Datastore Capabilities API

Motivation:	There is a need for an extensible API to query the datastore about what it can do (and adding a delete method in the process)
Contact:	Andrea Aime
Tracker:	http://jira.codehaus.org/browse/GEOT-XXXX
Tagline:	Datastore capabilities

Description

Dealing with datastores often means adding instanceof and try/catch blocks around the code to figure out what the data store can - actually do. There is no straight way to query the datastore about simple things like:

- is it possible to create a new feature type?
- is it possible to update a feature type?
- is type "xyz" read only? Can we do locking on top of it?
- what are the data types supported by a certain store?
- how is createSchema going to mangle the feature type definition provided by the user?

In most of the cases to actually know if something is supported one has actually to try to do it, which is bad as requires to add un-necessary try/catch blocks and does not allow to inspect the capabilities to build a proper user interface with only the available commands.

This proposal tries to address the above by introducing a new class, DataCapabilities, that can be used to perform the above questions. The DataAccess interface will be extended by adding a new method:

```
/**
    * Returns the store capabilities
    */
   DataCapabilities getCapabilities();
```

where DataCapabilities is defined as:

```
public class DataCapabilities {
   // omitted fields

   public boolean supportsCreateSchema() {
      ...
   }

   public boolean supportsUpdateSchema() {
      ...
   }

   public boolean isReadOnly(Name schema) {
      ...
   }

   public boolean supportsLocking(Name
schema) {
      ...
   }

}
```

DataCapabilities has been defined as a class so that it can be extended in the future without breaking the API.

Status

This proposal is under construction.

Voting has started:

- Andrea Aime +1
- Ben Caradoc-Davies +1
- Christian Mueller +1
- Ian Turton +1
- Justin Deoliveira

- Jody Garnett +1
- Michael Bedward +1
- Simone Giannecchini +0

Tasks

This section is used to make sure your proposal is complete (did you remember documentation?) and has enough paid or volunteer time lined up to be a success

	no progress	✅	done	❌	impeded	⚠	lack mandate /funds/ti me	❓	voluntee r needed

1. API changed based on BEFORE / AFTER
2. Update default implementation
3. Update wiki (both module matrix and upgrade to to 2.5 pages) |
4. Remove deprecated code from GeoTools project
5. Update the user guide
6. Update or provided sample code in demo
7. review user documentation

API Changes

BEFORE

This is some bits of code trying to create a schema and then trying to insert some data

```
SimpleFeatureType ft = ...;
   DataStore ds = ...;
   try {
      ds.createSchema(ft);
   } catch(UnsupportedOperationException e)
{
     System.out.println("Whoops, creation
not supported!");
   } catch(IOException e) {
     System.out.println("Failures occurred
during type creation");
   }

   ...
   try {
     FeatureSource source = (FeatureSource)
ds.getFeatureSource(name);
     if(source instanceof FeatureStore) {
         FeatureStore store = (FeatureStore)
source;
         // write something
     } else {
         System.out.println("Ops, the
feature type cannot be written to!");
     }
   } catch(IOException e) {
     System.out.println("Ops, the feature
type cannot be written to!");
   }
```

AFTER

By quering the capabilities it's possible to know in advance what can be done without having to actually try to do it, meaning it's possible to build a GUI that only has the operations that are actually available.

```
DataStore ds = ...;
   DataCapabilities caps =
ds.getCapabilities();
   if(!caps.supportsCreateSchema()) {
     System.out.println("Creation will be
disabled in the GUI, as it's not
supported");
   } else {
     SimpleFeatureType ft = ...;
     try {
        ds.createSchema(ft);
     } catch(IOException e) {
       System.out.println("Failures occurred
during type creation");
     }
   }

   ...

   if(caps.isReadOnly(name)) {
      System.out.println("Ops, the feature
type cannot be written to!");
   } else {
     try {
       FeatureStore store = (FeatureStore)
ds.getFeatureSource(name);
```

```
    // write something
  } catch(IOException e) {
    System.out.println("Ops, the feature
type cannot be written to!");
     }
   }
```

Documentation Changes

We may need to update the demos.