Istruzioni per l’uso

Il programma richiede l’uso del modulo **markdown** che va installato (e.g. **pip install markdown**)

Argomenti (principali) del programma:

**--xml nome\_file (default='demo.xml') 🡪output moodle xml file**

**--questionFile nome file (default='domande.txt') 🡪 input txt question file. Il file si deve trovare nella workDir**

**--footer nome\_file (default='footer.txt') 🡪 footer txt for questions (contenuto della parte terminale delle domande, uguale per tutti e che contiene attualmente il link alla documentazione online). Il file viene cercato nella directory del sorgente (a meno che non specifichiate un path diverso)**

**--workDir dir\_name (default='') 🡪 working dir (quella da cui legge il file delle domande, in cui genera il file xml e in cui va a cercare eventuali file da inserire nelle domande)**

**--category name (default=’** **"Default per 14BHDLZ\_16"’) 🡪 nome della categoria delle domande (serve per includerle in sottocategorie nel deposito domnade e trovarle più facilmente. Se non si scrive niente, tutte le domande vanno nella categoriale principale, quella di default)**

**--enableACE value (default=False) 🡪 Enabling ACE editor (per gli esami con Respondus ACE va sempre disabilitato, per gli esercizi senza respondus, si può abilitare)**

**--defaultPythonGrade value (default=26.0)🡪 Python grade**

**--defaultEssayGrade value (default=2.0) 🡪 Essay grade**

Altri comandi (per usi particolari…):

**--processSubDirs (default=False) 🡪 Permette di puntare ad una directory che contiene un elenco di compiti (uno per directory) e di elaborarli individualmente (uno per dir…). Il risultato dell’elaborazione dipende dagli altri parametri impostati (e.g., o crea un file di import per ogni compito. O un file di import unico con tutti i compiti). Se l’impostazione createMBZ è True, non viene considerata**

**--createUniqueImport (default=False) 🡪 crea un unico file di import xml con tutti i compiti che sono contenuti in una cartella principale (processSubDir deve essere True). Se -–merge (default=False) è True, allora crea una UNICA categoria di domande, altrimenti se vale False, ogni compito è salvato nel deposito domande in una sua categoria. Se l’impostazione createMBZ è True, non viene considerata**

**--createMBZ (default=False) 🡪 Crea un file di backup di moodle (mbz) che permette di caricare in una volta sola tutte le domande nel deposito e tutti i compiti creati. Se l’impostazione createMBZ è True, NON vengono creati i file xml di import, e neanche gli HTML di verifica delle domande**

Il programma legge le domande da un file di testo (nota: **deve essere codificato in utf-8** per evitare problemi con le lettere accentate), che ha un formato proprio e che permette di inserire tutti i campi delle domande (attualmente solo quelle di Coderunner, inclusi eventuali file che devono essere letti durante l’esercizio, e gli essay).

Procedura

1. **Create una cartella** (con il nome dell’esame, e.g. Info2021.01.25-Corso-XX) e create lì i file di testo per la generazione delle domande
2. La cosa migliore è creare un file **domande.txt codificato in UTF-8** (in Windows: create un file di testo, apritelo e fate salva con nome, usando lo stesso nome ma selezionando come formato utf-8). Sulle altre piattaforme lascio gli esperti indicare come fare
3. Il programma python prende un sacco di parametri, ma sono tutti facoltativi. Quindi, se avete seguito le istruzioni qui sopra (e il file con le domande si chiama domande.txt) basta chiamarlo con **python CodeRunnerGenerator.py --category Info2021.01.25-Corso-XX –workDir Info2021.01.25-Corso-XX**

Domande Coderunner

Il testo della domanda viene completato dal footer, che è uguale per tutte le domande, e che contiene le indicazioni su come usare l’interfaccia e il link alla documentazione Python.

All’interno del file possono essere presenti più di una domanda, e il file xml finale contiene l’elenco di tutte le domande convertite in formato xml moodle che quindi si possono importare direttamente all’interno di exercise o exam.

I vari file possono essere memorizzati in una directory di lavoro (che si può specificare al programma), in cui verranno poi anche salvati gli output prodotti. Nella directory del sorgete deve SEMPRE essere incluso il file template.py che contiene il codice necessario a creare la stringa json da inviare al server che fa girare pyhon (e per recuperare e leggere correttamente l’output dell’esecuzione). Quindi riassumendo:

* **Directory del sorgente**: deve includere anche i file **template.py** e **footer.txt**
* **Directory di lavoro**: contiene il **file con le domande** e tutti gli **eventuali file (da leggere…)** che devono essere inclusi nelle domande

Essay

Gli essay, servono per le domande di teoria, e consistono in un testo (codificato in markdown), e un eventuale template di risposta (sempre in markdown)

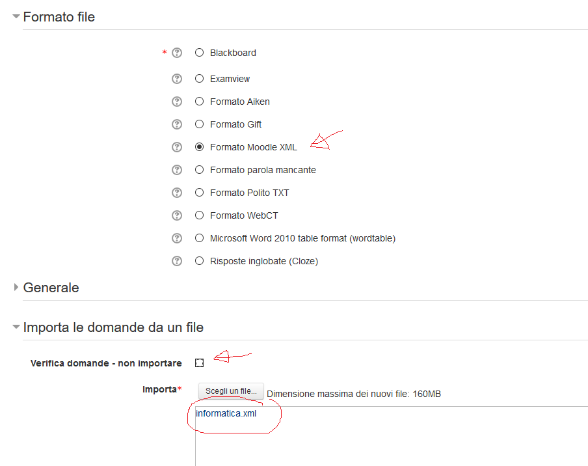
## Upload in moodle

Una volta creato il file xml, i passaggi per importarlo in exercise/exam sono i seguenti:

* Deposito domande 🡪 importa



* Selezionare come formato moodle xml, togliere il check su verifica domande e trascinare il file xml generato



## Formato file di testo delle domande coderunner

Per evitare duplicati incoerenti, la descrizione del formato è stata spostata nel file “istruzioni solo txt.doc”