**Documentazione per il Notebook chk3D e specifiche delle Funzioni Richiamate**

**1. Introduzione**

Il notebook **chk3D** è progettato per verificare la presenza e la validità dei file necessari per il caricamento dei modelli geologici 3D in formato GOCAD e CSV secondo le specifiche del visualizzatore web GeoIT3D. Include funzioni per la validazione della struttura dei file, la verifica della formattazione e la coerenza dei dati. Questo strumento è essenziale per garantire che i dati utilizzati nelle analisi siano accurati e conformi agli standard richiesti.

**2. Struttura del Notebook**

**2.1 Importazione delle Librerie e Funzioni**

* **Librerie Importate:**
  + **sys**: Fornisce accesso a vari parametri e funzioni del sistema.
  + **os**: Permette di interagire con il sistema operativo, gestendo file e directory.
  + **pandas**: Utilizzato per la manipolazione e l'analisi dei dati, in particolare per la gestione di file CSV.
  + **IPython.display**: Consente di visualizzare output formattati.
  + **datetime**: Gestisce date e orari.
* **File Importati:**
  + **file\_utils**: Contiene funzioni per la verifica della presenza di file.
  + **csv\_validator**: Contiene funzioni per la validazione della struttura e del contenuto dei file CSV.
  + **ts\_validator**: Contiene funzioni per l'analisi e la validazione dei file GOCAD.
  + **json\_validator**: Contiene funzioni per la validazione del file **descriptor.json**.

**2.2 Specifiche dei File**

* **File Necessari: NB unico parametro modificabile**
  + Elenco dei file richiesti per l'analisi, inclusi file CSV e JSON. I file che compongono il modello 3D devono essere prodotti secondo le specifiche CARG per i modelli 3D e posti in un'unica cartella per garantire il corretto funzionamento del notebook. Il ‘nome\_cartella’ è l’unico input per il quale è necessaria la definizione all’interno del Notebook
* **Validazione della Struttura dei File CSV:**
  + Specifiche per ogni file CSV, inclusi i campi attesi e le lunghezze massime. Queste specifiche sono fondamentali per garantire che i dati siano formattati correttamente e contengano le informazioni necessarie. (da non modificare)

**2.3 Logging**

* Configurazione del logging per registrare le informazioni e gli errori durante l'esecuzione del notebook. Necessario al debug e per tenere traccia delle operazioni eseguite.

**2.4 Verifica della Presenza dei File**

* Richiama la Funzione **verifica\_file\_presenti** contenuta nel file file\_utils.py per controllare la presenza dei file necessari nella cartella specificata. Questa funzione restituisce un riepilogo dei file presenti, mancanti e simili (per verificare refusi).

**2.5 Verifica della Formattazione dei File CSV**

* Richiama le Funzioni contenute nel file csv\_validator.py per controllare la codifica, le terminazioni di riga, il numero di campi e la struttura dei file CSV. Questi controlli assicurano che i file CSV siano conformi agli standard richiesti.

**2.6 Verifica dei File GOCAD**

* Richiama le Funzioni contenute nel file ts\_validator.py per analizzare i file GOCAD, verificare le geometrie descritte e la sintassi delle keywords. Queste funzioni sono cruciali per garantire che i dati geologici siano rappresentati correttamente.

**2.7 Verifica del File Descriptor**

* Richiama la Funzione **check\_descriptor\_structure** dal file json\_validator.py per validare la struttura del file **descriptor.json**. Questa funzione assicura che il file contenga tutti i campi richiesti e che siano formattati correttamente.

**3. Funzioni Dettagliate**

**3.1 Funzioni di file\_utils.py**

* **verifica\_file\_presenti(cartella, file\_necessari, soglia\_similarita=0.95)**
  + **Descrizione:** Verifica la presenza di file necessari in una cartella e segnala file simili e file aggiuntivi.
  + **Parametri:**
    - **cartella**: Percorso della cartella da controllare.
    - **file\_necessari**: Lista dei nomi dei file da verificare.
    - **soglia\_similarita**: Soglia per considerare due nomi di file simili (tra 0 e 1).
  + **Restituisce:** Tuple contenente file presenti, file mancanti, file simili e file aggiuntivi.

**3.2 Funzioni di csv\_validator.py**

* **verifica\_csv(cartella, file\_csv)**
  + **Descrizione:** Controlla che i file CSV siano in formato UTF-8 e abbiano terminazioni di riga LF.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente i file CSV.
    - **file\_csv**: Lista dei file CSV da verificare.
  + **Restituisce:** Dizionario con i risultati della verifica per ogni file CSV.
* **valida\_csv(cartella, specifiche\_csv)**
  + **Descrizione:** Verifica che i file CSV rispettino i vincoli specificati.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente i file CSV.
    - **specifiche\_csv**: Dizionario con le specifiche per ogni file CSV.
  + **Restituisce:** Dizionario con i risultati della validazione per ogni file.
* **verifica\_numero\_campi\_csv(cartella, specifiche\_csv, verbose=True)**
  + **Descrizione:** Controlla che ogni riga dei file CSV contenga esattamente il numero di campi attesi.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente i file CSV.
    - **specifiche\_csv**: Dizionario con le specifiche per ogni file CSV.
    - **verbose**: Se True, stampa un report formattato durante l'esecuzione.
  + **Restituisce:** Tuple con i risultati della validazione e un riepilogo.
* **valida\_id\_univoci\_csv(cartella, specifiche\_id\_csv, verbose=True)**
  + **Descrizione:** Verifica che gli ID nelle colonne specificate siano univoci.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente i file CSV.
    - **specifiche\_id\_csv**: Dizionario con le specifiche per ogni file CSV.
    - **verbose**: Se True, stampa un report formattato durante l'esecuzione.
  + **Restituisce:** Tuple con i risultati della validazione e un riepilogo.
* **valida\_campi\_booleani\_csv(cartella, lista\_file, colonne\_da\_controllare=["active\_fault","seismogenic\_fault","capable\_fault"], verbose=True)**
  + **Descrizione:** Controlla che le colonne contengano solo valori booleani validi.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente i file CSV.
    - **lista\_file**: Lista dei nomi dei file CSV da controllare.
    - **colonne\_da\_controllare**: Colonne da controllare per valori booleani.
    - **verbose**: Se True, stampa un report formattato durante l'esecuzione.
  + **Restituisce:** Tuple con i risultati della validazione e un riepilogo.
* **valida\_campi\_numerici\_csv(cartella, lista\_file, verbose=True)**
  + **Descrizione:** Verifica che i campi specificati contengano valori numerici validi.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente i file CSV.
    - **lista\_file**: Lista dei nomi dei file CSV da controllare.
    - **verbose**: Se True, stampa un report formattato durante l'esecuzione.
  + **Restituisce:** Tuple con i risultati della validazione e un riepilogo.

**3.3 Funzioni di ts\_validator.py**

* **analyze\_gocad\_files(cartella, filenames, valid\_header\_kw=None, valid\_coord\_kw=None, valid\_conn\_kw=None, special\_keywords=None)**
  + **Descrizione:** Analizza i file GOCAD e verifica la loro geometria e validità delle keywords.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente i file.
    - **filenames**: Lista dei nomi dei file GOCAD da analizzare.
    - **valid\_header\_kw**: Keywords valide per l'header.
    - **valid\_coord\_kw**: Keywords valide per le coordinate.
    - **valid\_conn\_kw**: Keywords valide per la connettività.
    - **special\_keywords**: Keywords speciali e regole di validazione.
  + **Restituisce:** Dizionario con i risultati dell'analisi.
* **print\_gocad\_summary(analysis\_results, cartella)**
  + **Descrizione:** Stampa un report completo dei risultati dell'analisi con riferimenti alle linee.
  + **Parametri:**
    - **analysis\_results**: Risultati dell'analisi come restituiti da **analyze\_gocad\_files**.
    - **cartella**: Percorso della cartella contenente i file.
  + **Restituisce:** Nessuno (stampa il report).

**3.4 Funzioni di json\_validator.py**

* **check\_descriptor\_structure(cartella: str, required\_fields: Dict[str, Any]) -> Dict[str, Any]**
  + **Descrizione:** Valida il file **descriptor.json** e restituisce un report sintetico.
  + **Parametri:**
    - **cartella**: Percorso della cartella contenente **descriptor.json**.
    - **required\_fields**: Dizionario con i campi richiesti e i loro tipi attesi.
  + **Restituisce:** Dizionario con il risultato della validazione.