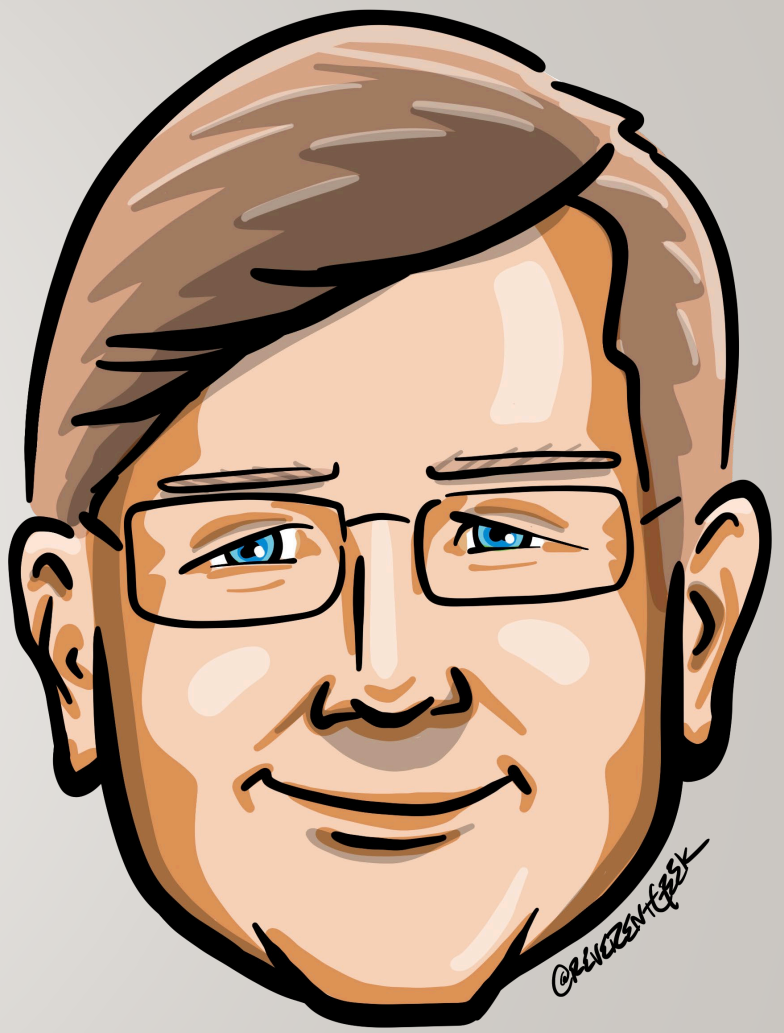

Building Great Libraries

Who is Chad Green

- chadgreen@chadgreen.com
- TaleLearnCode
- ChadGreen.com
- ChadGreen & TaleLearnCode
- ChadwickEGreen



Public Service Announcement


Listen to
your body

Listen to your body

Listen to your body

Preamble

Building Great Libraries

What is a class library?

collection of precompiled code modules, functions, classes, and resources

What is a class library?

designed to be reused and shared

What is a class library?

ready-made components

What is a class library?

leverage to expedite development process

What is a class library?

organized into namespaces

What is a class library?

promote code reuse and modular development

What is a class library?

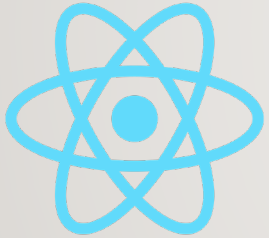
provided by the programming languages or created and distributed by third-parties

What is a class library?

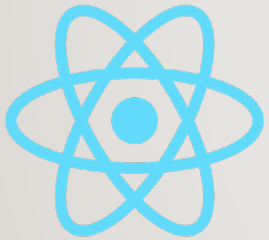
Reusable collection of prewritten code modules providing common functionalities to developers, enabling them to build software more efficiently and with less effort.

Examples of Class Libraries

Examples of Class Libraries



Examples of Class Libraries



Class Libraries in .NET

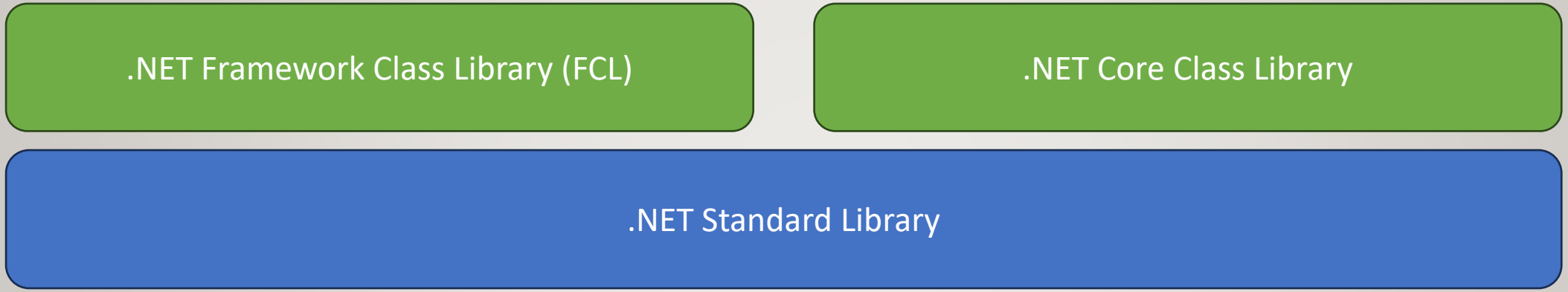
.NET Framework Class Library (FCL)

Class Libraries in .NET

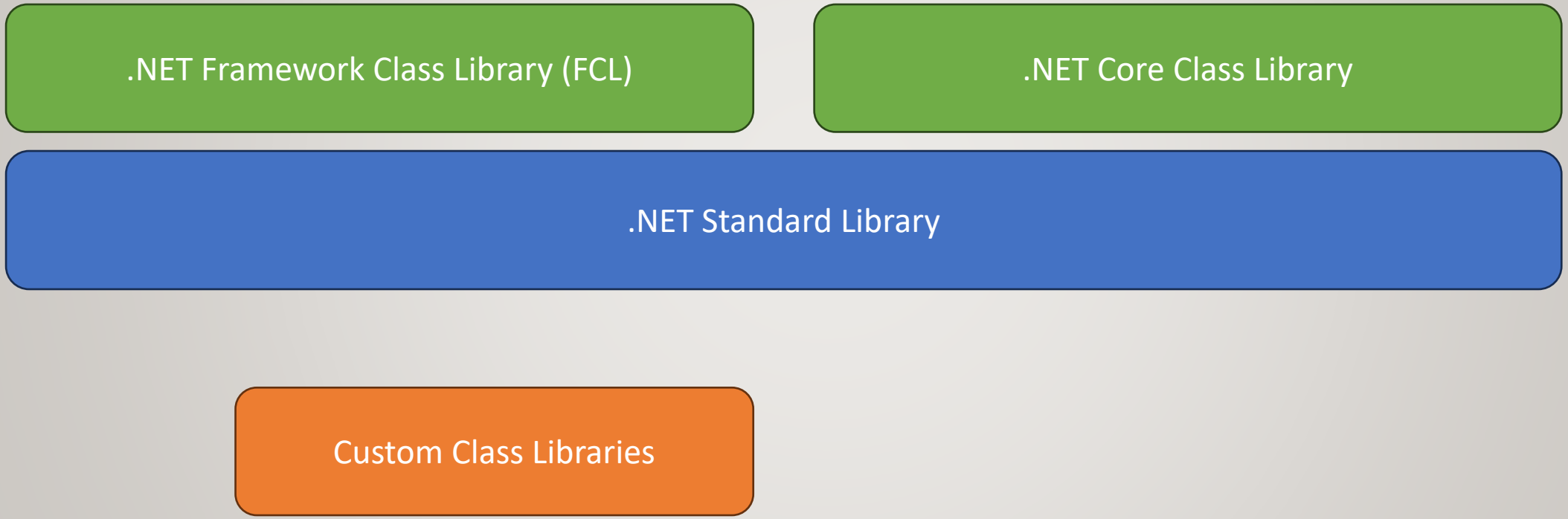
.NET Framework Class Library (FCL)

.NET Core Class Library

Class Libraries in .NET



Class Libraries in .NET



Class Libraries in .NET

.NET Framework Class Library (FCL)

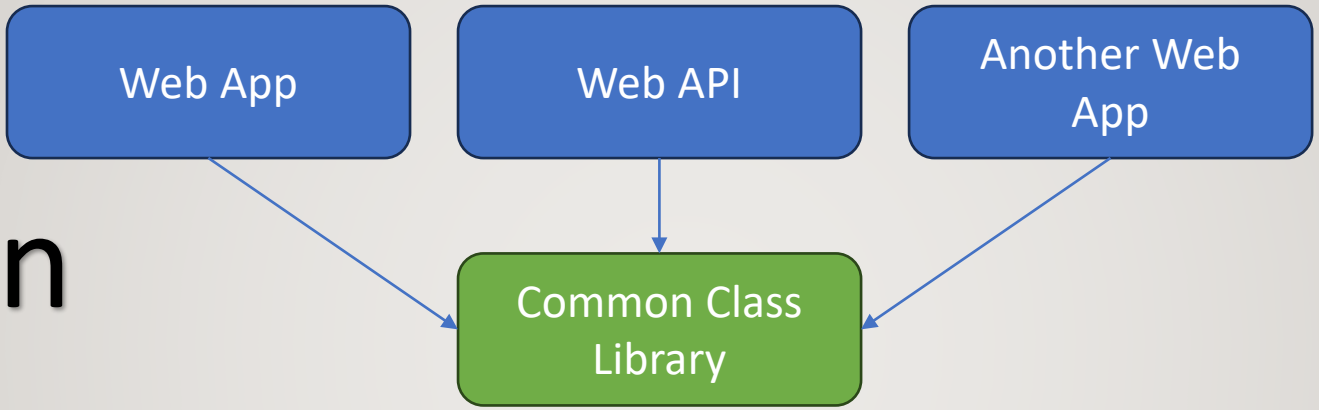
.NET Core Class Library

.NET Standard Library

Custom Class Libraries

Packages

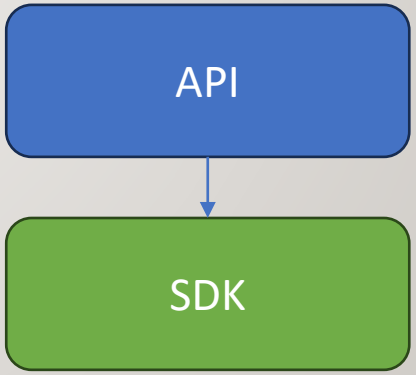
What are we talking about?



Common

Open Source Library

Open Source



APIs

Best Practices

Building Great Libraries

Best Practices

Define a clear and focused purpose

Best Practices

Design and reusability and modularity

Best Practices

Provide clear documentation

Best Practices

Follow naming conventions and guidelines

Best Practices

Implement error handling and exception management

Best Practices

Perform thorough testing

Best Practices

Consider performance and scalability

Best Practices

Versioning and backward compatibility

Best Practices

Use source control and package management

Best Practices

Solicit and incorporate user feedback

Best Practices

- Define a clear and focused purpose
- Design for reusability and modularity
- Provide clear documentation
- Follow naming conventions and guidelines
- Implement error handling and exception management
- Perform thorough testing
- Consider performance and scalability
- Versioning and backward compatibility
- User source control and package management
- Solicit and incorporate user feedback

Anti-Patterns

Building Great Libraries

Anti-Patterns

Bloated or Monolithic Libraries

Anti-Patterns

Tight Coupling and Lack of Abstraction

Anti-Patterns

Lack of Documentation

Anti-Patterns

Inconsistent Naming and Conventions

Anti-Patterns

Lack of Error Handling and Exception Management

Anti-Patterns

Neglecting Testing and Quality Assurance

Anti-Patterns

Ignoring Versioning and Backwards Compatibility

Anti-Patterns

Lack of Extensibility and Customization

Anti-Patterns

Insufficient Performance Optimization

Anti-Patterns

Failure to Consider Security

Anti-Patterns

- Bloated or Monolithic Libraries
- Tight Coupling and Lack of Abstraction
- Lack of Documentation
- Inconsistent Naming and Conventions
- Lack of Error Handling and Exception Management
- Neglecting Testing and Quality Assurance
- Ignoring Versioning and Backward Compatibility
- Lack of Extensibility and Customization
- Insufficient Performance Optimization
- Failure to Consider Security

Additional Information

Building Great Libraries

Additional Information

Coding Standards

Additional Information

Coding Standards

Common C# Coding Conventions

Additional Information

Coding Standards

Design Guidelines for Developing Class Libraries

Additional Information

Documentation

Project Readme

Documentation Website

Package Readme

XML Documentation

Additional Information

Documentation

Project Readme

Documentation Website

Package Readme

XML Documentation

Additional Information

Documentation

XML Documentation

```
/// <summary>
/// Returns the sum of <paramref name="addend1"/> and <paramref name="addend2"/>.
/// </summary>
/// <param name="addend1">The first number to be added.</param>
/// <param name="addend2">The second number to be added.</param>
/// <returns>The sum of <paramref name="addend1"/> and <paramref name="addend2"/>.</returns>
1 reference | Chad Green, 1 day ago | 1 author, 1 change
public double Add(double num1, double num2)
{
    return num1 + num2;
}
```

Additional Information

Documentation

XML Documentation

- General tags
  - summary
  - remarks
- Document members
  - returns
  - param
  - paramref
  - exception
  - value
- Format document output
  - para
  - list
  - c
  - code
  - example
- Reuse documentation text
  - inheritdoc
  - include
- Generate links and references
  - see
  - seealso
- Generic types and methods
  - typeparam
- User-defined tags

Additional Information

Versioning

Semantic Versioning

7.20.2023

Major.Minor.Patch

Additional Information

Publishing

7.20.2023-preview

1.0.123-dev

Additional Information

Breaking Changes

Additional Information

Breaking Changes

Virtual Methods

Additional Information

Breaking Changes

Method Signatures

Additional Information

Breaking Changes

Obsolete Attribute

Additional Information

Breaking Changes

Optional Arguments

Coding Demonstration

Building Great Libraries

Coding Demonstration

```
namespace TaleLearnCode.MathOperationsLibrary
{
    1 reference | Chad Green, 19 minutes ago | 1 author, 1 change
    public interface IMathOperations
    {

        /// <summary>
        /// Returns the sum of <paramref name="addend1"/> and <paramref name="addend2"/>.
        /// </summary>
        /// <param name="addend1">The first number to be added.</param>
        /// <param name="addend2">The second number to be added.</param>
        /// <returns>The sum of <paramref name="addend1"/> and <paramref name="addend2"/>.</returns>
        1 reference | Chad Green, 19 minutes ago | 1 author, 1 change
        double Add(double addend1, double addend2);

        /// <summary>
        /// Returns the results of removing <paramref name="subtrahend"/> from the <paramref name="minuend"/>.
        /// </summary>
        /// <param name="minuend">: a number from which the subtrahend is to be subtracted</param>
        /// <param name="subtrahend">The number that is to be subtracted from a minuend.</param>
        /// <returns>The results of removing the <paramref name="subtrahend"/> from the <paramref name="minuend"/>.</returns>
        1 reference | Chad Green, 19 minutes ago | 1 author, 1 change
        double Subtract(double minuend, double subtrahend);

        /// <summary>
        /// Returns the product of multiplying the <paramref name="multiplicand"/> by the <paramref name="multiplier"/>.
        /// </summary>
        /// <param name="multiplicand">The number that is to be multiplied by the <paramref name="multiplier"/>.</param>
        /// <param name="multiplier">The number by which the <paramref name="multiplicand"/> is multiplied</param>
        /// <returns>The product of multiplying the <paramref name="multiplicand"/> by the <paramref name="multiplier"/>.</returns>
        1 reference | Chad Green, 19 minutes ago | 1 author, 1 change
        double Multiply(double multiplicand, double multiplier);

        /// <summary>
        /// Returns the calculation of the number of times one number (<paramref name="dividend"/>) is contained within another (<paramref name="divisor"/>).
        /// </summary>
        /// <param name="dividend">a number to be divided</param>
        /// <param name="divisor">the number by which a dividend is divided</param>
        /// <returns></returns>
        1 reference | Chad Green, 19 minutes ago | 1 author, 1 change
        double Divide(double dividend, double divisor);
    }
}
```

Coding Demonstration

```
1   using System;
2
3   namespace TaleLearnCode.MathOperationsLibrary
4   {
5
        4 references | Chad Green, 20 minutes ago | 1 author, 1 change
6       public class MathOperationsException : Exception
7       {
            0 references | Chad Green, 20 minutes ago | 1 author, 1 change
8           public MathOperationsException() { }
9
            1 reference | Chad Green, 20 minutes ago | 1 author, 1 change
10          public MathOperationsException(string message) : base(message) { }
11
            0 references | Chad Green, 20 minutes ago | 1 author, 1 change
12          public MathOperationsException(string message, Exception innerException)
13              : base(message, innerException) { }
14      }
15  }
```

Coding Demonstration

```
1   namespace TaleLearnCode.MathOperationsLibrary
2   {
3
                0 references | Chad Green, 21 minutes ago | 1 author, 1 change
4       public class MathOperations : IMathOperations
5       {
6
7           /// <summary> Returns the sum of addend1 and addend2.
                1 reference | Chad Green, 21 minutes ago | 1 author, 1 change
13          public double Add(double num1, double num2)
14          {
15              return num1 + num2;
16          }
17
18          /// <summary> Returns the results of removing subtrahend from the minuend.
                1 reference | Chad Green, 21 minutes ago | 1 author, 1 change
24          public double Subtract(double num1, double num2)
25          {
26              return num1 - num2;
27          }
28
29          /// <summary> Returns the product of multiplying the multiplicand by the multipl ...
                1 reference | Chad Green, 21 minutes ago | 1 author, 1 change
35          public double Multiply(double num1, double num2)
36          {
37              return num1 * num2;
38          }
39
40          /// <summary> Returns the calculation of the number of times one number ( divide ...
                1 reference | Chad Green, 21 minutes ago | 1 author, 1 change
46          public double Divide(double num1, double num2)
47          {
48              if (num2 == 0)
49              {
50                  //throw new DivideByZeroException("Cannot divide by zero
51                  throw new MathOperationsException("Cannot divide by zero");
52              }
53              return num1 / num2;
54          }
55
56      }
57
58  }
```

Coding Demonstration

readme.md – MathOperationsLibrary × | README.md – BuildingGreatLibraries ×

```
Returns the sum of the supplied addends.

### Subtract
Returns the results of removing the subtrahend from the minuend.

### Multiply
Returns the product of multiplying the multiplicand by the multiplier.

### Division
Returns the calculation of the number of times one number (dividend) is
contained within another (divisor).
```

TaleLearnCode Math Operations Library

This is a fully operational library providing simple mathematical operations. The intent of this library is to demonstrate how to build great libraries and is a part of the **Build Great Libraries** presentation presented by Chad Green.

Operations

Add

Returns the sum of the supplied addends.

Subtract

Returns the results of removing the subtrahend from the minuend.

Multiply

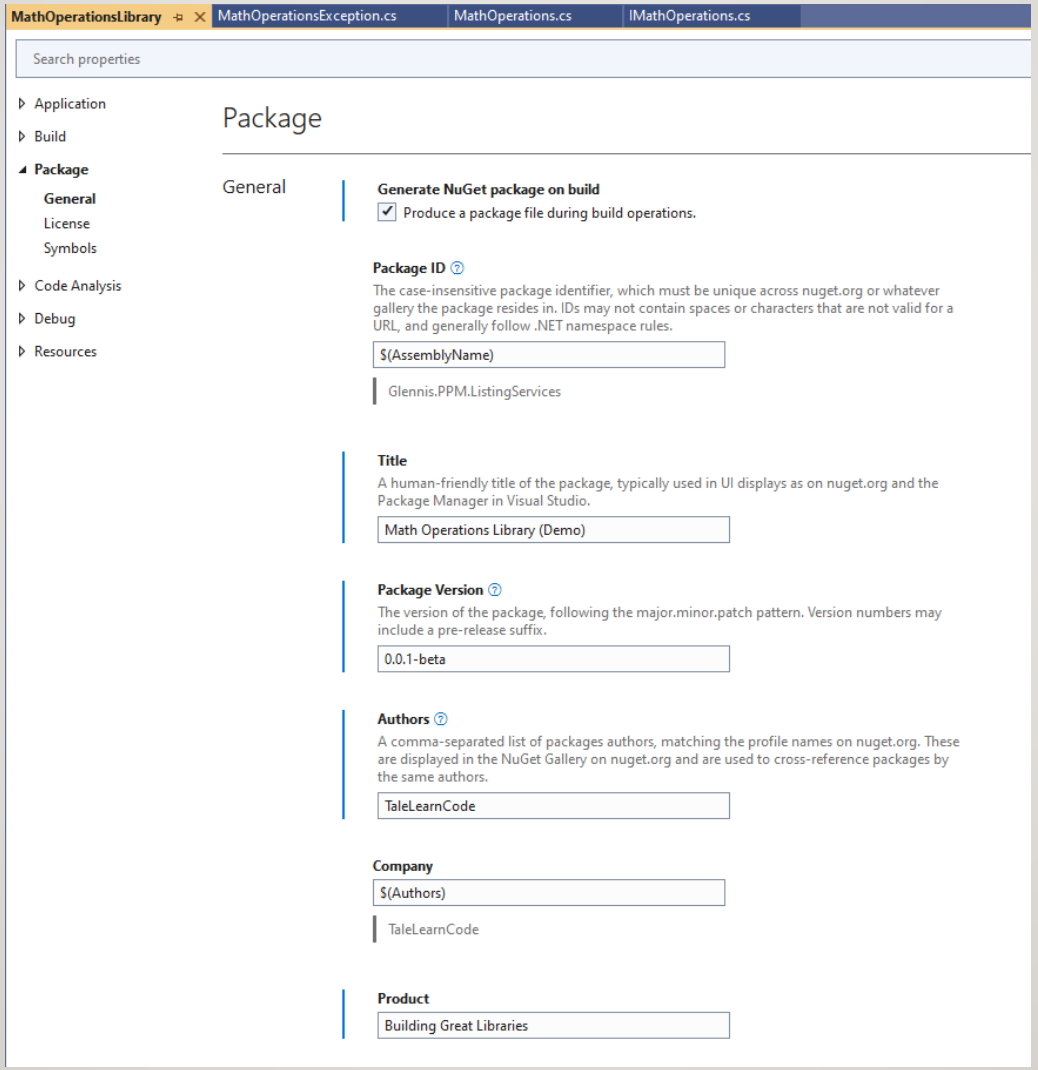
Returns the product of multiplying the multiplicand by the multiplier.

Division

Returns the calculation of the number of times one number (dividend) is contained within another (divisor).

created with the free version of **Markdown Monster**

Coding Demonstration



Coding Demonstration

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <TargetFramework>netstandard2.0</TargetFramework>
    <RootNamespace>TaleLearnCode.MathOperationsLibrary</RootNamespace>
    <AssemblyName>Glennis.PPM.ListingServices</AssemblyName>
    <Version>0.0.1-beta</Version>
    <Authors>TaleLearnCode</Authors>
    <Product>Building Great Libraries</Product>
    <Description>Provides a set of basic mathematic operation methods.  This is for presentation purposes only.</Description>
    <Copyright>© 2023 Green Events and Technology.  All rights reserved.</Copyright>
    <RepositoryUrl>https://github.com/TaleLearnCode/BuildingGreatLibraries</RepositoryUrl>
    <PackageProjectUrl>https://www.chadgreen.com/presentations/building-great-libraries</PackageProjectUrl>
    <Title>Math Operations Library (Demo)</Title>
    <RepositoryType>git</RepositoryType>
    <PackageTags>math;demo;presentation</PackageTags>
    <PackageReleaseNotes>Producing the initial demo release of the TaleLearnCode Math Operations Library.</PackageReleaseNotes>
    <NeutralLanguage>en-US</NeutralLanguage>
    <GeneratePackageOnBuild>True</GeneratePackageOnBuild>
    <PackageReadmeFile>readme.md</PackageReadmeFile>
    <PackageLicenseExpression>MIT</PackageLicenseExpression>
  </PropertyGroup>

  <ItemGroup>
    <None Update="readme.md">
      <Pack>True</Pack>
      <PackagePath>\</PackagePath>
      <CopyToOutputDirectory>PreserveNewest</CopyToOutputDirectory>
    </None>
  </ItemGroup>

</Project>

```

Thank You

 chadgreen@chadgreen.com

 TaleLearnCode

 ChadGreen.com

 ChadGreen & TaleLearnCode

 ChadwickEGreen