ArDiVa
(Arbitrary Dictionary Validator)

istruzioni, versione 1.0

Descrizione

ArDiVa è un modulo Python che permette di eseguire operazioni di validazione su strutture dictionary-like in base a modelli e processi definiti dall'utente.

ArDiVa fornisce un oggetto Validator che combina un oggetto Model e un oggetto Process per le diverse fasi e/o modalità della validazione.

Componenti

Validator

Validator inizializza il processo di validazione. Per essere creato il validatore richiede il passaggio di un dizionario per creare il Model o di una tupla per creare il Process.

Validator ha un metodo getLog() che può usare per recuperare la lista degli errori relativi al lancio più recente delle validazioni su Model o Process. Ogni volta che viene lanciata la validazione sul Model o sul Process, questi azzerano i relativi log prima di inserire nuovi messaggi.

Model

Model rappresenta la struttura generale del dizionario che si vuole validare.

È costituito da un dizionario (con alcune chiavi interpretate in maniera particolare) e può ricevere in aggiunta un parametro di severità della validazione. Questo può essere specificato anche al momento del lancio della validazione.

Severità della validazione

I livelli di severità sono:

- STRICT: tutte le chiavi e sottochiavi presenti nel dizionario Model devono essere presenti nel candidato. Il candidato non può avere nessuna chiave che non sia esplicitamente presente nel Model. I valori del candidato devono rispettare i constraint descritti per le stesse chiavi nel Model.
- SUPERSET: tutte le chiavi e sottochiavi esplicitate nel dizionario Model devono essere presenti nel candidato. Il candidato può anche presentare chiavi non esplicitamente presenti nel Model. I valori delle chiavi comuni tra Model e candidato devono rispettare i constraint descritti nel Model.
- SUBSET: tutte le chiavi e sottochiavi presenti nel candidato devono essere presenti nel

Model. I valori delle chiavi comuni tra Model e candidato devono rispettare i constraint descritti nel Model.

- LOOSE: almeno una chiave del Model deve essere presente anche nel candidato. I valori delle chiavi comuni tra Model e candidato devono rispettare i constraint descritti nel Model.

Sono possibili alcune eccezioni a queste indicazioni:

- Regular expression: quando viene indicata come chiave una regular expression (vedi sotto), ArDiVa non la considera come chiave reale ma applica il constraint a tutte le chiavi che danno un match positivo all'espressione. In presenza di una regular expression nel Model, quindi, la severità massima è di un SUBSET; ArDiVa cambierà automaticamente il parametro di severità al momento in cui verrà lanciata la validazione del Model.
- Sottovalidazioni: il valore corrispondente a una determinata chiave può essere descritto in generale nel Model come un dizionario (vedi sotto). In questo caso la validazione rispetto al Model non prosegue nel sottodizionario (ArDiVa si limita a verificare che i valore per quella chiave sia un dizionario anche nel candidato) e le chiavi presenti al suo interno non vengono considerate per la validazione STRICT o SUBSET.

Descrittori di chiavi

Le chiavi presenti nel Model vengono genericamente trattati come normali chiavi di un dizionario e confrontate direttamente con quelle del candidato. Esistono però alcune eccezioni:

- Tuple: quando una chiave è rappresentata da una tupla, ArDiVa ne espanderà i valori e applicherà il constraint corrispondente a tutte le chiavi del candidato indicate dai membri della tupla. Per indicare un'effettiva chiave-tupla sarà necessario inserirla come elemento all'interno di una tupla (e.g. ((val_tupla_1, val_tupla_2),))
- Regular expression: quando una chiave è rappresentata da una regular expression compilata (e.g. re.compile('regexstring')), il constraint corrispondente verrà applicato a tutte le chiavi del candidato che daranno match positivo rispetto alla regular expression.

Valori

I valori indicati nel Model sono confrontati con quelli del candidato secondo criteri particolari per ricavarne dei constraint da applicare al candidato. In particolare, il valore per una determinata chiave può essere un singolo oggetto o una collection.

Nel caso di una collection, il constraint viene considerato rispettato se un singolo criterio al suo interno viene soddisfatto dal valore del candidato. Altrimenti questa operazione viene fatta sul singolo valore disponibile. Collection contenute dentro la collection principale vengono considerate come un unico da confrontare per uguaglianza.

- Regular expression: il valore del candidato viene confrontato come match in stringa rispetto a quanto indicato dalla regular expression compilata.

- Tipo: il valore del candidato soddisfa il constraint se è istanza del tipo indicato (o di una sua sottoclasse)
- Dizionario: la validazione prosegue al livello inferiore in maniera ricorsiva.

Nel caso il constraint sul valore fosse particolarmente complesso, è consigliabile spostare la verifica relativa all'interno del Process e indicare un constraint generico nel Model.

Process

Process descrive una sequenza di operazioni di validazione dei valori da effettuare sulle chiavi corrispondenti ai descrittori indicati nel Process stesso; se una chiave descritta nel Process non è presente nel candidato, la validazione relativa viene saltata e considerata automaticamente vera.

Lista delle validazioni

Il Process può essere rappresentato completamente con una lista o tupla di validazioni, ognuna rappresentata a sua volta da una lista/tupla così composta:

```
(
      (descrittori_chiavi),
      (checklist),
      combinatore per check default su all(),
      combinatore per chiave default su all(),
      messaggio di errore opzionale
)
```

La checklist è costituita da una lista di check descritti in forma di lista/tupla:

```
(
      identificatore_funzione,
      risultato_previsto default su True,
      (argomenti_aggiuntivi_posizionali default su None),
      {argomenti_aggiuntivi_keyword default su None}
)
```

Quando Process effettua una validazione, espande tutti i key descriptor in un'unica lista, e applica a ogni chiave la sequenza di check; i risultati vengono prima combinati per tutti i check su una singola chiave, e alla fine di tutta la procedura vengono combinati come risultati delle chiavi. È possibile indicare una funzione che combini i risultati (all'interno di una singola chiave o tra le diverse chiavi della validazione). L'impostazione predefinita è la funzione builtin all().

I vari step definiti dalla lista di validazioni invece sono combinati sempre con And.

Descrittori di chiavi

Il descrittore di chiavi presente in ogni step di validazione viene usato da Process per passare il primo argomento all'identificazione di funzione del check; questo parametro è costituito dal valore associato alla chiave indicata dal descrittore (candidato[descrittore]). Altri parametri (posizionali e keyword) vengono passati dopo questo parametro principale.

Come per il Model, il Process esegue un'interpretazione dei descrittori di chiavi che gli vengono passati nelle validazioni:

- Tupla: il descrittore di chiavi è sempre inserito in una tupla. La validazione viene

eseguita su tutte le chiavi a cui si possono applicare i descrittori inseriti nella tupla.

- None: il descrittore None passa come argomento alla funzione l'intero dizionario. Questo permette di eseguire validazioni complesse che confrontino tra di loro i valori associati a più chiavi.
- Regular expression: questo descrittore viene applicato a tutte le chiavi dello stesso livello che corrispondono tramite re.match().
- Lista: la sequenza di valori di una lista viene usata per costruire ricerche di valori oltre il livello iniziale del dizionario. La lista parte sempre dal livello più alto del dizionario e scende di un livello ad ogni elemento. In ogni elemento il descrittore di chiave viene rielaborato da capo, quindi è possibile ampliare il campo di ricerca/match con tuple e regular expression. Gli unici valori che vengono estratti da questa ricerca sono quelli dell'ultimo livello raggiunto dalla lista.