Guida ai servizi di interoperabilità

Servizi di interoperabilità esposti da Geoweb

Questo documento ha l'obbiettivo di fornire le informazioni necessarie per effettuare richieste utilizzando i servizi di interoperabilità forniti da Geoweb.

```
Da notare che ogni esempio fornito in questo documento è relativo ad uno
specifico software chiamato "Postman",
tramite il quale sono stati eseguiti i test qui mostrati.
```

Tabella esempio: test_class

id_resource	cod_resource	fax	...
12345	001	aaa	...
54321	002	bbb	...
13245	003	ccc	...
...	...	...	...

Servizi di interoperabilità disponibili:

READ

- **detailEntity:** informazioni di un singolo record di un classe
- **listEntity:** informazioni dei record di una classe (paginazione e ordinamento)
- **listEntity:** informazioni dei record di una classe (paginazione e ordinamento e filtri)

INSERT

- **1.1) insertGwRecord (multipart/form-data):** richiede che i dati in ingresso siano di tipo **"multipart/form-data"**, ma permette che siano forniti sia in formato "testo" che in formato "file".
- **1.2) insertGwRecord (json):** richiede che i dati in ingresso siano di tipo "mappa chiave/valore", fornita tramite un oggetto **JSON**.
- **1.3) insertGwRecord (modelAttribute):** richiede che i dati in ingresso siano inseriti in un oggetto **"FormData"**, specificamente costruito nel corpo della richiesta seguendo la struttura di un oggetto 'FormData' di Geoweb (più avanti una spiegazione approfondita).

UPDATE

- **2.1) updateGwRecord (multipart/form-data):** richiede che i dati in ingresso siano di tipo **"multipart/form-data"**, ma permette che siano forniti sia in formato "testo" che in formato "file".
- **2.2) updateGwRecord (json):** richiede che i dati in ingresso siano di tipo "mappa chiave/valore", fornita tramite un oggetto **JSON**.
- **2.3) updateGwRecord (modelAttribute):** richiede che i dati in ingresso siano inseriti in un oggetto **"FormData"**, specificamente costruito nel corpo della richiesta seguendo la struttura di un oggetto 'FormData' di Geoweb (più avanti una spiegazione approfondita).

NOTA: Postman è in grado di determinare automaticamente il contenuto di una richiesta. Ciò significa che non è necessario definirla manualmente, a meno che l'obbiettivo dell'utente non sia stabilire un Content-Type fisso (può essere fatto nella sezione 'Headers' della richiesta).

Descrizione delle single richieste, utilizzando come esempio il comportamento di Postman

detailEntity (Entità)

		Esempi
Title	Informazioni di un record, dato il valore della chiave e il nome della classe	
Method	GET	
URL	http://[indirizzo]/[contesto]/services/detailEntity/[**id**]/[**className**]	http://localhost:8080/webclient/services/detailEntity/12345/test_class
Success Response Content	{"key1":"value1","key2":"value2",...}	{"id_resource":12345,"cod_resource":"001","fax":"aaa"}

listEntity (Lista Entità)

		Esempi
Title	Lista completa delle entità di una classe, con paginazione e ordinamento	
Method	GET	
URL	http://[indirizzo]/[contesto]/services/listEntity/[**className**]	http://localhost:8080/webclient/services/listEntity/test_class
Header	(optional) x-range=[startIndex-endIndex]	x-range=0-29
URL Params	(optional) sortBy=[+columnName1,+columnName2,...]	sortBy=+id_resource,-cod_resource
Success Response Content	{"keyColumn":"nome_campo_chiave", "data":[{"key1":"value1","key2":"value2",...},{"key1":"value4","key2":"value5",...},...]	{"keyColumn":"id_resource", "data":[{"id_resource":12345,"cod_resource":"001","fax":"aaa"},{"id_resource":13245,"cod_resource":"003","fax":"ccc"},{"id_resource":54321,"cod_resource":"002","fax":"bbb"}]}

- **x-range:** indica l'intervallo di valori da restituire (paginazione) (es.x-range=0-29 vengono restituiti i primi 30 record della classe). **Se non specificato, sono restituiti tutti i record della classe**.
- **sortBy:** indica il tipo di ordinamento da effettuare. I campi, separati dalla virgola, possono essere preceduti dal carattere "+" per indicare un ordinamento crescente, "-" per indicare un ordinamento decrescente (DESC). Default crescente.

listEntity (Lista Entità filtrate)

		Esempi
Title	Lista Entità: lista completa delle entità di una classe con paginazione, ordinamento e filtri	
Method	POST	
URL	http://[indirizzo]/[contesto]/services/listEntity/[**className**]	http://localhost:8080/webclient/services/listEntity/test_class
Header	(optional) x-range=[startIndex-endIndex]	x-range=0-29
URL Params	(optional) sortBy=[+columnName1,+columnName2,...]	sortBy=+id_resource,-cod_resource
Content-Type	application/json	
Body	{"filters" : [{"condition": [alphanumeric] "columnName": [alphanumeric] "operator": [alphanumeric] "filterType": [alphanumeric] "value": [array] } , [...] }	{"filters" : [{"condition":"AND","columnName":"id_resource", "operator":"<","filterType":"INTEGER","value":[20000]},{"condition":"AND","columnName":"fax", "operator":"is not null","filterType":"STRING","value":[]}]}
Success Response Content	{"keyColumn":"nome_campo_chiave", "data":[{"key1":"value1","key2":"value2",...},{"key1":"value4","key2":"value5",...},...]	{"keyColumn":"id_resource", "data":[{"id_resource":12345,"cod_resource":"001","fax":"aaa"},{"id_resource":13245,"cod_resource":"003","fax":"ccc"}]}

Headers

- **x-range:** indica l'intervallo di valori da restituire (paginazione) (es.x-range=0-29 vengono restituiti i primi 30 record della classe). **Se non specificato, non viene applicata la paginazione.**.

Params

- **sortBy:** indica il tipo di ordinamento da effettuare. I campi, separati dalla virgola, possono essere preceduti dal carattere "+" per indicare un ordinamento crescente, "-" per indicare un ordinamento decrescente (DESC). Default crescente.

Body

- **{“filters”: …}:** l'oggetto che contiene i filtri da applicare alla ricerca è costituito costituito dalla chiave “filters” e come valore una lista di oggetti, dove ogni oggetto rappresenta un singolo un filtro.
    - I filtri vengono tradotti in una clausola WHERE sql secondo questa struttura:
        - WHERE <beforeOpenBrackets1> <columnName1> <operator1> <value1> <endClosedBrackets1> <beforeOpenBrackets2> <condition2> <columnName2> <operator2> <value2> <endClosedBrackets2>
            - Esempio: l'oggetto filtri:
                - {“filters” : [{“condition”:“AND”,“columnName”:“id_resource”, “operator”:“<”,“filterType”:“INTEGER”,“value”:[20000]},{“condition”:“ AND”,“columnName”:“fax”,“operator”:“is not null”,“filterType”:“STRING”,“value”:[]}]}
            - sarà tradotto nella seguente clausola WHERE:
                - WHERE id_resource < 20000 AND fax is not null
    - Il singolo filtro è così costituito:
        - **beforeOpenBrackets**: (INTEGER) numero di “(” da inserire prima della condizione. Default: 0
        - **condition**: (STRING) operatore logico. Possibili valori: “AND” oppure “OR”. La condizione è applicata prima del filtro per cui è definita (per il primo filtro viene quindi ignorata),
        - **columnName**: (STRING) nome dell'attributo a sinistra dell'operatore,
        - **operator**: (STRING) l'operatore (es “=”,“<>”,“<”,“⇐”,“>”,“>=”,“LIKE”,“BETWEEN”,“IS NULL”,“IS NOT NULL”,…),
        - **filterType**: (STRING) il datatype del valore del filtro (es. “STRING”, “INTEGER”, “NUMBER”, “DATE”, “CUSTOM”)
            - i valori corrispondenti al tipo “DATE” devono essere di tipo STRING e formattati secondo il seguente pattern ISO8601: “yyyy-MM-dd'T'HH:mm:ss”
            - il formato di tipo “CUSTOM” è utilizzabile nel caso in cui il filtro non è esprimibile tramite il formato di filtro standard di GEOWEB. Nel caso di filtro CUSTOM, la clausola WHERE verrà costruita utilizzando il contenuto del valore del campo (opzionale) customFilter
        - **value**: (LIST) una lista di valori (STRING, INTEGER, NUMBER). Il numero di valori deve essere coerente con il tipo di operatore: es. nel caso di operatore “=” la lista deve contenere un solo valore. Nel caso di operatore “is null” o “is not null” la lista deve essere vuota.
        - **endClosedBrackets**: (INTEGER) numero di “)” da inserire dopo la condizione. Default: 0

1.1) insertGwRecord (multipart/form-data)

		Esempi
Title	Inserimento di un record (di una classe)	
Method	POST	
URL	http://[indirizzo]/[contesto]/services/insertGwRecord/[**className**]	http://localhost:8080/webclient/services/insertGwRecord/test_class
Content-Type	multipart/form-data	

Body	`Content-Disposition: form-data; name="key1"`
	`value1`
	`------WebKitFormBoundary--,`
	`Content-Disposition: form-data; name="key2"`
	`value2`
	`-------WebKitFormBoundary--`
	`-...`	

Success Response Content	{“success”:[boolean],“description”:“[messaggio_testuale]”,“itemId”:“[valore_chiave_primaria]”}	{“success”:true,“description”:“Operazione eseguita con successo”,“itemId”:“1870567”}

insertGwRecord (multipart/form-data) - TEST

Indirizzo della richiesta:

```
POST - http://[indirizzo]/[contesto]/services/insertGwRecord/[gwClassName]
(esempio) http://localhost:8080/webclient/services/insertGwRecord/test_class
```

Body:

Selezionare l'opzione *form-data* ed inserire ogni dato seguendo una struttura chiave/valore. Utilizzando Postman sarà possibile cambiare il tipo di chiave da "testo" a "file", permettendo dunque di inviare anche file attraverso questa richiesta (il tipo e l'estensione del file possono variare in base al contesto della richiesta, e non deve obbligatoriamente trattarsi di un file di testo).

(esempio)

Key	Value
fax (Text)	ddd
test (File)	[Scegli] test_file.txt

Inserisce un nuovo record con "fax" = "ddd" e "test" = test_file.txt

1.2) insertGwRecord (JSON)

		Esempi
Title	Inserimento di un record (di una classe)	
Method	POST	
URL	http://[indirizzo]/[contesto]/services/insertGwRecord/[**className**]	http://localhost:8080/webclient/services/insertGwRecord/test_class
Content-Type	application/json	
Body	{"key1":"value1","key2":"value2",...}	
Success Response Content	{"success":[boolean],"description":"[messaggio_testuale]","itemId":"[valore_chiave_primaria]"}	{success:true,"description":"Operazione eseguita con successo","itemId":"1870567"}

insertGwRecord (JSON) TEST

Indirizzo della richiesta:

```
POST -
http://[indirizzo]/[contesto]/services/insertGwRecord/[gwClassName]/json
(esempio)
http://localhost:8080/webclient/services/insertGwRecord/test_class/json
```

Header:

In questo caso è necessario impostare un header che abbia *Content-Type* come chiave, e *application/json* come valore. Questo perché il servizio richiesto necessita obbligatoriamente di una mappa chiave/valore in ingresso, sotto forma di oggetto JSON.

Body:

Selezionare l'opzione *raw*. Apparirà un menu a tendina, tramite il quale l'utente potrà scegliere il formato e/o il tipo di testo che andrà ad utilizzare (nel nostro caso, sarà *JSON (application/json)*). Dopo

di che, sarà possibile inserire i dati da inviare, ovviamente seguendo il formato JSON.

```
(esempio)
{
  "cod_resource": "004",
  "fax": "eee"
}
```

Inserisce un nuovo record con "cod_resource" = "004" e "fax" = "eee"

1.3) insertGwRecord (modelAttribute)

Indirizzo della richiesta:

```
POST -
http://[indirizzo]/[contesto]/services/insertGwRecord/[gwClassName]/modelAttribute
(esempio)
http://localhost:8080/webclient/services/insertGwRecord/test_class/modelAttribute
```

Body:

Selezionare l'opzione *form-data* ed inserire i dati da inviare al servizio, seguendo una struttura chiave/valore. Stavolta sarà necessario strutturare la chiave in modo specifico, cosicché possa essere interpretata come un oggetto `FormData` (uno specifico tipo di oggetto di Geoweb), ed inserire i dati in una mappa chiamata *editFormManager* (la quale è posizionata in profondità all'interno di questo oggetto). Inoltre, a differenza del servizio 'insertGwRecord', in questo non sarà possibile utilizzare dei file come valori.

(esempio)

Key	Value
mapFormData[editFormManager][itemId]	-1
mapFormData[editFormManager][10251]	005
mapFormData[editFormManager][10258]	fff

Inserisce un nuovo record con "cod_resource" = "005" e "fax" = "fff"

NOTA: In questo specifico servizio va notato che gli utenti non possono inviare dati utilizzando i "columnName" dei record che intendono inserire. Infatti, sarà necessario utilizzare i "gwid" degli attributi relative ai record: in questo esempio, "itemId" rappresenta il gwid della colonna "id_resource", mentre "10251" rappresenta il gwid della colonna "cod_resource" e "10258" rappresenta il gwid della colonna "fax".

NOTA: Durante la procedura di inserimento è comunque necessario fornire un "itemId" per l'oggetto che si intende inserire. Questo, per convenzione, è un valore fisso a -1, in modo tale che i servizi richiesti ed i metodi che ne conseguono possano essere in grado di riconoscere questo come un nuovo oggetto che non ha ancora un codice identificativo.

2.1) updateGwRecord (multipart/form-data)

		Esempi
Title	Inserimento di un record (di una classe)	
Method	POST	
URL	http://[indirizzo]/[contesto]/services/updateGwRecord/[**className**]	http://localhost:8080/webclient/services/updateGwRecord/test_class
Content-Type	multipart/form-data	
Body	`Content-Disposition: form-data; name="primaryKeyColumn" keyColumn ------WebKitFormBoundary--, Content-Disposition: form-data; name="key2" value2 -------WebKitFormBoundary--`	
Success Response Content	{“success”:[boolean],“description”:“[messaggio_testuale]”}	{success:true,“description”:“Operazione eseguita con successo”}

updateGwRecord (multipart/form-data) - TEST

Indirizzo della richiesta:

```
POST - http://[indirizzo]/[contesto]/services/updateGwRecord/[gwClassName]
(esempio) http://localhost:8080/webclient/services/updateGwRecord/test_class
```

Body:

Selezionare l’opzione *form-data* ed inserire ogni dato seguendo una struttura chiave/valore. Utilizzando Postman sarà possibile cambiare il tipo di chiave da *testo* a *file*, permettendo dunque di inviare anche file attraverso questa richiesta (il tipo e l’estensione del file possono variare in base al contesto della richiesta, e non deve obbligatoriamente trattarsi di un file di testo).

(esempio)

Key	Value
id_resource (Text)	12345
fax (Text)	ddd
test (File)	[Scegli] test_file.txt

Aggiorna il record con id_resource = 123454 nei campi “fax” e “test”

2.2) updateGwRecord (JSON)

		Esempi
Title	Inserimento di un record (di una classe)	
Method	POST	
URL	http://[indirizzo]/[contesto]/services/updateGwRecord/[**className**]	http://localhost:8080/webclient/services/updateGwRecord/test_class
Content-Type	application/json	
Body	{“primaryKeyColumn”:“primaryKeyValue”,“key2”:“value2”,...}	
Success Response Content	{“success”:[boolean],“description”:“[messaggio_testuale]”}	{success:true,“description”:“Operazione eseguita con successo”}

updateGwRecord (JSON) - TEST

Indirizzo della richiesta:

```
POST -
http://[indirizzo]/[contesto]/services/updateGwRecord/[gwClassName]/json
(esempio)
http://localhost:8080/webclient/services/updateGwRecord/test_class/json
```

Header:

In questo caso è necessario impostare un header che abbia *Content-Type* come chiave, e *application/json* come valore. Questo perché il servizio richiesto necessita obbligatoriamente di una mappa chiave/valore in ingresso, sotto forma di oggetto JSON.

Body:

Selezionare l'opzione *raw*. Apparirà un menu a tendina, tramite il quale l'utente potrà scegliere il formato e/o il tipo di testo che andrà ad utilizzare (nel nostro caso, sarà *JSON (application/json)*). Dopo di che, sarà possibile inserire i dati da inviare, ovviamente seguendo il formato JSON.

```
(esempio)
{
  “id_resource”: “54321”
  "cod_resource": "004",
  "fax": "eee"
}
```

Aggiorna il record con id_resource = 54321 nei campi “cod_resource” e “fax”

2.3) updateGwRecord (modelAttribute)

Indirizzo della richiesta:

```
POST -
http://[indirizzo]/[contesto]/services/updateGwRecord/[gwClassName]/modelAtt
ribute
(esempio)
http://localhost:8080/webclient/services/updateGwRecord/test_class/modelAttr
ibute
```

Body:

Selezionare l'opzione *form-data* ed inserire i dati da inviare al servizio, seguendo una struttura chiave/valore. Stavolta sarà necessario strutturare la chiave in modo specifico, cosicché possa essere interpretata come un oggetto `FormData` (uno specifico tipo di oggetto di Geoweb), ed inserire i dati in una mappa chiamata *editFormManager* (la quale è posizionata in profondità all'interno di questo oggetto). Inoltre, a differenza del servizio 'insertGwRecord', in questo non sarà possibile utilizzare dei file come valori.

(esempio)

Key	Value
mapFormData[editFormManager][itemId]	13245

Key	Value
mapFormData[editFormManager][10251]	005
mapFormData[editFormManager][10258]	fff

Aggiorna il record con id_resource = 13245 nei campi “cod_resource” e “fax”

NOTA: In questo specifico servizio va notato che gli utenti non possono inviare dati utilizzando i “columnName” dei record che intendono aggiornare. Infatti, sarà necessario utilizzare i “gwid” degli attributi relative ai record: in questo esempio, “itemId” rappresenta il gwid della colonna “id_resource”, mentre “10251” rappresenta il gwid della colonna “cod_resource” e “10258” rappresenta il gwid della colonna “fax”.