

代码生成器使用手册

CodeBuilder User Manuals

V2.5.0

1. 概述

CodeBuilder 是一款功能强大的代码生成工具。它能将你所设计的数据库结构转换成你 所想要的任何文本形式的文件,如 Java、C#、VB等代码文件,以及 SQL 脚本、数据库设计 文档等。通过开发插件,你甚至可将其转换成 Word、PDF 等二进制文件。

CodeBuilder 基于插件式、开放式的思想,你通过实现其定义的接口,就可轻松地将你 所开发的插件集成到 CodeBuilder 中来。目前 CodeBuilder 提供了数据源、模板以及工具三类 接口。CodeBuilder 基于.NET Framework 4.0 开发,它使用了动态编译技术,你可以嵌入 C#或 VB.NET 代码对对象属性进行个性化的扩展,结合灵活的模板生成你所想要的任何代码。

数据源(ISourceProvider)。提供数据结构的来源,目前提供数据库结构、PowerDesign设计文档两种数据源。数据库结构基于 Fireasy 的 SchemaProvider,目前支持 SQLite、MsSql(SqlServer)、MySql、PostgreSql、Oracle、Firebird 等数据库,以及 OleDb、Odbc 驱动。

模板(ITemplateProvider)。提供代码生成的模板,目前提供 Razor 和 T4 两种模板。你可以自行编写符合自己的模板来生成你所想要的代码。

工具(IToolProvider)。工具是一些常用的小程序,你可以自己开发其后集成到 CodeBuilder 里进行使用。

CodeBuilder 版权归帆影易动力(fireasy.cn)所有,目前开源于 github,如果你感兴趣,可到 <u>https://www.github.com/faib920/codebuilder</u> 下载查阅,希望你给予更好的反馈和建议。

2. 使用说明

2.1. 初识 CodeBuilder

打开代码生成器后,界面如下:

修代码生成器(CodeBuilde	er)						– 🗆 X
文件(F) 数据源(S) 模	板(T) 工具(O) 窗口(W)	帮助(H)					
📄 打开 💾 保存 🏸 生	E成						
对象列表 base.tt 国	实体类 扩展 控制	器类 ✓ 列表视图 ✓	编辑视图 实体关系	ト下文巻	-	属性	ά×
対象列表 base.tt 3 差称	文体类 扩展 / 空体炎 扩展 空体炎 扩展 空相判 衛注 礼品のmm_gift 礼品の無っgift 礼品の無っgift 人気口 人気口 会演 名称 日 名称 日 人気口 人気口 名称 日 日 人気切口 日 会気味戸mem_score_rule 会気が「市mem_score_rule 日 会気が「市mem_score_rule 日 日 人気切口 金気空別mem_sign 山口 日 日 第2回時間、品のm_gingulation ション スロー 日 東海部の別mem_sign ロー スロー スロー スロー 東海部の別mem_sign マー スロー スロー スロー 東京市のmage:rule マー スロー スロー スロー スロー 東京市のmage:rule マー スロー スロー スロー スロー スロー 大田 折田 (売音のmem_traingle ロー スロー スロー スロー スロー 大田 折田 (売音のmem_traingle ロー スロー スロー スロー スロー 大田 (市 大田 (市	<u>₩</u>	编辑初图 / 实体关系	<u> 上下文業 </u>	- 	廉性 → 余 顿 ClassName Conpany Description IsTree Hane	ل x fireasy WacofftClass fireasy Nal2022 ea_gift_class
输出					4 ×		
★清空 ⓑ 复制							
					^		
						Wane 表的名称。	
就绪					~	<u>दि</u> 性 🖾 变量	■ 模板 .::

界面分为三个区,右侧分别为属性窗口、变量窗口和模板窗口。中间为对象列表窗口、 以及模板文件编辑窗口、代码生成预览窗口等。下方为调试输出区。

认识了工具的界面后,下面我将带领你一步一步生成你所需要的代码。

2.2. 配置数据源

数据源主要是指数据库结构,包括数据表、字段、关系等,其架构如下(蓝色的项表示是 由工具自动生成的,橙色的项表示是隐藏的,属性窗口里看不到,但是模板里可以使用)。

名称	说明	类型
Name	表的名称	字符串
Description	表的说明	字符串
ClassName	生成的类的名称	字符串
Columns	字段集	集合
SubKeys	一对多的关系集	集合
ForeignKeys	多对一的关系集	集合

数据表(Table):

字段(Column):

名称	说明	类型
Name	字段的名称	字符串
Description	字段的说明	字符串
IsPrimaryKey	是否是主键	布尔
AutoIncrement	是否自增长	布尔
IsNullable	是否可空	布尔
DbType	标准的数据类型,如 String	DbType
DataType	数据类型,如 varchar2	字符串
Length	长度(字符型有效)	整型
Precision	精度	整型
Scale	小数位	整型
DefaultValue	默认值	字符串
ForeignKey	外键	关系
PropertyType	属性类型	字符串
PropertyName	属性名称	字符串
Owner	所性的数据表	数据表

关系(Reference):

名称	说明	类型
Name	键的名称	字符串
OnDelete	删除时如何处理	0限制 1层级 2设为 NULL
OnUpdate	更新地如何处理	0限制 1层级 2设为 NULL
PkTable	主表	数据表
PkColumn	主表的主键	字段
FkTable	子表	数据表
FkColumn	子表的外键	字段

默认可以使用 Database 和 Power Design 两种方式提供数据源。Database 即基于 Database Provider 方式,由 Fireasy.Data 的 Provider 提供,目前支持 SQLite、MsSql、MySql、PostgreSql、Oracle、Firebird 等常见的数据库,另外还可以使用 OleDb 和 Odbc 驱动适配更多类型的数据,如果你能力足够好,甚至可以自定义除此之外的其他数据库。库。Power Design 则是允许你从 Power Design 的设计文档(pdm)中导入数据库结构。

(1) 使用 Database

选择菜单"数据源"—"Database",打开数据源管理窗口,如下。

🌾 代码生成器(CodeBuilder)		- 0	×
文件(F) 数据	_{居源(} S) 模板(T) エ	[具(O) 窗口(W) 帮助(H)		
📄 打开 💾 🕯	呆存 🎽 生成			
対象列表			-	Ψ×
名称	《 数据源目理	ר באוות ש הזורא אוייר אוייר איירא אייר איירא איירא אייר		
	✓ 选择 □ 添加	▼ // 修改 × 删除 ▶ 测试		
	名称	连接串		
	SQLite			
	McSal	Jata Source- datadirectory db\Northwind.db3;fooling-irue		
	skiad	data source user id=sa:passwork initial catalog=si .st		
	MySal			
	local	data source=localhost;database=northwind;user id=root;password=faib;pooling=true;charse		
	📃 yunyi db	Data Souro database=yunyidb;User Id=hysoft;passwor pooling=tru		
	Oracle			
	📰 oracle	Data Source=orcl1;User ID=MORTHWIND;Password=faib		
	OleDb			
	📰 oledb_sql	Provider=SQLOLEDB.1;Passwor ;Persist Security Info=True;User ID=sa;Initial C		
	PostgreSql —			
	📰 pgsql	Server=localhost;User Id=postgres;Password=faib;Database=postgres		
	L			
		営 属性 🕅 变量 🖡	■模板│	
■ 1111				

这里列出了你所配置的所有数据源,双击其中的某一项,或选中某一项,然后点击工具 栏"选择"按钮即可导入该数据源的结构到"对象列表"中。

点击工具栏"添加"按钮,从下拉菜单中选择你所需要的数据库类型。

数据源名称			
主接字符串		 	^
			~

在弹出的窗口中,输入数据源名称,以及数据库连接串,你也可以单击"向导"按钮, 使用特定的连接串配置向导进行配置。

单击"测试"按钮,测试数据库是否连接成功,当数据库无法连接时你再尝试修改你的 连接字符串。

配置好数据源后,双击打开数据源,经过数秒的加载后,数据源中的所有表将加载到以 下的窗体中,此时勾选需要生成的表,注意如果有关系,那么关联的表也需要一起勾选过去, 否则生成的代码中可能缺失部分内容。

E贝):	表达式: 输入完成后按回车	过滤(F)	
	名称	备注	
1	🔲 📰 customercustomerdemo		
2	🔲 📰 customerdemographics		
3	🗌 🧱 customers		
4	🔲 📰 employeeterritories		
5	🗌 📰 order details		
6	🗌 📰 suppliers		
7	🗌 📰 territories		
8	🔲 📰 products		
9	🗌 📰 sqlite_sequence		
10	🔲 📰 SYS_SEQUENCE		
11	🗌 🧮 orders		
12	🔲 🧮 employees		

(2) 使用 Power Design

选择菜单"数据源"一"Power Design",定位到 Power Design 文档,双击打开,如下图:

医选择要生成的表		×
名称	备注	^
🗉 🗌 🕥 会员模块		
🖃 🛄 🔜 member		
mem_gift	礼品mem_gift	
🗌 📰 mem_gift_class	礼品分类mem_gift_class	
🗌 🧮 mem_score_rule	积分规则mem_score_rule	
🗌 🧮 mem_account	会员帐户mem_account	
🗌 🧮 mem_bank_card	用户银行卡mem_bank_card	
🗌 🧮 mem_sign	会员签到mem_sign	
🗌 🧮 mem_lucky	抽奖mem_lucky	
🗌 📰 mem_sign_rule	签到规则mem_sign_rule	
🗌 📰 mem_gift_box	收到的礼品mem_gift_box	
🗌 🧮 mem_charge_log	充值记录mem_charge_log	
	10.012 D	~

如果 Power Design 设计文档比较复杂,建议使用包进行划分。

2.3. 调整参数

(1) 对象列表

从数据源中获取到数据库结构后,显示在如下的"对象列表"窗口中,如下图:

🌾 代码生成器(CodeBuilde	r)			- 0	×
文件(F) 数据源(S) 模	反(T) 工具(O) 窗口(W)	帮助(H)			
📄 打开 💾 保存 🎾 生	成				
/ 対象列表		•	属性		ų×
2称	备注		✓ 杂项		
			Author	fireasy	
🖽 🗹 🌐 mem_gift	礼品mem_gift		ClassName	TenAccount	
🖽 🗹 🧮 mem_gift_class	礼品分类mem_gift_class		Lompany	fireasy 合昌此白	
	新日本2月1日		TsTree	False	
™ ♥] ⊞ mem_score_rule	积力规则mem_score_rule		Name	nen account	
🖽 🗹 🊃 mem_account	会员帐户mem_account				
🗄 🗹 🧱 mem_bank_card	用户银行卡mem_bank_card				
🗄 🗹 🧱 mem_sign	会员签到mem_sign				
🗄 🗹 🧱 mem_lucky	抽奖mem_lucky				
🗄 🗹 🧮 mem_sign_rule	签到规则mem_sign_rule				
🗄 🗹 🧮 mem_gift_box	收到的礼品mem_gift_box				
🗄 🗹 🧮 mem_charge_log	充值记录mem_charge_log				
🗄 🗹 🧱 mem_score_log	积分记录mem_score_log				
🗄 🗹 🧱 mem_level	会员等级mem_level				
🗄 🗹 🧱 mem_charge_rule	充值折扣设置mem_charge_rule				
🗄 🗹 🧱 mem_trading_log	交易记录mem_trading_log				
🗄 🗹 🧱 mem_gift_log	礼品记录mem_gift_log				
🗄 🗹 🧱 mem_cmbchina_sign	招行签约mem_cmbchina_sign		Name 表的名称。		
🗄 🗹 🧮 mem_note	会员相关说明mem_note				
		-	🚰 属性 📼 变量 鵟 🕯		
■ 输出					
就绪					:

对象列表显示了所选择的表,以及表的所有字段。右键菜单可以选择要生成的表及字段, 也可以通过"生成预览"查看生成后的代码样式。

对象列表中的备注列可以单击进行修改。

(2) 属性窗口

在对象列表中,单击表后,在右侧的属性窗口中,显示出表的所有属性;单击对象列表中的"备注"列,可以修改其内容,修改后立即在属性窗口中同步显示。

表的属性不是太多,只要注意一下 ClassName 的生成是否正确即可。

在对象列表中,展开节点树,显示表的所有字段,从节点图标中可以区分出,■为主键, ■为外键。单击字段后,在右侧的属性窗口中,显示字段的所有属性。

需要注意的是 IsPrimaryKey 和 AutoIncrement 是否与表结构中的一致。注意 DbType 是 否与 DataType 对应。DbType 是一个标准的枚举集,而 DataType 则是每一种数据库的数据类 型,如 MsSql 的 varchar、Oracle 的 varchar2、SQLite 的 text,均是对应到 DbType 中的 String。 除此这外,还应检查 PropertyType 及 PropertyName 是否正确。

修 代码生成器(CodeBuilder)			- 0	×
文件(F) 数据源(S) 模板(T) 工具(O) 窗口(W)	帮助(H)			
📄 打开 💾 保存 🎽 生成				
対象列表	•	属性		ųΧ
	^	∨ EasyUI		
1910) B 2		ControlType	TextBox	
🖽 🗹 🧱 customercustomerdemo		GenerateField	True	
🗉 🔽 🚍 austanar dan amarki ar		✓ 杂项		
C V H customer demographics		Author	fireasy	
🗄 🗹 🧱 customers		AutoIncrement	False	
		Company	fireasy	
🖿 🗹 🏥 employeeterritories		DataType	integer	
🖽 🗹 🧱 order details		DbType	Int32	
		DefaultValue		
🗄 🗹 🌐 suppliers		Description		
🗄 🔽 🧮 territories		> ForeignKey	employees.EmployeeID	
		IsNullable	False	
🖽 🗹 🧮 products		IsPrimaryKey	False	
H 🔽 🚍 salita seguence		Length	8	
m 🖂 🖽 zdri re zednemes		Name	EmployeeID	
🗉 🗹 🧱 SYS_SEQUENCE		Precision	19	
		PropertyName	Employeeid	
🗠 🕑 🌐 orders		FropertyType	_	
🗹 🗾 OrderID		Scale	0	
🖂 🖂 CustomerID				
🗹 🔜 EmployeeID				
🗹 🚍 OrderDate				
🗹 🚍 RequiredDate		Tame 字段的复称。		
🗹 🚍 ShippedDate		T-1XH JHOD *		
🗹 🚍 ShipVia	¥		■模板	
■ 输出				
就绪				:

需要注意的是 ClassName 和 PropertyName,默认使用 Pascal 规则进行命名,它与 Name 是不相同的。如以上的 mem_lucky 表,生成的 ClassName 是 MemLucky,如果你的 ORMapper 不没有此种映射,此时需要进行调整。如下图,右侧切换至变量窗口。

🍝 代码生成器(CodeBuilde	er)			_		
文件(F) 数据源(S) 模	板(T) 工具(O) 窗口(W)	帮助(H)				
📄 打开 💾 保存 🎽 生	E成					
对象列表		•	变量		φ×	:
	备注					
🗄 🗹 🧮 mem_gift	礼品mem_gift		✓ 杂项 Author	Waanawd		
🗄 🗹 📰 mem_gift_class	礼品分类mem_gift_class		ColumeRegex	huangau		
+ V men score rule	和分期Dimen score rule		CompanyName	Fireasy		
	<u> </u>		Mv cAr e a NameMode	Iest Pascal	~	
mem_account	云, yuyg, Phnem_account		Namespace	Inherit		1
🗄 🗹 🌐 mem_bank_card	用尸银行卡mem_bank_card		ProjectName T.Ll.P.com	Pascal		
🗄 🗹 🧮 mem_sign	会员签到mem_sign		Tablevegex	10_113_		
🗄 🗹 🧮 mem_lucky	抽奖mem_lucky					
🗄 🗹 🧮 mem_sign_rule	签到规则mem_sign_rule					
🗄 🗹 🧮 mem_gift_box	收到的礼品mem_gift_box					
🗄 🗹 🧮 mem_charge_log	充值记录mem_charge_log					
🗄 🗹 🧮 mem_score_log	积分记录mem_score_log					
🗄 🗹 🧮 mem_level	会员等级mem_level					
🗄 🗹 🧮 mem_charge_rule	充值折扣设置mem_charge_rule					
🗄 🗹 🧮 mem_trading_log	交易记录mem_trading_log					
🗄 🗹 🧱 mem_gift_log	礼品记录mem_gift_log					4
🗄 🗹 🧮 mem_cmbchina_sign	招行签约mem_cmbchina_sign		BameEode 命名方式 <u>。Pascal</u> 首字母	∃大写,适用于用−分隔的;	命名;	
🗄 🗹 🧮 mem_note	会员相关说明mem_note		Inherit 延续原始名称。			
			📸 属性 🖾 变量 🗎	模板		Ī
■ 輸出						
就绪						

NameMode 提供了两种命名方式, Inherit 继承方式, 即生成的 ClassName 和 Name 保持

一致;当更改为 Pascal 后,表名首字母大写,其余字母小写,当中间有下划线时,则认为是 一个单词的开始,如果没有下划线,则不会以新单词开始,如下:

mem_lucky => MemLucky

memlucky => Memlucky

(3) 变量窗口

变量窗口中的变量名称,在模板里都有可能会用得到,如果你觉得还不够用,你可以自 己定义更多的变量名称(见 3.1)。

变量名称	含义
Company	标示生成的文件的版权归属
Author	标示由谁生成的代码
ProjectName	标示所生成的项目的名称
Namespace	标示所生成的代码中的根命名空间
NameMode	命名方式, Inherit 和 Pascal 命名,在上级中已说明
TableRegex	表名的前缀正则表达式,表示需要剔除的前缀,若表达式为 TB_, 那么类
	似 TB_USER 这样命名的表,其 ClassName 将自动剔除 TB_,即为 USER
ColumnRegex	与 TableRegex 类似,应用于字段名称上,PropertyName 受此影响

在生成代码之前,你需要调整这些变量值,使生成的代码更符合你的需求。

设置这些变量值后,你可以单击变量窗口的工具栏上的"保存到文件"□按钮,将这些 变更保存到文件中,下次通过单击"打开外部文件"□加载这些设置。

2.4. 使用模板

一切准备就绪后,你可以选择一个模板,生成最终的代码。

(1) 模板文件

选择菜单"模板",从下拉菜单里选择一种模板。目前提供的 Razor 和 T4 两种模板引擎。 Razor 仅提供了一个生成 Entity Framework 的示例,如果你对此模板比较熟悉,你可尝试在 此之下创建自己的 Razor 模板。T4 比较灵活且功能强大,作者建议你使用 T4 模板来生成代 码,目前已经提供了好几套 T4 模板,你可以参考项目的模板来创建适合自己的模板。

下面以生成 Fireasy 的 ORMapper 为例。从菜单中选择 T4 Template 中的 Fireasy for C#模板然后,将主界面右侧切换至模板窗口,如下图:

修代码生成器(CodeBuilder)	– 🗆 X
文件(F) 数据源(S) 模板(T) 工具(O) 窗口(W) 帮助	(H)
📷 打开 💾 保存 🎾 生成	
/ 对象列表	▼ 模板 ¹ ×
小説家グリス 音注 名称 音注 第二 customeroustomerdemo 1 第二 customeroustomerdemo 1 第二 customeroustomerdemo 1 第二 customers 1 第二 customer	FW F
(
就绪	

一套模板由多个**分部**构成,每个分部负责生成一类文件。如上图中,entity.tt负责生成 实体类文件,entity_ex.tt负责生成实体类的扩展文件,context.tt负责生成 DbContext 上下文 类文件,controller.tt负责生成 MVC 中的控制器类文件,listview.tt 和 editview.tt 分别负责生 成列表视图和编辑视图,等等。

~	件	输出	循环	语法
en	tity.tt	Models\{ClassName}.cs	Tables	C#
Ē en	tity_ex.tt	Models\{ClassName}_Ex.cs	Tables	C#
re	ation.tt	Models\Relationships.cs	None	C#
co	ntext. tt	Data\DbContext. cs	None	C#
co	ontroller.tt	Areas\{MvcArea}\Controllers\	Tables	C#
1i	.stview.tt	Areas\{MvcArea}\Views\{Class	Tables	ASP/XHTML
ed	litview.tt	Areas\{MycArea}\Views\{Class	Tables	ASP / XHTML
1000	er 表 er re のの のの し に のの のの のの のの のの のの のの のの のの	entity.tt Entity_ex.tt relation.tt context.tt controller.tt listview.tt	entity.tt Models\[ClassName].cs R entity_ex.tt Models\[ClassName]_Ex.os relation.tt Models\Relationships.os context.tt Data\DbContext.os controller.tt Areas\[MvoArea]\Controllers\ listview.tt Areas\[MvoArea]\Views\[Class	entity.tt Models\{ClassName}.os Tables R entity_ex.tt Models\{ClassName}_Ex.os Tables relation.tt Models\Relationships.os None context.tt Data\DbContext.os None controller.tt Areas\{MvoArea}\Controllers\ Tables listview.tt Areas\{MvoArea}\Views\{Class Tables

单击工具栏"修改模板"按钮,弹出模板定义的窗口,如下图:

列表中的每一项目即是一个分部,它的输出和循环遵循一定的规则。

如果要为每一个表生成一个文件,则循环类型选择 Tables,那么输出的路径里,就可以

使用表中的相关属性来进行占位,如 ClassName、Name,除此之外,还可以使用变量窗口中 设置的变量来进行占位。

如果只生成一个文件,则循环选择 None,那么输出的路径里就不能再使用数据源中的 相关属性,但仍可以使用变量窗口中的变量。

双击打开一个模板文件, 映入你眼前的是你所熟悉的 T4 模板语法, 因此 T4 的语法这里 就不再赘述了, 重点介绍一下里面的 Host 变量。

Host 是 T4 模板的核心,其承载了整个数据源中的表、字段、关系,以及变量窗口里所 设置的值等等。

如果循环类型为 Tables, host.Current 返回的是当前的表 Table;如果后来类型为 None,则 host 中没有 Current 对象,取而代之的是 Tables、References 两个集合。

host.Profile 即变量窗口里所设置的值,如下图所示模板中的 host.Profile.ProjectName、 host.Profile.Namespace 等等。



(2) 生成代码

模板选择就绪后,单击主界面工具栏上的"生成"按钮,或选择"菜单"一"生成代码

文件",在弹出的分部选择对话框中,勾选需要生成的文件,首次生成需要选择文件输出的目录。



单击确定"按钮"后,CodeBuilder将开始工作,为你生成你所需要的代码文件。

修 代码生成器(CodeBuilder)		_	o x
文件(F) 数据源(S) 模板(T) 工具(O) 窗口(W) 帮助(H)		
📷 打开 💾 保存 🎾 生成			
/ 对象列表	•	模板	ąΧ
名称 备注			
🗄 🗹 🧱 customercustomerdemo		🖶 🧰 public	
🗄 🗹 🧱 customerdemographics		base. tt	
🗄 🗹 🧱 customers		- a entity.tt (实体类)	
🗄 🗹 🧱 employeeterritories		ー a entity_ex.tt (实体类_扩展)	
🗄 🗹 🧮 order details		- relation.tt (实体天系)	
🗄 🗹 🧮 suppliers			e
🗄 🗹 🧮 territories	正在处理,请耐心等待	r.tt(控制器类)	
E 🖉 🃰 products	按下 Esc 键可取消	tt (列表视图)	
E 🗹 🃰 sqlite_sequence		たた (5冊7時70月27)	
E Z SYS_SEQUENCE			
🗄 🗹 🧮 orders			
🗄 🗹 🧱 employees			
🗄 🗹 🧱 categories			
🗄 🗹 🧱 shippers			
🗄 🗹 🧱 region			
		🖀 属性 🕎 变量 🗎 模板	
■ 输出			
19%,正在生成 customers 的代码文件			

生成完毕,你就可以将生成的代码拷贝至你的项目里面了。

3. 开发说明

3.1. 变量扩展

变量,即变量窗口中的设置项,Razor 模板里的 Model.Profile 和 T4 模板里的 host.Profile 对象。变量是一个动态的、由外部代码文件定义的、经过动态编译而提供给模板使用的对象,其定义文件位于 CodeBuilder 工作目录 extensions\profile 文件夹下。以下是默认的定义,源

```
自 C#语法。
```

```
using System.ComponentModel;
public class ProfileExt1
{
   [Description("命名空间。")]
   public string Namespace { get; set; }
   [Description("项目名称。")]
   public string ProjectName { get; set; }
   [Description("公司名称。")]
   public string CompanyName { get; set; }
   [Description("作者。")]
   public string Author { get; set; }
   [Description("表名转类名的替换正则。")]
   public string TableRegex { get; set; }
   [Description("字段名转类名的替换正则。")]
   public string ColumeRegex { get; set; }
   [Description("Mvc中区域的名称。")]
   public string MvcArea { get; set; }
   [Description("命名方式。Pascal 首字母大写,适用于用-分隔的命名; Inherit 延续原始名称。")]
   public NameMode NameMode { get; set; }
}
public enum NameMode
{
   Inherit, //延续
   Pascal //首字母大写,适用于用-分隔的命名
}
```

你可以在此基础上增减变量,也可以创建自己的代码文件,以满足你自己的需要。注意, 此文件夹下的代码将会自动编译。下面我们通过一个小小的示例来说明该扩展的作用。

新增一个 C#语法的代码文件,编写如下代码,它的作用是在变量窗口里增加一个 DbType

变量。

<pre>using System.ComponentModel;</pre>
<pre>public class ProfileExt2 {</pre>

```
[Description("数据库类型。")]
public DBType DbType { get; set; }
}
public enum DBType
{
   SQLServer,
   MySQL,
   SQLIte,
   Oracle
}
```

保存完毕后,运行 CodeBuilder,效果如下图所示:



通过扩展后,你就可以在模板里通过 Profile 对象来使用你定义的变量了,如下图:

修 代码生成器(CodeBuilder)	- 🗆 X
文件(F) 数据源(S) 模板(T) 工具(O) 窗口(W) 帮助(H)	
	模板 [♀] ×
D:\FAIB\项目\CodeBuilder_2015\CodeBuilder\bin\Debug\templates\T4\htmldoo\index.tt 定位	
1 <#@ template hostSpecific="true" debug="true" #> ^	📮 🧰 public
<pre>2 <#@ output extension=".cs" #></pre>	here ++
3 <#	
<pre>4 TemplateHost nost = (TemplateHost)Host; - *></pre>	- 🛃 index. tt(主页)
> T / CONTYPE html>	
<pre>// <html></html></pre>	
8 <head></head>	
<pre>9 <link href="css/bootstrap.min.css" rel="stylesheet"/></pre>	
10	
11 <body></body>	
12 <div class="container"></div>	
13 <hl class="text-center">数据库设计说明书 (<#= host.Profile.DbType #>) </hl>	
14 <#	
15 toreach (var table in host.Tables)	
17 #2	
18 Chicks* table class="table table-bordered">	
<pre>cond ch class="text-center" style="width:160px">名称</pre>	
<pre>22 是否主键</pre>	
23 数据类型	
24 长度	
25 精度	
26 小数位数	
27 是否可空 Y	
<	🖺 属性 🕅 变量 📑 模板
■ 输出]	
就绪	



🗅 index.	html ×	+							- 0	>	<
$\leftrightarrow \rightarrow c$	③ 文件 C:/Users/	'faib/Desktop/in	dex.html					🖈 📘 🖫	ЭН	9	:
1 应用	aBP框架理论 🤇	▶ Net知识图谱	😂 Redi	s 📕 ASP.NE	۲ Cor 🗅 ۱	websocket在.	୬ SQLSE	RVER A sqlserve	死	>	•
	customercu	数 stomerc	据库设 lemo	计说明]书(SC	QLServ	/er)				4
	名称	是否主键	数据类型	长度	精度	小数位数	是否可空	备注			
	CustomerID	是	text	5	0	0	是				
	CustomerTypeID	否	text	10	0	0	是				
	customerde	mograp	hics								
	名称	是否主键	数据类型	长度	精度	小数位数	是否可空	备注			
	CustomerTypeID	是	text	10	0	0	是				
	CustomerDesc	否	text	2147483647	0	0	否				
	customers										
	名称	是否主键	数据类型	长度	精度	小数位数	是否可空	备注			
	CustomerID	是	text	5	0	0	是				
	CompanyName	否	text	40	0	0	是				
	ContactName	否	text	30	0	0	否				
	ContactTitle	否	text	30	0	0	否				
	Address	否	text	60	0	0	否				
	City	否	text	15	0	0	否				+

3.2. 数据源扩展

数据源也可在其定义之外进行扩展。数据源固有的属性见 2.2 节。与变量扩展类似,增加的属性也可以在模板里使用。其定义文件位于 CodeBuilder 工作目录 extensions\schema 文件夹下。下面以一个简单的示例来进行说明。

```
using System;
using System.ComponentModel;
using CodeBuilder.Core.Source;
[SchemaExtension(typeof(Column))]
public class ColumnExtForEasyUI
{
   [Category("EasyUI")]
   [Description("EasyUI控件类别。")]
   public EasyUIFieldType ControlType { get; set; }
   [Category("EasyUI")]
   [Description("是否生成域。")]
   [DefaultValue(true)]
   public bool GenerateField { get; set; }
}
public enum EasyUIFieldType
{
   TextBox,
   NumberBox,
   ComboBox,
   ComboTree,
   DateBox,
   DateTimeBox,
}
```

类需要添加 SchemaExtensionAttribute 特性,它表明将此类的属性扩展到对应的数据源, 该特性的唯一参数为表 Table、字段 Column 和关系 Reference。

你可以使用 CategoryAttribute 对扩展的属性进行分组,这样会更清晰明了,方便管理。 运行 CodeBuilder,打开数据源后,选择某一个字段,右侧属性窗口可以看到 EasyUI 组 下增加了两个属性,分别是 ControlType 和 GenerateField:

⇐ 代码生成器(CodeBuilde)	er)						- 0	\times
文件(F) 数据源(S) 模	扳(T) 工具(O)	窗口(W)	帮助(H)					
🗀 打开 💾 保存 🎽 生	- इ.र.							
	<u></u>							
_					•	属性		Ψ ^
名称	备注				^	∨ EasyUI	• . •	_
🕀 🖂 💳 solite sequence						Controllype	TextBox	~
						Generaterield	TextBox	
📧 🗹 🧮 SYS_SEQUENCE						* Author	ComboBox	
E 🔽 🚍 orders						AutoIngrement	ComboTree	
						Company	DateBox	
🗹 🔜 OrderID						DataType	DateTimeBox	
Customer TD						DbType	String	
						DefaultValue		
🗹 🔜 EmployeeID						Description		
OrderDate						ForeignKey		
						IsNullable	False	
🗹 🚍 RequiredDate						IsPrimaryKey	False	
ShinnedDate						Length	2147483647	
E Shippearte						Name	RequiredDate	
🗹 🚍 ShipVia						Precision	0	
Register						PropertyName	Requireddate	
C E Hergan						PropertyType	_	
🗹 🚍 ShipName						Scale	U	
🗹 🚍 ShipAddress								
🗹 🚍 ShipCity						ControlType		
🗹 🚍 ShipRegion						EasyUI控件类别。		
🗹 🚍 ShipPostalCode					~			
■ 榆屮			1				⊾ I — 1¥#1/X I	
就绪								

在模板里,可以通过字段对象来访问所扩展的属性,如下图:

(چ	代码生成	뉞器(CodeBuilder)		- 0	×
Ż	(F)	数据源(S) 模板(T) 工具(O) 窗口(W) 帮助(H)			
: -	ŧT∓	💾 保存 🖙 生成			
	1111				
	对象列表	editview.tt	×	模板	4 X
D:\]	AIB\项目\	CodeBuilder_2015\CodeBuilder\bin\Debug\templates\T4\fireasy\editview.tt 定位	Ì	🕞 📝 🕸	
64	}		^	🖶 🧰 publi c	
65	<td>ipt></td> <td></td> <td>D1</td> <td></td>	ipt>		D 1	
66	<div< td=""><td>data-options="region:'center',border:false"></td><td></td><td>a base tt</td><td></td></div<>	data-options="region:'center',border:false">		a base tt	
67	<	table class="form-body">		- 🔜 entity tt(实体类)	
68	<# forea	ch (dynamic column in table Columns)		- 🔜 entity_ex.tt(实体类_扩展)	
70	{	(dynamio obranii in basiciostamis)		relation tt (实体关系)	
71	i	f (column.IsPrimaryKey !column.GenerateField			
72	{			- 🛃 context.tt(上下文类)	
73		continue;		- 🔜 controller.tt(控制器类)	
74	}			listains the (周末初回)	
75	# >			a listview. tt (>jzojx(s)/	
76		<pre></pre>		況 editview.tt(编辑视图)	
78					
79					
80	<#				
81	}				
82	# >				
83	<	table>			
84	<td>data antiona-Weggion (aputh) barder false alaga-Wform butter }</td> <td></td> <td></td> <td></td>	data antiona-Weggion (aputh) barder false alaga-Wform butter }			
85	Curv	acta-options="region: South", border:Taise" Class="form=button"/ a class="easymi_linkbutton" onclick="saveInfo()" id="btnSave" data-options="icon[]s:'			
87	~	a class="easyui-linkbutton" onclick="saveInfo(true)" id="btnSaveAndNew" data-options=	-		
			- *		
×		· · · · · · · · · · · · · · · · · · ·		🔄 馮任 🗉 受量 🗏 模板	
	输出				
就约	者				

3.3. 定义新模板

略。

3.4. 开发数据源插件

略。

3.5. 开发模板插件

略。

3.6. 开发工具插件

略。