**Lambda**

Lambda不用多说了，就一个匿名函数吧，如下一个简单的示例

Func<int, num> func = (int num) => num;

**Expression表达式**

Expression平时用的不多，但再ORM中比较常见

Expression就如其名称一样，用于表达某个语句

示例，你如何用代码去表达 i\*j+w\*x 这个算数，查看如下函数

static void Main(string[] args)

{

// 创建参数 i j w x

ParameterExpression i = Expression.Parameter(typeof(int), "i");

ParameterExpression j = Expression.Parameter(typeof(int), "j");

ParameterExpression w = Expression.Parameter(typeof(int), "w");

ParameterExpression x = Expression.Parameter(typeof(int), "x");

// 创建表达式 i\*j , Expression.Multiply 是表达式的乘法

BinaryExpression r1 = Expression.Multiply(i, j);

// 创建表达式 w\*x

BinaryExpression r2 = Expression.Multiply(w, x);

// 创建表达式 r1 + r2 即 i\*j + w\*x

BinaryExpression result = Expression.Add(r1,r2);

// 生成 Lambda 表达式, result 是该表达式的结果，i, j, w, x 是该表达式所需的4个参数

Expression<Func<int, int, int, int, int>> lambda = Expression.Lambda<Func<int, int, int, int, int>>(result, i, j, w, x);

// 编译表达式，生成 Lambda

Func<int, int, int, int, int> f= lambda.Compile();

// 执行 Lambda

var sum = f(1, 1, 1, 1);

}

Expression通过Expression.Lambda方法生成Lambda表达式，Lambda表达式在ORM中被翻译成SQL

**示例：ORM根据传入参数名动态排序**

// 扩展 IQueryable 方法，sort 为排序的字段名，des 指定是否倒序

public static IQueryable<T> IceOrderBy<T>(this IQueryable<T> queryable, string sort, bool des = false) {

// 排序的类型

var param = Expression.Parameter(typeof(T));

// 生成属性选择表达式，其所需要的参数为 param，其选择的字段名为 sort

var propExpression = Expression.Property(param, sort);

// 这里进行转换一下，应为 DateTime 无法自动转换为 object

var body = Expression.Convert(propExpression, typeof(object));

// 生成 Lambda 表达式

var keySelector = Expression.Lambda<Func<T, object>>(body, param);

// EF 的正序/倒序查询

if (des) {

return queryable.OrderByDescending(keySelector);

}

return queryable.OrderBy(keySelector);

}

// 示例

// 其生成的对应表示式是 (user) => user.name

userQueryable.IceOrderBy("name")