Goetas Web Services

Emanuele Davide
5BII
14/02/2018

Documentazione

Con “goetas-webservices / xsd2php” si puòi convertire qualsiasi XSD in classi PHP.
XSD2PHP può anche generare metadati compatibili con Serializer JMS che possono essere utilizzati per serializzare-deserializzare le istanze dell'oggetto.
Per fare uso di questo strumento è necessario installare XSD2PHP aggiungendo la dipendenza al file “composer.json”.

E’ necessario avere un file di configurazione (“config.yml”) dove sono contenuti tutti i namespace e le cartelle di associazione delle informazioni.

Parametri obbligatori

“xsd2php.namespaces” definisce le associazioni fra i namespace dell’XML e quelli del PHP.

“xsd2php.destinations_php” specifica la cartella dove salvare le classi PHP appartenenti al namespace PHP fornito.

“xsd2php.destinations_jms” specifica la cartella dove salvare i metadati di JMS Serializer appartenenti al namespace PHP fornito.

“vendor/bin/xsd2php convert config.yml <cartella_xsd>.xsd”

Questo comando genera classi PHP e file di metadati (JMS) per tutti gli XSD che sono nella cartella specificata, usando la configurazione disponibile in “config.yml”.

Serializzazione e Deserializzazione

La serializzazione è un processo per salvare un oggetto in un supporto di memorizzazione lineare, o per trasmetterlo su una connessione di rete. La serializzazione può essere in forma binaria o può utilizzare codifiche testuali (XML) direttamente leggibili. Lo scopo della serializzazione è di trasmettere l'intero stato dell'oggetto in modo che esso possa essere successivamente ricreato nello stesso identico stato dal processo inverso, chiamato deserializzazione.
Deserializzare un oggetto, infatti, significa ricostruirlo a partire dalla sua rappresentazione binaria ottenuta da un file o da un canale di rete.

Strategia di denominazione

Esistono due tipi di strategie di denominazione: brevi e lunghi. L'impostazione di default è breve, tuttavia questa strategia di denominazione può generare conflitti.
La strategia di denominazione lunga avrà come suffisso degli elementi con “Element” e dei tipi con “Type”.

Ex :
MyNamesapce \ User diventerà MyNamesapce \ UserElement
MyNamesapce \ UserType diventerà MyNamesapce \ UserTypeType

Un XSD ad esempio con un tipo denominato “User”, un tipo denominato “UserType”, un elemento root denominato “User” e “UserElement”, funzionerà solo quando si utilizza la strategia di denominazione lunga.

Fonti

https://github.com/goetas-webservices/xsd2php/blob/master/README.md
https://msdn.microsoft.com/it-it/library/bb412179(v=vs.110).aspx