GeoTools filter cleanup

Introduction

GeoTools classes and tests still use the old org.geotools.filter.Filter class a lot, despite it being deprecated for at least two years. We must put an end to this.

The rough idea is relatively simple:

1. find and kill any usage point of the old GeoTools filter api.
2. remove the GeoTools interfaces from the filter implementations
3. move the GeoTools filters into the legacy module

Module status

Here is a list of modules and the filter status in them:

Module	Uses Filter*	Cleanup done	Notes
library\api	⚠️		The old interfaces reside there
library\coverage			
library\cql			Apparently not, some deeper investigation needed
library\data	⚠️		
library\jdbc	⚠️		Big user, not clear if it's worth fixing it thought
library\legacy	⚠️		This one does not need fixes, it's legacy

library\main	⚠	In progress	• FilterFactory switch:
			  • ExpressionParser needs to know function argument count in advance
			  • FilterSAXParser/ExpressionSAXParser/LogicSAXParser needs redesign, now it creates the filters, empty, on the leading edge, and fills them as parsing continues
			  • FilterFactoryFinder simply needs to go the way of the dodo (move into legacy module?)
			  • FilterCapabilities Test should go away along with gt2 FilterCapabilities (same as above)
			• Expression removal:
			  • EnvironmentVariable, what shall we do with it?

library\metadata			
library\referencing			
library\render	⚠		
library\sample-data			
library\xml	⚠		Is this module actually being in current usage anymore? It is a heavy user of the old filters in the FilterEncodingPreProcessor class (which is a deprecated FilterVisitor)
plugin\arcgrid			

plugin\arcsde			
plugin\db2			Ideally, won't be fixed. We switch to the jdbc-ng one, and make this one either go away, or depend on the legacy module for one release cycle before removing it? The only question is, will we have a jdbc-ng DB2 module and have it supported as well?
plugin\epsg-access			
plugin\epsg-extension			
plugin\epsg-hsql			
plugin\epsg-postgresql			
plugin\epsg-wkt			
plugin\geometry			
plugin\geotiff			
plugin\gtopo30			
plugin\image			
plugin\imagemosaic			
plugin\imagemosaic-jdbc			
plugin\imagepyramid			
plugin\postgis	⚠		Ideally, won't be fixed. We switch to the jdbc-ng one, and make this one either go away, or depend on the legacy module for one release cycle before removing it? The only question is, will we have a jdbc-ng DB2 module and have it supported as well?
plugin\property			

plugin\referencing3D			
plugin\shapefile			
plugin\wfs			
plugin\wms			
unsupported\caching			
unsupported\community-schemas			
unsupported\coverageio			
unsupported\coverageio-hdf			
unsupported\coverageio-netcdf			
unsupported\coveragetools			
unsupported\directory			
unsupported\epsg-oracle			
unsupported\example			
unsupported\geomedia			
unsupported\geometry			
unsupported\geometryless			
unsupported\gml			
unsupported\gml3			
unsupported\go			
unsupported\gpx			
unsupported\h2			

unsupported\hsql	?		
unsupported\imageio-ext-gdal			
unsupported\imagemosaic-jdbc			
unsupported\jts-wrapper	?		
unsupported\mappane			
unsupported\mif	⚠		
unsupported\mysql	⚠		Won't be fixed. We switch to the jdbc-ng one, and make this one either go away, or depend on the legacy module for one release cycle before removing it?
unsupported\notifying-collections	?		
unsupported\ogc			
unsupported\ogr	?		
unsupported\oracle-spatial	⚠		Won't be fixed. We switch to the jdbc-ng one, and make this one either go away, or depend on the legacy module for one release cycle before removing it?
unsupported\postgis-versioned			
unsupported\process			
unsupported\repository			
unsupported\sql-datastore	?		
unsupported\temporal			

unsupported\tiger			
unsupported\tile	❓		
unsupported\vpf			
unsupported\widgets-swing-pending	❓		
unsupported\wps	❓		
unsupported\xml-gpx			
extension\brewer	⚠️		
extension\graph			
extension\openoffice			
extension\shapefile-renderer	⚠️		
extension\validation	⚠️		
extension\widgets-swing			
extension\xsd-core			
extension\xsd-filter			
extension\xsd-gml2			
extension\xsd-gml3	❓		
extension\xsd-kml			
extension\xsd-ows			
extension\xsd-sld	⚠️	✅	Cleaned up
extension\xsd-wfs			
extension\xsd-wps			
extension\coverage-use			
extension\example			

extension\introduction			
extension\libraryJTS			
extension\mappane-use			
extension\referencing			
extension\svgsupport			
extension\xml-po			
demo\coverage-use			
demoexample			
demo\introduction			
demo\libraryJTS			
demo\mappane-use			
demo\referencing			
demo\svgsupport			
demo\xml-po			

*: ⚠️ if the module was a org.geotools.filter.Filter user at the time the work started, ❓ if the status is unknown due to the module being out of the build. The column was filled looking at the result of searches for references to org.geotools.filter.Filter and sub-interfaces, org.geotols.filter.FilterFactory, and presence of "import org.geotools.filter", and only for the modules I had loaded in Eclipse at the time of writing (all the ones in the build plus a few selected ones that are not normally in the build).
As such it is not accurate, thought it's believed to be representative enough to get the work started.

Steps taken

- Find and eliminate all explicit references to org.geotools.filter.FilterFactory (replace with GeoApi FilterFactory)
- Find and eliminate all references to FilterFactoryFinder (replace with CommonFactoryFinder)
- Find all references to Filter, Filter.ALL, Filter.INCLUDE, and replace with GeoApi filter
- Same as above for all Filter GT2 subinterfaces

Issues and tasks

- In the `main` module we have a ExpressionBuilder class that needs to know how many arguments a function will take before proceeding into the parsing. Now, in GeoApi there is completely no way to know the argument and types of a function. To put it into other terms, we have an extensible API that's not discoverable, inspectable, with machine code. This is a major issue that has to be fixed at the GeoApi level. Oh, ExpressionBuilder is deprecated, we could just move it out into legacy, but that does not change the flawed nature of GeoApi function.

- Remove all classes that do use org.geotools.filter.FilterVisitor and provide a replacement that uses org.opengis.filter.FilterVisitor instead
- Remove all users of the old FilterCapabilities class, and replace with GeoApi one

History

A history of the changes performed. If you find that a commit breaks your code feel free to revert it, and send a mail to gt2-devel saying what the problem was.

Author	Revision	Notes
aaime	31681	Removed all explicit usages of org.geotools.filter.FilterFactory from the main module
aaime	31682	Removed all explicit usages of FilterFactoryFinder from the main module
aaime	31747	Cleaned up the xsd-sld module
aaime	31748	Fixing render (factory and factory finder)
aaime	31748	Fixing shapefile-render (factory and factory finder)
aaime	31750	Fixing DataTestCase
aaime	31751	Cleaned up the jdbc-ng module
aaime	32247	Cleaned up the brewer module
aaime	32248	Cleaned up the validation module
aaime	32249-32252	Minor fixes related to Filter imports