عنوان: **WF:Windows Workflow #۵**
نویسنده: محمد جواد تواضعی
تاریخ: ۱۳۹۱/۰۹/۰۸ ۲۰:۲۰
آدرس: www.dotnettips.info
برچسب‌ها: WorkFlow Foundation, Workflow

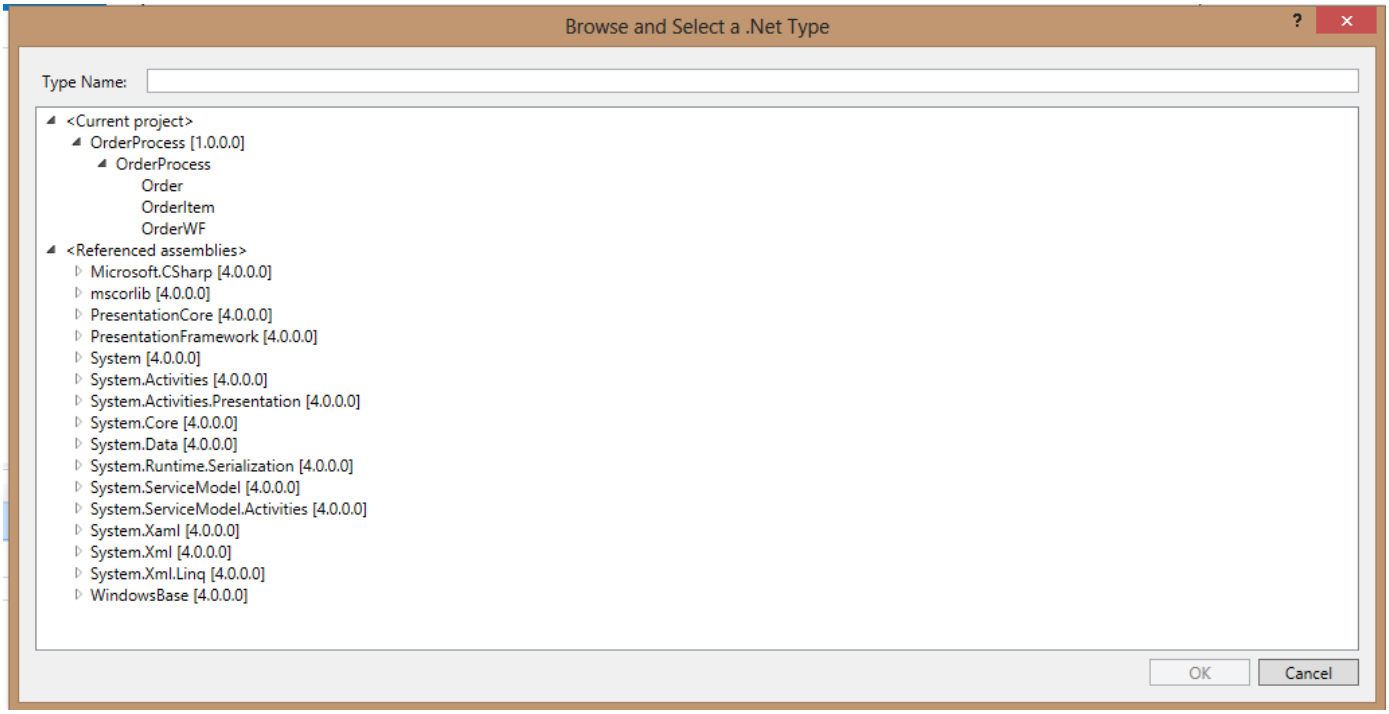
در این قسمت به پیاده سازی یک فرآیند سفارش ساده می‌پردازیم. ابتدا یک پروژه از نوع Workflow Console Application را ایجاد کرده و نام آن را Order Process می‌گذاریم و سپس کلاس‌های زیر را به آن اضافه می‌کنیم:

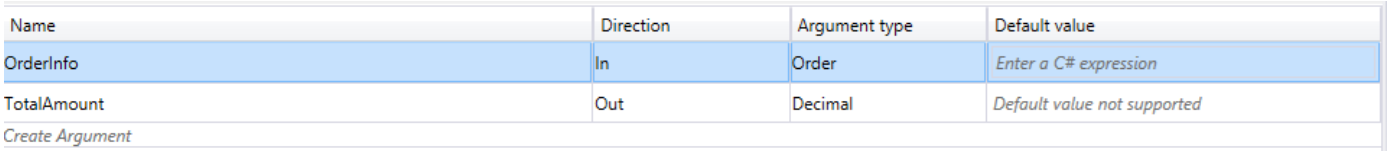
```
public class OrderItem
    {
        public int OrderItemID { get; set; }
        public int Quantity { get; set; }
        public string ItemCode { get; set; }
        public string Description { get; set; }
    }

    public class Order
    {
        public Order()
        {
            Items = new List<OrderItem>();
        }
        public int OrderID { get; set; }
        public string Description { get; set; }
        public decimal TotalWeight { get; set; }
        public string ShippingMethod { get; set; }
        public List<OrderItem> Items { get; set; }
    }
```

در اینجا دوکلاس تعریف شده است؛ یکی به نام OrderItem می‌باشد که شامل اطلاعات مربوط به میزان سفارش بوده و دیگری کلاس Order می‌باشد که شامل مشخصات سفارش است. سپس فایل OrderWF.xaml را باز کرده و شروع به ساخت فرآیند مورد نظر می‌کنیم. ابتدا یک Sequence را به درون صفحه کشیده و پس از آن در قسمت Arguments دو متغییر را تعریف می‌کنیم. یکی به نام TotalAmount و از نوع Decimal و Out می‌باشد و دیگری به نام OrderInfo که از نوع کلاس Order و In می‌باشد. سپس یک کنترل WriteLine را به آن اضافه می‌کنیم و در خاصیت Text آن رشته "Order Received" را قرار می‌دهیم. در ادامه یک کنترل Assign را در زیر آن قرار داده و مقدار متغییر TotalAmount را مساوی صفر وارد می‌کنیم.

نکته : برای اینکه نوع متغییر OrderInfo را از نوع کلاس Order قرار دهیم, ابتدا DropDown مربوطه را انتخاب کرده و گزینه Browse For Type را انتخاب می‌کنیم تا پنجره مورد نظر باز شود و از طریق آن، کلاس مورد نظر را انتخاب می‌کنیم. اگر در این قسمت کلاس مورد نظر یافت نشد، نیاز است ابتدا عمل Build Project را یک بار انجام دهیم.





بعضی از کنترل‌های Workflow در قسمت Toolbox موجود نمی‌باشند. از جمله این کنترل‌ها می‌توان به کنترل Add اشاره کرد. برای استفاده از این کنترل، ابتدا باید آن را به لیست کنترل‌ها اضافه نمود. جهت این امر, ابتدا در قسمت Toolbox یک Tab جدید را با نام دلخواه ایجاد کرده و سپس بر روی Tab کلیک راست نموده و گزینه Choose Items را انتخاب می‌کنیم. سپس از قسمت System.Activities.Components کنترل Add را انتخاب کرده و سپس بر روی دکمه OK کلیک می‌نمائیم. حال کنترل Add به لیست کنترل‌ها در Tab مورد نظر اضافه شده است.

در ادامه یک کنترل Switch را به فرایند خود اضافه کرده و مقدار T آن را برابر String قرار می‌دهیم؛ زیرا نوع داده‌ای که در قسمت Expression کنترل Switch قرار می‌گیرد، از نوع رشته می‌باشد. پس از اضافه کردن کنترل مورد نظر، کد زیر را به قسمت Expression کنترل اضافه خواهیم کرد:

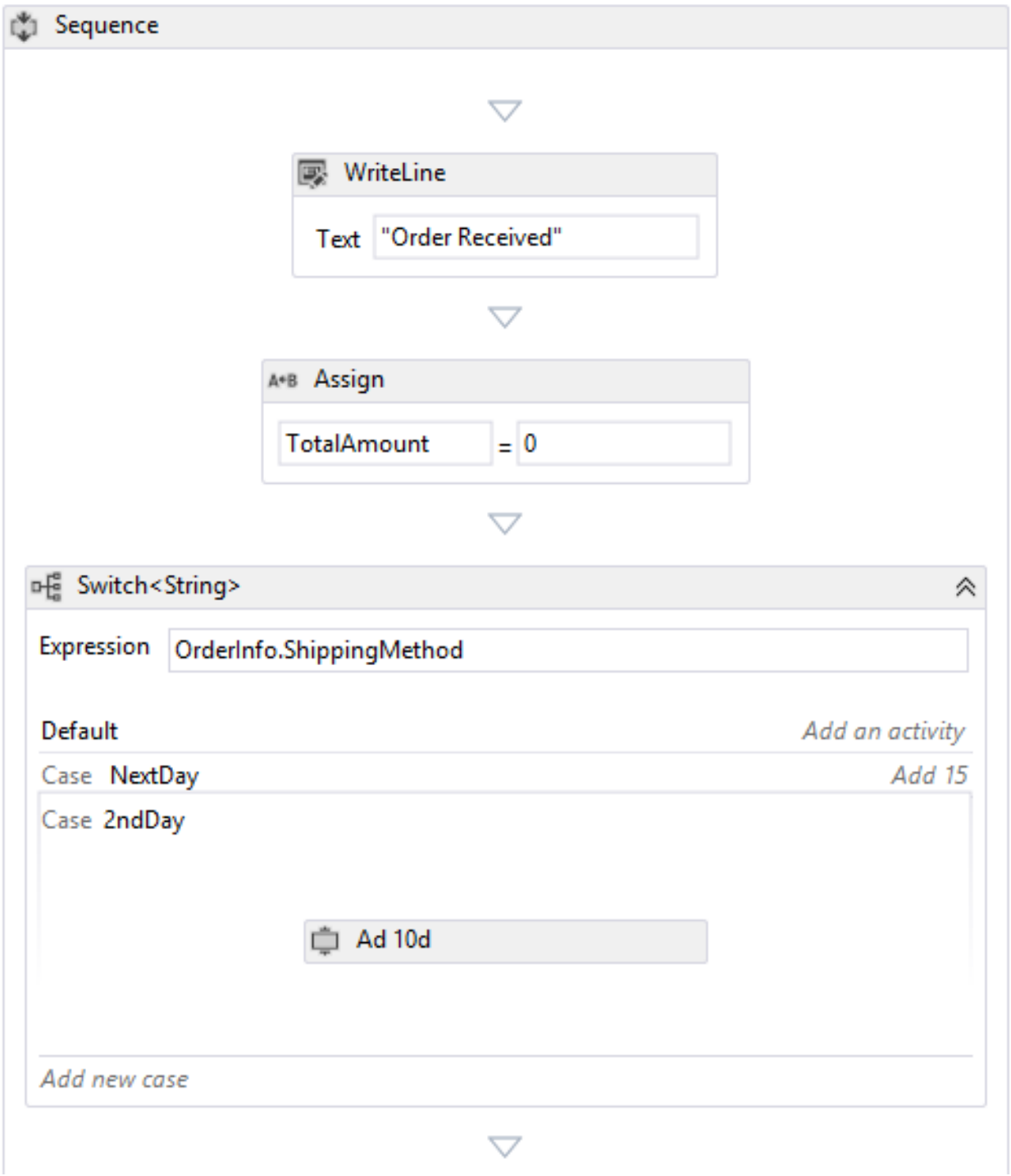
```
OrderInfo.ShippingMethod
```

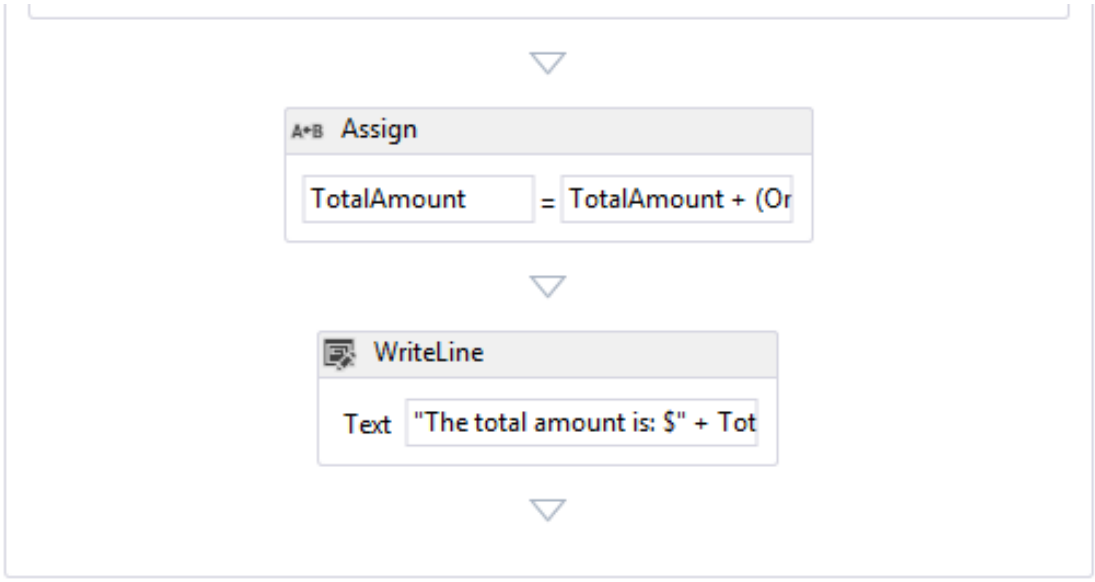
سپس در کنترل Switch، بر روی قسمت Add new case کلیک کرده و رشته‌های مورد نظر را اضافه می‌کنیم که شامل "" "NextDay"" و ""2ndDay"" می‌باشند. اکنون در بدنه هر دو Case، کنترل Add را اضافه می‌کنیم. در هنگام اضافه کردن باید برای سه خصوصیت، نوع مشخص شود و نوع هر سه را برابر Decimal قرار می‌دهیم.

در ادامه کنترل Add را انتخاب کرده و به خاصیت Right آنها به ترتیب مقدار های 10.0m و 15.0m را اضافه می‌کنیم و برای خصوصیت Result هر دو کنترل، متغیر TotalAmount را انتخاب می‌کنیم. سپس یک کنترل Assign را به صفحه اضافه کرده و در قسمت To، متغییر TotalAmount را قرار می‌دهیم و در قسمت Value کد زیر را:

```
TotalAmount + (OrderInfo.TotalWeight * 0.50m)
```

و در آخر با ستفاده از کنترل WriteLine به چاپ محتوای متغییر TotalAmount می‌پردازیم.





اکنون برای اینکه بتوانیم برنامه را اجرا کنیم، کد زیر را به کلاس Program.cs اضافه می‌کنیم:

```
static void Main(string[] args)
        {
            Order myOrder = new Order
            {
                OrderID = 1,
                Description = "Need some stuff",
                ShippingMethod = "2ndDay",
                TotalWeight = 100
            };
            IDictionary<String, object> input = new Dictionary<String, Object>
            {
                { "OrderInfo",myOrder}
            };
            IDictionary<String, Object> output = WorkflowInvoker.Invoke(new OrderWF(), input);
            Decimal total = (Decimal)output["TotalAmount"];
            Console.WriteLine("Workflow returned ${0} for my order total", total);
            Console.WriteLine("Press ENTER to exit");
            Console.ReadLine();

            //Activity workflow1 = new OrderWF();
            //WorkflowInvoker.Invoke(workflow1);
        }
```

در اینجا علت استفاده از IDictionary، نوع خروجی متد Invoke می‌باشد. در ادامه به کامل کردن این مثال پرداخته می‌شود.

نظرات خوانندگان

نویسنده: علیرضا جهانشاهلو
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۲:۲۵

خیلی ممنون از آموزش مفیدتون لطفا ادامه بدید.

نویسنده: محسن موسوی
تاریخ: ۱۳۹۱/۰۹/۰۹ ۱۲:۵۲

تشکر از این سری آموزش ها.
لطفا به مقوله چگونگی کاربرد آنها در وب نیز بپردازید.
منتظر ادامه مطلبتون هستیم.

نویسنده: محمد جواد تواضعی
تاریخ: ۱۳۹۱/۰۹/۰۹ ۲۳:۴

آقا محسن حتما مثالی که چگونگی کاربرد Workflow در برنامه‌های وب و دسکتاب به چه نحو است حتما گفته می‌شود .