Login and Logout

So that the User can work with a database, He must connect to it first. As we know the work with the database is possible via the Database Layer only. Furthermore, we can use the Communication Layer to work with a remote Database Layer, that can be created for us by a remote Server.

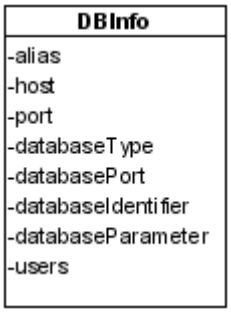
In this section we will discuss the process of a new Database Layer creation from the Client's perspective, i.e. how the client connects the Server and obtains a new Database Layer.

Participants

All participants can be found in the net.sf.plantlore.client.login and net.sf.plantlore.middleware package. Almost all the business logic of this operation is concentrated in the Login.java.

DBInfo

The Database Info, DBInfo in short, holds information about the database and where it is located. Its source is in the net.sf.plantlore.client.login.DBInfo.java.



Information from this holder object are presented to the User who can alter them at will. They are used when the creation of a new Database Layer is required.

DBLayerFactory

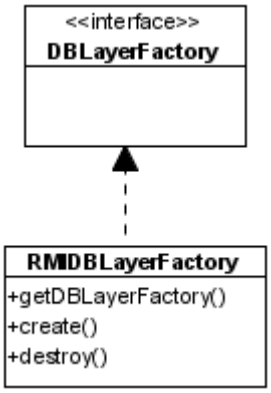
DBLayerFactory is an interface that allows the client to control a Database Layer Factory which is an object that shields other parts from the particular implementation of the Database Layer creation and destruction. There should be only one Database Layer Factory at a time. It can be found in net.sf.plantlore.middleware.DBLayerFactory.java.

RMIDBLayerFactory

RMIDBLayerFactory can create and destroy Database Layers. It creates them based on the supplied DBInfo. It creates them either in the current JVM if the database runs on the local computer, or contacts the RemoteDBLayerFactory and asks it to create a new Database Layer for it.

RMIDBlayerFactory is implemented as a Singleton in order to be sure there is only one instance in the Application at a time. Use the **getDBlayerFactory()** method to obtain an instance of RMIDBLayerFactory.

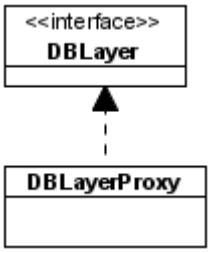
The source is in net.sf.plantlore.middleware.RMIDBLayerFactory.java.



DBLayerProxy

The Database Layer Proxy is a wrapper of a spinoff of the Dabatase Layer. When the DBLayerProxy is asked to wrap a newly created Database Layer, it first creates its spin off, which is a safety object that ensures that every call will be performed in its own thread. And after that it wraps the spinoff and ads a verification that methods of the Database Layer are safe to call, which they generally are until the Logout is performed.

The source is in the net.sf.plantlore.client.login.Login.java#DBLayerProxy.

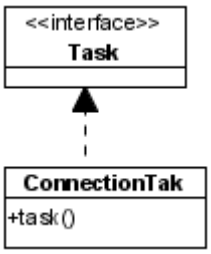


Connection Task

The Connection task extends the Task. To learn more about tasks see the Section **Tasks and Dispatcher**. It can create and initialize a new Database Layer using the supplied information and the DBLayerFactory. It instructs the DBLayerProxy to wrap the newly created Database Layer from the Database Layer Factory.

The second important task is to notify the Application that a new Database Layer has been created and instruct them to obtain it from Login. See the Section **The GUI Communication** for futher details.

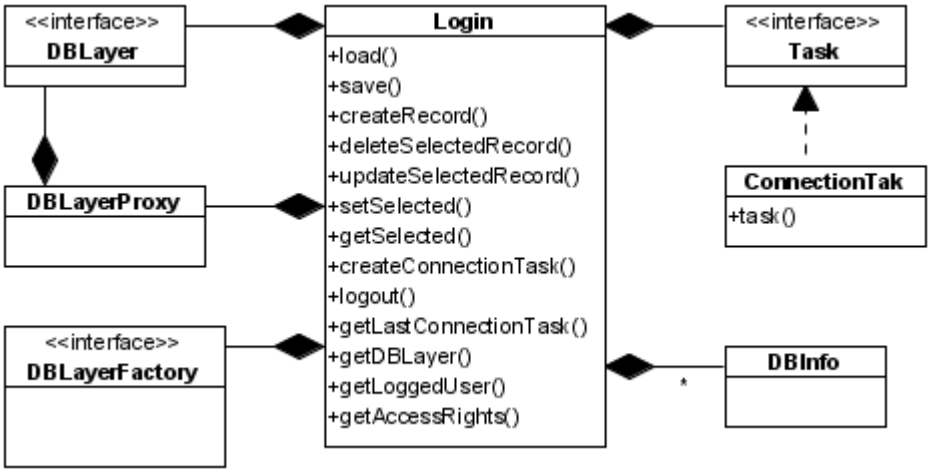
The source is in the net.sf.plantlore.client.login.Login.java#ConnectionTask.



Login

Login acts mostly as a Connection Task factory. It stores information required for the task creation and creates it when it is asked to. Login is equipped with a Database Layer factory that is used to obtain the Database Layer in the Connection task and then it is wrapped by the DBLayerProxy. Every time Login is asked to return the newly created DBLayer, the DBLayerProxy is returned in its stead. All parts of the Presentation Layer work with this wrapper.

The source is in the net.sf.plantlore.client.login.Login.java.



Login is also responsible for the management of DBInfos. It can create the DBInfo, alter it, or delete it.

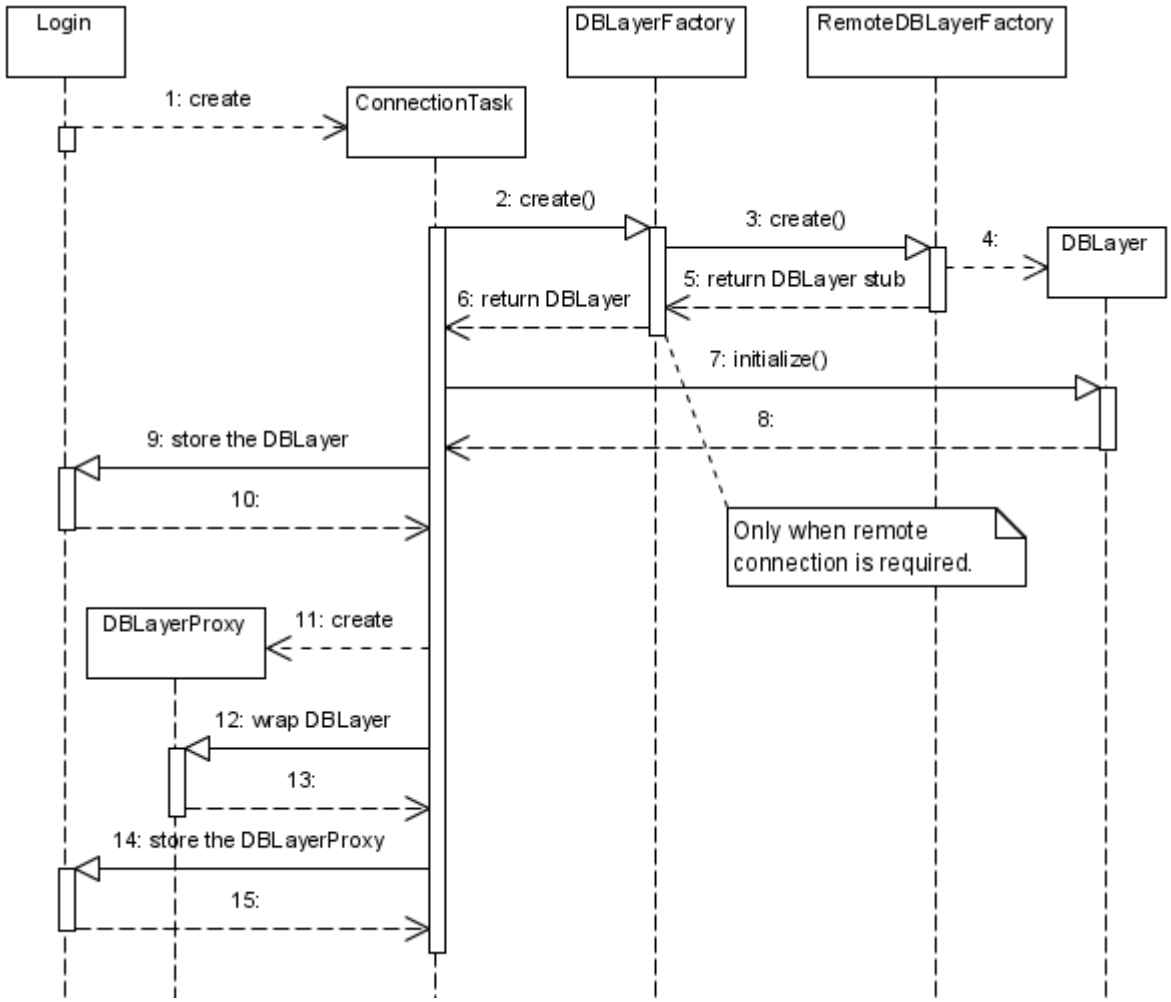
The last responsibility is to perform a proper logout, that is take the Database Layer as it was returned from the DBLayer Factory and ask that DBLayerFactory to destroy it. After that the Database Layer is not accessible.

Interaction

In this section we will depict the exact process of the new Database Layer creation and its destruction.

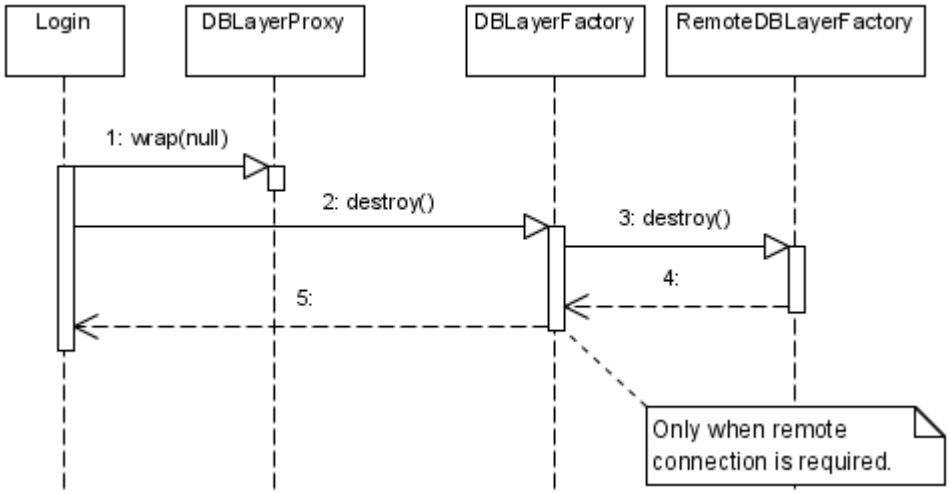
Creating and Initializing a New Database Layer

In the picture below there you can see how the remote Database Layer is created.



Destroying the Database Layer

The destruction of the Databse Layer.



The DBLayerProxy is instructed to wrap null which is a signal to reject all attempts to communicate with the wrapped Database Layer.