CONTACT

专注Xamarin.Android与ASP.NET MVC 🔤

组建XAMARIN中国社区与培训机构 (希望有更多人的帮助)

C#函数式编程之由函数构建函数

2015-03-17 16:30 by v-z-f, 1654 阅读, 1 评论, 收藏, 编辑

在面向对象的编程中,如果我们需要复用其他的类,我们可以通过继承来实现。而在函数式编程中我们也可以采取不同的方式来复用这些函数。今天的教程将会讲述两种方式,其中一个就是组合,将多个函数组合成为一个函数,另一个则是之前我们介绍过的部分应用,当然我们将会讲述如何将其高级化,来符合我们的使用要求。

组合

顾名思义,组合就是将函数A的结果传递给函数B。但是我们并不关注函数A的结果,当然大多数一定会这样去做:

```
1 var r1 = funcA(1);
2 var r2 = funcB(r1);
```

这样显然不是我们希望的那样,假设我们后面需要经常利用到这样的函数。问题就出现了,所以我们就需要利用组合来将他们合成一个新的函数,首先我们先写出两个用来组合的函数:

```
1 public static int FuncA(int x)
2 {
3    return x + 3;
4 }
5 
6 public static int FuncB(int x)
7 {
8    return x + 6;
9 }
```

如果我们不借助任何的自动化函数,我们可以通过这样的写法来进行组合:

```
Func<int, int> funcC = x => FuncB(FuncA(x));
```

但是我们这里无法使用var,因为C#的自动推断类型无法推断出这个类型。这样我们就有了一个新的函数funcC,我们可以试着执行这个函数看看最终的结果。上面我们通过手动的方式完成了组合,下面我们将编写一个自动化的函数来完成这个操作:

```
1 public static Func<T1, T3> Compose<T1, T2, T3>(Func<T1, T2> fu
nc1, Func<T2, T3> func2)
2 {
3    return x => func2(func1(x));
4 }
```

About 目前工作 : 技术经理 × 工作单位: 江苏斯诺物联科技有限公司 学过语言 : C/C++、Objective-c、java、 C#, javascript, Perl 读过书籍:《精诵ASP.NET Web程序测试》, 《SQL SERVER 2012编程入门经典》、《精通 ASP.NET MVC 3框架》、《C#高级编程》、 《C++ Primary 四》、《C#精髓》、 《ASP.NET 4高级程序设计》、《ASP.NET 设 计模式》、《JavaScript设计模式》、《JavaScript高级程序设计》等诸多名 兴趣爱好 : 巧克力、技术、天文学、写技术博客 相关社区 专注Xamarin开源社区 专注Mvvmcross开源社区 掌握Backbone.js、Knockout.js、Castle Active Record、Ninject、EF等框 架,染指C#、Javascript、SQL Server、Jquery、Jquery unit、Juqery mobil、Jquery ui、EasyUI、ASP.NET MVC、ASP.NET、HTML5、 Xamarin.Android等技术,对TDD、BDD、DDD有强热的追求、对代码工整非 堂苛刻。 Visitors 🍱 365 ,988 ጆ 230 164 4,148 ****** 2,784 95 2,289 **9**5 2,032 845 94 373 94 91 319 **200** 84 232 TRANSLATE FLAG counter **b** bing 昵称:<u>y-z-f</u> 园龄: <u>4年11个月</u> 粉丝:<u>713</u> 关注:<u>14</u> +加关注

最新随笔

SEARCH

如何在Ubuntu_16_04下使用MySql的GR	
如何利用Github+Appveyor+Nuget打造自己的.net core开源库	
.NET Core下使用gRpc公开服务 (SSL/TLS)	
.Net Core下如何管理配置文件	
如何在ASP.NET 5上搭建基于TypeScript的Angular2项目	
Xamarin.Android之UI Test简单入门	
TypeScript之接口类型	
TypeScript之基本数据类型	
Xamarin.Android经验之谈	
Xamarin.Android之MvvmCross	
MongoDB for C#基础入门	
Xamarin.IOS之多视图	
Xamarin.IOS之快速入门	
为Xamarin更好的开发而改写的库	
C#函数式编程之可选值	

接着我们利用这个函数来实现上面的功能:

```
var funcC = Compose<int, int, int>(FuncA, FuncB);
```

但是我们发现我们需要提供泛型参数,而不能依赖类型推断。但如果FuncA和FuncB在此之前显式的声明过则不需要提供泛型参数,例如将FuncA和FuncB写成如下的方式:

```
Func<int, int> FuncA = x => x + 2;
Func<int, int> FuncB = x => x + 6;
```

这样在调用Compose函数就不需要提供泛型参数了,顺便在这里介绍下其他语言下如何实现相同的功能,在F#中通过FuncB>> FuncA来实现,而在Haskell中则是用过FuncA.FuncB来实现,相比C#来说实现起来就非常的简单。通过上面的例子我们也发现了一个问题,就是函数A的返回类型必须和函数B的参数类型一致,并且在这个函数链中只有首个函数可以拥有多个参数,其他的函数只能拥有一个函数。当然函数链的最后一个函数可以是Action,就是说可以没有返回值,下面笔者写一个可以将三个函数进行组合的自动化函数:

```
1 public static Func<T1, T4> Compose<T1, T2, T3, T4>(Func<T1, T2
> func1, Func<T2, T3> func2, Func<T3, T4> func3)
2 {
3    return x => func3(func2(func1(x)));
4 }
```

当然实际开发中我们并不需要写,可以直接利用FCSLib中提供的函数。

高级的部分应用

学习过《函数式编程之部分应用》的人一定知道,部分应用就是将需要多个参数的函数,拆成一个函数链,每个函数链都只需要一个参数,假如FuncA需要三个参数,则使用部分应用后调用这个函数就需要按照如下的方式来使用FuncA(2)(3)(2),所以下面的内容笔者不会重复的介绍已经介绍过的内容,如果读者没有学习过,可以进入到上面对应的页面中进行学习。

我们知道在C#中如果传入部分应用这个自动化函数中的参数是方法,类型推断会无法工作,那么我们就需要输入繁琐的类型参数,比如下面这种情况:

```
Functional.Curry<Converter<int,int>,Ienumerable<int>,Ienumerable
<int>> (Functional.Map<int,int>);
```

读者会发现类型参数就占据的一半,上面我们也介绍了如何解决这个问题,所以我们可以写个已经显式声明过类型的函数来封装下Ma p函数:

最新评论

Re:MongoDB for C#基础入门

[quote][b]跪求[/quote]小仓鼠[/b] -- cbhe

Re:ASP.NET MVC学习之模型绑定(1)

厉害了 涨知识了 -- 缺舟一帆渡

Re:C#之玩转反射

博主的精神值得表扬,希望能再深入一下,大家一起努力学习!!! -- 小王子的 博客

Re: ASP.NET MVC学习之控制器篇

找到了 -- 越过那个限制

Re:ASP.NET MVC学习之路由篇(2)

mark。 mvc路由 -- MyCoolDog

Re: ASP. NET MVC学习之Ajax (完结)

@孤独影[code=csharp]Ajax.BeginForm("action","Controller",null,new AjaxOptions{HttpMethod="post",OnBegin..... -- 虚_心_?受教

Re:如何在ASP.NET 5上搭建基于TypeScript的Angular2项目

@jianyi 有实际运用 -- y-z-f

Re:如何在ASP.NET 5上搭建基于TypeScript的Angular2项目

博主在实际项目中用angular2了吗? -- jianyi

Re:ASP.NET MVC学习之路由篇(1)

alert('sss') -- 子赛

Re:jqPlot图表插件学习之饼状图和环状图

已经出来了 -- 就是你baby

4. IQueryable和IQueryProvider初尝

日历						随笔档案			
<	2017年10月 >						2017年3月(1)		
日	_	=	≡	四	五	六	2016年12月(2)		
24	25	26	27	28	29	30	2016年4月(3)		
1	2	3	4	5	6	7	2016年1月(1)		
8	9	10	11	12	13	14	2015年11月(2)		
15	16	17	18	19	20	21	2015年9月(1)		
22	23	24	25	26	27	28	2015年8月(1)		
29	30	31	1	2	3	4	2015年6月(1)		
D. 下 4 4 4 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7							2015年5月(2)		
随笔分类							2015年4月(1)		
ADO.NET(6)						2015年3月(3)			
Android开发(8)						2015年1月(6)			
ASP.NET(23)						2014年12月(2)			
ASP.NET MVC(15)							2014年11月(2)		
ASP.NET MVC源码分析(2)						2014年10月(2)			
C#(22)							2014年9月(12)		
HTML5(2)							2014年8月(2)		
JAVASCRIPT(11)							2014年7月(14)		
jqPlot(7)							2014年6月(5)		
knockout(8)							2014年5月(14)		
Linq(2)							2014年4月(3)		
MongoDB(1)							2014年3月(2)		
Perl(1)							2014年2月(5)		
SQL数据库(3)							2014年1月(2)		
Xamarin.Android(24)							2013年12月(6)		
Xamarin.IOS(2)							2013年11月(4)		
逐	数式组	式编程(8)				2013年5月(5)			
推荐排行榜						2013年4月(5)			
						2013年3月(6)			
1. C#之玩转反射(48)							2013年2月(3)		
2. 需要弥补的那部分SQL(30)						2013年1月(5)			
3. 为 (25)	Xama	rin更	好的开	发而。	2012年12月(8)				

```
public static Func<Converter<int, int>, IEnumerable<int>, IEnume
rable<int>> MapDelegate<T1, T2>()
{
    return Map<T1, T2>;
}
```

这样我们在调用Curry函数就不需要提供类型参数了:

```
Functional.Curry(Functional.MapDelegate<int,int>());
```

至此,类型推断的问题我们就解决了。在实际开发中部分应用虽然十分有用,但是在某些情形下却十分的麻烦,比如函数Filter需要两个算法,最后一个参数为数据。在实际使用中我们都会将两个算法赋进去,而在后面的使用中仅仅只会改变对应的数据,但是在采用部分应用后就显得麻烦了,下面是Filter函数的实现:

具体的功能就是通过compare函数判断是否符合条件,然后通过map函数返回需要的部分。我们可以通过如下的方式来调用这个函数:

在采用部分应用前,我们先写出这个函数的Delegate版本,这样我们就可以利用类型推断了:

```
public static Func<Func<T, R>, Func<T, bool>, IEnumera
ble<T>, IEnumerable<R>> FilterDelegate<T, R>()

{
    return Filter<T, R>;
}
```

然后我们就可以轻松的使用Currey函数将其部分应用了,这里笔者 直接自己实现了一个Currey函数,并没有使用FCSLib中提供的。读 者可以参考下:

- 5. ASP.NET MVC学习之Ajax (完结) (18)
- 6. ASP.NET MVC学习之路由篇 (1) (18)
- 7. Xamarin.Android之UI Test简单入 门(17)
- 8. MongoDB for C#基础入门(13)
- 9. ASP.NET MVC学习之模型验证篇 (13)
- 10. .NET Core下使用gRpc公开服务 (SSL/TLS)(10)

阅读排行榜

- 1. C#之玩转反射(41405)
- 2. javascript库之Mustache库使用说明 (26881)
- 3. Xamarin.Android快速入门(26405)
- 4. Android开发之使用 HttpURLConnection进行POST请求 (26169)
- 5. 使用 jQuery 进行前端验证 -- 1 (17555)
- 6. MongoDB for C#基础入门(16746)
- 7. ASP.NET MVC学习之路由篇 (1) (15063)
- 8. ASP.NET MVC学习之Ajax (完结) (14819)
- 9. Android开发之使用Handler刷新UI 控件(14214)
- 10. Android开发之底部导航栏标准 (12884)

```
public static Func<T1,Func<T2,Func<T3,R>>> Currey<T1,T</pre>
2,T3,R>(Func<T1,T2,T3,R> func)
2
3
                return x \Rightarrow y \Rightarrow z \Rightarrow func(x, y, z);
```

最后我们通过实际的使用来看看:

```
var f = Currey(FilterDelegate<int, int>());
            foreach (int x in f(x \Rightarrow x)(x \Rightarrow x \leq 10) (new int
[] { 2, 3, 1, 4, 5, 3, 34 }))
            {
               Console.WriteLine(x);
            }
           Console.ReadKey();
6
```

即使这样也很繁琐,所以我们需要进行更高级的部分应用,这里我 们需要另一个自动化函数来帮助我们实现:

```
public static Func<T3.R> Apply<T1, T2, T3, R>(Func<T1,
Func<T2, Func<T3, R>>> func,T1 arg1,T2 arg2)
             return x => func(arg1)(arg2)(x);
```

这个函数的作用就是将原本的部分应用的函数变成一个接收两个参 数并返回一个只接收一个参数的函数,因为算法部分不会变动,但 是数据会经常的变动。下面我们通过一个实际的运用来展示:

```
var f = Apply(Currey(FilterDelegate<int, int>()),
x => x, x => x <= 10);
            foreach (int x in f(new int[] { 2, 3, 1, 4, 5,
3, 34 }))
               Console.WriteLine(x);
            foreach (int x in f(new int[] { 1, 2, 3, 4, 5,
6, 7, 8, 9, 10, 11, 12, 13 }))
8
         {
                 Console.WriteLine(x);
10
           Console.ReadKey();
```

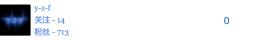
通过这样一番折腾后,我们就得到的我们真正需要的函数了,我们 在一开始的时候确定算法。然后在后面的使用中我们就可以只传递 数据即可。

Xamarin.Android -> Xamarin.IOS -> 混合 -> Xamarin.Forms









+加关注

« 上一篇:C#函数式编程之序列 » 下一篇:C#函数式编程之可选值

分类: 函数式编程

#1楼 赵御辩

ADD YOUR COMMENT

0

2015-03-21 08:58

读者会发现类型参数就占据的一半,上面我们也介绍了如何解决这个问题,所以我们可以写个已经显式声明过类型的函数来封装下Map函数: 在这里有个笔误:

在这里有个笔误: MapDelegate应该使用泛型

public static Func<Converter<T1, T2>, IEnumerable<T1>, IEnumerable<T1>, MapDelegate<T1, T2>() {

return Map<T1, T2>;
}

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论,请 登录 或 <u>注册</u>,<math><u>访问</u>网站首页。

【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【推荐】腾讯云 十分钟定制你的第一个微信小程序

【推荐】报表开发有捷径:快速设计轻松集成,数据可视化和交互



最新IT新闻:

- · 荷兰宣布2030年终结煤炭发电厂!淘汰所有燃油车
- · 美国最权威《消费者报告》:三星S8比iPhone 8更棒
- · 真· 稳如狗: 中国团队推出四足机器人 对标波士顿动力
- · 小冰向虚拟歌姬洛天依粉道歉 并没想diss你们心中的偶像
- ・360安全门锁淘宝众筹上线:0.5秒开锁/被盗最高赔10万
- » 更多新闻...



最新知识库文章:

- ·实用VPC虚拟私有云设计原则
- ·如何阅读计算机科学类的书
- · Google 及其云智慧
- · 做到这一点,你也可以成为优秀的程序员
- ·写给立志做码农的大学生
- » 更多知识库文章...