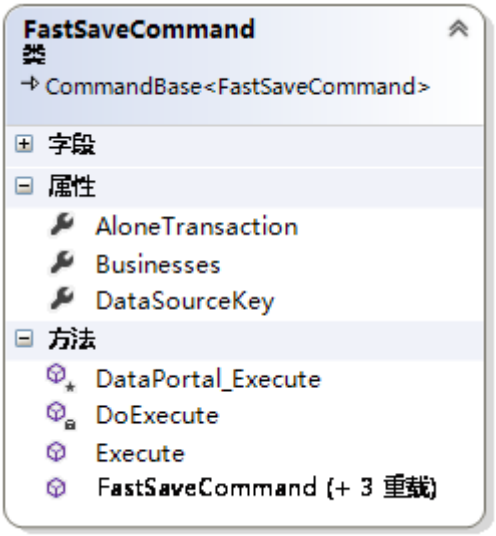
1 业务数据的读写-快速 Save

1.1 需求场景

有时，我们保存大批数据，需优先照顾到性能问题，不希望由中间层来验证数据，完全依赖数据库这最后一道关卡。为此，Phenix丶提供了相应的快速通道，支持这种特殊需求。

1.2 Command 类的实现方法

Phenix.Business.Tunnel.FastSaveCommand 类继承自 Csla.CommandBase<T>虚拟类，操作它和操作自己编写的 Command 类没任何区别：



以下我们用批量添加、删除 PH_ASSEMBLY 记录的 demo，来演示如何使用 FastSaveCommand 类：

1.2.1 New 业务对象

Phenix丶的 Addin 工具可自动生成 New 业务对象的方法代码，比如 Assembly<T>类中：

```
/// <summary>
/// New
/// </summary>
public static T New(long? AS_ID, string name, string caption)
{
  T result = NewPure();
  result._AS_ID = AS_ID;
  result._name = name;
  result._caption = caption;
  return result;
}
```

业务类提供的 New()函数，封装了字段赋值，不仅性能最佳，而且也是比较符合设计原则的。

1.2.2 批量提交数据

有了上述方法，我们可以将构建好的业务对象依次 Add()到 FastSaveCommand 的 Businesses 属性里，然后直接执行它的 Execute()函数：

```
//新增
FastSaveCommand command = new FastSaveCommand();
command.Businesses.Add(Assembly.New(10000, "Test1", "测试1"));
command.Businesses.Add(Assembly.New(10001, "Test2", "测试2"));
command.Execute();

//删除
command.Businesses.Clear();
AssemblyList assemblys = AssemblyList.Fetch(Assembly.AS_IDProperty.In(10000, 10001));
foreach (Assembly item in assemblys)
  command.Businesses.Add(item);
foreach (Assembly item in command.Businesses)
  item.Delete();
command.Execute();
```

执行结束，再来看看记录下的日志内容：

2014-01-17 21:18:12 <Command Text="insert into PH_ASSEMBLY(AS_ID,AS_NAME,AS_CAPTION) values (:PH_ASSEMBLY_AS_ID,:PH_ASSEMBLY_AS_NAME,:PH_ASSEMBLY_AS_CAPTION)" Timeout="0" /><Parameter Name="PH_ASSEMBLY_AS_ID" Value="10000" /><Parameter Name="PH_ASSEMBLY_AS_NAME" Value="Test1" /><Parameter Name="PH_ASSEMBLY_AS_CAPTION" Value="测试 1" />

2014-01-17 21:18:12 <Command Text="insert into PH_ASSEMBLY(AS_ID,AS_NAME,AS_CAPTION) values (:PH_ASSEMBLY_AS_ID,:PH_ASSEMBLY_AS_NAME,:PH_ASSEMBLY_AS_CAPTION)" Timeout="0" /><Parameter Name="PH_ASSEMBLY_AS_ID" Value="10001" /><Parameter Name="PH_ASSEMBLY_AS_NAME" Value="Test2" /><Parameter Name="PH_ASSEMBLY_AS_CAPTION" Value="测试 2" />

2014-01-17 21:18:13 <Command Text="select AS_ID,AS_NAME,AS_CAPTION from PH_ASSEMBLY where (AS_ID in (10000,10001))" Timeout="0" />

2014-01-17 21:18:13 Fetch Phenix.Security.Business.AssemblyList take 218.4004 millisecond, count = 2

2014-01-17 21:18:15 <Command Text="delete PH_ASSEMBLY where AS_ID=:PH_ASSEMBLY_AS_ID" Timeout="0" /><Parameter Name="PH_ASSEMBLY_AS_ID" Value="10000" />

2014-01-17 21:18:15 <Command Text="delete PH_ASSEMBLY where AS_ID=:PH_ASSEMBLY_AS_ID" Timeout="0" /><Parameter Name="PH_ASSEMBLY_AS_ID" Value="10001" />

1.3 Entity 类的实现方法

Phenix 在 Phenix.Core.Data 命名空间中提供了轻量级 EntityListBase<T, TEntity>基类，有直接批量操作数据库的 UpdateRecord()、DeleteRecord()、GetRecordCount()函数：

```
#region UpdateRecord

/// <summary>
/// 更新记录
/// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
/// </summary>
/// <param name="criteria">条件对象</param>
/// <param name="propertyValues">属性值队列</param>
public static int UpdateRecord(ICriteria criteria, IDictionary<IPropertyInfo, object> propertyValues)

/// <summary>
/// 更新记录
/// </summary>
/// <param name="criteriaExpression">条件表达式</param>
/// <param name="propertyValues">属性值队列</param>
public static int UpdateRecord(Expression<Func<TEntity, bool>> criteriaExpression, IDictionary<IPropertyInfo, object> propertyValues)

/// <summary>
/// 更新记录
/// </summary>
/// <param name="dataSourceKey">数据源键</param>
/// <param name="criteriaExpression">条件表达式</param>
/// <param name="propertyValues">属性值队列</param>
public static int UpdateRecord(string dataSourceKey, Expression<Func<TEntity, bool>> criteriaExpression, IDictionary<IPropertyInfo, object> propertyValues)

/// <summary>
/// 更新记录
/// </summary>
/// <param name="criteriaExpression">条件表达式</param>
/// <param name="propertyValues">属性值队列</param>
public static int UpdateRecord(CriteriaExpression criteriaExpression, IDictionary<IPropertyInfo, object> propertyValues)

/// <summary>
/// 更新记录
/// </summary>
```

```
        /// <param name="criterions">条件集</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(ICriterions criterions, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteria">条件对象</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbConnection connection, ICriteria criteria, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbConnection connection, Expression<Func<TEntity, bool>> criteriaExpression, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbConnection connection, CriteriaExpression criteriaExpression, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criterions">条件集</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbConnection connection, ICriterions criterions, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
```

```
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteria">条件对象</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbTransaction transaction, ICriteria criteria, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbTransaction transaction, Expression<Func<TEntity, bool>> criteriaExpression, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbTransaction transaction, CriteriaExpression criteriaExpression, IDictionary<IPropertyInfo, object> propertyValues)

        /// <summary>
        /// 更新记录
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criterions">条件集</param>
        /// <param name="propertyValues">属性值队列</param>
        public static int UpdateRecord(DbTransaction transaction, ICriterions criterions, IDictionary<IPropertyInfo, object> propertyValues)

        #endregion

        #region DeleteRecord

        /// <summary>
        /// 删除记录
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="criteria">条件对象</param>
        public static int DeleteRecord(ICriteria criteria)

        /// <summary>
        /// 删除记录
```

```
        /// </summary>
        /// <param name="criteriaExpression">条件表达式</param>
        public static int DeleteRecord(Expression<Func<TEntity, bool>> criteriaExpression)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="dataSourceKey">数据源键</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static int DeleteRecord(string dataSourceKey, Expression<Func<TEntity, bool>>
criteriaExpression)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="criteriaExpression">条件表达式</param>
        public static int DeleteRecord(CriteriaExpression criteriaExpression)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="criterions">条件集</param>
        public static int DeleteRecord(ICriterions criterions)

        /// <summary>
        /// 删除记录
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteria">条件对象</param>
        public static int DeleteRecord(DbConnection connection, ICriteria criteria)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static int DeleteRecord(DbConnection connection, Expression<Func<TEntity, bool>>
criteriaExpression)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static int DeleteRecord(DbConnection connection, CriteriaExpression criteriaExpression)
```

```
        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criterions">条件集</param>
        public static int DeleteRecord(DbConnection connection, ICriterions criterions)

        /// <summary>
        /// 删除记录
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteria">条件对象</param>
        public static int DeleteRecord(DbTransaction transaction, ICriteria criteria)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static int DeleteRecord(DbTransaction transaction, Expression<Func<TEntity, bool>>
criteriaExpression)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static int DeleteRecord(DbTransaction transaction, CriteriaExpression criteriaExpression)

        /// <summary>
        /// 删除记录
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criterions">条件集</param>
        public static int DeleteRecord(DbTransaction transaction, ICriterions criterions)

        #endregion

        #region RecordCount

        /// <summary>
        /// 获取记录数量
        /// </summary>
         public static long GetRecordCount()
```

```
        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="criteria">条件对象</param>
        public static long GetRecordCount(ICriteria criteria)

        /// <summary>
        /// 获取记录数量
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="criteria">条件对象</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(ICriteria criteria, object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="criteriaExpression">条件表达式</param>
        public static long GetRecordCount(Expression<Func<TEntity, bool>> criteriaExpression)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="dataSourceKey">数据源键</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static long GetRecordCount(string dataSourceKey, Expression<Func<TEntity, bool>>
criteriaExpression)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(Expression<Func<TEntity, bool>> criteriaExpression, object master,
string groupName)
```

```
        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="dataSourceKey">数据源键</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(string dataSourceKey, Expression<Func<TEntity, bool>> criteriaExpression, object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="criteriaExpression">条件表达式</param>
        public static long GetRecordCount(CriteriaExpression criteriaExpression)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(CriteriaExpression criteriaExpression, object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="criterions">条件集</param>
        public static long GetRecordCount(ICriterions criterions)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="connection">数据库连接</param>
        public static long GetRecordCount(DbConnection connection)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbConnection connection, object master, string groupName)
```

```
        /// <summary>
        /// 获取记录数量
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteria">条件对象</param>
        public static long GetRecordCount(DbConnection connection, ICriteria criteria)

        /// <summary>
        /// 获取记录数量
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteria">条件对象</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbConnection connection, ICriteria criteria, object master, string
groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static long GetRecordCount(DbConnection connection, Expression<Func<TEntity, bool>>
criteriaExpression)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbConnection connection, Expression<Func<TEntity, bool>>
criteriaExpression, object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static long GetRecordCount(DbConnection connection, CriteriaExpression criteriaExpression)

        /// <summary>
        /// 获取记录数量
```

```
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbConnection connection, CriteriaExpression criteriaExpression,
object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="connection">数据库连接</param>
        /// <param name="criterions">条件集</param>
        public static long GetRecordCount(DbConnection connection, ICriterions criterions)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        public static long GetRecordCount(DbTransaction transaction)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbTransaction transaction, object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteria">条件对象</param>
        public static long GetRecordCount(DbTransaction transaction, ICriteria criteria)

        /// <summary>
        /// 获取记录数量
        /// 条件类的字段映射关系请用 Phenix.Core.Mapping.CriteriaFieldAttribute 标注
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteria">条件对象</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbTransaction transaction, ICriteria criteria, object master,
```

```
string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static long GetRecordCount(DbTransaction transaction, Expression<Func<TEntity, bool>>
criteriaExpression)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbTransaction transaction, Expression<Func<TEntity, bool>>
criteriaExpression, object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        public static long GetRecordCount(DbTransaction transaction, CriteriaExpression criteriaExpression)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criteriaExpression">条件表达式</param>
        /// <param name="master">主对象</param>
        /// <param name="groupName">分组名, null 代表全部</param>
        public static long GetRecordCount(DbTransaction transaction, CriteriaExpression criteriaExpression,
object master, string groupName)

        /// <summary>
        /// 获取记录数量
        /// </summary>
        /// <param name="transaction">数据库事务</param>
        /// <param name="criterions">条件集</param>
        public static long GetRecordCount(DbTransaction transaction, ICriterions criterions)

        #endregion
```

示例代码见“Phenix.Test.使用指南.03.5”测试工程：

```
…
User user = User.New(test, test, null, test, null, null, null, 0, null, null, null, 0);
…
int i = UserEasyList.DeleteRecord(User.US_IDProperty == user.US_ID);
Console.WriteLine("删除 User 对象 " + (i == 1 ? "ok" : "error"));
…
```

执行后看看日志：

2017-04-10 16:49:25 <Command Text="insert into PH_USER(US_ID,US_USERNUMBER,US_PASSWORD,US_PASSWORDCHANGEDTIME,US_NAME,US_LOGIN,US_LOGOUT,US_LOGINFAILURE,US_LOGINFAILURECOUNT,US_LOGINADDRESS,US_DP_ID,US_PT_ID,US_LOCKED) values (:Pfdd1b8e485bb45cab700d2d101901,:P641d2cce688441a4b727f1ef8caff,:Pf2d495dd46c74b0298729b0660853,:Peac17bab760f48ba96655c33cf19b,:P335c9af674104dbfb57313310377d,:P8900b7482fb14dfaad2491fbc6688,:Pf05c649faa7d49d5be79f3db2f9cc,:P1c531a859cef4b28bfe304ac984a3,:P1cdaf2c497284402b79580a9c5f30,:Pb7e019a0b1444b33b4f08e706719b,:Pd5b2a3a33eff403792611c19b81ea,:P369079d373f34d9e8b994d69e6a89,:P57de7f8c78974f88a40284b3da232)" Timeout="0" /><Parameter Name="Pfdd1b8e485bb45cab700d2d101901" Value="513535753528972" /><Parameter Name="P641d2cce688441a4b727f1ef8caff" Value="t513535749" /><Parameter Name="Pf2d495dd46c74b0298729b0660853" Value="t513535749" /><Parameter Name="Peac17bab760f48ba96655c33cf19b" Value="null" /><Parameter Name="P335c9af674104dbfb57313310377d" Value="t513535749" /><Parameter Name="P8900b7482fb14dfaad2491fbc6688" Value="null" /><Parameter Name="Pf05c649faa7d49d5be79f3db2f9cc" Value="null" /><Parameter Name="P1c531a859cef4b28bfe304ac984a3" Value="null" /><Parameter Name="P1cdaf2c497284402b79580a9c5f30" Value="0" /><Parameter Name="Pb7e019a0b1444b33b4f08e706719b" Value="null" /><Parameter Name="Pd5b2a3a33eff403792611c19b81ea" Value="null" /><Parameter Name="P369079d373f34d9e8b994d69e6a89" Value="null" /><Parameter Name="P57de7f8c78974f88a40284b3da232" Value="0" />

2017-04-10 16:49:33 <Command Text="delete PH_USER where US_ID = :P395042a94dd34ef2997217d654247 " Timeout="30" /><Parameter Name="P395042a94dd34ef2997217d654247" Value="513535753528972" />

就这么简单。