EZDML 配置文件说明

http://www.ezdml.com

——20191027 BY HUZ

——本文最后修订日期：20200411，对应 EZDML 版本：3.01。

本章节比较枯燥乏味，就是告诉大家有哪些文件是做什么用的，大家可大概扫描一下，大致了解下可以做哪些坏事，具体内容则可在用到时再看。没问题就不要细看了。

EZDML 没有做设置界面（我都懒到了什么地步啊），而配置文件就比较多，包括：

- ezdml.INI——配置文件，作用当然是程序的主要配置了
- dict.txt——词典文件，这个是用于自动大小写识别的，一般不需要改
- MyDict.txt——自定义词典文件，这个也是用于自动大小写识别，是供用户自定义添加自己的词用的，优先级比 dict.txt 高
- Templates\reserved_words.txt——保留关键字字典文件，你可以根据需要修改它。如果你的表名或字段名有跟这里的冲突，则生成 SQL 时会产生警告：表名/字段名可能非法
- Templates\new_table.txt——新建表的配置文件，新建表要包含哪些字段，就在这里配置了
- Templates\add_system_fields.txt——表字段列表右键菜单添加系统字段的配置
- GlobalScript.ps_——全局事件脚本，你可以在这接管生成 SQL、弹出窗口等事件
- Templates\CustomTableDef.js_/.ps_——自定义表属性界面脚本
- Templates\CustomFieldDef.js_/.ps_——自定义字段属性界面脚本
- Templates\ export_doc_html.js_/.ps_——导出 WORD、HTML 脚本
- CustomTools*——自定义工具
- Templates*.js/*.pas——自定义生成脚本

因为配置文件较多，因此，不建议将 EZDML 安装在受保护的系统盘。

一、 ezdml.INI——配置文件

位于主目录下。正常程序配置都会做个界面，不过既然都是程序员我就偷懒了，改 INI 也很方便。执行菜单命令“工具/修改 INI 配置”即可打开 INI 文件进行编辑（注：x64 版已经增加了设置界面，没有 INI 菜单了）。

目前 INI 可配置的项目如下：

```
[DefaultFieldTypes]
1=String:VARSTR(2000)
2=Integer:DECIMAL

[CustFieldTypes]
```

```
1=BigInt
2=Decimal
3=TestUnk:VARCHAR(500)

[CustDataTypeReplaces]
1=VARCHAR2:NVARCHAR2
2=NUMBER(10):DECIMAL
3=%TEXT%:NCLOB

[Options]
AutoSaveMinutes=5
SaveTempFileOnExit=1
FieldNameMaxDrawSize=64
FieldTypeMaxDrawSize=48
TableFieldMaxDrawCount=50
TableFieldMaxDrawCount=50
HugeModeTableCount=500
MaxRowCountForTableData=25
LogicNamesForTableData=0
CreateSeqForOracle=0
OCILIB=D:\oracle\ora112\instantclient\oci.dll
QuotReservedNames=1
BackupBeforeAlterColumn=0
WriteConstraintToDescribeStr=1
AddColCommentToCreateTbSql=1
EnableCustomPropUI=0
LANG=
AppDefFontName=default
AppDefFontSize=0
AppFixWidthFontName=default
AppFixWidthFontSize=0
DmlGraphFontName=default


[TbNamePrefixDefs]
1=CT_
2=IMS_
```

解释一下 INI 配置：

[DefaultFieldTypes] ——修改缺省数据类型对应的物理类型

1=String:VARSTR(2000) ——修改逻辑类型名为 String 的物理类型名为 VARSTR，默认长度为 2000

2=Integer:DECIMAL ——修改逻辑类型名为 Integer 的物理类型名为 DECIMAL，无默认长度

[CustFieldTypes] ——添加额外的数据类型到类型下拉列表

1=BigInt

2=Decimal

3=TestUnk:VARCHAR(500) ——自定义数据类型 TestUnk，对应的物理数据类型为 VARCHAR，默认长度为 500

[CustDataTypeReplaces] ——指定替换字段类型

1=VARCHAR2:NVARCHAR2 ——将 VARCHAR2 替换为 NVARCHAR2（全字匹配）

2=NUMBER(10):DECIMAL ——将 NUMBER(10)替换为 DECIMAL（全字匹配）

3=%TEXT%:NCLOB ——将 TEXT 替换为 NCLOB（模糊匹配，类似 SQL 中的 like 操作）

4=%RAW:BLOB ——将 RAW 替换为 BLOB（左边模糊匹配，右边精确匹配，类似 SQL 中的 like 操作）

[Options] ——选项，示例值为默认值

AutoSaveMinutes=5 ——定时每 5 分钟自动保存，保存为用户临时目录.~dmh 文件，并且会保留历史文件（无修改时不会保留历史文件），为 0 则不保存

SaveTempFileOnExit=1 ——退出时是否自动保存临时文件（.~dmh0），为 0 时每次退出会提示用户是否保存

FieldNameMaxDrawSize=64 ——在模型图上允许显示的最大字段名长度

FieldTypeMaxDrawSize=48 ——在模型图上允许显示的最大字段类型长度

TableFieldMaxDrawCount=50 ——在模型图上单个表允许显示的最大字段数量

HugeModeTableCount=500 ——文件中所有表的数量达到此值后将认为文件巨大,将不再自动保存和生成临时表；单个模型图的表数量达到此值后，在自动布局和布线时将采用最简方式提高速度

MaxRowCountForTableData=25 ——表属性页中查看表数据的最大记录数量

LogicNamesForTableData=0 ——表属性页中查看表数据时是否使用逻辑名

CreateSeqForOracle=0 ——是否为 ORACLE 表整形主键生成序列号

OCILIB=D:\oracle\ora112\instantclient\oci.dll ——指定 OCI 库文件的路径（用以前的 OCIDLL 配置也兼容），用于 instant client 或装了多个 ORACLE home 时，请注意 oci.dll 有 32 位和 64 位之分，EZDML for win32 一定要是 32 位的 OCI，而 EZDML for x64 的 oci.dll/.dylib/.so 一定要是 64 位的（数据库服务默认自带的 oci 一般都是 64 位的）。默认情况下，EZDML 会从程序所在目录和系统搜索路径查找 OCI。修改此设置可能需要重启程序

QuotReservedNames=1 ——是否为系统保留字名称加上引号（ORACLE）或中括号（SQLSERVER）

BackupBeforeAlterColumn=0 ——字段类型有变化时，是否要生成备份字段的 SQL

WriteConstraintToDescribeStr=1 ——是否将约束输出到描述字

AddColCommentToCreateTbSql=1 ——是否将注释输出到创建表的 SQL 中

EnableCustomPropUI=0 ——是否启用自定义的表和字段属性页

LANG= ——指定界面语言（如 en、ch，默认为空表示自动，仅限 64 位版本 EZDMLx）

AppDefFontName=default ——界面字体名称（仅限 64 位版本 EZDMLx）

AppDefFontSize=0 ——界面字体大小（仅限 64 位版本 EZDMLx）

AppFixWidthFontName=default ——SQL 及代码字体名称（仅限 64 位版本 EZDMLx）

AppFixWidthFontSize=0 ——SQL 及代码字体大小（仅限 64 位版本 EZDMLx）

DmlGraphFontName=default　——模型图字体名称（仅限 64 位版本 EZDMLx）

[TbNamePrefixDefs] ——已知表名前缀列表。生成索引名称时，如果发现表名带已知前缀，会去掉前缀，以减短索引名称

1=CT_

2=IMS_

再强调一下：x64 版已经有设置界面了，不需要手工折腾 INI 了。

二、 dict.txt——词典文件

这个是用于自动大小写识别的，里面就是一堆的常见单词，一行一个，全小写（大写也会自动转成小写），看了下有 4 千多个，我也不记得是从哪里抄来的了。你可以自行增删编辑此文件。

分词的处理过程，大概就是从左到右尽量分出长的单词。

三、 MyDict.txt——自定义词典文件

这个也是用于自动大小写识别，是供用户自定义添加自己的词用的，优先级比 dict.txt 高，而且它跟 dict.txt 有区别，它可以大小写混合，它不参与分词处理，而是直接查找替换，只要名字中匹配到相应的单词，则直接替换。格式的话也是一行一个。

假设有一个字段 USER_REALNAME，正常大小写识别结果是 User_RealName；这时我们在 MyDict.txt 中加一行：USer_realN，则最终识别结果变为 USer_realName，里面的大小写格式全保留。

注：x64 版里把这个也加到设置界面了。

四、 reserved_words.txt——保留关键字字典文件

位于 Templates 目录下。保留关键字字典文件，你可以根据需要修改它。如果你的表名或字段名有跟这里的冲突，则生成 SQL 时会产生警告: 表名/字段名可能非法。

这个文件的内容是从网上找了所有数据库的关键词列表拼一块再去重后形成的。

如果不想要这个功能，可以删除掉字典文件内容。

五、 new_table.txt——新建表的配置文件

位于 Templates 目录下。新建表的配置文件，新建表要包含哪些字段，就在这里配置了。格式就是我们的描述字，默认内容如下：

Id(编号)	PKInteger
Pid(父编号)	Integer
Rid(关联编号)	FKInteger
Name(名称)	String(255)
Memo(注释)	String

```
TypeName(类名)              String
CreatorId(创建人编号)       Integer     //<<非空,普通索引>>当前用户 ID
CreatorName(创建人姓名)     String
CreateDate(创建日期)        Date        //<<Default:sysdate>>默认为当前时间
DataLevel(数据级别)         Enum        //0 正常 1 审核中 2 被否决 -1 已删除 -2
草稿
OrderNo(排序号)             Float
```

注：~~英文版将只使用括号前的物理名称。~~各语言可以有自己的版本。

六、 add_system_fields.txt

位于 Templates 目录下。表字段列表右键菜单添加系统字段的配置。跟新建表的配置差不多，只不过它是用在添加字段的时候。

七、 GlobalScript.ps_——全局事件脚本

位于主程序目录下。你可以在这接管生成 SQL、弹出窗口等事件。该脚本仅支持 PASCAL，不支持 JAVASCRIPT。

```
//EZDML Global Event Scripts (Only support Pascal-Script) Ver20191006
//EZDML 全局事件脚本（注：只支持 Pascal 脚本）

//Generate SQL for a single Table. defRes is the default result.
//生成单个表的 SQL，传入表对象、是否生成创建表 SQL、是否生成创建约束 SQL、默认生成的 SQL 结果、数据库类型、选项，返回自定义结果 SQL
function OnEzdmlGenTbSqlEvent(tb: TCtMetaTable; bCreateTb, bCreateConstrains: Boolean; defRes,
dbType, options: String): string;
begin
  Result := defRes;
end;

//Generate upgrade SQL for an exists Table in a database. defRes is the default result.
//生成数据库的更新 SQL，传入新旧表对象、默认生成的 SQL 结果、数据库类型、选项，返回自定义结果 SQL
function OnEzdmlGenDbSqlEvent(designTb, dbTable: TCtMetaTable; defRes, dbType, options: String):
string;
begin
  Result := defRes;
end;

//Generate SQL for a single Field. defRes is the default result.
//生成单个字段的类型（varchar(255) nullable），传入表对象、字段对象、默认生成的结果、数据库类型、选项，返回自定义结果
function OnEzdmlGenFieldTypeDescEvent(tb: TCtMetaTable; fd: TCtMetaField; defRes, dbType,
options: String): string;
begin
  Result := defRes;
end;

//Generate upgrade SQL for an exists database-Field of a Table. defRes is the default result.
```

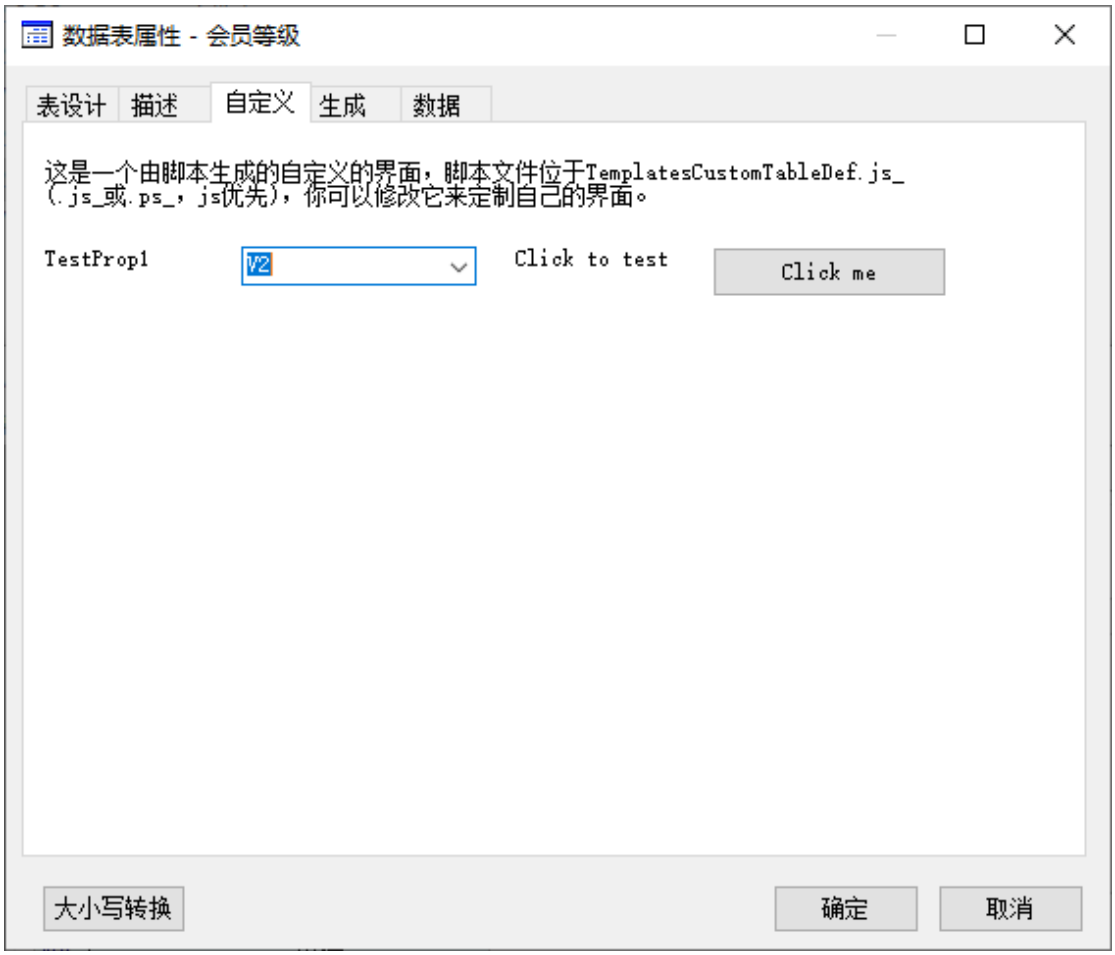
```
//生成增删改单个字段的SQL（alter table add xxx），传入要执行的操作(alter/add/drop)、表对象、新旧字段对象、默认生成的结果、数据库类型、选项，返回自定义结果
function OnEzdmlGenAlterFieldEvent(action: String; tb: TCtMetaTable; designField, dbField: TCtMetaField; defRes, dbType, options: String): string;
begin
  Result := defRes;
end;

//Reserved custom events
//自定义命令事件
function OnEzdmlCmdEvent(cmd, param1, param2: String; parobj1, parobj2: TObject): string;
begin
  Result := '';
end;

begin
end.
```

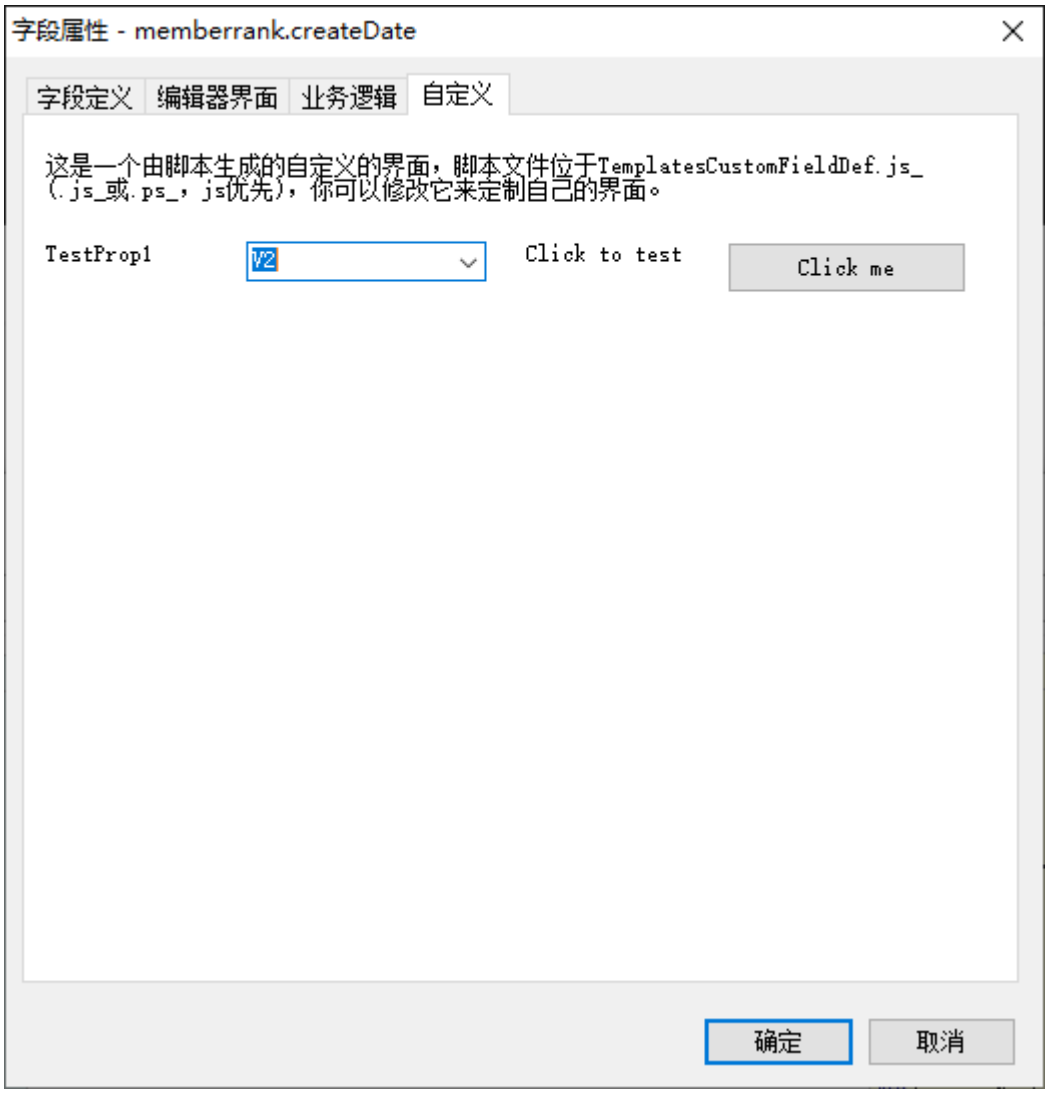
八、 CustomTableDef.js_/.ps_——自定义表属性界面脚本

位于 Templates 目录下。在 INI 中设置 EnableCustomPropUI=1 后，表属性会多一页自定义，自定义的界面由脚本生成。脚本的具体配置参见《EZDML 脚本配置》。



九、 CustomFieldDef.js_/.ps_——自定义字段属性界面脚本

位于 Templates 目录下。在 INI 中设置 EnableCustomPropUI=1 后，字段属性会多一页自定义，自定义的界面由脚本生成。脚本的具体配置参见《EZDML 脚本配置》。



十、 export_doc_html.js_/.ps_——导出 WORD、HTML 脚本

位于 Templates 目录下。EZDML 的导出 WORD、HTML、MHT 功能，其实都是导出 HTML，都是由这个脚本生成（脚本可以是 js_或 ps_，JS 优先）。脚本的具体配置参见《EZDML 脚本配置》。

WORD 是先导出 MHT 再由 WORD 打开，打开时 WORD 会尝试转换，你需要自己再保存一下。

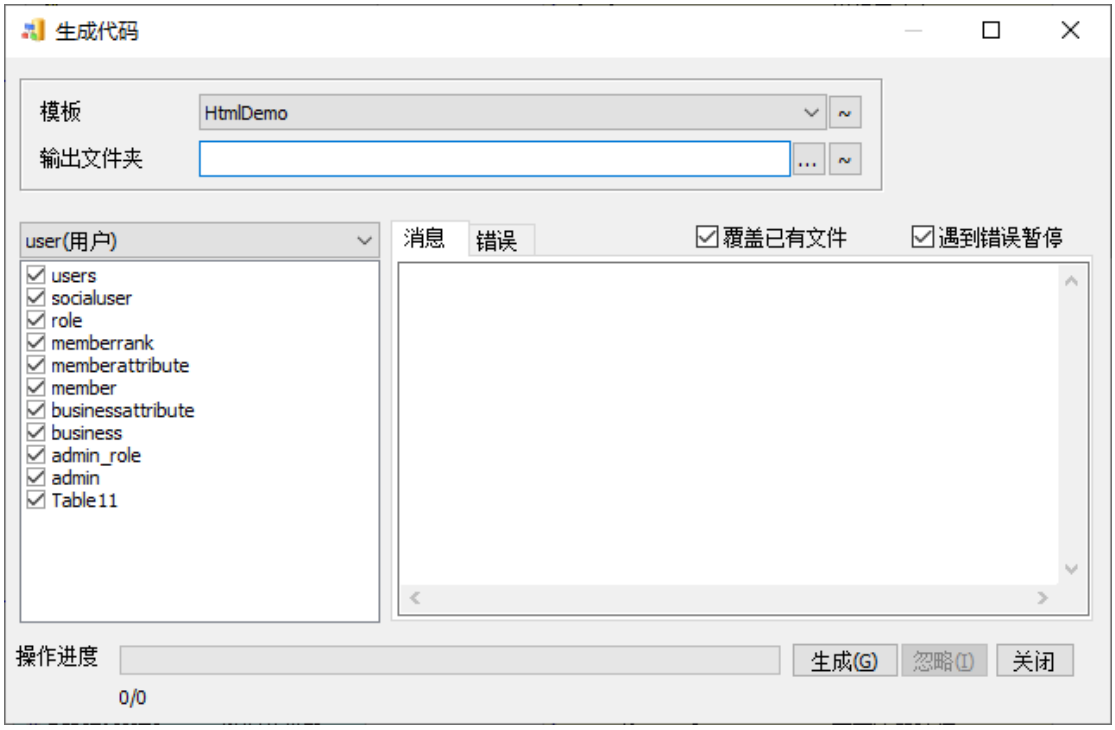
十一、 CustomTools*——自定义工具

此文件夹下的每个文件都会生成一个工具菜单。点击时会直接打开相应的文件；如果是 PAS 或 JS 文件，则尝试加载为脚本执行。脚本的具体配置参见《EZDML 脚本配置》。

十二、 Templates*.js/*.pas——自定义生成脚本

位于 Templates 目录下。生成代码及各种内容的模板文件。此目录下的 JS 和 PAS（JS 优先）会直接在表属性的“生成”页中显示，可直接以当前表加载脚本生成内容。脚本的具体配置参见《EZDML 脚本配置》。

执行菜单命令“模型|生成代码”，也会使用这些文件。如果有子目录，则子目录将作为一个项目来生成。



对子目录生成代码时，程序会读取_dml_config.INI 这个生成配置文件，此文件告诉程序要如何执行生成操作，示例如下：

[start.html]——对于 start.html 这个文件的配置

run_as_script=pas——该文件其实是一个 PAS 脚本文件，要作为 PAS 脚本加载执行，生成最终结果

[index_top.html] ——对于 index_top.html 这个文件的配置

rename=#curmodel_name#_index_top.html——该文件生成时要重命名，前面加当前模型名

run_as_script=pas——该文件其实是一个 PAS 脚本文件，要作为 PAS 脚本加载执行，生成最终结果

[table_ui] ——对于 table_ui 这个目录的配置

rename=ui_#curtable_name#——该目录生成时要重命名

loop_each_table=1——该目录下的文件、脚本需要对每一个表都执行一遍生成过程，该子目录下应该也要有一个_dml_config.INI 来控制生成过程

[dml_settings]

auto_open_on_finished=start.html——生成结束时打开这个文件