**XDB 框架中的各种对应关系**

**数据库结构定义**

数据定义文件：gsx.xdb.xml.ftl

举例说明

定义数据结构  
<xbean name="User" maxjsonname="20">  
   <variable name="roleIds" type="list" value="long" jsonname="1"/> 所有角色  
   <variable name="createTime" type="long" jsonname="2"/> 用户创建时间  
   <variable name="lastLoginRole" type="long" jsonname="3"/> 上次登录的角色  
</xbean>  
​  
定义表结构  
<table key="int" name="user" value="User" lock="userlock" cacheCapacity="8192"/>

variable 为定义表属性

xbean 为定义数据结构

xtable 为定义表

**协议文件定义**

协议文件定义在以 protocol 结尾的文件夹中，通常关注 gameprotocol 下的文件即可

定义 Java Bean 结构  
<bean name="RoleInfo">  
   <variable name="roleid" type="long"/> ID  
   <variable name="rolename" type="string"/> 名称  
   <variable name="level" type="int" /> 等级  
   <variable name="school" type="int" />   职业  
   <variable name="lastLogintime" type="long" />最后登出时间  
   <variable name="fashions" type="list" value="int"/> value是时装id  
   <variable name="equips" type="list" value="int"/> value是身上装备id  
</bean>  
​  
定义获取角色列表的协议类  
<protocol name="CRoleList" type="802" maxsize="65535">  
</protocol>  
​  
定义返回角色列表的协议类  
<protocol name="SRoleList" type="803" maxsize="65535" tolua="1">  
   <variable name="roles" type="list" value="protocol.role.RoleInfo"/>  
   <variable name="lastLoginRoleid" type="long"/> 上次的登录的角色id  
</protocol>

bean 为定义 Java Bean 类结构，结构可以嵌套

protocol 为定义协议类，协议类中的 type 为协议 id，maxsize 为协议最大大小，tolua 为是否生成 lua 代码

·C 开头的协议为客户端发往服务器的协议

·S 开头的协议为服务器发往客户端的协议

variable 为定义类属性

**配置文件**

配置文件由策划修改和生成，具体位置在服务器文件夹的 gamedata/xml 中

<bean name="Sdayawardconfig" from="r日常活跃度奖励表.xlsx" genxml="server">  
   <variable name="id" type="int" fromCol="ID"/>  
   <variable name="activenum" type="int" fromCol="活跃度值"/>   
   <variable name="activeaward" type="vector" value="int" fromCol="对应奖励1,对应奖励2"/>  
   <variable name="activeawardnum" type="vector" value="int" fromCol="奖励数量1,奖励数量2"/>  
</bean>

**gbeans**

gbeans 中配置了 office 表格文件和 xml 文件的对应关系，每个表格文件都会通过 xml 生成一个匹配 xml 结构的 Java 类用来读取数据。

比如上面的 xml 就会生成一个名为 Sdayawardconfig 的 Java 类：

package exceldata.award;  
​  
​  
public class Sdayawardconfig implements mytools.ConvMain.Checkable ,Comparable<Sdayawardconfig>{  
​  
public int compareTo(Sdayawardconfig o){  
return this.id-o.id;  
}  
​  
  
  
static class NeedId extends RuntimeException{  
​  
/\*\*  
\*   
\*/  
private static final long serialVersionUID = 1L;  
  
}  
public Sdayawardconfig(){  
super();  
}  
public Sdayawardconfig(Sdayawardconfig arg){  
this.id=arg.id ;  
this.activenum=arg.activenum ;  
this.activeaward=arg.activeaward ;  
this.activeawardnum=arg.activeawardnum ;  
}  
public void checkValid(java.util.Map<String,java.util.Map<Integer,? extends Object> > objs){  
}  
/\*\*  
\*   
\*/  
public int id  = 0 ;  
  
public int getId(){  
return this.id;  
}  
  
public void setId(int v){  
this.id=v;  
}  
  
/\*\*  
\*   
\*/  
public int activenum  = 0 ;  
  
public int getActivenum(){  
return this.activenum;  
}  
  
public void setActivenum(int v){  
this.activenum=v;  
}  
  
/\*\*  
\*   
\*/  
public java.util.ArrayList<Integer> activeaward ;  
  
public java.util.ArrayList<Integer> getActiveaward(){  
return this.activeaward;  
}  
  
public void setActiveaward(java.util.ArrayList<Integer> v){  
this.activeaward=v;  
}  
  
/\*\*  
\*   
\*/  
public java.util.ArrayList<Integer> activeawardnum ;  
  
public java.util.ArrayList<Integer> getActiveawardnum(){  
return this.activeawardnum;  
}  
  
public void setActiveawardnum(java.util.ArrayList<Integer> v){  
this.activeawardnum=v;  
}  
  
  
};

可以看到，这个类的属性和上面的 xml 项是一一对应的。

**配置数据 xml**

auto 中是策划通过 office 表格文件生成的 xml 数据，Java 类具体读取的就是这些 xml 数据。

同样以 exceldata.award.Sdayawardconfig.xml 举例：

<?xml version="1.0" encoding="utf-8" ?>  
<tree-map>  
 <no-comparator/>  
 <entry>  
   <int>1</int>  
   <exceldata.award.Sdayawardconfig>  
     <id>1</id>  
     <activenum>20</activenum>  
     <activeaward>  
       <int>360002</int>  
       <int>360034</int>  
     </activeaward>  
     <activeawardnum>  
       <int>50</int>  
       <int>3</int>  
     </activeawardnum>  
   </exceldata.award.Sdayawardconfig>  
 </entry>  
 <entry>  
   <int>2</int>  
   <exceldata.award.Sdayawardconfig>  
     <id>2</id>  
     <activenum>40</activenum>  
     <activeaward>  
       <int>360035</int>  
       <int>360046</int>  
     </activeaward>  
     <activeawardnum>  
       <int>1</int>  
       <int>1</int>  
     </activeawardnum>  
   </exceldata.award.Sdayawardconfig>  
 </entry>  
 <entry>  
   <int>3</int>  
   <exceldata.award.Sdayawardconfig>  
     <id>3</id>  
     <activenum>60</activenum>  
     <activeaward>  
       <int>360002</int>  
       <int>360033</int>  
     </activeaward>  
     <activeawardnum>  
       <int>100</int>  
       <int>50</int>  
     </activeawardnum>  
   </exceldata.award.Sdayawardconfig>  
 </entry>  
 <entry>  
   <int>4</int>  
   <exceldata.award.Sdayawardconfig>  
     <id>4</id>  
     <activenum>80</activenum>  
     <activeaward>  
       <int>360094</int>  
       <int>360067</int>  
     </activeaward>  
     <activeawardnum>  
       <int>1</int>  
       <int>1</int>  
     </activeawardnum>  
   </exceldata.award.Sdayawardconfig>  
 </entry>  
 <entry>  
   <int>5</int>  
   <exceldata.award.Sdayawardconfig>  
     <id>5</id>  
     <activenum>100</activenum>  
     <activeaward>  
       <int>360147</int>  
       <int>360068</int>  
     </activeaward>  
     <activeawardnum>  
       <int>10</int>  
       <int>1</int>  
     </activeawardnum>  
   </exceldata.award.Sdayawardconfig>  
 </entry>  
</tree-map>

这些数据会在服务器起服的时候被读取，每个 entry 都会构造一个 Sdayawardconfig 类对象，然后会以 key-value 的形式存在服务器缓存的 map 中，key 则是 entry 中的 <int> 项的值，value 则是根据 <exceldata.award.Sdayawardconfig> 项中的数据构造出的 Sdayawardconfig 对象