# Express与Express-G

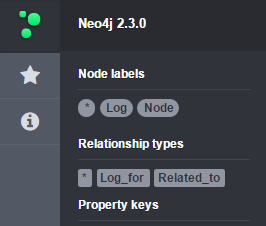
1. Express抽象语法树存储接口
   1. Neo4j实体关系存储

实体共有三种：Node ,Log, User，其中User是没有用到的，原因是Neo4j Community暂时是单用户登录的，企业版支持高可用性（HA）;

关系分为两种：Log\_for,Related\_to。其中Log\_for是创建每个正常树节点自动生成的，包括ip等信息；Related\_to是树节点之间的单向关系。

具体功能如下图





* 1. 创建抽象语法树节点

每次创建节点则生成两类节点：树节点以及对应的日志节点，日志节点记录每个节点上进行的操作

/\*\*

\* 创建节点

\* **@param** name 创建的节点名称

\* **@param** children\_num 子节点的个数

\* **@return** 返回创建的节点id

\*/

**public** **int** creatNode(String name,**int** children\_num);

/\*\*

\* 写日志信息

\* **@param** id 日志节点

\* **@param** ip

\* **@param** time

\* **@param** operation 操作内容

\*/

**private** **void** writeLog(**int** id,String ip,String time,String operation);

* 1. 创建抽象语法树节点间的关系

/\*\*

\* 根据节点id创建关系

\* **@param** id1

\* **@param** id2

\* **@return** 返回受影响的行数

\*/

**public** **int** createRelationshipTo(**int** id1,**int** id2);

* 1. Express查询接口
     1. 根据Neo4j图反向生成抽象语法树的接口

主要为查询图中根节点以及查询每个树节点的子节点的函数（子节点分为直接子节点与间接子节点）

/\*\*

\* 查询图中根节点的id

\* **@return**

\*/

**public** Integer getRoot();

/\*\*

\* 返回指定节点的直接子节点的id和name

\* **@param** id

\* **@return**

\*/

**public** List<Map<String, Object>> getDirectChildren(**int** id);

/\*\*

\* 返回指定节点的全部子节点

\* order by ID(m)指定是否要排序

\* **@param** id

\* **@return**

\*/

**public** List<Map<String, Object>> getChildren(**int** id);

/\*\*

\* 返回指定节点的直接子节点个数

\* **@param** id

\* **@return**

\*/

**public** **int** getDirectChildrenNum(**int** id);

/\*\*

\* 返回指定节点的全部子节点个数

\* **@param** id

\* **@return**

\*/

**public** **int** getChildrenNum(**int** id);

* + 1. 其他查询接口

/\*\*

\* 查询id节点的name

\* **@param** id

\* **@return**

\*/

**public** String getName(**int** id);

/\*\*

\* 寻找id的叶子节点 ,添加数值属性

\* **@param** id

\* **@return**

\*/

**public** String getLeaf(Integer id);

/\*\*

\* 登出

\*/

**public** **void** logout()

* 1. Express-G查询接口

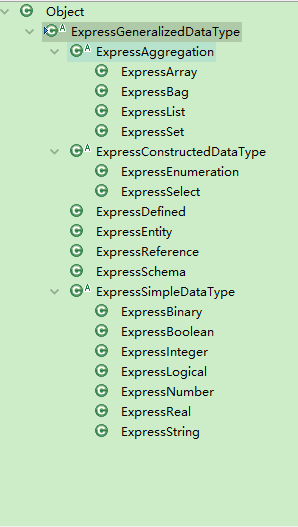
每种类型都对应一个ExpressXXX类以及对应的实例类XXXInstance，且每个类型的逻辑处理类为ExpressXXXDao。

例如string对应的类型类与实例类分别为ExpressString与StringInstance，则处理ExpressString与StringInstance逻辑处理类为ExpressStringDao，封装了所有的类型和实例的操作。

* + 1. 类型类



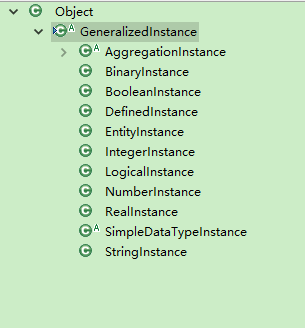
其继承关系如下



* + 1. 实例类



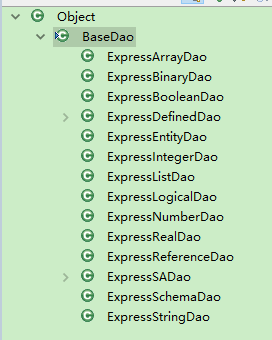
其继承关系如下



* + 1. 类型对应的类型的逻辑处理类



其继承关系如下



* + 1. 查询接口

提供ExpressSchema接口获取Schema的所有信息

**public** ExpressSchemaDao() {}

**public** ExpressSchemaDao(String ip) {}

/\*\*

\* 遍历图，返回图中所有的schema

\* **@return**

\*/

**public** List<ExpressSchema> getAllExpressSchema() {}

* + 1. 数据转换

通过Express语法对应的规则进行转换