

# idgen support for WFS 1.1 transaction request

## The problem

The WFS 1.1 allows for multiple key generation strategies, and asks the server to implement at least one: "A specific web feature service implementation must support one of these methods of assigning features identifiers to new feature instances and may support all methods".

The methods are:

- **GenerateNew** (default): the web feature service shall generate unique identifiers for all newly inserted feature instances.
- **UseExisting**: in response to an <Insert> operation, a web feature service shall use the gml:id attribute values on inserted features and elements. If any of those IDs duplicates the ID of a feature or element already stored in the WFS, the WFS shall raise an exception.
- **ReplaceDuplicate**: a WFS client can request that the WFS generate IDs to replace the input values of gml:id attributes of feature elements that duplicate the ID of a feature or element already stored in the WFS, instead of raising an exception, by setting the idgen attribute of the InsertElementType to the value "ReplaceDuplicate"

## Our current support

This is less how FID are managed by our current datastore. I made an informal review, feel free to correct my findings:

Datastore	FID handling	Idgen supported
JDBC data stores (Postgis, Mysql, Hsql, DB2, Oracle)	FidMapper framework. Handles:* single varchar column <ul style="list-style-type: none"><li>• int column, maxinc, sequence or serial</li><li>• multicolumns, url-encode and concatenate values (basically, assigned in columns)</li></ul>	Depending on the fid mapper, <b>UseExisting</b> (single and multicolumn) or <b>GenerateNew</b> (single and multicolumn may work when the column is shown in the feature type, and will work with increment ones). ReplaceDuplicate may work with increment ones, but it's bound to be dangerous
Shapefile	Reading, FID is the row number, during writing whatever is in the FID is ignored, only order is imporant.	GenerateNew

ArcSDE	ArcSdeFeatureWriter seems to ignore the FID for new features, and uses whatever FID it produced for updating existing features. There's no control over feature id apparently. There is an improvement, though, pending to be merged back from complex-features branch to trunk, that would allow to use one or another strategy based on what exactly is happening on the database for each feature type.	GenerateNew Possible: GenerateNew, UseExisting
MifDataStore	FID is ignored during writing (MIFFile.Writer, line 1748)	GenerateNew
PropertyDataStore	Uses whatever ID has been assigned to features, no ID generation going on apparently?	UseExisting
GMLDataStore	No write support, so it doesn't apply.	GenerateNew
TigerDataStore	Hmm... does not provide a writer, on its own, so I guess it's read only	
VPFDataStore	Same as above	
WFSDDataStore	Hum.... does not provide a feature writer, so I guess it's read only as well	

There's no generic way to tell the above from the data stores.

A way could have been to try and write the feature as is, and check back the fid to see if it has been assigned or if an error occurs, but in some datastore the generated fid won't be seen until closing the transaction and re-reading the features (shapefile for example).

So no luck, it seems we need some in-deep Geotools change in any meaningful way.

What about a new method for the datastore that looks like:

```
isIdGenSupported(String featureTypeName, int idGenStrategy)
```

where the strategy is a list of the 3 modes?

The downside is this seems quite ad hoc, what we're really missing is a support for capabilities advertising in datastores (and this one would be one of the capabilities).

## A few notes on FID mappers

Depending on the FID mapper we may support one but not the others, and support may vary depending on how the FidMapper itself is configured.

- GenerateNew works with auto-generated primary keys (sequences, serials), and were the primary key is provided by the user and mapped among the attributes (so we look into the feature attribute and we can "generate" the fid
- UseExisting works only with SimpleFidMapper and MulticolumnFidMapper. If we try to use this with sequence based values, we are going to face trouble now or when the sequence reaches the selected value. (we could update the sequence to the value next to the biggest inserted, but that's a recipe for disaster since we may end up very near to the sequence overflow).
- ReplaceDuplicate does not really work with any datastore/fidmapper in the case were an actual replacement is needed afaik. Unless we create a new kind of fid mapper that uses big strings and a GUID generator. That may work for JDBC data stores, but it's really ugly IMHO...

## Actions to solve this problem

Have datastores advertise their FID management capabilities and support at least one method (the other possible solution would have been to check IDs after the fact and make sure they did match with the request, but unfortunately WFS allows generated IDs to be checked only after the transaction is complete).