3.0, while the release is
something like 2.5.1 or 3.0a1. If you don't need this dual structure,
just set both to the same value.
> Project version: 1.0
> Project release [1.0]: 1.0b2

The file name suffix for source files. Commonly, this is either ".txt"
or ".rst". Only files with this suffix are considered documents.
> Source file suffix [.rst]:

One document is special in that it is considered the top node of the
"contents tree", that is, it is the root of the hierarchical structure
of the documents. Normally, this is "index", but if your "index"
document is a custom template, you can also set this to another filename.
> Name of your master document (without suffix) [index]:

Sphinx can also add configuration for epub output:
> Do you want to use the epub builder (y/N) [n]:

Please indicate if you want to use one of the following Sphinx extensions:
> autodoc: automatically insert docstrings from modules (y/N) [n]: y
> doctest: automatically test code snippets in doctest blocks (y/N) [n]:
> intersphinx: link between Sphinx documentation of different projects (y/N) [n]:
> todo: write "todo" entries that can be shown or hidden on build (y/N) [n]:
> coverage: checks for documentation coverage (y/N) [n]:
> pngmath: include math, rendered as PNG images (y/N) [n]:
> jsmath: include math, rendered in the browser by JSMath (y/N) [n]:
> ifconfig: conditional inclusion of content based on config values (y/N) [n]:
> viewcode: include links to the source code of documented Python objects (y/N) [n]: y

A Makefile and a Windows command file can be generated for you so that you
only have to run e.g. 'make html' instead of invoking sphinx−build
directly.
> Create Makefile? (Y/n) [y]:
> Create Windows command file? (Y/n) [y]: n

Finished: An initial directory structure has been created

You should now populate your master file doc/source/index.rst and
create other documentation source files. Use the Makefile to build
```

```
the docs, like so:

$ make builder

where "builder" is one of the supported builders, e.g. html, latex or linkcheck.

sebas@bubinga: /tmp$
```

Aquesta comanda crea una estructura de directoris dins de `doc` preparada per a rebre la documentació. El directori principal és `doc/source`, que és el que contindrà els fitxers `reST` amb el gruix de la documentació original o font. El directori `doc/build` contindrà la documentació generada en el format de sortida.

4.3 Com escriure la documentació

4.3.1 L'estructura del directori `source`

La documentació s'escriu en el directori `source`. En aquest directori hi ha dos fitxers fonamentals:

conf.py Aquest fitxer és un mòdul `Python` i conté la configuració que usarà `Sphinx` per processar els nostres documents.

index.rst Aquest és un fitxer de text en format `reST` que defineix l'índex de la documentació. En seu contingut s'indica la col.lecció de fitxers que conformen la documentació i l'ordre com han de sortir en l'índex del document.

A banda dels fitxers anteriors el directori `source` també conté els fitxers en format `reST` que conformen la resta de la documentació. Així doncs, per escriure documentació cal:

1. Escriure-la en un fitxer en format `reST`.
2. Declarar el fitxer en l'índex (`index.rst`).

4.3.2 La forma d'afegir un document

Suposem que volem afegir un nou capítol a la nostra documentació. Aleshores caldrà crear un nou fitxer en el directori `source`. Suposem que en diem `cap1.rst`. Aquest fitxer conté un capítol escrit en format `reST` com ara el següent:

```
===========
Introducció
===========

En la producció de programari i en l'àmbit tecnològic en general la
documentació tècnica té un paper fonamental. En aquesta pràctica
estudiarem el problema de la documentació del programari i també
algunes eines com **Sphinx** que simplifiquen aquesta feina.

Entenem per documentació tècnica d'un programari tot aquella
documentació que descriu amb precissió com és aquest programari i com
se'n fa ús quan el lector target és una persona de perfil tècnic.
Entre altres coses pot descriure:

  * L'arquitectura del programa o mòdul. És a dir de quins blocs està
```

```
    format, quina funcionalitat tenen i quines interrelacions hi ha
    entre ells.

  * Les funcions de cada mòdul i la seva especificació.

  * Documentació d'utilització incloent doctests i altres exemples
    d'ús.

  * Principis de disseny que s'han utilitzat.

  * Referències a altres documents importants en relació a aquest
    software.
```

A continuació cal modificar el fitxer `index.rst` i citar el fitxer anterior. El resultat ha de ser semblant al que segueix:

```
.. Prova Sphinx documentation master file, created by
   sphinx-quickstart on Tue Feb  1 10:00:29 2011.
   You can adapt this file completely to your liking, but it should at least
   contain the root {}‘toctree{}‘ directive.

================================
Documentació tècnica amb Sphinx
================================


Índex
=====

.. toctree::
   :maxdepth: 2

   cap1
```

4.4 Com generar la documentació en fomat HTML

Generar la documentació en format `HTML` és senzill. Simplement cal, des del directori arrel de la documentació, executar la comanda:

```
$ make html
```

Aquesta comanda usa l'eina `make`, per reconstruir un conjunt de pàgines `HTML` correctament enllaçades en el directori `build/html`.

Podeu veure el resultat arrencant un navegador i citant el fitxer `build/html/index.html`. Us apareixerà la documentació que heu escrit prèviament adequadament enllaçada.

4.5 Documentació de codi font

`Sphinx` és una eina de documentació específicament pensada per a documentar programari. Per a facilitar aquesta tasca, `Sphinx` té eines que analitzen els mòduls `Python` i n'obtenen automàticament la documentació associada. Per a entendre com s'usen aquestes eines, seguiu l'exemple d'Sphinx..

En el directori `pr1-sphinx/test/build/html/` hi trobareu la documentació construïda amb `Sphinx` i en el directori `pr1-sphinx/test/source/` hi trobareu els mòduls `Python` que s'estan

documentant. Consulteu els dos directoris simultàniament per a entendre com es documenten els mòduls `Python` de manera automàtica mitjançant les directives d'`Sphinx`.

L'exemple de l'assignatura s'ha construït usant una extensió d'`Sphinx` que automàticament insereix el codi font en les pàgines que documenten els mòduls `Python`. Aprofiteu aquesta funcionalitat per tal de navegar per les dues estructures amb més agilitat.

5 Automatització de documentació .html i .pdf de codi font (mòduls .py)

1. Abans d'establir l'entorn de treball amb *sphinx-quicksart*, col·loqueu-vos en el vostre directori de treball, i creeu dos directoris: el directori *src* i el directori *test*.

2. Dins el directori *src* anterior, copieu-hi els mòduls .py dels quals voleu que es generi l'autodocumentació. Per simplificar aquest exemple, suposem que el fitxer s'anomena *prova.py*

3. Situeu-vos en el directori test. En aquest directori executeu *sphinx-quickstart* i estabiu l'entorn de treball seguint les pautes de l'enunciat de la pràctica 1. Automàticament, en el directori *test* se us ha creat el directori *build*, el directori *source* i un fitxer *Makefile*.

4. El primer que cal és indicar en quin directori es troba el mòdul python a documentar. Col.loqueu-vos en el directori *source* i editeu el fitxer *conf.py* i afegiu la següent línia:

   sys.path.insert(0, os.path.abspath('../../src'))

   just abans del text: –General configuration

5. El segon que cal fer és incorporar la informació per la generació automàtica de la documentació. Editeu un fitxer de nom *prova.rst* amb el següent contingut:

   ```
   .. automodule:: prova
      :members:
   ```

6. A continuació editeu el fitxer *index.rst* i afegiu-hi *prova*

7. Situeu-vos ara en el directori *test/doc* i escriviu *make html*. Veureu que en el directori *build/html* s'ha creat correctament la documentació en .html del mòdul. Navegueu amb el fitxer *index.html* i proveu-ho.

8. Si es vol generar l'autodocumentació en pdf, *make latex*, es crea el subdirectori *build/latex* i aquí cal fer *make all-pdf* (Generació automàtica document pdf)

9. Recordeu que teniu a la vostra disposició l'exemple complet Exemple sphinx de l'ocw itic. Preneu per exemple, el mòdul bdu.py i repetiu els passos anteriors.

6 Documenteu els mòduls del projecte d'INF

Tasca 6.1 Obteniu una versió dels mòduls del projecte d'INF. Podeu usar la vostra versió o bé la d'algun altre company. Trieu-ne un subconjunt coherent de 4 mòduls.

Tasca 6.2 Editeu els mòduls i completeu els docstrings de cada funció amb la informació adequada en format `reST`. Feu el mateix amb el docstring del mòdul.

TASCA 6.3 Creeu un entorn `Sphinx` específic destinat a construir una documentació per als dos mòduls anteriors. Aquesta documentació ha de contenir una descripció general del projecte així com una documentació dels seus (dos) mòduls. La documentació dels mòduls cal obtenir-la automàticament dels docstrings corresponents.

Es valorarà:

- L'organització i claredat de la documentació.
- La correcció de redacció, sintàctica i ortogràfica.
- Que la documentació estigui escrita en anglès (tot i no ser obligatori).

[1] Georg Brandl i project contributors. Sphinx. Python documentation generator. Ang. 2013. `http://sphinx-doc.org/`
(vis. 26-02-2015).

[2] Python project contributors. Python documentation. Ang. Ver. 2.7.3. Python Software Foundation. 2015. `http://sphinx-doc.org/`
(vis. 26-02-2015).

[3] Sebastià Vila-Marta i Aleix Llusà-Serra. Exemple de documentació Sphinx. Dept. DiPSE. Universitat Politècnica de Catalunya. 2011.
`http://itic.cat/assignatures/tecpro/laboratori-material/exemple-sphinx`
(vis. 26-02-2015).

[4] Wikipedia contributors. Make. Ang. Wikipedia, The Free Encyclopedia. 2015.
`http://en.wikipedia.org/wiki/Make_(software)`
(vis. 26-02-2015).