

Guida agli script Automatizzati

| | |
|---------------------|---|
| Approvazione | Daniele Penazzo |
| Redazione | Daniele Penazzo |
| Verifica | Marco Giollo |
| Stato | Approvato |
| Uso | Interno |
| Destinato a | Commandline Team Prof. Tullio Vardanega Prof. Riccardo Cardin |

Descrizione

Questo documento contiene una breve guida a tutti gli script atti a velocizzare la produzione di documenti

Contenuto

| | | |
|---|--|---|
| 1 | Changelog | 2 |
| 2 | Introduzione | 3 |
| 3 | documenter.py - Creazione di documenti da template | 3 |
| 4 | bookKeeper.py | 5 |
| 5 | Altre funzionalità di documenter.py | 6 |
| 6 | glossarizer.py - Creazione di voci di glossario | 7 |

1 Changelog

| Versione | Data | Nominativo | Ruolo | Descrizione |
|----------|------------|-----------------|--------------|---------------------------|
| 1.0.0 | 2017-12-08 | Daniele Penazzo | Responsabile | Approvazione ed Emissione |
| 0.1.0 | 2017-12-08 | Marco Giollo | Verificatore | Verifica del documento |
| 0.0.1 | 2017-11-26 | Daniele Penazzo | Responsabile | Prima redazione |

Tabella 1: Changelog di questo documento

2 Introduzione

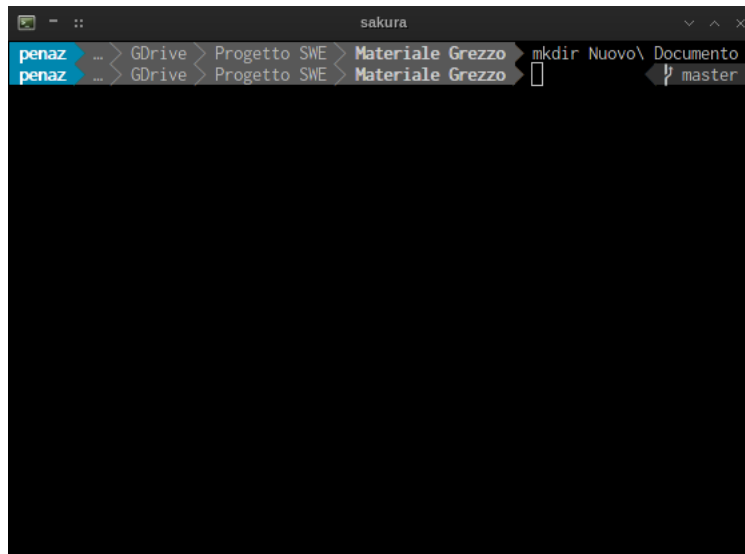
In questo documento verranno presentati gli script atti alla creazione e gestione di alcune fasi della documentazione. Tutti gli script richiedono che sia installato Python, versione 3.4 o superiore.

3 documenter.py - Creazione di documenti da template

Dipendenze: Python 3.4+, pdflatex (per la ricompilazione dei documenti), latexmk (per la live preview)

Questo script ha la funzione di copiare nella cartella corrente il template di documento e modificarlo secondo alcuni parametri, in modo da velocizzare la creazione del documento stesso.

Iniziamo creando una nuova cartella, che conterrà il documento che creeremo



Ora avviamo lo script `documenter.py`, con un comando che cambia a seconda del sistema operativo usato.

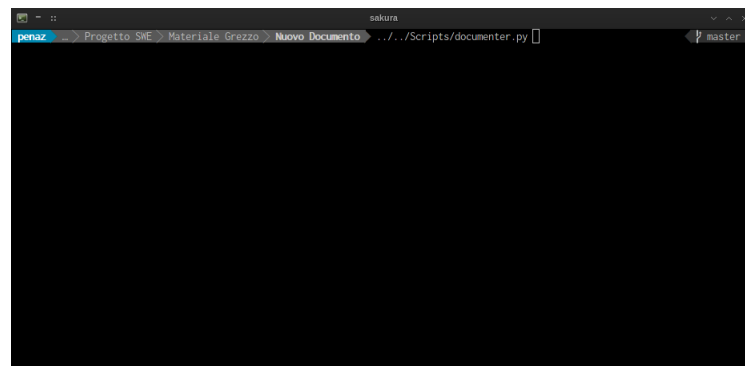
Linux e Unix-Like Lo script può essere avviato direttamente senza alcun comando particolare, usando una coppia di punti ("..<") per ogni cartella padre, fino ad arrivare a poter esplorare la cartella "Script".

Ad esempio: `../../Scripts/documenter.py`

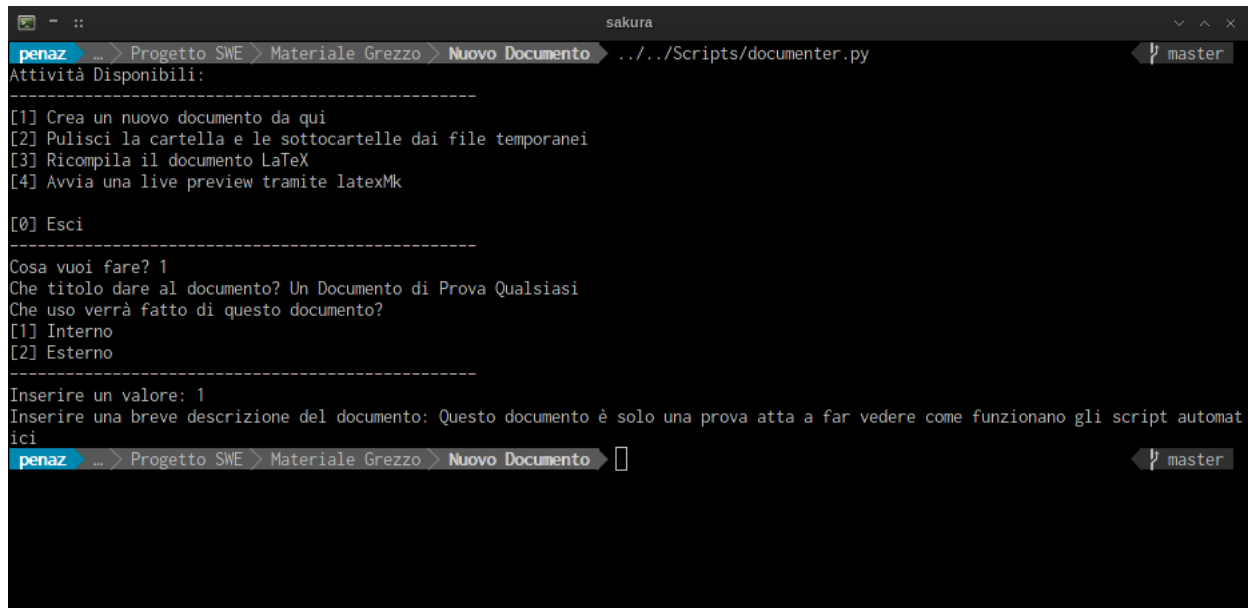
Windows Con la stessa idea di base, si dovrà però aggiungere in testa il comando `python`.

Ottenendo il comando: `python ../../Scripts/documenter.py`

Per esempio, in Linux:



Lo script identificherà automaticamente la cartella di lavoro e procederà a richiedere alcuni dati:



```
penaz ... > Progetto SWE > Materiale Grezzo > Nuovo Documento ../../Scripts/documenter.py master
Attività Disponibili:
-----
[1] Crea un nuovo documento da qui
[2] Pulisci la cartella e le sottocartelle dai file temporanei
[3] Ricompila il documento LaTeX
[4] Avvia una live preview tramite latexMk

[0] Esci
-----
Cosa vuoi fare? 1
Che titolo dare al documento? Un Documento di Prova Qualsiasi
Che uso verrà fatto di questo documento?
[1] Interno
[2] Esterno
-----
Inserire un valore: 1
Inserire una breve descrizione del documento: Questo documento è solo una prova atta a far vedere come funzionano gli script automat
ici
penaz ... > Progetto SWE > Materiale Grezzo > Nuovo Documento
```

È sufficiente seguire le istruzioni.

Importante: Il documento non può ancora compilare in questo stato, dato che non è presente alcuna voce nel changelog, è importante aggiungerne una tramite `bookKeeper.py`

5 Altre funzionalità di `documenter.py`

Come è possibile notare, `documenter` non crea solo documenti, ma permette di ripulire le directory (e tutte le sottodirectory) da file temporanei creati da $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$, ricompilare il documento e fare uso di `latexmk` per avere una preview in quasi-tempo reale (la preview si aggiorna ad ogni salvataggio) del documento che si sta creando. Ricompilando il nostro documento di prova avremo un risultato del genere (solo le parti modificate sono state prese, per risparmiare spazio):

Un Documento di Prova Qualsiasi

| | |
|--------------|---|
| Approvazione | |
| Redazione | |
| Verifica | |
| Stato | |
| Uso | Interno |
| Destinato a | Commandline Team Prof. Tullio Vardanega Prof. Riccardo Cardin |

Descrizione

Questo documento è solo una prova atta a far vedere come funzionano gli script automatici

6 glossarizer.py - Creazione di voci di glossario

Dipendenze: Python 3.4+, pdflatex (per la ricompilazione dei documenti)

Questo script permette di aggiungere rapidamente, in ordine alfabetico, nel giusto file, semplici definizioni di termini, senza dover modificare codice L^AT_EX.

Questo script va **assolutamente** avviato dalla cartella del glossario.

```

sakura
penaz ... > Progetto SWE > Materiale Grezzo > Glossario .../Scripts/glossarizer.py master
#-----#
#           Glossarizer v0.0.1           |
#           by Penaz                     |
#           Piccolo script per velocizzare la |
#           creazione di voci di glossario   |
#-----#

Che parola vuoi definire? Travis-CI
Qual'è il suo significato? Servizio di integrazione continua che permette l'esecuzione automatica di comandi, oltre
all'esecuzione automatica di test
Vuoi ricompilare il documento? (y/N): y
This is makeindex, version 2.15 [TeX Live 2017] (kpathsea + Thai support).
Scanning input file Document.idx...done (19 entries accepted, 0 rejected).
Sorting entries...done (74 comparisons).
Generating output file Document.ind...done (50 lines written, 0 warnings).
Output written in Document.ind.
Transcript written in Document.ilg.
This is makeindex, version 2.15 [TeX Live 2017] (kpathsea + Thai support).
Scanning input file Document.idx...done (19 entries accepted, 0 rejected).
Sorting entries...done (74 comparisons).
Generating output file Document.ind...done (50 lines written, 0 warnings).
Output written in Document.ind.
Transcript written in Document.ilg.

Grazie per aver usufruito dei miei servizi
penaz ... > Progetto SWE > Materiale Grezzo > Glossario d master

```

Lo script si occuperà dell'ordinamento delle voci, della creazione di codice L^AT_EXe della ricompilazione del documento:

21 T

Tag

Identificatore, rappresentante un'etichetta evidenziata automaticamente ed a cui si può saltare velocemente.

Text-to-speech

Per text-to-speech (TTS) si intendono i sistemi di sintesi vocale che danno la possibilità di convertire il testo in parlato.

Travis-CI

Servizio di integrazione continua che permette l'esecuzione automatica di comandi, oltre all'esecuzione automatica di test