



## RSGC - Roslyn Source Code Generators with examples

### Content

Nr.	Name	Summary	Link
1	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion">ThisAssembly</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion">https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion</a> )	The ThisAssembly.Info allows you access to the Assembly Information as constants, instead of going to reflection each time. I found useful to see the assembly version right away in any project that I have.	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion">https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion">https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion</a> )
2	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum">Enum</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum">https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum</a> )	This will generate code to fast parsing a int or a string to an enum	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum">https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum">https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum</a> )
3	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/JsonToClass">JsonByExampleGenerator</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/JsonToClass">https://github.com/ignatandrei/RSCG_Examples/tree/main/JsonToClass</a> )	This will generate C# classes from json files.	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/JsonToClass">https://github.com/ignatandrei/RSCG_Examples/tree/main/JsonToClass</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/JsonToClass">https://github.com/ignatandrei/RSCG_Examples/tree/main/JsonToClass</a> )
4	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor">CopyConstructor + Deconstructor</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor">https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor</a> )	This will generate code for a POCO to generate copy constructor and deconstructor	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor">https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor">https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor</a> )
5	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper">GeneratedMapper</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper">https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper</a> )	AutoMapping from a POCO to a DTO. Lots of customizations	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper">https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper">https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper</a> )
6	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/SkinnyControllers">Skinny Controllers</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/SkinnyControllers">https://github.com/ignatandrei/RSCG_Examples/tree/main/SkinnyControllers</a> )	This will generate code for WebAPI for each method of a field in the controller	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/SkinnyControllers">https://github.com/ignatandrei/RSCG_Examples/tree/main/SkinnyControllers</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/SkinnyControllers">https://github.com/ignatandrei/RSCG_Examples/tree/main/SkinnyControllers</a> )
7	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder">data-builder-generator</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder">https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder</a> )	Implements the Builder Design pattern for any class. Useful , at least, for test projects	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder">https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder">https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder</a> )
8	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MetadataFromObject">Metadata from object</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MetadataFromObject">https://github.com/ignatandrei/RSCG_Examples/tree/main/MetadataFromObject</a> )	This will generate code to retrieve the values of properties directly, not by reflection	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MetadataFromObject">https://github.com/ignatandrei/RSCG_Examples/tree/main/MetadataFromObject</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MetadataFromObject">https://github.com/ignatandrei/RSCG_Examples/tree/main/MetadataFromObject</a> )
9	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking">MockSourceGenerator</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking">https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking</a> )	This will generate Mock classes directly for any interface - with your implementation.	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking">https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking">https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking</a> )
10	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator">Method decorator</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator">https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator</a> )	This will generate code to decorate methods with anything you want ( stopwatch, logging , authorization...)	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator">https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator">https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator</a> )
11	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyApplied">PartiallyApplied</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction">https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction</a> )	This will generate curry for your functions	<a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction">https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction</a> ( <a href="https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction">https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction</a> )

### Links for Source Generators

<https://github.com/dotnet/roslyn/blob/master/docs/features/source-generators.md>

<https://github.com/dotnet/roslyn/blob/master/docs/features/source-generators.cookbook.md>

<https://github.com/dotnet/roslyn-sdk/tree/master/samples/CSharp/SourceGenerators>

### Helper for see the files

```
<EmitCompilerGeneratedFiles>true</EmitCompilerGeneratedFiles>  
<CompilerGeneratedFilesOutputPath>$(BaseIntermediateOutputPath)Generated</CompilerGeneratedFilesOutputPath>
```

## RSGC Name: ThisAssembly

Nuget :  
<https://www.nuget.org/packages/ThisAssembly>  
link : <https://www.clarius.org/ThisAssembly/>  
author :Daniel Cazzulino

### What can do

The ThisAssembly.Info allows you access to the Assembly Information as constants, instead of going to reflection each time. I found useful to see the assembly version right away in any project that I have.

### The code that you start with is

```
<PropertyGroup>

<Version>2021.2.15.800</Version>

</PropertyGroup>
```

The code that you will use is

```
var strVersion=ThisAssembly.Info.Version;

Console.WriteLine(strVersion);
```

The code that is generated is

```
/// <summary>
/// Provides access to the current assembly information as pure constants,
/// without requiring reflection.
/// </summary>

partial class ThisAssembly
{
    /// <summary>
    /// Gets the AssemblyInfo attributes.
    /// </summary>
    [GeneratedCode("ThisAssembly.AssemblyInfo", "1.0.0")]
    [CompilerGenerated]

    public static partial class Info
    {
        public const string Company = @"RSCG_Version";

        public const string Configuration = @"Debug";

        public const string FileVersion = @"2021.2.15.800";

        public const string InformationalVersion = @"2021.2.15.800";

        public const string Product = @"RSCG_Version";

        public const string Title = @"RSCG_Version";

        public const string Version = @"2021.2.15.800";

    }
}
```

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/ApplicationVersion](https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/ApplicationVersion](https://github.com/ignatandrei/RSCG_Examples/tree/main/ApplicationVersion))

All Generators: [https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/) ([https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/))

## RSGC Name: Enum

Nuget :  
<https://www.nuget.org/packages/AOPMethodsCommon/>  
<https://www.nuget.org/packages/AOPMethodsGenerator/>  
link : <http://msprogrammer.serviciipeweb.ro/category/roslyn/>  
author :Andrei Ignat

### What can do

This will generate code to fast parsing a int or a string to an enum

### The code that you start with is

```
[AutoEnum(template = EnumMethod.GenerateExtensionCode)]  
  
public enum MathematicalOperation  
{  
  
    None=0,  
  
    Add=1,  
  
    Multiplication=2  
  
}
```

### The code that you will use is

```
var fromInt = enumMathematicalOperation.ParseExactMathematicalOperation(1);  
  
var fromString = enumMathematicalOperation.ParseExactMathematicalOperation("add");  
  
Console.WriteLine(fromInt + "="+fromString);
```

### The code that is generated is

```

[GeneratedCode("AOPMethods", "")]
[CompilerGenerated]

public static partial class enumMathematicalOperation{

/*

public static int idMathematicalOperation(){

System.Diagnostics.Debugger.Break();

return 1;

}

*/

public static RSCG_Enum.MathematicalOperation ParseExactMathematicalOperation(this long value, RSCG_Enum.MathematicalOperation? defaultValue = null){

    if(0 == value)

        return RSCG_Enum.MathematicalOperation.None;

        if(1 == value)

            return RSCG_Enum.MathematicalOperation.Add;

            if(2 == value)

                return RSCG_Enum.MathematicalOperation.Multiplication;

if(defaultValue != null)

    return defaultValue.Value;

throw new ArgumentException("cannot find " + value + " for RSCG_Enum.MathematicalOperation ");

}

public static RSCG_Enum.MathematicalOperation ParseExactMathematicalOperation(this string value, RSCG_Enum.MathematicalOperation? defaultValue = null){

    //trying to see if it is a value inside
    //if(!string.IsNullOrEmpty)

    if(long.TryParse(value, out long valueParsed)){

        return ParseExactMathematicalOperation(valueParsed);

    }

    if(0==string.Compare("None" , value, StringComparison.InvariantCultureIgnoreCase))

        return RSCG_Enum.MathematicalOperation.None;

        if(0==string.Compare("Add" , value, StringComparison.InvariantCultureIgnoreCase))

            return RSCG_Enum.MathematicalOperation.Add;

            if(0==string.Compare("Multiplication" , value, StringComparison.InvariantCultureIgnoreCase))

                return RSCG_Enum.MathematicalOperation.Multiplication;

if(defaultValue != null)

    return defaultValue.Value

throw new ArgumentException("cannot find " + value + " for RSCG_Enum.MathematicalOperation ");

}

/*

*/

}

```

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/Enum](https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/Enum](https://github.com/ignatandrei/RSCG_Examples/tree/main/Enum))

All Generators: [https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/) ([https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/))

## RSGC Name: JsonByExampleGenerator

Nuget :  
<https://www.nuget.org/packages/JsonByExampleGenerator/>  
link : <https://github.com/hermanussen/JsonByExampleGenerator/>  
author :Robin Hermanussen

### What can do

This will generate C# classes from json files.

### The code that you start with is

```
{  
  "FirstName": "Andrei",  
  "LastName": "Ignat",  
  "Blog": "http://msprogrammer.serviciipeweb.ro/"
```

The code that you will use is

```
var p1 = new Person();  
p1.Blog = "http://msprogrammer.serviciipeweb.ro/";  
var config = new ConfigurationBuilder()  
    .AddJsonFile("persons.json")  
    .Build();  
  
var p = config.Get<Person>();  
var p2 = Person.FromConfig(config);
```

The code that is generated is

```
[DataContract(Name = "Person", Namespace = "JsonToClass.Json.Persons")]  
public partial class Person  
{  
    [DataMember(Name = "FirstName", EmitDefaultValue = false, Order = 0)]  
    public string FirstName { get; set; }  
    [DataMember(Name = "LastName", EmitDefaultValue = false, Order = 1)]  
    public string LastName { get; set; }  
    [DataMember(Name = "Blog", EmitDefaultValue = false, Order = 2)]  
    public string Blog { get; set; }  
  
    public static Person FromConfig([System.Diagnostics.CodeAnalysis.NotNull] IConfiguration config)  
    {  
        return config.Get<Person>();  
    }  
}
```

Example Code: [https://github.com/ignatandre/RSCG\\_Examples/tree/main/JsonToClass](https://github.com/ignatandre/RSCG_Examples/tree/main/JsonToClass) ([https://github.com/ignatandre/RSCG\\_Examples/tree/main/JsonToClass](https://github.com/ignatandre/RSCG_Examples/tree/main/JsonToClass))

All Generators: [https://github.com/ignatandre/RSCG\\_Examples/](https://github.com/ignatandre/RSCG_Examples/) ([https://github.com/ignatandre/RSCG\\_Examples/](https://github.com/ignatandre/RSCG_Examples/))

## RSGC Name: CopyConstructor + Deconstructor

Nuget:  
<https://www.nuget.org/packages/AOPMethodsCommon/>  
<https://www.nuget.org/packages/AOPMethodsGenerator/>  
link: <http://msprogrammer.serviciipeweb.ro/category/roslyn/>  
author: Andrei Ignat

### What can do

This will generate code for a POOCO to generate copy constructor and deconstructor

### The code that you start with is

```
[AutoMethods(template = TemplateMethod.CustomTemplateFile, CustomTemplateName = "CopyConstructorDestructor.txt")]

partial class Person
{
    public string FirstName { get; set; }

    public string LastName { get; set; }
}
```

### The code that you will use is

```
var pOldPerson = new Person();

pOldPerson.FirstName = "Andrei";
pOldPerson.LastName = "Ignat";

var newPerson = new Person(pOldPerson);
Console.WriteLine(newPerson.FirstName);

var (_, last) = newPerson;
Console.WriteLine(last);
```

### The code that is generated is

```
public Person () {
    OnConstructor();
}

public Person(IPerson other):base() {
    BeforeCopyConstructor(other);
    CopyPropertiesFrom(other);
    AfterCopyConstructor(other);
}

public void CopyPropertiesFrom(IPerson other){

    this.FirstName = other.FirstName;
    this.LastName = other.LastName;
}

public void Deconstruct( out string FirstName, out string LastName)
{
    FirstName = this.FirstName;
    LastName = this.LastName;
}
```

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/CopyConstructor](https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/CopyConstructor](https://github.com/ignatandrei/RSCG_Examples/tree/main/CopyConstructor))

All Generators: [https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/) ([https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/))

## RSGC Name: GeneratedMapper

Nuget :  
<https://www.nuget.org/packages/GeneratedMapper/>  
link : <https://github.com/ThomasBleijendaal/GeneratedMapper>  
author : ThomasBleijendaal

### What can do

AutoMapping from a POCO to a DTO. Lots of customizations

### The code that you start with is

```
public class Department
{
    public int ID { get; set; }

    public string Name { get; set; }

    public List<string> Employees { get; set; }
}

[IgnoreInTarget("Employees")]
[MapFrom(typeof(Department))]
public class DepartmentDTO
{
    public int ID { get; set; }

    public string Name { get; set; }

    [MapWith("Employees", typeof(ResolverLength))]
    public int EmployeesNr { get; set; }
}

public class ResolverLength
{
    public int Resolve(List<string> input)
    {
        return ((input?.Count) ?? 0);
    }
}
```

### The code that you will use is

```
static void Main(string[] args)
{
    var dep = new Department();

    dep.Name = "IT";

    dep.ID = 1;

    dep.Employees = new List<string>();

    dep.Employees.Add("Andreï");

    var dto = dep.MapToDepartmentDTO();

    Console.WriteLine(dto.Name+"->" + dto.EmployeesNr);
}
```

### The code that is generated is



```

namespace DTOMapper
{
    public static partial class DepartmentMapToExtensions
    {
        public static DTOMapper.DepartmentDTO MapToDepartmentDTO(this DTOMapper.Department self)
        {
            if (self is null)
            {
                throw new ArgumentNullException(nameof(self), "DTOMapper.Department -> DTOMapper.DepartmentDTO: Source is null.");
            }

            var resolverLength = new DTOMapper.ResolverLength();

            var target = new DTOMapper.DepartmentDTO
            {
                ID = self.ID,
                Name = (self.Name ?? throw new GeneratedMapper.Exceptions.PropertyNullException("DTOMapper.Department -> DTOMapper.DepartmentDTO: Property Name is null.")),
                EmployeesNr = resolverLength.Resolve((self.Employees ?? throw new GeneratedMapper.Exceptions.PropertyNullException("DTOMapper.Department -> DTOMapper.DepartmentDTO: Property Employees is null."))),
            };

            return target;
        }
    }
}

```

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/DTOMapper](https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/DTOMapper](https://github.com/ignatandrei/RSCG_Examples/tree/main/DTOMapper))

All Generators: [https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/) ([https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/))

## RSGC Name: Skinny Controllers

NuGet :  
<https://www.nuget.org/packages/SkinnyControllersCommon/>  
<https://www.nuget.org/packages/SkinnyControllersGenerator/>

link : <http://msprogrammer.serviciipeweb.ro/category/roslyn/>

author : Andrei Ignat

### What can do

This will generate code for WebAPI for each method of a field in the controller

### The code that you start with is

```
public class PersonRepository
{
    public async Task<Person> Get(int id)
    {
        await Task.Delay(1000);
        return new Person()
        {
            ID = id,
            Name = "Andrei " + id
        };
    }

    //add more Functions here to make the demo
}
```

### The code that you will use is

```
[AutoActions(template = TemplateIndicator.AllPostWithRecord, FieldsName = new[] { "" }, ExcludeFields = new[] { "_logger" })]
[ApiController]
[Route("[controller]/[action]")]
public partial class PersonController : ControllerBase
{
    private readonly PersonRepository pr;
    private readonly ILogger<PersonController> _logger;

    public PersonController(PersonRepository pr, ILogger<PersonController> logger)
    {
        this.pr = pr;
        _logger = logger;
    }
}
```

### The code that is generated is

```
[HttpPost]
public System.Threading.Tasks.Task<AOPSkinnyController.Classes.Person> Get( recGet_143266108 data ){

    return

    pr.Get(data.id);
}
```

Example Code: [https://github.com/ignatandre/RSCG\\_Examples/tree/main/SkinnyControllers](https://github.com/ignatandre/RSCG_Examples/tree/main/SkinnyControllers) ([https://github.com/ignatandre/RSCG\\_Examples/tree/main/SkinnyControllers](https://github.com/ignatandre/RSCG_Examples/tree/main/SkinnyControllers))

All Generators: [https://github.com/ignatandre/RSCG\\_Examples/](https://github.com/ignatandre/RSCG_Examples/) ([https://github.com/ignatandre/RSCG\\_Examples/](https://github.com/ignatandre/RSCG_Examples/))

## RSGC Name: data-builder-generator

NuGet : <https://www.nuget.org/packages/DasMulti.DataBuilderGenerator/>

link : <https://github.com/dasMulti/data-builder-generator>

author :Martin Andreas Ulrich

### What can do

Implements the Builder Design pattern for any class. Useful , at least, for test projects

### The code that you start with is

```
[GenerateDataBuilder]

public class Person
{
    public string FirstName { get; set; }

    public string? MiddleNames { get; set; }

    public string LastName { get; set; }
}
```

The code that you will use is

```
var pOld = new Person();

pOld.FirstName = "Andrei";

pOld.LastName = "Ignat";

pOld.MiddleNames = "G";

var build = new PersonBuilder(pOld).WithoutMiddleNames().WithFirstName("Florin");

var pNew = build.Build();

Console.WriteLine(pNew.FirstName);
```

The code that is generated is

```
public partial class PersonBuilder
{
    private string? _firstName;

    private string? _middleNames;

    private string? _lastName;

    public PersonBuilder()
    {
    }

    public PersonBuilder(PersonBuilder otherBuilder)
    {
        _firstName = otherBuilder._firstName;

        _middleNames = otherBuilder._middleNames;

        _lastName = otherBuilder._lastName;
    }

    public PersonBuilder(Person existingInstance)
    {
        _firstName = existingInstance.FirstName;

        _middleNames = existingInstance.MiddleNames;

        _lastName = existingInstance.LastName;
    }

    public PersonBuilder WithFirstName(string firstName)
    {
        var mutatedBuilder = new PersonBuilder(this);

        mutatedBuilder._firstName = firstName;

        return mutatedBuilder;
    }

    public PersonBuilder WithMiddleNames(string? middleNames)
    {
        var mutatedBuilder = new PersonBuilder(this);

        mutatedBuilder._middleNames = middleNames;

        return mutatedBuilder;
    }
}
```

```
public PersonBuilder WithoutMiddleNames()
{
    var mutatedBuilder = new PersonBuilder(this);
    mutatedBuilder._middleNames = null;
    return mutatedBuilder;
}

public PersonBuilder WithLastName(string lastName)
{
    var mutatedBuilder = new PersonBuilder(this);
    mutatedBuilder._lastName = lastName;
    return mutatedBuilder;
}

public Person Build()
{
    var instance = new Person();
    if (!(_firstName is null))
        instance.FirstName = _firstName;
    if (!(_middleNames is null))
        instance.MiddleNames = _middleNames;
    if (!(_lastName is null))
        instance.LastName = _lastName;
    return instance;
}
}
```

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/DP\\_Builder](https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/DP\\_Builder](https://github.com/ignatandrei/RSCG_Examples/tree/main/DP_Builder))

All Generators: [https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/) ([https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/))

## RSGC Name: Metadata from object

Nuget:  
<https://www.nuget.org/packages/AOPMethodsCommon/>  
<https://www.nuget.org/packages/AOPMethodsGenerator/>  
link: <http://msprogrammer.serviciopeweb.ro/category/roslyn/>  
author: Andrei Ignat

### What can do

This will generate code to retrieve the values of properties directly, not by reflection

### The code that you start with is

```
[AutoMethods(template = TemplateMethod.CustomTemplateFile, CustomTemplateName = "GenerateFromPOCO.txt")]

public partial class Person
{
    public string FirstName { get; set; }

    public string LastName { get; set; }
}
```

### The code that you will use is

```
var p = new Person();
p.FirstName = "Andrei";
p.LastName = "Ignat";

var last = p.ValueProperty(Person_EnumProps.LastName);
var first = p.ValueProperty("FirstName");

Console.WriteLine(last + " " + first);
```

### The code that is generated is

```
public enum Person_EnumProps{
    None
    ,FirstName // Public
    ,LastName // Public
}

partial class Person{
    public object ValueProperty(Person_EnumProps val){
        if(val == Person_EnumProps.FirstName) {
            return this.FirstName;
        }

        if(val == Person_EnumProps.LastName) {
            return this.LastName;
        }

        throw new ArgumentException("cannot find "+ val);
    }

    public object ValueProperty(string val){
        if(string.Compare("FirstName",val,StringComparison.CurrentCultureIgnoreCase)==0) {
            return this.FirstName;
        }

        if(string.Compare("LastName",val,StringComparison.CurrentCultureIgnoreCase)==0) {
            return this.LastName;
        }

        throw new ArgumentException("cannot find "+ val);
    }
}
```

Example Code: [https://github.com/ignatandrei/RSGC\\_Examples/tree/main/MetadataFromObject](https://github.com/ignatandrei/RSGC_Examples/tree/main/MetadataFromObject) ([https://github.com/ignatandrei/RSGC\\_Examples/tree/main/MetadataFromObject](https://github.com/ignatandrei/RSGC_Examples/tree/main/MetadataFromObject))

All Generators: [https://github.com/ignatandrei/RSGC\\_Examples/](https://github.com/ignatandrei/RSGC_Examples/) ([https://github.com/ignatandrei/RSGC\\_Examples/](https://github.com/ignatandrei/RSGC_Examples/))

## RSGC Name: MockSourceGenerator

Nuget :  
<https://www.nuget.org/packages/MockSourceGenerator/>  
link : <https://github.com/hemanussen/MockSourceGenerator/>  
author :Robin Hermanussen

### What can do

This will generate Mock classes directly for any interface - with your implementation.

### The code that you start with is

```
public interface IMatOps
{
    public int Add(int a, int b);

    public int Division(int a, int b);
}
```

### The code that you will use is

```
var mock = (IMatOps)new MatOpsMock
{
    MockAdd = (a, b) => a+b,
    MockDivision = (a,b)=> a/b
};
```

### The code that is generated is

```
public partial class MatOpsMock : global::MatOps.IMatOps
{
    /// <summary>
    /// Set this to true, if you want members that don't have a mock implementation
    /// to return a default value instead of throwing an exception.
    /// </summary>
    public bool ReturnDefaultIfNotMocked { get; set; }

    private System.Collections.Generic.List<HistoryEntry> historyEntries = new System.Collections.Generic.List<HistoryEntry>();
    public System.Collections.ObjectModel.ReadOnlyCollection<HistoryEntry> HistoryEntries
    {
        get
        {
            return historyEntries.AsReadOnly();
        }
    }

    /// <summary>
    /// Implemented for type global::MatOps.IMatOps (Public, same assembly: False)
    /// </summary>
    public Func<int,int,int>? MockAdd { get; set; }
    public int Add(int a, int b)
    {
        historyEntries.Add(new HistoryEntry("Add", new [] { $"{a}", $"{b}" }));

        if (MockAdd == null)
        {
            if (ReturnDefaultIfNotMocked)
            {
                return default(int);
            }
            else
            {
                throw new NotImplementedException("Method 'MockAdd' was called, but no mock implementation was provided");
            }
        }

        return MockAdd(a, b);
    }
}
```

```
/// <summary>
/// Implemented for type global::MatOps.IMatOps (Public, same assembly: False)
/// </summary>
public Func<int,int,int>? MockDivision { get; set; }

public int Division(int a, int b)
{
    historyEntries.Add(new HistoryEntry("Division", new [] { $"{a}", $"{b}" }));

    if (MockDivision == null)
    {
        if (ReturnDefaultIfNotMocked)
        {
            return default(int);
        }
        else
        {
            throw new NotImplementedException("Method 'MockDivision' was called, but no mock implementation was provided");
        }
    }

    return MockDivision(a, b);
}
}
```

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/DynamicMocking](https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/DynamicMocking](https://github.com/ignatandrei/RSCG_Examples/tree/main/DynamicMocking))

All Generators: [https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/) ([https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/))

### What can do

This will generate code to decorate methods with anything you want ( stopwatch, logging , authorization...)

The code that you start with is

The code that you will use is

The code that is generated is

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/MethodDecorator](https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/MethodDecorator](https://github.com/ignatandrei/RSCG_Examples/tree/main/MethodDecorator))

All Generators: [https://github.com/ignatandre/RSCG\\_Examples/](https://github.com/ignatandre/RSCG_Examples/) ([https://github.com/ignatandre/RSCG\\_Examples/](https://github.com/ignatandre/RSCG_Examples/))



## RSGC Name: PartiallyApplied

Nuget :  
<https://www.nuget.org/packages/PartiallyApplied/>  
link : <https://github.com/JasonBock/PartiallyApplied>  
author :Andrei Ignat

### What can do

This will generate curry for your functions

### The code that you start with is

```
public class Accounting
{
    public static float Discount( float discount, float price)
    {
        var val= price * (1- discount);
        return val;
    }
}
```

The code that you will use is

```
var disc10Percent = Partially.Apply(Accounting.Discount, 1/10f);
Console.WriteLine(disc10Percent(disc10Percent(100)));
```

The code that is generated is

```
public static partial class Partially
{
    public static Func<float, float> Apply(Func<float, float, float> method, float discount) =>
        new((price) => method(discount, price));
}
```

Example Code: [https://github.com/ignatandrei/RSCG\\_Examples/tree/main/PartiallyFunction](https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction) ([https://github.com/ignatandrei/RSCG\\_Examples/tree/main/PartiallyFunction](https://github.com/ignatandrei/RSCG_Examples/tree/main/PartiallyFunction))

All Generators: [https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/) ([https://github.com/ignatandrei/RSCG\\_Examples/](https://github.com/ignatandrei/RSCG_Examples/))