libro10min Documentation

Release 0.1.0

Daniele Zambelli

December 03, 2016

Contents

Contents:

CHAPTER 1

Introduzione

In questo documento cerco di descrivere come si può realizzare un libro con l'uso di due strumenti potenti e flessibili:

- ReStructuredText un linguaggio di Markup che Sphinx può compilare producendo documenti in diversi formati.
- Mercurial un programma che permette la gestione di codice che può essere depositato in uno dei tanti servizi presenti in rete io userò: `bitbucket.org`.

Il ciclo di produzione consiste in alcuni passi:

1. Scrivere il contenuto del testo usando i marcatori adatti, con un qualunque editor di testo.
2. Compilare i file `.rst`.
3. Controllare il risultato.

Tutti i programmi utilizzati sono liberi, si possono scaricare dalla rete e installare sul proprio sistema. Per avere informazioni basta ricercare in internet, una delle parole:

- ReStructuredText
- Sphinx
- Mercurial

Ma cosa possiamo mettere nel nostro testo?

CHAPTER 2

Contenuti

2.1 Sorgenti

Il contenuto del nostro testo viene suddiviso in vari file con estensione `.rst`. I nomi di questi file vanno scritti in `index.rst`. Quindi creiamo un file per ogni capitolo:

```
00intro.rst
01contenuti.rst
02strumenti.rst
```

e modifichiamo `index.rst` in modo che la sezione `contents` diventi:

```
Contents:

.. toctree::
  :maxdepth: 2

  00intro
  01contenuti
  02strumenti
```

Salviamo il file `index.rst` e incominciamo a riempire il file `intro.rst`. In generale è una buona idea ricompilare spesso e controllare il risultato in modo da identificare più facilmente eventuali errori. Quindi:

1. Salvare i file modificati,
2. dare il comando: `make latexpdf`,
3. controllare il risultato ottenuto.

I tre punti precedenti costituiscono il ciclo di lavoro: modificare e salvare i file `.rst`, compilare i sorgenti, controllare il risultato, modificare...

2.2 Struttura

È possibile strutturare il testo in capitoli, paragrafi e sotto paragrafi e introdurre svariati elementi grafici.

I diversi livelli della struttura sono definiti dal tipo di sottolineatura dei titoli:

```
Capitolo
========

Paragrafo
```

```
---------

Sotto paragrafo
...............

Testo normale
```

2.3 Immagini

Possiamo inserire immagini separate dal testo:

Fig. 2.1: Libri

Magari aggiungendo semplicemente la scala di rappresentazione o una didascalia come nel caso della figura 2.1.

Le immagini possono anche essere inserite all'interno di un paragrafo aggiungendo una serie di parametri opzionali che permettono di specificare diverse caratteristiche della figura come le dimensioni, un testo alternativo, l'allineamento...

Bisogna stare attenti, che, passando attraverso LaTex, le figure sono messe dove vanno messe non dove vorremmo noi. E questo può, a volte, portare uno certo scompiglio durante la stesura di un testo.

Fig. 2.2: Altri Libri

2.4 Tabelle

Facilmente in un testo scientifico devono essere inserite delle tabelle. Sphinx permette di descriverle in uno di questi modi:

riga 1, colonna 1	colonna 2	colonna 3	colonna 4
riga 2	Fusione di due celle		altro
riga 3			

A	B	A e B
False	False	False
True	False	False
False	True	False
True	True	True

La faccenda più rognosa: bisogna stare attenti al numero di caratteri usato per definire le tabelle che deve corrispondere nelle varie righe.

2.5 Formule

Se siamo, per caso interessati a scrivere qualcosa di legato alla matematica avremo bisogno di scrivere delle definizioni e delle formule. Ad esempio se siamo arrivati al capitolo delle potenze potremo scrivere:

Definizione Dati due numeri naturali a e b, con b>1 il primo detto base, il secondo esponente, la potenza di a con esponente b è il numero p che si ottiene moltiplicando fra loro b fattori tutti uguali ad a. Si scrive e si legge "a elevato a b uguale a p".

Per esempio $5^3 = 5 \times 5 \times 5 = 125$

$$5^3 = \underbrace{5 \times 5 \times 5}_{3volte} = 125$$

ReStructuredText ci mette a disposizione tutto l'armamentario di LaTex per produrre tutte le formule possibili e immaginabili.

CHAPTER 3

Strumenti

Abbiamo visto come scrivere il libro, come inserire immagini e tabelle, ma abbiamo detto ben poco sugli strumenti che trasformano il nostro testo in un libro. In questo capitolo vedremo come installare tutto il necessario.

3.1 Sphinx

Il programma che trasforma il file `.rst` in LaTex e poi `.pdf` è Sphinx bisogna installarlo. Il sito da cui attingere è:

http://sphinx.pocoo.org/

Da lì si può accedere alla documentazione e Conviene installarlo attraverso i repository della propria distribuzione.

Una volta installato Sphinx vediamo le operazioni da fare per avviare un nuovo progetto:

1. Creare una directory nel mio caso l'ho chiamata: `libro10minuti`.
2. Aprire un terminale in questa directory e dare il comando:

```
$> sphinx-quickstart
```

#. Rispondere in modo sensato alle domande che `sphinx-quickstart` ci pone, nel dubbio conviene accettare la proposta predefinita.

Terminato questa fase possiamo creare il nostro testo in formato `pdf` con il comando:

```
$> make latexpdf
```

Nella directory `./build/latex/` ci sarà il file `libro10minuti.pdf`... Un po' scarno, ma completo. Ora lo si può riempire seguendo le indicazioni del capitolo precedente.

3.2 Bitbucket

`bitbucket.org` ci mette a disposizione un repository che può ospitare il nostro progetto, e ci permette di condividerlo con altri. Quello proposto da me è uno dei tanti presenti nella rete e sono utilizzati principalmente per gestire progetti software.

Per prima cosa dobbiamo registraci e poi creare un nuovo progetto seguendo le istruzioni del sito. Io mi sono registrato con il nomignolo "zambu" e questo progetto si chiama: "libro10minuti". L'indirizzo a cui si trova questo lavoro è quindi:

```
bitbucket.org/zambu/libro10minuti
```

Essendo pubblico, chiunque può accedervi, navigare il sorgente scaricare un file o tutto in blocco in un formato compresso o clonare il progetto e contribuirvi.

3.3 Mercurial

Il terzo strumento che utilizziamo per il nostro testo è un programma di gestione di versione. È uno strumento usato nella produzione di software e fornisce strumenti per gestire progetti in continua evoluzione e realizzati a più mani. Dobbiamo installarlo sempre usando gli strumenti messi a disposizione del nostro sistema operativo. Il sito di riferimento è:

mercurial.selenic.com

Anche in Mercurial c'è un programma che facilita l'inizio di un nuovo progetto. Restando nel terminale all'interno della directory creata per il nostro lavoro, diamo il comando:

```
$> hg init
```

Questo comando non ci chiede né ci dà molte informazioni e, apparentemente non produce nessun effetto. Se però diamo il comando:

```
$> ls -a
```

vediamo che è stata creata una directory nascosta: `.hg`.

A questo punto, per utilizzare un repository, bisogna aggiungere a mano alcuni file. Entrati nella directory `.hg`, creiamo il file di testo `hgrc` e dentro questo file scriviamo qualcosa che assomigli a:

```
[paths]
default = https://zambu@bitbucket.org/zambu/libro10minuti

[ui]
username = Daniele Zambelli <daniele.zambelli@gposta.com>
verbose = True
```

A questo punto possiamo già lavorare con i nostri strumenti ma io non voglio intasare il mio repository con un mucchio di file temporanei o creati automaticamente ma voglio che vengano presi in considerazione solo i file essenziali del mio lavoro. Quindi ritorno nella directory del progetto e creo il file nascosto `.hgignore` con un contenuto simile a:

```
# use glob syntax.
syntax: glob

*~
*.backup

build/*
```

Queste righe dicono semplicemente a Mercurial di non considerare i file che terminano con una tilde o con l'estensione `.backup` o che si trovano nella directory `build`.

A questo punto Mercurial si assume l'incarico di tracciare tutte le modifiche apportate al nostro progetto. Quando riteniamo di mettere un punto al lavoro svolto, dopo alcuni cambiamenti significativi o semplicemente a fine giornata,diamo il comando:

```
$> hg commit -m "<un commento sensato>"
```

E quando vogliamo condividere nel repository le modifiche apportate:

```
$> hg push
```

Procuratici questi strumenti vediamo come utilizzarli.

3.4 Ciclo di lavoro

Realizzare un progetto significa modificare un progetto si può partire da uno vuoto e la modifica consisterà principalmente nel aggiungere o partire da uno già realizzato e quindi la modifica consisterà principalmente nel cambiare. In ogni caso non conviene mai cambiare molte parti senza controllare il risultato: è molto più semplice affrontare un errore-problema alla volta piuttosto che trovarsi di fronte ad un lungo elenco.

Il tipico ciclo di lavoro è costituito da due cicli annidati:

1. Modifico il progetto
   (a) modifico un file `.rst`
   (b) compilo i sorgenti con `make latexpdf`
   (c) controllo il risultato e ritorno al punto (a)
2. Fisso le modifiche apportate con: `hg commit -m "<commento>"`
3. Invio al repository le modifiche: `hg push` e ritorno al punto 1.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search